TPFlow: Progressive Partition and Multidimensional Pattern Extraction for Large-Scale Spatio-Temporal Data Analysis

Dongyu Liu, Panpan Xu, and Liu Ren

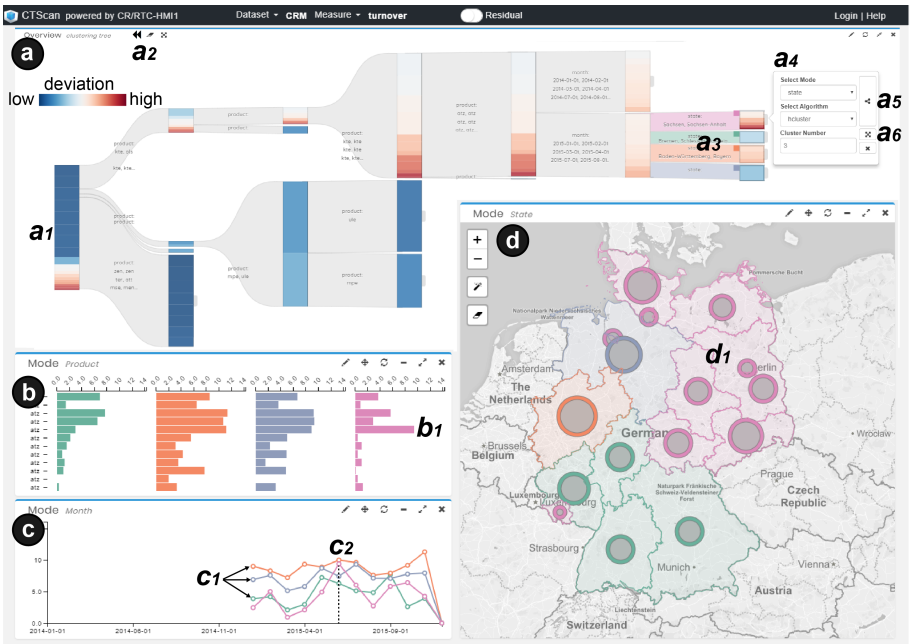

Fig. 1. The user interface of TPFlow. ⓐ Tree View starts with a root node (***a1***) representing the original data and visualizes the overall data partitioning process. Every other node on the tree represents a subset of data. The system supports a steerable and iterative workflow by allowing analysts to directly interact with every node (***a2***, ***a4***, ***a5***, ***a6***). ⓑⓒⓓ Each individual chart visualizes the latent patterns extracted by our tensor-based model. The analysts select multiple nodes highlighted in different colors on the Tree View (***a3***) to compare the patterns across different subsets of data. One interesting pattern here is that the pink states (***d1***), located mostly in the northeast region of Germany, perform significantly different from the other states (***b1***, ***c1***, ***c2***). Please refer to the details in Sec. 5.1.

Abstract—Consider a multi-dimensional spatio-temporal (ST) dataset where each entry is a numerical measure defined by the corresponding temporal, spatial and other domain-specific dimensions. A typical approach to explore such data utilizes interactive visualizations with multiple coordinated views. Each view displays the aggregated measures along one or two dimensions. By brushing on the views, analysts can obtain detailed information. However, this approach often cannot provide sufficient guidance for analysts to identify patterns hidden within subsets of data. Without a priori hypotheses, analysts need to manually select and iterate through different slices to search for patterns, which can be a tedious and lengthy process. In this work, we model multidimensional ST data as tensors and propose a novel *piecewise rank-one tensor decomposition* algorithm which supports automatically slicing the data into homogeneous partitions and extracting the latent patterns in each partition for comparison and visual summarization. The algorithm optimizes a quantitative measure about how faithfully the extracted patterns visually represent the original data. Based on the algorithm we further propose a visual analytics framework that supports a top-down, progressive partitioning workflow for level-of-detail multidimensional ST data exploration. We demonstrate the general applicability and effectiveness of our technique on three datasets from different application domains: regional sales trend analysis, customer traffic analysis in department stores, and taxi trip analysis with origin-destination (OD) data. We further interview domain experts to verify the usability of the prototype.

Index Terms—Spatio-temporal data, tensor decomposition, interactive exploration, automatic pattern discoveries

1 Introduction

Interactive visualization is the key technique for explorative analysis of large-scale multidimensional spatio-temporal (ST) data. However, as the size and the dimensionality of the data increase, more scalable techniques are needed. Data aggregation performs binning and rollup operations on the original data to reduce the number of visual items displayed, hence improving the perceptual scalability of the visualization techniques. The most popular approaches for multidimensional ST data visualization display aggregated measures along each individual dimension of interest (e.g. spatial, temporal, categorical, numerical) with coordinated views [36, 41, 45]. For example, consider a traffic volume dataset with the schema $(day, hour, region) \rightarrow traffic\_volume$. A typical ST visualization displays the total traffic distributed over hours,

- *Dongyu Liu is with the Hong Kong University of Science and Technology. This work was done during his internship at Bosch Research North America. E-mail: dliuae@cse.ust.hk.*
- *Panpan Xu and Liu Ren are with Bosch Research North America, Sunnyvale, CA. E-mail: panpan.xu, liu.ren@us.bosch.com.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

days and geographical regions on different views. With brushing and linking, analysts can directly specify queries on the views, select and highlight a subset of data for detailed examination, e.g., select certain days and hours and look at the geospatial distribution of the traffic volume in the corresponding time interval. This approach is generic and scalable with respect to the dimensionality of the data. Many techniques have also been proposed to enhance its real-time interactivity for large data through pre-computation of data cubes [36, 41], parallel processing (with GPU) [41] or a clever arrangement of data items in the storage [45].

However, high-level aggregated overviews are not sufficient in providing visual cues or guidances for analysts to identify patterns hidden within subsets of data [21]. For instance, traffic volume may exhibit consistent but different hourly patterns during weekdays and weekends or in residential and commercial areas. The aggregated values displayed in the charts are unable to direct the analysts to such insights, esp. without a priori hypotheses. With brushing and linking, fine-grained information can be revealed. However, the analysts have to manually select different slices, compare and correlate different subsets, and search for patterns. This can be an extremely challenging task.

Our approach models multidimensional ST data cubes as tensors. We propose a novel algorithm named *piecewise rank-one tensor decomposition* which automatically identifies an optimal way to slice the tensor into homogeneous partitions and extracts the patterns for each partition along spatial, temporal and other domain-specific dimensions. We visually summarize each partition with a set of simple charts (e.g., line charts, bar charts, bubble map/ heat-map) to display the extracted patterns along different dimensions. The algorithm is grounded on a quantitative measurement about how faithfully the visualization represents the original data. The algorithm aims at alleviating the analysts' burden to manually iterate through different slices in the data for pattern discovery.

Building upon the basic algorithm, we propose TPFlow (short name for **T**ensor **P**artition **Flow**) that supports a top-down, progressive partitioning workflow for level-of-detail multidimensional ST data exploration. The system guides analysts to iteratively partition the data, visualizes the extracted multidimensional patterns and supports comparative analysis across different subsets. The framework tightly couples visual representation, computation, and user interaction to guide the analysts to the salient patterns hidden within the data.

We present example usage scenarios on three real-world datasets which illustrate the general applicability and effectiveness of our approach: New York taxi traffic data, product sales data over different geographical regions, and customer traffic data over different zones in a brick-and-mortar department store. The system is capable of addressing a set of drastically different analytical tasks in these application domains. We also conduct expert interview to validate the usability and effectiveness of the approach.

To sum up, our contributions are:

- A novel *piecewise rank-one tensor decomposition* algorithm which automatically seeks for and recommends the best way to slice multidimensional ST data and extract the multidimensional trend for comparison and visual summarization.
- A novel visual analytics framework that supports a progressive partitioning and level-of-detail explorative analysis workflow of ST data. The system incorporates a set of visualization and interaction designs to facilitate pattern discovery, comparison and verification.
- Three example usage scenarios on real-world ST data across a wide range of application domains and analytic tasks.

2 Related Work

In the past years, researchers have spent great effort to develop interactive visualization systems for ST data analysis. There are many excellent surveys [3, 7, 14, 40, 66, 67]. In this section, we focus on the most relevant work.

2.1 Visualizing large-scale multidimensional ST data

Scalability. Large-scale ST dataset usually contains millions or even more data records. Mapping all of them to the screen is usually infeasible or ineffective. A number of methods (pixel-oriented methods [31], spatial transformation [54], alpha blending [31], etc.) have been proposed to alleviate this issue. However, since these methods still draw all records, they will hit bottleneck eventually. A more popular approach is to employ data reduction techniques such as filtering, sampling and aggregation. However, to develop effective filtering and sampling strategies, prior knowledge is usually required, otherwise important structures or outliers may remain hidden to analysts.

Many recent systems for large scale ST data exploration aggregate the measures to present an overview of the data along different dimensions. They usually leverage multiple views for presenting multidimensional data and support visual queries across different views through brushing and linking. Examples of research along this line include [23, 49] and [29, 30, 36, 41, 45]. To enable visual query on extremely large scale data at an interactive rate, many techniques have been developed recently such as imMens [41], Datavore [30], DICE [29], Nanocubes [36] and Hashedcubes [45]. Nanocubes [36] and Hashedcubes [45] pre-aggregate data to speed up queries on ST data cubes, while imMens [41] leverages GPUs to achieve fast queries over datasets at large scale. In despite of those recent advances, these techniques still require analysts to iterate through different query parameters to identify the patterns that are associated with a particular subset of the data. With no prior knowledge or hypotheses, analysts have to compare and correlate different slices of data repeatedly in search for patterns, which can be a tedious and lengthy process.

