ACCELERATING THE DETECTION OF SPACE-TIME PATTERNS UNDER NON-STATIONARY BACKGROUND POPULATION

by

Alexander Hohl

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Geography and Urban Regional Analysis

Charlotte

2018

Approved by:

_____
Dr. Eric Delmelle

_____
Dr. Wenwu Tang

_____
Dr. Xun Shi

_____
Dr. Erik Saule

ABSTRACT

ALEXANDER HOHL. Accelerating The Detection Of Space-Time Patterns Under Non-Stationary Background Population
(Under the direction of Dr. Eric Delmelle)

The advancement of technology has enabled us to collect increasing quantities of spatial and spatiotemporal data at rapidly increasing rate through sensor systems, automated geocoding abilities and social media platforms, such as Facebook or Twitter. Processing, analyzing and making sense of big data, which is characterized by high volume, velocity and variety, is challenging and hence, calls for increased computing performance. Exploratory spatial data analysis approaches, such as kernel density estimation, allow us to detect patterns that facilitate the formation of hypotheses about their driving processes. However, it is important to recognize that patterns of disease and other social phenomena emerge from an underlying population, which has to be accounted for in order to extract actual trends from the data. My dissertation research challenges a key assumption of many prominent methods of estimating disease risk, which is that population is static through time. I put forward the method of adaptive kernel density estimation by accounting for spatially and temporally inhomogeneous background populations. In addition, I develop a flexible spatiotemporal domain decomposition approach, which allows for tackling the big data challenge of developing scalable approaches to compute spatiotemporal statistics, using high-performance parallel computing. Last, I propose a framework for sensitivity analysis of spatiotemporal computing, which allows for quantifying the effect of model parameter values on computing performance and scalability. The results of my dissertation contribute to

scalable applications for analyzing social geographic phenomena and elucidate the computational requirements of spatiotemporal statistics.

DEDICATION

To Peilin Chen, my loving wife and friend.

ACKNOWLEDGEMENTS

I would like to thank my committee (Drs. Eric Delmelle, Wenwu Tang, Xun Shi, and Erik Saule) for the support and critique of my dissertation project. Dr. Irene Casas, thank you for providing the data, as well as good spirits and support. In addition, former and current members of the Center for Applied GIScience at UNC Charlotte were very helpful and deserve recognition (Jing Deng, Meijuan Jia, Huifang Zuo, Douglas Shoemaker, Adam Griffith, Michael Desjardins, Minrui Zheng, Claudio Owusu, Yu Lan, Jianxin Yang, Coline Dony, Wenpeng Feng, Jiyang Shi, Michael Howe), as well as students and faculty of the Department of Geography and Earth Sciences at UNC Charlotte (Missy Eppes, Elizabeth Delmelle, Danny Yonto, Ran Tao, Paul Jung, Abel Ayon, Thomas Howarth, Liz Morell). Thanks to my wife Peilin Chen for your endless love and support, as well as proofreading, critique, inspiration and assistance. Lastly, I want to thank my parents Yuhsin Hohl and Niklaus Hohl for being supportive of my academic endeavors.

TABLE OF CONTENTS

LIST OF TABLES x

LIST OF FIGURES xi

CHAPTER 1: INTRODUCTION 1

1.1 Background 1

1.2 Research objectives 7

1.2.1 Objective 1 8

1.2.2 Objective 2 8

1.2.3 Objective 3 8

1.3 Road map 9

CHAPTER 2: LITERATURE REVIEW 10

2.1 Spatial and spatiotemporal statistics 10

2.1.1 Autocorrelation-based approaches 10

2.1.2 Geostatistics 12

2.1.3 Point pattern analysis 12

2.3.3.1 Ripley's K function 13

2.3.3.2 Kernel density estimation 15

2.2 Kernel methods for disease mapping 16

2.2.1 Spatial filters 16

2.2.2 The spatial relative risk function 17

2.3 Parallel strategies for spatial and spatiotemporal statistics 20

2.4 Sensitivity analysis 24

2.4.1 Local approaches 25

2.4.2 Regression approaches 26

2.4.3 Variance-based approaches 27

CHAPTER 3: Methodology 29

3.1 Overview 29

3.2 Research objective 1 methodology 31

3.2.1 Case-side adaptive bandwidth kernel density estimator 31

3.2.1.1 Kernel density estimation 32

3.2.1.2 Kernel density estimation with inhomogeneous background 33

3.2.1.3 Space-time kernel density estimation 36

3.2.1.4 Space-time kernel density estimation with inhomogeneous background 37

3.2.3 Case data 41

3.2.4 Population data 43

3.2.5 Research objective 1 analysis 47

3.2.5.1 Uncertainty from population simulation 48

3.2.5.2 Benefit of considering time and cluster significance 49

3.3 Research objective 2 methodology 51

3.3.1 The existing method 52

3.3.2 The ST-FLEX-D approach 56

3.3.2.1 ST_FLEX_D_base 56

3.3.2.2 ST_FLEX_D_uneven 60

3.3.2.3 ST_FLEX_D_alternate 62

3.3.3 Research objective 2 analysis 63

3.4 Research objective 3 methodology 66

3.4.1 Global sensitivity analysis 67

3.4.2 Research objective 3 analysis 71

CHAPTER 4: RESULTS 77

4.1 Research objective 1 results 77

4.1.1 The uncertainty from population simulation 77

4.1.2 Benefit of adding the temporal to our analysis 80

4.1.3 Significant clusters 82

4.2 Research objective 2 results 83

4.2.1 Execution time of decomposition 83

4.3.2 Total number of cut circles 85

4.3.3 Average leaf node depth 87

4.3.4 Average leaf node size 88

4.3 Research objective 3 results 90

4.3.1 Logistic regression 90

4.3.2 Sensitivity Indexes 92

CHAPTER 5: DISCUSSION AND CONCLUSIONS 97

5.1 General discussion 97

5.2 Research objective 1 discussion 103

5.3 Research objective 2 discussion 106

5.4 Research objective 3 discussion 107

REFERENCES 108

APPENDIX: PSEUDOCODES 123

LIST OF TABLES

Table 1: Parameter values for ST-STATIC-D and ST-FLEX-D. 66

Table 2: Input parameters and ranges. 74

Table 3: Logistic regression results. 91

Table 4. First-order Sensitivity Indexes. 93

Table 5. Total-order Sensitivity Indexes. 94

Table 6. Difference between Total-order and First-order Sensitivity Indexes. 95

Table 7. 1 – sum of First-order Sensitivity Indexes. 96

LIST OF FIGURES

Figure 1: Three issues of space-time pattern detection. 30

Figure 2: The contribution map. 31

Figure 3: Distinction between Site-side and Case-side kernel density estimation with inhomogeneous background population. 34

