Transportation Research Part B 91 (2016) 250–269

Contents lists available at ScienceDirect

Transportation Research Part B

journal homepage: www.elsevier.com/locate/trb

Clustering of heterogeneous networks with directional flows based on "Snake" similarities

Mohammadreza Saeedmanesh, Nikolas Geroliminis*

Urban Transport Systems Laboratory (LUTS), School of Architecture, Civil and Environmental Engineering, École Polytechnique Fédérale de Lausanne (EPFL), GC C2 390, Station 18, CH-1015 Lausanne, Switzerland

ARTICLE INFO

Article history:
Received 23 December 2015
Revised 27 April 2016
Accepted 17 May 2016
Available online 4 June 2016

Keywords:
Graph partitioning
Spatiotemporal correlation
Macroscopic fundamental diagram
Non-negative matrix factorization
Missing data

ABSTRACT

Aggregated network level modeling and control of traffic in urban networks have recently gained a lot of interest due to unpredictability of travel behaviors and high complexity of physical modeling in microscopic level. Recent research has shown the existence of well-defined Macroscopic Fundamental Diagrams (MFDs) relating average flow and density in homogeneous networks. The concept of MFD allows to design real-time traffic control schemes specifically hierarchical perimeter control approaches to alleviate or postpone congestion. Considering the fact that congestion is spatially correlated in adjacent roads and it propagates spatiotemporaly with finite speed, describing the main pockets of congestion in a heterogeneous city with small number of clusters is conceivable. In this paper, we propose a three-step clustering algorithm to partition heterogeneous networks into connected homogeneous regions, which makes the application of perimeter control feasible. The advantages of the proposed method compared to the existing ones are the ability of finding directional congestion within a cluster, robustness with respect to parameters calibration, and its good performance for networks with low connectivity and missing data. Firstly, we start to find a connected homogeneous area around each road of the network in an iterative way (i.e. it forms a sequence of roads). Each sequence of roads, defined as 'snake', is built by starting from a single road and iteratively adding one adjacent road based on its similarity to join previously added roads in that sequence. Secondly, based on the obtained sequences from the first step, a similarity measure is defined between each pair of the roads in the network. The similarities are computed in a way that put more weight on neighboring roads and facilitate connectivity of the clusters. Finally, Symmetric Non-negative Matrix Factorization (SNMF) framework is utilized to assign roads to proper clusters with high intra-similarity and low inter-similarity. SNMF partitions the data by providing a lower rank approximation of the similarity matrix. The proposed clustering framework is applied in medium and large-size networks based on micro-simulation and empirical data from probe vehicles. In addition, the extension of the algorithm is proposed to deal with the networks with sparse measurements where information of some links is missing. The results show the effectiveness and robustness of the extended algorithm applied to simulated network under different penetration rates (percentage of links with data).

* Corresponding author. Tel.: +41 21 69 32481; fax: +41 21 69 35060.
E-mail addresses: mohammadreza.saeedmanesh@epfl.ch (M. Saeedmanesh), nikolas.geroliminis@epfl.ch (N. Geroliminis).

http://dx.doi.org/10.1016/j.trb.2016.05.008
0191-2615/

1. Introduction

Traffic congestion appears with different shapes and patterns and might propagate in particular directions varying from day to day. The network infrastructure in smart cities, allows for sensing and collecting massive spatiotemporal data, such as the trajectories of many vehicles from navigation devices, which represent proxies for human mobility patterns. This 'big mobility data' provides a unique observatory that can help to understand how congestion develops and evolves, discover hidden patterns and identify models that can contribute in efficient traffic management techniques to improve cities' mobility and accessibility. However, unpredictability of travel behavior and high complexity of accurate physical modeling have challenged researchers to develop realistic microscopic level models that are able to replicate congestion spreading and growth in large traffic networks.

There is a strong understanding and vast literature of congestion dynamics and spreading in one-dimensional traffic systems with a single mode of traffic, e.g. a highway section with cars. Besides traffic scientists, mathematicians and physicists have also contributed to the field of traffic flow modeling. Because of the numerous publications, we refer the reader to Helbing (2001) for an overview. Briefly speaking, the main modeling approaches can be classified as car-following models (e.g. Gazis et al., 1959; Gipps, 1981); cellular automata (e.g. Nagel and Schreckenberg, 1992); gas-kinetic models (e.g. Herman et al., 1972); first-order and higher order flow models such as Lighthill and Whitham (1955), Payne (1971) and Whitham (1974).

Literature in network level dynamics and congestion propagation is limited especially in large urban networks. Previous works have mainly built on micro-simulations of link-level traffic dynamics or graphical visualizations of congestion without any metrics and dynamic models. However, both the unpredictability of travel behaviors and high complexity of accurate physical modeling remain challenging and simulation results may be time consuming and not realistic for dynamic systems with stochastic characteristics. The main characteristic of traffic in urban networks is that congestion is spatially correlated in adjacent roads and it can propagate with some finite speed in time and space. These correlations allow describing the main pockets of congestion in a city with a small number of clusters without the need for detailed information in every link of the network (Ji et al., 2014). However transportation networks have unique dynamic features and a direct application of an arbitrary clustering algorithm may not produce a desired solution.

With respect to network level, it has been observed with empirical and simulated data in Geroliminis and Daganzo (2008), Buisson and Ladier (2009), and Gayah and Daganzo (2011) that by spatially aggregating the highly scattered plots of flow vs. density from individual links (e.g. 1 min data), the scatter almost disappears and a well-defined curve exists between space-mean flow and density. The idea of an MFD with an optimum accumulation belongs Godfrey (1969) and later re-introduced by Daganzo (2007), Herman and Prigogine (1979) and Mahmassani et al. (1987). Geroliminis and Daganzo (2008) showed that the shape of MFD is a property of the network infrastructure and control and not very sensitive to the demand. This is important for modeling purposes, as details in individual links are not necessary to describe congestion in cities. It can also be utilized to introduce simple control strategies to improve mobility in multi-region city centers building on the concept of an MFD, like in Ramezani et al. (2015), Keyvan-Ekbatani et al. (2015), Haddad and Shraiber (2014), and others. A detailed literature review of network modeling and control can be found for instance in Haddad et al. (2013) or Mahmassani et al. (2013). Furthermore, latest works extend the single-mode MFD to a bi-modal where cars and buses share the same infrastructure and look at passenger flow dynamics in addition to vehicular dynamics (Chiabaut et al., 2014; Zheng and Geroliminis, 2013), and (Chiabaut, 2015). Estimating MFDs with different type of data (loop detectors, probe vehicles or a combination) are investigated in Leclercq et al. (2014), Du et al. (2015), Ji et al. (2014) and others.

Recent findings from empirical and simulated data Geroliminis and Sun (2011), Mazloumian et al. (2010), Daganzo et al. (2011), and Knoop and Hoogendoorn (2013) have identified the spatial distribution of vehicle density in the network as one of the key components that influence the shape and the scatter of an MFD. These findings are of great importance because the concept of an MFD can be applied for heterogeneously loaded cities with multiple centers of congestion, if these cities can be partitioned into a number of homogeneous clusters. The objectives of partitioning are to obtain (i) small variance of road densities within a cluster, which increases the network flow for the same average density and (ii) connectivity and spatial compactness[1] of each cluster which makes the application of perimeter control strategies feasible. The proposed mechanism in this paper can produce a partitioning with a desired number of clusters that contain connected roads with small density variance. Furthermore, this mechanism demonstrated superiority of both effectiveness and robustness in comparison with standard clustering algorithms.

There is a vast literature on studying clustering algorithms in several fields such as community detection (Lancichinetti and Fortunato, 2009), data-mining (Jiawei Han, 2000), and image segmentation (Shi and Malik, 2000). Depending on the type of application, problems related to clustering generally fall into two main categories: unconstrained and constrained clustering. The former does not have any constraints on objects to be in the same cluster (i.e. all the set partitions are feasible) while the latter is limited in search space meaning that some of the clustering assignments are not feasible. In our urban partitioning problem, we look for connected and compact shaped clusters with homogeneous traffic conditions to have low-scatter MFDs, which allows us to utilize perimeter control strategies that work based on the concept of MFD.

[1] A measure of compactness can be defined as a summation of distances (shortest path) of all the roads in the cluster from the center of that cluster. The center of a cluster is defined as a road or set of roads with minimum average distance to all the roads in that cluster.

Hence, our problem, known as contiguity-constrained clustering, is considered as an instance of constrained clustering. In this type of problems, nearby objects (regions, links, etc.) have similar characteristics and therefore should belong to the same cluster (i.e., that structure should be preserved as much as possible).

Existing methods for solving contiguity constrained clustering problems can be divided into two main groups, considering the approach taken to satisfy the constraints. The first group includes the approaches where the spatial contiguity is indirectly fulfilled whereas in the second group, spatial contiguity is explicitly imposed within the algorithm. The idea of the majority of the methods that implicitly enforce contiguity is to feed spatial information either into the data or into the similarity between data objects and then utilize existing techniques in unconstrained clustering (e.g k-means, spectral clustering, etc.). For instance in Ji and Geroliminis (2012), authors investigate the performance of k-means in partitioning urban networks by considering spatial locations of the roads as new features in the data. These algorithms might not always work properly in transport networks as there is no guarantee to have connected and compact clusters. Connectivity and compactness could come by incorporating spatial information elegantly into similarity. The second group of algorithms explicitly impose spatial contiguity within their solution process (Duque et al., 2007). Hierarchical clustering approaches are instances of this category of methods. They create a hierarchical decomposition of the given set of data objects by merging or splitting database. These methods preserve connectivity only by merging neighboring objects at each step. Guo (2008) utilizes hierarchical agglomerative clustering method and proposes six different criteria to find best candidate objects to be merged.

There are some other proposed ways that impose connectivity during the solution process. For instance, the work in Ji and Geroliminis (2012) proposed a clustering method building on Normalized-Cut (N-Cut) algorithm (Shi and Malik, 2000). The authors enforced connectivity by considering similarities only between the roads that are adjacent. This work has laid a solid approach for static partitioning in grid-type networks with similar level of congestion on both traffic directions of the same road. It also requires a connected graph of the network with data, while missing values or malfunctioning detectors might create difficulties in application of the method. Nevertheless, this approach has specific limitations. First, many congested urban networks in central business districts (CBD) of cities might experience strong directional flows during different times of the day, e.g. high demand for directions towards the CBD in the morning peak and the opposite during the evening. The network topology might not be symmetric (grid-type) but some areas might experience higher connectivity than others. In this case, the algorithm identifies boundaries of clusters with low connectivity instead of parts with strong change in the level of congestion. The current paper tries to overcome the aforementioned difficulties and develop a clustering methodology that (i) is able to find directional congestion within a cluster, (ii) is robust with respect to parameters' calibration and (iii) has good performance for networks with low connectivity and missing data.

The remainder of the paper is organized as follows: Section 2 introduces the proposed methodological framework and explains each algorithmic step separately in details. Briefly speaking, first we identify connected homogeneous components by running snakes from different roads in the network. Snakes grow in a way to preserve homogeneity and connectivity by iteratively grabbing the most similar adjacent road. A cumulative measure is then introduced and computed to quantify spatial correlation between different pairs of snakes. At the final step, roads are assigned to the cluster using the lower rank matrix approximation. Section 3 extends the approach to be applied in cases where portion of data is missing. In Section 4, results of the proposed algorithm are presented and compared with other approaches for two case studies. The first case study is a large size asymmetric network with real data from probes (Shenzhen city with more than 10,000 links) and the second case study is a grid type medium size network with simulation data (San Francisco with about 400 links). Moreover, performance of the algorithm is evaluated under different amounts of missing data in the network of San-Francisco. Paper concludes in the last section by identifying current and future research directions.