We propose a novel intelligent dataset subdivision algorithm which can help analysts automatically partition the data into meaningful subsets with similar trends/patterns along multiple dimensions for further observation and comparison. The algorithm allows analysts to refine the subdivision through hierarchical, iterative partitioning on different dimensions. In this way, the system provides detail-on-demand for explorative analysis.

Multidimensionality. Existing approaches for multidimensional ST data visualization can be classified into two categories, namely, multiple coordinated views visualization and multivariate visualization. The previous one provides several separated views to display multidimensional information, each view displays one or two dimensions and the views are linked together for coordinated analysis [38, 58–60]. The latter one focuses on simultaneously encoding spatial, temporal and other attributes in one view in a more compact manner. For example, the most common forms for the multivariate visualization are ring maps [65], glyphs on maps [4], space-time cube [35], and small multiples [9]. One critical limitation of these visualization techniques is that the visual channels will be exhausted to encode multivariate data attributes eventually.

In practice, the combination of both methods has been successfully used in many visual analytics systems for analyzing various ST data [3, 14]. In this work, we use tensor-based model to capture interdependencies along multiple dimensions and then summarize the patterns (i.e., low dimension vectors) for each dimension accordingly. Thus, the patterns in each dimension can be visualized in a relatively simple manner (no concern on the exhaustion of visual channels). By showing the patterns of every dimension in the form of multiple coordinated views, the patterns across each dimension can be easily observed and compared.

2.2 Mining large-scale multidimensional ST data

Using data mining methods to automatically extract patterns and derive insights out of large-scale multidimensional ST data is popular [3, 66]. However, this is notably hard, due to the existence of complex dependencies among space, time, and other domain-specific variables.

Various statistic models [1, 10, 17, 18, 28] have been proposed to automatically summarize the temporal trends or location distribution patterns. However, most of them lack the capability to directly deal with the correlations among multiple dimensions in a scalable manner.

Recently, tensor-based methods [34, 43] have been successfully applied for ST data analysis, for their inherent capability to model multifaceted data. The canonical polyadic (CP) decomposition (a.k.a., PARAFAC or CANDECOMP) [13,26] is one of the most popular tensor

analysis methods, which decomposes a tensor into a sum of rank-one components to describe the latent structures in the data. The method has been used to analyze a variety of ST data, including location-based social network (LBSN) data [25,42], mobile phone GPS data [22], bike-sharing data [61] and traffic flow data [8,15,53]. For example, Takeuchi et al. [53] model traffic flow data as a three-dimensional tensor with each dimension representing days, hours, and geographical locations, respectively. The tensor then is decomposed into three non-negative rank-1 factors (i.e., feature descriptors) to describe the daily, hourly and spatial trend/distribution patterns.

However, all these methods have not fully examined the interpretability of the tensor decomposition results in the spatio-temporal context. Cao et al. [12] present a visualization system which is built on CP decomposition methods to analyze traffic flow data, but the system is designed for anomaly detection and is not suitable for general spatio-temporal pattern discovery (e.g., temporal trend patterns or spatial distribution patterns). More importantly, all the aforementioned methods cannot support automatically identifying the patterns that exist on subsets of data, which is a fundamental task for ST data analysis [3].

Clustering is another powerful tool that has been usually utilized to partition data by grouping similar objects [39]. Van Wijk and Van Selow [55] use a hierarchical clustering method to group days with similar hourly traffic variations. The method is limited to matrix shaped data with two dimensions (days and hours in this instance). Andrienko et al. [2] use self-organizing map (SOM) to perform clustering on time series and spatial locations to generate clusters of places of similar time variation or clusters of time intervals of similar spatial distributions. However, SOM requires the appropriate feature vector selection and careful parameters setting. Besides, their system cannot support human-in-the-loop analysis for iteratively tuning the clustering result. Doraiswamy et al. [21] model ST data as time-varying scalar functions and use topology analysis method to identify spatio-temporal events in data slices. The approach cannot support multidimensional pattern analysis and progressive data partition.

In this work, we present a unified framework that lays emphasis on effectively identifying and interpreting the spatio-temporal patterns that exist on data subsets. In particular, we propose a novel tensor-based algorithm, based on which we present a semi-automatic and steerable approach for efficiently identifying sub-groups of data with coherent multidimensional patterns and performing comparative analysis.

3 ALGORITHM

In this section, we first describe how to model multidimensional ST data as tensors, and then introduce a basic tensor decomposition method to extract the patterns along different dimensions (Fig. 2). Subsequently, we propose a novel tensor partitioning algorithm to facilitate the exploration of patterns that exist in subsets of data.

3.1 Modeling spatio-temporal data as tensors

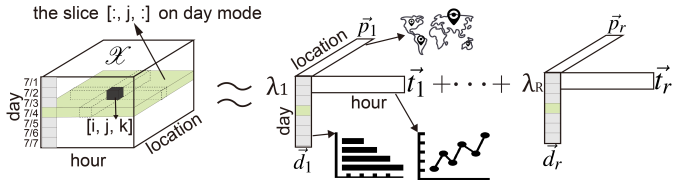

Fig. 2. Model traffic flow data as a three-dimensional tensor (left) and perform successive rank-one CP decomposition (right). The 1st loading vectors $(\vec{p}_1,\vec{d}_1,\vec{t}_1)$ visually summarize the distribution on each dimension.

A tensor (denoted as $\mathscr{X}$) is a multi-dimensional array, which can meaningfully represent a wide range of spatio-temporal datasets. For example, as illustrated in Fig. 2 (left), a traffic flow dataset can be modeled as a three-dimensional tensor $\mathscr{X} \in \mathbb{R}_{\geq 0}^{\|P\|\times\|D\|\times\|T\|}$, where P is the locations, D is the days and T is the hours. The element $\mathscr{X}[i,j,k]$ represents the traffic volume (e.g. total number of vehicles) at location i on the j-th day during hour k.

In general, we denote a m-dimensional non-negative tensor as $\mathscr{X} \in \mathbb{R}_{\geq 0}^{\|D_1\|\times\cdots\times\|D_m\|}$, where D_n is the set of all possible values at the n-th dimension. For the ease of discussion, we will continue to use the three-dimensional tensor with traffic flow data to illustrate the main idea of our approach without loss of generality. For the rest of the paper, we will use **dimension** and **mode** interchangeably, same as in the literature of tensor decomposition [34]. Some tensor related terminologies are described below:

Outer product: the operator $\otimes$ denotes the outer product of two or more vectors. The outer product of three vectors $\vec{a}\otimes\vec{b}\otimes\vec{c}$ is a three-dimensional tensor $\mathscr{X}^{\|\vec{a}\|\times\|\vec{b}\|\times\|\vec{c}\|}$ with entries $\mathscr{X}[i,j,k] = \vec{a}[i]\cdot\vec{b}[j]\cdot\vec{c}[k]$.

Rank-one tensor: a tensor $\mathscr{X}$ is *rank-one* if it can be expressed as the outer product of vectors ($\mathscr{X} = \vec{a}\otimes\vec{b}\otimes\vec{c}$).

Some additional denotations are listed in Table. 1.

Table 1. Denotations

Symbol	Description
$\mathscr{X}[i,:,:]$	a **slice** of $\mathscr{X}$ by fixing the index on one mode, similar for $\mathscr{X}[:,j,:]$ and $\mathscr{X}[:,:,k]$
$I \subseteq \{0,1,2,,\|P\|-1\}$	a **subset of the indices** on one mode, similar for J and K
$\mathscr{X}[I,:,:]$	a **sub-tensor** composed by stacking the slices $\mathscr{X}[i,:,:], i \in I$, similar for $\mathscr{X}[:,J,:]$ and $\mathscr{X}[:,:,K]$

To ensure the interpretability of the extracted patterns, we apply successive rank-one CP decomposition methods [16,33,64] instead of traditional ones (i.e., PARAFRAC/CANDECOMP) [13,26] to factorize the original tensor. The successive approaches first determine the best rank-one approximation for $\mathscr{X}$: $\mathscr{X} \approx \widehat{\mathscr{X}} = \lambda_1\vec{p}_1\otimes\vec{d}_1\otimes\vec{t}_1$. Then they compute the rank-one approximation for the residual $\mathscr{X} - \widehat{\mathscr{X}}$ as the 2-nd loading vectors $(\vec{p}_2,\vec{d}_2,\vec{t}_2)$. For iteration i, we optimize the following cost function:

$$cost = \|\mathscr{X}_{current\_resdisual} - \lambda_i\cdot\vec{p}_i\otimes\vec{d}_i\otimes\vec{t}_i\| \quad (1)$$

where $\|\cdot\|$ denotes the Frobenius norm. The iteration continues until a certain stopping criterion is met (e.g., improve little between two subsequent iterations or achieve a maximum number of iterations).

Finally, the original tensor is approximated with a set of **rank-one tensor component**s (Fig. 2(right)): $\mathscr{X} \approx \sum_{r=1}^{R}\lambda_r\vec{p}_r\otimes\vec{d}_r\otimes\vec{t}_r$, where $\vec{p}_r$, $\vec{d}_r$ and $\vec{t}_r$ are the normalized ***r*-th loading vectors** (i.e., patterns) in the ***r*-th component**, and $\lambda_1 \geq \lambda_2 \ldots \geq \lambda_R$ indicate the relative strength of the corresponding patterns. The 1-st loading vectors $(\vec{p}_1,\vec{d}_1,\vec{t}_1)$ capture the most prominent variations in the data and appear much easier to explain compared with the remaining loading vectors that capture the variations in residual tensors. Thus, by plotting the 1-st loading vectors with basic visualizations (Fig. 2 right and col #1 in Fig. 4), we can easily summarize tensorial data with three or more dimensions.