Figure 4: Adaptive bandwidth kernel with inhomogeneous background. 35

Figure 5: The multiway problem. 39

Figure 6: Spatiotemporal nearest neighbors. 40

Figure 7: The city of Cali, Colombia. 42

Figure 8: Flowchart for population disaggregation. 46

Figure 9: Octree-based recursive spatiotemporal domain decomposition. 54

Figure 10: Buffer implementation for handling edge effects. 55

Figure 11: Rule 1 of ST-FLEX-D. 57

Figure 12: Rule 2 of ST-FLEX-D. 58

Figure 13: Rule 3 of ST-FLEX-D. 59

Figure 14: Example of ST-FLEX-D. 60

Figure 15: Uneven candidate splits. 61

Figure 16: ST-FLEX-D-alternate. 63

Figure 17: Domain decomposition. 65

Figure 18: Performance metrics and their influencing factors. 66

Figure 19: Matrices A, B, C_i and D_i. 70

Figure 20: Inputs, Model and Outputs. 72

Figure 21: Histogram of differences between upper and lower envelope. 77

Figure 22: The upper simulation envelope (population simulation). 79

Figure 23: Difference between upper and lower simulation envelope. 80

Figure 24: Difference between odds ratios S-IB - ST-IB. 81

Figure 25: Voxels that form a significant cluster at the 0.01-level. 83

Figure 26: Average execution time in seconds. 84

Figure 27: Number of cut circles. 85

Figure 28: Number of cut circles vs. bandwidths. 87

Figure 29: Average leaf node depth. 88

Figure 30: Average leaf node size ST_STATIC_D, ST_FLEX_D_base, ST_FLEX_D_uneven. 89

Figure 31: Average leaf node size. ST_FLEX_D_alternate. 90

CHAPTER 1: INTRODUCTION

1.1 Background

Cyberinfrastructure and high-performance computing (HPC) are transforming many disciplines such as Geography, Engineering or Biology, by enabling them to solve computational problems that were previously inconceivable or intractable (Armstrong 2000). This has led the scientific community to extend the classic method of science, which links the two pillars of *Theory* and *Experimentation*, by adding the third and fourth pillars, *Simulation* and *Data-Intensive Computing* (Karin and Graham 1998; Hey, Tansley, and Tolle 2009). Even prominent computer scientists who oppose such extension of the existing model agree that HPC is an universal enabler of science, which substantially supports theory and experimentation, essentially making the initial pillars thoroughly computational (Vardi 2010). Therefore, most if not all of the diverse scientific disciplines share a consistent integrating principle: Using mathematical models to gain knowledge, to conduct scientific investigations, and to assist decision making. Models of complex human and natural phenomena require computation in order to produce results, and geographers have been creating and applying them for many years (Armstrong 2000). Therefore, we are well equipped to capitalize on the strengths and opportunities of HPC for understanding and modelling spatial processes.

The advancement of technology has enabled us to collect increasing volumes of spatial and spatiotemporal data at rapidly increasing rate through sensor systems, automated geocoding abilities and social media platforms, such as Facebook or Twitter

(Goodchild 2007). These data are characterized by unprecedented volume, velocity and variety, and hence, call for increased computing performance which is indicative for the big data era we live in (Zikopoulos and Eaton 2011). On the other hand, geographic models are computationally intensive, because their underlying algorithms are typically of exponential complexity. Furthermore, the computational burden increases manifold when simulations (e.g. Monte-Carlo) are used for significance testing (Tang, Feng, and Jia 2015). Last, the recent trend to include a true representation of time in geographic models, together with the advent of big spatiotemporal data, further increases the importance of strategies to handle complex computations on massive datasets (Kwan and Neutens 2014).

Due to the recent abundance of geospatial information, scientists have developed methods for spatial and temporal analysis of georeferenced data (Anselin 2011). The combination of geographical information systems (GIS) and space-time analytics has enabled them to explore large databases of individual-level observations. Examples include crime or disease, which usually form a non-random spatiotemporal pattern, for instance clustering at city centers or exhibiting seasonal cyclic patterns. Knowledge about the intensity, spatial location and time of such clusters can inform authorities on their decision to allocate resources, such as staff for disease prevention efforts (Casas, Delmelle, and Varela 2010). Spatial and spatiotemporal statistics are a set of popular analytical methods for identifying and quantifying such patterns, as they capture geospatial phenomena and their variability in space and time (Bailey and Gatrell 1995; Cressie and Wikle 2015). They can be grouped into geostatistical appraoches,

autocorrelation-oriented approaches, and point pattern analysis, whereas the latter is particularly suited to analyzing individual-level point events (Diggle 2013). Among the armada of exploratory statistics to characterize a given spatiotemporal point pattern, space-time kernel density estimation (STKDE; Nakaya and Yano 2010) stands out. It allows for visualizing the occurrence of events in space and time by computing the localized intensity of the point process at hand and hence, summarizing the distribution of a spatial variable through time. STKDE has been employed as a key analytical procedure for identifying clusters of crime (Nakaya and Yano 2010), exploring human mobility patterns (Gao 2015), as well as outbreaks of dengue fever (Delmelle et al. 2014). However, methods for analyzing spatiotemporal data of increasing size, diversity and availability are limited by their exorbitant computational cost, which results in prohibitively slow execution times, coarse resolutions, and low statistical significance levels. High-performance and parallel computing (HPC) offer solutions to computationally demanding problems in limited time frame.

HPC meets the demand for increased computing performance for big data analytics by deploying multiple computing resources concurrently (Wilkinson and Allen 2004). A general approach for parallel problem solving is to decompose the spatiotemporal domain of a dataset into smaller subsets, distribute them to multiple concurrent processors, i.e. computing a spatial statistic, and finally collect and reassemble the results (Ding and Densham 1996). Balancing computational intensity among processors by accounting for the explicit characteristics of the data is crucial for developing scalable applications (Wang and Armstrong 2003; Wang 2008). Recursive

domain decomposition methods, such as quadtrees and octrees, have been widely used for mitigating workload imbalance for spatially and temporally heterogeneous data (Turton 2000; Hohl, Delmelle, et al. 2016). At each step of this procedure, one has to make choices about parameters, which determine the result of the analysis, as well as the execution time of the computation. Hence, we face a considerable uncertainty about the computational cost of our analysis, especially if parameter values are not set in stone. If we knew how computing performance relates to different parameter values and their combinations, we could allocate parallel resources in a more efficient way. Sensitivity Analysis (SA) offers a solution to investigate this relationship in a systematic way.

SA is a domain that studies "how the uncertainty in the output of a model (numerical or otherwise) can be apportioned to different sources of uncertainty in the model input" (Saltelli et al. 2004). It allows for evaluating the contribution of model parameters to variability in model outputs in a quantitative manner, and facilitates the understanding of driving factors and model structure. Sensitivity Analysis has been useful for applications in ecology, hydrology, engineering and economy (Saltelli et al. 2008; Lilburne and Tarantola 2009; Tang and Jia 2014). There exist multiple approaches for SA that have unique characteristics pertaining to model dependency, computational requirements, and compatibility with spatiotemporal variables: local, regression, and global variance-based. Sobol's approach is an example of the variance-based approaches, which is particularly interesting because of its model independency, support of nonlinearity and spatially explicit data (Sobol 1993).

