UNSUPERVISED DISCOVERY AND REPRESENTATION OF SUBSPACE TRENDS IN MASSIVE BIOMEDICAL DATASETS

A Dissertation

Presented to

the Faculty of the Department of Electrical and Computer Engineering

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

in Electrical Engineering

by

Yan Xu

December 2015

UNSUPERVISED DISCOVERY AND REPRESENTATION OF SUBSPACE TRENDS IN MASSIVE BIOMEDICAL DATASETS

_____

Yan Xu

Approved:

_____

Chair of the Committee
Dr. Badrinath Roysam,
Professor and Department Chair,
Electrical and Computer Engineering

Committee Members:

_____

Dr. David Mayerich,
Assistant Professor,
Electrical and Computer Engineering

_____

Dr. Jared Burks, Assistant Professor,
MD Anderson Cancer Center

_____

Dr. Peng Qiu, Assistant Professor,
Georgia Tech and Emory University

_____

Dr. Saurabh Prasad,
Assistant Professor,
Electrical and Computer Engineering

_____

Dr. Zhu Han, Professor,
Electrical and Computer Engineering

_____

Dr. Suresh K. Khator,
Associate Dean,
Cullen College of Engineering

_____

Dr. Badrinath Roysam,
Professor and Department Chair,
Electrical and Computer Engineering

Acknowledgements

I would love to thank my advisor, Dr. Badri Roysam, for advising me during my PhD study. During the four and a half years at the bio-analytics laboratory, Dr. Roysam has helped me a lot with his invaluable advice and inspiring insight on scientific research as well as his professional attitude to the academic work. I have also learned great writing and presentation skills from him as an efficient researcher to communicate the research to a broad audience and really make an impact in the field.

I have my sincere gratitude to Dr. Peng Qiu for his help and guidance during my research work since I started. His immediate feedbacks and insightful questions have greatly inspired my dissertation.

I would love to thank all my committee members, Dr. David Mayerich, Dr. Jared Burks, Dr. Saurabh Prasad, and Dr. Zhu Han for agreeing to serve as my dissertation committee and spending their valuable time reviewing my dissertation and slides. I would love to thank Dr. Shain and Dr. Ascoli for their great help in interpreting the results from the biological perspective. I also want to thank all my lab mates for their efficient hard work and all the great time we had together. Working with them is my great pleasure.

Besides, I would love to thank my family members, especially my mother, Rennan Li for her support during my PhD study. Their support and understanding encourage me to move forward during my study to fulfill my academic goals. I would also love to thank my special someone, Xuan Qin, who will always stand by my side to support me and encourage me when I face difficulties and I really enjoy the wonderful time we spent together.

Finally, I would love to thank DARPA and NSF to support this work.

UNSUPERVISED DISCOVERY AND REPRESENTATION OF SUBSPACE TRENDS IN MASSIVE BIOMEDICAL DATASETS

A Dissertation

Presented to

the Faculty of the Department of Electrical and Computer Engineering

University of Houston

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

in Electrical Engineering

by

Yan Xu

December 2015

Abstract

The goal of this dissertation is to develop unsupervised algorithms for discovering previously unknown subspace trends in massive multivariate biomedical data sets without the benefit of prior information. A subspace trend is a sustained pattern of gradual/progressive changes within an unknown subset of feature dimensions. A fundamental challenge to subspace trend discovery is the presence of irrelevant data dimensions, noise, outliers, and confusion from multiple subspace trends driven by independent factors that are mixed in with each other. These factors can obscure the trends in traditional dimension reduction and projection based data visualizations. To overcome these limitations, we propose a novel graph-theoretic neighborhood similarity measure for sensing concordant progressive changes across data dimensions. Using this measure, we present an unsupervised algorithm for trend-relevant feature selection and visualization. Additionally, we propose to use an efficient online density-based representation to make the algorithm scalable for massive datasets.

The representation not only assists in trend discovery, but also in cluster detection including rare populations. Our method has been successfully applied to diverse synthetic and real-world biomedical datasets, such as gene expression microarray and arbor morphology of neurons and microglia in brain tissue. Derived representations revealed biologically meaningful hidden subspace trend(s) that were obscured by irrelevant features and noise. Although our applications are mostly from the biomedical domain, the proposed algorithm is broadly applicable to exploratory analysis of high-dimensional data including visualization, hypothesis generation, knowledge discovery, and prediction in diverse other applications.

Table of Contents

List of Figures

List of Tables

1. Introduction

In real world dataset, we often find trends, characterized by gradual variation of variables along with a common underlying factor such as time. Especially in biological systems, there usually exhibits gradual progression when transitioning between states, rather than quantum leaps of state changes. For example, from gene expressions measured by microarray, we can find gradual transition from one state to another in the cell cycle process. We can infer about hematopoietic cell differentiation from flow cytometry and understand the morphological development of neurons from microscope. We are not only interested in how many cell states there are in these biomedical datasets, but also interested in the trend, representing the gradual transition among states. Unlike one variable trend, which can be easily detected by simple visualization, these biomedical datasets are multivariate. They either have very high dimensionality with respect to the number of available samples or can easily reach millions of samples with a high-throughput machine. A direct visualization of these datasets sometimes cannot reveal an interesting pattern due to curse of high dimensionality and noise. Therefore, it is necessary and essential to identify relevant dimensions first to derive meaningful patterns. With the goal of trend discovery in mind, we aim to identify "trend-relevant" features and derive efficient representation for the data points within the subspace of the identified features, i.e., subspace trend. Finding subspace trends is valuable since they can provide clues to the underlying progression and mechanisms. We will introduce the basic concepts of subspace trend and the motivating example in the following paragraphs.

Given a set of N multivariate data points$\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \dots, \mathbf{x}^{(N)}\}$ in an M-dimensional space, $\mathbf{x}^{(i)} \in \mathbf{R}^M$, the problem of interest is to identify subspace trends in an unsupervised

manner, without using prior information of the process that gave rise to the data. A trend is a special type of pattern characterized by sustained gradual/progressive changes among the data points. The progressive changes are often (though not necessarily) guided by an unknown underlying driving parameter (e.g., time, disease progression, developmental stage, etc.). A subspace trend occurs within a subspace, i.e., the progressive changes occur within an unknown subset of the dimensions. Geometrically, subspace trends can be thought of as a set of smooth curves (1-manifolds) in a multivariate subspace within $\mathbf{R}^M$. The curves may be nonlinear, and may fork/intersect. In achieving this goal, we are interested in algorithms that translate to practically usable tools for trend discovery. With usability in mind, we are interested in algorithms that are efficient, scalable, and requiring the fewest-possible adjustable parameter settings.

Some trends can become visually apparent when a suitable multivariate data visualization method is used. Specifically, the high-dimensional data can be projected onto a low-dimensional (typically 2D or 3D) space for visualization. Several dimension reduction or embedding methods have been proposed for exploratory visualization, including Locally Linear Embedding (LLE) [1], ISOMAP [2], Laplacian Eigenmap [3], t-distributed stochastic neighbor embedding (t-SNE) [4], and the diffusion map [5]. Although these methods have proven valuable in many applications, they can fail to reveal trends in the presence of a large number of irrelevant dimensions that can obscure the manifold(s) of interest that are hidden in an unknown subspace. Therefore it is necessary and important to identify the trend-relevant dimensions first, and then visualize the manifold using only those relevant dimensions. Additional challenges include noise, outliers, and the presence of multiple independent trends driven by independent factors.

Figure 1 shows a motivating example illustrating the effect of irrelevant dimensions. Plotted in this figure is a microarray dataset from a published study [6] representing 7,288 microarray gene expression measurements taken across 99 breast tumor samples. The microarray data were also accompanied by records of the patients' estrogen receptor (ER) status (ER positive/negative) [7]. Panels (a – c) of Figure 1 show the visualization results produced by LLE, ISOMAP and t-SNE, respectively, when projecting the full multivariate data onto three dimensions (3D). In these panels, each dot corresponds to a patient, and its color indicates the ER status, with red indicating ER-positive, and blue indicating ER-negative.

With this in mind, we ask if there is a progressive trend from ER negative to positive status among the tumor samples. Panels (a – c) of Figure 1 do not reveal a clear trend that correlated with the ER status (note: changing the viewing angle does not help). However, panels (d – f) show the same data visualized using the same projection methods, but with only 48 relevant genes selected by a t-test that utilized the ER status data [6]. In this rendering, ER negative and positive samples are better separated, and there are clear indications of a gradual transition between them, rather than two distinct clusters. This trend was hidden within the subspace defined by these 48 relevant features, and the trend-irrelevant features obscured it in the initial visualization in Panels (a – c). From this motivated example, we can see that without feature selection, the two ER status samples are intermixed with each other. With selection, the trend of gradual transition from ER negative to positive is revealed.

We now pose the following practical question: could these 48 trend-relevant features be identified without the benefit of the class labels (ER status)? This goal is quite

distinct from the widely studied problem of selecting features for classification, where training class labels are available for evaluating feature subsets. In other words, we are interested in trend-relevant feature selection, rather than classification-relevant feature selection. Interestingly, as with classification, removal of irrelevant dimensions not only helps uncover hidden subspace trends, but also "strengthens" the discovered trends, and makes the data visualization more efficient and effective.

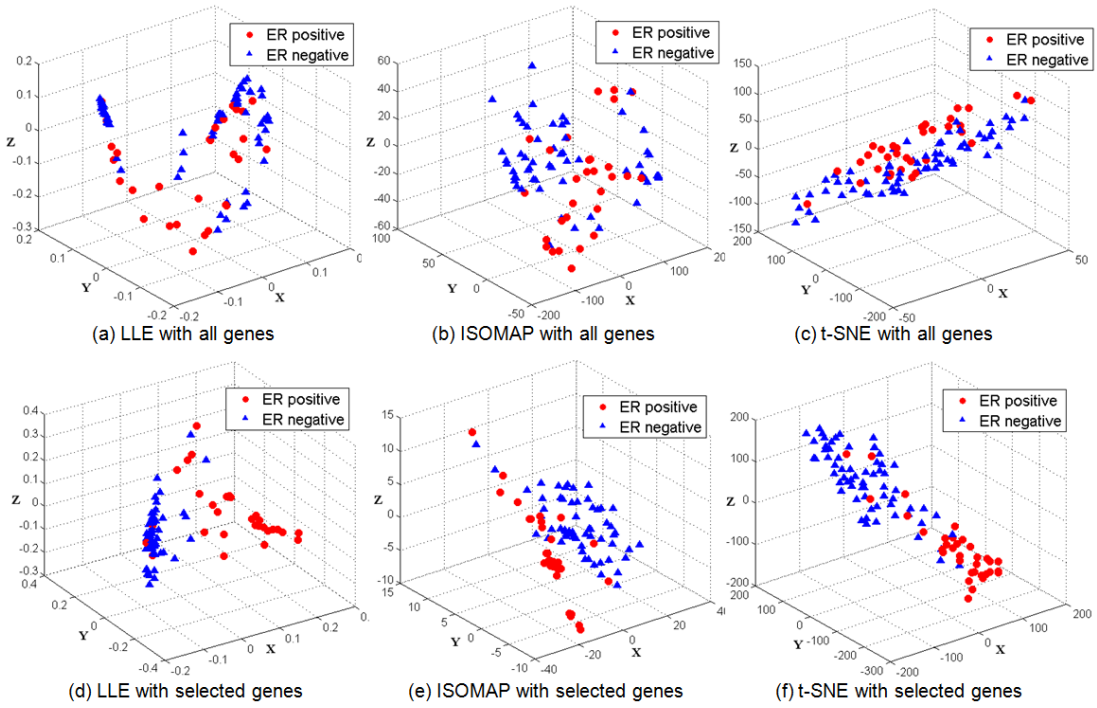

Figure 1. Illustrating the impact of feature selection on subspace trend discovery. cDNA microarray data measuring the expression of 7,288 genes in 99 breast tumor samples were visualized using three widely used methods LLE, ISOMAP, and t-SNE, respectively.

In considering the trend-relevant feature selection problem, we find that the majority of feature selection methods were designed for classification rather than trend discovery. Guyon & Eliseef [8] organized these methods into three broad categories: filter, wrapper, and embedded methods, respectively, based on how they are combined with the class models. Graph embedding [9] has been proposed as a way of unifying several dimension reduction algorithms within a common framework. Overall, feature

selection methods for classification seek to organize the data into well-separated groups, and identify the best decision boundaries separating these groups. On the other hand, unsupervised selection of trend-relevant features must be guided by different considerations, such as detecting sustained patterns of progressive changes, and detecting hidden relationships between data dimensions that specifically affect subspace trends.

In this work, we propose an algorithm to identify trend-relevant feature dimensions in an unsupervised manner, by analyzing the associations among features within a graph-theoretic framework, specifically: (i) effective methods to analyze relationships among features for the purpose of trend discovery; and (ii) methods to identify the features that are relevant to a common underlying subspace trend. (iii) online density-based representation for trends in massive datasets. To demonstrate its efficacy in discovering subspace trends, we evaluated our method on diverse synthetic and real-world datasets. We validated our algorithm on datasets for which auxiliary data are available for validation of the detected trends, for example, cell-cycle microarray data and simulated neuron arbor morphology data. We show that our algorithm can discover trends without utilizing the auxiliary data. We show that removing trend-irrelevant dimensions not only uncovers hidden subspace trends, but also strengthens/clarifies the discovered trends, and makes the data visualization more efficient. We also show that the density-based representation is efficient to derive visualization for the gradual changes of selected features. Furthermore, for the first time in literature, we have applied our algorithm for trend exploratory discovery in 3-D arbor morphology of neurons and microglia in the brain tissue.

2. Prior Literature

The prior literature has several related areas: subspace clustering, temporal ordering recovery, data representation for massive datasets and arbor analytics, which will be introduced in the following sections respectively.

2.1. Subspace Clustering

Subspace clustering and biclustering [10] are related topics in computer vision and data mining applications. In computer vision [11], algorithms like LRR (low-rank representation) and SSC (sparse subspace clustering) [12][13] have been proposed to segment high-dimensional data into clusters with each corresponding to a subspace structure. In data mining, subspace clustering algorithms have been described to search for subspaces that accommodate clusters. They usually restrict the search space to axis-parallel subspaces and make use of the downward closure property, usually based on a global density threshold, or by making locality assumptions to enable efficient search heuristics. CLIQUE [14], ENCLUS [15], and the Cell-based Clustering Method (CBF) [16] use a bottom-up search strategy based on merging dense low-dimensional units to form clusters. While these methods rely on a global density threshold, adaptive density thresholding in different subspace cardinalities to discover clusters has been proposed in [17]. Methods like PROCLUS [18], FINDIT [19], and Halite [20] use a top-down strategy based on iteratively generating clusters in the updated subspace.

Biclustering restricts the search space to special forms or locations to search for mainly four different categories of sub-matrices (biclusters): constant values, constant values on either columns or rows, coherent values, and coherent evolutions [21]. Most biclustering algorithms are restricted to simple positive correlations among features. A

more general approach, known as oriented clustering or generalized subspace clustering, attempts to find clusters in arbitrarily oriented subspaces, and nonlinearly correlated clusters, for example ERiC [22], CASH [23] based on the Hough Transform and model-based projective clustering [24]. Although subspace clustering and biclustering algorithms are useful for grouping data points in subspaces, they are not useful for subspace trend discovery where we are interested in progressive changes of latent patterns in subspaces, especially when the selection of relevant dimension becomes essential for small sample sizes in high dimensions, and the ordering relationship of samples is more important than clusters.

2.2. Temporal Ordering Recovery

Estimation of temporal orderings from unordered sets of samples is also related. A Traveling Salesman Path (TSP) based model is used to infer the underlying temporal order of microarray experiments in [25]. Diameter path statistics and PQ-tree are used to estimate and represent uncertainty in the reconstructed ordering of unordered sets of sample elements in [26]. These methods assume that the underlying trend is a path with no branching points. There are also methods that infer more complex structures. By imposing a tree structure on a set of tumors as a function of their gene expression levels, a classification tree is built for class discovery and class prediction problems in [27]. Cancer phylogenies and progression pathways from microarray data are inferred using Bayesian networks and regression models in [28]. Although these methods have proven useful in uncovering the relationships among samples, they are based on carefully designed or preselected features. More recently, pseudo-temporal ordering [29] have been proposed to recover single-cell gene expression kinetics from a wide array of cellular

processes including differentiation and proliferation. Diffusion maps have been used to define differentiation trajectories [30]. Although these algorithms are useful for recovering cell ordering, they heavily relied on the preselected features by the users. Our algorithm tackles this challenge by proposing an unsupervised feature selection specifically for trend discovery, i.e., sample ordering.

Several aspects of the proposed subspace trend discovery have been inspired by recently published algorithms with feature selection ability. A Minimum Spanning Tree (MST) was built for seeking cluster centroids in the Information-Rich Subsets (IRS) [31] method, in which a collection of subspace clusters are derived from biclustering. Trends are presented as a set of optimal MST sub-paths that cover all the cluster centroids. The drawback is that it cannot work when samples are too few to form clusters. Another paper that inspired us is the sample progression discovery (SPD) algorithm [32]. It is designed to discover branched acyclic progressions and the associated features in microarray data. Based on the statistical concordance between all the modules of linearly correlated features and the MSTs constructed from all modules, SPD is able to identify the feature modules that support a common progression pattern aided by human visual inspection. The proposed subspace trend discovery algorithm improves upon these methods in several ways. It is much more efficient since it does not require computation of random permutations to derive the statistical concordance between feature modules. Its output is deterministic for a given set of parameter settings and it can be computed in a fully automatic manner without user intervention.

2.3. Representation for Massive Data

Many algorithms have been published recently to identify cell subpopulations in the massive datasets, especially for flow cytometry dataset. There are visualization based methods such as SPADE [33] and viSNE [34]. Both methods require cluster identification from multiple colored feature maps. SPADE uses a force-directed graph layout algorithm and viSNE uses a dimension reduction algorithm to project high-dimensional data points into two dimensional space. While both methods can detect both subpopulations and their relationship information in some way, they require a pre-sampling to reduce the computational complexity. Wanderlust [35] uses ensemble graphs to identify a main path to model the biological process; however it does not allow branches and requires an initializing point from the user. Mixture models, such as Gaussian finite mixture and non-parametric Bayesian model, have also been used for cluster detection in flow cytometry data. FLAME [36] uses finite mixture of skew-t distribution models considering the presence of skew in the data and proposes a way to select the optimal number of skew-t distributions. However it requires classical expectation maximization (EM) to train the model, which is not efficient for large datasets. Furthermore, it requires user input to identify interesting population before digging into rare populations. Gaussian mixture models with weighted iterative sampling [38] and non-parametric Bayesian modeling [37] have also been proposed. However both methods suffers from heavy parameter tuning, especially non-parametric Bayesian modeling, which has seven free model parameters for training. All these methods require some sampling strategy for massive dataset as a preprocessing step and do not allow an online model update without retraining the model. We propose an online density

representation for massive flow cytometry data based on an online kernel density estimation algorithm [39]. It uses adaptive Gaussian mixtures to approximate Gaussian or non-Gaussian density distribution of data. Based on the derived Gaussian mixtures, we derive a heatmap representation to automatically identify cell subpopulations and gradual variation of features. It is by far the first online algorithm to deal with massive biomedical data.

2.4 Arbor Analytics

Quantitative analysis of whole brain-cell arbors is a long sought goal, and an essential component of the quest to understand structure-function relationships in normal brain function, as well as responses to injury or disease. While statistical analysis of individual (typically, user-selected) arbor features is not uncommon [40][41], unsupervised analysis of standardized collections/libraries of arbor features is an emerging need aimed at data-driven biological discovery, especially screening and large-scale mapping studies. For example, neurons have been statistically analyzed in past studies based on manually chosen morphometric parameters extracted from neuronal images [41]. Fractal analysis has been applied to quantify dendritic morphology with a modified box-counting method [42]; and neural networks have been trained on quantified histo-morphological properties to classify neurons [43]. While these analyses require the use of cell-specific labels, large-scale and unsupervised arbor analysis is a newly emerged need made possible by the convergence of advances in large-scale automated reconstruction, widespread adoption of standard file formats for storing arbor reconstructions [44], the advent of large neuro-morphological databases [45], and the development of standardized general-purpose libraries of arbor features, notably

Scorcioni's L-measure [46]. Based on this unprecedented data availability, Lu described the application of unsupervised harmonic co-clustering algorithms [47] based on the diffusion distance [48] for analyzing L-measure data for neurons drawn from the NeuroMorpho database (www.neuromorpho.org). This method simultaneously identifies the hierarchical grouping patterns among the cells and their L-measure features, and is therefore a significant advance.

An important advance of the present work is the development of a precise, cellular-resolution method for determining microglial status. The prior literature on microglia arbor morphology has focused heavily on normal (unperturbed) brain tissues, and on acute (localized) perturbations [49]. For example, Wu reported multiple gradients of differentiation stages of microglia in the developing cerebral cortex of rat brains [50]. Soltys described a fractal analysis based method for quantifying the morphology of reactive microglia in the injured cerebral cortex [51]. Jonas described the morphological stages of microglial activation and deactivation in detail, and termed this the "spider effect" based on an analogy with the behaviors of spiders [40]. Our work extends the prior work by providing a detailed three-dimensional data-driven characterization of microglial activation at a much larger spatial scale (multiple millimeters, much larger than the cell size), and over much larger cell populations (tens of thousands of cells). The advent of technologies for mosaiced multi-millimeter extent brain slice imaging enable the study of microglial activation patterns across such extended perturbed and injured tissues. A particular need in this area is the ability to compare perturbed tissues against normal/unperturbed tissues in a quantitative manner, with due attention to the activation states of microglia as revealed by their arbor morphologies. We propose a direct solution

that adapts to biological and imaging variability. In applying our approach, we only require that the tissues being compared are subjected to the same specimen preparation and imaging protocols, a reasonable requirement. Our proposed approach provides precise methods of measuring microglia responses to implanted neuroprosthetic devices. Considerable investigational interest revolves around the device-tissue interactions that result in a loss of device electrical recording performance [52][53]. Until recently, tissue responses have been studied qualitatively with changes observed in the relative abundance or general morphology of cells [54][55]. More recently several groups of quantified these responses by measuring changes in fluorescence signals as a function of distance along a projected line away from the implant site [56]. While these methods provide measurements of antibody binding, they are only indirect measures of changes in cell numbers or protein expression. The results we report here provide the first comprehensive, cell-resolution measurements of response-associated gradual changes in cell phenotypes, numbers, and distributions.

3. Method

3.1. Subspace Trend Discovery

3.1.1. Background propositions

We are given N data points in an M-dimensional space, denoted $\mathbf{x}^{(i)} = [x_1^{(i)}, x_2^{(i)}, \dots, x_M^{(i)}]^T \in \mathbf{R}^M, 1 \le i \le N$. As a feature-centric notation, we denote the data for each dimension/feature as $\mathbf{x}_\mathrm{j} = [x_j^{(1)}, x_j^{(2)}, \dots, x_j^{(N)}]^T \in \mathbf{R}^N, 1 \le j \le M$. Denote the set of trend-relevant features **J**, so the dimension of the corresponding subspace is $p = |\mathbf{J}|$, the cardinality of **J**. For simplicity, we refer to "the subspace indicated by **J**" simply as "subspace **J**" in the following text. As noted earlier, a subspace trend can be considered as an unknown smooth curve (1-manifold) in a multivariate subspace within $\mathbf{R}^M$. Let $\mathbf{g_J}(\lambda^{(i)}) = [g_{j_1}(\lambda^{(i)}), g_{j_2}(\lambda^{(i)}), \cdots, g_{j_p}(\lambda^{(i)})]^\mathrm{T}$ denote a vector of p continuous non-constant parametric curves that are fitted to the data points $\mathbf{x}_\mathbf{J}^{(\mathrm{i})} = [x_{j_1}^{(i)}, x_{j_2}^{(i)}, \dots, x_{j_p}^{(i)}]$ $(1 \le i \le N)$, in the p -dimensional subspace, with the parameter λ scaled to the interval [0, 1]. The curve parameter values that are proximal to the data points are given by $\lambda^{(i)} = \inf_u ||\mathbf{x}_\mathbf{J}^{(\mathrm{i})} - \mathbf{g_J}(u)||_2$. Alternatively, this can be written as $\mathbf{x}_\mathbf{J}^{(\mathrm{i})} = \mathbf{g_J}(\lambda^{(i)}) + \boldsymbol{\varepsilon}_\mathbf{J}^{(i)}$, where $\boldsymbol{\varepsilon}_\mathbf{J}^{(i)}$ is an error vector. If $\mathrm{E}[\boldsymbol{\varepsilon}_\mathbf{J}^{(i)} | \mathbf{x_J}, \mathbf{g_J}] \le \boldsymbol{\epsilon}$, where $\boldsymbol{\epsilon}$ is a small vector close to 0, we claim that a trend curve is present near $\mathbf{x}_\mathbf{J}^{(\mathrm{i})}$ in subspace **J,** and that the feature subset **J** is trend-relevant.

We focus on finding the subspace **J** that can potentially lead to discovery of subspace trends. We seek an efficient methodology for identifying the trend-relevant subspace without incurring an exhaustive and intractable search of the 2^M possible

subspaces. Before introducing our algorithm to find $\mathbf{J}$, we use the following three propositions to motivate our approach.

Proposition 1. For a set of data points, the presence of a subspace trend in **J** implies the presence of a subspace trend in every 2-D subspace of **J**.

The proof for this Proposition is obvious since a parametric curve in$\mathbf{J}$: $\mathbf{g_J}(\lambda) = [g_{j_1}(\lambda), g_{j_2}(\lambda), \cdots, g_{j_p}(\lambda)]^{\mathrm{T}}$, will manifest as a smooth curve in *any* 2-D projected subspace$\{ j_1, j_2\}$ in **J**, in the form: $[g_{j_1}(\lambda), g_{j_2}(\lambda)]^{\mathrm{T}}$.

Proposition 2. Presence of subspace trends in the 2-D spaces $\{j_1, j_2\}$ and $\{j_2, j_3\}$, implies the presence of a subspace trend in $\{j_1, j_2, j_3\}$.

Proof: Assume that in space $\{j_1, j_2\}$,

$$\left[\mathbf{x_{j_1}}, \mathbf{x_{j_2}}\right] = \left[g_{j_1}(\boldsymbol{\lambda}), g_{j_2}(\boldsymbol{\lambda})\right] + \left[\boldsymbol{\varepsilon}_{\mathbf{j_1}}, \boldsymbol{\varepsilon}_{\mathbf{j_2}}\right], \tag{1}$$

and in space $\{j_2, j_3\}$,

$$\left[\mathbf{x_{j_2}}, \mathbf{x_{j_3}}\right] = \left[g'_{j_2}(\boldsymbol{\lambda}'), g_{j_3}(\boldsymbol{\lambda}')\right] + \left[\boldsymbol{\varepsilon}'_{\mathbf{j_2}}, \boldsymbol{\varepsilon}_{\mathbf{j_3}}\right]. \tag{2}$$

Since $g_{j_2}(\lambda)$ and $g'_{j_2}(\lambda')$ both fit the data x_{j_2}, we have

$$\mathbf{x_{j_2}} = g_{j_2}(\boldsymbol{\lambda}) + \boldsymbol{\varepsilon}_{\mathbf{j_2}} = g'_{j_2}(\boldsymbol{\lambda}') + \boldsymbol{\varepsilon}'_{\mathbf{j_2}}. \tag{3}$$

We assume that $g_{j_2}(\lambda) = g'_{j_2}(\lambda')$. To find the relationship between λ and λ', we need to invert either g_{j_2} or g'_{j_2}. To make g_{j_2} invertible, we break g_{j_2} into piece-wise invertible monotonic functions $g_{j_2}^1 \ldots g_{j_2}^\tau$ with parameter values $\{\lambda^{(i)}\}$ broken into continuous sets of intervals $I_1 \ldots I_\tau$. The values of $\{\lambda'^{(i)}\}$ are correspondingly broken into intervals $I_1' \ldots I_\tau'$. Then $\lambda^{(i)} = {g_{j_2}^1}^{-1}(g'_{j_2}(\lambda'^{(i)}))$ when $\lambda'^{(i)} \in I_i'$ and $\lambda^{(i)} \in I_i$. Thus,

$$\lambda = {g_{j_2}}^{-1}\left({g'}_{j_2}(\lambda')\right) = \left[{g_{j_2}^1}^{-1}({g'}_{j_2}(I_1')), {g_{j_2}^2}^{-1}({g'}_{j_2}(I_2')), \dots, {g_{j_2}^\tau}^{-1}({g'}_{j_2}(I_\tau'))\right], \quad (4)$$

where ${g_{j_2}}^{-1}$ is the pseudo inverse defined by the piece-wise invertible functions. Then we can derive

$$g_{j_1}(\boldsymbol{\lambda}) = g_{j_1}({g_{j_2}}^{-1}({g'}_{j_2}(\boldsymbol{\lambda}'))) = {g'}_{j_1}(\boldsymbol{\lambda}'). \quad (5)$$

In the space $\{j_1, j_2, j_3\}$, we have

$$[\mathbf{x_{j_1}}, \mathbf{x_{j_2}}, \mathbf{x_{j_3}}] = [{g'}_{j_1}(\boldsymbol{\lambda}'), {g'}_{j_2}(\boldsymbol{\lambda}'), g_{j_3}(\boldsymbol{\lambda}')] + [\boldsymbol{\varepsilon_{j_1}}, \boldsymbol{\varepsilon'_{j_2}}, , \boldsymbol{\varepsilon'_{j_3}}]. \quad (6)$$

Therefore, we have a subspace trend in the subspace $\{j_1, j_2, j_3\}$.