Notice that $(\vec{p}_1,\vec{d}_1,\vec{t}_1)$ are obtained by optimizing the cost function, $\|\mathscr{X} - \lambda_1\cdot\vec{p}_1\otimes\vec{d}_1\otimes\vec{t}_1\|$, to capture as much variation in the data as possible. The cost function can be naturally viewed as a **quantitative measurement** about how **faithfully** the visual summary represents the original data. In the ideal case when $cost = 0$, the original tensor data can be fully recovered by integrating the spatial, daily and hourly variations ($\mathscr{X}[i,j,k] = \lambda_1\cdot\vec{p}_1[i]\cdot\vec{d}_1[j]\cdot\vec{t}_1[k]$). In this situation, $\vec{p}_1$, $\vec{d}_1$, $\vec{t}_1$ is sufficient to represent the original data without loss of information. Besides, to further facilitate interpretation, we add additional constraints to ensure non-negativity of the 1-st loading vectors $(\vec{p}_1,\vec{d}_1,\vec{t}_1)$, where the alternating least square (ALS) algorithm [16] is used.

3.2 Piecewise rank-one tensor decomposition

The 1-st loading vectors ($\vec{p}_1$, $\vec{d}_1$, $\vec{t}_1$) of $\mathscr{X}$ give an overview of the entire dataset. However, it is seldom the case that they can fully capture

the variations in the data. For example, the traffic-volume may have different geospatial distribution on weekdays and weekends. Since $\vec{p_1}$ averages over all the days, such information is unfortunately lost. Indeed, visual summarization of data can introduce potential discrepancies between the visualization and the dataset [19].

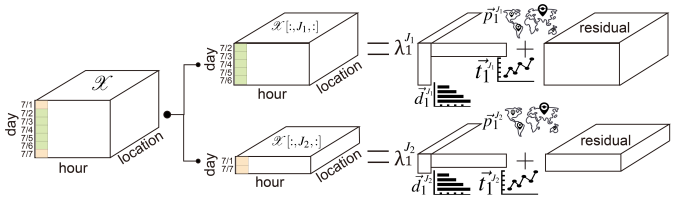

Fig. 3. The piecewise rank-one tensor decomposition automatically identifies two groups of days and partitions $\mathscr{X}$ into two sub-tensors accordingly. It extracts the 1st loading vectors $(\vec{p_1}, \vec{d_1}, \vec{t_1})$ through rank-one tensor decomposition to visually summarize and compare different subsets of data.

To overcome this limitation, we propose a novel approach which we refer to as **piecewise rank-one tensor decomposition**. As illustrated in Fig. 3, the method automatically detects sub-tensors with similar variations along spatial, temporal and other domain-specific dimensions. Given a selected mode and the number of parts to create, it performs simultaneous tensor partitioning and multi-mode pattern extraction by solving the following optimization problem:

$$\underset{P,\ \|P\|=k}{argmin} \sum_{J \in P(\{1,\ldots,k\})} \|\mathscr{X}[:,J,:] - \lambda_1^J \cdot \vec{p_1}^J \otimes \vec{d_1}^J \otimes \vec{t_1}^J\| \qquad (2)$$

where P is a partition of the indices on a specified mode and k is the number of parts in P. $\mathscr{X}[:,J,:]$s are the sub-tensors created by the partition P. $\vec{p_1}^J$, $\vec{d_1}^J$ and $\vec{t_1}^J$ are the 1st loading vectors of the sub-tensor $\mathscr{X}[:,J,:]$. Similar to Equation. 1, the optimization goal measures how faithfully the loading vectors (hence the visualizations) represent the original data.

However, directly optimizing Eq. 2 is a non-trivial task due to the combinatorial number of different ways to partition the data. We propose an algorithm that leverages the 1-st to r-th loading vectors produced by successive rank-one tensor decomposition to create low dimensional feature descriptors for the selected mode and then apply clustering algorithms to automatically detect the partitions.

More concretely, let's assume that we select *days* as the dimension to perform partition on. First, a feature vector is created for each day j from the corresponding entries in the loading vectors $x_j = (\vec{d_1}[j], \ldots, \vec{d_R}[j])$. Given the feature vectors, we can apply a variety of clustering algorithms, including k-means, hierarchical clustering [24], or OPTICS [6] to cluster the days. Based on the clustering result, we create a partition of the indices on the selected mode and generate the corresponding sub-tensors, and then perform rank-one non-negative CP decomposition on each individual sub-tensor to obtain the latent patterns (see illustration in Fig. 3). Those latent patterns are displayed in the visualization to summarize the original multidimensional ST data. The patterns extracted from different sub-tensors can be further compared for fine-grained analysis.

Grouping similar days into one sub-tensor can significantly facilitate recovering tensor elements more accurately with the 1-st loading vectors [42]. This also helps optimize the cost function effectively (please refer to algorithm evaluation in Sec. 6), hence summarizing the data more faithfully.

4 TPFlow

We introduce a prototype system TPFlow which is built on the piecewise rank-one tensor decomposition method to support explorative analysis of multi-dimensional ST data. The system tightly couples computation, visual representation and user interaction to support a top-down, divide-and-conquer analytics workflow. An overview is given first, and progressively more details and fine-grained patterns are revealed [50] as analysts iteratively partition the data into smaller sub-tensors. In this section, we first formulate a set of requirements and then continue to describe the detailed visual and interaction designs.

4.1 Design Requirements

For eight months, we worked closely with four experts from two different application domains, each including two experienced data scientists. The first application is customer traffic analytics for brick-and-mortar retailers. The data schema is $(day \times hour \times retail\_area\_id) \rightarrow traffic\_volume$, describing the spatial and temporal variation of the visitor traffic in a large department store. The second one is regional sales trend analytics for a company's product portfolio. The data schema is $(month \times product\_id \times region) \rightarrow sales\_volume$, encoding the sales volume at different geographical locations for a variety of products. Despite that the diverse analytical tasks arise in these two application domains: market segmentation [32, 46], product demand trend detection and forecasting [32, 46], periodical traffic pattern analysis [56] and etc., we identify a common need for detecting clusters and trends in the data. Piecewise rank-one tensor decomposition (Section 3) is a generic method that can help with these tasks. However, simply applying the algorithm and visualizing the latent trends/patterns alone are not sufficient due to the following reasons:

- Nonoptimal parameter settings (e.g. the number of partitions) may result in poor fit of the data. The patterns produced by the algorithm can deviate significantly from the underlying data distribution and it is not advisable to blindly follow the results.
- Cluster and trend analysis are essentially explorative analytical tasks [3, 48]. To reach valuable insights, analysts have to try out different modes for partitioning and a variety of clustering algorithms with different parameter settings.
- Even with no prior knowledge on the dataset, analysts may gradually form a set of hypothesis in the exploration process [5] and prefer more focused analysis at a later stage with more control over selecting which subset of data to be fed into the algorithm, and specify other constraints if necessary.

Thus, we identified the need for a highly interactive ST data exploration environment which could support *iterative* and *steerable* data partitioning/clustering and allow analysts to assess the reliability of the extracted latent patterns with *confidence*. To this end, we had several rounds of back-and-forth discussion and system prototyping together with the two groups of experts and distilled the following concrete requirements for a generic ST data analysis system.

R1 **Display the latent patterns on multiple dimensions.** We need to design a suite of charts to display the latent vectors (i.e. 1-st loading vectors) on different dimensions (spatial, temporal, categorical and numerical dimensions) for an overview of the data.

R2 **Enable comparative analysis.** Analyzing the similarities and differences across different data partitions/clusters is essential for understanding their unique characteristics to support further decision making.

R3 **Visualize the deviation of the raw data from the extracted patterns.** As discussed, the extracted patterns may not be able to reflect the underlying data faithfully due to non-optimal parameter setting or other factors. We need to provide visual cues for such discrepancies such that analysts can make informed decisions about whether the pattern is reliable or further refinement of the clusters is necessary.

R4 **Support steerable data partitioning/clustering**. The partitioning mechanisms should be flexible and steerable and can be easily adapted to various applications for different analytical tasks with varying degrees of user intervention. Besides that, obtaining meaningful partitions often requires several rounds of iteration.

R5 **Support tracking and provenance of partition history.** To support iterative partition refinement and a progressive, top-down analysis workflow, the system should enable analysts to track the steps they have taken to reach the insight, keep the contextual information about which subset they are looking at and reverse the data partition steps if necessary.

4.2 Visualization and Interaction

In this section, we introduce the main visualization components, user interactions, and the detailed analytic workflow supported in TPFlow.

4.2.1 Basic charts

In the system, we employ several basic chart types to display the latent trends/distributions on spatial (geographical), temporal, categorical and numerical dimensions (R1). Binning on continuous domains (e.g., geographical, temporal, and numerical domains) is performed before constructing the tensor. We assume that spatial (geographical) locations are grouped into nominal units such as states, cities or zones, temporal values are binned into days, hours, months or years, and numerical values are grouped into equal-sized and adjacent intervals.

Fig. 4 (col #1) lists the visualizations included in TPFlow to depict the latent patterns on different dimensions. The bar chart (row #1) and line chart (row #2) directly visualize the values in the 1st loading vectors. Notice that both the bar chart and the line chart can be used to visualize the distribution over temporal dimension. The difference is that bar chart emphasizes individual values while line chart performs better for trend detection [44]. For thematic map, we choose bubble map (row #3) and use the area of the circles to encode the corresponding values in the 1st loading vectors. The flow map (row #4) shows OD (origin-destination) data which describe spatial movements such as taxi trips with pickup and dropoff locations. It draws curves connecting the origins and destinations of movement. When a pickup/dropoff region is selected, the map now will display the flow from/to the selected region and the corresponding values in the 1st loading vecotrs are encoded with the area of the circles on the map (the same with bubble map).