With the computational power of HPC and SA, we gain the capability to run spatiotemporal analysis at scale. This enables us to solve real-world problems by applying this capability towards a problem domain, such as spatial epidemiology. The transformation of spatial epidemiology through the advent of cyberinfrastructure is particularly interesting due to the increase of emerging and re-emerging infectious diseases (EIDs), caused by demographic, environmental, social and technological changes in human ecology which reshape the relationship between humans and microbes (McMichael 2004). EIDs transmission cycles are complex processes, which require monitoring strategies for effective public health responses to disease outbreaks under critical space-time conditions (Eisen and Eisen 2011). Identifying clusters of illness facilitates timely measures to cope with outbreaks and is thus a critical element to reduce the burdens associated with diseases (Grubesic, Wei, and Murray 2014). Spatial and spatiotemporal statistics, such as STKDE, are suited for disease-surveillance because they enable discovery of previously concealed patterns, such as intensity and risk of diffusion to new regions, directionality, the rate of disease spread, and cyclic patterns, (Jacquez, Greiling, and Kaufmann 2005; Rogerson and Yamada 2008; Robertson and Nelson 2010; Kulldorff 2010).

Many challenges of using advanced computational methods to enable spatiotemporal analytics at a large scale have remained unaddressed. First, when mapping disease risk (density of disease cases for instance), conventional STKDE conveys the assumption of a spatially and temporally homogeneous background (population at risk). Therefore, a cluster of elevated disease risk might merely be due to a high population

density. Approaches exist to adjust for spatially varying background by allowing for variable bandwidths to gain constant population support (Shi 2010; Davies and Hazelton 2010; Davies, Jones, and Hazelton 2016; Tiwari and Rushton 2005), but their temporal extensions have not been addressed so far. Hence, adjusting for a spatially varying background is a common procedure to date, but by assuming a temporally homogeneous background, existing approaches are unable to capture population that may increase or dwindle through time. In the face of the current rapid urbanization and migration (Castles, De Haas, and Miller 2013), this assumption no longer holds, and methods for mapping disease risk need to be updated accordingly. Second, spatiotemporal domain decomposition methods suffer from inefficiency because they repetitively split the dataset, thereby introducing new boundaries, along with undesired boundary effects. Methods to deal with this problem have so far been unsatisfactory (Hohl, Delmelle, and Tang 2015), and improving spatiotemporal domain decomposition strategies by making educated choices about the bisection positions holds a substantial potential for gaining efficiency and therefore, scalability. Third, quantifying computational intensity for load balancing using a single set of parameters is not realistic. It ignores uncertainty that arises from variable parameter values along with their interactions and therefore, is only valid for one specific case. Accelerating spatiotemporal data mining algorithms for all reasonable inputs necessitates analyzing the sensitivity of computing performance to various parameter sets in systematic ways. Employing concepts from the domain of sensitivity analysis (Saltelli et al. 2004) for studying the influence of model parameters on scalability of spatiotemporal domain decomposition for parallel space-time analytics remains an effort that is missing in the literature so far. Finally, we need to design,

modify, or extend new geographic algorithms, taking advantage of HPC and cyberinfrastructure, allowing us to mitigate performance and computational complexity issues for better support of scientific discovery and decision making.

1.2 Research objectives

My dissertation contributes to the body of literature on high-performance computing of spatiotemporal statistics within the domain of GIScience. Specifically, I challenge a key assumption of many prominent population adjustment methods for kernel density estimation of disease risk, a popular point pattern analysis application. I present Space-Time Kernel Density Estimation for Spatially and Temporally Inhomogeneous Background Populations (ST-IB), an improvement upon existing work by taking into account a spatially and temporally inhomogeneous background population. In addition, I develop a spatiotemporal domain decomposition approach called Flexible Spatiotemporal Domain Decomposition (ST-FLEX-D), which allows for tackling the big data challenge of developing scalable approaches to compute spatiotemporal statistics, using high-performance parallel computing. Lastly, I propose Sensitivity Analysis for Spatiotemporal Computing (ST-SA), a framework for assessing the effect of model parameter values on computing performance and scalability. The results of ST-SA might guide practitioners on the computational requirements of their application of spatiotemporal statistics on large scales. Therefore, I aim for advancing the body of knowledge with three distinct contributions:

1.2.1 Objective 1

Develop and implement adaptive kernel density estimation to address spatially and temporally inhomogeneous background populations (ST-IB). Develop a metric to measure its performance in detecting clusters of elevated disease risk. Compare the performance of ST-IB to an existing approach that ignores the temporal dimension and assume temporal homogeneity (Shi 2010).

1.2.2 Objective 2

Accelerate kernel density estimation using spatiotemporal domain decomposition for parallel processing. Develop a flexible splits heuristic to minimize domain replication (ST-FLEX-D). Compare flexible decomposition to static decomposition (ST-STATIC-D) using execution time of decomposition, as well as standard parallel performance metrics for subsequent STKDE.

1.2.3 Objective 3

Using Sobol's method, a variance-based approach for global sensitivity analysis, as well as multivariate regression, study how uncertainty in the computational cost of a model can be apportioned to different sources of uncertainty in the model input parameters. Use spatiotemporal domain decomposition (ST-STATIC-D) for parallel STKDE as case study.

1.3 Road map

This dissertation is organized as follows: Chapter 2 provides a literature review on four main parts: spatiotemporal statistics (point pattern analysis, kernel density estimation), disease mapping (inhomogeneous population support, spatial relative risk function), parallel strategies for spatiotemporal statistics (HPC, domain decomposition), global sensitivity analysis. Chapter 3 describes the methodology for each of the 3 research objectives. Chapter 4 contains results for each of the 3 research objectives. Chapter 5 contains discussion and conclusions.

CHAPTER 2: LITERATURE REVIEW

This section contains a thorough literature review of relevant topics with regard to the research objectives, including 1) *spatiotemporal statistics* and its subfields of autocorrelation-based approaches, geostatistics, and point pattern analysis, 2) *parallel strategies,* focusing on spatiotemporal domain decomposition, 3) *sensitivity analysis* with its local approaches, regression, and variance-based approaches.