Proposition 2 enables a bottom-up strategy. We can first examine each pair of dimensions, and evaluate which 2D subspace contains a trend. Then, we can aggregate the identified 2D subspaces to find higher-dimensional subspaces that contain interesting trend(s). Accordingly, the next question is how to detect trends for a given pair of dimensions $\{j_1, j_2\}$. Since a general curve $[g_{j_1}(\lambda), g_{j_2}(\lambda)]^{\mathrm{T}}$ can be complicated and difficult to evaluate, we focus on cases when $g_{j_1}(\lambda)$ or $g_{j_2}(\lambda)$ is a monotonic function of λ. Proposition 3 lays the foundation for evaluating subspace trends in 2D, and motivates the Neighborhood Similarity measure described in the next section.

Proposition 3. Assume curve $[g_{j_1}(\boldsymbol{\lambda}), g_{j_2}(\boldsymbol{\lambda})]$ in subspace $\mathbf{J} = \{j_1, j_2\}$ fits the data $[\mathbf{x}_{j_1}, \mathbf{x}_{j_2}]$, and $g_{j_1}(\lambda)$ is monotonic with respect to λ. For two neighboring points along j_1, i.e., $|x_{j_1}^{(2)} - x_{j_1}^{(1)}| < \delta_{j_1}$, there exists an upper bound for $\mathrm{E}\left[\left|x_{j_2}^{(2)} - x_{j_2}^{(1)}\right|\right]$, which is their difference in j_2.

Proof: Assume $\mathbf{x}_{\mathbf{J}}^{(1)}$ and $\mathbf{x}_{\mathbf{J}}^{(2)}$, are two neighboring points according to subspace j_1, i.e., $|x_{j_1}^{(2)} - x_{j_1}^{(1)}| < \delta_{j_1}$. Also assume curve $[g_{j_1}(\boldsymbol{\lambda}), g_{j_2}(\boldsymbol{\lambda})]$ fits the data in

subspace $\{j_1, j_2\}$, $\mathbf{x_J^{(1)}} = \mathbf{g_J}(\lambda^{(1)}) + \boldsymbol{\varepsilon}_\mathbf{J}^{(1)}$, $\mathbf{x_J^{(2)}} = \mathbf{g_J}(\lambda^{(2)}) + \boldsymbol{\varepsilon}_\mathbf{J}^{(2)}$ and $\mathrm{E}[\boldsymbol{\varepsilon}_\mathbf{J}^{(i)} | \mathbf{x_J}, \mathbf{g_J}] \leq \boldsymbol{\epsilon}$. Since the two points are neighbors in j_1, we have $|x_{j_1}^{(2)} - x_{j_1}^{(1)}| < \delta_{j_1}$. Therefore we have

$$|g_{j_1}(\lambda^{(2)}) - g_{j_1}(\lambda^{(1)}) + \varepsilon_{j_1}^{(2)} - \varepsilon_{j_1}^{(1)}| \leq \delta_{j_1} \tag{7}$$

and

$$\mathrm{E}(|g_{j_1}(\lambda^{(2)}) - g_{j_1}(\lambda^{(1)})|) \leq \delta_{j_1} + 2\epsilon_{j_1} = \delta_{j_1}'. \tag{8}$$

Since $g_{j_1}(\lambda)$ is assumed to be monotonic and continuous, $\exists \gamma$ such that $|\lambda^{(2)} - \lambda^{(1)}| \leq \gamma$. And because of continuity of $g_{j_2}(\lambda)$, $\exists \delta_{j_2}$ such that $|g_{j_2}(\lambda^{(2)}) - g_{j_2}(\lambda^{(1)})| \leq \delta_{j_2}$

Assume $\left(\mathbf{g_J}(\lambda^{(1)}) - \mathbf{g_J}(\lambda^{(2)})\right) \perp \boldsymbol{\varepsilon}_\mathbf{J}^{(1)}, \boldsymbol{\varepsilon}_\mathbf{J}^{(2)}$, then

$$\mathrm{E}[\left(x_{j_2}^{(2)} - x_{j_2}^{(1)}\right)^2]$$

$$= \mathrm{E}[\left|\left|\mathbf{g_J}(\lambda^{(2)}) - \mathbf{g_J}(\lambda^{(1)})\right|\right|^2 + \left|\left|\boldsymbol{\varepsilon}_\mathbf{J}^{(2)} - \boldsymbol{\varepsilon}_\mathbf{J}^{(1)}\right|\right|^2 - \left(x_{j_1}^{(2)} - x_{j_1}^{(1)}\right)^2]$$

$$= \mathrm{E}[|g_{j_2}(\lambda^{(2)}) - g_{j_2}(\lambda^{(1)})|^2 + |g_{j_1}(\lambda^{(2)}) - g_{j_1}(\lambda^{(1)})|^2 + \left|\left|\boldsymbol{\varepsilon}_\mathbf{J}^{(2)} - \boldsymbol{\varepsilon}_\mathbf{J}^{(1)}\right|\right|^2 - \left(x_{j_1}^{(2)} - x_{j_1}^{(1)}\right)^2]$$

$$\leq \delta_{j_2}^2 + \delta_{j_1}'^2 - \left(x_{j_1}^{(2)} - x_{j_1}^{(1)}\right)^2 + 4||\boldsymbol{\epsilon}||^2, \tag{9}$$

which is a bound for the difference between the two points in j_2. Intuitively, Proposition 3 says that, if the data points in a 2D subspace fit well with a 2D subspace curve, neighboring points with respect to one of the dimensions are likely to be neighboring points according to the other dimension, primarily because of continuity of the subspace curve.

The above three propositions inspired the neighborhood similarity measure and trend-relevant subspace identification method (following sections). For each 2D subspace $\{j_1, j_2\}$, we ask whether neighboring points along j_1 are also potentially neighbors along j_2. The neighborhood similarity measure reflects whether a subspace curve is likely to exist in $\{j_1, j_2\}$, according to Proposition 3. After examining each 2D subspace according to Proposition 2, we aggregate the 2D subspaces that are likely to contain subspace curves, and build up a higher dimensional subspace **J** that contains a subspace curve. If the aggregation process generates multiple non-overlapping subspaces, we consider the one with the highest dimensionality to be the most meaningful subspace. This is because a more significant trend in the data is likely to manifest in a larger number of dimensions.

In the following sections, we describe the Neighborhood Similarity measure, and use it to determine whether a subspace curve is likely to be present in each 2D subspace. Then, we denoise the pairwise similarity matrix using an entropy-based method, and select the feature pairs. Based on Proposition 2, we aggregate the selected pairs systematically to generate non-overlapping feature subsets and hypothesize that the largest feature subset is likely to reveal the salient subspace trends in the data. The symbols have been summarized in Table 1.

Table 1 Glossary of Symbols

Symbol	*Representation*
N	number of samples
M	feature dimensions
$\mathbf{x}^{(i)}$	data point $[x_1^{(i)}, x_2^{(i)}, \dots, x_M^{(i)}]^T \in \mathbf{R}^M$
$\mathbf{x_i}$	feature vector $[x_j^{(1)}, x_j^{(2)}, \dots, x_j^{(N)}]^T \in \mathbf{R}^N$
$\mathbf{g_J}(\lambda)$	parametric curve $[g_{j_1}(\lambda), g_{j_2}(\lambda), \cdots, g_{j_p}(\lambda)]^{\mathrm{T}}$ in subspace J
$\boldsymbol{D}_{\boldsymbol{x_i}}$	pair-wise distance matrix
$E_{\boldsymbol{x_i}}^k$	edge set of *k*-NNG built on $\mathbf{x_i}$
$\hat{E}$	edge set of fully connected graph of samples
$S_{D_{\boldsymbol{x_i}}, E_{\boldsymbol{x_j}}^k}$	multiset of edge lengths
B	number of bins
$W_{D_{\boldsymbol{x_i}}, E_{\boldsymbol{x_j}}^k}$	distribution of values in $S_{D_{\boldsymbol{x_i}}, E_{\boldsymbol{x_j}}^k}$
p_{ij}	the amount of product transported in earth mover's distance (EMD)
c_{ij}	cost function in EMD
$NS_{\boldsymbol{x_i} \to \boldsymbol{x_j}}$	directional neighborhood similarity
$NS_{\boldsymbol{x_i}, \boldsymbol{x_j}}$	mutual neighborhood similarity
$\boldsymbol{\Phi}$	neighborhood similarity matrix
h_l	occurrence frequency of level l
L	number of bins for entropy estimation
T	similarity threshold
$\rho(l, T)$	membership coefficient
$R_{Z1}(T)$	average information of fuzzy set Z_1
γ	threshold indication parameter

3.1.2 The Neighborhood Similarity Measure

Consider two features $\mathbf{x_1}$ and $\mathbf{x_2}$ that can be modeled by the continuous functions: $\mathbf{x_1} = g_1(\boldsymbol{\lambda}) + \boldsymbol{\varepsilon_1}$, $\mathbf{x_2} = g_2(\boldsymbol{\lambda}) + \boldsymbol{\varepsilon_2}$. To appreciate Proposition 3, consider the toy dataset drawn from a parabolic curve**,** where $\mathbf{x_1} = \boldsymbol{\lambda}, \mathbf{x_2} = (\boldsymbol{\lambda} - 1)^{\mathbf{2}}, \boldsymbol{\lambda} \in \mathbf{R^N}$, as shown in Figure 2(a). Each of the blue dots represents a simulated data point. The dashed vertical and horizontal lines indicate small intervals along the respective feature axes. In Figure 1(a), it can be observed that the neighboring data points (within δ_1) with respect to $\mathbf{x_1}$, i.e., the points between the red dashed vertical lines, are bounded within a small range with respect to $\mathbf{x_2}$, between the green dashed horizontal lines. In contrast, Fig. 2(b) shows simulated data points for two independent variables $\mathbf{x_3}$ **,** $\mathbf{x_4}$ drawn from independent uniform distributions. Data points that are neighbors with respect to feature $\mathbf{x_3}$ can be arbitrarily distant with respect to $\mathbf{x_4}$**,** and *vice versa*. Based on Proposition 3, we proposed a quantitative metric to measure the association between two features by examining neighboring relationships among data points in selected subspaces as described further below.

To analyze neighborhood relationships, we propose to use *k*-nearest neighbor graphs (*k*-NNG's) to make δ_1 adapt to the local density of data points. In one *k*-NNG, the nodes represents data points, and edges connect nodes that are *k*-nearest neighbors defined by a distance measure based on one individual feature. To compare features $\mathbf{x_1}$ and $\mathbf{x_2}$**,** we first build a *k*-NNG based on a distance measure defined by only one feature, say $\mathbf{x_1}$, as

$$\boldsymbol{D_{x_1}}(i,j) = \left| x_1^{(i)} - x_1^{(j)} \right|.$$

Let $E_{x_1}^k$ denote the set of edges for this *k*-NNG. The set of edges for a fully

connected graph is denoted $\hat{E}$. We next turn to $\mathbf{x_2}$ to examine whether the neighboring data points with respect to $\mathbf{x_1}$ are also likely to be close to each other with respect to $\mathbf{x_2}$. Similarly, we construct the edge set $E_{x_2}^k$ and examine whether the *k*-nearest neighbors with respect to $\mathbf{x_2}$ are also likely to be proximal with respect to $\mathbf{x_1}$. We define the following multisets of edge lengths as

$$S_{D_{x_2},E_{x_1}^k} = \left\{D_{x_2}(i,j)\middle| i,j \in E_{x_1}^k\right\},$$

$$S_{D_{x_2},\hat{E}} = \left\{D_{x_2}(i,j)\middle| i,j = 1,\dots,N\right\},$$

$$S_{D_{x_1},E_{x_2}^k} = \left\{D_{x_1}(i,j)\middle| i,j \in E_{x_2}^k\right\},$$

$$S_{D_{x_1},\hat{E}} = \left\{D_{x_1}(i,j)\middle| i,j = 1,\dots,N\right\}.$$

A multiset, a.k.a. *bag* is a generalization of the notion of a set in which members are allowed to appear more than once. The distributions of edge lengths that can be estimated from these multisets are highly informative. For example, if the multiset of edge lengths, $S_{D_{x_2},E_{x_1}^k}$ or $S_{D_{x_1},E_{x_2}^k}$ contain predominantly small values from $S_{D_{x_2},\hat{E}}$ or $S_{D_{x_1},\hat{E}}$, it is an indication that the two features are associated. As a sanity check, consider the problem of comparing feature $\mathbf{x_1}$ against itself, i.e., comparing the two multisets $S_{D_{x_1},E_{x_1}^k}$ (lengths of edges of *k*-NNG built based on $\mathbf{x_1}$) and $S_{D_{x_1},\hat{E}}$ (distances in terms of $\mathbf{x_1}$ between all pairs of data points). From the definition of *k*-NNG, values in the multiset $S_{D_{x_1},E_{x_1}^k}$ will take on small values from $S_{D_{x_1},\hat{E}}$, indicating that $\mathbf{x_1}$ is associated with itself.

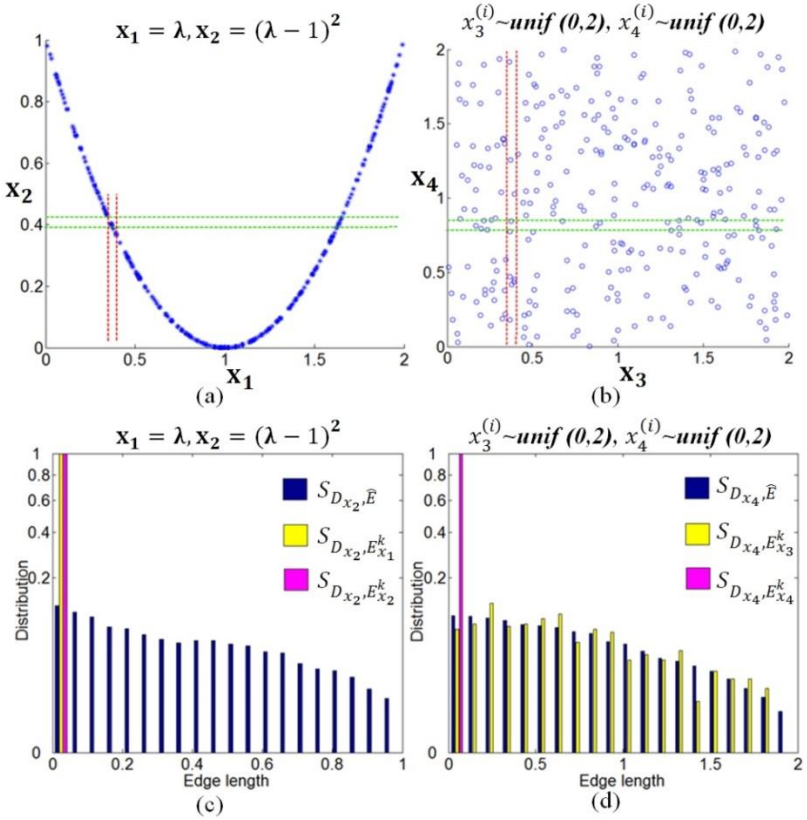

Figure 2. The neighborhood similarity for measuring associations between two features (k = 4, B = 20). (a) Parabolic association, dashed lines indicate the neighboring data points; (b) Independent variables; (c, d) the distributions derived from the k-NNGs for the associations in (a) and (b) respectively.

For the toy example, we have plotted the distributions of $S_{D_{x_2},E^k_{x_1}}$, $S_{D_{x_2},E^k_{x_2}}$ and $S_{D_{x_2},\hat{E}}$ in Panels c and d. This example illustrates the intuition behind the idea of detecting a trend-relevant association between features by comparing the distributions of edge lengths derived from their respective feature-wise k-NNGs. Specifically, for the parabolic association between $\mathbf{x_1}$ and $\mathbf{x_2}$ in Figure 2(a), the distribution of $S_{D_{x_2},E^k_{x_1}}$ is located to the left and is distinct from $S_{D_{x_2},\hat{E}}$, as shown in Figure 2(c). In contrast, for the two independent variables $\mathbf{x_3}$ and $\mathbf{x_4}$, the distributions of $S_{D_{x_4},E^k_{x_3}}$ and $S_{D_{x_4},\hat{E}}$ are similar, as shown in Figure 2(d).

To quantify the similarity between distributions, the Earth Mover's Distance

(EMD) [58] is preferable, since it provides us with flexibility in defining the cost function, and intrinsically avoids "infinite distances". Basically, the EMD between two distributions is the solution to a transportation problem that can be formulated as a linear programming (LP) task. Specifically, denote the empirical distribution of $S_{D_{x_2},E^k_{x_1}}$ as $W_{D_{x_2},E^k_{x_1}}$, and the empirical distribution of $S_{D_{x_2},\hat{E}}$ as $W_{D_{x_2},\hat{E}}$. We establish B equally spaced bins for the values in $S_{D_{x_2},\hat{E}}$ to estimate its distribution $W_{D_{x_2},\hat{E}} = \{w_1^{x_2}, w_2^{x_2}, \dots, w_B^{x_2}\}$, and apply the same bins to the values in $S_{D_{x_2},E^k_{x_1}}$ to estimate the distribution $W_{D_{x_2},E^k_{x_1}} = \{w_1^{x_2,x_1}, w_2^{x_2,x_1}, \dots, w_B^{x_2,x_1}\}$. Treating $w_b^{x_2}$ as the amount of product supplied at location b, and $w_b^{x_2,x_1}$ as the product demanded at location b, the EMD is defined as the minimum work required for solving the transportation task as

$$\text{EMD}\left(W_{D_{x_2},\hat{E}}, W_{D_{x_2},E^k_{x_1}}\right) = min \sum\nolimits_{ij} p_{ij} c_{ij},$$

subject to:

$$\begin{cases} p_{ij} \geq 0 \,, 1 \leq i, j \leq B \\ \sum_{j=1}^{B} p_{ij} = w_i^{x_2} \\ \sum_{i=1}^{B} p_{ij} = w_j^{x_2,x_1} \end{cases}, \tag{10}$$

where p_{ij} is the amount of product transported from location i to location j, and c_{ij} is the cost of moving a unit of product from location i to j, and given by $c_{ij} = i - j$ when $i > j$, and $c_{ij} = 0$ when $i \leq j$. The cost c_{ij} is defined in such a way that only the movements from (right) high-indexed locations (location i) to the (left) low-indexed locations (location j) are considered. This is because the distribution $W_{D_{x_2},E^k_{x_1}}$ will be situated to the left of $W_{D_{x_2},\hat{E}}$ when $\mathbf{x_1}$ and $\mathbf{x_2}$ are associated (since $S_{D_{x_2},E^k_{x_1}}$ take small values from

$S_{D_{x_2},\hat{E}}$ as mentioned earlier). The more $W_{D_{x_2},E^k_{x_1}}$ is situated to the left of $W_{D_{x_2},\hat{E}}$, the larger the EMD value between $W_{D_{x_2},E^k_{x_1}}$ and $W_{D_{x_2},\hat{E}}$, and the more closely the two features are associated. Using the EMD measure, we now define the directional *neighborhood similarity* of features $\mathbf{x_1}$ and $\mathbf{x_2}$, denoted $NS_{x_1\to x_2}$ as

$$NS_{x_1\to x_2} = \frac{\text{EMD}\left(W_{D_{x_2},\hat{E}}, W_{D_{x_2},E^k_{x_1}}\right)}{\text{EMD}\left(W_{D_{x_2},\hat{E}}, W_{D_{x_2},E^k_{x_2}}\right)}. \tag{11}$$

Similarly, the directional neighborhood similarity of $\mathbf{x_2}$ to $\mathbf{x_1}$, is denoted $NS_{x_2\to x_1}$, and written as

$$NS_{x_2\to x_1} = \frac{\text{EMD}\left(W_{D_{x_1},\hat{E}}, W_{D_{x_1},E^k_{x_2}}\right)}{\text{EMD}\left(W_{D_{x_1},\hat{E}}, W_{D_{x_1},E^k_{x_1}}\right)}. \tag{12}$$

In the above definitions, the denominators normalize the *NS* measure to [0,1]. When the approximating function of x_1, $g_1(\lambda)$ is a monotonic function of λ, $NS_{x_1\to x_2}$ is close to 1, while $NS_{x_2\to x_1}$ can be as high as 1 when $g_2(\lambda)$ is also a monotonic continuous function or much lower than 1 when $g_2(\lambda)$ is non-monotonic. We define the larger of these values as the mutual neighborhood similarity between $\mathbf{x_1}$ and $\mathbf{x_2}$ as

$$NS_{x_1,x_2} = \max\left(NS_{x_1\to x_2}, NS_{x_2\to x_1}\right). \tag{13}$$

Neighborhood similarity values lie in the interval between 0 and 1. $NS_{x_1,x_2} = 1$ when $NS_{x_1\to x_2} = 1$ or $NS_{x_2\to x_1} = 1$, which indicates that $[\mathbf{x_1}, \mathbf{x_2}]$ can be approximated by $[g_1(\boldsymbol{\lambda}), g_2(\boldsymbol{\lambda})]$ with either $g_1(\lambda)$ or $g_2(\lambda)$ being a monotonic continuous function of λ or both.

$NS_{x_1,x_2} = 0$ when $\text{EMD}\left(W_{D_{x_1},\hat{E}}, W_{D_{x_1},E^k_{x_2}}\right) = 0$ and $\text{EMD}\left(W_{D_{x_2},\hat{E}}, W_{D_{x_2},E^k_{x_1}}\right) = 0$, which indicates that the distribution $W_{D_{x_1},E^k_{x_2}}$ is the same as $W_{D_{x_1},\hat{E}}$ when two

features are independent. It is the same for $W_{D_{x_2}, E^k_{x_1}}$.

3.1.2. Trend-relevant feature selection

Given a multivariate data set, our strategy for trend-relevant feature selection is to perform an automatic partitioning of a neighborhood similarity graph that is built from the neighborhood similarity matrix $\boldsymbol{\Phi}$, with entries $\boldsymbol{\Phi}(i,j) = NS_{x_i,x_j}$. This is a "feature graph" in which the nodes represent features, and edge weights are defined by the neighborhood similarity values between the respective pairs of features. Based on the pair-wise neighborhood similarities among the features, our strategy is to identify pairs of features for which subspace curves are present in the corresponding 2-D space. Then we construct the trend-relevant feature subset by merging the identified pairs according to Proposition 2.

We consider a pair of features to be associated with a trend if their mutual neighborhood similarity exceeds a certain threshold. If the threshold were too high (an overly selective analysis), too few features would be selected to effectively capture the trend. If the threshold is too low, we face the possibility of including irrelevant features that can obscure the trend. The critical problem, therefore, is how to identify the threshold effectively to lead to meaningful trend discovery. For this we use an entropy-based algorithm (described below) to automatically estimate the correct threshold. Once the threshold is identified, we partition the neighborhood similarity graph by cutting all the edges with weights below the threshold. After the partitioning, each connected component represents a subset of the features that are associated with each other in terms of the neighborhood similarity. Intuitively, components containing a larger number of features indicate strong underlying trends. As we gradually decrease the threshold from

the maximum similarity value, the size of the largest component grows. The order in which additional features are included in the largest connected component also defines the order in which features should be selected for trend discovery.

For selecting an appropriate threshold, we adopt the Shanbhag algorithm [59] that utilizes an information measure to estimate the threshold for the matrix by modeling $\boldsymbol{\Phi}$ as a composition of two fuzzy sets, Z_1 and Z_2. Z_1 contains the neighborhood similarity values that are mainly generated by the noise and Z_2 contains the values generated by the curve "signal." We derive an empirical distribution using an equally spaced histogram with L bins and calculate the membership coefficient of each bin level to belong to Z_1 or Z_2. Suppose h_l is the occurrence frequency at bin level l and $\sum_{l=1}^{L} h_l = 1$. If the threshold is set at bin level T, then the overall occurrence frequency of levels belonging to Z_1 is $h_{Z1}(T) = \sum_{l=1}^{T} h_l$, and that belonging to Z_2 is $h_{Z2}(T) = \sum_{l=T+1}^{L} h_l$. The membership coefficient of a level l when the threshold is set at level T is denoted $\rho(l,T)$. It measures the degree of membership of the level to either fuzzy set, and is given as

$$\rho(l,T) = \begin{cases} \frac{1}{2}\left(1 + \frac{1}{h_{Z1}(T)} \sum_{i=l}^{T} h_i\right), 1 \le l \le T \\ \frac{1}{2}\left(1 + \frac{1}{h_{Z2}(T)} \sum_{i=T+1}^{l} h_i\right), T < l \le L \end{cases}. \quad (14)$$

The boundary conditions for $\rho(l,T)$ are $\rho(1,T) = 1$, and $\rho(L,T) = 1$ since the lowest bin level 1 always belongs to Z_1 and the highest bin level L always belongs to Z_2. The terms $1/(2h_{Z1}(T))$ and $1/(2h_{Z2}(T))$ ensure that the boundary conditions are met. The average information at threshold level T, as measured by the entropy R is given by

$$R_{Z1}(T) = -\sum_{l=1}^{T} \frac{h_l}{h_{Z1}(T)} \log \rho(l,T) \quad (15)$$

and $$R_{Z2}(T) = -\sum_{l=T+1}^{L} \frac{h_l}{h_{Z2}(T)} \log \rho(l,T). \quad (16)$$

In Shanbhag's approach, T is chosen to minimize $|R_{Z1}(T) - R_{Z2}(T)|$. As we decrease T from L to 1, we expect this absolute difference to decrease rapidly and then decay slowly in most cases. Instead of finding the minimum, T is set to the bin where the rapid decrease ceases. In our implementation, we start by iterating T through all bin levels from 1 to L to find the minimum of $|R_{Z1}(T) - R_{Z2}(T)|$ as R_{MIN}. Then we decrease T from L to find the first bin T, denoted $\hat{T}$ that makes $|R_{Z1}(T) - R_{Z2}(T)|$ fall below $R_{\mathrm{MIN}} + \gamma$. Here γ is small-value indication constant added to R_{MIN} to help indicate where the rapid decrease stops. The threshold is then set at bin level $\hat{T}$. After cutting the edges below this threshold, the trend-relevant features are identified from the connected component(s) that contain a significant number of features. Each feature subset is used for further dimension reduction to visualize the corresponding subspace trend, separately.

