Discovering Spatial Mixture Patterns of Interest

Yiqun Xie
xie@umd.edu
University of Maryland

Han Bao
han-bao@uiowa.edu
University of Iowa

Yan Li
lixx4266@umn.edu
University of Minnesota

Shashi Shekhar
shekhar@umn.edu
University of Minnesota

ABSTRACT

Given a collection of N geo-located point samples of k types, we aim to detect spatial mixture patterns of interest, which are sub-regions of the study area that have significantly high or low mixture of points of different types. Spatial mixture patterns have important applications in many societal domains, including resilience of smart cities and communities, biodiversity, equity, business intelligence, etc. The problem is challenging because ranking and selection of candidate patterns can be highly susceptible to the effect of natural randomness, and real-world data often consists of various mixture patterns. In related work, the multi-nomial scan statistic does not support identification of high or low mixture due to its "direction-less" nature and high sensitivity to the composition of mixture patterns in data. While species richness indices in biodiversity research allow specification of directions, the measures are very sensitive to spatial randomness effects. To bridge the gap, we first propose a spatial mixture index to provide robust ranking among candidate patterns. Then, we present a dual-level Monte-Carlo estimation method with a baseline algorithm for spatial mixture pattern detection. Finally, we propose both an exact algorithm and a distribution-inspired sequence-reduction heuristic to accelerate the baseline approach. Experiment results with both synthetic and real-world data show that the proposed approaches can detect mixture patterns with high accuracy, and the acceleration methods can greatly reduce computational cost while maintaining high solution quality.

CCS CONCEPTS

• **Information systems → Data mining**; **Spatial-temporal systems**.

KEYWORDS

Spatial mixture pattern, spatial mixture index, statistical robustness

ACM Reference Format:
Yiqun Xie, Han Bao, Yan Li, and Shashi Shekhar. 2020. Discovering Spatial Mixture Patterns of Interest. In *28th International Conference on Advances in Geographic Information Systems (SIGSPATIAL '20), November 3–6, 2020, Seattle, WA, USA*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3397536.3422217

SIGSPATIAL '20, November 3–6, 2020, Seattle, WA, USA

ACM ISBN 978-1-4503-8019-5/20/11…$15.00
https://doi.org/10.1145/3397536.3422217

1 INTRODUCTION

Given a collection of N geo-located point samples of k types or classes (e.g., genera or species, service types) in a study area, we aim to detect spatial mixture patterns of interest, that is, sub-regions of the study area that have a significantly high or low mixture of different types of points.

Patterns of spatial mixtures are important in many application domains, such as smart cities, biodiversity, agriculture, equity, business intelligence, etc. In smart cities, for example, identifying regions with low-mixtures of tree genera (i.e., regions with a treescape dominated by very few types of trees) has become a critical and urgent task for ensuring the resilience and security of green infrastructure [15, 17]. In the last few decades, trees diseases (e.g., by insects or fungus infestation) have ravaged treescapes in many US states and caused tree deaths in the tens of millions [2, 3, 15, 17]. The damage has especially impacted urban settings that are dominated by few tree genera. As an example, after elm trees were wiped out by Dutch elm disease in Midwest regions, ash trees were chosen as a common replacement. However, the recent spread of Emerald ash borer is now threatening to wipe out the ash tree population estimated to cost over $10 billion to remedy. These waves of disturbance exposed the weakness of the low-mixture system and pushed policy makers to re-evaluate the resilience of the composition of tree types as well as other natural resources to reduce the impact of the next threat. Identification of city zones with a low mixture of tree types has become a critical need for decision making. The ability to detect spatial mixture patterns can also provide valuable information to related biodiversity applications (e.g., protect high-mixture regions and make timely interventions to vulnerable low-mixture zones). These are just a few of many examples.

The problem has three major challenges. First, detection of spatial mixture patterns requires ranking among a large number of candidate regions, but such ranking can be easily disturbed by effects of natural randomness exhibited in the process (detailed in Sec. 3.1). Second, real-world data often consists of many different spatial mixture patterns, especially when the cardinality of types is large (e.g., a typical zone in a city often has tens of tree genera and hundreds of tree species). This challenges traditional point-process based measures, which assume very few varieties. Third, the enumeration space of candidate sub-regions is often very large, leading to high computational cost especially with significance testing.

In the literature, there are two lines of work related to the topic of spatial mixture patterns. The most relevant is from the family of spatial scan statistics [12] – the multi-nomial scan statistic (MNSS) [1, 10, 11]. The goal of MNSS is to identify a sub-region of the study area that has a "different" mixture of different types of points compared to the outside of the sub-region (i.e., different proportions of different types). As explicitly noted by the authors of MNSS, the method is "direction-less", meaning it can only tell the mixture is different; it cannot indicate any property of a mixture or put a

preference on any mixture. Thus, the method is not suitable when outputs need to be directional (e.g., high or low mixture). A similar formulation of MNSS, the ordinal scan statistic [10], is designed for the case when there is an order among point types or classes (e.g., graduate > undergraduate > high school). It also cannot be used to find high or low mixture patterns. In addition, these methods focus on finding a sub-region that is different from its outside. Thus, the score of a candidate is measured by comparing its mixture to its outside mixture. This only works when the data consists of very few mixture patterns (e.g., one for inside-the-pattern and one for outside), and is not suitable when data contains many different mixture patterns (i.e., every sub-region can be potentially different from every other). Another line of work, from biodiversity studies, mainly focuses on evaluating the biodiversity of a study area and its change across time. Its relevance to spatial mixture pattern detection is that the species richness indices [5, 7, 9] used in these studies can be potentially useful as a score function for ranking candidate regions during detection. However, while the species richness indices (e.g., Simpson's index, entropy) are "directional" and can be used to favor high- or low-mixtures, they are very sensitive to natural randomness in the process and often lead to non-interesting outputs (concrete examples in Sec. 3.1). In addition, some measures rely only on the cardinality of distinct species [8] and cannot tell whether the distribution is balanced or biased.

To bridge the gaps, we first formulate a new Spatial Mixture Index (SMI) to rank and select candidate regions. SMI is directional and can be used with various traditional indices. Then, we present a dual-level Monte-Carlo estimation with a baseline algorithm to compute SMI and identify spatial mixture patterns of interest. Finally, we propose both an exact algorithm and a distribution-inspired sequence reduction heuristic to improve computational efficiency.

Experiment results using both synthetic and real-world data show that the proposed approach with SMI can identify spatial mixture patterns with high accuracy, and that the proposed acceleration techniques can greatly improve computational efficiency while maintaining high solution quality.

2 PROBLEM DEFINITION

2.1 Key Concepts

Distribution of point samples: The input data with geo-located points, each point having one type or class (e.g., species) from a set of size k.

Direction of mixture: Specifies whether a high or low mixture is of interest. A high mixture means the region is less dominated by one or very few types/classes of points, and a low mixture is the opposite.

Test statistic: A function mapping a candidate region to a scalar score representing the degree of mixture (either high or low). The proposed spatial mixture index is a test statistic.

Spatial mixture pattern: A sub-region of the study area that has a significantly high or low mixture of types, measured by the test statistic.

Hypothesis testing: Used to make sure a detected mixture pattern is not formed purely by natural randomness. The null hypothesis H_0 states that the types of points in the input data are randomly assigned (i.e., any high or low mixture region is created by random chance). Significance testing uses the test statistic as a measure but does not contribute to its calculation.

2.2 Formal Problem Formulation

The problem is formally defined as follows:

Inputs:

- A distribution of point samples D where $|D| = N$;
- A direction of mixture (i.e., high or low);
- A significance level α;
- Thresholds for pattern size and count: ρ and r_{max};

Output: Statistically significant spatial mixture patterns;
Objectives: Solution quality and computational efficiency;
Constraints:

- The number of points in any output pattern $\leq \rho N$;
- The maximum number of patterns returned is r_{max}.