2. Methodological Framework

In this section, we introduce and explain the main steps of the proposed clustering method. We are seeking to develop an algorithm to partition the network into connected and homogeneous clusters. The ideal clusters should have the three following conditions: (i) low variance of link density (or speed); (ii) connectivity within the clusters, i.e. links inside clusters should have spatial connections; (iii) reasonable size, i.e. ideal clusters are balanced in size.

In fact, our problem is multi-objective and the first two objectives (homogeneity and connectivity) usually conflict with each other. The way we see this multi-objective problem is first by solving condition (i) that minimizes homogeneity and assuming other conditions as constraints. In the second step, based on the results of the first step, a similarity measure is defined between each pair of the roads. Definition of the similarities is done in a way that put more weight on the condition (ii) that takes connectivity and compactness into consideration. Finally in the last step, a mathematical framework is used that group similar links in clusters with balanced sizes. As in different parts of the methodology, intermediate results in the study networks are presented; we briefly describe the data and networks now.

2.1. Case studies

The first case study is the dataset of the megacity of Shenzhen in China, which contains the network structure and daily GPS data of taxis for one month. Shenzhen is a major city in the south of Southern China's Guangdong Province, situated immediately north of Hong Kong. There are 11,125 nodes connected by 12,305 links. The number of taxis is around 20 thousand and varies from day to day. Average speed of each link is used as a representative value which computed using

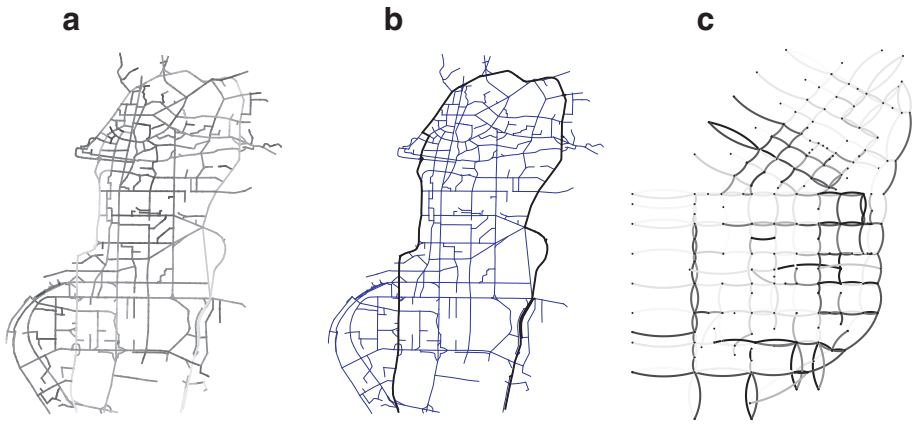

Fig. 1. First case study: (a) gray-scale representation of link speeds in network of Shenzhen; (b) highway around the network of Shenzhen (depicted with black color). Second case study: (c) gray-scale representation of link densities in network of San-Francisco.

map-matching algorithm, described in Ji et al. (2014). Upper part of Shenzhen, which has about 2000 links, is analyzed in this study, since more data is available for that part and the speed estimation is more reliable. Fig. 1(a) shows the upper part of the network with estimated values of speed for different links depicted in gray-scale format. Note that links with darker colors are more congested and have lower speed. The study area includes a highway going around the Shenzhen which is depicted with black color in Fig. 1(b). As highways have different traffic properties (e.g. lane width, free flow speed) than urban networks and are not utilized in MFD-based control approaches, we consider two different cases: (i) the whole network including the highway; (ii) the urban part of the network by excluding the highway. The results reveal the ability of our proposed approach in grouping the highway links within the clusters with high average speed in the case where the study area includes the highway. It is worth highlighting that this network has no grid structure with many hierarchical components and different spatial connectivity, which makes the application of the methodology challenging.

The second case study is a 2.5 square mile area of Downtown San Francisco (Financial District and South of Market Area) modeled and calibrated in Aimsun 7. The network includes about 100 intersections with link lengths varying from 400 to 1300 ft. The number of lanes for through traffic varies from 2 to 5 and the free flow speed is 30 miles per hour. Traffic signals are all multiphase fixed-time operating on a common cycle length of 100 s for the west boundary of the area (The Embarcadero) and 60 s for the rest. A 4hr time-dependent traffic demand (120 time intervals of 2 minutes) is applied to this network, which produces different spatial and temporal levels of congestion. For each link in the network, we first extract the number of vehicles every second and then aggregate them to find average number of vehicles over periods of 2 minutes. To have a comparable congestion indicator, the average density [$veh/(km.lane)$] is calculated which takes the size of links (number of lanes and length) into account. We select a time at which there is a high range of average density values and sub-regions with different levels of congestion could be easily seen in different locations. Fig. 1(c) illustrates the network and level of congestion in a certain time in a gray-scale format. Note that arcs represent different directions in two way roads (counter clockwise direction). It could be easily seen in Fig. 1(c) that the network is heterogeneous with different levels of congestion. Moreover, there are some bi-directional roads with only one congested direction, which will facilitate to test the performance of model for detecting directional congestion.

2.2. Definitions and conceptualization

To explain the proposed clustering algorithm, first based on the network structure, a graph $G = (V, E)$ is built in which V and E are sets of nodes and edges respectively. Each link (road) in the network is represented by a node $u \in V$ in a graph and a value of traffic parameter is assigned to that node (i.e. density, speed, etc.). There is an edge between links which are connected to each other. Based on our definition, two links are spatially connected if either end or beginning of them are connected to the same intersection. In other words, all the roads entering or exiting the same intersection are assumed to be connected[2]. Note that, we do not want to limit adjacent links by imposing directional connectivity and we want to let the algorithm to find directional congestion in the parts of network where this situation happens. Hence, roads with different directions connecting to the same intersection are considered adjacent.

The proposed algorithm utilizes the fact that congestion has strong spatial correlation in transportation networks. To investigate this characteristic, we take any road in the network and compute the standard deviation of the roads densities

[2] Other definitions can be developed; for instance, one can define the connectivity based on the existing movements (and allowed turnings) in the intersection. This can be even more realistic since the traffic condition in these roads are affected by each other (i.e. spill back from downstream link propagates to the upstream links and causes congestion). We do not have access to such level of information for the Shenzhen network , so we focus on the simpler definition mentioned above (given also that GPS measurements contain errors).

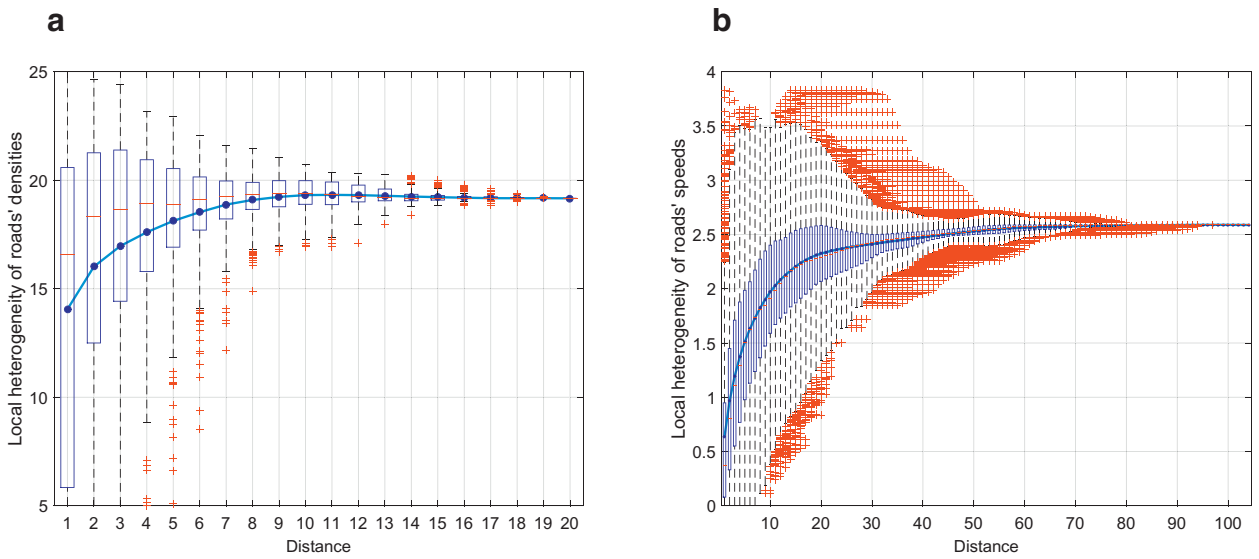

Fig. 2. Distributions of local heterogeneity for different distances: (a) San-Francisco [*veh*/(*km.lane*)]; (b) Shezhen[*m*/*sec*].

(speeds) up to a certain distance around it, i.e. this value can be interpreted as local heterogeneity in the vicinity of that road. Therefore, for any given distance, we come up with a distribution of local heterogeneity values where each corresponds to a single road in the network. Note that, the distance between two roads is considered as the minimum number of edges needed to connect corresponding two nodes in the graph (i.e. shortest path between two nodes). Figs. 2(a) and 2(b) depict a box-plot [3] representation of the local heterogeneity values as a function of distance for the networks of San-Francisco (heterogeneity of road densities) and Shenzhen (heterogeneity of road speeds) respectively. As we expected, the average local heterogeneity increases by the distance meaning that neighboring roads have stronger spatial correlation. Furthermore, it can be observed that the distribution of local heterogeneity has higher variance for small distances. Indeed, for small distances, high local heterogeneity can be occurred in areas with directional congestion or close to the borders of different clusters where we have different level of congestion in a small area. This shows the necessity of searching in appropriate direction even if the average local heterogeneity is low for small distances. As we will explain in details, the proposed approach will introduce a robust and smart way of searching along neighboring roads.

In the first step of the algorithm, an iterative process is applied starting from each individual link in the network. For each of the links in the network, we build a sequence of links iteratively, with an objective of having minimum variance of all the chosen links density (or speed) at each step. One can consider this as a 'snake' that starts from a link and grows by attracting the most similar adjacent link iteratively. Note that, at each step, the most similar link to the snake is the adjacent link with the closest density (speed) value to the average density (speed) values of the links in the snake. This procedure identifies interesting patterns as the variance grows in a non-linear way with the size of the links. This observation is related to the fact that some parts of the graph are more similar than others and if a large number of dissimilar links is added, high variance is unavoidable. This step needs graph information about connectivity as only adjacent links are added at each step.

In the second step, based on the obtained sequences, a measure of similarity is defined and computed for each pair of the links. Using this new similarity measures, a fully connected graph is built in which one edge exists between every two nodes meaning that every two links in the network have similarity. The value of an edge shows the similarity between two nodes that are connected using that edge. In graph theory, many approaches were developed to partition graph into sub-graphs based on the similarity values. Among the existing approaches, Symmetric Non-negative Matrix Factorization (SNMF) (Kuang et al., 2015) is applied in the third step as it has more flexibility to capture innate clusters. The similarity matrix is used as an input in SNMF framework. Different steps of the partitioning algorithm are described in details in the following sections.