4.2.2 Design for visual comparison

To support comparative analysis (R2), we extend the basic charts (Fig. 4 col #2) to display multidimensional trends in different partitions (the sub-tensors in Fig. 3). Visual designs for comparison tasks can be categorized into three groups based on how they organize the display [37]: *juxtaposition* (i.e., small multiples, present each data subsets separately), *superposition* (present multiple data subset in a shared coordinate system) and *explicit encoding* (directly encode data difference). In our system, we prioritize the usage of *superposition* since it is more space efficient and co-location facilitates visual comparison. *Juxtaposition* is applied whenever overlaying data on the same visualization will cause visual clutter.

As displayed in Fig. 4 (col #2), for the bar chart, we extend it to parallel bar charts (juxtaposition) to enable comparison. For the line chart, we replace it with a multi-series line chart (superposition) to highlight the temporal trend differences. For the bubble map, we introduce a multivariate glyph design (Fig. 5c) which can be embedded on a map (superposition). Another possibility is placing a set of maps side by side as small multiples (juxtaposition). Since sharing spatial context facilitates comparative analysis, we overlay multiple data partitions on the same map. We have also considered several alternative glyph designs to encode multivariate data on maps, including donut charts, pie charts, and polar area charts. Considering that the polar area charts do not usually have regular shapes and it can be difficult to compare their areas, we are left choosing between donut charts and pie charts. Pie charts are special cases of donut charts where the radiuses of the inner circles are zero. Donut charts and pie charts are comparable in their performance for estimating the relative proportions [51]. For comparing the sizes of different glyphs on a map, the inner circles (extra visual cues) in the donut charts can further help [11]. Finally, the domain experts decided to choose donut charts.

In Fig. 4, there is an additional col #4. It shows the chart status when the corresponding dimension is partitioned. For example on the map, the circles are assigned with different colors showing the regions in different partitions.

4.2.3 Encode deviation

As discussed in Section 4.1, we need to provide visual cues for analysts to assess the reliability of the patterns which are automatically generated by the algorithm (R3). Thus, we compute the deviation of the raw data

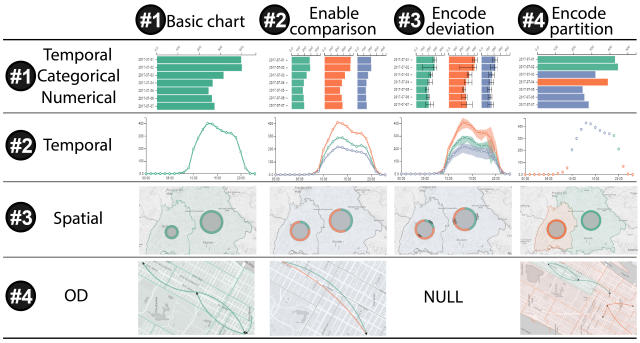

Fig. 4. Col #1 depicts the basic charts to show the patterns on different types of dimensions; col #2 extends the basic charts to support comparative analysis; col #3 encodes the raw data's deviation from the depicted patterns; col #4 presents the partitioning results.

from the latent trends, and display it on top of the entries (Fig. 4 col #3). For each element $\mathscr{X}[i,j,k]$ in the original tensorial data, we compute its deviation percentage from the element recovered using the latent rank-one factors $\widehat{\mathscr{X}}[i,j,k] = \vec{p}[i] \cdot \vec{d}[j] \cdot \vec{t}[k]$:

$$\Delta[i,j,k] = \frac{\widehat{\mathscr{X}}[i,j,k] - \mathscr{X}[i,j,k]}{\mathscr{X}[i,j,k]} \tag{3}$$

We thus obtain a residual tensor $\Delta^{\|P\|\times\|D\|\times\|T\|}$. It is not quite feasible to display all the entries in Δ (otherwise we can already directly display the original data). Therefore, we propose to summarize the deviation along each mode with descriptive statistics. More specifically, take the traffic flow data as an example, to summarize the deviation from the hourly patterns, we compute quartiles $Q_{1/4}(\Delta[:,:,k])$, $Q_{1/2}(\Delta[:,:,k])$, $Q_{3/4}(\Delta[:,:,k])$ for $k \in [0,24)$. The quartiles are calculated for each mode, including the geographic regions and days in the example traffic data. Fig. 5ⓐⓑⓒ show how we overlay the residual statistics on the entries for different chart types. Note that when $Q_{1/2}$, $Q_{3/4}$, and $Q_{3/4}$ are all positive or negative, they will appear on only one side of the base point (e.g., ***a2*** in Fig. 6ⓐ).

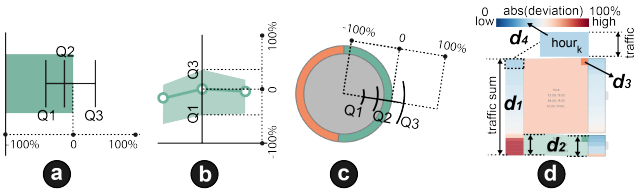

Fig. 5. ⓐⓑⓒ The encoding of the deviation of the raw data from the depicted patterns on each dimension. We visualize quartile statistics for the normalized deviation ranging from 0%-100%. ⓓ The encoding of the deviation on the tree nodes. To highlight high discrepancy, we take the absolute value of the normalized deviation and use a diverging color scheme (blue-white-red). For details, please refer to Section 4.2.4.

4.2.4 Steerable and iterative tensor partitioning

TPFlow supports a steerable and iterative workflow such that analysts can progressively divide the data into smaller subsets (R4) along different dimensions. To support this workflow, we visualize the successive subdivision of the tensor along different dimensions similar as in decision trees (R5). Each node in the tree represents a subset of data created in the partitioning process. Analysts can directly interact with the tree nodes to partition them further or select a few nodes to compare the latent patterns across different partitions.

Visualize the partitioning process. The tree starts with a root node on the left (***a1*** in Fig. 1ⓐ), representing the original tensor $\mathscr{X}$. The height of the node is proportional to the sum of all the entries in $\mathscr{X}$. Take the traffic flow data as an example, the height of the root node (***d1*** in Fig. 5ⓓ) is proportional to the total traffic volume aggregated over all the days, hours and spatial regions. Analysts can manually select a dimension (e.g. days, hours, or locations) to partition the original tensorial data into several sub-tensors. For each sub-tensor, a child

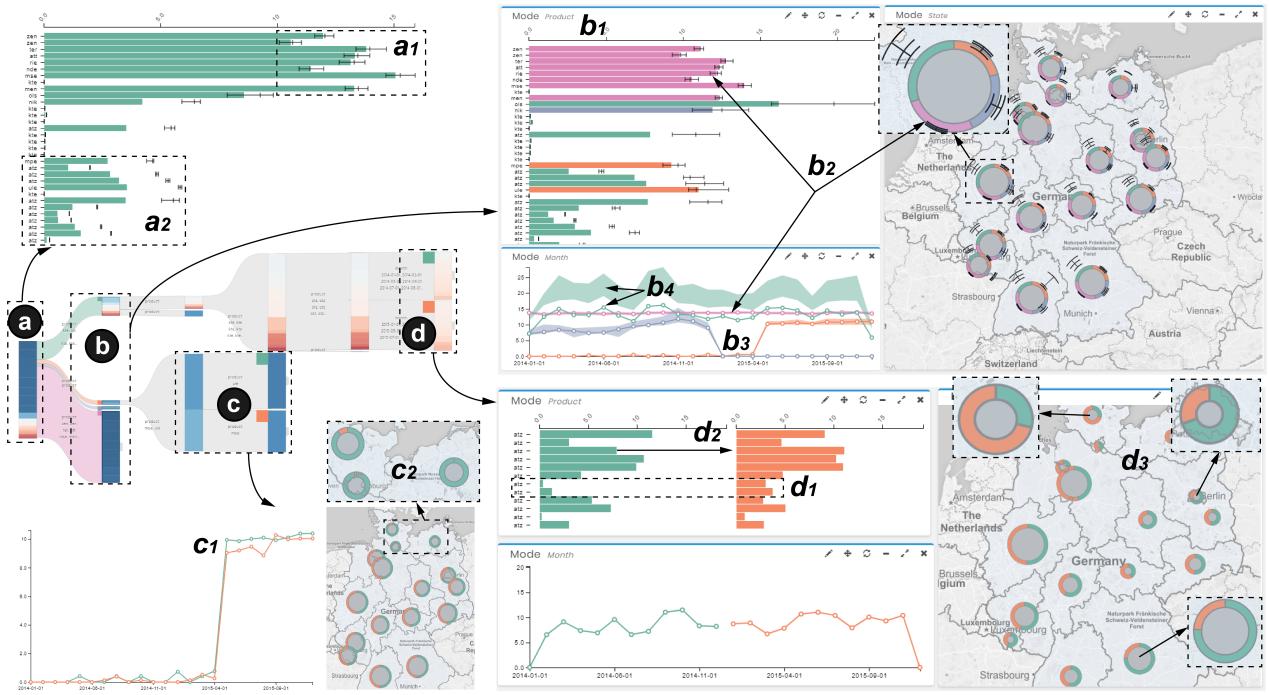

Fig. 6. ⓐ A group of products having large amount of sales have small deviations (***a1***), while some products with relatively lower sales have high deviations (***a2***). ⓑ Among the four groups of products (***b1***), the pink group includes most of the products with high sales volume and their deviations on every dimension are very small (***b2***). The other three product groups also show different monthly patterns (***b3***, ***b4***). ⓒ The two products in orange group of very different monthly patterns are selected and partitioned for fine-grained analysis (***c1***, ***c2***). ⓓ The yearly sales performance of one product group is compared (***d1***, ***d2***, ***d3***).

node will be created and attached to the root. The heights of the child nodes as well as the widths of the links are proportional to the sum of the entries in the corresponding partition (***d2*** in Fig. 5ⓓ). Based on the widths of the links and the heights of the nodes, analysts can focus on the partitions that account for a more significant portion the traffic volume. Analysts can perform further partitioning on the sub-tensors and the tree will be updated correspondingly to display the results. Thus, the overall partitioning process is visually represented all the time, which not only reflects analytical provenance but also facilitates navigation and refinement on the partitions. Notice that every selected node is attached with a small rectangle of an automatically assigned categorical color (***d3*** in Fig. 5ⓓ), which shares the same color scheme with the other sub-views.