The first constraint is used to limit the size of a pattern so that it represents an interesting sub-region rather than the majority of data (commonly $\rho = 1/2$). This can be made flexible by user needs. The second constraint allows users to prioritize the top r_{max} patterns. If not specified, all significant patterns will be returned.

3 SPATIAL MIXTURE PATTERN DETECTION

In this section, we introduce the new and general formulation of the spatial mixture index (SMI), and propose both exact and heuristic algorithms to detect mixture patterns with it.

3.1 Spatial Mixture Index

The spatial mixture index is motivated by the need for the ability to (1) explicitly specify a direction (i.e., high- or low-mixture) for the detection, and (2) explicitly model the effect of natural randomness. As we will show through an illustrative example in Fig. 1, the absence of either of the two will lead to pitfalls in spatial mixture pattern mining.

Fig. 1 shows a distribution of $N = 120$ points of three types, whose cardinalities are [|red|, |blue|, |yellow|] = [78,21,21], respectively. For illustrative purposes, five candidate regions C1 to C5 are shown as circles inside the study area, and the cardinality of points inside each candidate is: C1 = [6,5,5], C2 = [1,1,1], C3 = [1,0,15], C4 = [0,0,16] and C5 = [30,0,0]. In this illustrative example, the goal is to find the region with **high-mixture**. By comparing the candidates, we can see that C1 and C2 have high mixtures and C3 to C5 have low mixtures (i.e., dominated by a single type). C2 is a sub-region of C1. Although it also has a high mixture, the fact that it only has three points makes it statistically less interesting. In other words it is something that can be commonly observed in pure random point distributions. So ideally a measure or test statistic should give the highest score to candidate C1, then C2, and then the rest of the low-mixture candidates.

3.1.1 *Pitfalls of Existing Measures.*

Multi-nomial scan statistic (MNSS) [11]: As we introduced earlier in Sec. 1, MNSS uses a likelihood ratio to measure the interestingness of a mixture as shown in Eq. (1).

$$\log LR = \log \left(\frac{\prod_{i=1}^{k} p_i^{n_i} \cdot q_i^{N_i - n_i}}{\prod_{i=1}^{k} (p_i')^{n_i} \cdot (q_i')^{N_i - n_i}} \right) \tag{1}$$

where n_i is the number of points of type i inside the candidate region where the score is computed and N_i is the total number of points of type i in the study area; $p_i = \frac{n_i}{\sum_{i=1}^{k} n_i}$ is the fraction of points of type i inside the candidate, and $q_i = \frac{N_i - n_i}{\sum_{i=1}^{k}(N_i - n_i)}$ is the fraction of points of type i outside the candidate; and $p'_i = q'_i = \frac{N_i}{\sum_{i=1}^{k} N_i}$ is the fraction of points of type i in the entire study area.

This function is direction-less and its goal is to find a sub-region that has a different mixture compared to its outside. By definition it is based on likelihoods that cannot be used to favor a specific direction of the mixture, and it only cares if there is a difference between inside and outside of a candidate. Also, as we can see through the definition of q_i, the likelihood ratio assumes that the outside of the candidate is generated by a single point process (this assumption is common for scan statistic methods). As a result, it is not suitable for mixture pattern detection in which the data often consists of many different point processes (i.e., a statistical process defining probabilities of a point having type i in a region).

These issues can be seen through the illustrative example in Fig. 1. First, because the measure is direction-less and focuses on the "difference" in the fractions of types between inside and outside, it cannot be used to favor the high-mixture candidates C1 and C2. We can also see that the score of C3 (29.2) is in between C4 (35.6) and C5 (7.5), which further illustrates its direction-less nature because both C4 and C5 are completely homogeneous with only one type of point (i.e., minimal mixture) while C3 has at least two types of points. The fact that C4 and C5 are very different in scores is due to the function's focus on "inside" vs. "outside" instead of degree of mixture, which is the goal of this paper.

Directional mixture measures: In contrast to MNSS, measures popularly used in biodiversity evaluations (e.g., Simpson's index in Eq. (2), Shannon's entropy in Eq. (3)) allow explicit specification of the direction. However, they do not consider the natural randomness commonly exhibited by spatial point distributions, leading to undesired favors (i.e., higher scores) towards statistically non-interesting patterns. To make our discussion concrete, here we will use Simpson's index and Shannon's entropy to illustrate this issue.

We start with Simpson's index (SI) shown in Eq. (2):

$$SI = 1 - \sum_{i=1}^{k} p1_i \cdot p2_i \tag{2}$$

where $p1_i = \frac{n_i}{\sum_{i=1}^{k} n_i}$ and $p2_i = \frac{n_i - 1}{\max(\epsilon^+, (\sum_{i=1}^{k} n_i) - 1)}$ where ϵ^+ is a very small positive number for numerical stability; and n_i is the number of points of type i in the candidate region.

The term $p1_i \cdot p2_i$ is the probability that two random draws of a point (without replacement) in the candidate are both of type i. Thus, Simpson's index here is the probability that two random draws from the candidate are of two different types. [1] The higher the value, the more likely the candidate has a high mixture (i.e., less likely to be dominated by one or very few types).

[1]The original definition of Simpson's index does not have "1-" in front. Here we are using the modified version so a higher value corresponds to a higher mixture

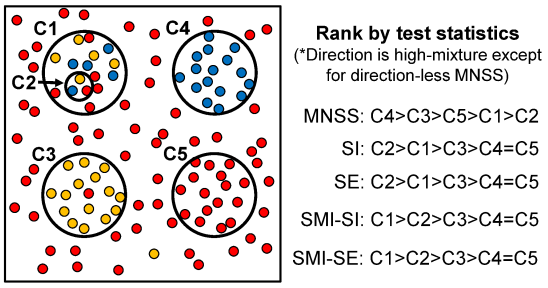


Figure 1: An illustrative example. Scores of candidates are listed in Table 1. (best in color)

Table 1: Scores of candidates in Fig. 1

Candidate	MNSS	SI	SE	SMI-SI	SMI-SE
C1	2.9	0.7	1.095	**1.1**	**1.1**
C2	0.6	**1**	**1.099**	1	1
C3	29.2	0.1	0.2	0.2	0.2
C4	**35.6**	0	0	0	0
C5	7.5	0	0	0	0

**Direction is high-mixture except for MNSS (direction-less).*
***SI: Simpson's index; SE: Shannon's entropy.*

Shannon's entropy (SE) has similar intentions as Simpson's index; it measures the degree of uncertainty in the information by:

$$SE = -\sum_{i=1}^{k} p_i \log p_i \tag{3}$$

where $p_i = \frac{n_i}{\sum_{i=1}^{k} n_i}$.

Table 1 shows the SI and SE scores of the candidates in the example in Fig. 1. As we can see, both measures are able to generally favor high-mixture candidates C1 and C2 over low-mixture candidates C3, C4 and C5 (C3 is also ranked higher than monotone C4 and C5).

However, regarding the two high-mixture candidates C1 and C2, both SI and SE favor the three-point C2 over C1. In fact, for SI, any neighboring points with different types will have the highest score 1. For SE, the score is higher for C2 because C1's type distribution is [6 red,5 blue,5 yellow], and the one extra red-point makes it not a "perfect" balance. Such "imperfection" is highly typical in real-world scenarios due to the effect of natural randomness, especially when the number of points is high. Natural randomness also explains why C2 is less interesting. From a statistical point of view, such tiny candidates (e.g., two-point, three-point) with distinct classes can be easily formed just by a purely random assignment of types among points, so they themselves are not considered as interesting or meaningful. Since these random effects are not considered in Eqs. (2) and (3), they put C2 over C1.