[3] Box-and-whisker plot with outliers represents how data is distributed. On each box, the central red line is the median, the blue point is the average (mean), and the edges of the box are the 25th and 75th percentiles denoted by q_1 and q_3 respectively. An outlier is any value that lies more than one and a half times the length of the box from either end of the box. That is, if a data point is below $q_1 - 1.5 \times (q_3 - q_1)$ or above $q_3 + 1.5 \times (q_3 - q_1)$, it is viewed to be deviated markedly from the central point. The whiskers (lower and upper boundaries) extend to the most extreme data points not considered outliers, and outliers are plotted individually with red crosses (McGill et al., 1978).

2.3. Step 1: Running snakes

Transportation networks experience strong spatial correlations due to congestion propagation. If a link i of the network is congested at time t, then the adjacent link j has high probability to be congested at t or get congested soon (see Fig. 2). We will further elaborate this property later. Given this inspiration, in the first step, an iterative approach is developed to find locally similar links for each link in the network. Basically, for each specific link in the network, we start creating an array (simply speaking a sequence of links) by iteratively adding one of the adjacent links to the current links in the array until all the links in the network are added. Note that an array corresponding to a specific link, starts with that link as an initial element and continues growing by adding other links. We name these arrays as 'snakes' since arrays are growing by absorbing different links near them. The objective of each snake is to grow by swallowing adjacent links while keeping the variance as low as possible. So in each step, each snake identifies all the adjacent links and adds the one, which has the closest value to the average value of data in it. This procedure continues until snake encompasses all the links in the network. Variance at each step can be calculated as follows:

$$\sigma_k^2 = \frac{(k-1)\sigma_{k-1}^2 + (x_k - \bar{x}_k)(x_k - \bar{x}_{k-1})}{k} \tag{1}$$

$$\bar{x}_k = \frac{(k-1)\bar{x}_{k-1} + x_k}{k} \tag{2}$$

where $\bar{x}_k$, σ_k^2 denote the mean and variance of the snake with size k and x_k is the value (speed or density) of the link added in the kth step. By simply calculating the $\frac{d\sigma_k^2}{dx_k}$, we could find that best candidate in each step is the one that has closest density (speed) value to the current average. By enforcing snake to grow in this way, we achieved the following goals: (i) in each step the snake contains connected parts of the network since it only allows to add one of the adjacent links; (ii) the sequence of each snake contains the links which have a high degree of homogeneity as the link with the nearest value to the average is added to guarantee minimum variance. We run the snakes and collect the sequence of added links for each of the snakes separately. It is worth to mention that the sequence of links is important in each array and shows priority of links in different snakes.

The idea of the snake algorithm could be better explained using an example of ideal case where a certain number of fully homogeneous regions with different levels of congestion exists in the network (i.e. all links in one region have the same density or speed value). Snakes starting from different links in each of these regions, first integrate all the points in that regions and then start adding from other regions with different level of congestion. We can conclude that after reaching this state, all the snakes within one component will show the same behavior. Moreover, they have many common links during their evolution before reaching the whole component. We use these features in order to define a similarity measure between pairs of links in the next step of the algorithm. According to this simple example, snakes with different initial points have a robust trend inside each cluster and converge to the same state before adding links from different clusters. Our conjecture is that this property will remain partially valid even in heterogeneous networks due to spatial correlations, even if a fully robust trend cannot be guaranteed because of many stochastic factors. In order to investigate the existence of the components with high value of robustness, we apply the snake algorithm in the San-Francisco network. Results of this part could give some understanding about the possibility of partitioning a network.

By running the snake algorithm through this network from all different points, the robust components are identified. Robust component in a network refers to a set of links in which all snakes starting from any initial link in that component will end up having the exact same links in this sequence. Plenty of small components are found and many of them have overlap with the bigger ones. The four biggest components which are not part of a bigger component(excluding the whole network) are illustrated in Fig. 3(a-d). These components include 292 out of 366 (approximately 80%) links. The reason that these components could not merge to create bigger ones can be explained based on the difference between values of the densities inside each component. For instance, components 1 and 3 are adjacent but they have completely different values. Note that two components have much larger mean density values than the other two, which shows that this approach is able to separate different pockets of congestion in the network. To further investigate, we plot the variance of the different snakes inside these components in the Fig. 4. Note that while Fig. 3 shows only the robust part of the snake (see above for the definition), we continue to run the snake for the whole network to understand the evolution of variance and utilize them later in calculating similarity measure.

As it was expected, the difference between values of different components is the reason why bigger regions do not have full robustness. Note the large jump in the evolution of variance happens for size larger than the size of the robust component (e.g. size 21 for component 1). This jump coincides with the time at which snakes in that component converge to the same state (see Fig. 4(e)). There are some stochastic fluctuations in the first steps as well which should not be considered as changing of clusters since for small sizes of snake adding one different point could change the variance significantly. However, for bigger sizes, sharp changes have the meaning of adding very different sub-regions. An interesting observation is that these jumps are unavoidable even if the snake adds links that minimize the total variance at this specific step. One option to create clusters in the network is to identify with a stochastic procedure (e.g. with a wavelet analysis) the sharp change of slope in the variance plot and consider this as a new cluster. While this is under current investigation, a difficulty

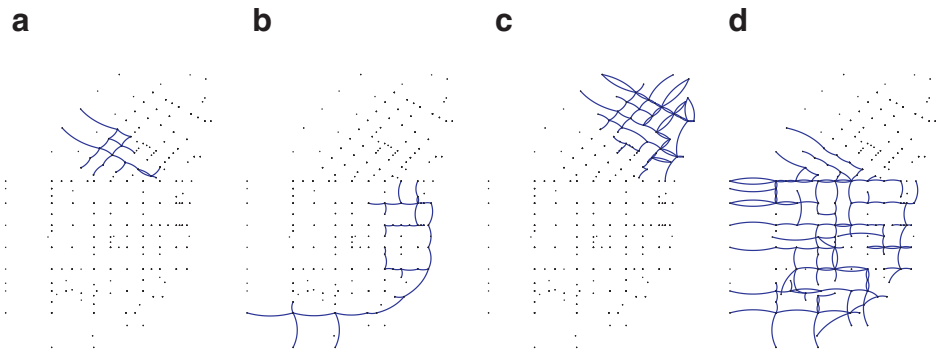

Fig. 3. Four big components in the network of San-Francisco (size, average density of the component): (a) (21, 48.51); (b) (39, 51.63); (c) (79, 5.36); (d) (153, 4.86); (unit of density: [*veh*/(*km.lane*)]).

is that this procedure cannot guarantee that all links of a network will finally belong to a cluster. Nevertheless, the proposed snake approach is very useful to identify similarities between links in the network (step 2) that can contribute to an efficient clustering method (step 3).

Existing robust components is a promising feature as it shows that the methodology is not very sensitive to initial conditions of the network. This property also justifies the existence of strong spatial correlation in urban networks despite the existence of stochastic factors. Moreover, it allows us to reduce the computational time in the first step of the algorithm. Indeed, we can first let all the snakes grow up to a certain size (e.g. one third of network size) and identify the robust component; then, running one snake from each component would be enough to complete the sequence for all the snakes belong to that component. The computational complexity increases with the size of snake as more candidate links appear to be added to the snake in later steps. Hence, running smaller number of snakes can significantly increase the computational speed in the first step of the algorithm. What is challenging is how to simultaneously run different snakes in the network and choose about potential conflicts of specific links that could belong in more than one snake. We resolve this issue by integrating a similarity matrix[4] between any two links in the network using the proposed snake approach as we will describe in the next step of the algorithm.

2.4. Step 2: Defining similarities

In the second step, the goal is to define a similarity between each pair of the links in the network that takes into account both homogeneity and spatial connectivity (compactness) and later be utilized to create clusters. As we mentioned, the sequence of the (links)roads in each arrays not only represents the links with the close density (speed) values, but also has some information about the spatial connectivity of the links. Based on these properties of the snakes, we propose a method that put more weight on the links that are spatially closer to each other and their corresponding snakes converge to the same point. This would be achieved if the similarity calculated as (3):

$$w(i,j) = \sum_{k=1}^{N} \text{intersect}(S_{ik}, S_{jk}) \tag{3}$$

where S_{ik}, S_{jk} are the snakes of size k corresponding to the links i and j (i.e. snake with size k) and N denotes the total number of links in the network. Function 'intersect' in Eq. (3) computes the number of common elements between two subsets with equal size. To calculate the similarity between each pair of snakes (corresponding to different links), the number of common elements is calculated for all possible snake sizes and a cumulative metric based on Eq. (3) is developed.

In fact, Eq. (3) puts more weight on the snakes that have more common links in the first steps, as it will consider links that are closer to each other. As size of a snake grows, this measure of similarity gives less weight on the links that are collected. This is consistent with the fact that after some iterations, snake does not have any good options to add and starts adding link with different values and the variance becomes bigger (see Fig. 4). Therefore, we should put less weight on the links appear at the end of snakes. By defining similarity in this way, we aim to achieve connectivity in the clusters. This similarity measure is illustrated with two examples in Fig. 5. Fig. 5(a) represents snakes corresponding to two links that are close to each other and have similar level of congestion, while Fig. 5(b) could represent case in which links are either far from each other or belong to different groups with different levels of congestion. This definition of similarity results in having cluster with high degree of connectivity even though we do not explicitly enforce them.

In Fig. 5(a), links (m, n, k, j) are collected in the first steps by both snakes i and j. Based on our definition once a link appears in the two snakes, it will be counted in the similarity from that moment to the end. Hence, in the similarity

[4] The spectral clustering methods utilize similarities of data points (see (Luxburg, 2007) for detailed review)