2.1 Spatial and spatiotemporal statistics

The domain of *spatiotemporal analysis* encompasses the three subfields of *spatiotemporal statistics*, *optimization*, and *simulation*. Here, I focus on *point pattern analysis, a subfield of spatiotemporal statistics,* which also includes *autocorrelation-based approaches* and *geostatistics.*

2.1.1 Autocorrelation-based approaches

The first law of geography states that "Everything is related to everything else, but near things are more related than distant things" (Tobler 1970). Therefore, strong positive spatial autocorrelation indicates that within the area of interest, objects that are located close to each other are very similar compared to distant objects (attraction). On the other hand, negative spatial autocorrelation means that close objects are dissimilar (repulsion). Moran's *I* (Moran 1950) is an instance of an autocorrelation-based approach within the field of spatial statistics. For a given set of points in geographic space, it

measures whether the values of an attribute of interest are clustered, dispersed, or random. Moran's *I* has many practical applications in geography, ecology, epidemiology, criminology, and econometrics, to name a few (Assuncao and Reis 1999; Zhang et al. 2008; Lin and Zhang 2007). Efforts for parallelizing the computation of Moran's *I* statistic have been taken early on by using Linda, a coordination language that supports parallel processing (Rokos and Armstrong 1996). Similarly, the G* statistic measures spatial association of high/low attribute values (Getis and Ord 1992). Parallel implementations exist (Armstrong, Pavlik, and Marciano 1994), utilizing algorithmic decomposition, and by deployment on a grid computing platform (Wang, Cowles, and Armstrong 2008). The analysis of computational intensity by reference of the spatial domain was formulated using the G* statistic (Wang and Armstrong 2009). In addition, geospatial analysis capabilities using G* have been taken to the MapReduce platform (Liu et al. 2010) for true big data processing, and cyberinfrastructure was employed to host Web-GIS services accessing TeraGrid computational resources (Wang and Liu 2009).

So far, I discussed global autocorrelation-based approaches, which produce one statistic for the entire study area. However, we may be interested in knowing where clustering (or repulsion) occurs. Global statistics can be decomposed into their local constituents. Therefore, local indicators of spatial autocorrelation (LISA) allow for mapping clustering of attribute values (Anselin 1995). Local Moran's *I* has been extended from purely spatial to space-time and applied to analyzing water distribution

networks and cancer mortality rates (Difallah, Cudre-Mauroux, and McKenna 2013; Goovaerts and Jacquez 2005).

2.1.2 Geostatistics

Geostatistical methods were developed to predict probability distributions for ore grades in the mining industry and therefore, are frequently used in the petroleum industry and engineering (Cressie 1990). Kriging is a popular geostatistical method, which spatially interpolates attribute values assuming a gaussian process. As kriging involves costly computations, scientists developed parallel approaches utilizing general purpose graphics processing units (GPGPU), message passing interface (MPI), and advanced data structures (Cheng 2013; Wei et al. 2015; Hu and Shu 2015). Spatiotemporal Kriging is challenging because characterizing correlation requires advanced statistics for the space-time domain, therefore further increasing the computational burden (Heuvelink and Griffith 2010). Kriging examples include the modelling of copper deposits in southeastern Iran (Daya and Bejari 2015), and estimation of coal layer quality for reducing the costly use of borehole drilling during geophysical exploration (Webber, Costa, and Salvadoretti 2013).

2.1.3 Point pattern analysis

Spatial and spatiotemporal *point pattern analysis* studies the arrangement of points and aims to distinguish random, clustered and dispersed patterns (Cressie and Wikle 2015). As opposed to *autocorrelation-based* approaches and *geostatistics*, spatial and spatiotemporal *point pattern analysis* do not focus on attribute values. This approach

has been useful for many applications, such as disease ecology (Kelly and Meentemeyer 2002), neurology (Jafari-Mamaghani, Andersson, and Krieger 2010), epidemiology (Shi 2010; Delmelle et al. 2014), and criminology (Nakaya and Yano 2010).

2.3.3.1 Ripley's K function

Ripley's *K* function (Ripley 1976) is a popular quantitative approach within the domain of spatial and spatiotemporal point pattern analysis. It characterizes a point pattern as either random, clustered or regular, by estimating the second-order property (variance) of the data. It considers 1) the number of and 2) the distance between points, to quantify the deviation of the observed pattern from randomness at multiple spatial scales (Bailey and Gatrell 1995; Dixon 2002). Essentially, Ripley's *K* function centers a circular search window on each data point and counts the number of neighboring points observed in the window. This process is repeated using varying search radii (a.k.a. spatial scales). To gain statistical confirmation on the regular, clustered or random pattern, at each of the tested radii, Monte Carlo simulations are used: Each simulation run randomly generates a number of points (the number is equal to the number of observed points). The *K* function can be transformed to *L*-function to obtain a benchmark of zero, which allows for direct comparison across all spatial scales assessed. Therefore, a point pattern is clustered if $L > 0$, within a given spatial distance. If $L < 0$, then the point pattern is regular, and if $L = 0$, the pattern conforms to complete spatial randomness (CSR). If the observed number *L* is above the upper envelope resulting from the simulations, clustering at the corresponding spatial scale is significant. The radius that yields the largest difference between observed *K* function and upper envelope is considered the most significant (and therefore, *optimal*

in that sense) scale. *L* values below the lower simulation envelope indicate regularity. Undesired edge effects may arise upon intersection of the search windows with the study area boundary. Methods to cope with edge effects have been studied intensively, and include the circumference method, toroidal method and inner guard method (Yamada and Rogerson 2003).

The Ripley's *K* function has applications in many scientific domains: early detection of breast cancer (de Oliveira Martins et al. 2009), analyzing oak mortality in California (Kelly and Meentemeyer 2002), and analyzing the geographic distribution of road traffic accidents (Jones, Langford, and Bentham 1996), to name a few examples. There are multiple improvements to the original formulation of Ripley's *K* function. The incremental *K* function improves the ability to detect the scale of clustering by altering the search window to a search band formed by two concentric circles, just like a doughnut (Yamada and Thill 2007; Tao, Thill, and Yamada 2015). The multivariate (cross) *K* function evaluates clustering of marked point patterns (Dixon 2002; Boots and Okabe 2007). Therefore, it measures the scale at which one set of points (e.g. crime events) cluster around another set of points (e.g. train stations). The cross *K* function was extended to test the association between spatial objects of differing dimensionality, such as points, lines, and polygons (Guo et al. 2013). The local *K* function belongs to the class of local indicators of spatial association (LISA; Anselin 1995) and allows for detecting and mapping clusters of points, as well as the spatial scale at which the clustering occurs (Getis and Franklin 1987). The network-constrained *K* function evaluates clustering of points along networks, such as roads, railways, or utility grids, which is advantageous,

i.e. for studying vehicle collisions or power outages. It has been enhanced by the incremental *K* function, as well as by corrections for network geometry (Okabe and Yamada 2001; Yamada and Thill 2007; Ang, Baddeley, and Nair 2012). Based on the network –constrained *K* function, Tao (2017) develops three new methods for exploratory spatial flow data analysis. Flow data quantifies the stream of goods, people or any object between two or more locations. Lastly, GPU were utilized to speed up the computation of Ripley's *K* for spatial point pattern analysis (Tang, Feng, and Jia 2015).