3.1.2 *A New Spatial Mixture Index (SMI).*

SMI has two goals: to allow explicit direction specification (i.e., high or low mixture), and to explicitly model the effect of natural randomness. At a high level, SMI is designed to produce a ratio between the mixture degrees of a candidate from input data and a candidate with the same number of points in random data. The

structure of SMI is shown in Eq. (4), and it can be used with different mixture measures that are directional.

$$SMI = \begin{cases} \frac{f_{dir}(C)}{R(f_{dir}(\cdot),|C|,\beta)}, & \text{if direction = high,} \\ \frac{f_{dir}(C)}{R(f_{dir}(\cdot),|C|,1-\beta)}, & \text{otherwise} \end{cases} \quad (4)$$

where $f_{dir}(\cdot)$ is a directional measure of mixture degree (e.g., traditional measures such as SI in Eq. (2) and SE in Eq. (3)); C is the candidate pattern being evaluated, and $|C|$ is the number of points in C; and function R returns a f_{dir} score of a candidate:

$$R(f_{dir}(\cdot), |C|, \beta) = PMF^{-1}_{f_{dir},|C|}(\{\beta\}) \quad (5)$$

where $PMF_{f_{dir},|C|}(x)$ is a probability mass function,[2] in which the random variable x is the f_{dir} score of a candidate of size $|C|$ in random data (i.e., data with the same spatial distribution of points and overall fraction of each type as the input data, but the type labels on points are randomly assigned); $PMF^{-1}_{f_{dir},|C|}(\{\beta\})$ is a standard math notation referring to the solution x_0 that achieves $PMF_{f_{dir},|C|}(x_0) = \beta$; and $\beta \in (0, 1)$.

In plain language, $R(f_{dir}(\cdot), |C|, \beta)$ is the score achieved by a candidate of size $|C|$ that is greater than $(100\beta)\%$ of scores from candidates in random data. By normalizing the score from a directional function $f_{dir}(\cdot)$ (e.g., SI or SE) with $R(f_{dir}(\cdot), |C|, \beta)$, SMI is able to evaluate whether the candidate is statistically interesting.

In the illustrative example (Fig. 1 and Table 1), we show the results of two concrete realizations of SMI with f_{dir} being SI and SE, respectively. The β is set to 0.9. As we can see, the normalization in SMI effectively suppresses the statistically non-interesting high-values of candidate C2 for both SI and SE (the value "1" means that the mixture achieved by C2 can be found in at least 10% of the same-size candidates in random data), and is able to favor the statistically more meaningful high-mixture candidate C1. More interestingly, although the original ranges of SI and SE are very different as we can see through their scores, the normalized scores by SMI-SI and SMI-SE become much more similar.

Finally, regarding the choice of β, a higher value (e.g., 0.9, 0.99) is recommended because the goal is to suppress high (or low) mixture candidates that also commonly exist due to random chance, and "commonly exist" does not mean "ubiquitous". For example, while the three-point candidate C2 in Fig. 1 can be easily formed by a random assignment of point types (i.e., at least 10% of same-size candidates), this does not mean all or most of the same-size candidates in random data will have the same mixture. In other words, both C1 and C2 may not be the majority, but C2 can be much more easily formed by random chance. Having a high value of β makes it easier to show that difference through normalization. By default, β can be set the same as the significance level.

3.2 A Baseline Algorithm with Dual-Level Monte-Carlo Estimation

Here we will present a baseline algorithm to describe the key computational steps for enumerating, evaluating and selecting candidates as well as significance testing. Due to the unique characteristic of the denominator in SMI, we add a dual-level Monte-Carlo estimation which will be introduced in Sec. 3.2.3.

[2]The cumulative probability that a random variable has a value $\leq x$.

3.2.1 Enumeration of candidates. For candidate enumeration, we adopt the same strategy used in scan statistics (e.g., [1, 10–12]), which exhaustively enumerates sub-regions of a certain geometric shape (e.g., circle, square, ring). Since circles are one of the most widely used shapes in related research and applications, in this version we also use circles as the shape of candidates during enumeration. Specifically, given a set S_T of T centers (e.g., uniformly sampled from the study area), we will enumerate all circles with a point in S_T as the center and a data point on the circumference, leading to $O(TN)$ combinations/candidates.

Since we need to know the composition of point types (i.e., number of points of each type) inside a candidate, a naive brute-force way will require another loop through all the points to see which ones are inside the candidate and what types they are, costing $O(T \cdot N^2)$. This can be avoided simply by sorting all the points by distance from each center in S_T all at once, and then enumerating candidates from the nearest to the farthest from each center [1]. This will sequentially add a point each time to form a new candidate, eliminating the need for an extra range query. As a result, the number of points of each type can be updated in an incremental manner, reducing the cost from $O(T \cdot N^2)$ to $O(T \cdot N \log N)$.

3.2.2 Evaluation of candidates. Next, we need to consider the calculation of the test statistic, i.e., SMI. Basically, given the composition of types of a candidate C, we need to calculate both $f_{dir}(C)$ and $R(f_{dir}(\cdot), |C|, \beta)$. To make our discussion concrete, $f_{dir}(C)$ will be based on SI (Eq. 2) or SE (Eq. 3). Both SI & SE require $O(k)$ to compute, where k is the number of types (a new point added through the sorted sequence mentioned above will incur changes on all p_i). For the denominator $R(f_{dir}(\cdot), |C|, \beta)$, its value will stay the same for all candidates of the same size $|C|$, so with pre-computation it will be $O(1)$. Thus, the total calculation of SMI needs $O(k)$.

So far the total complexity of the baseline is $O(kTN + TN \log N)$. Among all the candidates, each time we will select the one with the maximum score for significance testing. If this candidate is significant, we will remove its associated points from the data and start the next round of detection/testing for a secondary pattern or more. This strategy is also used in MNSS to reduce mutual influences among patterns in both evaluation and significance testing.

Next, we show a dual-level Monte-Carlo estimation for calculating the denominators in SMI as well as significance testing.

3.2.3 Dual-Level Monte-Carlo Estimation. Since there is still no closed-form solution to Eq. (5) (i.e., denominator of SMI), especially considering the additional complexities brought by the spatial distribution of data points as well as the candidate enumeration scheme, we use Monte-Carlo simulation to estimate $R(f_{dir}(\cdot), |C|, \beta)$. For the same reasons, the distribution of SMI scores also needs to be estimated via the Monte-Carlo method to compute the p-value during significance testing.

While both $R(f_{dir}(\cdot), |C|, \beta)$ and p-value require Monte-Carlo simulation, there are fundamental differences in their goals and estimation processes (Table 2).

Candidate-level Monte-Carlo estimation: As shown in Table 2, at this level we are estimating the distribution of scores of all candidates of the same size from random data. In other words, all candidates of the same size from a single simulation trial will be used as members of this distribution. The number of same-size

Table 2: Two levels of Monte-Carlo estimation

Usage	Goal	Distribution to estimate	Level of Monte-Carlo Est.
Eq. (5)	Normalize f_{dir} of a size-$\|C\|$ candidate in real data by statistical interestingness	Distribution of f_{dir} scores of all candidates of the same size $\|C\|$ in random data	**Candidate-level** enumeration of candidates: one score per candidate
p-value	Make sure the method falsely rejects H_0 (outputs a pattern) in only αM out of M random data (e.g., $\alpha = 0.01$)	Distribution of M maximum SMI scores (regardless of size) achieved in M random data	**Data-level** enumeration of M random data: one max-score per data

candidates from a single trial is T, which is equal to the number of circle centers for enumeration (Sec. 3.2.1). As a result, we only need a small number of simulation trials to get a large number of candidate scores to compute Eq. (5) for a normalization purpose. This, as we will show next, is different for p-value estimation.