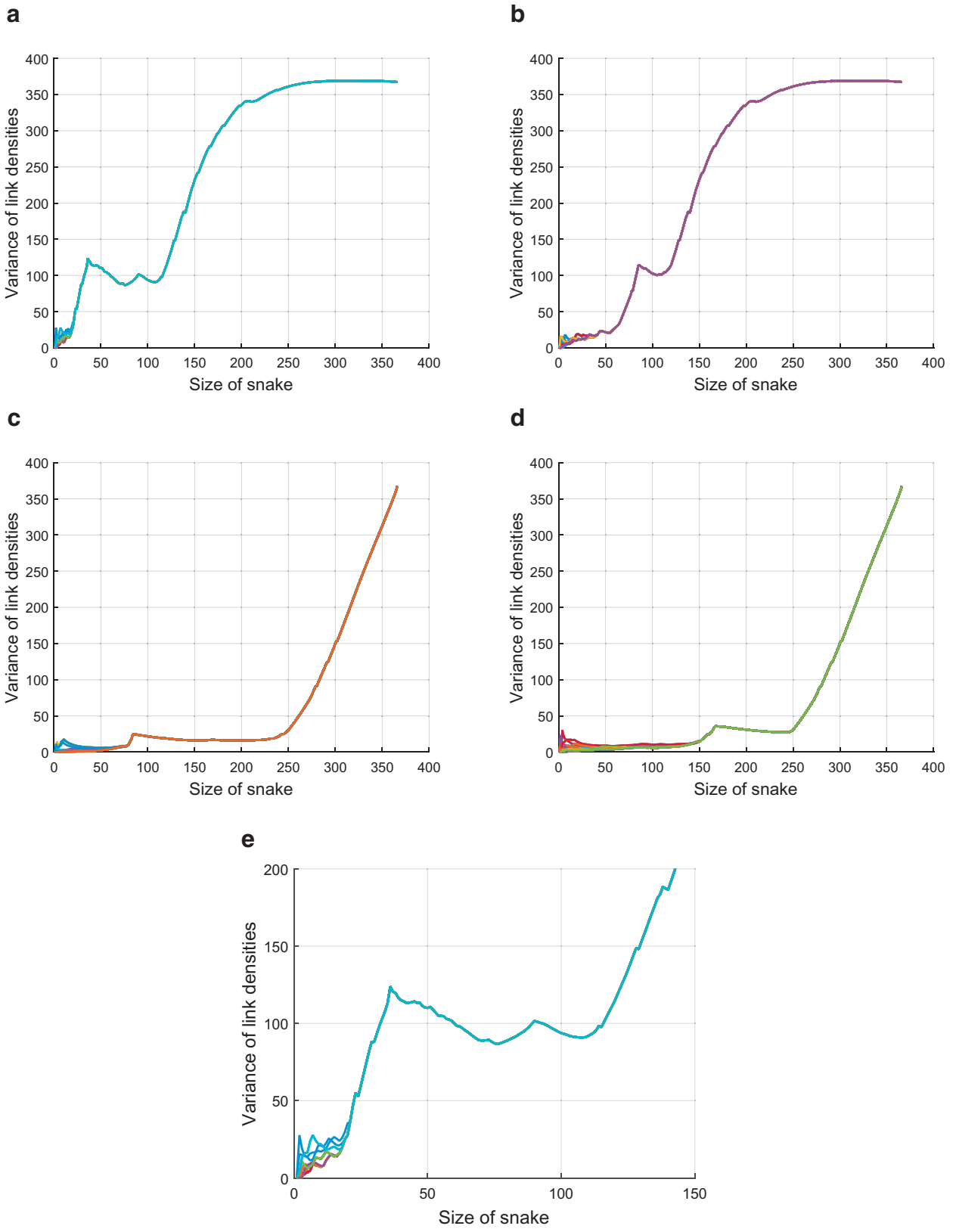

Fig. 4. Variance of link densities for different snakes in each robust component as a function of size (a-d). A zoom in for the first component is provided (e). Each color represents a different snake in that component.

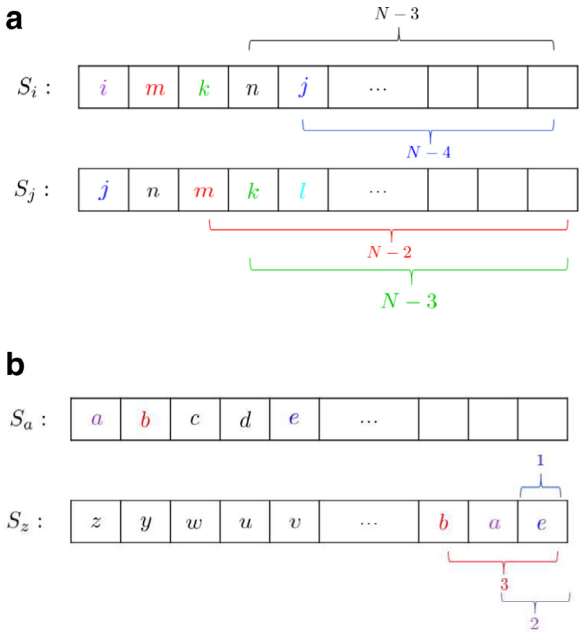

Fig. 5. Contribution of different elements in the similarity measure between two sample pairs of snakes representing: (a) two close roads with similar density (speed) values; (b) two distant roads with different density (speed) values.

measure, these links are counted many times. For instance, link m in the similarity measure is counted from window size 3 to the end which is equal to N-2 times. Fig. 5(b) represents the conditions in which sequences of the numbers in the snakes a and z are completely different. Links (a, b, e) count very few times since they appear in the last window sizes. In conclusion, links that are added in both snakes earlier, they count more and this is consistent with the connectivity objective. In fact, Eq. (3) could be rewritten as follows:

$$w(i,j) = \sum_{h=1}^{N} \{N - \max(\mathcal{L}(h, S_{iN}), \mathcal{L}(h, S_{jN})) + 1\} \tag{4}$$

where S_{iN} and S_{jN} are full size snakes corresponding to links i and j. Function $\mathcal{L}(h, S_{iN})$ gives the location of link h inside the snake i. With the new form of equation, we can better see different weights on the value of similarity.

Within this framework, it is also possible to change these weights more sharply, by defining a weighting coefficient $\phi \geq 1$ as in Eq. (5). Note that by selecting bigger ϕ, you can put more weight on the spatial information and the clusters will be more compact in shape. In other words, links that are added at the end of the snakes are considered much less since the weighting coefficient becomes smaller as the size of snake grows. An alternative way to achieve this is to run the snake only up to a certain size as final links has small impact on the similarity which can significantly improve the computational times. The similarity matrix in this case will be an sparse matrix as some links have zero similarity which can simplify the complexity of optimization problem in the next step. We investigate the performance of clustering under different size of snakes for two case studies in Section 4.

$$w(i,j) = \sum_{k=1}^{N} \phi^{N-k} \times \text{intersect}(S_{ik}, S_{jk}) \tag{5}$$

2.5. Step 3: Symmetric Non-negative Matrix Factorization

In graph theory, there is a large class of clustering algorithms called “Spectral clustering” that assigns objects into proper clusters with respect to their pair-wise similarity measure rather than looking at data itself. The success of spectral clustering is mainly based on the fact that it does not make any assumptions on the form of the clusters. The three most common objective functions used in spectral clustering are Normalized cut (Shi and Malik, 2000), Min-max cut (Ding et al., 2001), and Ratio cut (Hagen and Kahng, 1992) that capture clusters with different characteristics. However, the behavior of such algorithms highly depends on the eigenvalues and eigenvectors of the Laplacian matrix ($L = D - W$), where W is a similarity matrix and degree matrix D is a diagonal matrix with elements $d_i = \sum_{j=1}^{N} w_{ij}$ on the diagonal. The eigen structure of L has a stable lower rank representation only when the difference between two consecutive eigenvalues of matrix L is sufficiently large (Kuang et al., 2015; Ng et al., 2001) and (Stewart and Sun, 1990). Therefore, a robust mathematical framework called

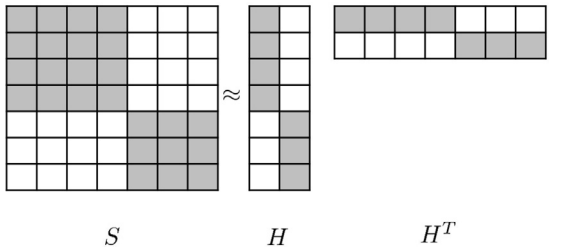

Fig. 6. An illustration of symmetric lower rank matrix approximation (dark and white colors represent big and small values respectively) Kuang et al. (2015).

"Symmetric Non-negative Matrix Factorization" (SNMF) has been utilized in this study to obtain clusters with high intra-similarities. SNMF is a type of lower rank matrix approximation (e.g. singular value decomposition, etc.) which enforces non-negativity to all the elements of the lower rank matrices. Indeed, spectral clustering and SNMF are two closely related approaches in terms of clustering objective function but with a totally different way of optimizing this objective function. The objective is to maximize the within-cluster similarities which can be reduced to a trace maximization form (Kuang et al., 2015; Kulis et al., 2009):

$$\begin{aligned}&\max_{H\geq 0,\ H^TH=I} \mathbf{Tr}(H^TWH) \Leftrightarrow\\ &\min_{H\geq 0,\ H^TH=I} \mathbf{Tr}(W^TW) - 2\mathbf{Tr}(H^TWH) + \mathbf{Tr}(H^TH) \Leftrightarrow\\ &\min_{H\geq 0,\ H^TH=I} \|W - HH^T\|^2 \end{aligned} \tag{6}$$

where $W \in \mathbb{R}^{N\times N}$ is the similarity matrix $H \in \mathbb{R}^{N\times N_s}$ is a clustering assignment matrix in which each row shows the membership values of a data object to different clusters. Note that N and N_s denote number of objects and desired number of clusters respectively where normally $N_s \ll N$. **Tr** is the trace of the matrix which is defined as the sum of the elements on the main diagonal. The two constraints $H^TH = I, H \geq 0$ on H make the problem become NP-hard. Spectral clustering and SNMF solve two different relaxations of the aforementioned problem (see Eq. (6)). SNMF retains the $H \geq 0$ constraint while spectral clustering retains $H^TH = I$ (see Luxburg, 2007 for details). The orthogonality constraint($H^TH = I$) points out that each data object should only belong to one of the clusters. However, by relaxing this constraint in SNMF, each data object can belong to multiple clusters with different membership values. In Ding et al. (2005), authors show that retaining $H \geq 0$ by SNMF also leads to near orthogonality of columns of matrix H (i.e. $H^TH \approx I$) which is necessary for partitioning purposes.

To find clusters with good balance in size, we use a normalized similarity matrix $\tilde{W} = D^{-1/2}WD^{1/2}$ in this study. This normalized similarity matrix corresponds to minimizing Normalized Cut objective function:

$$\mathrm{Ncut}(A_1, A_2, \ldots, A_k) = \sum_{i=1}^{k} \frac{\mathrm{cut}(A_i, \bar{A}_i)}{\mathrm{vol}(A_i)} \tag{7}$$

where $\mathrm{cut}(A_i, \bar{A}_i) = \sum_{u\in A_i, v\in \bar{A}_i} \tilde{w}(u, v)$ and $\mathrm{vol}(A) = \sum_{u\in A_i} d_u$. In this study, we utilize the Normalized Cut objective function as it takes into account the size of different clusters and makes a balance between them.

In SNMF framework, given a non-negative and symmetric similarity matrix $\tilde{W}$, and a reduced rank N_s (desired number of clusters), the aim is to find a lower rank matrix approximation H by minimizing the Frobenius norm of distances formulated as:

$$\min_{H\geq 0} \|\tilde{W} - HH^T\|^2 \tag{8}$$

where $\tilde{W} \in \mathbb{R}_+^{N\times N}$ and $H \in \mathbb{R}_+^{N\times N_s}$ are matrices with all non-negative elements.

The main concept of the algorithm can be easily understood using Fig. 6. Intuitively, data objects in each cluster should have large similarities to each other (high intra-similarity) while similarity values between objects in different clusters should be small (low inter-similarity). In Fig. 6, a similarity matrix with its lower rank approximation matrices are depicted where gray and white colors show large and small similarity values between points in the matrix S respectively. For a given object (row), different columns in the lower rank matrix approximation H can be interpreted as membership values of that object to different clusters. Hence, each link would be assigned to the cluster where it has the largest value of membership.

Different methods have been proposed in the literature to solve the optimization problem described in Eq. (8). For instance, Kuang et al. (2015) proposed a projected Newton framework that utilizes the gradient vector and Hessian matrix to find optimal matrix H^*. They also presented a non-symmetric formulation for SNMF and utilized Two-block Coordinate Decent Framework which has been successfully applied for standard NMF problems (Lee and Seung (1999), Lin (2007), Gong and Zhang (2012) and Gillis and Glineur (2012)). In this study, we utilize MATLAB 'fmincon' toolbox with interior-point method solver to optimize (8). The pseudo code of the proposed algorithm with all three steps is depicted in Algorithm 1.