Enable steerable data partitioning. Our system enables direct interaction with the tree nodes to perform data partitioning. When analysts hover over a node, a menu (***a4*** in Fig. 1ⓐ) will pop up displaying different options. The options include the dimension to perform partition on, the number of clusters to create and the clustering algorithm to use. We provide several different options of clustering algorithms including k-means, hierarchical clustering [24], and OPTICS [6] (no need to specify the cluster number). If the partition dimension is set to "auto", our system will perform partitioning on every mode and recommend the optimal one with minimized cost function Eq. 2. The button (***a5***) is for confirming split and the button (***a6***) is used to resize the node for further exploration when node becomes too small. Every action is reversible by clicking the button (***a2***).

Encode discrepancy. On each node, we use a diverging color scheme from blue to red to visualize the discrepancies (***d4*** in Fig. 5ⓓ). Blue suggests a low discrepancy between the raw data and the latent patterns extracted by the algorithm while red indicates high discrepancy. More concretely, for each node, on a selected dimension, we create a set of sorted colored stripes, each representing an entry in the corresponding dimension. For example, 24 entries will be created if the *hours* dimension is selected for the traffic flow data. The color encodes the average deviation from the recovered tensor data (for hour k, the value is $avg(|\Delta[:,:,k]|)$). Strips with similar colors are grouped for scalability. With such visual encoding, analysts can quickly get an insight on the faithfulness of the patterns extracted by the algorithm for each partition and refine the partition if necessary.

5 Example Usage Scenarios

For each usage scenario, we describe the schema, dimensionality and scale of the dataset, introduce the domain-specific analytic tasks, and then report how the domain experts used TPFlow for these tasks.

5.1 Regional sales data analysis

The regional sales data (*RS* data) contains two million sales records of a company's product portfolio in Germany in a two-year period. Each record follows the schema: $(month, product, state) \rightarrow sales\_volume$. In total there are 24 months, 34 products and 16 federal states. We therefore construct a $24 \times 34 \times 16$ tensor, where each entry records the total sales in one month within one state for a particular product.

The experts are interested in the following topics to develop better marketing strategies: (1) *product segmentation.* Identify cross selling opportunities by grouping products with similar demand distributed over time and geographical regions. (2) *temporal comparison.* Compare the distribution of sales over different products for selected time intervals. (3) *market segmentation.* Divide the federal states into subgroups with common characteristics such as shared needs for a set of products or similar temporal variations in sales.

Product segmentation. Before segmenting the products, the experts first wanted to understand whether these products are indeed significantly different in terms of their temporal and regional sales variations. They observed that a large portion of the root node is colored with deep blue (Fig. 6ⓐ), which means that a group of products accounting for a large portion of the sales have similar temporal and geospatial distribution. By contrast, the smaller red colored portion in the root node suggests that some products with relatively lower sales can be quite different. This observation can be further verified by showing the deviations in the bar chart (***a1***, ***a2***). Therefore, the experts decided to group the products for further analysis.

Fig. 6ⓑ shows that the products are split into four groups, whose detailed partition information can be observed (***b1***). The pink group includes most of the products with high sales volume. They have very small deviation on every dimension (***b2***) which indicates that the products in this group have very similar monthly and spatial sales variation. Thus, the experts did not need to partition this group further. The orange and purple groups are separated for their abnormal temporal

variations (***b3***). The purple group contains one product and the sales of this product suddenly dropped down to zero in February 2015. On the other hand, the two products in the orange group start to sell since April 2015. By further splitting the two products (Fig. 6ⓒ), they observed more fine-grained patterns (***c1***, ***c2***). For example, the two products have similar sales volume in most of the states, while the orange product has nearly zero sales in the three states on the top of the map (***c2***).

Temporal comparison. The products in the green group have fluctuating sales over time (***b4*** in Fig. 6ⓑ). The high deviations of this product group suggest that this group can be further split. The experts found half of the products in the green group belong to the same category (related to xxx-alz). Therefore they manually specified these products as a new group. To compare the sales volume in the two consecutive years, they manually separated the months into two groups (Fig. 6ⓓ). They found several interesting patterns, for example, the sales volume of two products (***d1***) increased significantly in 2015 and the third product (***d2***) became the first place regarding sales. From the map chart (***d3***), the experts found that many states have the same sales proportion. However, there are still several states whose sales volume in the two years are significantly different.

Market segmentation. To further compare different regions, the experts split the node on the state dimension (***a3*** in Fig. 1ⓐ). The group information is shown in Fig. 1ⓓ. One interesting pattern is that the states in non-pink groups present a similar monthly trend to some extent (***c1***). However, the pink states, which are located mostly in the northeast region of Germany, perform significantly different from the other states. In particular, they have an outstanding peak in June (***c2***) and their sales are mostly contributed by one product (***b1***).

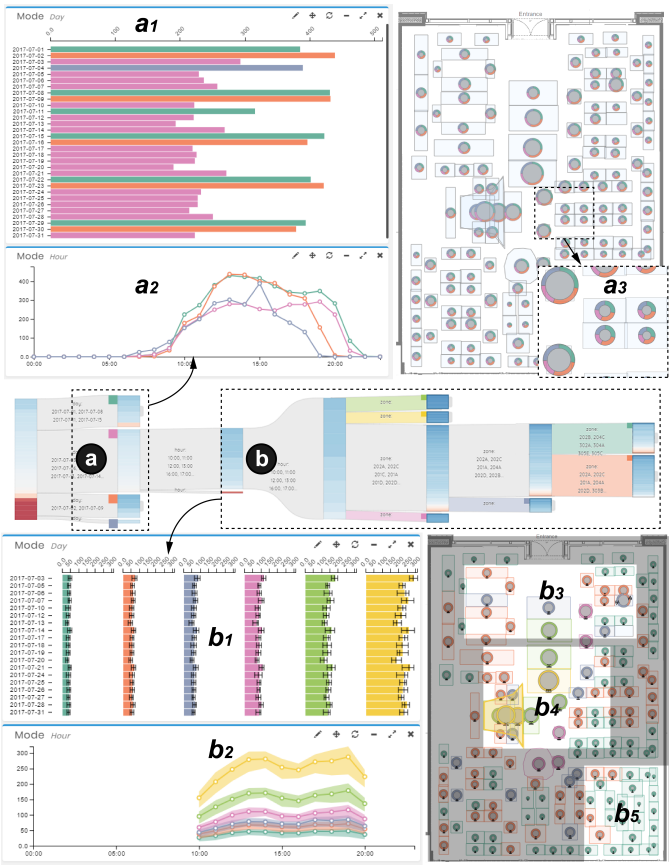

Fig. 7. ⓐ The green, orange, pink and purple groups (***a1***) contain Saturdays, Sundays, weekdays, and one U.S. Holiday, respectively. They have quite different hourly patterns (***a2***) but similar spatial distributions (***a3***). ⓑ The retail areas' performance on weekdays' daytime hours are compared (***b1***, ***b2***, ***b3***, ***b4***, ***b5***).

5.2 Customer in store traffic data analysis for brick-and-mortar retailers

The customer in-store traffic data (*CST* data) contains about 25 million records in a large department store in the U.S. in July, 201x. A record is generated when a person enters or leaves a particular area in the store and the schema is $(record\_id,\ retail\_area\_id,\ event\_timestamp,\ event\_type)$, where $event\_type \in \{enter,\ leave\}$. With these records, the aggregated area statistics with the schema $(day \times hour \times retail\_area\_id) \rightarrow traffic\_volume$ can be obtained, which records the number of customers that pass through an area during one hour in one day. In total there are 31 days, 24 hours and 163 different areas in the store and we construct a $31 \times 24 \times 163$ tensor. Different from the previous example, this schema contains two temporal dimensions (i.e., days and hours).

The experts are interested in the following tasks for better management of store hours, store layout, staff capacity, etc.: (1) *identify daily periodical patterns.* Get an overview about how the hours affect the traffic and group the days with similar hourly traffic patterns; (2) *analyze retail area performance.* Group and compare the retail areas based on temporal variations in traffic volume.

Identify daily periodical patterns. In the beginning, the experts had no prior knowledge on the dataset and did not know where to start. As a result, they ran the clustering algorithm (i.e., OPTICS) on auto mode (i.e., let the machine find the optimal dimension to cluster). Accordingly, our system recommended splitting the days into four groups (Fig. 7ⓐ). As shown in the bar chart (***a1***), the purple group contains only one day (Independence Day in US). The green and orange groups include all the Saturdays and Sundays, respectively. The pink group consists of all the weekdays. From the line chart (***a2***), the experts found the hourly trend differences among the four groups: Saturdays (green) and Sundays (orange) have the largest number of customers overall, whereas on Sundays the customers appear to leave much earlier than Saturdays. For the weekdays (pink), the traffic is more stable during the normal business hours and start to decrease after 9pm (later than Sunday but earlier than Saturday). The purple (Independence Day) is an outlier, which has an obvious peak hour around 3pm and has almost zero traffic since 7pm (even earlier than Sundays). From the map chart (***a3***), the experts found that most of the glyphs have the similar color proportions on the outer rings. This suggests the overall traffic distribution over different retail areas has no clear difference over different days. In other words, special days (e.g., weekends, Independence Day) do not have much influence on the areas' relative traffic volume.