Data-level Monte-Carlo estimation is typically used in significance testing [4, 11–13, 16]. As shown in Table 2, the goal of this estimation needs to make sure that only αM out of M random data will cause the method to falsely reject H_0 and returns a pattern, where α is the significance level. Thus, to guarantee this, we have to do this enumeration at a data-level, and only get the maximum score achieved in each random data. Then, if the maximum score achieved by a dataset is in the top $(100\alpha)\%$ of this data-level maximum distribution, we are confident to say that the probability to falsely reject H_0 for this data is α.

For candidate-level Monte-Carlo estimation, we basically enumerate all candidates for each size (bounded by N) in m random datasets (m can be small, e.g., 5) to form the distribution and select the $\lceil 100\beta \rceil^{th}$ percentile (this needs another sorting) of the scores for each size as the value for Eq. (5). This leads to $O(m \cdot (kTN + TN \log N) + N \cdot mT \log(mT))$ complexity. Data-level Monte-Carlo estimation also requires a full enumeration for each random dataset, but the number M of random data is typically large (e.g., 1,000) because only one maximum score is selected per data. This leads to $O(M \cdot (kTN + TN \log N) + M \log M)$ complexity. Computation-wise, data-level Monte-Carlo estimation dominates its candidate-level sibling due to the big difference in m and M.

Note that candidate-level Monte-Carlo estimation has to happen before data-level estimation because the output values are necessary to compute the actual SMI scores. Thus, candidate-level estimations are computed at the very beginning of the program.

Also since in data randomization we only randomly shuffle point types but do not change the spatial distribution of points, the previous sorting done during enumeration in real data can be re-used. This fixed distribution is also used in MNSS [1, 10, 11] for data-level Monte-Carlo estimation (it does not need the dual-level). While we can also re-distribute the locations, that is typically less needed in real-world applications. For example, locations of trees, resident houses, buildings for businesses (e.g., grocery) are relatively stationary.

3.2.4 Time Complexity. With sorting re-use, the complexities become $O(mkTN + N \cdot mT \log(mT))$ and $O(MkTN + M \log M)$ for candidate- and data-level estimation. The overall complexity is then $O(mkTN + N \cdot mT \log(mT) + MkTN + M \log M + kTN + TN \log N)$, where m and M are number of trials in candidate- and data-level Monte-Carlo estimation, k is number of types, T is number of centers to enumerate and N is the number of data points. Since in the vast majority of cases we have $m << M$, $\log N < M$, $\log(mT) < M$, and $\neg(m >> k)$, the complexity can be simplified to $O(M \cdot kTN)$, revealing that the cost is dominated by data-level Monte-Carlo estimation.

3.3 Acceleration by an Exact Algorithm for SMI computation

To accelerate the computation for the data-level Monte-Carlo simulation, we first propose an exact algorithm to minimize the computational cost on SMI. As an exact algorithm, it guarantees the solution is exactly the same as the baseline algorithm, while reducing the calculation of SMI from $O(k)$ to $O(1)$. In the following we show the new calculation for both SMI-SI and SMI-SE.

3.3.1 SMI for Simpson's Index. According to Eq. (4), SMI-SI$_0$ for the current candicate C_0 can be written as follows (for simplicity the original directional condition is taken out by using x to represent either β or $(1-\beta)$):

$$\frac{1-\sum_{i=1}^{k}(p1_i \cdot p2_i)}{R(SI, \sum_{i=1}^{k} n_i, x)} = \frac{1-\sum_{i=1}^{k}\left(\frac{n_i}{\sum_{i=1}^{k} n_i} \cdot \frac{n_i-1}{\max(\epsilon^+, (\sum_{i=1}^{k} n_i)-1)}\right)}{R(SI, \sum_{i=1}^{k} n_i, x)}$$
$$= \left(1 - \frac{\sum_{i=1}^{k}(n_i^2 - n_i)}{\max(\epsilon^+, (\sum_{i=1}^{k} n_i)^2 - \sum_{i=1}^{k} n_i)}\right) / R(SI, \sum_{i=1}^{k} n_i, x)$$

where n_i is the number of points of type i in the **current** candidate C_0, and ϵ^+ is a very small positive number for numerical stability.

Now suppose we move to the next point (i.e., next candidate) in the sorted sequence (Sec. 3.2.1), and its type ID is j. We have the new SMI-SI$_1$ for candidate C_1 as:

$$\left(1 - \frac{\left(\sum_{i=1, i\neq j}^{k}(n_i^2 - n_i)\right) + (n_j+1)^2 - (n_j+1)}{(1+\sum_{i=1}^{k} n_i)^2 - (1+\sum_{i=1}^{k} n_i)}\right) / R(SI, \sum_{i=1}^{k} n_i + 1, x)$$

where n_i or n_j is the number of points of type i or j in candidate C_0 from the **previous** step.

Denote $\Theta_0 = \sum_{i=1}^{k}(n_i^2 - n_i)$, which is a part of SMI-SI$_0$. We have $\Theta_1 = \Theta_0 + 2n_j$ for the corresponding part of SMI-SI$_1$, i.e.:

$$\text{SMI-SI}_1 = \left(1 - \frac{\Theta_1}{(1+\sum_{i=1}^{k} n_i)^2 - (1+\sum_{i=1}^{k} n_i)}\right) / R(SI, 1 + \sum_{i=1}^{k} n_i, x)$$

Since the R function in Eq. (5) is pre-computed at the beginning in the baseline algorithm, we can get both $R(SI, \sum_{i=1}^{k} n_i, x)$ and $R(SI, 1 + \sum_{i=1}^{k} n_i, x)$ in $O(1)$ time. In addition, the baseline also already has the values of n_i and $\sum_{i=1}^{k} n_i$ updated in an incremental way, so we have their values in $O(1)$ time. Thus, by only keeping track of Θ and performing a constant-time update at each step (i.e.,

$\Theta = \Theta_{prev} + 2n_j$, where j is the type of the new point), we can calculate SMI-SI scores in $O(1)$ time, leading to a reduced overall time complexity of $O(MTN)$.

3.3.2 SMI for Shannon's Entropy. Following the same n_i and n_j definitions (i.e., from C_0), here we directly provide the update rule and simplified version for SMI-SE$_1$. Denote $\Theta_0 = -\sum_{i=1}^{k} n_i \log n_i$, we have $\Theta_1 = \Theta_0 + n_j \log \frac{n_j}{n_j+1} - \log(n_j + 1)$. Then:

$$\text{SMI-SE}_1 = \left(\frac{\Theta_1}{(1 + \sum_{i=1}^{k} n_i)} + \log(1 + \sum_{i=1}^{k} n_i) \right) / R(SE, 1 + \sum_{i=1}^{k} n_i, x)$$

Note that to get the above form, it is helpful to first simplify SE to $\frac{-\sum_{i=1}^{k}(n_i \log n_i)}{\sum_{i=1}^{k} n_i} + \log \sum_{i=1}^{k} n_i$.

3.4 Acceleration by a Distribution-Inspired Sequence Reduction Heuristic

In this section we will be a little more aggressive and further reduce the computation by proposing a distribution-inspired sequence reduction heuristic. Due to its heuristic nature, this algorithm will not guarantee that it can always find the optimal candidate (i.e., one with highest SMI for high-mixture or lowest for low-mixture), but will try to reach it by searching within subspaces that are more likely to contain it. In our experiments, we will show the effectiveness of our heuristic strategy with empirical evaluation.