The above analysis also provides a basis for quantifying the strength of a feature subset. Specifically, the number of features contained in the resulting connected component and the automatically identified threshold $\hat{T}$ are useful indicators of the strength of the discovered trend. The higher the $\hat{T}$, the stronger the trend-relevant features are associated; and the larger the number of features contained, the more strongly the trend is supported. Intuitively, a random pattern should result in low $\hat{T}$ and will have a small feature set if we manually set the threshold high, and *vice versa*. As a result, the largest connected component derived from a relatively high $\hat{T}$ is usually of most interest for further analysis and verification, as described next.

3.1.3. Feature evaluation and trend validation

Once we have selected the candidate feature subset and derived the corresponding trend based on the selected features subset, the next question is how to evaluate the features comparing with the trend. How are the features aligned with the derived trend? Are there any missing features? To evaluate the feature alignment with the trend, we can borrow the idea from previously proposed neighborhood similarity by checking whether the neighboring points along the trend are also neighboring points along each feature. The procedure is shown in Table 2.

Similar to the way to calculate neighborhood similarity introduced in Section 3.1.2, we first construct a *k*-NNG to represent the derived trend and compare it with the *k*-NNG built from the feature. If the trend *k*-NNG picks the small distances just like the feature *k*-NNG, then we say the feature is well aligned with the trend. After calculating the scores for all the features, we order the features descending and see whether we have missed any features that have relatively high scores. Basically, we include the unselect features between the selected features if these two features are not far apart by this descending order (not greater than 10) and exclude the selected features that are too far apart from the rest of the selected features in the descending order (greater than 10). Thus we update the feature subset and corresponding update the trend. Then we undergo the same iteration until the feature subset remains the same. At this time, all the selected features have higher scores than the unselected features.

The reason behind the possible missed feature is that since it is an unsupervised feature selection, we select the features collectively by a pair-wise evaluation. With an assumed target based on the selected features, there can be features more aligned with the

target than with the other features. According to our definition of neighborhood similarity, there are some pair-wise relationships that cannot be caught. By this iterating process, we can make up the flaws of neighborhood similarity to some extent.

Table 2. Procedure for feature evaluation and post tuning update

Input: Candidate feature subset $\hat{\mathbf{J}}$
Output: Updated feature subset $\hat{\mathbf{J}}$ and feature weight $NS_{\hat{J}\to x_i}$ for all features $\{\mathbf{x_i}\}$.

Step 1. Construct k-NNG of the data points based on candidate feature subset as $E^k_{\hat{J}}$.
Step 2. Construct k-NNG based on each individual feature $\mathbf{x_i}$ as $E^k_{x_i}$.
Step 3. Calculate neighborhood similarity (NS) from trend to feature $NS_{\hat{J}\to x_i}$:

$$NS_{\hat{J}\to x_i} = \frac{\mathrm{EMD}\left(W_{D_{x_i}\hat{E}}, W_{D_{x_i}E^k_{\hat{J}}}\right)}{\mathrm{EMD}\left(W_{D_{x_i}\hat{E}}, W_{D_{x_i}E^k_{x_i}}\right)},$$

where $\boldsymbol{D}_{x_i}(p,q) = \left|x_i^{(p)} - x_i^{(q)}\right|$,
$W_{D_{x_i}E^k_{\hat{J}}}$: distribution of edge weights $S_{D_{x_i}E^k_{\hat{J}}} = \left\{D_{x_2}(i,j)\middle|i,j \in E^k_{x_1}\right\}$,
$\hat{E}$: fully connected graph,
EMD : Earth Mover's Distance to compare distributions.

Step 4. Order all the features $\{\mathbf{x_i}\}$ descendingly according to $NS_{\hat{J}\to x_i}$ and smooth the selection status of features (include and exclude features to $\hat{\mathbf{J}}$).

Step 5. Go to Step 1 with updated $\hat{\mathbf{J}}$ and iterate till $\hat{\mathbf{J}}$ does not change.

Next we aim to validate the hypothesized trend quantitatively provided the ground truth. Basically it can be formulated as a question of how to compare the ground-truth trend, represented by the graph G and the estimated trend, represented by the MST $\hat{G}$. We encode the actual trend with graph G describing the progressive relationship among different states or classes represented by the graph nodes. If the states are adjacent in the trend, the corresponding nodes are connected in the graph. The goal is to calculate the correct edge connections in $\hat{G}$. If the two connected nodes in $\hat{G}$ are within a preset

distance in G, we consider the edge connection correct. Under some circumstances, the data points can come from the same cluster or state. Switching the data points belonging to the same cluster should not affect the final evaluation result. Therefore, we assign a cluster label to each data point and redefine the graph node distance, i.e., the shorted hop distance based on the cluster labels. Suppose G and $\hat{G}$ are the corresponding graph adjacency matrices. We define the connection accuracy of the estimated trend $\hat{G}$, given G, and the class label l when available (otherwise treat each data point as an individual class), as

$$A_{G,l}^{d}(\hat{G}) = \frac{\sum_{\hat{G}(i,j)=1} \mathbf{1}(H_G(i,j) \leq d)}{\sum_{i,j} \hat{G}(i,j)}, \tag{17}$$

where H_G is the shortest-hop matrix for graph G depending on class label l. Specifically, $H_G(i,j) = 0$ when $l(i) = l(j)$, since exchanging their positions in G does not affect the trend. The hop is incremented by 1 only when reaching a node from a different class. In the above equation, the notation "$\mathbf{1}(< condition >)$" represents an indicator function that is 1 when the condition indicated within the parentheses is true, and is 0 otherwise. An edge in $\hat{G}$ is considered to be a correct connection when it connects two data points that are within a hop distance d in graph G, which indicates that they are adjacent in the actual trend or belong to the same class. The connection accuracy $A_{G,l}^{d}(\hat{G})$ describes the percentage of correct connections in the derived MST compared with the known connections from the ground truth trend.

In the result section, we will calculate the connection accuracy for the hypothesized trend with the selected features and compare it with the connection accuracy for the trend derived with all the features. We can see how greatly the unsupervised feature selection will help in revealing the trend from high dimensional dataset.

3.1.4. Trend visualization

Once the trend-relevant features are identified as described above, it becomes possible to detect and visualize the trends by imposing a suitable ordering over the data points. For this, we construct a graph whose nodes are the data points, and whose edge lengths are the Euclidean distances defined on just the trend-relevant features. The minimum spanning tree (MST) for this graph imposes an ordering over the data points in this selected subspace. Traversals of the MST reveal gradual variations of the trend-relevant features. In this work, we initiate the graph traversal at one end of the longest path of the MST, and then perform a depth-first traversal of the remaining nodes of the MST [60].

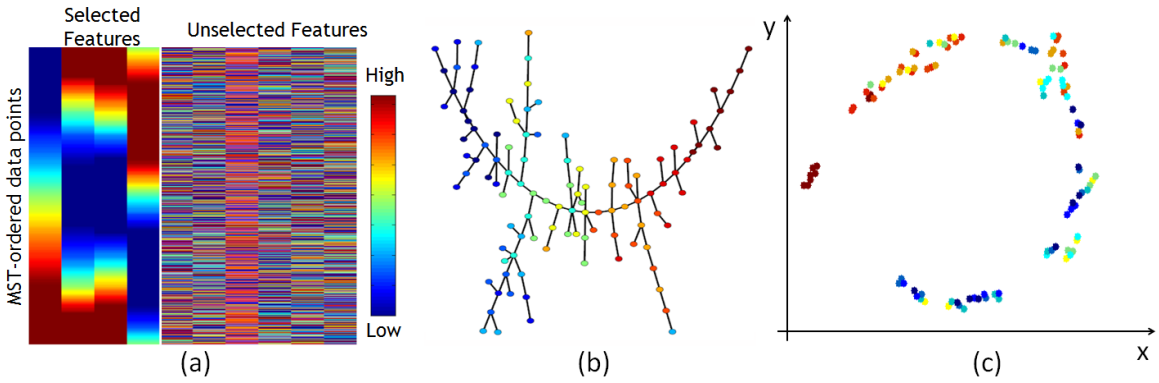

Figure 3. Trend visualization. (a) MST-ordered heatmap; (b) Tree visualization, the graph nodes are data points; (c) t-distributed stochastic neighborhood embedding. The color indicates the underlying progression factor in (b) and (c).

For visualization, we construct a set of MST-ordered heatmaps, one for each identified feature subset, as illustrated in Figure 3(a). In these heatmaps, each row corresponds to a data point, each column corresponds to a feature, and the color indicates the feature value, indexed by a color table. The row ordering of the heatmaps is determined by the depth-first traversal order of the MST [60]. The column order of the heatmap is determined by an agglomerative clustering of the features based on Pearson's

correlation [61], so the linearly correlated features that look alike within the selected feature subset are placed closer in the heatmap representation. These heatmaps show the concordant gradual variations as gradual variations in colors of the trend-relevant features associated with the common trend they support. It is possible to have occasional abrupt changes of features in the heatmap (jumps). They indicate significant branches in the MST caused by the intersection or forking of curves in the feature subspace.

In addition to the heatmap representation described above, subspace trends can also be visualized by directly visualizing the data points using various graph layout or dimension reduction algorithms. For example, we can use a force directed graph-layout algorithm, TreeVis [62], as shown in Figure 3(b) to visualize the MST built on the selected features. The goal of this algorithm is to make the graph as spread out as possible to reveal the relationship among the data points by simulating a force-attracting-repelling physics system. We can also directly reduce the dimension of the high dimensional subspace to 2 dimensions or 3 dimensions. T-SNE [63], t-distributed stochastic neighborhood embedding, as shown in Figure 3(c) is one of these dimension reduction algorithms to preserve the neighborhood of data points in high dimensional space in low dimension. In both Figure 3(b) and (c), we colored the nodes by the underlying progression factor to illustrate how the discovered trend is aligned with the underlying factor. We can also color the nodes with feature values to illustrate the gradual changes of features values along the trend.

3.2. Online Density-based Representation

When the data sets become huge, it is unlikely to visualize the data points efficiently with the previous visualization approaches since the computational complexity is usually around $O(n^2)$ with the number of data points and the rare population can easily get dominated by the major populations. The computational complexity of the popular visualization algorithms is summarized in the following Table 3. We can see that none of these visualization algorithms is online and has at least computational complexity of $O(n \log (n))$. Our proposed algorithm has a sublinear computational complexity and can be updated in an online way, which means the visualization can be updated directly with the new data points without rerunning the algorithm with the whole updated data set.

Table 3 Computation complexity of visualization algorithms

Visualization algorithm	Computational complexity	Online algorithm
Force-directed graph layout	above $O(n^2)$	N
ISOMAP	$O(n^3)$	N
Landmark ISOMAP	$O(d\ l\ n \log(n) + l^3)$ (l: the number of landmarks)	N
tSNE	$O(n^2)$	N
Scalable tSNE	$O(n \log(n))$	N
Proposed online density-based representation	$O(K\ n)$ (K: the number of derived Gaussian components)	Y

The basic idea behind this representation is to use an online density based method to summarize the data points with Gaussian mixtures and derive a trend map representation based on these Gaussians. The key is to approximate the data density, either Gaussian or non-Gaussian, with weighted Gaussian components. We can further

cluster the data points into homogeneous groups according to the Gaussian components and derive the MST-ordered heatmap from these Gaussian components. Data density is estimated by a kernel density estimation algorithm with an auto compression to reduce the number of components and thus to constrain the computational complexity with increasing number of data points.

In the following sections, we introduce the classical kernel density estimation, the online kernel density estimation algorithm and the derived online trend representation.

3.2.1. Kernel density estimation: A compressed model

Kernel density estimation (KDE) is a non-parametric way to estimate the probability density function of a random variable. It is basically a smoothing problem to estimate the empirical distribution function. Comparing the KDE with a compressed model with a classical density model, the difference lies in whether we have compressed the data points with representative components, such as Gaussian components, as illustrated in Figure 4. In the classical model, we directly convolve the kernel with each individual data point to represent the data density; however in compressed model, we convolve the kernel with the representative Gaussian components and derive a similar density representation with the classical model. With this component representation, the computational complexity can be maintained regarding the number of components instead of the increasing number of data points and we do not need to save all the data points to approximate the density but just save the components parameters, such as the mean and covariance matric of Gaussian component.

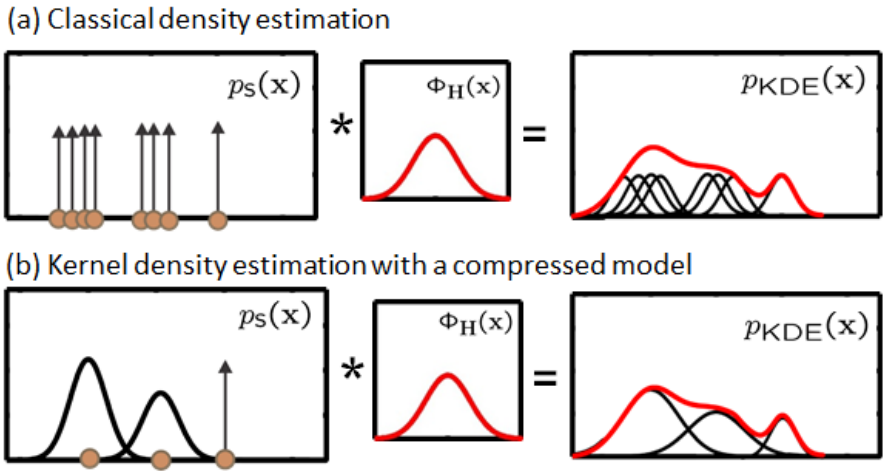

Figure 4. Illustration of kernel density estimation with and without compression. (a) Classical kernel density estimation: kernel is convolved with each individual data point; (b) Kernel density estimation with a compressed model: the kernel is convolved with each Gaussian component.

One challenge with this non-parametric density approximation is the choosing an appropriate kernel bandwidth of the kernel function which is crucial for precise estimation. Either over-smoothing or under-smoothing can substantially reduce precision. Several approaches have been proposed to estimate the bandwidth such as cross validation and plug-in method [64] which minimizes the mean integrated square error (MISE). In our implementation, we used the Gaussian component for compressing the data and MISE for bandwidth selection. In the following section, we introduced the kernel density estimation model with/without data compression and how to select the kernel bandwidth according to MISE.

Let $(\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)})$ be an independently and identically distributed sample drawn from some distribution with an unknown density $\boldsymbol{p}(\boldsymbol{x})$. The original data points, the sample distribution can be expressed as Gaussian components with zero covariance:

$$p_S(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \phi_0(\mathbf{x} - \mathbf{x}^{(i)}), \tag{18}$$

where $\phi_{\mathbf{\Sigma}}$ is the Gaussian kernel with bandwidth $\mathbf{\Sigma}$, defined as

$$\phi_{\mathbf{\Sigma}}(\mathbf{x}-\boldsymbol{\mu}) = (2\pi)^{-d/2}|\mathbf{\Sigma}|^{-1/2}e^{(-1/2(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}))}. \quad (19)$$

The kernel density estimation $\hat{p}_{\mathrm{KDE}}(\mathbf{x})$ is defined as

$$\hat{p}_{\mathrm{KDE}}(\mathbf{x}) = \phi_{\mathbf{H}}(\mathbf{x}) * p_s(\mathbf{x}) = \sum_{i=1}^{N} \alpha_i \phi_{\mathbf{H}+\mathbf{\Sigma}_{si}}(\mathbf{x}-\mathbf{x}^{(i)}) = \sum_{i=1}^{N} \alpha_i \phi_{\mathbf{H}}(\mathbf{x}-\mathbf{x}^{(i)}), \quad (20)$$

which is the convolution of the sample distribution with a Gaussian kernel with bandwidth **H.**

The bandwidth **H** can be estimated by asymptotic integrated squared error (AMISE) [65], which is

$$\mathrm{AMISE} = (4\pi)^{-\mathrm{d}/2}|\mathrm{H}|^{-1/2}\mathrm{N}_{\alpha}^{-1} + \frac{1}{4}\mathrm{d}^2 \int \mathrm{tr}^2\{\mathrm{H}\mathcal{G}_{\mathrm{p}}(\mathrm{x})\}\mathrm{dx}\,. \quad (21)$$

Assume $\mathbf{H}_{\mathrm{opt}} = \beta_{\mathrm{opt}}{}^{\mathbf{2}}\mathbf{F}$, AMISE is minimized at

$$\beta_{\mathrm{opt}} = [\mathrm{d}(4\pi)^{\mathrm{d}/2}\mathrm{N}_{\alpha}\mathrm{R}(\mathrm{p},\mathrm{F})]^{-1/(\mathrm{d}+4)}, \quad (22)$$

where $R(p,\mathbf{F})$ is a functional of the second-order partial derivatives $\mathcal{G}_p(\mathbf{x})$:

$$\mathrm{R}(\mathrm{p},\mathrm{F}) = \int \mathrm{tr}^2\{\mathrm{F}\mathcal{G}_{\mathrm{p}}(\mathrm{x})\}\mathrm{dx}\,. \quad (23)$$

As we can see from eq. 21, the computational complexity of bandwidth estimation is at least O(N^2) since $R(p,\mathbf{F})$ involves matrix multiplication of N by N, which is not efficient for large datasets. Furthermore, we have to store all data points for model update with new arriving data points. We are interested in maintaining a low complexity model of $\hat{p}_{\mathrm{KDE}}(\mathbf{x})$. This can be achieved by compressing the data points with Gaussian mixtures $\{N\{\boldsymbol{\mu}_j,\mathbf{\Sigma}_j\}\}_{j=1:M}$and derive the corresponding kernel bandwidth selection based on the Gaussian mixtures.

The sample distribution of the compressed data points and kernel density estimation is

$$\hat{p}_{\mathrm{s}}(\mathbf{x}) = \sum_{j=1}^{M} \alpha_j \phi_{\boldsymbol{\Sigma}_j}(\mathbf{x} - \boldsymbol{\mu}_j), \tag{24}$$

then and the density function can be defined as

$$\hat{p}_{\mathrm{KDE}}(\mathbf{x}) = \phi_{\mathbf{H}}(\mathbf{x}) * \hat{p}_{\mathrm{s}}(\mathbf{x}) = \sum_{j=1}^{M} \alpha_j \phi_{\mathbf{H}+\boldsymbol{\Sigma}_{\mathrm{j}}}(\mathbf{x} - \boldsymbol{\mu}_j). \tag{25}$$

Similarly, we have, according to [39],

$$\beta_{\mathrm{opt}} = [d(4\pi)^{d/2} N_\alpha \hat{R}(p, \mathbf{F}, \mathbf{G})]^{-\frac{1}{d+4}}, \tag{26}$$

and

$$\hat{R}(p, \mathbf{F}, \mathbf{G}) = \sum_{i=1}^{M}\sum_{j=1}^{M} \alpha_i \alpha_j \phi_{\mathbf{A}_{ij}^{-1}}(\Delta_{ij}) \times [2\mathrm{tr}\big(\mathbf{F}^2 \mathbf{A}_{ij}^2\big)\big[1 - 2m_{ij}\big] \tag{27}$$

$$+ \mathrm{tr}^2\big(\mathbf{F}\mathbf{A}_{ij}\big)\big[1 - m_{ij}\big]^2\big],$$

where **F** is the structure of bandwidth H, estimated by structure of covariance matrix as

$$\mathbf{F} = \hat{\boldsymbol{\Sigma}}_{\mathbf{smp}}. \tag{28}$$

G is the pilot bandwidth estimated by the multivariate normal scale rule [65] as

$$\mathbf{G} = \hat{\boldsymbol{\Sigma}}_{\mathrm{smp}} \left(\frac{4}{(d+2)N_\alpha}\right)^{2/(d+4)}. \tag{29}$$

$\mathbf{A}_{ij}, \Delta_{ij}, m_{ij}$ are related to Gaussian mixture model $\mathrm{N}\{\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j\}$ and pilot bandwidth $\mathbf{G}$, defined as respectively

$$\mathbf{A}_{ij} = (\boldsymbol{\Sigma}_{\mathrm{g}i} + \boldsymbol{\Sigma}_j)^{-1}, \boldsymbol{\Sigma}_{\mathrm{g}j} = \mathbf{G} + \boldsymbol{\Sigma}_j,$$

$$\Delta_{ij} = \boldsymbol{\mu}_i - \boldsymbol{\mu}_j, \tag{30}$$

$$m_{ij} = \Delta_{ij}^T \mathbf{A}_{ij} \Delta_{ij}.$$

3.2.2. Online kernel density estimation

The issue now is how we can maintain the Gaussian mixture model with new arriving data points. We want *M* in the compressed model as small as possible within a predefined local approximation error, which is defined as the Hellinger distance [66]

between two KDEs before and after compression. The problem be approached with the "revitalization-compression" updating scheme [39] that control the approximation error of data compression to a predefined local density approximation error D_{th}. With the new arriving data point, the optimal bandwidth is re-estimated. With the new kernel bandwidth, each Gaussian component $\alpha_j \phi_{\mathbf{\Sigma}_j}(\mathbf{x} - \boldsymbol{\mu}_j)$ is re-evaluated against its detailed model $q_j(x)$ to verify whether the approximation error exceeds D_{th}. The detailed model $q_j(x)$ is a two component mixture that is initialized by the splitting along the major axis of$\alpha_j \phi_{\mathbf{\Sigma}_j}(\mathbf{x} - \boldsymbol{\mu}_j)$. Those Gaussian components whose approximation error exceeds D_{th} are replaced by their detailed two component models. The revitalization process goes on until the approximation error of all components is below D_{th}. Then the compression of the components from the revitalization is undertaken by a hierarchical clustering of the Gaussian components using Goldberger's K-means [57]. In the beginning, the cluster is initialized to contain all components. Then the cluster that has the largest approximation error is spitted into two clusters by setting K as two in Goldberger's K-means. The process is iterated until the approximation errors of all components fall below D_{th}.

Suppose the model of observed sample is denoted as $\mathbf{S_{model}} = \{\hat{p}_s(\mathbf{x}), \{\hat{q}_i(\mathbf{x})\}_{i=1:N'}\}$, where $\hat{q}_i(\mathbf{x})$ is the detailed two component model for *i*-th component in $\hat{p}_s(\mathbf{x}) : \alpha_i \phi_{\mathbf{\Sigma}_i}(\mathbf{x} - \boldsymbol{\mu}_i)$, when $\mathbf{\Sigma}_i \neq 0$ and

$$\hat{q}_i(\mathbf{x}) = \sum_{k=1}^{2} \alpha_k \phi_{\mathbf{\Sigma}_k}(\mathbf{x} - \boldsymbol{\mu}_k), \tag{31}$$

where

$$\boldsymbol{\mu}_1 = \mathbf{FM} + \boldsymbol{\mu}_i, \boldsymbol{\mu}_2 = \mathbf{FM} - \boldsymbol{\mu}_i, \tag{32}$$

$$\boldsymbol{\Sigma}_k = \mathbf{FCF}^{\mathrm{T}}, \alpha_k = \frac{1}{2}\alpha_i,$$

where $\mathbf{UDU}^{\mathrm{T}} = \boldsymbol{\Sigma}_i, \mathbf{F} = \mathbf{U}\sqrt{\mathbf{D}}, \mathbf{M} = [0.5, \mathbf{0}_{1\times(d-1)}]^{\mathrm{T}}, \mathbf{C} = diag\left(\left[\frac{3}{4}, \mathbf{0}_{1\times(d-1)}\right]\right)$.

The compression process seeks to identify the clustering assignment $\Xi(M) = \{\pi_i\}_{i=1:M}$ where M is the number of clusters, π_i is the set of component indexes in cluster i. The sample distribution of cluster i is

$$\hat{p}_{\mathrm{s}}(\mathbf{x}; \pi_i) = \sum_{j\in\pi_i} \alpha_j \phi_{\boldsymbol{\Sigma}_j}(\mathbf{x} - \boldsymbol{\mu}_{\mathrm{j}}). \tag{33}$$

Let $\hat{p}_{\mathrm{s}}(\mathbf{x}; \pi_i)$ be approximated with one single Gaussian [67] as

$$\hat{p}'_s(\mathbf{x}; \pi_i) = \hat{\alpha}_i \phi_{\hat{\boldsymbol{\Sigma}}_{\mathrm{i}}}(\mathbf{x} - \hat{\boldsymbol{\mu}}_{\mathrm{i}}), \tag{34}$$

where

$$\begin{aligned} \hat{\alpha}_i &= \sum_{j\in\pi_i} \alpha_j, \\ \hat{\boldsymbol{\mu}}_{\mathrm{i}} &= \hat{\alpha}_i^{-1} \sum_{j\in\pi_i} \alpha_j \boldsymbol{\mu}_{\mathrm{j}}, \\ \hat{\boldsymbol{\Sigma}}_{\mathrm{i}} &= \hat{\alpha}_i^{-1} \sum_{j\in\pi_i} \alpha_j (\boldsymbol{\Sigma}_{\mathrm{j}} + \boldsymbol{\mu}_{\mathrm{j}}\boldsymbol{\mu}_{\mathrm{j}}^{\mathrm{T}}) - \hat{\boldsymbol{\mu}}_{\mathrm{i}}\hat{\boldsymbol{\mu}}_{\mathrm{i}}^{\mathrm{T}}. \end{aligned} \tag{35}$$

Then the approximation error for $\hat{p}_{\mathrm{s}}(\mathbf{x}; \pi_i)$ is defined as

$$\hat{E}\left(\hat{p}_{\mathrm{s}}(\mathbf{x}; \pi_j), \mathbf{H}_{\mathrm{opt}}\right) = HD(\hat{p}_{\mathrm{s}}(\mathbf{x}; \pi_i) * \phi_{\mathbf{H}_{\mathrm{opt}}}(\mathbf{x}), \hat{p}'_s(\mathbf{x}; \pi_i) * \phi_{\mathbf{H}_{\mathrm{opt}}}(\mathbf{x})), \tag{36}$$

where HD is the Hellinger distance to quantify the distance between distributions.

With the basic components defined above, the online kernel density estimation algorithm is shown in Table 4 below:

Table 4 Online kernel density estimation

Input: Previous data model: $\mathbf{S}_{\mathbf{model}}^{(t-1)} = \{\hat{p}_s^{(t-1)}(\mathbf{x}), \{\hat{q}_i^{(t-1)}(\mathbf{x})\}_{i=1:M_{t-1}}\}$

New arriving data point: $\mathbf{x}^{(t)}$

D_{th}: the maximum allowed approximation error.

Output: Updated data model: $\mathbf{S}_{\mathbf{model}}^{(t)}$

Procedure:

1. Update previous data model $\tilde{p}_s^{(t)}(\mathbf{x}) = (1 - w_0)\hat{p}_s^{(t-1)}(\mathbf{x}) + w_0\phi_0(\mathbf{x} - \mathbf{x}^{(t)})$, $\{\tilde{q}_i^{(t)}(\mathbf{x})\}_{i=1:\tilde{M}_t} = \{\{\hat{q}_i^{(t-1)}(\mathbf{x})\}_{i=1:M_{t-1}}, \tilde{q}_{\tilde{M}_t}^{(t)}(\mathbf{x})\}$.
2. Estimate current optimal bandwidth $\mathbf{H}_{\text{opt}} = \beta_{\text{opt}}{}^{\mathbf{2}}\hat{\Sigma}_{\text{smp}}$ according to Section 3.2.1.
3. Revitalize the *i*-th component in $\tilde{p}_s^{(t)}(\mathbf{x})$ for which the approximation error $\hat{E}\left(\tilde{q}_i^{(t)}(\mathbf{x}), \mathbf{H}_{\text{opt}}\right) > D_{th}$ and update $\{\tilde{p}_s^{(t)}(\mathbf{x}), \{\tilde{q}_i^{(t)}(\mathbf{x})\}_{i=1:\tilde{M}_t}\}$ until the approximation errors of all components in $\tilde{p}_s^{(t)}(\mathbf{x})$ are small than D_{th}.
4. Compress $\tilde{p}_s^{(t)}(\mathbf{x})$ in a hierarchical way:

   Cluster initialization:

   $$\Xi\left(M^{(t)}\right) = \{\pi_1\}, \pi_1 = \{1, \dots, \tilde{M}_t\}, M^{(t)} = 1$$

   While $\max_{\boldsymbol{\pi}_j \in \Xi\left(M^{(t)}\right)} \hat{E}\left(\tilde{p}_s^{(t)}(\mathbf{x}; \pi_j), \mathbf{H}_{\text{opt}}\right) > D_{th}$:

   - Select the cluster with the maximum local error:

     $$\pi_j = \underset{\boldsymbol{\pi}_j \in \Xi(M)}{argmax}\hat{E}(\tilde{p}_s^{(t)}(\mathbf{x}; \pi_j))$$

   - Split the sub-mixture $\tilde{p}_s^{(t)}(\mathbf{x}; \pi_j)$ into two sets using Goldberger's K-means [57]:

     $$\pi_j \to \{\pi_{j1}, \pi_{j2}\}$$

   - Update $\Xi\left(M^{(t)}\right)$:

     $$\left\{\left\{\Xi\left(M^{(t)}\right) \backslash \pi_j\right\}, \pi_{j1}, \pi_{j2}\right\} \to \Xi\left(M^{(t)}\right), M^{(t)} + 1 \to M^{(t)}$$

End while

5. Regroup the components of $\hat{p}_s^{(t)}(\mathbf{x})$ according to $\Xi(\mathrm{M}^{(t)})$ and construct compressed sample distribution $\hat{p}_s^{(t-1)}(\mathbf{x})$ according to eq. 33 and 34.