2.3.3.2 Kernel density estimation

Kernel density estimation (KDE; Silverman 1986) is a popular technique for producing heat maps. It essentially generates a regular grid of points (sites) that hold density estimates which depend on the number and position of surrounding data points. Each pair of data point and site that are separated by less than a maximum distance contributes to density at the site by applying a weight determined by the kernel function (the smaller the distance between data point and site, the higher the weight). Many different kernel functions exist: Quartic, Gaussian, Epanechnikov, to name a few. Space-Time Kernel Density Estimation (STKDE; Nakaya and Yano 2010) produces a density volume, which consists of regularly spaced points (sites, voxels) for which a density estimate is calculated. It illustrates that the computational burden is very high and depends on the spatial or spatiotemporal distribution of the point-events to be analyzed. STKDE is great for identifying space-time clusters, and has been implemented for visualizing disease patterns (Delmelle et al. 2014), identifying clusters of crime (Nakaya and Yano 2010), exploring human mobility patterns (Gao 2015).

Unlike the selection of bandwidth, the choice of kernel function is not critical for the distribution of the resulting density estimates and the visual properties of the heat map (Bowman and Azzalini 1997; Silverman 1986). If the bandwidth is chosen too large, important details of the point pattern disappear, as they are oversmoothed. On the other hand, if the bandwidth is too small, the density surface is too rough or spikey and important patterns are unobservable. There are a number of bandwidth selection techniques: The spatial (and temporal) scale resulting from analyzing spatial data using *K* function can be used as bandwidth (search radius) for KDE, optimization approaches, least squares cross validation, and several rules of thumb have been developed (Wand and Jones 1994). So far, a consensus about which method is universally best does not exist.

2.2 Kernel methods for disease mapping

2.2.1 Spatial filters

Spatial filters (a.k.a. box kernels) are a special case of kernels, where the weight does not decrease within bandwidth. Spatial filters are used for detecting local hot spots and significance tests on case and control data, for instance to compute birth defect rates in Iowa (Rushton and Lolonis 1996). The problems of using aggregated disease data within artificial (administrative) units, such as census tracts, motivated Rushton's work. Hence, he and his colleagues use individual geocoded maternal addresses of births, as well as birth defects. They evaluate the birth defect rates at regular grid points, by

imposing circles within which births and defects are considered. They test for significance by simulating 1000 random datasets in which points were marked as “birth” or “defect” according to the probability arising from their ratio in the observed data. Therefore, the birth defect rate at each grid point (site) can be compared to the upper simulation envelope for significance. Spatially adaptive filters address the problems of 1) losing geographic detail and 2) producing unreliable estimates in sparsely populated areas (Tiwari and Rushton 2005). An adaptive filter is a circle centered on a grid point, which increases its radius until it achieves a minimum population support. Hence, they achieve statistical stability in less populous areas and high spatial certainty in more populous areas. Cai, Rushton and Bhaduri investigate the multiple testing problem, where large numbers of individual tests result in inflated type 1 error, whereas autocorrelation arises from overlapping neighboring filters. They improve upon spatially adaptive filters by weighting observations according to their proximity to the center of the filter using a staircase kernel (Cai, Rushton, and Bhaduri 2012).

2.2.2 The spatial relative risk function

The spatial relative risk function, computed as the ratio between density of disease cases and density population at risk, is another important kernel approach in health geography, with applications in mapping Type 2 Diabetes Mellitus (Kauhl et al. 2016), and cancer incidences (Lemke et al. 2015). Fixed and adaptive bandwidths can be used and produce different results for detecting risk areas. Lemke and colleagues (2015) found that the fixed kernels tend to oversmooth in urban areas, while overestimating the risk in rural areas. An adaptive bandwidth kernel can reduce the effect. They choose a

fixed bandwidth using the oversmoothing principle (Terrell 1990), and the adaptive kernel bandwidth as a function of population density. Other efforts towards case-based adaptive bandwidth relative risk function (Davies and Hazelton 2010) use case and control bandwidths computed for each evaluation point, depending on density of surrounding observations. Essentially, this includes computing pilot densities using cross validation to determine fixed pilot bandwidths separate for cases and controls (Bowman and Azzalini 1997), a global bandwidth (chosen by the maximal smoothing principle, common to case and control densities) and a geometric mean term. Davies and Hazelton's simulation study suggests that the adaptive density estimator is more desirable because the fixed bandwidth estimators are not able to properly capture clusters while maintaining a sufficient degree of smoothness over the rest of the region. However, the authors admit that their method of bandwidth selection is rather ad-hoc and that there is scope for new methods of bandwidth selection for relative risk estimation.

Adaptive kernels using different bandwidths for computing the case- and control densities are likely to produce artificial risk halos, which are undesired artefacts in the resulting risk surface (Davies, Jones, and Hazelton 2016). Applying a symmetric adaptive smoothing scheme addresses the problem. The case and control bandwidths are the same, and determined by population density, similar as in (Lemke et al. 2015). Martin Hazelton considers the problem of statistically comparing relative risk between two time periods and develops a statistic to test for change in the pattern of relative risk (Hazelton 2017). Note that this is a global measure of change, i.e. it tests the entire study area, which raises the question whether a localized test is feasible. The observed test statistic is compared to

a null distribution, which is generated by either 1) randomization with a substantial number of replications (~1000), or 2) using asymptotic theory, which shows that a kernel estimator has an asymptotically normal distribution.

Fernando and Hazelton postulate the generalized spatial relative risk function (Fernando and Hazelton 2014). Most interestingly, their approach generalizes the spatial relative risk function to the spatiotemporal relative risk function by making the assumption of constant control density through time. However, if this assumption does not hold, the spatial relative risk function generalizes to the conditional spatiotemporal relative risk function. It uses common bandwidths for case and control densities, which has been shown to be advantageous, see (Davies, Jones, and Hazelton 2016). In addition, tolerance contours delineate statistically significant areas/times of elevated risk using asymptotic theory (as an alternative to the computationally costly Monte Carlo methods). For lack of better alternatives, Fernando and Hazelton select optimal bandwidths by minimizing the mean integrated squared error (MISE) using approximate least-squares cross-validation (LSCV), but admit that the procedure is unsatisfactory in practice due to high variability of outcomes. They conclude that the generalized spatial relative risk function is a useful data visualization tool, but admit that it suffers from problems in data-driven bandwidth selection. Further research towards selecting bandwidth for the spatial relative risk function is critically needed.