3.4.1 A Sequence Optimization View. To better illustrate the ideas and subspaces enumerated by the heuristic algorithm, we first introduce a sequence-optimization view of the enumeration process on a dataset. In this view each sequence is an array of N points sorted based on their distances to a single candidate center, and each member in a sequence represents a candidate containing all the points up to it in the sequence. The full enumeration space then contains T sequences where T is the number of centers, and the goal is to find the candidate that maximizes or minimizes the SMI score depending on the input direction.

Since the sorting of all points to all centers only needs to be done once at the beginning (Sec. 3.2.1) and SMI can be computed in $O(1)$, going through this full space in each dataset only requires $O(TN)$ steps. It is important to note that there exists a "smaller-first-larger-later" (SFLL) constraint for the enumeration: the points in each sequence must be enumerated one-by-one strictly from the first (nearest) point to the last (farthest), because the values needed for $O(1)$ SMI calculation rely on those from the previous point. In other words, candidates with smaller number of points must be evaluated before larger ones.

3.4.2 Distribution-Inspired Sequence Reduction. Given the SFLL constraint, the sequence reduction heuristic starts from the smallest candidates and tries to dynamically narrow down the enumeration space of larger candidates as the search propagates.

The heuristic criterion we use for narrowing down the search space is inspired by the characteristics in the distribution of type-composition (proportions of each type) in candidates of difference sizes. Specifically, we observe that the distribution of type-composition "squeezes" as the candidate size increases.

Denote candidate $set_C = \{C_1, C_2, ...C_T\}$ as a set of T candidates of the same size where C_j is from the j^{th} sequence; $\mathbf{U} \in \mathbb{R}^{T \times k}$ as

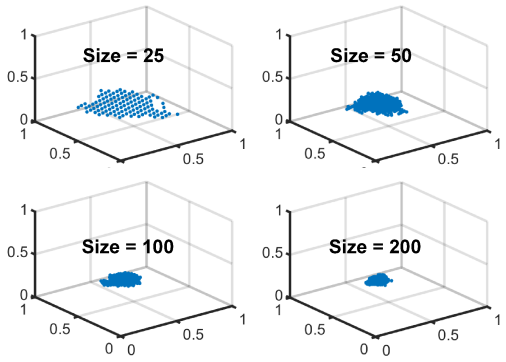


Figure 2: Visualizing of the "squeezing" effect.

a matrix containing type-compositions of the T candidates where $\mathbf{U}_{jh} = \frac{C_j.n_h}{\sum_{i=1}^{k} C_j.n_i}$, $\forall j = 1...T, h = 1...k$. Considering each row $\mathbf{U}_j$ as a k dimensional point, first we can see the T points in $\mathbf{U}$ are actually always distributed in a hyperplane of dimension $(k-1)$ as $\sum_{i=1}^{k} \mathbf{U}_{ji} = 1$, $\forall j = 1...T$. For example, when $k = 2$, the points are on a 1D line described by $X + Y = 1$, and when $k = 3$ they are on a 2D plane described by $X + Y + Z = 1$.

The "squeezing" distribution we observe refers to that the range covered by most of the points on the hyperplane has a tendency to become narrower as the number of points in the candidate increases. The intuition of this is that as the number of points increases, the type-composition of a candidate tends to become more stable with less variation (i.e., less susceptible to random effects). Many of the sequences will start to converge to the overall type-composition of the whole dataset (e.g., with N points, the composition is always equivalent to the overall composition). Fig. 2 empirically visualizes this "squeezing" effect for a 2000-point distribution with $k = 3$, and number of points per type being [500, 600, 900].

As we can see, the range of the distributions starts to narrow towards the overall composition as the size of the candidate increases along the sequences.

This leads to the idea of the sequence reduction heuristic, which starts by enumerating candidates along all sequences to cover the wider range, and gradually reduces the number of sequences to enumerate (i.e., a smaller search space) as the size increases.

Denote λ as the number of steps to take in sequence reduction, N as the length of the sequence (max number of points), and γ as the proportion of sequences to keep after the reduction in each step. At the first step, we enumerate the first $\lceil N/\lambda \rceil$ sizes of all sequences. In each sequence, we will additionally keep track of the best SMI so far from it. After the enumeration, we sort the sequences by their contained best SMIs, and only keep the sequences with a best SMI in the top $(100\gamma)\%$ for the next round of enumeration. We repeat the same procedure in all the following steps as shown in Alg. 1.

3.4.3 Time Complexity. The time complexity of sequence reduction on each dataset is:

$$O\left(\left(\sum_{i=1}^{\lambda} \lceil \frac{N}{\lambda} \rceil \cdot T \cdot \gamma^{i-1} \right) + \sum_{i=1}^{\lambda-1} T \cdot \gamma^{i-1} \log(T \cdot \gamma^{i-1}) \right)$$

$$= O\left(\lceil \frac{N}{\lambda} \rceil \cdot T \cdot \frac{1-\gamma^{\lambda}}{1-\gamma} + \sum_{i=1}^{\lambda-1} T \cdot \gamma^{i-1} \log(T \cdot \gamma^{i-1}) \right)$$

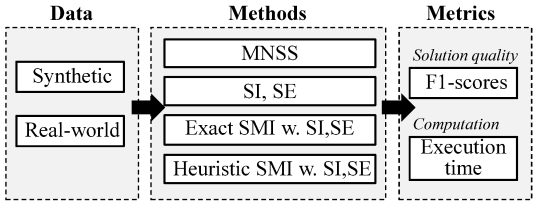


Figure 3: Overall validation framework.

Assuming $\log T < \lceil N/\lambda \rceil$, the complexity can be simplified to $O(\lceil \frac{N}{\lambda} \rceil \cdot T \cdot \frac{1-\gamma^{\lambda}}{1-\gamma})$. With $\lambda = 10$ and $\gamma = 0.5$, the number of candidates enumerated will reduce from TN to $(0.1998 \cdot TN)$.

Finally, to ensure consistency between enumeration algorithms in the detection phase and significance testing phase, the sequence reduction heuristic is recommended to be used either for both or for none especially if strict statistical robustness is desired.

Table 3 summarizes the time complexity of the baseline, exact, and heuristic algorithms. As noted in Sec. 3.2.2, the algorithm only outputs the best candidate in each round (if it is significant; otherwise, it terminates). After that, the pattern is removed from data before the next round of detection/testing. For clarity, the complexity in Table 3 is for a single round of detection in this process.

Algorithm 1: Sequence Reduction Heuristic

Require:
- Set_T: T sequences of length N
- Number of steps λ
- Reduction parameter γ

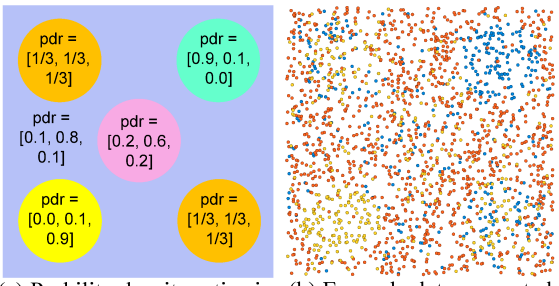
C_{best} = init()
SMI_{track} = init($|Set_T|$)
SMI_{opt} = init($|Set_T|$)
for i = 1 to λ **do**
  $base = (i-1) \cdot \lceil N/\lambda \rceil$
  for j=($base$+i) to min($base$+$\lceil N/\lambda \rceil$, N) **do**
    for seq in Set_T **do**
      $[seq^*, seq_{track}]$ = getOptimum(seq)
      $SMI_{opt}(seq)$ = best($SMI_{opt}(seq)$, seq^*)
      C_{best} = best(C_{best}, seq^*)
      UpdateSMITrack(SMI_{track}, seq_{track})
    end for
  end for
  Set_T = selectTopSeqsForNextStep(Set_T, SMI_{opt}, γ)
end for

4 VALIDATION

We evaluate the solution quality and computational performance of the proposed approaches via both synthetic and real-world data. Fig. 3 shows the overall validation framework.