3.2.3. Density normalized trend representation

Suppose the final Gaussian mixture model for the sample distribution is $\hat{p}_{\mathrm{s}}(\mathbf{x}) = \sum_{i=1}^{M} \alpha_i \phi_{\mathbf{\Sigma}_{\mathrm{i}}}(\mathbf{x} - \boldsymbol{\mu}_{\mathrm{i}})$. For every data point $\mathbf{x}^{(j)}$, we have

$$\hat{p}_{\mathrm{s}}(\mathbf{x}^{(j)}) = \sum_{i=1}^{M} \alpha_i^{(j)} \phi_{\mathbf{\Sigma}_{\mathrm{i}}}(\mathbf{x}^{(j)} - \boldsymbol{\mu}_{\mathrm{i}}). \quad (37)$$

The data points lying in the dense regions tend to have higher values for $\alpha_i^{(j)}$. Since we are more interested in structures rather than densities, we normalize the coefficients $\alpha_i^{(j)}$ so that $\sum_{i=1}^{M} \alpha_i^{(j)} = 1$. Each data point is labeled as $\hat{l}_j = \underset{\mathrm{i}}{\operatorname{argmax}}\, \alpha_i^{(j)}, 1 \le i \le M$, assigned to Gaussian component $G(\hat{l}_j)$. The corresponding largest value is denoted as $\alpha_{\max}^{(j)}$. Since the overlap among Gaussian components is small, usually $\alpha_{\max}^{(j)}$ is significant larger than other coefficients.

We can then visualize the data in a MST-ordered heatmap representation, similar to MST-ordered heatmap introduced in Section 3.1.4. Since we can expect millions of data points, we use columns for data points and rows for features for convenience. The order of data points is determined by two orders. The first order is the MST depth-first transversal order of the centers of the Gaussian components, initialized from one end of the longest path of the MST. It determines the order of the Gaussian component. For each Gaussian component $G(i), 1 \le i \le M$, the order of data points $\mathbf{x}^{(j)}, j \in G(i)$, is the

decreasing order of the largest coefficients $\alpha_{\max}^{(j)}$.

The heatmap representation can be generated in an online way along with the density approximation with Gaussian mixtures, which is useful for immediate data inspection.

4. Algorithm Validation Result

4.1. Implementation and Availability

We developed a multi-threaded C++ implementation of this algorithm (source code is provided, and a pseudo-code summary is provided in Appendix A) and applied it to diverse synthetic and real-world datasets. We focused on datasets for which independent information about the actual trends is available for validating the automatically detected trends. The independent data was not used as input to our algorithm. We examined whether our algorithm can discover these verifiable trends and whether trend-relevant feature selection assists in visualization for trend discovery from a practical standpoint.

There are mainly two parameters for running the algorithm: the parameter k for the k-NNG for computing the neighborhood similarity; and γ for computing the similarity threshold. We set $k = 4$ and $\gamma = 0.01$ and kept them the same across all the datasets in the experiments described here. Varying k does not significantly affect the results because the automatically estimated thresholds adapt to these changes. For example, Figure 11(b) shows the effect of varying k on the discovered subspace trends in terms of its connection accuracy for the microarray dataset. The value of $\gamma = 0.01$ was chosen empirically keeping in mind the extent of the rapid decrease in $|R_{Z1}(T) - R_{Z2}(T)|$. Too high a value of this parameter makes the algorithm conservatively miss potentially trend-relevant features, and vice versa. To speed up the analysis of high-dimensional datasets, we introduced an additional pre-processing parameter σ that is used to first group linearly correlated features using an agglomerative clustering method and then calculate the neighborhood similarity matrix for the generated feature modules. Basically, we set σ as high as 0.95 or 1 (no clustering) for datasets with only a few hundred features, and ~0.8

for the datasets with tens of thousands of features so as to reduce the size of the generated feature modules. We experimented with various σ and L values on the dataset and found the resulting accuracy to be comparable, as shown in Figure 11 (a) and (c).

The overall computational complexity of our algorithm depends on the complexities of the three main parts: 1) initial agglomerative clustering: $O(M^2Nh(\sigma))$, where $h(\sigma)$ denotes the height of the agglomerative tree. Larger threshold σ gets smaller $h(\sigma)$; 2) Neighborhood similarity matrix calculation:$O(M'^2N^2B^3)$, where M' is the number of feature clusters remained after the agglomerative clustering, $O(B^3)$ is the complexity to calculate earth mover's distance. Since we set $B = 20$, its computational cost is negligible; 3) Automatic thresholding: $O(M'^2)$. With our multi-threaded implementation, the relevant features can be identified within seconds on contemporary desktop computers. It took ~2 seconds to analyze the synthetic dataset and the cell cycle and mitosis morphology datasets, and ~10 seconds for the B cell dataset that contains more than 10,000 features.

In the following sections, we studied how the neighborhood similarity measure performs on common associations between features, and how the proposed algorithm works on a synthetic dataset with embedded subspace trends. Then we applied it to three real-world datasets and quantitatively verified the detected trends by the above-mentioned method, and examined the impact of trend-relevant feature selection on data visualization. The open-sourced software and some example data sets can be downloaded at http://www.farsight-toolkit.org/wiki/Subspace_Trend_Discovery.

4.2. Neighborhood Similarity Measure on Synthetic Associations

To obtain a basic understanding of the proposed neighborhood similarity metric described in Section 4.1.2, we first applied it to several simulated datasets, with a variety of underlying linear and nonlinear association patterns, including linear, parabolic, cubic, sinusoidal, exponential, circular, and random. We also compared the neighborhood similarity with a few alternative metrics, such as Pearson's correlation, distance correlation [68], normalized mutual information ($I(X;Y)/\min[H(X),H(Y)]$), and maximal information coefficient (MIC) [69]. For each form of association, we generated 320 simulated data points. As shown in Table 5, the neighborhood similarity values reach 1 for most of the associations, whereas the other correlation measures (PC, NMI, DC) are much lower. The maximal information coefficient has comparable performance for the simulated data since it is specifically designed to capture a wide range of associations. The neighborhood similarity for the circular association is relatively low because a neighboring relationship in neither axis implies a neighboring relationship in the other.

We next studied the robustness of the neighborhood similarity measure by adding zero-mean Gaussian noise to the simulated data points, following [70] as shown in Figure 5. All the correlation measurements decrease gradually with increasing noise, down to the measured value for the random dataset. The neighborhood similarity measure (red line) exhibits a more consistent behavior, starting off at 1 for the noise-free case, and gradually approaching 0 as the noise level increases. In contrast, the numerical ranges for other metrics (PC, NMI, DC) vary across the different associations. MIC's performance is roughly comparable to NS and it converges to approximately 0.2 as the noise level increases, possibly due to the lack of enough data points while NS would converge to 0.

Furthermore, NS does not involve grid search or dynamic programming as MIC does and therefore demands less.

Table 5 Evaluation of the neighborhood similarity on synthetic associations

Correlation Measurement	Linear	Parabolic	Cubic	Sinusoid (f = 2)	Sinusoid (f = 8)	$X^{1/4}$	Circle	Exponential	Random
Neighborhood Similarity (NS)	1.00	1.00	1.00	0.99	0.87	1.00	0.30	0.99	0.00
Pearson Correlation (PC)	1.00	0.01	0.25	0.15	0.01	0.89	0.00	0.74	0.00
Normalized Mutual Information (NMI)	1.00	0.70	0.75	0.56	0.30	1.00	0.66	1.00	0.22
Distance Correlation (DC)	1.00	0.50	0.69	0.43	0.13	0.98	0.17	0.91	0.09
Maximal Information Coefficient (MIC)	1.00	1.00	1.00	1.00	0.90	1.00	0.63	0.99	0.18

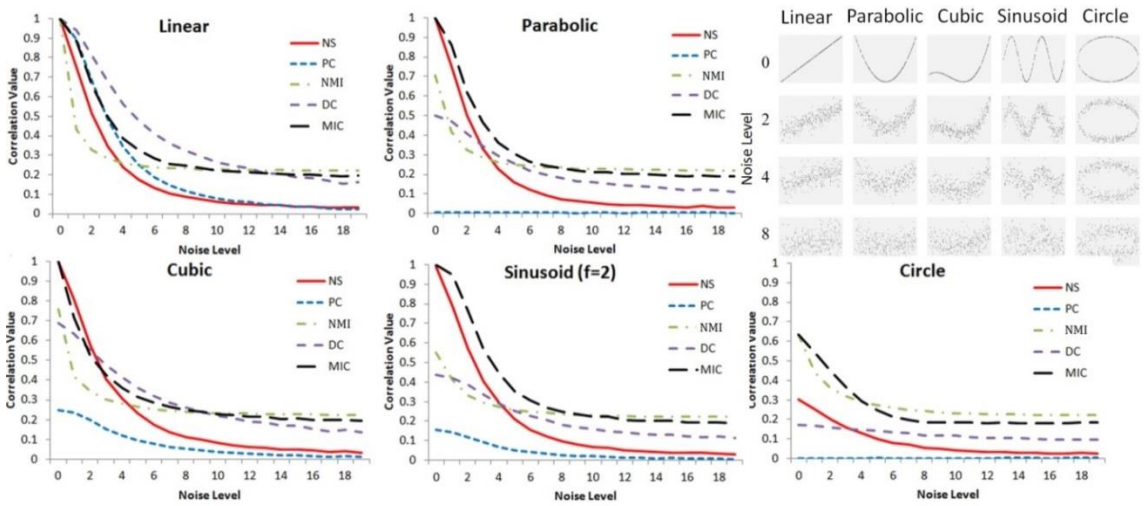

Figure 5. Synthetic data experiments designed to evaluate the neighborhood similarity measure's (NS) ability to capture various associations with added Gaussian noise, in comparison with PC, NMI DC and MIC.

4.3. Synthetic 10-D Dataset with Two Subspace Trends Embedded

We illustrate the algorithm process with this synthetic dataset consists of 1,000 data points in a 10-dimensional space $\{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_{10}}\}$ with two subspace trends embedded (Trend I & Trend II, respectively). Trend I is embedded in the subspace formed by features $\mathbf{x_1}$ … $\mathbf{x_4}$ and Trend II is embedded in another subspace formed by features $\mathbf{x_5}$

… $\mathbf{x_7}$. The remaining features $\mathbf{x_8}$ … $\mathbf{x_{10}}$ are simulated random noise. The simulated data in dimensions $\mathbf{x_1}$ … $\mathbf{x_4}$ representing Trend I are functions of a common driving parameter $\boldsymbol{\lambda}$, where $\mathbf{x_1}$ is a linear function of $\boldsymbol{\lambda}$, $\mathbf{x_2}$ a quadratic function, $\mathbf{x_3}$ a sine function and $\mathbf{x_4}$ a cosine function. Mathematically, this is written as

$$[\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3}, \mathbf{x_4}] = [\boldsymbol{\lambda},\ 4(\boldsymbol{\lambda} - 0.5)^2, \sin(2\pi\,\boldsymbol{\lambda}), \cos(2\pi\,\boldsymbol{\lambda})],$$

where $\boldsymbol{\lambda} \in \mathbf{R^{1000}}$ and $\lambda^{(i)} \sim unif(0,1)$. The simulated data representing Trend II was generated by two intersecting curves in the subspace of $\mathbf{x_5}$ and $\mathbf{x_6}$, and in the subspace of $\mathbf{x_5}$ and $\mathbf{x_7}$, as described as

$$[\mathbf{x_5}, \mathbf{x_6}, \mathbf{x_7}] = \begin{bmatrix} \boldsymbol{\lambda_1} & \boldsymbol{\lambda_1} & \boldsymbol{\lambda_1} + 5 \\ \boldsymbol{\lambda_2} & -\boldsymbol{\lambda_2} + 1 & 80\left(\boldsymbol{\lambda_2} - \frac{1}{3}\right)^3 - 12(\boldsymbol{\lambda_2} - \frac{1}{3}) \end{bmatrix},$$

where $\mathbf{x_1}, \mathbf{x_2} \in \mathbf{R^{500}}$ and $\lambda_1^{(i)}, \lambda_2^{(i)} \sim unif(0,1)$. Figure 6 shows the results produced by several existing projection based visualization algorithms applied to this dataset. A visualization based on the top three principal components is shown in Figure 6(a), where we can only vaguely discern the embedded circular trend in the data. The results from three other visualization algorithms (LLE, ISOMAP, and t-SNE) are shown in Figure 6(b – d), respectively. These methods aim to preserve the global or local manifolds, but fail to reveal the subspace trends. This is because the simulated data contains two independent trends in two separate subspaces. When the visualization algorithms are applied to all the dimensions together, the two independent trends inter-mix, and the added random noise further obscures them.

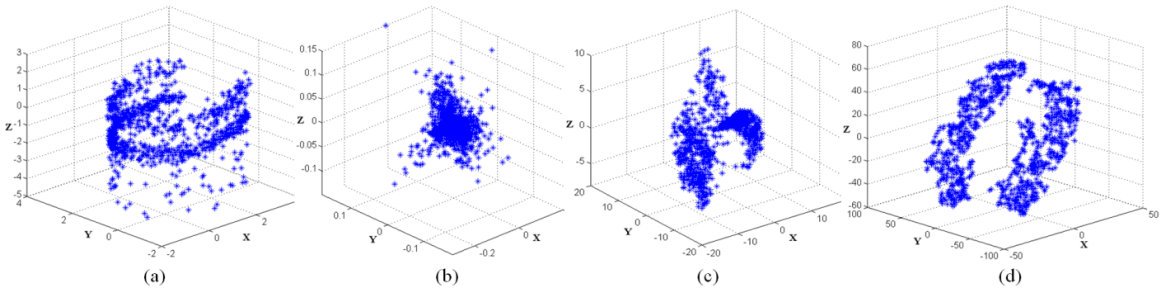

Figure 6. Projection based visualization of the synthetic 10-D dataset without trend-relevant feature selection fails to reveal the embedded subspace trends: (a) using the top three PCA components; (b) LLE (k = 6); (c) ISOMAP (k = 6); (d) t-SNE (perplexity = 6, smooth measure of k).

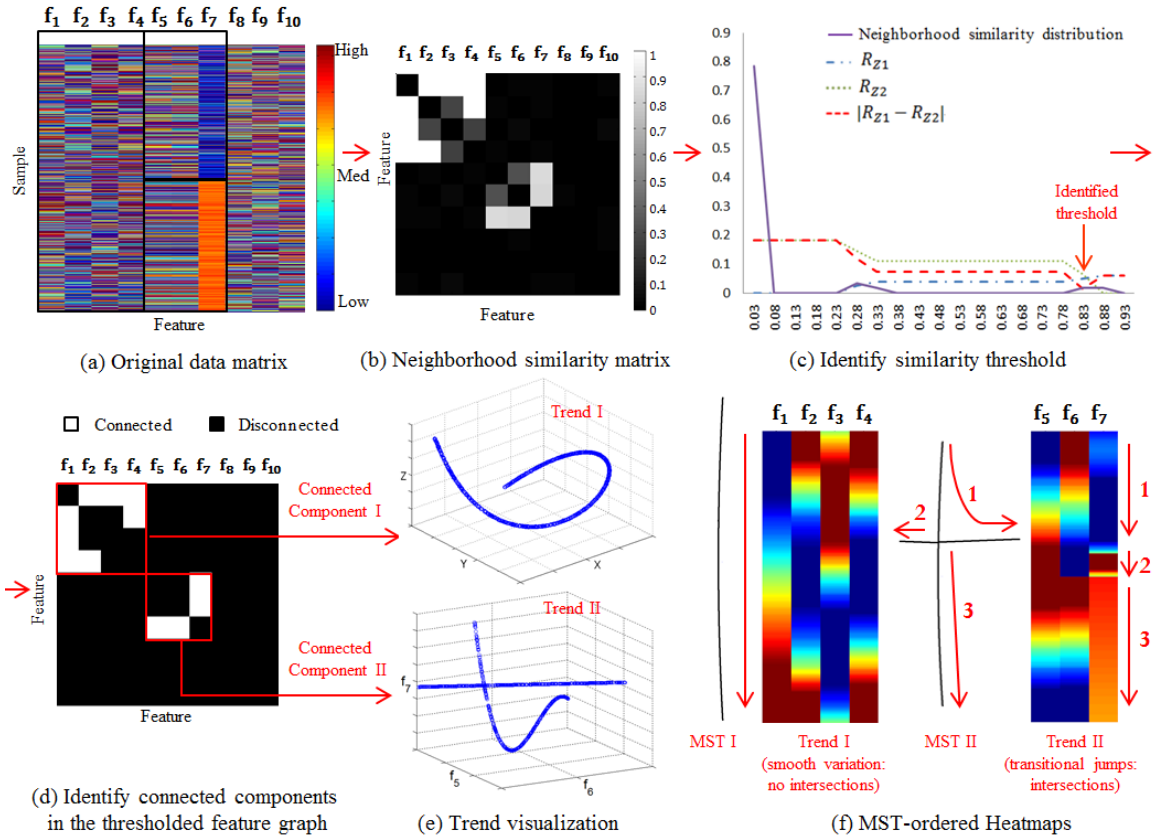

Figure 7. Illustrating the main steps to automated subspace trend analysis from (a)-(f).

Figure 7 illustrates the steps in our neighborhood similarity based analysis. Figure 7(a) shows the simulated data using a heatmap representation in which each row corresponds to one data point, and each column represents one feature dimension. Figure 7(b) shows the pair-wise neighborhood similarity matrix (with values in the range 0 to 1) displayed as a grayscale map. Figure 7(c) shows the results of the modified Shanbhag

algorithm applied to the neighborhood similarity matrix. The distribution of neighborhood similarity values is displayed in blue, and the red dotted line is the information measure used to identify the similarity threshold. It is used to partition the feature graph into two prominent connected components (I and II). These two components correctly and clearly identified the two synthetic subspace trends I and II, and the random data containing features are not selected. Figure 7(e) displays the two detected subspace trends using the top three PCA components. Finally, Figure 7(f) shows two MSTs constructed for the two detected subspace trends: the MST for Trend I looks like a chain, the one for Trend II has branches. The MSTs are visualized together with the MST-ordered heatmaps, as described in Section 3.1.5. These heatmaps show the concordant gradual changes that the neighborhood similarity metric is designed to capture. The abrupt changes (jumps) in the heatmap for Trend II are due to branch(es) in the MST as visualized in Figure 7(f).

4.4. Microarray Gene Expression Data

4.4.1. Cell-cycle time-series microarray data

We next applied the proposed neighborhood similarity based analysis to a cell-cycle time-series dataset [71]. In this dataset, gene microarray technology was used to capture snapshots of gene expression activity at 17 time points during one cell cycle of a human cancer cell line. At each time point, expression levels of 3,169 genes were measured. Therefore, this data set contains 17 data points and 3,169 features per data point. We analyzed this dataset to see whether the neighborhood similarity based approach can identify the genes relevant to the cell cycle, and recover the temporal ordering of the samples without being provided any prior information. The input to our

algorithm was a matrix of 3,196 genes across 17 unordered samples. If we visualize the samples with all genes (features) using t-SNE in Figure 8(a), we cannot observe any indication of ordering among the samples and the connection accuracy is as low as 0.31 with all the genes. With the selected genes, we successfully revealed the trend regarding the temporal ordering and achieved the connection accuracy of 1.

To reduce the computational complexity of calculating pair-wise neighborhood similarities for 3,196 genes, we applied an agglomerative algorithm [61] to cluster the genes into 216 feature modules of linearly correlated genes ($\sigma = 0.8$). The similarity between two feature modules is the average Pearson's correlation between the feature vectors of one module with the average feature vector of the other. Then the neighborhood similarity matrix is generated for these feature modules as shown in Figure 8(c). Based on the distribution of similarity values, we computed a similarity threshold of 0.61 using the modified Shanbhag thresholding approach. After applying the threshold, we identified a group of 17 feature modules that share high neighborhood similarity values. These 17 modules contain 661 genes. Then we undertook the iterative process to evaluate and tune the gene modules as mentioned in Section 3.1.4 so that the scores of selected features are higher than the unselected ones. One gene module was excluded from this process, which ends up with 656 genes selected. We used t-SNE to visualize the 17 samples based on those 656 genes and noticed a clear trend which is consistent with the known temporal order of the samples Figure 8(b). To examine the gradual variation of the selected features, we displayed the MST-ordered heatmap in Figure 8(d), where each row represents a sample, and each column corresponds to a gene. We found that the selected genes show clear concordant gradual variations while the unselected genes do

not show any such pattern. The variation patterns of a few selected genes are shown in Figure 9. The genes are evaluated comparing with the trend with the scores shown in Figure 10.

We checked whether the modified Shanbag method is effective in selecting the right similarity threshold to reveal the trend. We calculated the connection accuracy $A^1_{\hat{G},l}(\hat{G})$ for a range of possible similarity thresholds by varying σ from 0.8 to 1 and k from 2 to 4. The features from the largest connected component after applying a chosen threshold are selected. The identified similarity threshold by the modified Shanbag method lies in the optimal or suboptimal interval of high connection accuracy, as indicated by the arrows in Figure 11(a) and (b). Additionally, we have varied L from 100-400 and found that the identified threshold is unchanged and the connection accuracy stays at a high level when L is larger than 200, as shown in Figure 11(c). For varying d in calculating connection accuracy as shown in Figure 11(d), we noticed that the connection accuracy reaches above 0.9 when d equals 4 for the trend built on all the genes, however it stays 1 with our unsupervised feature selection even when only a small portion of genes are selected.

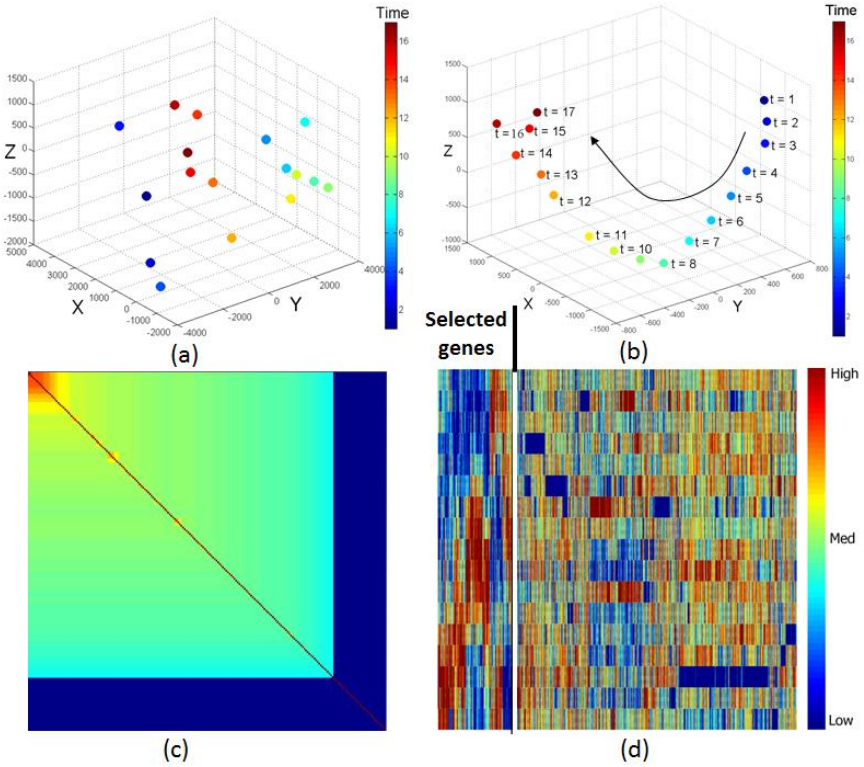

Figure 8. Subspace trend discovery correctly reveals the temporal order of cells from cell-cycle time time-series microarray data. Visualization using t-SNE with all genes (a), with the selected genes (b); (c) Neighborhood similarity matrix; (d) MST-ordered heatmap.

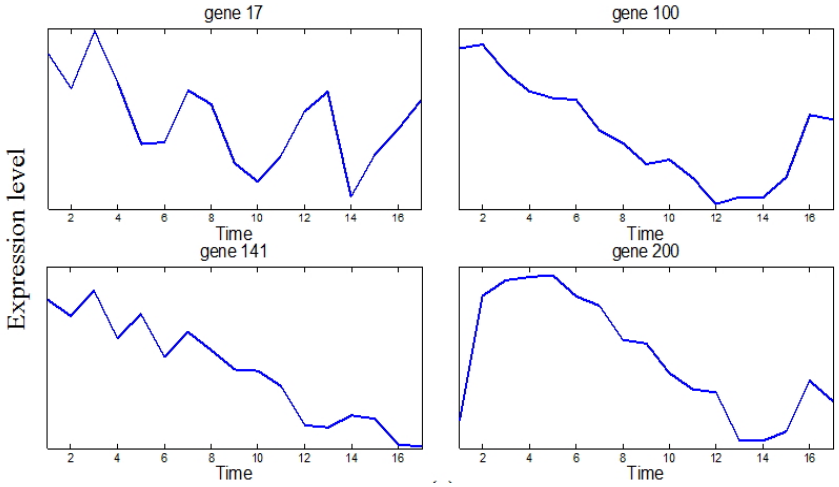

Figure 9. Variation patterns of selected genes along the temporal order. Gene 17 exhibits a sine wave variation pattern; Gene 100 exhibits a quadratic variation along time; Gene 141 exhibits a linear relationship and gene 200 exhibits third-order variation with two obvious peaks.

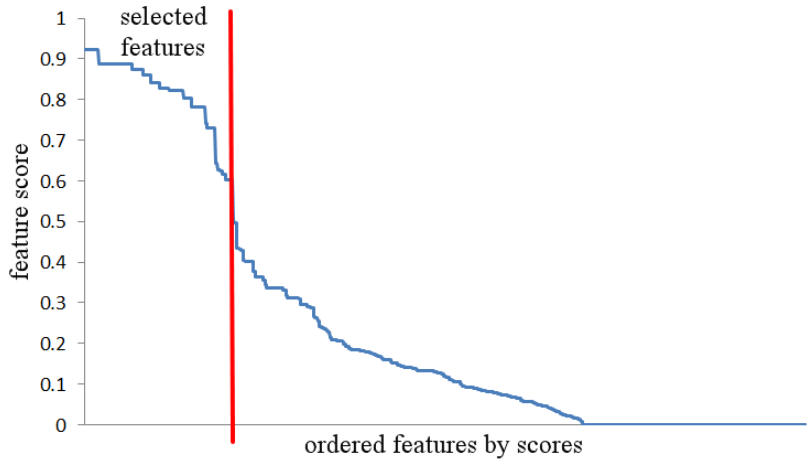

Figure 10. Feature evaluation: features (genes) are ordered descending by scores.