Algorithm 1: 'Snake' algorithm

A. Running 'Snakes'
X: set of roads (links)
for $x \in X$ **do**
  $S_x \longleftarrow x$;
  while $Size(S_x) < N$ **do**
    $S' = \text{Adj}(S_x)$; (Neighboring links of the snake)
    $k^* = \{k | \min_{k \in S'} var(S_x \cup k)\}$;
    $S_x \longleftarrow [S_x, k^*]$;

B. Computing similarities
initialize ϕ;
$W \longleftarrow \mathbf{0}_{[\mathbf{N} \times \mathbf{N}]}$;
for $\forall i, j \in X$ **do**
  $k \longleftarrow 1$;
  while $k \leq N$ **do**
    $w(i,j) = w(i,j) + \phi^{N-k} \times \text{intersect}(S_{ik}, S_{jk})$;
    $k \longleftarrow k + 1$;

C. Symmetric Non-negative Matrix Factorization
$D = diag(d_i)$ **where** $d_i = \sum_{j=1}^{N} w(i,j)$;
$\tilde{W} = D^{-1/2} W D^{1/2}$;
$H^* = \{H | \min_{H \in \mathcal{R}_+^{N \times N_s}} \|\tilde{W} - HH^T\|^2\}$;
for $i \in X$ **do**
  $j^* = \{j | \max_{j \in \{1,\ldots,N_s\}} H(i,j)\}$;
  $i \in A_{j^*}$;
Output: $\{A_1, A_2, \ldots, A_{N_s}\}$ (Set of clusters)

The Non-negative Matrix Factorization (NMF) has been successfully applied as a clustering tool in various fields such as document clustering (Xu et al., 2003), community detection (Mankad and Michailidis, 2013), and transportation networks (Han and Moutarde, 2013). However, in this paper, we utilize SNMF approach which is more appropriate for clustering, as it gives us the flexibility to define any kind of similarity matrix.

3. Extension of the methodology for missing data

In real traffic networks, it is common that some links do not have data for specific time intervals. This could happen either when there are no sensors in the link or some sensors do not provide valid measurements. Hence, the graph of the networks (including only links with measurement) might have several disjoint parts and the snake algorithm will not work properly. In this section, we extend the framework of the clustering method to be applied in cases with missing data. In such models, we could estimate the value of missing data and try to make the graph connected and then partition the network. However, in this way we impose some bias by assuming values for missing data. Hence, we now extend the proposed snake algorithm to deal with cases with missing data especially when disjoint parts in the network are created.

Given that strong spatial correlations exist, we assume that a snake could utilize connected parts of the network without data to reach links with high level of similarity to the existing links. Nevertheless, this action can create by-pass to reach distant links of the network and violate the objective of connectivity. For this reason, we assign a penalty factor α that varies with distance to put a priority to the links that are closer. In addition, we define a distance threshold $\bar{d}$ up to which, we look for links with data to avoid by-passing heterogeneous sub-regions due to missing data. The value of penalty factor could be selected based on the number and location of the missing measurements. To grow snakes iteratively, we first find all the links with data that have distance up to $\bar{d}$. Among these links, we keep the two following groups: (i) links that are adjacent to the current set and have distance equal to one; (ii) links with distance more than one to the current set that have a connection with this set through some links with no data. The weighted difference (F_i) between the new data (x_i) and the average of the snake (μ) is calculated as follows:

$$F_i = \alpha^{r_i - 1} \times |x_i - \mu| \tag{9}$$

where r_i denotes the shortest path distance of the candidate link i to the snake and x_i shows the density (or speed) value of that link. We perform a sensitivity analysis on the parameters α and $\bar{d}$. Indeed, by increasing (decreasing) the penalty factor α (threshold $\bar{d}$), we prevent the algorithm to bypass some heterogeneous areas and add similar distant links. Therefore, the user is able to make a trade-off between the connectivity and homogeneity by adjusting these parameters.

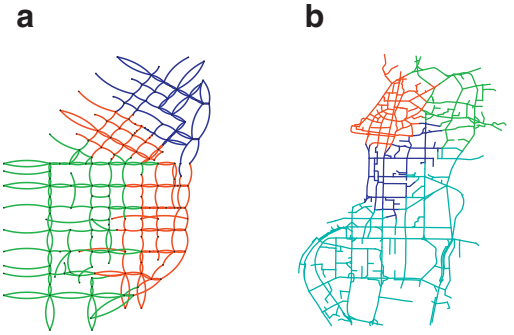

Fig. 7. Clustering results of NMB method applied in the two case studies: (a) San-Francisco; (b) Shenzhen.

Table 1
Average values μ and standard deviations σ of link densities [*veh*/(*km.lane*)] and speeds [*m*/*sec*] for obtained clusters using NMB method in the two case studies: (a) San-Francisco; (b) Shenzhen.

(μ/σ)	Blue	Red	Green	Cyan	TV_n
San-Francisco	7.48/8.34	28.6/21.73	14.97/16.76	-	0.8548
Shenzhen	9.18/2.56	8.26/1.86	9.96/2.20	10.44/2.67	0.8895

Basically, at each step, the best candidate (i.e. the link with the lowest weighted distance) together with the links (with no data) connecting it to the current snake are added to the snake sequence. This approach continues until all the links with data are added. By running from different initial points, we obtain arrays with different sizes since different number of links with missing data might be added to snakes with different initial points. To compute the similarity in the same way with the previous section (see Eq. (3)), different snakes should have the same size. Hence, we keep only the links with the data and delete the rest from the sequence. As a result, the effect of these links (with no data) is disregarded and all the arrays have the same size equal to the number of the links with data. As we will show in the result section, even with 40% of data, the method performs significantly well.

4. Results

In this section, the proposed clustering method is applied to two different case studies. The first one contains real data from GPS of 20,000 taxis in Shenzhen, China; while the second one is a microscopic simulation of downtown area of San-Francisco. The outcomes are presented and discussed in details and compared to the results of the three steps method called 'NCut-Merging-Boundary Adjustment' (NMB) proposed in Ji and Geroliminis (2012). Network is partitioned with different number of clusters and the Normalized total variance (TV_n) of the network is utilized to evaluate the performance of the algorithm for different number of clusters. Normalized total variance is defined as the ratio between total variance of the partitioned and unpartitioned network. Note that, smaller TV_n values imply bigger improvement gained by partitioning. Normalized total variance for N_s clusters is calculated as follows:

$$TV_n = \frac{\sum_{i=1}^{N_s} N_{A_i} \times var(A_i)}{N \times var(A)} \tag{10}$$

where $A = \bigcup_{i=1}^{N_s} A_i$ denotes the whole set of clusters and N_{A_i} denotes the number of links in cluster A_i while N is the total number of links in the network. Note that, $var(A_i)$ denotes the variance of the speed (density) of all links belong to cluster i. This metric not only evaluates the homogeneity of clusters, but also takes into account size of the clusters. The value of TV_n shows the performance of the proposed clustering algorithm. This metric prefers larger number of clusters and it does not consider the inter-cluster similarity (i.e. it is minimized to zero when each link is a single cluster)[5]. There are other criteria from a traffic point of view which are also important in finding the desired number of regions. For instance in control strategies (like perimeter control and gating) that work based on MFD concept in the partitioned networks, each region is modeled by a fitted MFD curve (polynomial) relating the average flow (production) to the average density (accumulation). Hence, regions with low scatter MFD can be better aligned into a fitted curve which implies a more precise modeling. The scatter of MFD curve in each region depends on the degree of homogeneity (see Geroliminis and Sun, 2011) and number of aggregated links (size of a region). Individual detectors has high scatter fundamental diagram and it is not accurate to

[5] There are some metrics in the literature that contribute to the estimation of the optimal number of clusters like the 'Silhouette' metric proposed by (Rousseeuw, 1987). The silhouette of a datum is a measure of how closely (loosely) it is matched to data within its (the neighbouring) cluster, i.e. the cluster whose average distance from the datum is lowest.

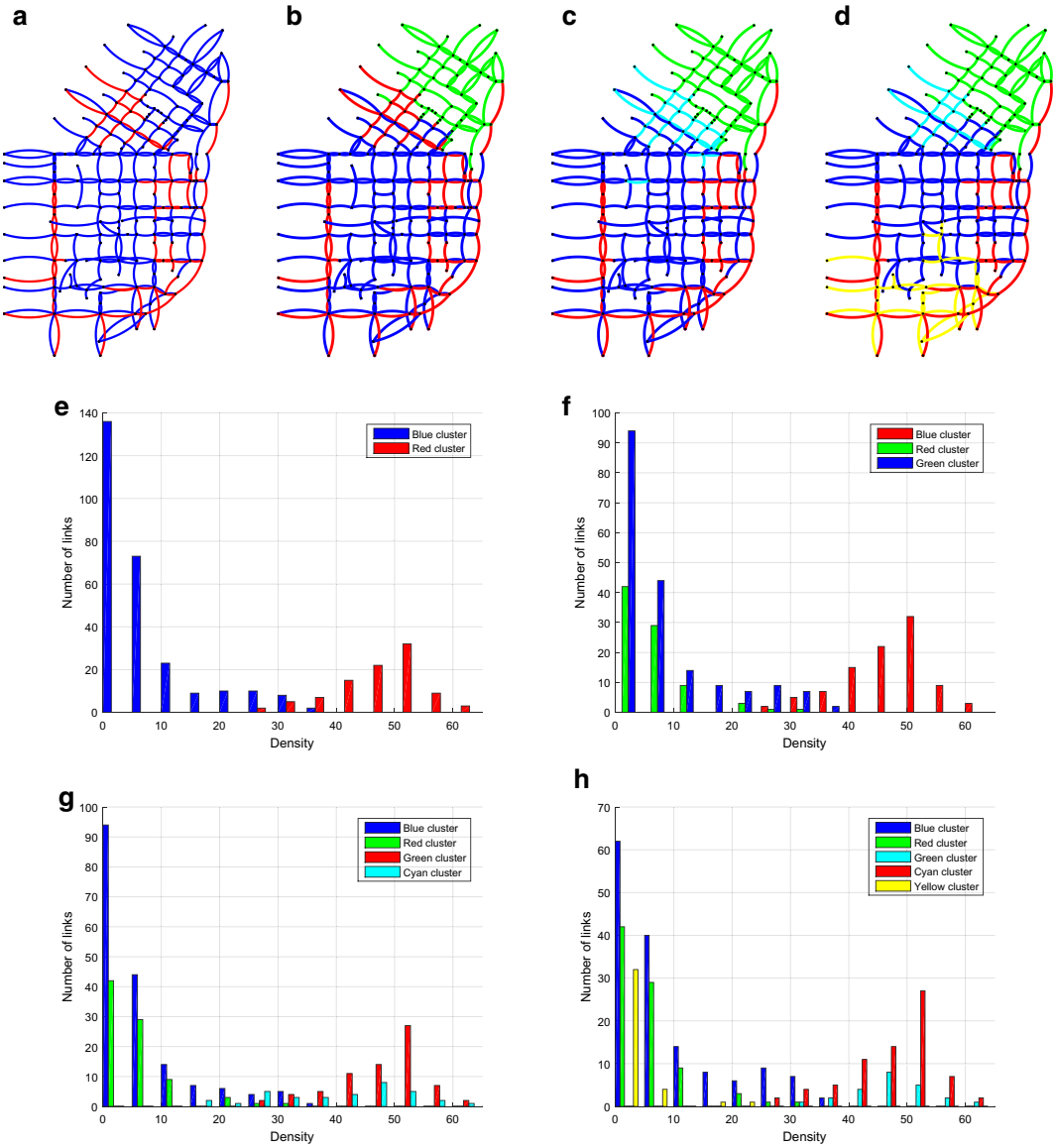

Fig. 8. Clustering results (a-d) and histogram of link densities [*veh*/(*km.lane*)] in different clusters (e-h) for cases with 2–5 clusters.