Analyze retail area performance. The experts first selected the weekdays and manually seperated the daytime hours from 10am to 8pm. Then they iteratively split on the area dimension. Fig. 7ⓑ displays the final result after several rounds of split operation. The line chart (***b2***) displays the hourly traffic of each group of areas, where the traffic volume is becoming less and less from top to bottom. However, their shapes roughly remain the same. The bar chart shows a similar pattern (***b1***). From the map chart, the experts observed that the areas close the center have much more traffic (***b4***). The areas on the top (***b3***) have relatively more traffic, as they are close to the main entrance. In general, the areas on the bottom right (***b5***) have much less traffic.

5.3 The New York taxi trip OD data

The New York taxi trip OD data (referred to as *NYT* dataset) is a public dataset provided by NYC TLC[1]. It includes the taxi trip information in 2016 in NY. Each record contains the following information for a taxi trip: pickup timestamp, dropoff timestamp, pickup taxi zone, dropoff taxi zone, passenger count, trip distance, fare amount and tip amount. From the additional table provided by TLC, we obtain the name, borough, and geometry boundaries for each taxi zone. To study the taxi trip patterns with OD information we construct a tensor from the original records with a new schema: $(day \times pickup\_hour \times pickup\_zone \times dropoff\_zone) \rightarrow passenger\_count$. In this example,

[1] http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml

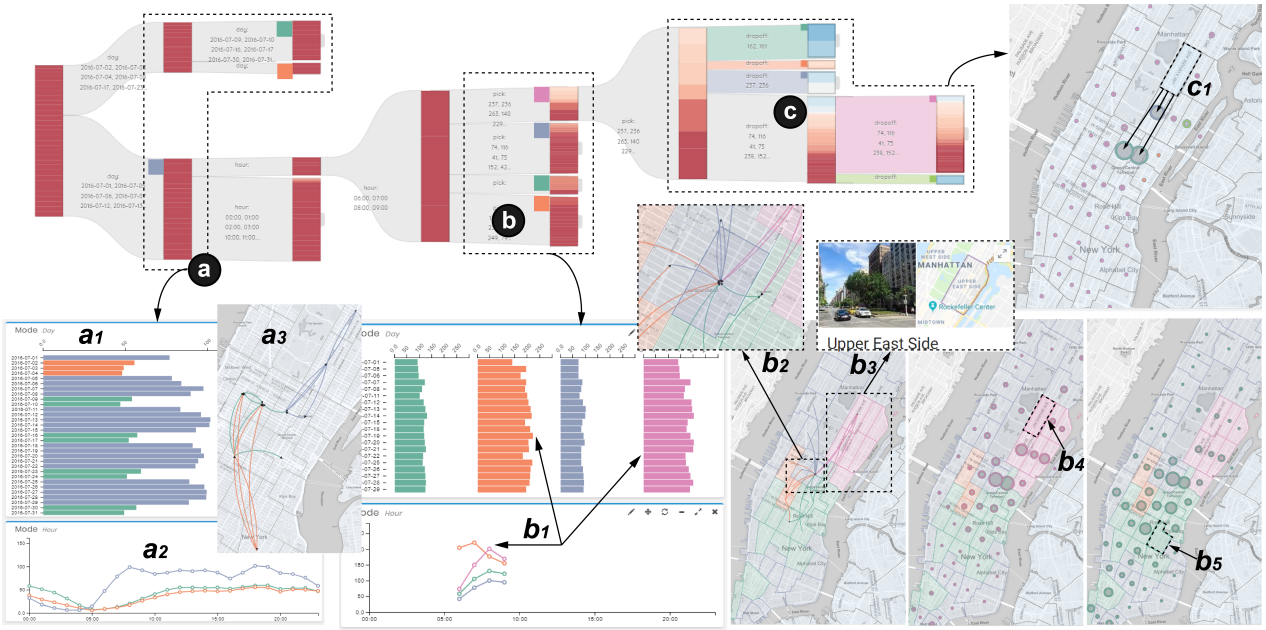

Fig. 8. ⓐ The green, purple, and orange groups (***a1***) contain weekends, weekdays, and U.S. holidays, respectively. They have a different hourly taxi demand (***a2***) and top-k flow distributions (***a3***). ⓑ For the morning rush hours in weekdays, our system identifies four groups of pickup zones (***b2***, ***b3***), each group has different patterns in days, hours (***b1***), and dropoff zones (***b4***, ***b5***). ⓒ The dropoff patterns can be explored in a fine-grained level by partitioning on dropoff zone mode. The overwhelming majority of passengers from (***c1***) take taxis to few areas nearby.

we extracted over 10 million taxi trip records within the Manhattan area (67 taxi zones) in July, constructing a $(31 \times 24 \times 67 \times 67)$ tensor. We performed 4-way tensor decomposition to identify the hourly ($\vec{t}_1$) and daily ($\vec{d}_1$) variation of traffic volume, together with the traffic flow out ($\vec{p}_1$) and in ($\vec{p}_2$) patterns for different taxi zones. The value for a flow from zone i to zone j can be calculated as the product of the corresponding entries in the 1st loading vectors of the flow in and flow out dimension: $\vec{p}_1[i] \times \vec{p}_2[j]$. In the system, analysts can select the top-k largest flows to observe the hottest OD combinations.

This example usage scenario focus on understanding the taxi demand for more efficient resource planning, which includes (1) *understanding the temporal and spatial distribution of taxi demand* (2) *extracting traffic OD patterns* based on the analysis of the 4-mode tensor.

In general, this dataset is more complex and it is more difficult to directly use rank-1 factors to describe the original data, as the deviations are still very high even after several rounds of partitions (ⓐ → ⓑ → ⓒ in Fig. 8). We first iteratively partitioned on *days* to study how date influences taxi demand (Fig. 8ⓐ). The bar chart (***a1***) shows the three identified groups. The orange group contains three days including July 4th (Independence Day) and the Saturday and Sunday before it. The green group includes all the other weekends. The remaining weekdays form the third purple group. From the line chart (***a2***), we observed that the number of passengers on weekdays (purple) is significantly higher than others (i.e., holidays and weekends) during the daytime, but is lowest during the nighttime. The holiday (orange) daytime passengers are slightly less than the weekends (green), but the nighttime passengers are obviously less than weekends. Probably during holidays, there are fewer taxi drivers and people tend to go home much earlier than usual weekends. The flow map (***a3***) shows that the top six OD flows for the three groups of days span across different areas in Manhattan.

Afterwards, we turned our focus to compare the taxi demand in different zones during weekday rush hours. We selected the weekdays and manually separated the morning rush hours from 6am to 9am. After that, we partitioned the pickup taxi zones and identified four different groups of zones (Fig. 8ⓑ). By filtering the top-k in and out flows for each group, we found a common popular dropoff location for all the four groups (***b2***). From the bar chart and line chart (***b1***), we examined their different behaviors with respect to days and hours. For example, the pink zones and orange zones have the largest number of passengers, but their corresponding hourly trends are quite different. One interesting pattern here is that the pink area (***b3***) is almost equivalent to the Upper East Side[2], which is one of the largest residential areas in Manhattan and known for its wealthy residents. We further compared the detailed dropoff locations for the zones in different groups. Location ***b4*** and ***b5*** are selected as the pickup zones, one from pink group and the other one from the green group. We found that the overwhelming majority of passengers from the pink zone (***b4***) take taxis to a few nearby areas, while the destinations for the passengers from the green zone (***b5***) spread more in the bottom of Manhattan. We did a partition on the pink group on the dropoff zone mode (Fig. 8ⓒ) to explore the more fine-grained dropoff patterns. Then we observed there are significantly more passengers dropping off in the zones identified by ***c1***.

6 Algorithm Performance

Time complexity. For a p-dimensional tensor where each mode includes n entries, computing the best rank-one approximation in each iteration is expensive in terms of space and time complexity, both requiring $O(n^p)$ [13,26]. For a server with 2.20GHz Intel i7-6650U CPU and 16GB RAM, the algorithm can return results within reasonable time for the three datasets in our case studies (<1sec for RS and CST, and <10sec for NYT). As the running time heavily depends on the size of the tensor to be decomposed, the algorithm becomes more responsive when applied on subsets (partitions) of the three datasets.

Baseline method. We compare the piecewise rank-one tensor partition algorithm with a baseline approach. The baseline method simply flattens the tensor into a matrix and performs clustering on it using classical methods such as k-means, hierarchical clustering and OPTICS (in the experiment we use hierarchical clustering with average linkage and Euclidean distance). Take the traffic volume data as an example. To partition on days D, the baseline method flattens the tensor by converting each slice $\mathcal{X}[:,j,:]$ to a feature vector with length $\|P\| \times \|T\|$, which contains all the entries from one day $D[j]$. The feature vectors of all the days are inputed to the clustering algorithm. Based on the clustering result, the method slices the tensor on D to obtain a series of sub-tensors and perform rank-one tensor decomposition on each.