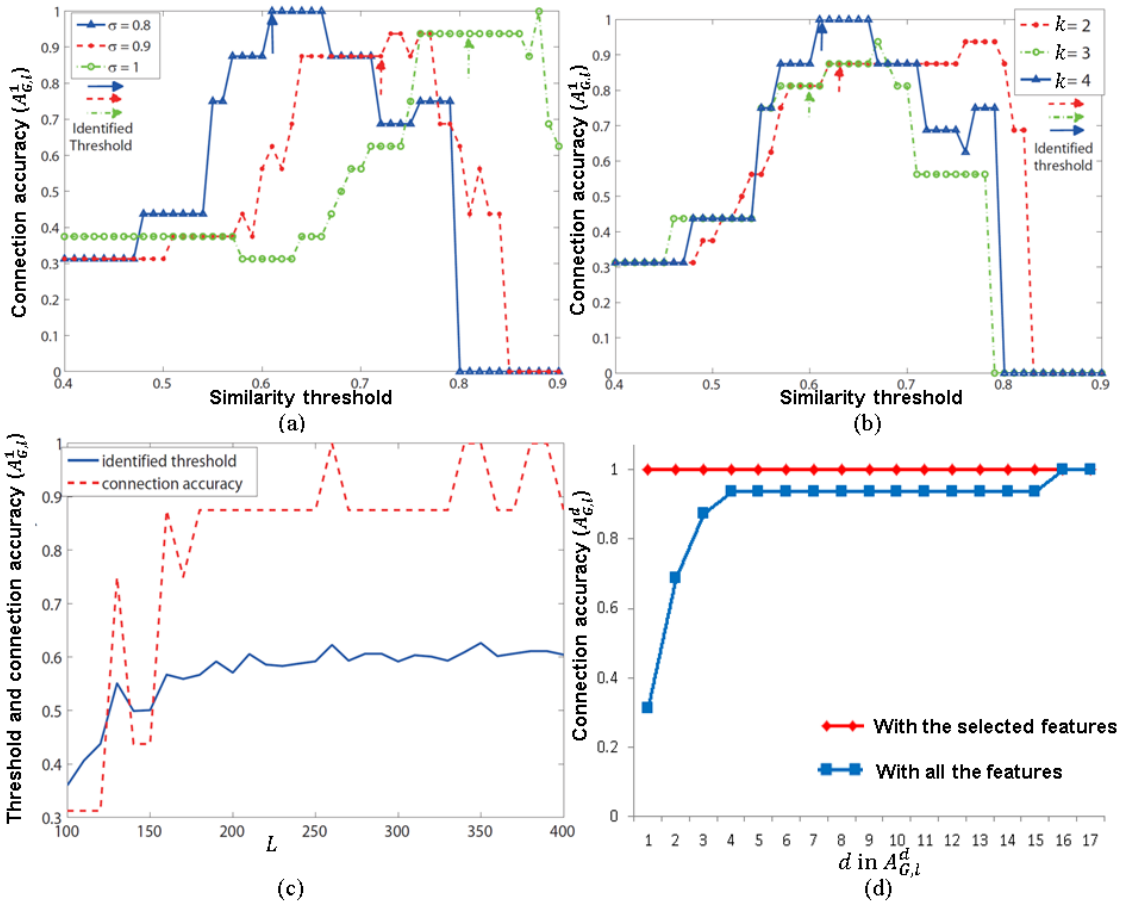

Figure 11. Sensitivity analysis of algorithm parameters. (a)(b) For σ= 0.8, 0.9 & 1 and k = 2, 3, & 4, the automatically identified thresholds (indicated by the colored arrows) lie in the optimal or suboptimal intervals. (c) varying L values; (d) varying d in connection accuracy.

4.4.2. B-cell differentiation microarray data

We applied the proposed approach to a B cell differentiation dataset [72] containing 11,292 genes and 34 samples across 5 normal differentiation stages of stem cells: 6 hematopoietic stem cells (HSC), 6 common lymphoid progenitors (CLP), 6 proB cells, 6 preB cells, 6 Immature B cells (IM), 4 more terminally differentiated B cells (1 naive B cell, 1 centroblast, 1 centrocyte, 1 memory B cell). For the preprocessing, we removed the genes with more than 10% missing values, after which 10,077 genes remained. Missing values for the remaining genes were computationally imputed using the *k*-NN algorithm [73]. This dataset has multiple samples at each differentiation stage. We examined if our approach can group the samples at common differentiation stages together, and uncover the biologically correct ordering of B-cell differentiation: HSC, CLP, proB, preB, IM, naiveB/CB/CC/ memoryB).

We first clustered the genes into 1,291 gene modules using the Pearson correlation based agglomerative clustering ($\sigma = 0.8$). After generating the neighborhood similarity matrix for all gene modules (Figure 12(c)), the similarity threshold was automatically identified as 0.67 by the modified Shanbhag method. The largest connected component of 78 gene modules containing 1,895 genes were identified as being potentially trend-relevant. After iterative feature tuning process, 1,652 genes were finally kept for trend visualization. The trend in the data is so strong that the trend is easy to detect even without feature selection as shown in (Figure 12(a)). However our algorithm is able to sense the trend with fewer genes and make the trend clearer (Figure 12(b)) compared to the original visualization in Figure 12(a). Moreover, by reducing the dimensionality from 11,292 to 1,652, it takes much less time to generate the visualization.

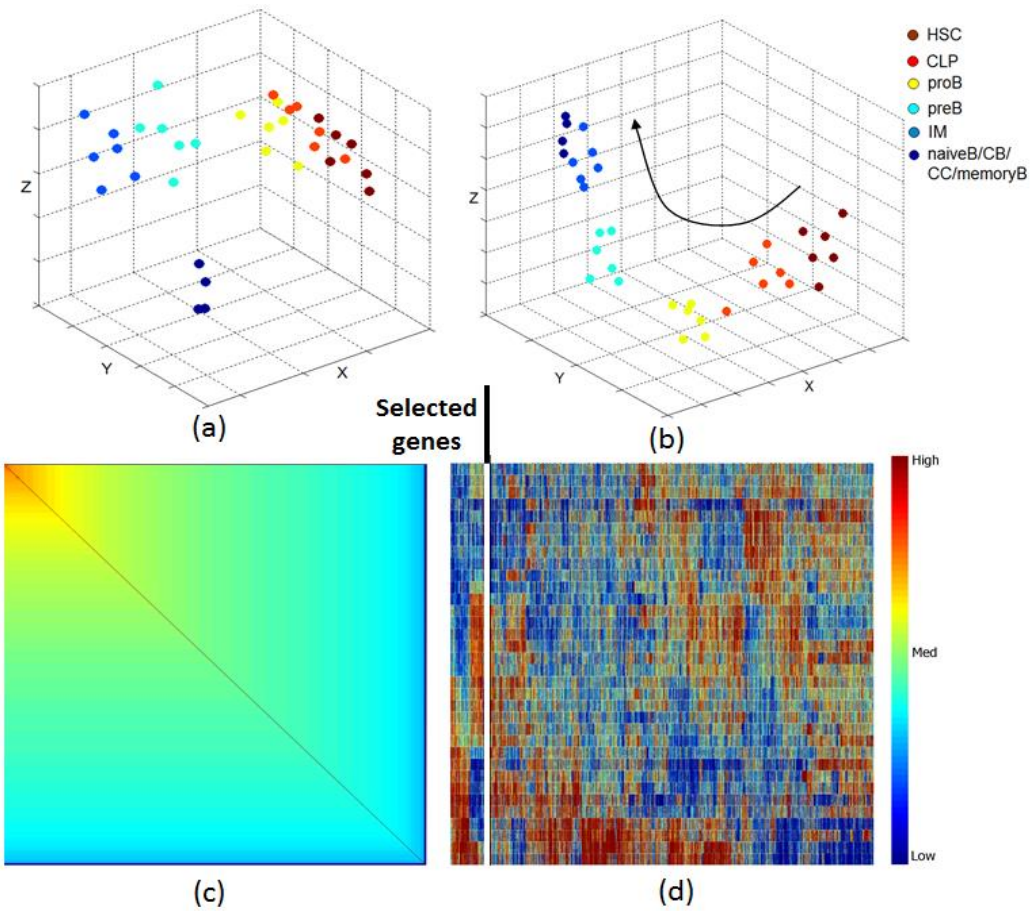

Figure 12. Subspace trend discovery correctly reveals B cell differentiation stages. Visualization with all genes (a); with the selected genes (b); (c) Reordered neighborhood similarity matrix; (d) MST-ordered heatmap showing the selected and unselected genes.

For a range of possible similarity threshold values from 0.3 to 0.8, we found that the connection accuracy $A^1_{G,l}(\hat{G})$ stay relatively high, mostly above 0.9 for the selected features derived from the corresponding largest connected component. However with smaller set of relevant features, we are able to find a smaller feature representation for the progression stages that preserve the trend and make the dimension reduction more efficient with genes reduced.

4.5. Arbor Morphology

We are interested in morphology of neurons and microglia in the brain tissue. The neurons are basic signal processing unit in the brain and microglia is one type of glia cells which protect and support neurons. They all have branching arbor morphology. Usually, the neuron or microglia has a cell body, also called soma, that contains the nuclei and have arbors spreading from the soma. The 3-D arbor morphology can be modeled with 3-D connected cylinders stored in universal format: SWC [44]. In the following section, we will introduce how the arbor morphology can be quantified and how we discovered progression of arbor morphology of neurons.

4.5.1. Quantitative arbor features

We used an extension of the L-measure [74] to extract 35 core measurements such as compartment diameters, segment path lengths and bifurcation parameters, and some derived statistical measurements (average, min, max and total) from these core measurements for each cell, as listed in Figure 13. Other cellular-scale features such as the total number of branch points and the total path length were also generated. To capture the overall shape of each cellular arbor, we also computed features from fitting a minimum-wrapping convex hull and an ellipsoid, such as the total surface area and volume of the convex hull, and the major and minor axis lengths of the ellipsoid. Thus, we generated a list of features at multiple levels of granularity, from the compartment level, branch level, and the whole cell level. In the end, this results in a total of 136 features representing a comprehensive quantification of the arbor for each cell, and used as the input to the subspace trend discovery analysis.

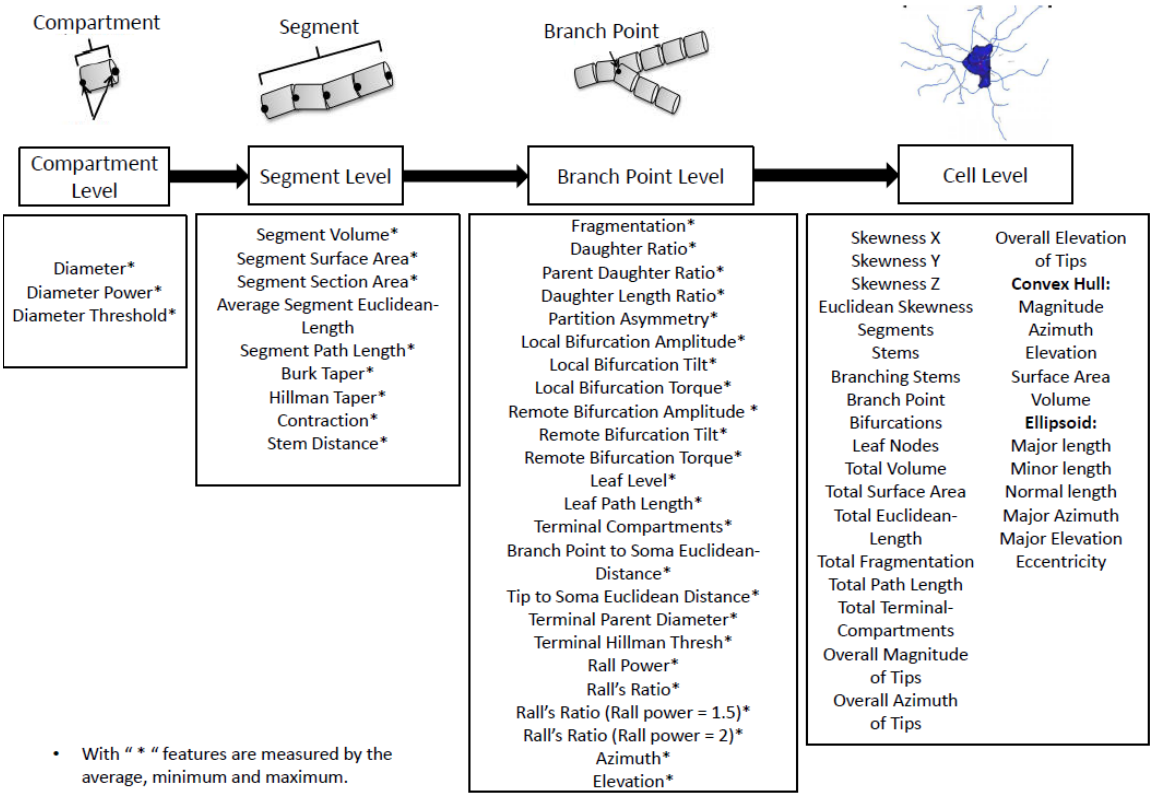

Figure 13. Quantification of arbor morphology at multiple levels. Each cell is composed of a central soma, and arbors, connected series of compartments. Statistical measures such as average, minimum and maximum were extracted from the 35 core features indicated with an asterisk (*).

4.5.2. Artificial DA neurons with varying balancing factor

Using the real Class I and Class IV DA neuron reconstructions as input, the artificial neurons were generated using a global rule based method created by Cuntz [75], where the emergent morphology is determined by the global matrix of branch density produced from all the dendritic trees of a single morphological group, and a factor that balances the costs of wiring length and conduction time. The balancing factor (BF) represents the balance between resource minimization and conduction time minimization. Increasing BF means an increasing influence of conduction time, causing a morphological progression from branched to spoke-like structures. Different balancing factors were applied to generate these sets of artificial neurons (from 0.05 to 1.05). We looked at the emerging morphological patterns.

We loaded all simulated traces of 109 neurons in TraceEditor [76]. The sample traces are shown in Figure 14. We calculated a total of 136 features for all 109 neurons which are the input to the progression analysis. The algorithm automatically selects 30 features and visualizes the hypothesized trend with TreeVis [62], where each tree node corresponds to a neuron. When we colored the tree node according the underlying balancing factor, we found that the neurons at the left end of the tree are mostly generated by low balancing factors while those at the right end of the tree are mostly generated by high balancing factors and in between we saw a gradual transition from low-BF neurons to high-BF neurons in Figure 15(a). The connection accuracy $A^1_{G,l}(\hat{G}) = 0.76$ and $A^2_{G,l}(\hat{G}) = 0.95$. The MST-ordered heatmap along with the selected features are shown in Figure 15(b). The descending-ordered feature scores are shown in Figure 16, where we can see a sharp drop of the scores between the selected features and unselected ones. We saw that when BF is high, some features such as total volume, total path length, average PK Classic are relatively high and average burk taper, average partition asymmetry and average leaf level are relatively low, as shown in Figure 17.

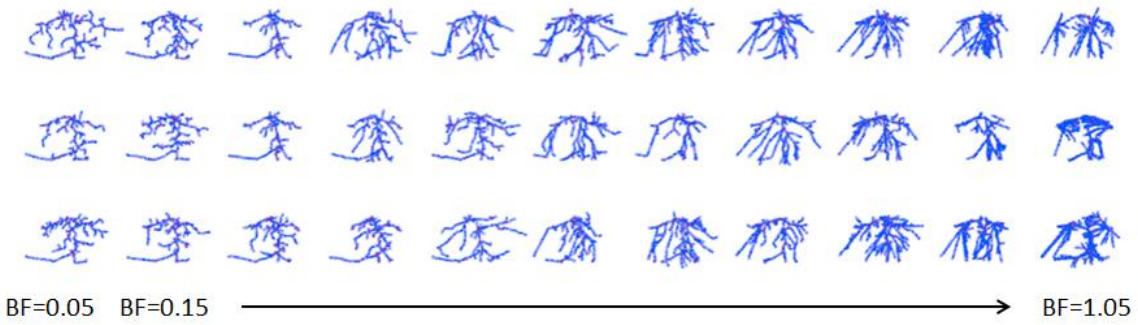

Figure 14. Traces of artificial neurons with varying balancing factor loaded in TraceEditor.

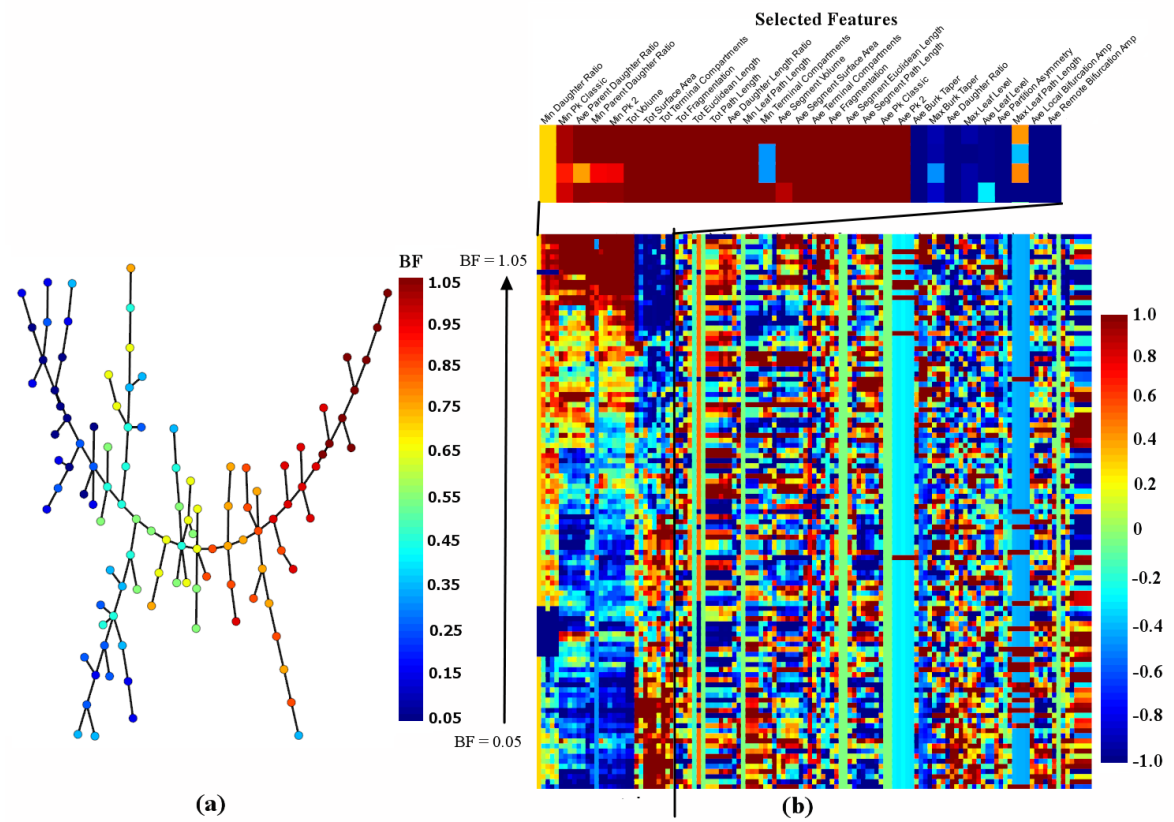

Figure 15. Discovered subspace trend successfully reveals the underlying balancing factor used to simulate the neurons. (a) Tree visualization with selected features; (b) MST-ordered heatmap. The detailed list of selected features is shown above the heatmap.

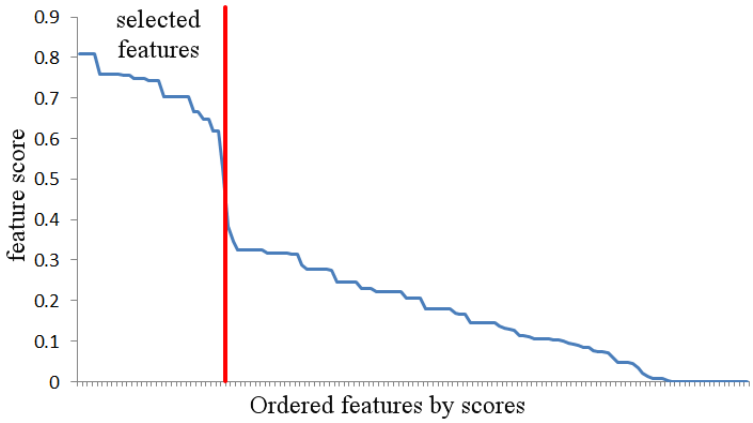

Figure 16. Feature evaluation: features (genes) are ordered descending by scores.

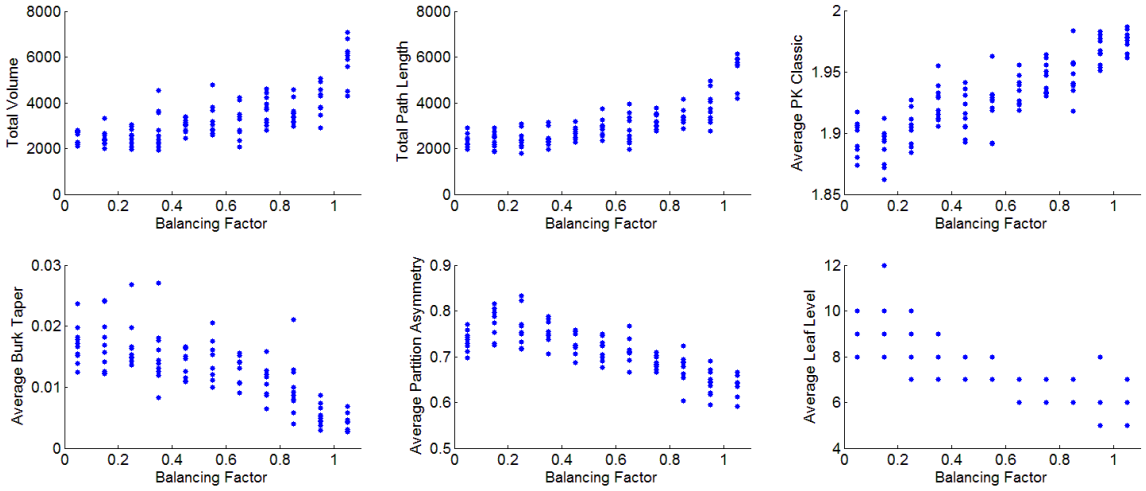

Figure 17. Variations of selected features along balancing factor. Total volume, total path length and total PK classic increases with balancing factor while average burk taper, average partition asymmetry and average leaf level decreases.

4.5.3. Real neuron reconstructions

We have applied subspace trend analysis to real constructions of DA neurons from class I to class IV from NeuroMorpho.org [45]. There are a total of 68 neurons consisting 8 neurons from class I, 18 from class II, 26 from class III and 16 from class IV. The representative neuron reconstruction traces are shown in Fig. 11. A table of 68 neurons with 136 features is input into the progression algorithm. The algorithm automatically selects 52 features and outputs tree visualization in Figure 19(a) and MST-ordered heatmap as shown in Figure 19(b). The tree node is colored according to the class label. The tree visualization reveals a clear progression from class I to class III and IV.

From Figure 19(a), we can see that the neurons from the same class are close in the tree except that four neurons from class III are put close to class II neurons on the tree. When we looked into the four neurons, we found that these four neurons have elongated spread-out shape similar to shape of class II. From MST-ordered heatmap in Figure 19(b), we can clearly identify the arbor features to distinguish the type. For example, we saw that average segment path length usually remains high for class I neurons; the number of

segments and branch points of class I and II neurons are fewer than class III neurons which are fewer than class IV neurons; Class III neurons have higher average leaf levels than other classes; and class IV neurons have the largest features relating to size, such as convex hull surface area, ellipsoid major/minor length and total leaf nodes. More interestingly, we found that even within same class, we saw some differences in feature values. For example in Class IV, we can clearly identify two groups with different diameter related features such as average terminal Hillman thresh and average segment section area.

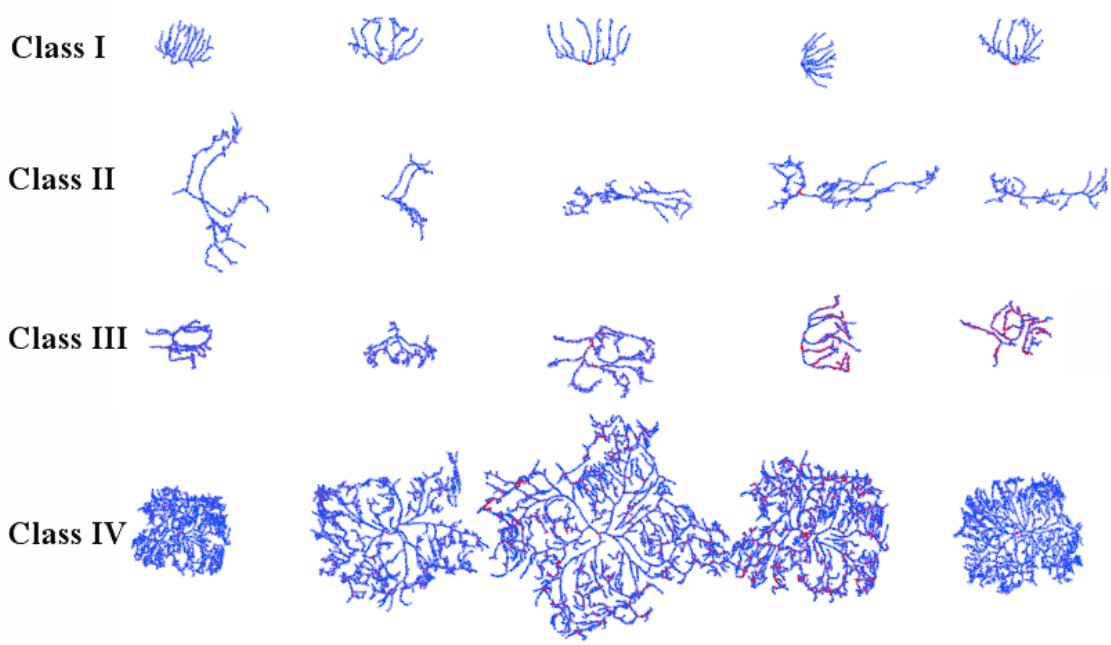

Figure 18. Sample traces of real DA neuron reconstructions from class I, II, III and IV.