Table 2
Average values (μ) and standard deviation (σ) of link densities ([*veh*/(*km.lane*)]) and TV_n of obtained clusters in San-Francisco network for cases with 2–5 clusters.

(μ/σ)	Blue	Red	Green	Cyan	Yellow	TV_n
2	7.73/8.28	47.43/7.22	-	-	-	0.175
3	8.30/9.16	47.43/7.22	6.48/5.71	-	-	0.174
4	7.11/7.94	47.32/7.38	6.48/5.71	41.15/11.56	-	0.166
5	9.57/9.64	47.32/7.38	6.48/5.71	47.80/6.68	3.36/4.33	0.165

model each individual detector by a fitted curve. Consequently, we are looking for homogeneous regions that are not small (individual links are not desirable). Moreover, it is not practically feasible to have many regions with many state variables as it increases the computational complexity of the modeling part and it cannot be applied in real-time traffic management scheme.

The results of NMB method applied in two case studies are depicted in Fig. 7 and Table 1 to highlight in the next sections the added values of the new method both in terms of TV_n decrease and the ability of identifying directional congestion.

Table 3
Normalized total variance (TV_n) versus size of snakes for cases with 2–5 clusters (San-Francisco).

Size of snake	Number of clusters			
	2 clusters	3 clusters	4 clusters	5 clusters
10	0.996	0.818	0.642	0.429
40	0.890	0.411	0.174	0.174
60, 80	0.890	0.174	0.174	0.165
100, 120, . . . , 180	0.175	0.174	0.174	0.165
200, 220, . . . , 360	0.175	0.174	0.166	0.165

4.1. San-Francisco network

The proposed algorithm is applied in the downtown area of San-Francisco network. As we explained in the methodological part, different snakes run from all initial points and then based on the obtained sequence of roads, a similarity is calculated for each pair of links using Eq. (3). The results of clustering with 2–5 clusters and the histogram of densities for different clusters are depicted in Fig. 8 using different colors. Average values and standard deviations of links densities, and TV_n for different clusters are presented in Table 2. All Standard deviations of the obtained clusters (ranged from 4 to 11 [*veh*/(*km.lane*)]) have lower values compared to the average local heterogeneity (ranged from 14 to 20 [*veh*/(*km.lane*)]) calculated for different distances (see Fig. 2(a)). This implies that spatial correlations cannot suffice efficient clustering if they are not combined with a proper algorithm.

The results show promising performance of the clustering method in separating homogeneous connected clusters. Note that for a small number of clusters, histograms might contain a "fat tail" that increase the heterogeneity of each cluster. It is clear in Fig. 8 that directional congestion is detected and belongs to different clusters. From histogram plots of link densities, it can be easily seen that clusters have low variance compared to the variance of the whole network. When increasing the number of clusters from 2 to 3, the algorithm bi-partitions (even if it is not enforced to do so) the uncongested cluster (blue color in Fig. 8(a) in two smaller clusters (blue and green in Fig. 8(b)). While the value of TV_n remains about constant (0.174 vs. 0.175), these two clusters are barely connected (only with a few links), so such a partitioning is considered beneficial as the clustered regions have smoother shape. Note also how well this framework can capture stripes of congestion, as the congested links in the left of the network. By comparing Figs. 1(c) with 7(a) and 8(a), it is clear that the congested stripe in the left part of the network is well identified with the snake algorithm, while it is part of a less congested cluster with the NMB method. The improvement compared to NMB method is about 80% (TV_n metric is 0.8548 for NMB method) and as you see in Table 1, clusters have bigger variance values compared to the obtained results with the proposed snake algorithm. The reason is mainly because of detecting directional congestion and the fact that snake algorithm does not depend on the network structure (grid-type, low connected).

Since a snake will grow and grab all links in the network and the number of snakes is equal to the number of links N, this algorithm is computationally expensive specifically for big networks in terms of running snakes, estimating the similarity matrix W, and finding optimal symmetric non-negative matrix H^* in Eq. (8). Now we study the effect of snake size on the performance of the proposed algorithm. The values of TV_n for different snake sizes and number of clusters are presented in Table 3. As can be observed, the algorithm produces similar results for sizes bigger than 100 links (almost 1/3 of the whole network) in all different number of clusters. Note that in case of SF with 360 links, a snake of size 100 links is sufficient for 2 and 3 clusters, while for a larger size of clusters (4 or 5) even snakes with a size 1/6 of the network are appropriate. Comparing these results to the results of NMB method in Table 1 shows the superiority of the snake method even with small sizes (i.e. TV_n in 4 and 5 clusters are smaller than the best achieved with NMB method). This is due to the fact that the similarity is defined in more precise way compared to NMB method where similarity is assumed only between neighboring links.

4.2. Shenzhen network

A more challenging test is to investigate the proposed method in a large network with hierarchical topology and complex connectivity since the way traffic evolves and congestion propagates is much different from the grid network presented above. The speed in the Shenzhen network has been estimated based on GPS of taxis with passengers and additional estimation errors exist due to small sample size, map-matching, and measurements error. To do so, we apply clustering algorithm in Shenzhen network during the peak period of congestion and examine the performance with different number of clusters. The results of the clustering method with histogram of speeds for different clusters are depicted in Fig. 9. It can be inferred from the clustering results in Fig. 9 that the network has two main pockets of congestion (clusters with small average speed) represented by blue and red colors in Fig. 9(b). An interesting observation is that, the proposed clustering algorithm detects part of highway depicted with yellow color in Fig. 9(d) in which the average speed is higher than the rest of clusters in the network.

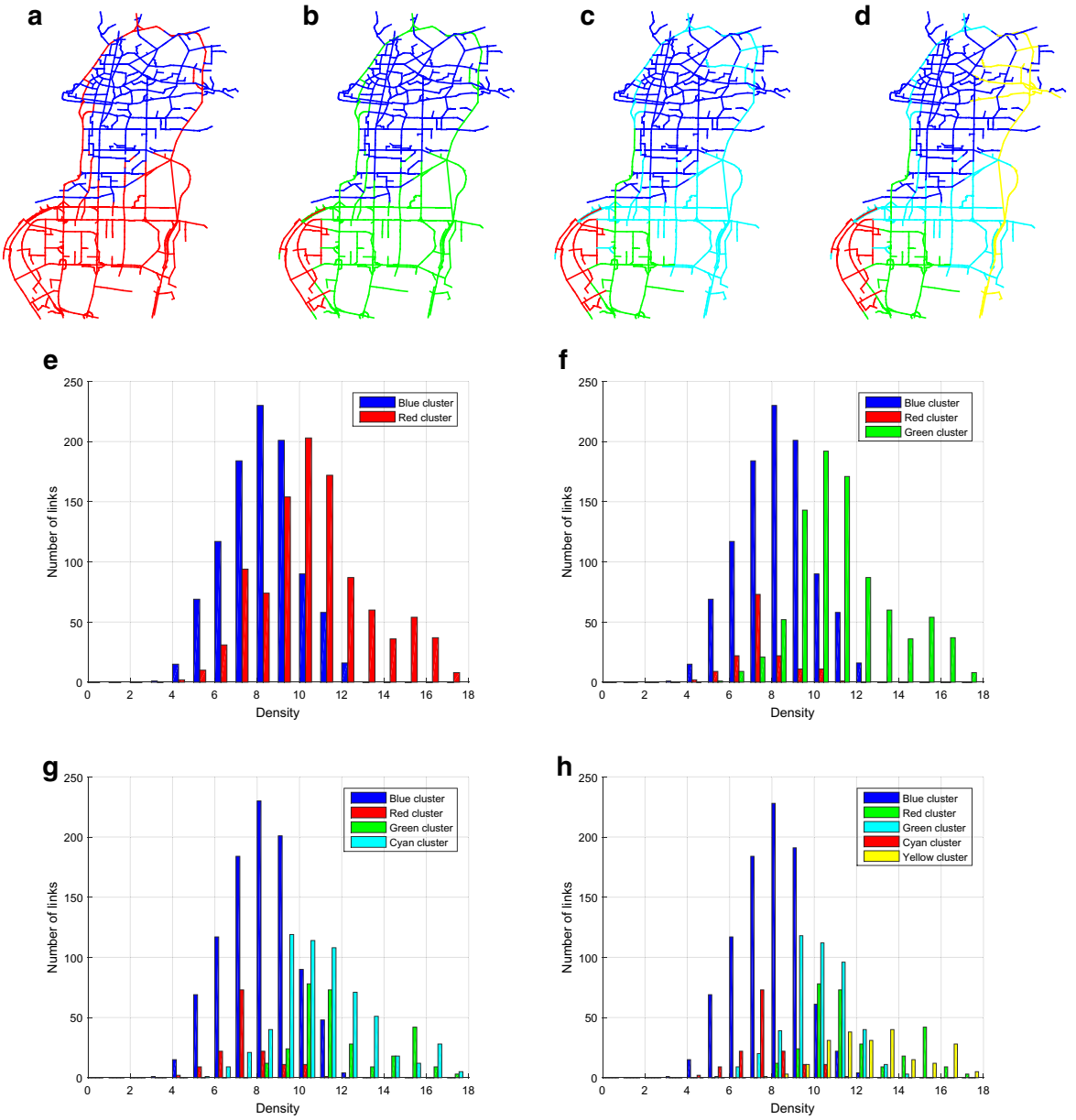

Fig. 9. Clustering results (a-d) and histogram of link speeds [*m*/*sec*] in different clusters (e-h) for cases with 2–5 clusters.

Table 4
Average values (μ) and standard deviation (σ) of link speeds ([*m*/*sec*]) and TV_n of obtained clusters in Shenzhen network for cases with 2–5 clusters.

(μ/σ)	Blue	Red	Green	Cyan	Yellow	TV_n
2	8.41/1.71	11.03/2.63	-	-	-	0.744
3	8.41/1.71	7.77/1.25	11.59/2.39	-	-	0.604
4	8.32/1.63	7.77/1.25	12.12/2.20	11.35/2.40	-	0.573
5	8.14/1.53	7.77/1.25	12.12/2.20	10.34/1.50	13.19/2.45	0.455

By comparing average values of speed in different clusters presented in Table 4, we could easily see that our method has the ability to differentiate between clusters with different level of congestion. Although NMB method works quite well in grid networks, this method has difficulties when it is applied to the networks with fewer connections. It was observed that for such networks, NMB tends to cut the graph in locations that have less

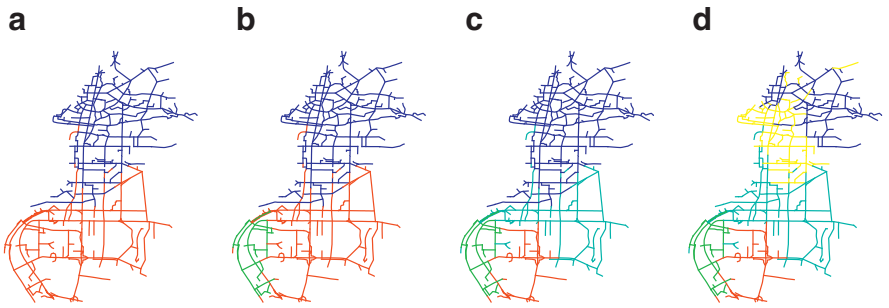

Fig. 10. Clustering results of the snake algorithm in Shenzhen network (excluding the highway) for cases with 2–5 clusters.