Experimental results on synthetic data. We apply our algorithm to synthetic data with ground truth cluster structure to assess the robustness and reliability. We generate a series of synthetic data ($100 \times 100 \times 100$ tensors) with Gaussian noise, which are designed to have an optimal partition number ranging from 2 to 8 on the first mode. We run our algorithm and the baseline method on every tensor's first mode with the ground truth clustering number. We compare the

[2]https://en.wikipedia.org/wiki/Upper_East_Side

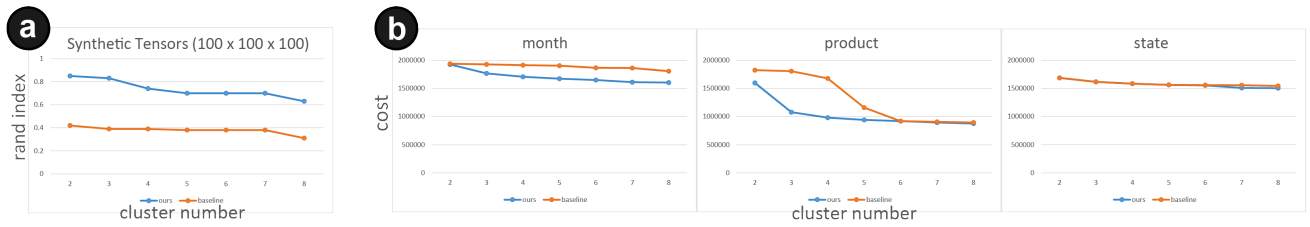

Fig. 9. ⓐ Our algorithm produces clustering results that align better with the ground truth on synthetic data as measured by ARI [27]. ⓑ The experimental results on RS data show that our algorithm produce sub-tensors and their 1st loading vectors that can more faithfully represent the original ST data as measured by the cost function (Eq. 2).

clustering results using the Adjusted Rand index (ARI) [27], which is one of the most widely adopted metrics to measure the similarity of clustering structure. ARI = 1.0 indicates that the partitions are identical. As shown in Fig. 9ⓐ, our partition algorithm (blue) aligns better with the ground truth compared to the baseline.

Experimental results on real-world data. We run the experiments on the tensorial data presented in the three case studies. For each experiment, we run both algorithms on every dimension with different setting in cluster number (from 2 to 8). We compare the algorithms using the cost function described in Eq. 2 (the smaller, the better). Fig. 9ⓑ presents the results of the experiments on RS dataset (refer to the Appendix for more). Our algorithm (blue) consistently has smaller cost when partitioning on the months and products dimension. This indicates the 1st loading vectors in sub-tensors can more faithfully represent the patterns. However, for states, there is no clear difference.

Summary. The experiments prove that our algorithm can find more reasonable ways to partition the tensor and represent the patterns more faithfully. The curse of dimensionality [20] could be a possible explanation. For the baseline algorithm, the dimensionality of feature vectors can easily exceed a thousand or even tens of thousands. This leads to the result that all the entries appear to be dissimilar, regardless of the distance metrics (e.g., Euclidean distance) being used. This prevents commonly-used clustering methods from being effective.

7 Expert Feedback

We collected and summarized the feedback provided by the domain experts when they were completing the first two case studies.

Usability. The experts all appreciated our system and confirmed the usefulness of the various features in TPFlow. They agreed that the system is efficient and effective not only for searching meaningful patterns within data but also for verifying their reliability. The CST experts emphasized that they identified some unexpected patterns using the system. One of them praised the smooth interactions and commented, "The steerable partition interactions are quite useful. It allows me to go forth and back to refine my operations without losing the context."

Improvement. Although the system is easy to use without a steep learning curve for the four data scientists, it can be still challenging for ordinary users (e.g., store managers) without technical background knowledge in data cube, clustering, or some statistical terminologies like quartiles. For those users, our experts suggest us to further simplify the visual interface and interactions. Another valuable comment from one of the RS experts is that the system can provide interactive widgets for dynamically setting the bin size (e.g., time interval in temporal dimension) to construct ST tensor, as the choices of units in each tensor dimension are closely related to the analytic tasks.

8 Discussion, Limitations and Future Work

At the current stage, we aim at developing a generic system for ST data in various application domains to demonstrate the effectiveness of the tensor-based analytical workflow. We put extra emphasis on the interpretability and faithfulness of the displayed patterns. We visualize the patterns with widely-used visualizations instead of sophisticated visual encodings for general applicability. The current system can be easily extended by integrating more advanced visualization techniques tailored to domain-specific tasks. Besides that, revealing more detailed information in the decomposition process can be useful for users with a strong technical background. For example, we can visualize the feature vectors (refer to Sec. 3.2) of the entries on one dimension on a 2D plane using dimension reduction techniques for interactive visual clustering.

Algorithm scalability. The decomposition algorithm may become inefficient when the tensor size increases. Thus, several techniques can be considered to improve the performance. For example, we can apply a more space and time efficient tensor decomposition method [57] which sacrifices a certain amount of accuracy in exchange for a 10x-100x speed-up. We can use GPU to accelerate the process since many tensor operations can be parallelized. We can also apply the progressive analytics [52] methodology: for long queries enable the analysts to observe the algorithm's partial results and interactively prioritize subspaces of interest. Besides that, hierarchical structures can be used to construct tensors with multi-resolution to further speed up computation.

Visual/perceptual scalability. The current system as we observe can reasonably support simultaneous comparison of one dozen partitions, due to the limited number of categorical colors that people can effectively differentiate [44]. Furthermore, some of the charts we use in the system cannot handle too many entries. For example, the bar chart cannot visualize too many rows on a single screen. Nevertheless, we believe that the current visual design can be applied to most application scenarios, as the current system is targeting at providing an overview of data instead of displaying every detail. For details, they are usually application specific and can be provided on-demand [50]. Besides that, the system can also integrate advanced interaction techniques such as semantic zoom [63], hierarchical exploration [62], and focus+context [47] to handle the scalability problem.

Spatio-temporal coherence. For some scenarios, analysts prefer to see neighboring hours or regions are grouped together with a higher probability. Our algorithm currently does not incorporate such contextual information (i.e., the order of timestamps and the distances between regions). To exploit such information, we consider adding a regularization term [53] in the tensor decomposition model to smooth the feature vectors on the spatial and temporal dimensions. This helps increase the possibility for the nearby time intervals or regions being assigned to one group. As a future work, we plan to integrate such regularization term and allow analysts to use them on demand.

9 Conclusion

We introduce an interactive visualization system, TPFlow, for explorative analysis of large-scale multidimensional ST data using a top-down, progressive partitioning approach. We propose a novel tensor-based algorithm to support automated partitioning and multidimensional pattern extraction on ST data. The algorithm is grounded on a quantitative measure about how faithfully the extracted patterns can visually represent the original data. We compare the algorithm to a baseline method. Our algorithm produces results that 1) align better with ground truth on synthetic data and 2) represent the original data more faithfully on real-world ST data. Built upon the algorithm, the TPFlow system features a novel combination of visualization and interaction designs to facilitate pattern discovery, comparison and verification. We present example usage scenarios with real-world ST data from three application domains to demonstrate the general applicability and effectiveness of our method.

Acknowledgments

We would like to thank Miaoyan Wang for discussing the idea, Quin Smith for supporting the video editing, and Sina Fiesta and Siheng Liu for proofreading the paper. We are grateful for the valuable comments provided by Prof. Huamin Qu and the VAST reviewers. This work is supported in part by grant RGC GRF 16241916.

REFERENCES

[1] T. W. Anderson. Estimating linear restrictions on regression coefficients for multivariate normal distributions. *The Annals of Mathematical Statistics*, pp. 327–351, 1951.

[2] G. Andrienko, N. Andrienko, S. Bremm, T. Schreck, T. Von Landesberger, P. Bak, and D. Keim. Space-in-time and time-in-space self-organizing maps for exploring spatiotemporal patterns. In *Computer Graphics Forum*, vol. 29, pp. 913–922. Wiley Online Library, 2010.

[3] G. Andrienko, N. Andrienko, W. Chen, R. Maciejewski, and Y. Zhao. Visual analytics of mobility and transportation: State of the art and further research directions. *IEEE Transactions on Intelligent Transportation Systems*, 18(8):2232–2249, 2017.

[4] G. Andrienko, N. Andrienko, G. Fuchs, and J. Wood. Revealing patterns and trends of mass mobility through spatial and temporal abstraction of origin-destination movement data. *IEEE transactions on visualization and computer graphics*, 23(9):2120–2136, 2017.

[5] N. Andrienko, G. Andrienko, and P. Gatalsky. Exploratory spatio-temporal visualization: an analytical review. *Journal of Visual Languages & Computing*, 14(6):503–541, 2003.

[6] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: ordering points to identify the clustering structure. In *ACM Sigmod record*, vol. 28, pp. 49–60. ACM, 1999.

[7] B. Bach, P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale. A review of temporal data visualizations based on space-time cube operations. In *Eurographics conference on visualization*, 2014.

[8] M. T. Bahadori, Q. R. Yu, and Y. Liu. Fast multivariate spatio-temporal analysis via low rank tensor learning. In *Advances in neural information processing systems*, pp. 3491–3499, 2014.

[9] R. Beecham, J. Dykes, W. Meulemans, A. Slingsby, C. Turkay, and J. Wood. Map lineups: effects of spatial structure on graphical inference. *IEEE transactions on visualization and computer graphics*, 23(1):391–400, 2017.

[10] E. V. Bonilla, K. M. Chai, and C. Williams. Multi-task gaussian process prediction. In *Advances in neural information processing systems*, pp. 153–160, 2008.

[11] X. Cai, K. Efstathiou, X. Xie, Y. Wu, Y. Shi, and L. Yu. A study of the effect of doughnut chart parameters on proportion estimation accuracy. In *Computer Graphics Forum*. Wiley Online Library.

[12] N. Cao, C. Lin, Q. Zhu, Y.-R. Lin, X. Teng, and X. Wen. Voila: Visual anomaly detection and monitoring with streaming spatiotemporal data. *IEEE transactions on visualization and computer graphics*, 24(1):23–33, 2018.

[13] J. D. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of "eckart-young" decomposition. *Psychometrika*, 35(3):283–319, 1970.