2.3 Parallel strategies for spatial and spatiotemporal statistics

There are two classes of parallel computing architectures: 1) SIMD (Single Instruction stream, Multiple Data stream) architectures focus on data parallelism, where multiple processors concurrently execute the same set of instructions on different datasets. In 2) MIMD (Multiple Instruction stream, Multiple Data stream) architectures, multiple processors execute different instructions on different datasets (Ding and Densham 1996). There are two popular paradigms for high-performance and parallel computing that base off SIMD and MIMD: multi-core and many-core computing. Multi-core computing extends single core computing through shared memory modules that are accessed by multiple processors (Wilkinson and Allen 2004). Using graphics processing units (GPUs) that receive instructions from CPUs to tackle large computations is a good example of many-core computing. GPUs have been originally developed to support graphic displays, but have been adapted to speed up scientific computations because of their suitability to exploit data parallelism (Tang and Bennett 2011). There are three parallel approaches for multi- and many-core computing: 1) embarrassingly parallel, 2) shared memory, and 3) message passing (Wilkinson and Allen 2004). Every approach has its unique characteristics, advantages and shortcomings and therefore, suits different parallel applications. The embarrassingly parallel approach does not allow concurrent processors to exchange data and instructions. If communication among processors is necessary, the shared memory and message passing approaches are preferred. In shared memory approaches, each memory module is accessible by multiple processors and data is exchanged through common memory space. Message-passing computing allows for communication among processors by sending or receiving messages. However, the

message-passing and shared memory approaches, suffer from communication overhead that reduces computing performance. On the other hand, the embarrassingly parallel approach requires prior decomposition of the computation, which may introduce issues of load balancing.

Spatial problems can be classified based on their domains into regular/irregular and homogeneous/inhomogeneous (Ding and Densham 1996). One remarkable feature of spatial modelling is the divisibility of its domain, for instance by using square or rectangular tessellations that are suitable partitioning strategies due to nature of spatial coordinate systems. The resulting subdomains can assigned to multiple concurrent processors, i.e. to compute a spatial statistic, using the embarrassingly parallel approach. The spatial domain decomposition can be one-dimensional, forming horizontal or vertical stripes, or many-dimensional, i.e. forming a checkerboard pattern. However, regular partitioning of irregular or inhomogeneous domains introduce workload imbalance, where computing resources are not used to their full potential because some processors might be finished earlier than others and therefore, stay idle. Hence, recursive approaches, such as quadtrees (Samet 1984) are used to partition the spatial domain into a hierarchical set of rectangles/squares, a.k.a. subdomains, which contain a more or less equal number of data elements and which then are assigned to processors for evenly balanced workloads. However, combining the results from non-overlapping subdomains may result in incorrect results due to the dependence of many spatial analysis and modelling approaches on neighborhood information, for instance through nearest-neighbor search. Using quadtrees for recursive spatial domain decomposition introduces

new boundaries due to the splits at axis midpoints, which cause edge effects in kernel density estimates if not properly dealt with (Hohl, Delmelle, and Tang 2015).

There are many applications and extensions to spatial domain decomposition procedures for various scientific disciplines: Inverse–distance interpolation (IDW) has been parallelized using quadtree spatial domain decomposition and task scheduling within a grid computing environment (Wang and Armstrong 2003; Desjardins et al. 2017). Interpolating massive LIDAR point clouds to create DEMs calls for accelerated processing capabilities. Guan and Wu (2010) use thread-based parallelism for multi-core CPUs, as well as regular spatial domain decomposition for their hybrid nearest neighbor search procedure: The search either terminates if there are no points found within a specified distance *d* from the unknown location, or if the maximum number of nearest neighbors *k*, is reached. *d* is determined by average point density, spacing, as well as the total number of points and area. Similarly, parallel implementations exist for the G*(d) statistic: Wang, Cowles and Armstrong (2008) decompose the spatial domain using an improved quadtree, where the size of the quads is restricted to twice the bandwidth (d). It allows for reducing distance computations across neighboring quads, while Morton quad indexing minimizes the use of points outside quads. Zhang and You (2013) construct BMMQ (binned min-max quadtrees) on raster datasets for spatial indexing to support range queries. The BMMQ tree stores statistics (minimum and maximum values) with each quadtree node, while the contained raster cells are binned to reduce tree complexity. A MapReduce (Dean and Ghemawat 2008) version of the G*(d) statistic, using quadtrees for spatial indexing was presented by Liu et al. (2010). They solve the boundary problem

by storing a matrix of the minimum distances between each quadtree node (subdomain). If the distance between node *a* and node *b* is larger than the bandwidth, node *b* does not have to be taken into consideration when querying from node *a*. This is an alternative approach to the buffer implementation, where we replicate points within bandwidth from subdomains. Finally, Hohl et al. (2016) used octree decomposition for accelerating the STKDE algorithm, and tackle the boundary problem by implementing subdomain buffers, for which data points are replicated and assigned to multiple subdomains. However, as already evidenced by Gu (2011), the inclusion of time complicates requirements for spatial indexing, resulting in low retrieval efficiency. For further acceleration, Hohl, Casas et al. (2016) use a hybrid strategy, performing octree-based recursive decomposition of the space-time domain and using k-d tree indexing within octree leaf nodes (Liu et al. 2008) for parallel computation of STKDE. K-d tree is a binary tree structure for arranging points in k-dimensional space (Bentley 1975), allows for efficient retrieval, and has been widely used for NN search. Merging multiple indexing methods to form hybrid spatiotemporal indices was recently proposed by Azri et al. (2013).

Recent advancements of cyberinfrastructure and HPC in the domain of geospatial applications (or GIScience) had significant impacts on health and wellbeing in urban settings through participatory data collection using mobile devices (Yin, Gao, and Wang 2017). The ability to process exorbitant amounts of data through HPC has enabled the study of human digital footprints (i.e. Tweets), which led to an understanding of urban land use at high resolution, as well as mobility patterns (Soliman et al. 2017). The

analysis of social media data enabled by cyberinfrastructure, has thrust a series of applications at the intersection of spatiotemporal analysis and health (Gao et al. 2018; Padmanabhan et al. 2014; Ye et al. 2016; Shi and Wang 2015). Other application domains include hydrologic modelling (Survila et al. 2016; Ye et al. 2014; Y. et al.), biomass and carbon assessment (Tang et al. 2017; Tang et al. 2016; Stringer et al. 2015), andagent-based modelling (Shook, Wang, and Tang 2013; Fachada et al. 2017; Tang 2008; Tang and Bennett 2010, 2012; Tang and Wang 2009; Tang et al. 2011).