Table 5
Average values (μ) and standard deviation (σ) of link speeds ([m/sec]) and TV_n of obtained clusters in Shenzhen network (excluding the highway) for cases with 2–5 clusters.

(μ/σ)	Blue	Red	Green	Cyan	Yellow	TV_n
2 clusters	8.51/1.78	9.96/1.81	-	-	-	0.86
3 clusters	8.51/1.78	10.49/1.50	7.77/1.25	-	-	0.72
4 clusters	8.51/1.78	11.16/1.37	7.77/1.25	10.10/1.44	-	0.69
5 clusters	9.52/1.48	11.16/1.37	7.77/1.25	9.88/1.45	7.25/1.37	0.53

Table 6
Normalized total variance (TV_n) vs. size of snake for cases with 2–5 clusters (Shenzhen).

Size of snake	Number of clusters			
	2 clusters	3 clusters	4 clusters	5 clusters
100	0.975	0.964	0.810	0.740
150	0.933	0.743	0.630	0.503
200	0.933	0.735	0.595	0.458
250	0.744	0.735	0.595	0.513
300	0.744	0.735	0.595	0.455
350, 400	0.744	0.604	0.595	0.455
600, 800, . . . , 2000	0.744	0.604	0.573	0.455

connectivity[6]. Fig. 7(b) depicts the clustering results obtained by NMB method. Note that the best clustering result obtained in NMB method has $TV_n = 0.8895$ for 4 clusters. Comparing the results presented in the Table 4 with NMB algorithm presented in Table 1 highlights the superiority of the proposed method in the networks with complex connectivity (about 50% improvement).

As it was stated before, the study area of Shenzhen includes a highway (see Fig. 1(b)) that has different traffic properties than urban streets. To evaluate the performance of algorithm for the cases where the types of the roads are available, we exclude the highway and apply the partitioning algorithm to the network with only urban streets that have similar traffic properties. Fig. 10 depicts the partitioning results for different number of clusters (2–5) and Table 5 represents the average speed values with standard deviation of the obtained clusters. Note how similar are the shapes of clusters by comparing Figs. 10(a), 10(b), and 10(c) with 9(a), 9(b), and 9(c). The results for 5 clusters are a bit different, because once freeways are included, the algorithm tries to integrate higher speed in the yellow cluster of Fig. 9(d). The average speed values in the new case are lower due to the fact that highway with higher speed value is excluded form the study area. It is worth mentioning that for the cases where the types of the roads are not available or in other applications like urban planning (population, economic, etc.), the proposed clustering approach performs well and is able to group roads with similar traffic properties in the same clusters and even identify the freeway links.

To investigate the performance of the algorithm under different snake sizes for the case study of Shenzhen network, we repeat the clustering considering only part of each snake in calculating the similarity matrix. Table 6 presents the results

[6] In spectral clustering, as the objective is to minimize the inter-similarity between the clusters, the definition of similarity plays a critical role. To impose connectivity, the similarity in NMB method is defined only between neighboring links. Hence, only adjacent roads between two neighboring clusters are considered in the calculation of inter-similarity. In this case, the algorithm prefers to cut the graphs from the parts with few connections.

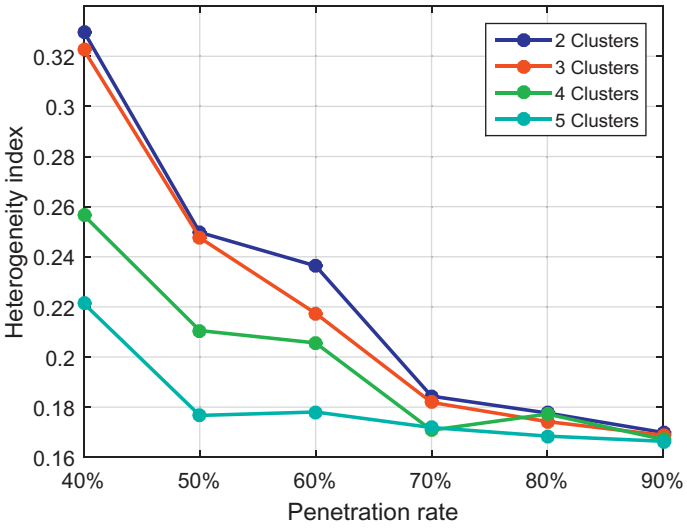

Fig. 11. Clustering performances ($TV_{missing}$) under different penetration rates for cases with 2–5 clusters.

corresponding to different snake sizes and number of clusters. Note that, in this case study we see a robust performance of the algorithm even for a size of 300 (which is the 1/7 of the network). By truncating snakes at the size of 300, we are able to decrease the computational time needed for the algorithm for about 8 times (i.e. the computational time to partition the network into 5 clusters is reduced from 9360 to 1140 seconds).

4.3. Missing data

To assess the performance of the proposed extended algorithm for the networks with missing data, we design several experiments with different penetration rates (P = 40%, 50%, . . . , 90%), indicating the percentage of links with data, in San-Francisco network. For each penetration rate, 12 different random sets of links are generated and the extended clustering algorithm is applied to each of these random sets. To quantify and compare clustering performance under different penetration rates, a metric similar to TV_n (see Eq. (10)) is proposed in Eq. (11). Note that, to have a fair comparison, the denominator of Eq. (11) considers only the set of links with data which differs depending on random selected links. The average value of 12 different random sets is calculated for each penetration rate and depicted in Fig. 11.

$$TV_{missing} = \frac{\sum_{i=1}^{N_s} N_{A_i} \times var(A_i)}{N_{A_{sample}} \times var(A_{sample})} \tag{11}$$

where $A_{sample} = \bigcup_{i=1}^{N_s} A_i$ represent the set of links with data that is partitioned into N_s clusters.

For high penetration rates, it is possible that extended framework achieves better results than the normal snake algorithm applied in the network with full information. The reason is that for high penetration rates, network is still connected and inside the extended framework, snake is allowed to look for similar links up to a certain distance rather than only considering adjacent links. However, by decreasing the penetration rates, it is more likely to have disjoint parts in the network and the efficiency of the algorithm starts decreasing (see sharp jump in corresponding to 40% penetration rate) but the results are still reliable compare to NMB method. The clustering results of a sample set for two cases with 80% and 50% penetration rates are depicted in Fig. 12. Note that, the results are for the case where $\alpha = 3$ and $\bar{d} = 3$. As it could be seen, the clustering result is close to the one obtained in Fig. 8 with full information. By decreasing the penetration rate, the total variance of obtained clusters increases. However, the main pockets of congestion can be still detected using the proposed extended approach (see Fig. 12(b)). Given these results, a future research direction is to identify the optimal number of detectors needed to be installed in the network. Note that not only the number of detectors is important but also the location of the detectors plays an important role. Note also that in this case, detectors will not be installed randomly as in this designed experiment, but spatial correlations of links can be considered. If detectors locations are properly chosen, it is expected that the algorithm can perform equally well even with a lower penetration rate (recent works have shown that a 10–20 percent of the links installed with detectors can provide a good estimate of MFD for single regions (Courbon and Leclercq, 2011; Ortigosa et al., 2013)). This should be a research priority.

We perform a sensitivity analysis of the parameters α and $\bar{d}$ (see Eq. (9)) for two cases of 50% and 40% penetration rates. As it is expected, by increasing (decreasing) the threshold $\bar{d}$ (penalty factor α), the algorithm is able to find clusters with lower normalized total variance variance (see Table 7). Indeed, snakes are allowed to bypass some heterogeneous areas and add similar links. Hence, clusters have less connectivity but more homogeneity. In the extended snake approach, some

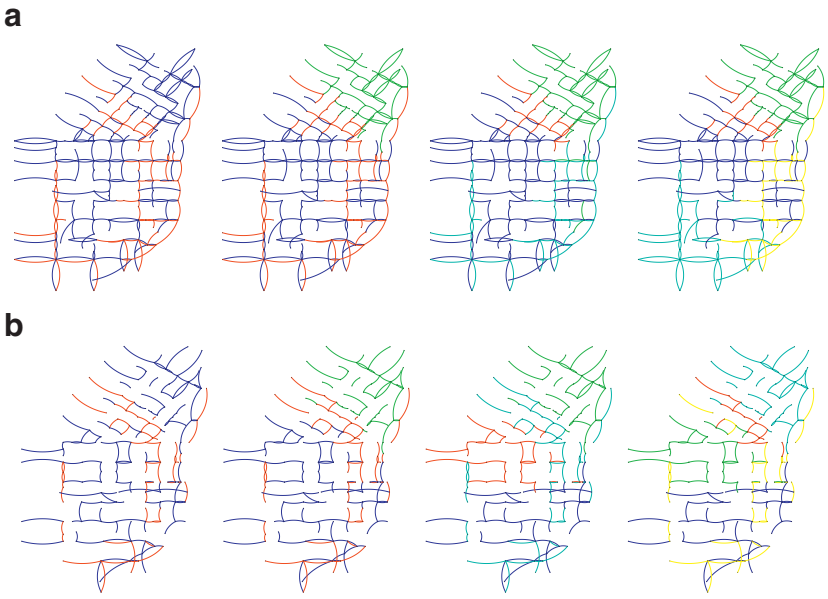

Fig. 12. Clustering results with different number of clusters for two penetration rates: (a) P = 80%; (b) P = 50%.

Table 7
Normalized total variance TV_n for different values of parameters ($\bar{d}$, α) in the extended snake algorithm.

	(a) P= 50%, $\bar{d} = 3$			
α	Number of clusters			
	2	3	4	5
1	0.157	0.127	0.107	0.095
2	0.174	0.192	0.155	0.122
3	0.250	0.248	0.211	0.177
5	0.389	0.357	0.329	0.256
10	0.877	0.587	0.498	0.433
	(b) P= 50%, $\bar{d} = 5$			
α	Number of clusters			
	2	3	4	5
1	0.137	0.105	0.073	0.053
2	0.160	0.137	0.124	0.010
3	0.181	0.187	0.172	0.135
5	0.368	0.314	0.273	0.246
10	0.777	0.560	0.491	0.412
	(c) P= 40%, $\bar{d} = 3$			
α	Number of clusters			
	2	3	4	5
1	0.162	0.132	0.137	0.112
2	0.233	0.262	0.182	0.171
3	0.320	0.322	0.256	0.221
5	0.661	0.511	0.443	0.345
10	0.882	0.788	0.646	0.560
	(d) P= 40%, $\bar{d} = 5$			
α	Number of clusters			
	2	3	4	5
1	0.152	0.116	0.079	0.060
2	0.163	0.157	0.139	0.111
3	0.210	0.290	0.244	0.191
5	0.535	0.417	0.360	0.306
10	0.881	0.752	0.632	0.553

Table 8
Percentage of improvement in $TV_{missing}$ gained by applying the extended snake algorithm compared to the case of estimating missing data, under different penetration rates and number of clusters.

Number of clusters	Penetration rate %					
	40	50	60	70	80	90
2 clusters	61.5	68.4	60.3	62.0	41.6	9.8
3 clusters	38.7	43.4	36.2	41.2	26.1	9.5
4 clusters	32.5	38.8	25.9	15.4	0.5	0.8
5 clusters	33.1	43.3	33.2	13.4	6.0	1.1

links with missing data might be utilized in several clusters to make connections between disjoint parts of the network. We are currently studying the formulation of clustering where connectivity is explicitly enforced by some constraints while heterogeneity is minimized in the objective function. In that framework, links with missing data are also assigned to clusters and contribute in connecting disjoint parts whereas they do not contribute in the objective function. This formulation is more accurate as it allows links with missing data to be only utilized in one cluster.