Table 3: Summary of time complexity

	Baseline	Exact	Heuristic
Complexity	$O(MkTN)$	$O(MTN)$	$O(M\lceil \frac{N}{\lambda} \rceil \cdot T \cdot \frac{1-\gamma^{\lambda}}{1-\gamma})$



(a) Probility density ratios in each sub-region (b) Example data generated from (a)

Figure 4: Statistical process for synthetic data.

4.1 Solution Quality

4.1.1 Synthetic Data Description. Fig. 4(a) shows an example of the statistical process that we used to generate the synthetic data with three types of points. Within each sub-region (i.e., distinct by color), the probability-density-ratio (*pdr*) vector contains the ratio of the probability density of each type to the sum of the probability densities of all types. The *pdr* in each sub-region is homogeneous. For example, in the orange regions, the probability density ratios of the three types are all 1/3, implying that this sub-region is a high-mixture region (i.e., the composition is not dominated by any type). Fig. 4(b) shows an example 2000-point distribution generated from the process in Fig. 4(a).

The goal of the experiments with synthetic data is to see if the methods can identify the two orange high-mixture regions.

4.1.2 A Visual Comparison. Fig. 5 shows a qualitative visual comparison of different methods' outputs on the example data in Fig. 4(b). To correctly show the output of the multi-nomial scan statistic (MNSS) we split the results into two sub-figures Fig. 5(a) and (b). As introduced in Sec. 3.2.2, a significant pattern needs to be removed from data before detecting the next one. Thus, Pattern 3 and 4 need to be visualized on a separate sub-figure to show the actual points they contain. As we can see, MNSS's direction-less nature and its focus on "comparing inside and outside compositions" make its results not suitable for application scenarios where (1) a direction is needed and (2) data consists of many different mixtures (i.e., "inside vs. outside" type of strict bi-partition is less meaningful).

Fig. 5(c) shows that both Simpson's index and Shannon's entropy, when used as the test statistic, cannot detect any significant pattern. This is due to their lack of robustness under the effects of natural randomness. As discussed in Sec. 3.1 (Table 1), these measures tend to give the highest scores to small patterns with high-mixture. Since such small patterns are likely to be formed by random chance, they cannot pass the significant testing, leading to empty outputs.

By contrast, both SMI-SI and SMI-SE are able to identify the two high-mixture patterns. Since the SMI-based results are nearly identical, we only show one of them in Fig. 5.

4.1.3 Quantitative Evaluation. Here we generate hundreds of datasets using processes based on Fig. 4(a) to compute F1-scores (harmonic mean of precision and recall). The goal is still to detect the two high-mixture patterns.

The parameters we vary are: (1) total number of points N, (2) the area A of the two target high-mixture regions (i.e., orange colored),

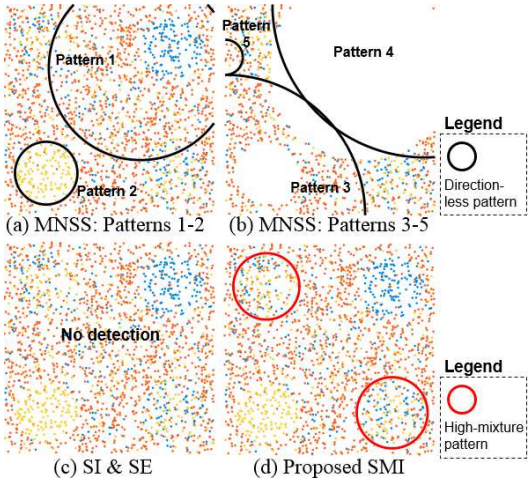


Figure 5: Qualitative visual comparison of outputs.

(3) the ratio R between number of points in the foreground (circular regions) and background (rest), and (4) an algorithm-related parameter T representing the total number of candidate centers.

The study area has dimension 100×100, the default area for all circles is $\pi 15^2$. By varying the area of the two target high-mixture patterns, we can evaluate how sensitive are the methods to the **relative sizes of the true patterns**. The ratio R has a different purpose. By default R is set to the ratio between the total area of the five foreground circles and the area of the background blue region (i.e., about 3500/6500), which means the density of points is roughly homogeneous across the study area. By varying R, we can evaluate the effect of **heterogeneity in point-density** on solution quality. When we vary one parameter, the others are kept at the default values $[N, A, R, T] = [5,000, \pi 15^2, 35/65, 400]$.

Since MNSS, SI and SE are not able achieve a reasonable solution quality for this task as shown in Fig. 5, we skip their F1-scores which are very low (e.g., near 0) and not as interesting. The candidates for this evaluation then are: (1) the exact version; (2) heuristic with number of steps $\lambda = 10$ and reduction parameter $\gamma = 0.75$; (3) heuristic with $\lambda = 10$ and $\gamma = 0.5$; (4) heuristic with $\lambda = 20$ and $\gamma = 0.75$; and (5) heuristic with $\lambda = 20$ and $\gamma = 0.5$. Heuristic methods in (2) to (5) reduces the enumeration space to about 37.75%, 19.98%, 19.94% and 10.00% of the original size, respectively, using different combinations of λ and γ.

The F1 scores of the methods are shown in Tables 4 to 7. Each F1 score is computed using results from 25 datasets generated using the statistical process with the corresponding parameters. Overall, the trend is that the SMI-based methods consistently achieve high F1 scores throughout most of the experiments.

SMI-SI vs. SMI-SE: According to the F1 scores, the solution quality achieved by both methods are very similar in the vast majority of cases. The two minor differences are seen in Table 5 when $A = 10^2\pi$ and Table 6 when $R = 30/70$. In these cases SMI-SE outperforms SMI-SI with a small margin $< 5\%$. Both of these cases correspond to the scenario when the number of points in the target patterns is relatively smaller, either due to a smaller area (i.e., $A = 10^2\pi$ for true patterns while the other three circles have $15^2\pi$) or lower density (i.e., point-density in circles are lower than that of the background blue region). In such a scenario, a true pattern becomes harder to detect since population-wise its signal is relatively weaker compared to other mixtures. According to the results, SMI-SE is a little more robust than SMI-SI in this case, which motivates future investigation of other participating functions in SMI.

Table 4: Solution Quality by F1 Scores: Varying N

	N	Exact	Heuristic: (λ, γ)			
			(10,0.75)	(10,0.5)	(20,0.75)	(20,0.5)
SMI-SI	2500	0.97	0.99	0.99	0.98	0.99
	5000	0.99	0.99	0.97	0.99	0.98
	10000	0.93	0.94	0.95	0.95	0.95
SMI-SE	2500	1.00	1.00	1.00	1.00	1.00
	5000	0.99	1.00	1.00	1.00	1.00
	10000	0.96	0.94	0.96	0.95	0.95

Table 5: Solution Quality by F1 Scores: Varying A

	A	Exact	Heuristic: (λ, γ)			
			(10,0.75)	(10,0.5)	(20,0.75)	(20,0.5)
SMI-SI	$10^2\pi$	0.97	0.94	0.96	0.97	0.95
	$15^2\pi$	0.99	0.99	1.00	1.00	1.00
	$20^2\pi$	0.99	0.99	0.98	0.99	0.99
SMI-SE	$10^2\pi$	0.99	0.99	0.98	0.99	1.00
	$15^2\pi$	0.99	0.99	0.99	0.99	0.99
	$20^2\pi$	1.00	1.00	1.00	0.99	1.00