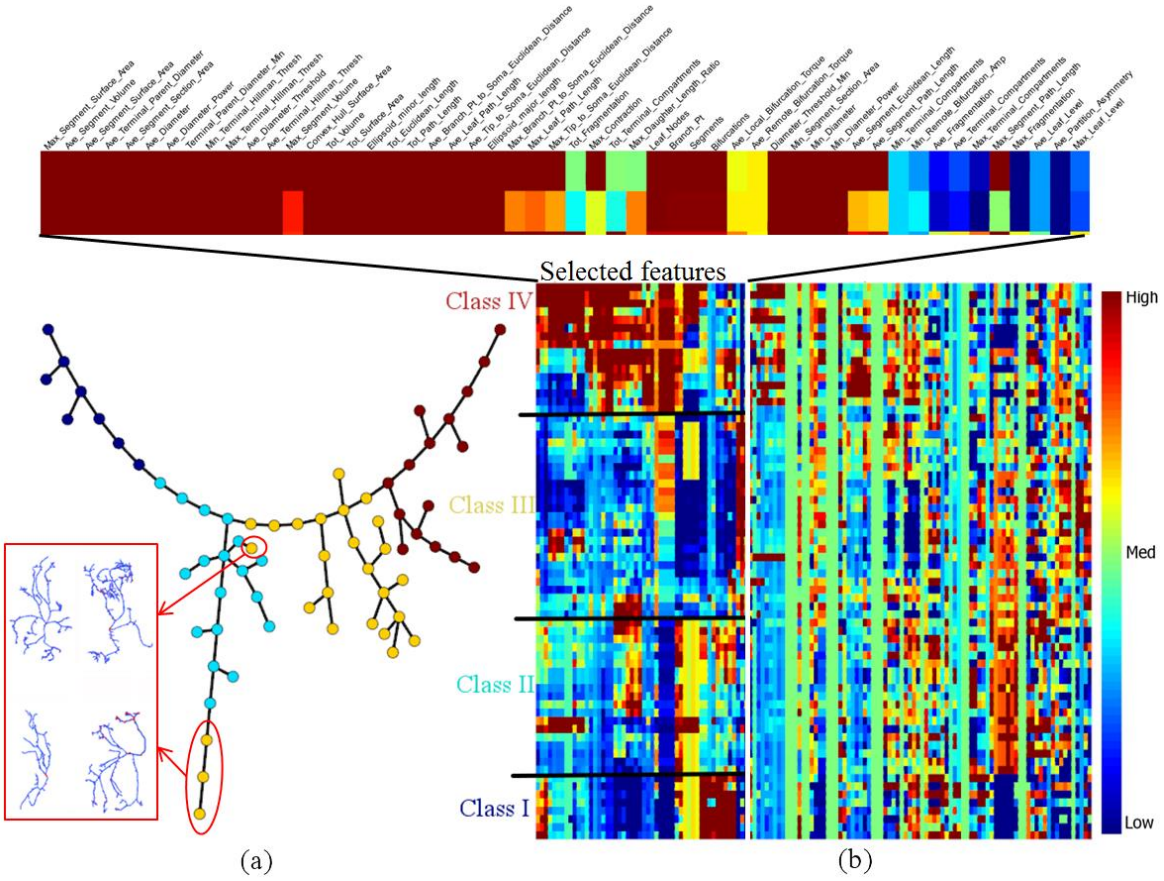

Figure 19. Discovered subspace trend successfully reveals the morphological progression of neurons from Class I to Class III and IV. (a) Tree visualization with selected features; (b) MST-ordered heatmap. The detailed list of selected features is shown above the heatmap.

5. Exploratory Analysis of 3-D Arbor Morphology

5.1. Progression of Microglia Arbor Morphology in Response to Implanted Device

Microglia represent the resident immune system of the brain, and are programmed to respond to a variety of tissue perturbations. Resting microglia contain highly branched and motile arbors that constantly screen their local environment for perturbations. When the tissue is perturbed by pathological stimuli, microglia respond with rapid changes in arbor morphology, and may even migrate toward the lesion site [77]. While many perturbations are localized in nature, some others are spread over large spatial regions. A perturbation of particular interest to us relates to the insertion of a microfabricated neural recording device [78]. Since removing these devices prior to tissue processing and imaging risks losing important cellular information because cells can remain attached to the device, we imaged tissue samples with the device still embedded in it. This way, we enabled a detailed three-dimensional data-driven characterization of microglial activation in response to implanted device at a much larger spatial scale (multiple millimeters, much larger than the cell size), and over much larger cell populations (tens of thousands of cells).

We processed the images with Farsight image processing pipeline (farsight-toolkit.org). We implemented the pipeline using a combination of C++ and Python languages, and incorporated it into the free and open source toolkit for quantitative studies of complex and dynamic tissue microenvironments. We used the open SWC file format to save the arbor reconstructions [44]. Next, we used the FARSIGHT TraceEditor [76] tool for three-dimensional visualization, manipulation, editing, and arbor feature

computation. This tool is particularly powerful for this type of work since it allows large ensembles of cellular arbors to be visualized at once, rather than one cell at a time. All the views are implemented using the powerful rendering engine in the open source NLM Visualization Toolkit (VTK) [79]. The cellular reconstructions are actively linked to the corresponding entries in the L-measure feature table, and a set of analytical and data analysis tools, including scatter plots, tables, heatmap display, clustering analysis [47], and our subspace trend discovery. Being "actively linked" means that selecting objects in one view will automatically trigger an automatic selection of corresponding data and objects in all the other views. Edits made in any one of these views are correctly reflected in all other views, and all the analytics are recomputed and refreshed. In the following sections, we first introduced the imaging and image processing pipeline in greater detail and then described our exploratory discovery with the subspace trend discovery algorithm we proposed.

5.1.1. Imaging and image processing

For the data presented here, thick coronal tissue slices of 4% paraformaldehyde fixed rat brain motor cortex were subjected to fluorescent cytochemistry and spinning-disk confocal microscopy. Using multiplex labeling and computer controlled stage movements during step-and image-data collection, identification of cells in large 3-D tissue volume was achieved (x and y dimensions of 1-2mm and z dimensions of 100-300μm, as much as 1.2mm^3). Thick tissue slices of 100μm were prepared from control brains and brains implanted with microfabricated recording devices (NeuroNexus, Ann Arbor, MI). Tissue sections from the implanted brains contained the implanted device. Tissue slices were fluorescently labeled for microglia (Ibal; anti-Iba1, Wako Chemical

Co., Richmond VA), nuclei (Hoechst 33342, Life Technologies, Carlsbad, CA), astrocytes (GFAP; anti-GFAP, Life Technologies) and neurons (NeuroTrace, Life Technologies), following previously published methods [78]. A Rolera EM-C2 camera (QImaging, Surrey, Canada) mounted on an Olympus spinning-disk confocal system (Center Valley PA) was used to record images (×30, 1,004 ×1,002 pixels at a resolution of 0.267μm/pixel, 14 bits/pixel, step size of 0.3μm). Overlapping image tiles were combined into a 3-D montage of extended fields. The tiles were collected with a 15-20% overlap. The image tiles were then combined into a 3-D montage of the whole field using an automated registration algorithm [80]. Typical fields were more than one mm wide with the device near the middle and 1.5-2.5mm long to provide one field of data more than the length of the device.

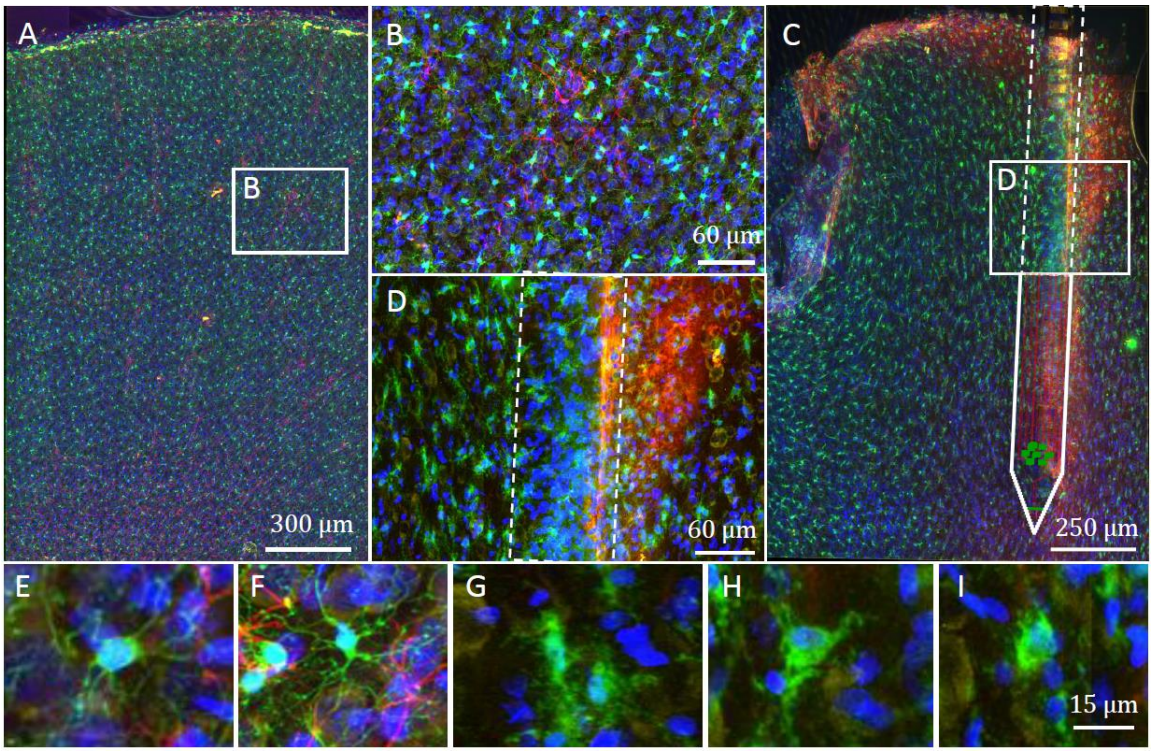

Figure 20. Maximum-intensity projection of a 4-channel 3-D confocal montage of a normal rat brain tissue slice (A) and with an embedded neural recording device (C). (B) and (D) are the close-up regions of regions B and D. (E)-(I) show images of individual microglia in green.

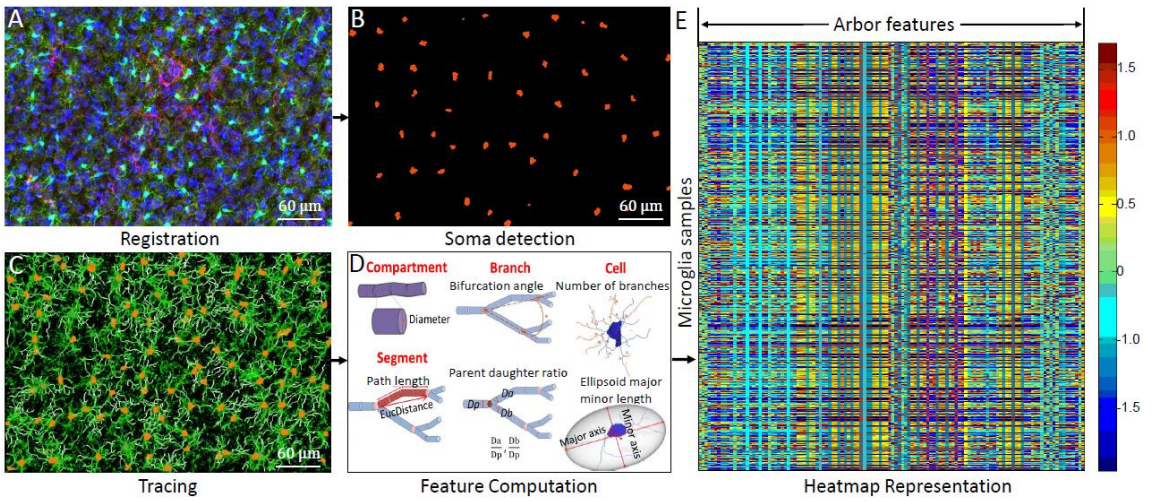

Figure 21. Image processing pipeline. (A) A maximum-intensity image projection; (B) Soma segmentation results (in orange); (C) Microglia arbor reconstruction (in white) overlaid on the Iba-1 signal (green); (D) L-measure feature computation; (E) Heatmap of original L-measure data.

The next step in our image processing pipeline is to isolate the microglia in the images from all the other brain cell types, and then reconstruct each indivudal cell's arbors using the recently reported method [81]. The main processing steps are illustrated in Figure 21 and summarized next. Microglia arbors consist of several trees of branching processes that emanate from a central soma. To detect the microglial somas, we proceed in two steps. First, we perform an automated segmentation of all the cell nuclei in the Hoechst image channel using a well-established method [82]. From the segmentations, we compute the 3-D location, volume, shape factor, and chromatin intensity variance (a texture measure) for each nucleus. Next, we compute the total expression of the microglial marker Iba-1 within a distance of 8 voxels from each nucleus [83]. These features were then used to train the logistic regression based classifier, the training examples were selected actively based on the Fisher information measure as in [84]. Villamizar [85] have reported the segmentation and classification accuracy for rat brain images that have been processed by the same specimen preparation and imaging protocol has been reported separately at 95%. Next, we segment the corresponding soma using a

3-D active contour algorithm [86]. In order to reconstruct the microglial arbors, we developed an over-complete dictionary based model for the image-specific local structures of microglial processes [81]. Next, we reconstructed the arbors for all of the microglia concurrently using a parallel version of Prim's minimal spanning forest algorithm [87], driven by a dynamically computed set of cost metrics that are computed using the Fast Marching Method [88]. This results in a set of reconstructions, one per microglial cell, with each reconstruction represented geometrically as a branching tree of tubes. The centerlines of the reconstruction are a connected series of 3-D nodes with local radii and local connectivity information, which are stored in the standard SWC file format [44] from which arbor measurements are computed, as described in Section 4.5.1. We combined the 3311 microglia from the normal tissue and 4409 microglia from the implanted tissue with the neural recording device for a total of 7720 microglia and 136 features.

5.1.2. Progression discovery of arbor morphology

The subspace trend discovery algorithm automatically selects a total of 109 features. To more efficiently and clearly visualize the trend, we do a hierarchical clustering of the cells based on the selected features and making a cut to the hierarchical dendrogram to generate 6 clusters of microglia, from C_1 to C_6 so that the features are homogeneous within each cluster. The resulting tree of the 6 clusters does not have any branches, as shown in Figure 22. As we selected the tree node from left end to right end, we found that the microglia arbor gradually changes from highly ramified state to amoeboid state, which is consistent with the known morphological progression of microglia [40]. When we evaluate the features with the discovered trend in Figure 23, we found that most

features have high scores with the trend (above 0.6). The selected features are mostly branching related and the unselected ones are mostly diameter-related. What interest us most is the population distribution difference between the normal tissue and implanted tissue, as shown in Figure 24. The highly ramified microglia, C_1 and C_2, are more abundant in normal tissue while the amoeboid is more abundant in the implanted tissue. It confirms the hypothesis that the microglia may shrink their arbors to move to the injury site more easily. In addition to the population distribution difference, we are also interested in the spatial distribution of microglia at different states in both the normal and implanted tissues. Thus we mapped their locations by spheres and colored each sphere by their state from red to blue in Figure 25. We noticed that the microglia are distributed uniformed in the normal tissue while in the implanted tissue, while C_4, C_5 and C_6 tend to locate around the device, especially for the amoeboid C_6, with a close up of the region around the device shown in Figure 26(a).

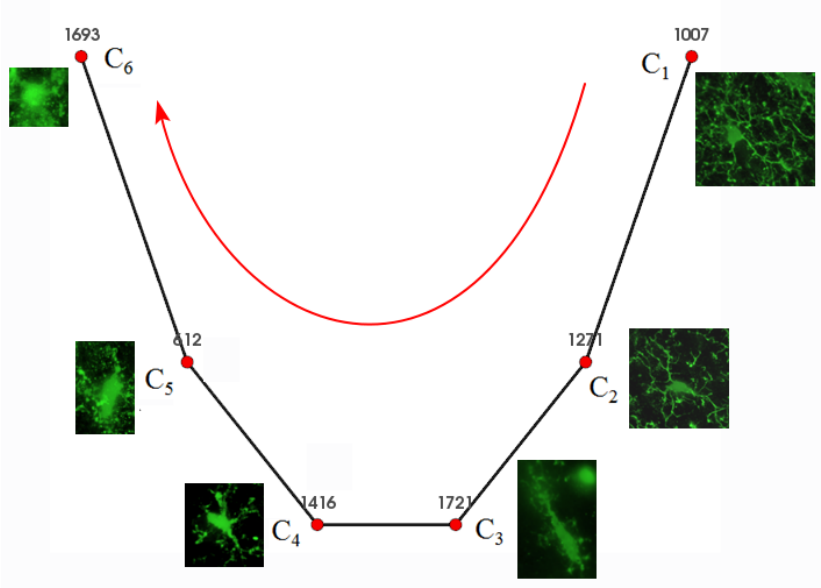

Figure 22. Tree visualization of derived 6 microglia clusters. The node number indicates the number of microglia cells within each cluster. The image close-ups of the representative cells are shown next to the node.

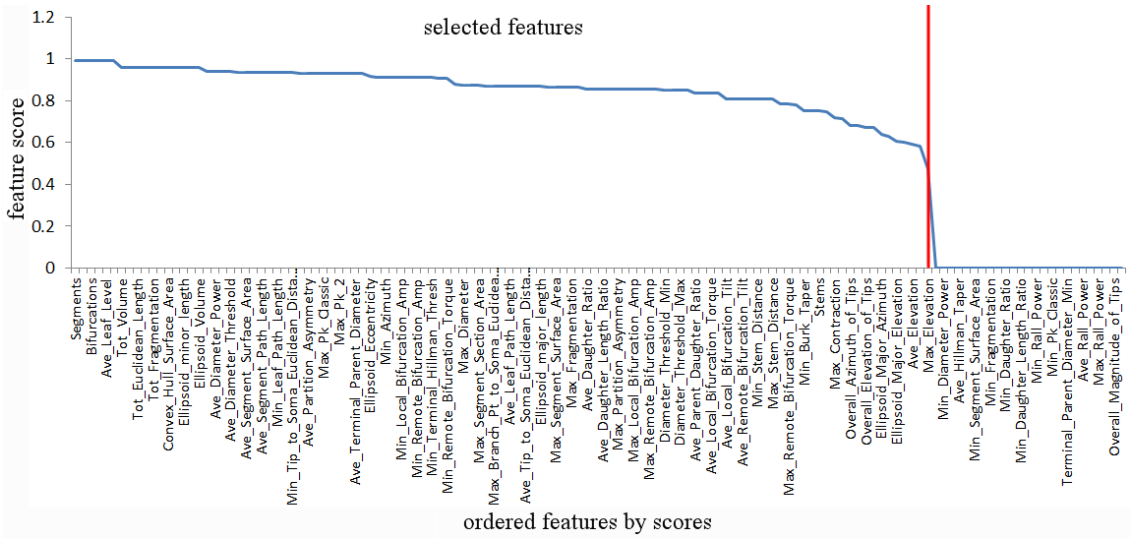

Figure 23. Feature evaluation: a large number of features are considered relevant to the trend.

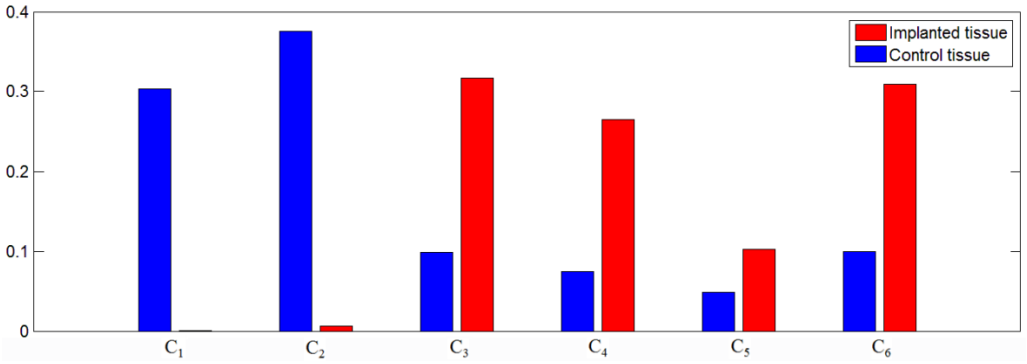

Figure 24. Microglia population distribution in normal/control tissue and implanted tissue. C_1 and C_2 are more abundant in control tissue while C_6 is more abundant in implanted tissue.

We have plotted the progression heatmap for the six clusters in Figure 26(c), where each row represents data points and each column represents features. The rows are arranged according to the ordering of hierarchical clustering within each cluster and the columns are arranged to the agglomerative clustering of features with the selected features separated from the unselected. From C_1 to C_6, we can see how the features vary gradually from highly ramified state to amoeboid state. For example, the total number of segments and the number of branch points are gradually decreasing. The original images and segmentations of the representative cells are shown in Figure 26(b), where the orange blob is the cell soma and the white lines are the arbors.

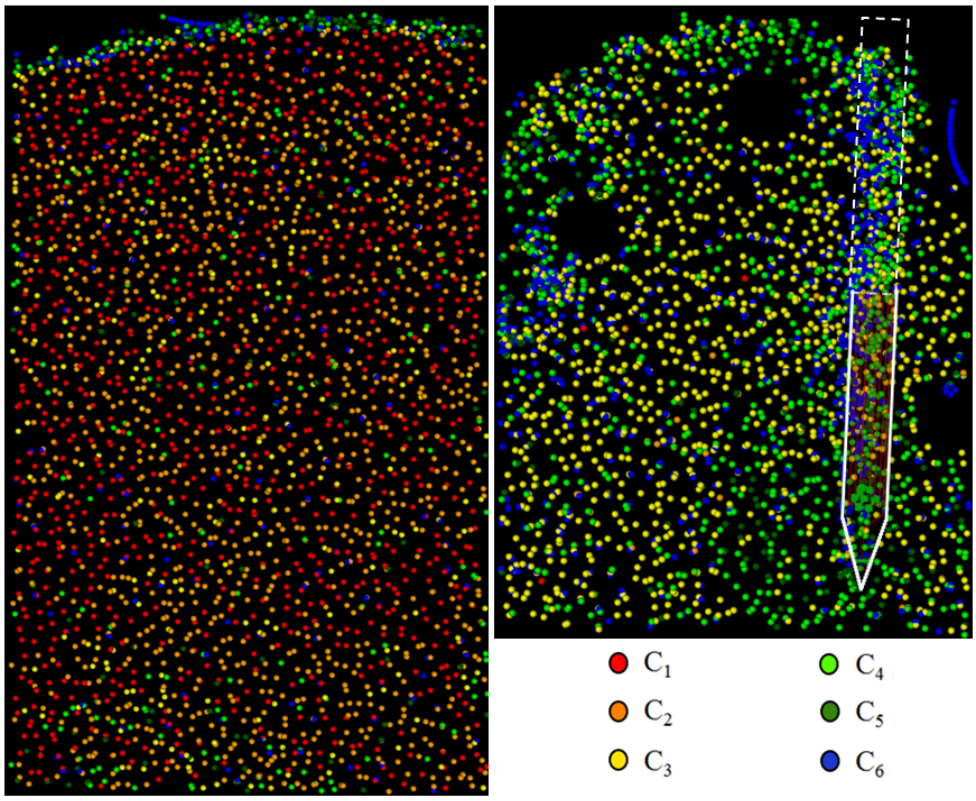

Figure 25. Spatial distribution of microglia six clusters from C_1 to C_6. Each cell is displayed as a sphere.

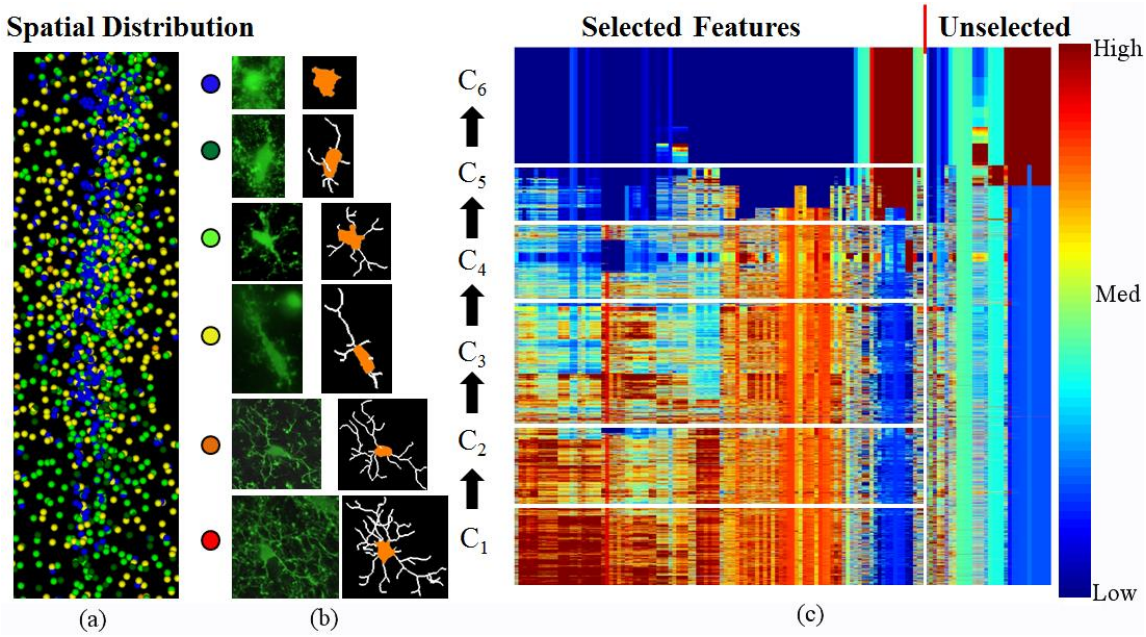

Figure 26. Microglia arbor progression: (a) Spatial distribution of microglia near the implant displayed as color-coded spheres. (b) Sample close-up images (green), and automated arbor traces (right), of microglia for each class. (c) Heatmap representation of the progression showing gradual variation of the selected features.

5.2. Massive Neuron Morphology Progression Discovery in Drosophila Brain

We have obtained 16,050 brain-wide neuron traces from Drosophila CNS for morphology progression discovery. Chiang [89] produced this first largescale brain-wide reconstruction of 16,050 Drosophila CNS neurons using a quasi-automated method, and embedded the neurons in a standardized fly brain atlas, to generate a virtual fly brain. This provides the first step in the analysis of neurons in a complete brain and it is the first time that the brain-wide neuron morphology is related with the birth time, neuron transmitter type and brain region information that are provided along with the morphology data. This dataset was also converted to the free SWC format, and was released in NeuroMorpho version 6.0 [45].

More specifically, the neuron images were collected as follows. Using genetic mosaic analysis with a repressible cell marker (MARCM) [90], the single brain neurons born at specific times during development were labeled with green fluorescent protein (GFP). Sample brains were imaged under a Zeiss LSM 510 confocal microscope with a 403C-Apochromat water-immersion objective lens. Only the brains containing one or a few non-overlapping single neurons with intense GFP labeling were imaged so that each neuron could be selectively extracted from the raw image. To compile single neurons taken from different flies, two standard model brains were generated to represent 6-day-old Drosophila adult brains, one female and one male. Each standard model brain consists of average external cell cortex and inner neuropil surfaces. The GFP-labeled neurons were then semi-automatically segmented with Amira 4.1.2 (www.fei.com). Then each of the 16,000 sample images was registered to one of the two template brains by a

global affine registration so that all the neurons are scaled and mapped to the same model, as shown in Figure 27. Lastly, we loaded the segmentation results into Farsight TraceEditor [76] for arbor morphology quantification.

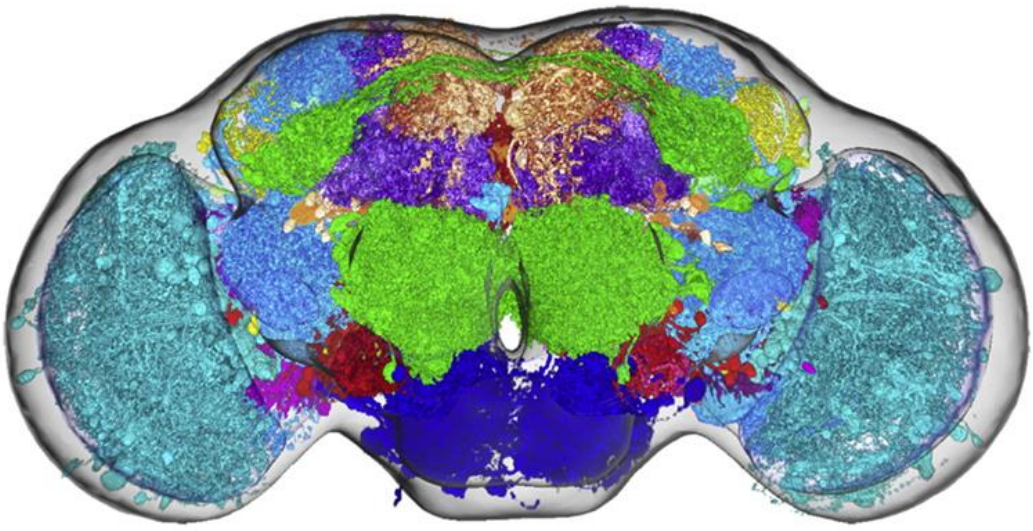

Figure 27. Integration of neurons in the standard brain models. Colors represent neurons in the same neuropils.