An alternative approach for the proposed extended algorithm would be to estimate the density (speed) values of the links with missing data using the values of neighboring links (e.g. average density of the adjacent links) and utilize the snake algorithm with full information. Table 8 presents the relative improvement gained by applying the extended snake algorithm compared to the approach with the estimation of missing data (i.e. relative decrease in the value of $TV_{missing}$). The result demonstrates the superiority of the extended snake algorithm specifically in cases with small penetration rates where estimation of missing information causes big bias.

5. Conclusions

This paper presents a method to partition urban traffic networks using link information and network structure. The significance of the proposed method is summarized as follows: (i) it could deal with networks with different structures varying from perfect grid to the networks with low connectivity; (ii) it could capture very well directional congestion which happens in many real networks in morning and evening peak hours; (iii) it could detect pockets of congestion in different locations and assign them to separate clusters. The proposed framework is also extended to deal with networks with missing data which makes this approach more flexible to work under real conditions. The method takes into account the dependencies between adjacent links, value of the links and size of clusters in three steps. In this framework, connected clusters have been obtained by putting more weight on the similarity of the links that are spatially close to each other rather than forcing it explicitly. The outcome of proposed method for two case studies with field and simulated data seems robust to network structure and more accurate compared to a NMB method (Ji and Geroliminis, 2012). This approach also needs less effort in terms of parameter calibration.

In addition, we plan to extend the proposed method to dynamic framework by incorporating time correlation in the method. Since homogeneous clusters have low scatter MFDs, the output of this work can be utilized in hierarchical control schemes like perimeter control which works based on the concept of MFDs. As a future work, it would be challenging to build the correlation in time to extend the framework from static to dynamic partitioning. Another research priority is to carefully investigate the interactions of clustering and control. While spatial homogeneity should be an important property of clusters, the shape of the clusters and also their boundaries should facilitate the control objectives. Given that drivers tend to choose routes without too many turns (especially for networks with close to grid structure), a non-smooth boundary where perimeter control is applied might create shortest paths with a large number of turns and change the behavior of drivers in non-predictable ways. More specifically, it can create optimal routes that are of non-smooth shape, which will make convergence of a system to equilibrium almost impossible.

Acknowledgments

This work has been supported by the (European Research Council) ERC Starting Grant “METAFERW: Modeling and controlling traffic congestion and propagation in large-scale urban multimodal networks”.

References

Buisson, C., Ladier, C., 2009. Exploring the impact of homogeneity of traffic measurements on the existence of macroscopic fundamental diagrams. Transp. Res. Rec. 2124, 127–136.
Chiabaut, N., 2015. Evaluation of a multimodal urban arterial: the passenger macroscopic fundamental diagram. Transp. Res. Part B 81 (Part 2), 410–420.
Chiabaut, N., Xie, X., Leclercq, L., 2014. Performance analysis for different designs of a multimodal urban arterial. Transportmetrica B 2 (3), 229–245.
Courbon, T., Leclercq, L., 2011. Cross-comparison of macroscopic fundamental diagram estimation methods. Procedia - Soc. Behav. Scie. 20, 417–426.
Daganzo, C.F., 2007. Urban gridlock: macroscopic modeling and mitigation approaches. Transp. Res. Part B 41 (1), 49–62.

Daganzo, C.F., Gayah, V.V., Gonzales, E.J., 2011. Macroscopic relations of urban traffic variables: bifurcations, multivaluedness and instability. Transp. Res. Part B 45 (1), 278–288.
Ding, C.H., He, X., Simon, H.D., 2005. On the equivalence of nonnegative matrix factorization and spectral clustering. In: SIAM International Conference on Data Mining, 5, pp. 606–610.
Ding, C.H.Q., He, X., Zha, H., Gu, M., Simon, H.D., 2001. A min-max cut algorithm for graph partitioning and data clustering. In: Proceedings of the IEEE International Conference on Data Mining, pp. 107–114.
Du, J., Rakha, H., Gayah, V.V., 2015. Deriving macroscopic fundamental diagrams from probe data: Issues and proposed solutions. Transp. Res. Part C.
Duque, J.C., Ramos, R., Suriñach, J., 2007. Supervised regionalization methods: a survey. Int. Reg. Sci. Rev. 30 (3), 195–220.
Gayah, V.V., Daganzo, C.F., 2011. Clockwise hysteresis loops in the macroscopic fundamental diagram: an effect of network instability. Transp. Res. Part B 45 (4), 643–655.
Gazis, D.C., Herman, R., Potts, R.B., 1959. Car-following theory of steady-state traffic flow. Oper. Res. 7 (4), 499–505.
Geroliminis, N., Daganzo, C.F., 2008. Existence of urban-scale macroscopic fundamental diagrams: some experimental findings. Transp. Res. Part B 42 (9), 759–770.
Geroliminis, N., Sun, J., 2011. Properties of a well-defined macroscopic fundamental diagram for urban traffic. Transp. Res. Part B 45 (3), 605–617.
Gillis, N., Glineur, F., 2012. A multilevel approach for nonnegative matrix factorization. J. Comput. Appl. Math. 236 (7), 1708–1723.
Gipps, P., 1981. A behavioural car-following model for computer simulation. Transp. Res. Part B 15 (2), 105–111.
Godfrey, J.W., 1969. The mechanism of a road network. Traffic Eng. Control 11, 323–327.
Gong, P., Zhang, C., 2012. Efficient nonnegative matrix factorization via projected newton method. Pattern Recognit. 45 (9), 3557–3565.
Guo, D., 2008. Regionalization with dynamically constrained agglomerative clustering and partitioning (redcap). Int. J. Geogr. Inf. Sci. 22 (7), 801–823.
Haddad, J., Ramezani, M., Geroliminis, N., 2013. Cooperative traffic control of a mixed network with two urban regions and a freeway. Transp. Res. Part B 54, 17–36.
Haddad, J., Shraiber, A., 2014. Robust perimeter control design for an urban region. Transp. Res. Part B 68, 315–332.
Hagen, L., Kahng, A., 1992. New spectral methods for ratio cut partitioning and clustering. IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 11 (9), 1074–1085.
Han, Y., Moutarde, F., 2013. Statistical traffic state analysis in large-scale transportation networks using locality-preserving non-negative matrix factorisation. Intell. Transp. Syst. IET 7 (3), 283–295.
Helbing, D., 2001. Traffic and related self-driven many-particle systems. Rev. Mod. Phys. 73, 1067–1141.
Herman, R., Lam, T., Prigogine, I., 1972. Kinetic theory of vehicular traffic: comparison with data. Transp. Sci. 6 (4), 440–452.
Herman, R., Prigogine, I., 1979. A two-fluid approach to town traffic. Science 204 (4389), 148–151.
Ji, Y., Geroliminis, N., 2012. On the spatial partitioning of urban transportation networks. Transp. Res. Part B 46 (10), 1639–1656.
Ji, Y., Luo, J., Geroliminis, N., 2014. Empirical observations of congestion propagation and dynamic partitioning with probe data for large-scale systems. Transp. Res. Rec. 2422, 1–11.
Jiawei Han, M.K., 2000. Data Mining: Concepts and Techniques. Morgan Kaufmann Series in Data Management Systems.
Keyvan-Ekbatani, M., Yildirimoglu, M., Geroliminis, N., Papageorgiou, M., 2015. Multiple concentric gating traffic control in large-scale urban networks. IEEE Trans. Intell. Transp. Syst. 16 (4), 2141–2154.
Knoop, V., Hoogendoorn, S., 2013. Empirics of a generalized macroscopic fundamental diagram for urban freeways. Transp. Res. Rec. (2391) 133–141.
Kuang, D., Yun, S., Park, H., 2015. Symnmf: nonnegative low-rank approximation of a similarity matrix for graph clustering. J. Global Optim. 62 (3), 545–574.
Kulis, B., Basu, S., Dhillon, I., Mooney, R., 2009. Semi-supervised graph clustering: a kernel approach. Mach. Learn. 74 (1), 1–22.
Lancichinetti, A., Fortunato, S., 2009. Community detection algorithms: a comparative analysis. Phys. Rev. E 80, 056117.
Leclercq, L., Chiabaut, N., Trinquier, B., 2014. Macroscopic fundamental diagrams: a cross-comparison of estimation methods. Transp. Res. Part B 62, 1–12.
Lee, D.D., Seung, H.S., 1999. Learning the parts of objects by non-negative matrix factorization.. Nature 401 (6755), 788–791.
Lighthill, M.J., Whitham, G.B., 1955. On kinematic waves. ii. a theory of traffic flow on long crowded roads. Proc. R. Soc. Lond. A 229 (1178), 317–345.
Lin, C.J., 2007. Projected gradient methods for nonnegative matrix factorization. Neural Comput. 19 (10), 2756–2779.
Luxburg, U., 2007. A tutorial on spectral clustering. Stat. Comput. 17 (4), 395–416.
Mahmassani, H.S., Saberi, M., Zockaie, A., 2013. Urban network gridlock: theory, characteristics, and dynamics. Transp. Res. Part C 36, 480–497.
Mahmassani, H.S., Williams, J., Herman, R., 1987. Performance of urban traffic networks. In: Proceedings of the 10th International Symposium on Transportation and Traffic Theory. Amsterdam, The Netherlands, pp. 1–20.
Mankad, S., Michailidis, G., 2013. Structural and functional discovery in dynamic networks with non-negative matrix factorization. Phys. Rev. E 88 (4), 042812.
Mazloumian, A., Geroliminis, N., Helbing, D., 2010. The spatial variability of vehicle densities as determinant of urban network capacity. Philos. Trans. R. Soc. London A 368 (1928), 4627–4647.
McGill, R., Tukey, J.W., Larsen, W.A., 1978. Variations of box plots. Am. Statist. 32 (1), 12–16.
Nagel, K., Schreckenberg, M., 1992. A cellular automaton model for freeway traffic. J. Phys. I France 2 (12), 2221–2229.
Ng, A.Y., Jordan, M.I., Weiss, Y., 2001. On spectral clustering: analysis and an algorithm. Adv. Neural Inf. Process. Syst. 14, 849–856.
Ortigosa, J., Menendez, M., Tapia, H., 2013. Study on the number and location of measurement points for an mfd perimeter control scheme: a case study of zurich. EURO J. Transp. Logist. 3 (3), 245–266.
Payne, H.J., 1971. Models of freeway traffic and control. Simul. Councils Proc. Ser. 1 (1), 51–61.
Ramezani, M., Haddad, J., Geroliminis, N., 2015. Dynamics of heterogeneity in urban networks: aggregated traffic modeling and hierarchical control. Transp. Res. Part B 74, 1–19.
Rousseeuw, P.J., 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. 20, 53–65.
Shi, J., Malik, J., 2000. Normalized cuts and image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 22 (8), 888–905.
Stewart, G.W., Sun, J.g., 1990. Matrix Perturbation Theory. Academic press, New York.
Whitham, G.B., 1974. Linear and Nonlinear Waves. Wiley, New York.
Xu, W., Liu, X., Gong, Y., 2003. Document clustering based on non-negative matrix factorization. ACM, New York, NY, USA, pp. 267–273.
Zheng, N., Geroliminis, N., 2013. On the distribution of urban road space for multimodal congested networks. Transp. Res. Part B 57, 326–341.