Table 6: Solution Quality by F1 Scores: Varying R

	R	Exact	Heuristic: (λ, γ)			
			(10,0.75)	(10,0.5)	(20,0.75)	(20,0.5)
SMI-SI	30/70	0.90	0.92	0.93	0.93	0.94
	35/65	0.98	0.99	0.99	0.99	1.00
	45/55	1.00	1.00	1.00	1.00	1.00
SMI-SE	30/70	0.96	0.96	0.98	0.97	0.95
	35/65	1.00	0.99	1.00	0.98	0.99
	45/55	1.00	1.00	1.00	0.99	1.00

Table 7: Solution Quality by F1 Scores: Varying T

	T	Exact	Heuristic: (λ, γ)			
			(10,0.75)	(10,0.5)	(20,0.75)	(20,0.5)
SMI-SI	100	0.95	1.00	0.97	0.96	0.96
	400	0.99	0.97	0.98	0.98	1.00
	2500	0.98	1.00	1.00	0.99	0.97
SMI-SE	100	0.99	0.99	1.00	0.99	1.00
	400	1.00	0.99	0.99	0.99	1.00
	2500	1.00	0.99	0.99	0.99	0.99

Exact vs. heuristic: Since the sequence reduction algorithm in Sec. 3.4 is a heuristic algorithm, it does not guarantee that the solution is always the same as that of the exact algorithm. Thus, to better understand its performance, we empirically compared its solution quality with the exact algorithm. According to the results in Tables 4 to 7, we can see that the distribution-inspired sequence reduction heuristic consistently achieves very similar F1 scores to the exact algorithm's under different combinations of λ and γ throughout the experiment.

Effect of parameters: The solution quality of the methods are relatively stable in the experiments with varying parameters and variations are mostly within 5%. The effects of changes in relative pattern sizes caused by A and R were discussed earlier in the SMI-SI vs. SMI-SE comparison. We can also see that the F1 scores of SMI-SI

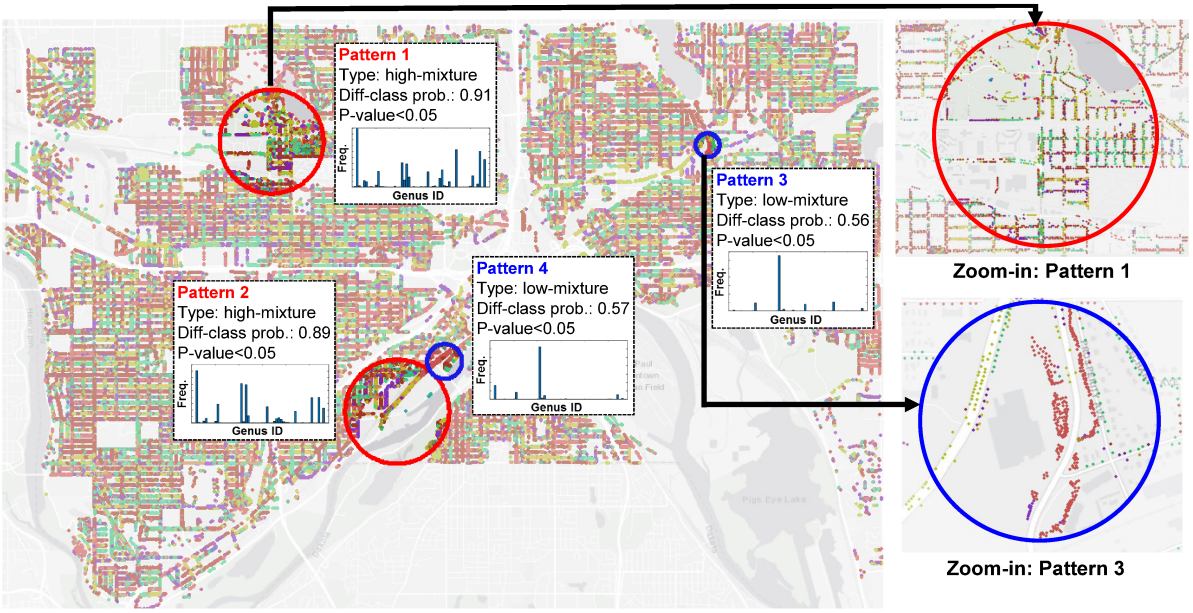


Figure 6: Case study on smart city tree management in St Paul, MN.

increases as T increases, which is expected since larger numbers of centers increases the number of candidates overlapping with a true pattern, making the detection easier. Note that the results on T here are limited as they mainly show the effects of randomness (true pattern centers are fixed and covered by all choices of T).

4.1.4 Case Study: Resilience of Smart Cities. We also conducted a real-world case study using data provided by our city partner in St Paul, Minnesota. The data is an ongoing multi-year effort, and contains the locations of 123, 104 public trees managed by the city along road-sides. The number of tree genera is 57, including Acer, Fraxinus (i.e., ash), Pine, etc. The 57 genera will be used as types of points. The goal of this case study is to use the proposed approach to identify high-mixture and low-mixture regions within the city, which is an important problem across many urban areas (Sec. 1). Note that since trees along a road segment (e.g., one side of a block) often have the same genus, city planners are more interested in relatively larger zones which can reveal more meaningful information about the mixtures. Thus, in the case study the minimum size of a pattern is constrained to be at least 500.

Fig. 6 shows detections of both high-mixture and low-mixture patterns, colored in red and blue, respectively. Each point in the background represents an individual tree (easier to see in the zoom-in windows). To maintain visual clarity, we show only the two most prominent patterns (i.e., highest or lowest SMI scores) for both high- and low-mixture. The results for SMI-SI and SMI-SE are very similar with heavy overlaps, so, similar to Fig. 5, we will use SMI-SI's results to represent both.

Along each detected pattern in Fig. 6, we added a bar-chart to visualize the type-composition of the pattern. In the bar-chart, the X-axis is type-ID (57 in total) and Y-axis is frequency. We additionally show the probability of having two draws of trees from the pattern returning different types (i.e., Simpson's index). All the patterns are statistically significant at the level of 0.05. As we can see, the type distributions of the two high-mixture patterns are much more balanced (i.e., more tall bars) compared to the distributions of the low-mixture patterns, which are dominated mostly by a single tree genus. This can also be seen from the above-mentioned probability values, which are 0.91 and 0.89 for the two high-mixture patterns, respectively, and 0.56 and 0.57 for the two low-mixture patterns, respectively. Finally, we show two zoom-in windows of Pattern-1 and Pattern-3 to help see more details. In the zoom-in window of Pattern-1, trees of more colors (i.e., types) can be observed whereas in Pattern-3, most of the trees are of the same type. Urban regions similar to Pattern-3 are typically very vulnerable when the dominant tree type within them is targeted by a disease and often need prescriptive interventions.

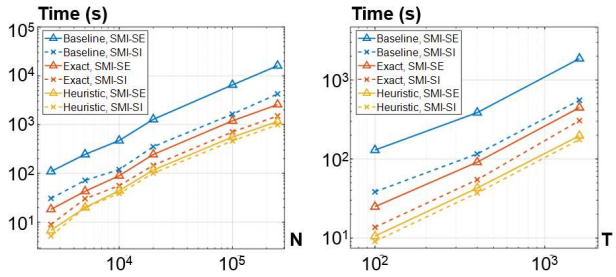
We skip the results of MNSS as it finds two huge patterns (one covering the west and the other covering a large part of the east of the study area) that are neither high or low mixture patterns but more of differences between the compositions inside and outside of a sub-region. SI and SE did not return any significant pattern.