2.4 Sensitivity analysis

Sensitivity analysis (SA) studies "how the uncertainty in the output of a model (numerical or otherwise) can be apportioned to different sources of uncertainty in the model input" (Saltelli et al. 2004, 45). It facilitates the understanding of relationships between the results of a model (i.e. regression coefficients) and its input factors (i.e. data and parameters), such as the polynomial degree, identifies driving factors, as well as model structure. Applications of SA are found in many scientific domains, such as ecology, hydrology, engineering and economy (Saltelli et al. 2008; Lilburne and Tarantola 2009; Tang and Jia 2014). SA encompasses the following steps: 1) defining an objective function, 2) selecting input factors, as well as their distribution functions, 3) random sampling, 4) model evaluation, 5) and analyzing model outputs. SA approaches are grouped into *local*, *regression*, and *variance-based*, each with their own advantages and drawbacks in terms of model dependency, computational requirements, and support of spatiotemporal variables. Sobol's approach is an example of the latter, which is particularly interesting because of its model independency, support of nonlinearity and

spatially explicit data (Sobol 1993). SA has proven useful for many applications within GIScience, i.e. to investigate the computational aspects of agent-based modelling (Tang and Jia 2014), or to study uncertainty in cellular-automata modelling for urban growth simulation (Şalap-Ayça et al. 2018).

2.4.1 Local approaches

The effect of a given model input factor on a given output defines local approaches (not to be confused with the notion of "local" in the context of spatial statistics). This is either obtained by changing one factor at a time (OAT), or by computing derivatives, i.e. the instantaneous rate of change, the ratio of the instantaneous change in model output to that of the input factor (parameter, variable) under study (Rabitz 1989; Turányi 1990). However, in different settings, such as the analysis of risk (financial, industrial, disease, disaster), a quantitative assessment of the uncertainty around the model output is desired. For the risk analyst, the degree of variation is important, which may not be captured by OAT or derivative-based approaches, especially when analyzing highly non-linear models. In addition, local methods for sensitivity analysis discard the possibility of interactions between factors: factors interact if their compound effect on the output cannot be expressed as a sum of their single effects. Hence, it is impossible to rank them in order of their importance, i.e. the amount of variance they account for in the model output (Saltelli et al. 2004). The only advantage of local approaches is their computational efficiency, as the number of model evaluations is small.

2.4.2 Regression approaches

Regression-based sensitivity analysis uses standardized regression coefficients (SRCs; Draper and Smith 2014), Pearson correlation measures, or partial correlation coefficients (PCC), to quantify the effect of uncertainty in model inputs. Hence, using standardized model inputs as independent variables and model outputs as dependent variable of a least-squares regression model quantifies the effects of model parameter values on model outputs. This method allows for ranking input factors according to their influence on output variance. We can sample the parameter values distributions, including non-uniform ones, and quantify the fraction of the output variance that is accounted for by regression model using the model coefficient of determination (R^2) (Saltelli 1999). Therefore, the closer the R^2 to 1.0, the better are the results. A low R^2 implies that there is a considerable fraction of the output variance left unaccounted for (Saltelli, Tarantola, and Chan 1999). In that case, the ranking of input factors might change if one were able to attribute the remaining fraction of the variance to the factors. Regression-based approaches belong to the class of global sensitivity analysis, and have the advantage of exploring the entire defined range of values for each parameter. They stand in contrast to local approaches, which only perform model evaluations for a predefined local set of parameter values (Saltelli et al. 2004). However, besides being model-dependent, discovering and quantifying interaction effects between factors is a key weakness of regression-based sensitivity analysis approaches (Saltelli et al. 2008).

2.4.3 Variance-based approaches

Variance-based approaches for global sensitivity analysis perform the decomposition of model output variance (Sobol 1993). Hence, they study how variance of the output relates to uncertain input factors. They are recommended because of model independence, ability to incorporate the full range of variation of each input factor, appreciation of interaction between inputs, support of nonlinearity, and spatially explicit variables (Archer, Saltelli, and Sobol 1997). Variance of model output is an important property that naturally arises within a Monte Carlo framework (Saltelli et al. 2004), and is seen as a summary measure of uncertainty (Saltelli et al. 2008). Variance-based sensitivity analysis comes at a high computational cost because the model is evaluated many times for different factor values.

Variance-based approaches were first employed by chemists who proposed to use conditional variances for sensitivity analysis (Cukier et al. 1973). The Fourier Amplitude Sensitivity Test (FAST) uses search-curves which explore the multidimensional parameter space (Cukier, Levine and Shuler 1978), to compute the contribution of each input factor to total variance. An extension of FAST (EFAST) allows capturing higher-order effects and therefore, interactions between factors, by computing sensitivity indexes of arbitrary groups of factors. EFAST employs Monte Carlo methods to overcome the curse of dimensionality resulting from enumerating and computing all possible combinations of factors (Saltelli and Bolado 1998; Saltelli, Tarantola, and Chan 1999). . Interaction is present if extreme output values are uniquely associated with a certain combination of model inputs in a way that cannot be described by first-order effects.

They represent important features of models, give insight on model structure, and are harder to detect than first-order effects (Saltelli et al. 2008). A first-order effect describes the contribution of each input factor to the variance of the output. It can be quantified by a first-order sensitivity index, which is obtained by computing the ratio between the variance of the expected output, conditioned on a given input factor value and the unconditional variance (Homma and Saltelli 1996). As an alternative, Sobol's method (Sobol 2001) decomposes the variance into its constituents, meaning the model inputs. Hence, it allows for identifying interaction effects by computing higher-order terms, whereas the second-order effect is computed by the joint effect of two model inputs minus their first-order effects. It differs from EFAST by its sampling scheme, which makes it computationally more expensive to compute higher-order effects.

CHAPTER 3: Methodology

3.1 Overview

This section introduces the methodology developed for my research, addressing each of my research questions. Figure 1 shows the framework used in my dissertation. It includes: 1) developing the case-side adaptive-bandwidth kernel density estimator for spatially and temporally inhomogeneous backgrounds (ST-IB), 2) implementing a flexible splits heuristic for adaptive spatiotemporal domain decomposition (ST-FLEX-D), and 3) performing global sensitivity analysis of computing performance (ST-SA). Figure 2 shows each of the components and their areas of contribution. Component 1) takes data and threshold parameters as inputs. It produces spatiotemporal risk estimates, clustering, as well as odds ratios as a measure to evaluate clustering performance. The odds ratios allow for comparing performance with other methods for computing risk estimates.The spatiotemporal risk estimates can be visualized and therefore, allow for visual assessment and comparison to other methods, i.e. STKDE of component 2). The case data and a set of parameters serve as inputs of component 2), which includes spatiotemporal domain decomposition for subsequent parallel STKDE. It produces kernel density estimates and computing performance metrics, allowing for comparison with the methodology in component 3). Component 3) takes the case data and parameter ranges as inputs. It performs sensitivity analysis on the two-stage procedure of spatiotemporal domain decomposition and parallel STKDE. Hence, it outputs sensitivity indexes and computing performance metrics.