5.2.1. Online density-based representation for neuron arbor morphology

We have implemented the online density-based representation in Matlab. The output of the TraceEditor is a table of 136 morphology measurements for the 16,050 neurons. Since we do not have the information regarding the cell body (soma) location for each neuron, the branching direction and order are unknown. Therefore, many of the parent-daughter branching features relying on the branching direction are not correct without the soma locations. Thus we removed those branching features and mainly kept the features at cell level, such as the number of branch points, the number of terminals and features of the smallest wrapping ellipsoid to describe the overall shape of the neurons. We then had 31 features, 16,050 neuron samples as the input for our subspace trend discovery algorithm and then selected 11 features as the input for the online representation algorithm to represent the trend by the data ordering.

The original selected 11-feature normalized data set of the unordered data points is shown in Figure 28. For the online representation, we used Gaussian components to represent the data density and set the approximation error threshold as 0.05 to keep a balance between the approximation accuracy and the computational complexity. Then we added the data points one by one till all the data points have been added to the representation. In the end, 32 Gaussian components, representing 32 homogeneous clusters, are derived. The data density is then represented by a weighted sum of the 32 Gaussian components. The data points are assigned to the Gaussian component with the maximum coefficient among the 32 weights. To further reduce the number of cluster for future analysis, we clustered the Gaussian components by their centers using an agglomerative clustering and achieved 10 clusters (progression stages) in the end. The Gaussian coefficients and the corresponding represented data points are shown in Figure 29. The rows are arranged by the optimal leaf ordering of the features; and the columns, the data points are ordered by the depth-first traversal of the means of each cluster. Within each cluster, the Gaussian components are ordered according to the depth-first traversal of the Gaussian centers. And within each Gaussian component, the data points are ordered by the descending maximum coefficients.

Through this three-level ordering of data points, we can clearly identify the gradual variation of features moving from one progression stage to another. For example, from progression stage 7 to 10, the number of segments is gradually increasing, which means the neurons are getting more arbors and branching more often. We calculated the feature mean for each progression stage in Figure 30 and selected the representative neurons for visualization. Stage 9 neurons are of special interest to me, they branch a lot like stage 10

neurons but the total path length is much lower than the neurons from stage 8 or 10, which means they tend to branch locally rather than spread out long distance across regions. It demonstrates an advantage of this online representation to reveal a small cluster that can otherwise get undermined by the abundant cluster.

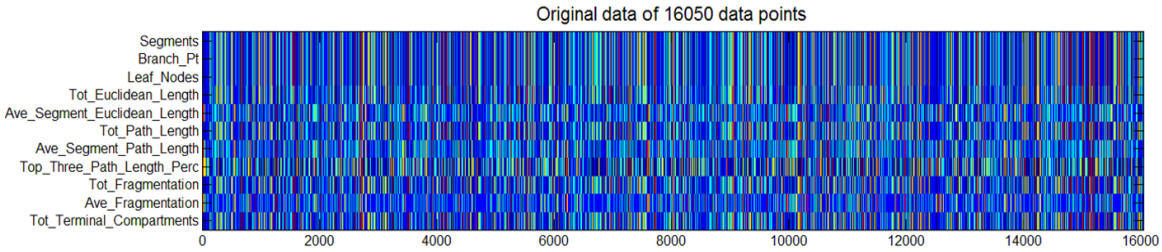

Figure 28. Original data set of the 16050 neurons without any data ordering.

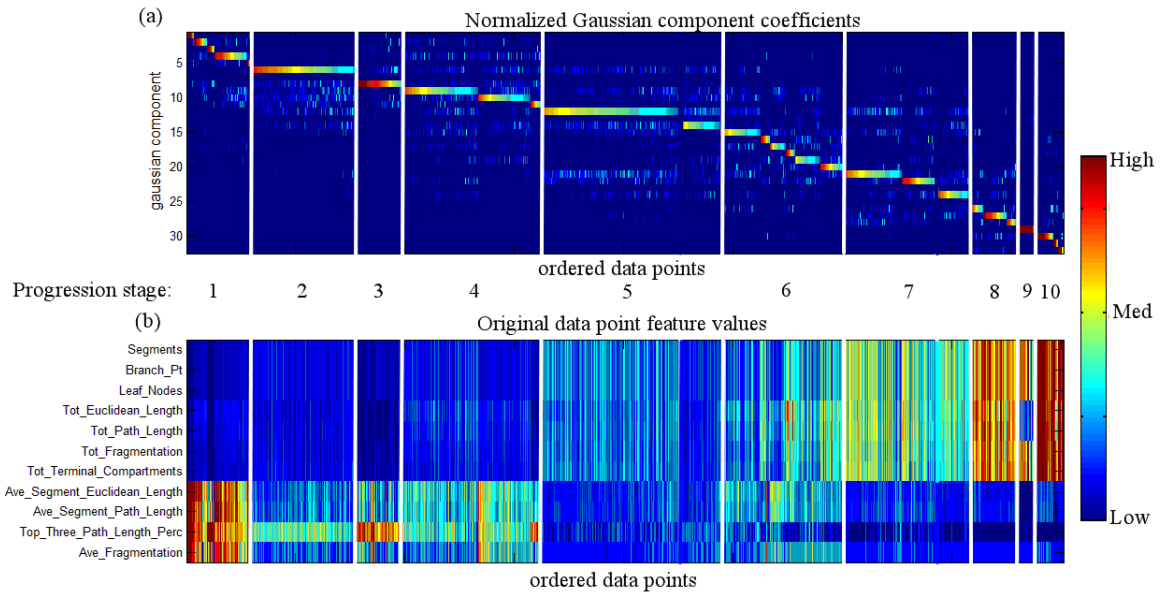

Figure 29. Online density based representation of the 16,050 data points with 11 selected features by the subspace trend discovery algorithm. (a) Normalized Gaussian component coefficients; (b) Ordered data points to show the gradual variation of features.

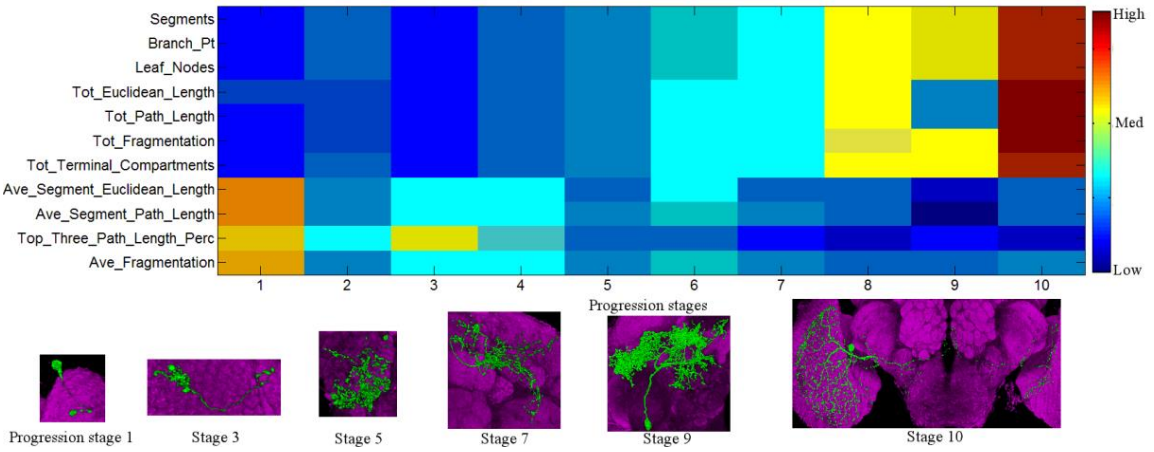

Figure 30. Mean of the 11 features for each progression stage and the representative arbor morphology.

5.2.2. Relating the morphological progression to birth time, transmitter type and brain regions

We obtained the putative birth time, transmitter type and lying-across brain regions for each neuron. We are interested to know whether the morphological progression is aligned with the birth time, the transmitter type or the brain regions. For birth time, we used 8 time points when there are sufficient neurons for analysis and calculated the neuron population distribution at each time point, as shown in Figure 31. We found that the progression stages of the neurons born at earlier time before birth time 4, are gradually increasing, whereas the progression stages of the neurons born at later time after 4 are decreasing. It indicates that the neurons are gaining morphological complexity over time, i.e., gaining arbors and branches. To identify the neurons at which birth time contribute most to the each progression stage, we normalized the percentages by dividing by the maximum percentage at each progression stage to bring the maximum up to 1 for each stage, as shown in Figure 32. We can clearly identify a watershed between progression stage 5 and 6, birth time 3 and 4. The neurons at birth time 4-7 are more abundant at progression stages 1-5 and the ones at birth time 0-4 are more abundant at

progression stages 6-10.

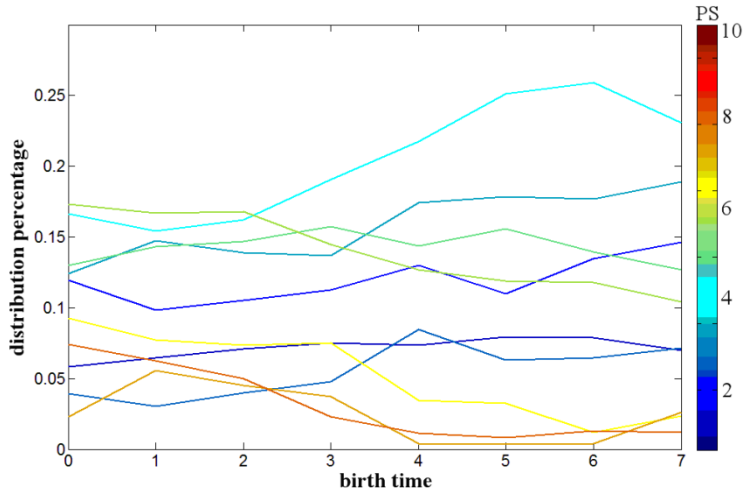

Figure 31. Neuron population distribution at birth time 0 to 7. The percentages are summed to 1 at each birth time.

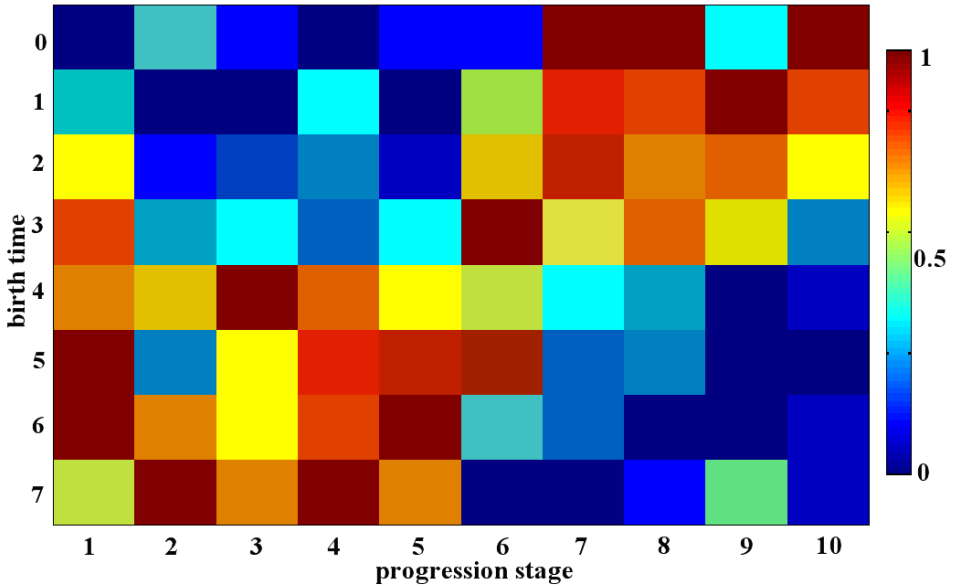

Figure 32. Normalized neuron population distribution for each progression stage with the maximum of each column brought up to 1.

Next, we compared the progression stages with the transmitter types. There are mainly 9 transmitter types available. The neuron populations for these transmitter types are shown in Figure 33. For the transmitter types that have sufficient neurons for female and male (above 100), namely TH, Trh and fru, we treated the neurons separately for female and male. Thus we have fru-M and fru-F types for fru transmitter type. We

calculated the neuron population distribution for each transmitter type and visualized the distribution vector in a heatmap in Figure 34. The rows, representing the transmitter types, are ordered by the optimal leaf ordering of the distribution vectors, the rows so that the similar rows are put close in the heatmap. We found that the female and male neurons from the same transmitter types are similar regardless of sex difference. TH transmitter neurons have the highest progression stages, especially at progression stage 9, a small cluster discovered by our online representation, which is very interesting for further biological interpretation and validation.

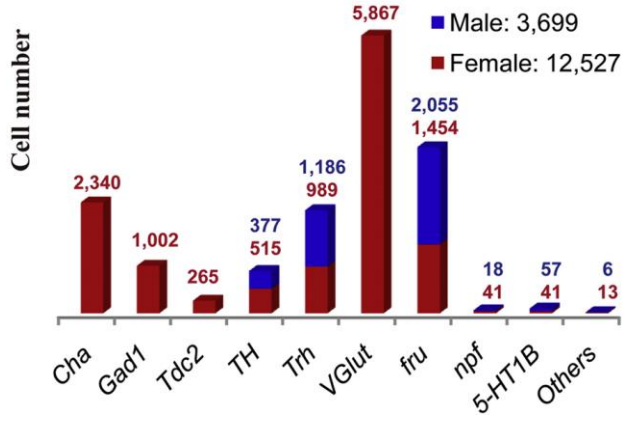

Figure 33. Neuron population of different transmitter types.

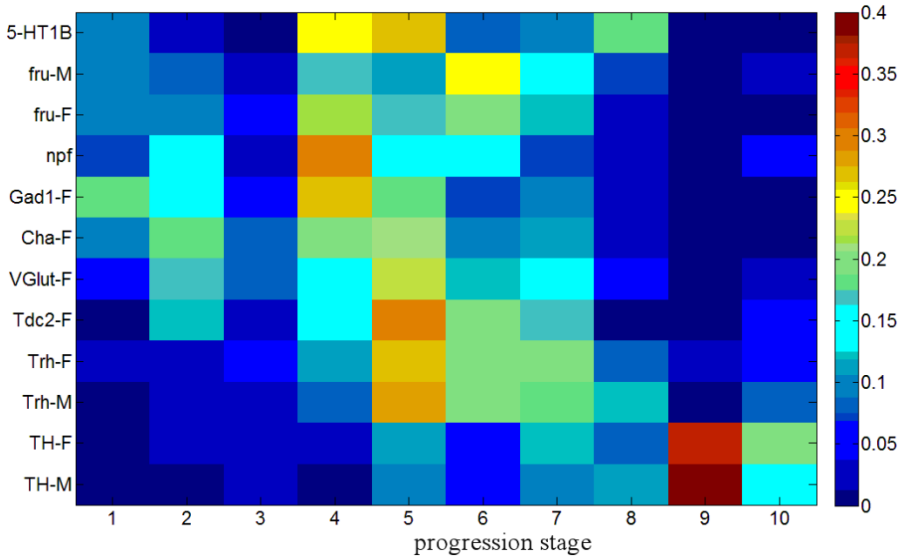

Figure 34. Neuron population distribution for each transmitter type. Each row is a distribution vector summed to 1.

Finally, we mapped the neurons to the brain regions according to their axon or dendrite locations, where their axons or dendrites are mostly distributed. We had a total of 29 morphologically distinguishable neuropil regions in both the left and right brain, as indicated in Figure 35(a). We counted the symmetric regions from the left and right brain as one brain region so as to collect more neurons for each region. Each neuron was assigned to exactly one brain region according to its dendrite locations and to exactly one brain region according to its axon locations. Therefore, each neuron has a unique dendrite brain region and unique axon brain region. The median level of progression stages for each brain region according to either dendrite or axon location is shown in Figure 35. Generally, the lateral regions, such as MED, LOB and LOP have lower progression stages than the medial regions, such as SDFP, SOG and VMP. Furthermore, we mapped the progression stage on the brain region connections from axon to dendrite, as shown in Figure 36. The rows are the axon locations and the columns are the dendrite locations. We took the mean of the progression stages for each connection and only the connections that have 3 or more neurons are shown in the heatmap. The rows and columns are ordered by the optimal leaf ordering of the rows so that it preserves matrix symmetry. The matrix is not quite symmetric since the neurons connecting brain regions from A to B do not necessarily indicate the existence of neurons connecting from B to A. Furthermore, since the neurons we collected are not yet complete for the drosophila brain and some connections have too few neurons to calculate the statistics, some connections may be missed. But we do notice the morphological differences in terms of brain region connections.

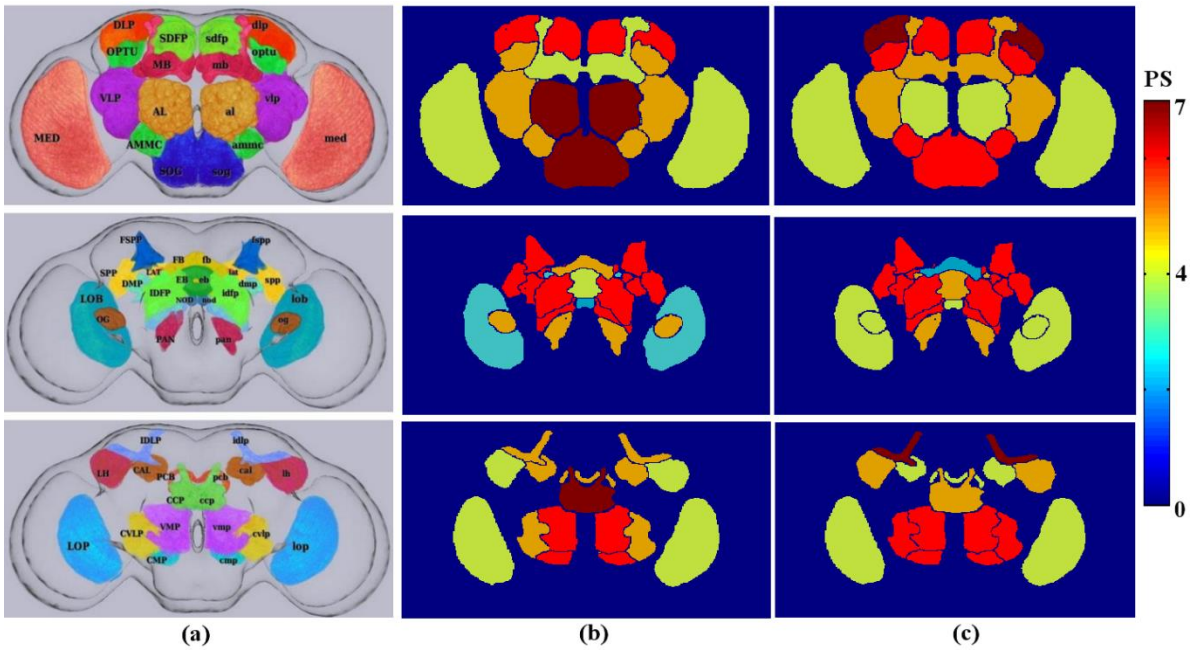

Figure 35. Progression stages mapped to brain regions according to axon or dendrite location. (a) Brain region names; (b) Median progression stage of each brain region w.r.t. axon location; (c) Median progression stage of each brain region w.r.t. dendrite location.

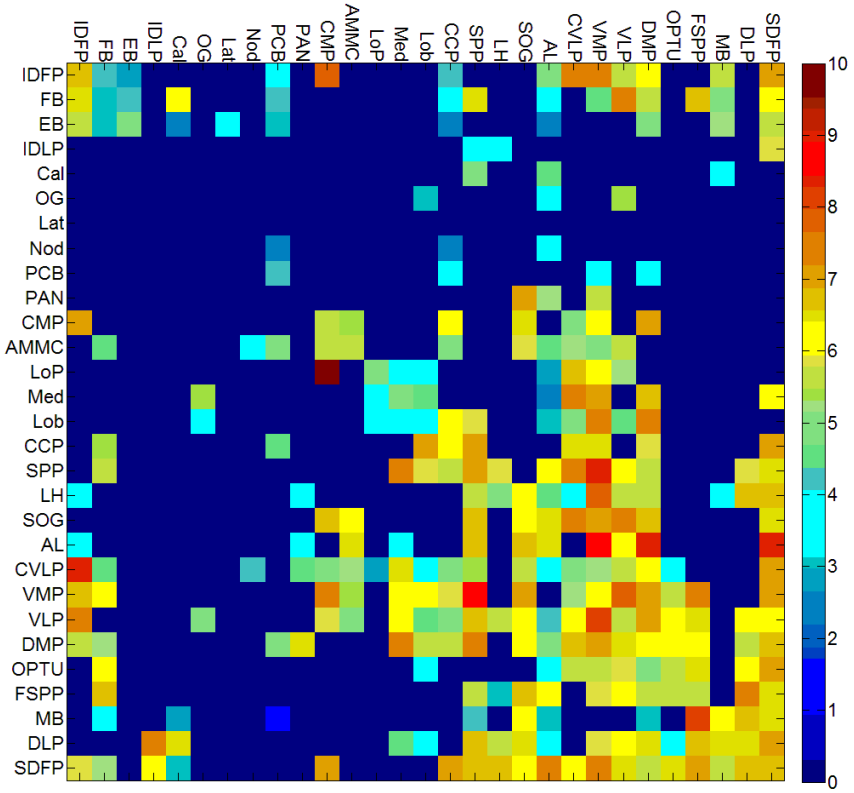

Figure 36. Progression stages mapped to brain region connections.

6. Conclusion and Future Work

In the real-world datasets that we analyzed, the subspace trends uncovered in an unsupervised and data-driven manner by the proposed algorithm, without the benefit of prior information, were in excellent concordance with the underlying known biology. In each case, visualizations derived from the selected trend-relevant features revealed meaningful hidden subspace trend(s) that were obscured by irrelevant features and noise. This echoes the fundamental importance of task-appropriate feature selection in the analysis of high-dimensional data. The task of interest is subspace trend discovery rather than classification or clustering, emphasizing progressive ordering patterns among data points rather than grouping/separation patterns. The neighborhood similarity metric successfully captures linear and nonlinear associations between features for this task, and provides a basis for identifying the trend-relevant features.

A major strength of the proposed method is that it is unsupervised, and makes minimal assumptions about the data. If the data points are sampled from a gradually changing process that is reflected in a subset of the features, the proposed method can identify the underlying subspace trend(s) and the features relevant to the trends. The granularity of the identified trend is dependent on how densely the underlying trend is sampled. The proposed method can handle data sets containing multiple independent trends driven by non-overlapping feature subsets, as in the example in Section 4.3. Although the real datasets have very different properties (some with higher dimensionality than the sample size and others with much larger sample size than the dimensionality), the two primary parameters in our algorithm were kept the same. The parameters are quite intuitive, and easy to set. The numbers of features in the

corresponding connected component of the feature graph and threshold $\hat{T}$ are useful indicators of the strength of discovered trend(s). The connection accuracy measure is helpful for verifying trends.

Specifically for massive dataset, our online density based representation enables updating the data visualization with one data point at a time with a sublinear computational complexity. For calculating the data density, only the derived Gaussian components and their corresponding weights are needed without storing all the data points. Our experiment with the massive neuron morphology demonstrates another advantage of this representation to reveal the small clusters in the data, which can be likely undermined by the abundant clusters.

The potential applications of this method are widespread, including computer-assisted hypothesis generation, knowledge discovery, modeling, and prediction. Although our example data sets were drawn from the biological domain, the proposed algorithm is broadly applicable to exploratory analysis of non-biological data as well. It can be used in a stand-alone manner, and in conjunction with a variety of projection based multi-dimensional data visualization algorithms and heatmap representations. To promote usage, we developed an efficient multi-threaded open-source software implementation that is compatible with multiple effective visualization tools.

Discovery and visualization of subspace trends is an interesting and a non-trivial problem in its own right, and much more work remains to be done. In our experiments, most trends exhibit linear, i.e., with no branches. Though it is potential to discover the branching differentiation, more work remains to be done to specifically validate the branches and identify the features that cause the branching. Another direction is to

simultaneously identify the clusters along the trend. Right now, the feature selection process is independent with the clustering in our algorithm, which is after the feature selection. However in some cases, the clusters in the data can greatly affect the feature selection result. The problem of subspace trend discovery offers abundant potential for innovation for visualization system researchers. Currently, we use a heatmap representation for visualizing the gradual changes of features along the trend. It will be very useful to derive an online embedding of the data points in low dimensions based on the derived Gaussian components. In future, we hope to link these different representations in a "live" manner to enable synergistic human-machine systems for real-time and large-scale trend discovery and exploitation. Overall, the proposed algorithm can form the core engine of an integrated *subspace trend analytics* toolkit.

Appendix A. Subspace Trend Discovery Algorithm

Algorithm 1 SubspaceTrendDiscovery ($\boldsymbol{X}$)

Input: normalized multivariate data over features:

$\boldsymbol{X}$: $\{\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_M}\}$.

Parameters: σ, k, B, γ, L.

Output: Trend-relevant feature subsets **J**.

Steps:

$\boldsymbol{MX}$ *(meta feature)* = AgglomerativeClustering($\boldsymbol{X}$, σ)

for $i = 1: M'$

  for $j = i + 1: M'$

    $\boldsymbol{\Phi}(i, j)$ = NeighborhoodSimilarity($\boldsymbol{MX_i}$, $\boldsymbol{MX_j}$, k, B)
    $\boldsymbol{\Phi}(j, i) = \boldsymbol{\Phi}(i, j)$

  end for

  $\boldsymbol{\Phi}(i, i)$ = *1;*

end for

J = TrendRelevantFeatureSelection($\boldsymbol{\Phi}$, γ, L)

Visualize the hypothesized subspace trends separately with the size-decreasing ordered trend-relevant feature subsets $\{\mathbf{J_1}, \mathbf{J_2}, \dots\}$, usually $\mathbf{J_1}$ is of most interest to trends.

Algorithm 2 AgglomerativeClustering ($\boldsymbol{X}$, σ)

Input: feature sets $\boldsymbol{X}$: $\{\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_M}\}$.
Parameters: coherence threshold σ.

Output: Meta feature sets $\boldsymbol{MX}$: $\{\boldsymbol{MX_1}, \boldsymbol{MX_2}, \dots, \boldsymbol{MX_{M'}}\}$.

Steps:

Initiate $\boldsymbol{MX_i} = \{\mathbf{x_i}\}$, $1 \le i \le M$.

Iterate

  for an unexamined meta feature $\boldsymbol{MX_i}$

    if there exists an unexamined meta feature $\boldsymbol{MX_j}$ that makes AveragePearsonCorrelation($\boldsymbol{MX_i}$, $\boldsymbol{MX_j}$) ≥ σ

$\boldsymbol{MX_i} = \{\boldsymbol{MX_i}, \boldsymbol{MX_j}\}$

Remove $\boldsymbol{MX_j}$.

end if

Set $\boldsymbol{MX_i}$ as examined.

end for

Reset the new meta features as unexamined.

until no meta feature pair can be merged.

Function AveragePearsonCorrelation ($\boldsymbol{MX_i}$, $\boldsymbol{MX_j}$)

Input: $\boldsymbol{MX_i} = \{\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_{M_i}}\}$, $\boldsymbol{MX_j} = \{\mathbf{x_1}, \mathbf{x_2}, \dots, \mathbf{x_{M_j}}\}$.
Output: α.
Steps:

$\bar{\mathbf{x}} = (\sum_{v=1}^{M_i} \mathbf{x_v} + \sum_{w=1}^{M_j} \mathbf{x_w})/(M_i + M_j)$

$$\alpha = \frac{\sum_{v=1}^{M_i} \mathrm{PC}(\mathbf{x}_{\boldsymbol{v}}, \bar{\mathbf{x}}) + \sum_{w=1}^{M_j} \mathrm{PC}(\mathbf{x}_{\boldsymbol{w}}, \bar{\mathbf{x}})}{M_i + M_j}$$

PC: Pearson correlation

Algorithm 3 NeighborhoodSimilarity($\boldsymbol{MX_i}$, $\boldsymbol{MX_j}$, k, B)

Input: A pair of meta features $\boldsymbol{MX_i}$, $\boldsymbol{MX_j}$.
Parameters: k, B.