4.2 Computational Performance

We evaluate the computational performance of the baseline algorithm as well as the proposed accelerations using the exact and sequence reduction heuristic algorithms. The data are generated using the process in Fig. 4(a) where the number of types is 3. The parameters we vary are the total number of points (i.e., data size) N, the number of candidate centers T, as well as λ and γ values for the heuristic algorithm. The default values are $[N, T] = [10000, 400]$, and $[\lambda, \gamma] = [10, 0.7]$.

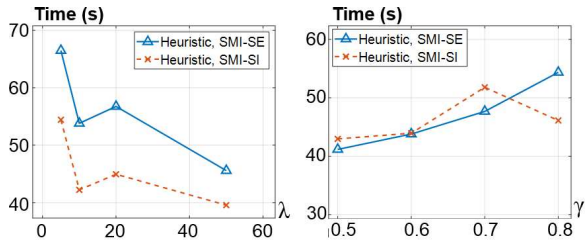
Fig. 7 shows the execution time for three algorithms with both SMI-SI and SMI-SE with varying N and T, and Fig. 8 shows the execution time of the heuristic algorithm with varying λ and γ.

Execution time comparison: As we can see in Fig. 7(a) and (b), the exact and heuristic versions of acceleration provide significant speed-ups to the baseline algorithms for both SMI-SI and SMI-SE. The speed-ups are relatively stable for different N from 2,500 to 250,000 (both X and Y axes are shown in log-scale). For example,



(a) Varying data size N (b) Varying #centers T

Figure 7: Execution time with varying N and T.



(a) Varying λ (b) Varying γ

Figure 8: Execution time with varying λ and γ.

for SMI-SI, the speed-up by the exact algorithm is about 3.4x when $N = 2,500$ (i.e., from 30.7s to 9.0s) and 2.8x when $N = 250,000$ (i.e., from 4257.8s to 1502.1s). This is roughly proportional to the number of types $k = 3$ as indicated by the complexity comparison in Table 3. For the heuristic algorithm, it will reduce the enumeration space to a constant proportion of the full space. Thus, its speed-up is also relatively stable (e.g., an additional 3x) in the experiments. Finally, we can see that the time cost of SMI-SE tends to be higher than SMI-SI. This is likely caused by the longer time for computing SE as compared to SI, which is a constant factor typically made implicit in the asymptotic complexity. We can also see that the differences between the execution time of SMI-SE and SMI-SE decreases after acceleration. This is potentially due to the fact that as the time-costs of the dominant terms in time complexity reduces, other terms (e.g., sorting) may start playing a relatively bigger role and dilute the differences in dominant terms.

Effects of parameters: First, in Fig. 7(a) and (b) we can see the execution time increases linearly as N and T increase, which is consistent with the complexities in Table 3. Then, results in Fig. 8(a) show that in general the execution time of the heuristic algorithm decreases as the number of steps λ increases ($\gamma = 0.7$). This is because a greater λ leads to more frequent reductions, reducing the total number of candidates being enumerated. Note that if γ is set very close to 1 (not recommended), we may see less of this trend because the reduction may be consumed by a higher cost of sorting. Finally, Fig. 8(b) shows that execution time increases as the reduction parameter γ increases (i.e., less aggressive reduction), which conforms to the complexity in Table 3.

5 CONCLUSIONS AND FUTURE WORK

We proposed a Spatial Mixture Index (SMI) to identify spatial mixture patterns of interest, i.e., sub-regions with significantly high or low mixture of different types of points. Then, we presented a baseline algorithm with dual-level Monte-Carlo estimation to compute SMI and detect patterns. We further proposed two acceleration schemes with an exact algorithm as well as a distribution-inspired sequence reduction heuristic to improve the computational performance by reducing time complexity. Experiment results using both synthetic and real-world data validated the solution quality of the proposed approach and also showed that the acceleration techniques can greatly reduce the execution time while maintaining high quality of results. For reproducibility, code is available at: https://github.com/yqthanks.

In future work, we will explore new opportunities opened by this new pattern. A short-term plan will explore alternative participating functions f_{dir} of SMI (e.g., Alpha, Beta or Gamma diversity measures) or other extensions for specific application needs. New computational strategies (e.g., approximation or distributed algorithms [6, 14]) will also be investigated to improve scalability. We will also explore other formulations of candidate regions (e.g., irregular shapes [4, 16]) and statistical processes (e.g., different types of point distributions). Finally, the current work does not explicitly model scenarios when the mixture inside a pattern is highly heterogeneous, and those cases need further investigation.

ACKNOWLEDGMENTS

This work is supported by the NSF under Grants No. 1029711 and 1737633, the USDA under Grant No. 2017-51181-27222, the Minnesota Supercomputing Institute, and the University of Maryland.

REFERENCES

[1] 2017. SaTScan. https://www.satscan.org/.
[2] 2018. Emerald Ash Borer. https://www.nrs.fs.fed.us/disturbance/invasive_species/eab/effects_impacts/cost_of_infestation/.
[3] 2019. Minnesota cities struggle to stay ahead of emerald ash borer's rapid spread. https://www.startribune.com/cities-struggle-to-stay-ahead-of-emerald-ash-borer-s-rapid-spread/563454422/?refresh=true.
[4] Renato Assuncao, M Costa, A Tavares, and S Ferreira. 2006. Fast detection of arbitrarily shaped disease clusters. *Statistics in medicine* 25, 5 (2006), 723–742.
[5] Benjamin Bandeira, Jean-Louis Jamet, et al. 2013. Mathematical convergences of biodiversity indices. *Ecological Indicators* 29 (2013), 522–528.
[6] Dong-Wan Choi et al. 2012. A scalable algorithm for maximizing range sum in spatial databases. *Proc. of the VLDB Endowment* 5, 11 (2012), 1088–1099.
[7] Alan Feest, Timothy D Aldred, and Katrin Jedamzik. 2010. Biodiversity quality: a paradigm for biodiversity. *Ecological Indicators* 10, 6 (2010), 1077–1082.
[8] Kaiyu Feng, Gao Cong, Sourav S Bhowmick, Wen-Chih Peng, and Chunyan Miao. 2016. Towards best region search for data exploration. In *Proc. of the 2016 International Conference on Management of Data*. 1055–1070.
[9] Fangliang He and Xin-Sheng Hu. 2005. Hubbell's fundamental biodiversity parameter and the Simpson diversity index. *Ecology Letters* 8, 4 (2005), 386–390.
[10] Inkyung Jung, Martin Kulldorff, and Ann C Klassen. 2007. A spatial scan statistic for ordinal data. *Statistics in medicine* 26, 7 (2007), 1594–1607.
[11] Inkyung Jung, Martin Kulldorff, and Otukei John Richard. 2010. A spatial scan statistic for multinomial data. *Statistics in medicine* 29, 18 (2010), 1910–1918.
[12] Martin Kulldorff. 1997. A spatial scan statistic. *Comm. in Statistics-Theory and methods* 26, 6 (1997), 1481–1496.
[13] Daniel B Neill et al. 2004. Rapid detection of significant spatial clusters. In *ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*. 256–265.
[14] Sushil K Prasad et al. 2017. Parallel processing over spatial-temporal datasets from geo, bio, climate and social science communities: A research roadmap. In *2017 IEEE International Congress on Big Data (BigData Congress)*. 232–250.
[15] Yiqun Xie, Han Bao, Shashi Shekhar, and Joseph Knight. 2018. A TIMBER framework for mining urban tree inventories using remote sensing datasets. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 1344–1349.
[16] Yiqun Xie and Shashi Shekhar. 2019. Significant DBSCAN towards Statistically Robust Clustering. In *Proceedings of the 16th International Symposium on Spatial and Temporal Databases*. 31–40.
[17] Yiqun Xie, Shashi Shekhar, Richard Feiock, and Joseph Knight. 2019. Revolutionizing tree management via intelligent spatial techniques. In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 71–74.