[14] W. Chen, F. Guo, and F.-Y. Wang. A survey of traffic data visualization. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):2970–2984, 2015.

[15] X. Chen, Z. He, and J. Wang. Spatial-temporal traffic speed patterns discovery and incomplete data recovery via svd-combined tensor decomposition. *Transportation Research Part C: Emerging Technologies*, 86:59–77, 2018.

[16] A. Cichocki, R. Zdunek, A. H. Phan, and S.-i. Amari. *Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation*. John Wiley & Sons, 2009.

[17] N. Cressie and H.-C. Huang. Classes of nonseparable, spatio-temporal stationary covariance functions. *Journal of the American Statistical Association*, 94(448):1330–1339, 1999.

[18] N. Cressie, T. Shi, and E. L. Kang. Fixed rank filtering for spatio-temporal data. *Journal of Computational and Graphical Statistics*, 19(3):724–745, 2010.

[19] Q. Cui, M. Ward, E. Rundensteiner, and J. Yang. Measuring data abstraction quality in multiresolution visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):709–716, 2006.

[20] D. L. Donoho et al. High-dimensional data analysis: The curses and blessings of dimensionality. *AMS Math Challenges Lecture*, 1:32, 2000.

[21] H. Doraiswamy, N. Ferreira, T. Damoulas, J. Freire, and C. T. Silva. Using topological analysis to support event-guided exploration in urban data. *IEEE transactions on visualization and computer graphics*, 20(12):2634–2643, 2014.

[22] Z. Fan, X. Song, and R. Shibasaki. Cityspectrum: a non-negative tensor factorization approach. In *UbiComp*, pp. 213–223. ACM, 2014.

[23] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva. Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2149–2158, 2013.

[24] J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[25] Y. Han and F. Moutarde. Analysis of large-scale traffic dynamics in an urban transportation network using non-negative tensor factorization. *International Journal of Intelligent Transportation Systems Research*, 14(1):36–49, 2016.

[26] R. A. Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[27] L. Hubert and P. Arabie. Comparing partitions. *Journal of classification*, 2(1):193–218, 1985.

[28] E. H. Isaaks and M. R. Srivastava. *Applied geostatistics*. Number 551.72 ISA. 1989.

[29] N. Kamat, P. Jayachandran, K. Tunga, and A. Nandi. Distributed and interactive cube exploration. In *ICDE*, pp. 472–483. IEEE, 2014.

[30] S. Kandel, R. Parikh, A. Paepcke, J. M. Hellerstein, and J. Heer. Profiler: Integrated statistical analysis and visualization for data quality assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pp. 547–554. ACM, 2012.

[31] D. A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on visualization and computer graphics*, 6(1):59–78, 2000.

[32] S. Ko, R. Maciejewski, Y. Jang, and D. S. Ebert. Marketanalyzer: an interactive visual analytics system for analyzing competitive advantage using point of sale data. In *Computer Graphics Forum*, vol. 31, pp. 1245–1254. Wiley Online Library, 2012.

[33] E. Kofidis and P. A. Regalia. On the best rank-1 approximation of higher-order supersymmetric tensors. *SIAM Journal on Matrix Analysis and Applications*, 23(3):863–884, 2002.

[34] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[35] M.-J. Kraak. The space-time cube revisited from a geovisualization perspective. In *Proc. 21st International Cartographic Conference*, pp. 1988–1996, 2003.

[36] L. Lins, J. T. Klosowski, and C. Scheidegger. Nanocubes for real-time exploration of spatiotemporal datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2456–2465, 2013.

[37] D. Liu, F. Guo, B. Deng, H. Qu, and Y. Wu. egocomp: A node-link-based technique for visual comparison of ego-networks. *Information Visualization*, 16(3):179–189, 2017.

[38] D. Liu, D. Weng, Y. Li, J. Bao, Y. Zheng, H. Qu, and Y. Wu. Smartadp: Visual analytics of large-scale taxi trajectories for selecting billboard locations. *IEEE transactions on visualization and computer graphics*, 23(1):1–10, 2017.

[39] M. Liu, J. Shi, Z. Li, C. Li, J. Zhu, and S. Liu. Towards better analysis of deep convolutional neural networks. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):91–100, 2017.

[40] S. Liu, W. Cui, Y. Wu, and M. Liu. A survey on information visualization: recent advances and challenges. *The Visual Computer*, 30(12):1373–1393, 2014.

[41] Z. Liu, B. Jiang, and J. Heer. imMens: Real-time visual querying of big data. In *Computer Graphics Forum*, vol. 32, pp. 421–430. Wiley Online Library, 2013.

[42] W. Luan, G. Liu, C. Jiang, and L. Qi. Partition-based collaborative tensor factorization for poi recommendation. *IEEE/CAA Journal of Automatica Sinica*, 4(3):437–446, 2017.

[43] M. Mørup. Applications of tensor (multiway array) factorizations and decompositions in data mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(1):24–40, 2011.

[44] T. Munzner. *Visualization analysis and design*. CRC press, 2014.

[45] C. A. Pahins, S. A. Stephens, C. Scheidegger, and J. L. Comba. Hashedcubes: Simple, low memory, real-time visual exploration of big data. *IEEE transactions on visualization and computer graphics*, 23(1):671–680, 2017.

[46] G. Punj and D. W. Stewart. Cluster analysis in marketing research: Review and suggestions for application. *Journal of marketing research*, pp. 134–148, 1983.

[47] R. Rao and S. K. Card. The table lens: merging graphical and symbolic representations in an interactive focus+ context visualization for tabular information. In *SIGCHI*, pp. 318–322. ACM, 1994.

[48] D. Sacha, M. Kraus, J. Bernard, M. Behrisch, T. Schreck, Y. Asano,

and D. A. Keim. SOMFlow: Guided Exploratory Cluster Analysis with Self-Organizing Maps and Analytic Provenance. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):120–130, 2018. doi: 10.1109/TVCG.2017.2744805

[49] R. Scheepens, C. Hurter, H. Van De Wetering, and J. J. Van Wijk. Visualization, selection, and analysis of traffic flows. *IEEE transactions on visualization and computer graphics*, 22(1):379–388, 2016.

[50] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *The Craft of Information Visualization*, pp. 364–371. Elsevier, 2003.

[51] D. Skau and R. Kosara. Arcs, angles, or areas: individual data encodings in pie and donut charts. In *Computer Graphics Forum*, vol. 35, pp. 121–130. Wiley Online Library, 2016.

[52] C. D. Stolper, A. Perer, and D. Gotz. Progressive visual analytics: User-driven visual exploration of in-progress analytics. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1653–1662, 2014.

[53] K. Takeuchi, Y. Kawahara, and T. Iwata. Structurally regularized non-negative tensor factorization for spatio-temporal pattern discoveries. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 582–598. Springer, 2017.

[54] M. Trutschl, G. Grinstein, and U. Cvek. Intelligently resolving point occlusion. In *Information Visualization*, pp. 131–136. IEEE, 2003.

[55] J. J. Van Wijk and E. R. Van Selow. Cluster and calendar based visualization of time series data. In *Information Visualization, 1999.(Info Vis' 99) Proceedings. 1999 IEEE Symposium on*, pp. 4–9. IEEE, 1999.

[56] T. Von Landesberger, F. Brodkorb, P. Roskosch, N. Andrienko, G. Andrienko, and A. Kerren. Mobilitygraphs: Visual analysis of mass mobility dynamics via spatio-temporal graphs and clustering. *IEEE transactions on visualization and computer graphics*, 22(1):11–20, 2016.

[57] Y. Wang, H.-Y. Tung, A. J. Smola, and A. Anandkumar. Fast and guaranteed tensor decomposition via sketching. In *Advances in Neural Information Processing Systems*, pp. 991–999, 2015.

[58] D. Weng, R. Chen, Z. Deng, F. Wu, J. Chen, and Y. Wu. Towards better spatial integration in ranking visualization. *To appear in IEEE Transactions on Visualization and Computer Graphics*.

[59] D. Weng, H. Zhu, J. Bao, Y. Zheng, and Y. Wu. Homefinder revisited: finding ideal homes with reachability-centric multi-criteria decision making. In *Proceedings of the 2018 CHI*, p. 247. ACM, 2018.

[60] Y. Wu, Z. Chen, G. Sun, X. Xie, N. Cao, S. Liu, and W. Cui. Streamexplorer: A multi-stage system for visually exploring events in social streams. *To appear in IEEE Transactions on Visualization and Computer Graphics*, (1):1–1, 2018.

[61] Y. Yan, Y. Tao, J. Xu, S. Ren, and H. Lin. Visual analytics of bike-sharing data based on tensor factorization. *Journal of Visualization*, pp. 1–15, 2018.

[62] J. Yang, M. O. Ward, and E. A. Rundensteiner. Hierarchical exploration of large multivariate data sets. In *Data Visualization*, pp. 201–212. Springer, 2003.

[63] J. S. Yi, Y. ah Kang, and J. Stasko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE transactions on visualization and computer graphics*, 13(6):1224–1231, 2007.

[64] T. Zhang and G. H. Golub. Rank-one approximation to high order tensors. *SIAM Journal on Matrix Analysis and Applications*, 23(2):534–550, 2001.

[65] J. Zhao, P. Forer, and A. S. Harvey. Activities, ringmaps and geovisualization of large human movement fields. *Information visualization*, 7(3-4):198–209, 2008.

[66] Y. Zheng, L. Capra, O. Wolfson, and H. Yang. Urban computing: concepts, methodologies, and applications. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(3):38, 2014.

[67] Y. Zheng, W. Wu, Y. Chen, H. Qu, and L. M. Ni. Visual analytics in urban computing: An overview. *IEEE Transactions on Big Data*, 2(3):276–296, 2016.