Output: $NS_{MX_i,\ MX_j}$.

Steps:

1. Compute distance matrix $\boldsymbol{D_{MX_i}}$ for $\boldsymbol{MX_i}$, $\boldsymbol{D_{MX_j}}$ for $\boldsymbol{MX_j}$.
2. Build k-NNG edge set $E^k_{MX_i}$ for $\boldsymbol{MX_i}$, $E^k_{MX_j}$ for $\boldsymbol{MX_j}$, and $\hat{E}$ for fully connected graph.
3. Derive edge length multisets:
$S_{D_{MX_i}, E^k_{MX_j}}$, $S_{D_{MX_i}, \hat{E}}$, $S_{D_{MX_j}, E^k_{MX_i}}$, $S_{D_{MX_j}, \hat{E}}$.
4. Get empirical distributions of the multisets by using B equally-spaced bins:
$W_{D_{MX_i}, E^k_{MX_i}}$, $W_{D_{MX_i}, E^k_{MX_j}}$, $W_{D_{MX_i}, \hat{E}}$ and $W_{D_{MX_j}, E^k_{MX_i}}$, $W_{D_{MX_j}, E^k_{MX_j}}$, $W_{D_{MX_j}, \hat{E}}$.
5. Solve EMD for directional neighborhood similarity:

$$NS_{MX_i \to MX_j} = \frac{\mathrm{EMD}\left(W_{D_{MX_j}, \hat{E}},\ W_{D_{MX_j}, E^k_{MX_i}}\right)}{\mathrm{EMD}\left(W_{D_{MX_j}, \hat{E}},\ W_{D_{MX_j}, E^k_{MX_j}}\right)}.$$

and

$$NS_{MX_j \to MX_i} = \frac{\mathrm{EMD}\left(W_{D_{MX_i},\hat{E}},\ W_{D_{MX_i},E^k_{MX_j}}\right)}{\mathrm{EMD}\left(W_{D_{MX_i},\hat{E}}, W_{D_{MX_i},E^k_{MX_i}}\right)}.$$

6. $NS_{MX_i,\ MX_j} = \max(NS_{MX_i \to MX_j}, NS_{MX_j \to MX_i})$.

Algorithm 4 TrendRelevantFeatureSelection($\boldsymbol{\Phi}$, γ, L)

Input: Neighborhood similarity matrix $\boldsymbol{\Phi}$.

Parameters: γ, L.

Output: J (Trend-relevant feature subsets) in decreasing order of size: $\{\mathbf{J_1}, \mathbf{J_2}, \ldots\}$.

Steps:

1. Get the empirical distribution of $\boldsymbol{\Phi}$ using L equally-spaced bins and the normalized occurrence frequency at bin l is h_l.

2. **for** $T = 1{:}\,L$

$h_{Z1}(T) = \sum_{l=1}^{T} h_l$
$h_{Z2}(T) = \sum_{l=T+1}^{L} h_l$
end for

3. **for** $T = 1{:}\,L$

$$R_{Z1}(T)(T) = -\sum_{l=1}^{T} \frac{h_l}{h_{z1}(T)} \log \frac{1}{2}\left(1 + \frac{1}{h_{z1}(T)} \sum_{i=l}^{T} h_i\right)$$
$$R_{Z2}(T)(T) = -\sum_{l=T+1}^{L} \frac{h_l}{h_{z2}(T)} \log \frac{1}{2}\left(1 + \frac{1}{h_{z2}(T)} \sum_{i=T+1}^{l} h_i\right)$$

end for

4. Find the minimum of $|R_{z1}(T) - R_{z2}(T)|$ as R_{MIN}.
5. Find the maximum T as $\hat{T}$ that makes

$$|R_{z1}(T) - R_{z2}(T)| \leq R_{\mathrm{MIN}} + \gamma$$

6. Apply $\hat{T}$ to $\boldsymbol{\Phi}$, $\boldsymbol{\Phi}'(i,j) = 1$ if $\boldsymbol{\Phi}(i,j) \geq \hat{T}$ else $\boldsymbol{\Phi}'(i,j) = 0$.

7. Derive connected components of meta features from $\boldsymbol{\Phi}'$ and order the according trend-relevant feature subsets in decreasing size: $\{\mathbf{J_1}, \mathbf{J_2}, \ldots\}$.

References

[1] S. T. Roweis and L. K. Saul, "Nonlinear Dimensionality Reduction by Locally Linear Embedding," *Science*, vol. 290, no. 5500, pp.2323-2326, Dec. 2000.

[2] J. B. Tenenbaum, V. Silva, and J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, vol. 290, no. 5500, pp. 2319-2323, Dec. 2000.

[3] M. Belkin, and P. Niyogi, "Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering," *Advances in Neural Information Processing System*, vol. 14, pp. 585-591, 2001.

[4] L. Maaten, and G. Hinton, "Visualizing Data using t-SNE," *J. of Machine Learning Research*, vol. 9, pp. 2579-2605, Nov. 2008.

[5] R. R. Coifman, S. Lafon, A. B. Lee , M. Maggioni , B. Nadler, F. Warner, and S. W. Zucker, "Geometric Diffusions as a Tool for Harmonic Analysis and Structure Definition of Data: Diffusion Maps," *Proc. of the National Academy of Sciences of the United States of America*, vol. 102, no. 21, pp. 7426-7431, Jan. 2005.

[6] C. Sotiriou, S. Y. Neo, L. M. McShane, E. L. Korn, P. M. Long, A. Jazaeri, P. Martiat, S. B. Fox, A.L. Harris, and E.T. Liu, "Breast Cancer Classification and Prognosis based on Gene Expression Profiles from a Population-based Study", *Proc. of the National Academy of Sciences of the United States of America*, vol. 100, no. 18, pp. 10393-10398, Sep. 2003.

[7] E. V. Jensen, and V. C. Jordan, "The Estrogen Receptor, a Model for Molecular Medicine", *Clinic Cancer Research*, vol. 9, no. 6, pp. 1980-1989, Jun. 2003.

[8] I. Guyon, and A. Elisseeff, "An Introduction to Variable and Feature Selection," *J. of Machine Learning Research*, vol. 3, pp. 1157-1182, Mar. 2003.

[9] S. C. Yan, D. Xu, B. Y. Zhang, H. J. Zhang, Q. Yang, and S. Lin, "Graph Embedding and Extensions: A General Framework for Dimensionality Reduction," *IEEE Tans.*

Pattern Analysis and Machine Intelligence, vol. 29, no. 1, pp. 40-51, Jan. 2007.

[10] H. P. Kriegel, P. Kroger, and A. Zimek, ”Clustering High-Dimensional Data: A Survey on Subspace Clustering, Pattern-Based Clustering, and Correlation Clustering,” *ACM Transactions on Knowledge Discovery from Data*, vol. 3, no. 1, pp. 1-58, Mar. 2009.

[11] S. Rao, R. Tron, R. Vidal, and Y. Ma, “Motion Segmentation in the Presence of Outlying, Incomplete, or Corrupted Trajectories,” *IEEE Tans. Pattern Analysis and Machine Intelligence*, vol. 32, no. 10, pp. 1832-1845, Oct. 2010.

[12] G. C. Liu, Z. C. Lin, S. C. Yan, J. Sun, Y. Yu, and Y. Ma, “Robust Recovery of Subspace Structures by Low-Rank Representation,” *IEEE Tans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 171-184, Jan. 2013.

[13] E. Elhamifar, and R. Vidal, “Sparse Subspace Clustering: Algorithm, Theory, and Applications,” *IEEE Tans. Pattern Analysis and Machine Intelligence*, vol. 35, no. 11, pp. 2765-2780, Nov. 2013.

[14] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan, “Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications,” *Proc. ACM Int’l Conf. on Management of Data (SIGMOD)*, pp. 94-105, Jun. 1998.

[15] C. H. Cheng, A. W. Fu, and Y. Zhang, “Entropy-based Subspace Clustering for Mining Numerical Data,” *Proc. fifth ACM International Conf. on Knowledge Discovery and Data Mining (SIGKDD)*, pp. 84-93, Aug. 1999.

[16] J. W. Chang, and D. S. Jin, “A New Cell-based Clustering Method for Large, High-dimensional Data in Data Mining Applications,” *Proc. ACM Symp. Applied Computing (SAC)*, pp. 503-507, Mar. 2002.

[17] Y. H. Chu, J. W. Huang, K. T. Chuang, D. N. Yang, and M. S. Chen, “Density Conscious Subspace Clustering for High-Dimensional Data,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 1, pp. 16-30, Jan 2010.

[18] C. C. Aggarwal, J. L. Wolf, P. S. Yu, C. Procopiuc, and J. S. Park, "Fast Algorithms for Projected Clustering," *Proc. ACM Int'l Conf. on Management of Data (SIGMOD)*, pp. 61-72, Jun. 1999.

[19] K. G. Woo, J. H. Lee, M. H. Kim, and Y. J. Lee, "FINDIT: a Fast and Intelligent Subspace Clustering Algorithm using Dimension Voting," *Information and Software Technology*, vol. 46, no. 4, pp. 255-271, Mar. 2004.

[20] R. L. F. Robson, A. J. M. Traina, C. Faloutsos, and C. Traina Jr., "Halite: Fast and scalable Multiresolution Local-Correlation Clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 25, no. 2, pp. 387-401, Feb 2013.

[21] S. C. Madeira, and A. L. Oliveira, "Biclustering Algorithms for Biological Data Analysis: A Survey," *IEEE Transactions on Computational Biology and Bioinformatics*, vol. 1, no. 1, pp. 24-45, Jan. 2004.

[22] E. Achtert, C. Bohm, H. P. Kriegel, P. Kroger, and A. Zimek, "On Exploring Complex Relationships of Correlation Clusters," *Proc. 19th Int'l Conf. on Scientific and Statistical Database Management (SSDBM)*, pp. 7-13, Jul. 2007.

[23] E. Achtert, C. Bohm, J. David, P. Kroger, and A. Zimek, "Robust Clustering in Arbitrarily Oriented Subspaces," *Proc. 8th SIAM International Conference on Data Mining (SDM)*, pp. 763-774, Apr. 2008.

[24] L. F. Chen, Q. S. Jiang, and S. R. Wang, "Model-Based Method for Projective Clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 24, no. 7, pp. 1291-1305, Jul. 2012.

[25] J. Giesen, "Curve Reconstruction, the Traveling Salesman Problem and Menger's Theorem on Length," *Discrete and Computational Geometry*, vol. 24, pp. 577-603, 2000.

[26] P. M. Magwene, P. Lizardi, and J. H. Kim, "Reconstructing the Temporal Ordering of Biological Samples using Microarray Data," *Bioinformatics*, vol. 19, no. 7, pp. 842-850, 2003.

[27] R. Desper, J. Khan, and A. A. Schaer. "Tumor Classification using Phylogenetic Methods on Expression Data," *Journal of Theoretical Biology*, vol. 228, pp. 477-496, 2004.

[28] Y. Park, S. Shackney, and R. Schwartz, "Network-based inference of cancer progression from microarray data," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 6, no. 2, pp.200-212, 2009.

[29] C. Trapnell, D. Cacchiarelli, J. Grimsby, P. Pokharel, S. Li, M. Morse, N. J. Lennon, K. J. Livak, T. S. Mikkelsen, and J. L. Rinn, "The Dynamics and Regulators of Cell Fate Decisions are Revealed by Pseudotemporal Ordering of Single Cells," *Nature Biotechnology*, vol. 32, pp. 381-386, 2014.

[30] L. Haghverdi, F. Buettner, and F. J. Theis, "Diffusion maps for high-dimensional single-cell analysis of differentiation data", *Bioinformatics*, vol. 31, no. 18, pp. 2989-2998, 2015.

[31] S. Pokharkar, and C. K. Reddy, "Identifying Information-Rich Subspace Trends in High-Dimensional Data," *Proc. SIAM International Conference on Data Mining (SDM '09)*, pp. 557-568, 2009.

[32] P. Qiu, A. J. Gentles, and S. K. Plevritis, "Discovering Biological Progression Underlying Microarray Samples," *PloS Computational Biology*, vol. 7, no. 4, p. e1001123, Apr. 2011.

[33] P. Qiu, E. F. Simonds, S. C. Bendall, K. D. Gibbs Jr, R. V. Bruggner, M. D. Linderman, K. Sachs, G.P. Nolan, and S.K. Plevritis, "Extracting a Cellular Hierarchy from High-Dimensional Cytometry Data with SPADE," *Nature Biotechnology*, vol. 29, pp. 886-891, 2011.

[34] 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.

[35] S. C. Bendall, K. L. Davis, E. D. Amir, M. D. Tadmor, E. F. Simonds, T. J. Chen, D. K. Shenfeld, G. P. Nolan, and D. Pe'er, "Single-Cell Trajectory Detection Uncovers Progression and Regulatory Coordination In Human B Cell Development," *Cell*, vol. 157, pp. 714-725, 2014.

[36] S. Pyne, X. L. Hu, K. Wang, E. Rossin, T. Lin, L. M. Maier, C. B. Allan, G. J. McLachlan, P. Tamayo, D.A. Hafler, P.L. Jager, and J.P. Mesirov, "Automated High-Dimensional Flow Cytometric Data Analysis," *Proceedings of the National Academy of Sciences*, vol. 106, no. 21, pp. 8519-8524, 2009.

[37] I. Naim, S. Datta, J. Rebhahn, J. S. Cavenaugh, T. R. Mosmann, and G. Sharma, "SWIFT: Scalable Clustering for Automated Identification of Rare Cell Populations in Large, High-Dimensional Flow Cytometry Datasets, Part 1: Algorithm Design," *Cytometry*, vol. 85A, pp. 408-421, 2014.

[38] M. Dundar, F. Akova, H. Z. Yerebakan, and B. Rajwa, "A Non-Parametric Bayesian Model for Joint Cell Clustering and Cluster Matching: Identification of Anomalous Sample Phenotypes With Random Effects," *BMC Bioinformatics*, vol. 15, pp. 314-328, 2014.

[39] M. Kristan, A. Leonardis, and D. Skocaj, "Multivariate Online Kernel Density Estimation with Gaussian Kernels," *Pattern Recognition*, vol. 44, no. 10, pp. 2630-2642, 2011.

[40] R. Jonas, T. Yuan, Y. Liang, J. Jonas, D. Tay, and R. Ellis-Behnke, "The Spider Effect: Morphological and Orienting Classification Of Microglia in Response to Stimuli in Vivo," *PLoS ONE*, vol. 7, no. 2, p. e30763, 2012.

[41] D. Ristanovic, N. Milosevic, I. Stefanovic, D. Maric, and I. Popov, "Cell Image Area as a Tool for Neuronal Classification," *Journal of Neuroscience Methods*, vol. 182, no. 2, pp. 272 - 278, 2009.

[42] D. Ristanovic, B. Stefanovic, and N. Puskas, "Fractal Analysis of Dendrite Morphology using Modified Box-Counting Method," *Neuroscience Research*, vol. 84, pp. 64-67, 2014.

[43] L. Grbatinic, D. Maric, and N. Milosevic, "Neurons from the Adult Human Dentate Nucleus: Neural Networks in the Neuron Classification," *Journal of Theoretical Biology*, vol. 370, no. 0, pp. 11- 20, 2015.

[44] R. Cannon, D. Turner, G. Pyapali, and H. Wheal, "An On-Line Archive of Reconstructed Hippocampal Neurons," *Journal of Neuroscience Methods*, vol. 84, no. 1-2, pp. 49-54, 1998.

[45] G. Ascoli, D. Donohue, and M. Halavi, "NeuroMorpho.Org: A Central Resource for Neuronal Morphologies," *Journal of Neuroscience*, vol. 27, no. 35, pp. 9247-51, 2007.

[46] R. Scorcioni, S. Polavaram, and G. Ascoli, "L-measure: A Web-Accessible Tool for the Analysis, Comparison, and Search of Digital Reconstructions of Neuronal Morphologies," *Nature Protocols*, vol. 3, no. 5, pp. 866-876, 2008.

[47] Y. Lu, L. Carin, R. Coifman, W. Shain, and B. Roysam, "Quantitative Arbor Analytics: Unsupervised Harmonic Co-Clustering of Populations of Brain Cell Arbors Based on L-Measure," *Neuroinformatics*, vol. 13, no. 1, pp. 47-63, 2015.

[48] R. Coifman and S. Lafon, "Diffusion Maps," *Applied and Computational Harmonic Analysis*, vol. 21, no. 1, pp. 5-30, 2006.

[49] C. Parkhurst and W. Gan, "Microglia Dynamics and Function in the CNS," *Current Opinion in Neurobiology*, vol. 20, no. 5, pp. 595-600, 2010.

[50] C. Wu, C. Wen, J. Shieh, and E. Ling, "A Quantitative Study of the Differentiation of Microglial Cells in the Developing Cerebral Cortex in Rats," *Journal of Anatomy*, vol. 182, no. 3, pp. 403-413, 1993.

[51] Z. Soltys, M. Ziaja, R. Pawlnski, Z. Setkowicz, and K. Janeczko, "Morphology of Reactive Microglia in the Injured Cerebral Cortex. Fractal Analysis and Complementary

Quantitative Methods," *Journal of Neuroscience Research*, vol. 63, no. 1, pp. 90-97, 2001.

[52] P. Tresco and B. Winslow, "The Challenge of Integrating Devices into the Central Nervous System," *Critical Reviews Biomedical Engineering*, vol. 39, no. 1, pp. 29-44, 2011.

[53] L. Karumbaiah, T. Saxena, D. Carlson, K. Patil, E. Patkar, R.and Gaupp, M. Betancur, G. B. Stanley, L. Carin, and R. Bellamkonda, "Relationship between Intracortical Electrode Design and Chronic Recording Function," *Biomaterials*, vol. 34, no. 33, pp. 8061-8074, 2013.

[54] D. Szarowski, M. Andersen, S. Retterer, A. Spence, M. Isaacson, H. Craighead, J. Turner, and W. Shain, "Brain Responses to Micro-Machined Silicon Devices," Brain Research, vol. 983, no. 1-2, pp. 23-35, 2003.

[55] J. Seymour, and D. Kipke, "Fabrication of Polymer Neural Probes with Sub-Cellular Features for Reduced Tissue Encapsulation," in *IEEE Engineering in Medicine and Biology Society*, pp. 4606-4609, 2006.

[56] B. Winslow, and P. Tresco, "Quantitative Analysis of the Tissue Response to Chronically Implanted Microwire Electrodes in Rat Cortex," *Biomaterials*, vol. 31, no. 7, pp. 1558-1567, 2010.

[57] J. Goldberger, and S. Roweis, "Hierarchical Clustering of a Mixture Model," *Proc. Neural Information Processing Systems*, pp. 505–512, 2005.

[58] Y. Rubner, C. Tomasi, and L. J. Guibas, "The Earth Mover's Distance as a Metric for Image Retrieval," *International J. Computer Vision*, vol. 40, no. 2, pp. 99-121, 2000.

[59] A.G. Shanbhag, "Utilization of Information Measure as a Means of Image Thresholding," *CVGIP: Graphical Models and Image Processing*, vol. 56, no. 5, pp. 414-419, Sep. 1994.

[60] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, "Introduction to

Algorithms, Second Edition," *MIT Press and McGraw-Hill*, pp. 540–549, 2001.

[61] T. Hastie, R. Tibshirani, and J. Friedman, "The Elements of Statistical Learning," *Springer-Verlag*, pp. 523-526, 2009.

[62] P. Qiu, and S. K. Plevritis, "Treevis: A Matlab-Based Tool for Tree Visualization", *Computer Methods and Programs in Biomedicine*, vol. 109, no. 1, pp. 75-76, 2013.

[63] L. Maaten, and G. Hinton, "Visualizing Data Using T-Sne," *J. of Machine Learning Research*, vol. 9, pp. 2579-2605, 2008.

[64] N. Heidenreich, A. Schindler, and S. Sperlich, "Bandwidth Selection for Kernel Density Estimation: A Review of Fully Automatic Selectors," *AStA Advances in Statistical Analysis*, Vol. 97, no. 4, pp. 403-433, 2013.

[65] M. P. Wand, and M. C. Jones, "Kernel Smoothing," *Chapman & Hall/CRC*, 1995.

[66] D. E. Pollard, "A User's Guide to Measure Theoretic Probability," *Cambridge University Press*, 2002.

[67] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, "Estimation with Applications to Tracking and Navigation," *John Wiley & Sons, Inc.*, Ch. 11, pp. 438–440, 2001.

[68] G. J. Szekely, M. L. Rizzo, and N. K. Bakirov, "Measuring and Testing Dependence by Correlation of Distances," *The Annals of Statistics*, vol. 35, no. 6, pp. 2769–2794, 2007.

[69] D. N. Reshef, Y. A. Reshef, H. K. Finucane, S. R. Grossman, G. McVean, P. J. Turnbaugh, E. S. Lander, M. Mitzenmacher, and P. C. Sabeti, "Detecting Novel Associations in Large Data Sets," *Science*, vol. 334, no. 6062, pp. 1518-1524, 2011.

[70] N. Simon, and R. Tibshirani, "Comment on 'Detecting Novel Associations in Large Data Sets' by Reshef et al., Science Dec 16, 2011", *Cornell University Library: Methodology*, arXiv:1401.7645, 2014.

[71] M. L. Whitfield, G. Sherlock, A. J. Saldanha, J. I. Murray, C. A. Ball, K. E.

Alexander, J. C. Matese, C. M. Perou, M. M. Hurt, P. O. Brown, and D. Botstein, "Identification of Genes Periodically Expressed in the Human Cell Cycle and Their Expression in Tumors," Molecular Biology of the Cell, vol. 13, no. 6, pp. 1977–2000, 2002.

[72] M. E. Hystad, J. Myklebust, T. Bo, E. Sivertsen, E. Rian, L. Forfang, E. Munthe, A. Rosenwald, M. Chiorazzi, I. Jonassen, L. M. Staudt, and E. B. Smeland,"Characterization of Early Stages of Human B Cell Development by Gene Expression Profiling," J. of Immunology, vol.179, no. 6, pp. 3662-3671, 2007.

[73] Troysanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R.B. Altman, "Missing Value Estimation methods for DNA Microarrays," Bioinformatics, vol.17, no. 6, pp.520-525, 2001.

[74] R. Scorcioni, S. Polavaram, and G. Ascoli, "L-Measure: A Web-Accessible Tool for the Analysis, Comparison and Search of Digital Reconstructions of Neuronal Morphologies," *Nature Protocols,* vol. 3, no. 5, pp.866-76, 2008.

[75] H. Cuntz, F. Forstner, A. Borst, and M. Häusser, "One Rule to Grow Them All: A General Theory of Neuronal Branching and Its Practical Application," *PLoS Comput Biol*, vol. 6, no. 8: e1000877, 2010.

[76] J. Luisi, A. Narayanaswamy, Z. Galbreath, and B. Roysam. "The Farsight Trace Editor: An Open Source Tool for 3-D Inspection and Efficient Pattern Analysis Aided Editing of Automated Neuronal Reconstructions," *Neuroinform*, vol. 9, pp. 305-315, 2011.

[77] K. Ohsawa, and S. Kohsaka, "Dynamic Motility of Microglia: Purinergic Modulation of Microglial Movement in the Normal and Pathological Brain," *Glia*, vol. 59, no. 12, pp. 1793-1799, Dec. 2011.

[78] C. Bjornsson, S. Oh, Y. Al-Kofahi, Y. Lim, K. Smith, J. Turner, S. De, B. Roysam, W. Shain, and S. Kim, "Effects of Insertion Conditions on Tissue Strain and Vascular Damage During Neuroprosthetic Device Insertion," *Journal of Neural Engineering*, vol.

3, no. 3, pp. 196-207, Sep. 2006.

[79] I. Kitware, "The Visualization Toolkit User's Guide", 11th ed. *Kitware Inc.*, 2010.

[80] C. Tsai, J. Lister, C. S. Bjornsson, C. Jornsson, K. Smith, W. Shain, C. Barnes, and B. Roysam, "Robust, Globally Consistent and Fully Automatic Multi-Image Registration and Montage Synthesis for 3-D Multi- Channel Images," *Journal of Microscopy*, vol. 243, no. 2, pp. 154-171, 2011.

[81] M. Megjhani, N. Villamizar, A. Merouane, Y. Lu, A. Mukherjee, K. Trett, P. Chong, C. Harris, W. Shain, and B. Roysam, "Population-Scale Three-Dimensional Reconstruction and Quantitative Profiling of Microglia Arbors," *Bioinformatics*, doi: 10.1093/bioinformatics/btv109, 2015.

[82] Y. Al-Kofahi, W. Lassoued, W. Lee, and B. Roysam, "Improved Automatic Detection and Segmentation of Cell Nuclei in Histopathology Images," *IEEE Transactions on Biomedical Engineering*, vol. 57, no. 4, pp. 841-852, 2010.

[83] C. S. Bjornsson, G. Lin, Y. Al-Kofahi, A. Narayanaswamy, K. L. Smith, W. Shain, and B. Roysam, "Associative Image Analysis: A Method for Automated Quantification of 3D Multi-Parameter Images of Brain Tissue," *Journal of Neuroscience Methods*, vol. 170, no. 1, pp. 165-178, 2008.

[84] R. Padmanabhan, V. Somasundar, S. Griffith, J. Zhu, D. Samoyedny, K. S. Tan, J. Hu, X. Liao, L. Carin, S. Yoon, K. Flaherty, R. DiPaola, D. Heitjan, P. Lal, M. Feldman, B. Roysam, and W. Lee, "An Active Learning Approach for Rapid Characterization of Endothelial Cells in Human Tumors," *PLoS ONE*, vol. 9, no. 3, p. e90495, 2014.

[85] N. Rey-Villamizar, V. Somasundar, M. Megjhani, Y. Xu, Y. Lu, R. Padmanabhan, K. Trett, W. Shain, and B. Roysam, "Large-Scale Automated Image Analysis for Computational Profiling of Brain Tissue Surrounding Implanted Neuroprosthetic Devices Using Python," *Frontiers in Neuroinformatics*, vol. 8, no. 39, doi: 10.3389/fninf.2014.00039, 2014.

[86] S. Li, and B. Acton, "Active Contour External Force Using Vector Field

Convolution for Image Segmentation," *IEEE Transactions on Image Processing*, vol. 16, no. 8, pp. 2096-2106, 2007.

[87] A. Mariano, D. Lee, A. Gerstlauer, and D. Chiou, "Hardware and Software Implementations of Prim's Algorithm for Efficient Minimum Spanning Tree Computation," in *Embedded Systems: Design, Analysis and Verification*. Springer Berlin Heidelberg, vol. 403, pp. 151-158. 2013.

[88] J. Sethian, "A Fast Marching Level Set Method for Monotonically Advancing Fronts," *Proceedings of the National Academy of Sciences*, vol. 93, no. 4, pp. 1591-1595, 1996.

[89] A. S. Chiang, C. Y. Lin, C. C. Chuang, H. M. Chang, C. H. Hsieh, C. W. Yeh, C. T. Shih, J. J. Wu, G. T. Wang, Y. C. Chen, C. C. Wu, G. Y. Chen, Y. T. Ching, P. C. Lee, C. Y. Lin, H. H. Lin, C. C. Wu, H. W. Hsu, Y. A. Huang, J. Y. Chen, H. J. Chiang, C. F. Lu, R. F. Ni, C. Y. Yeh, and J. K. Hwang, "Three-Dimensional Reconstruction of Brain-Wide Wiring Networks in Drosophila at Single-Cell Resolution", *Current Biology*, vol. 21, no. 1, pp. 1-11, 2010.

[90] D. T. Fox, and A. C. Spradling, "The Drosophila Hindgut Lacks Constitutively Active Adult Stem Cells but Proliferates in Response to Tissue Damage," *Cell Stem Cell*, vol. 5, no. 3, pp. 290–297, 2009.