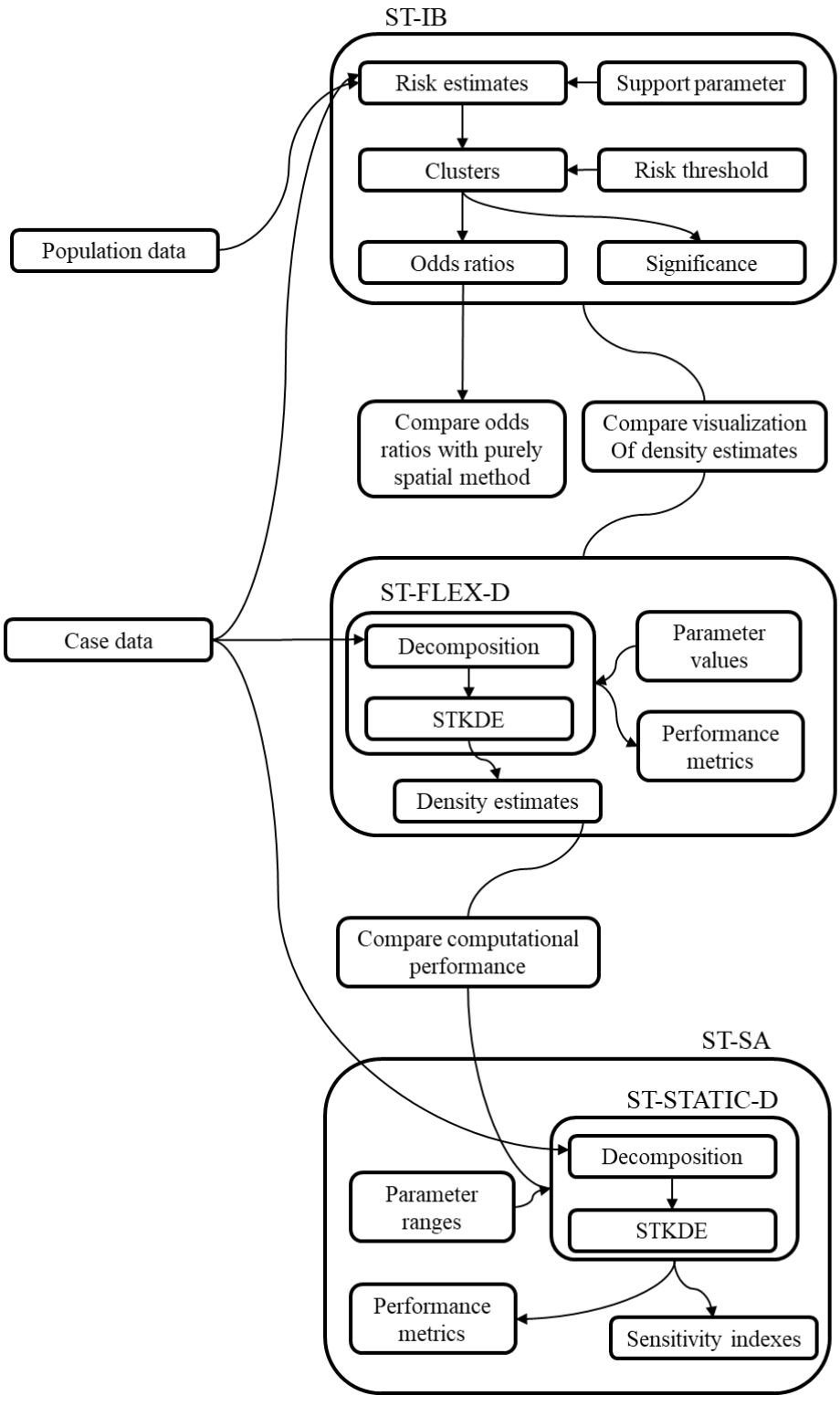


Figure 1: Three issues of space-time pattern detection.

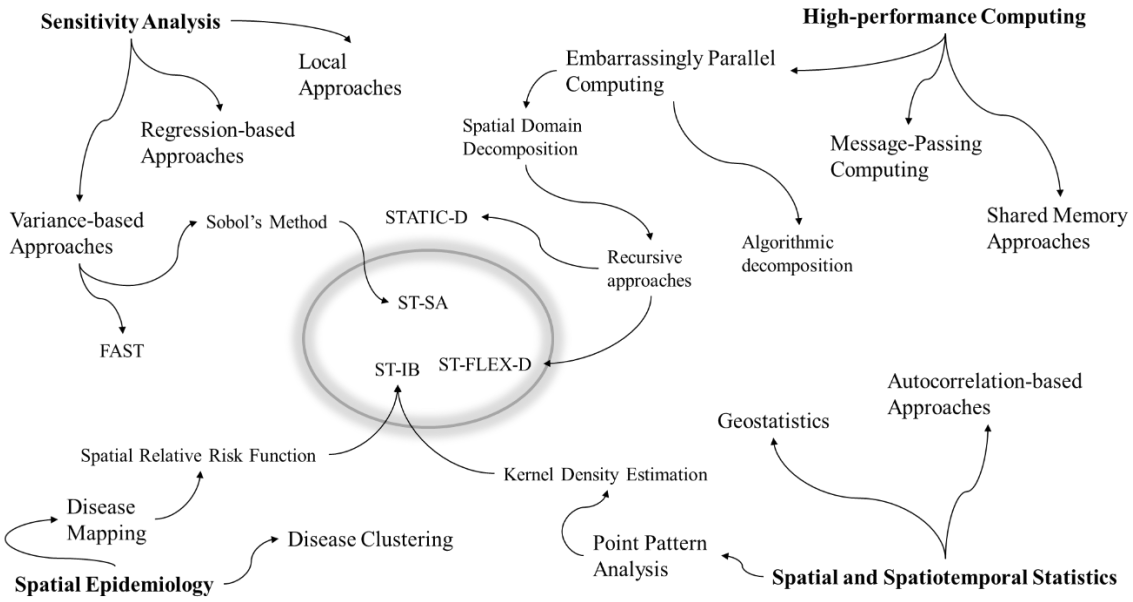


Figure 2: The contribution map. Domain areas of influence and contribution of each of the products developed in this research (ST-SA, ST-FLEX-D, ST-IB).

3.2 Research objective 1 methodology

In this section, I develop and implement ST-IB, an adaptive-bandwidth kernel density estimation approach, which addresses spatially and temporally inhomogeneous background populations. I do so within the framework of space-time cube (Nakaya and Yano 2010), which contains two planar spatial dimensions (x, y), whereas the vertical dimension reflects the temporal component, time (t).

3.2.1 Case-side adaptive bandwidth kernel density estimator

3.2.1.1 Kernel density estimation

Kernel density estimation (KDE) is a popular technique to visualize patterns of spatial point events (a.k.a data points). It imposes a regular grid of points (a.k.a pixels) on the study area, and evaluates density for each of them, based on surrounding point events: A circular window, defined by its radius h_s (*s* stands for "spatial", forming a distinction to "temporal" later on), is centered on a data point. Grid points that fall within the circle receive a contribution (a.k.a. weight) to their density value, which is determined by the kernel function and their distance to the center. I repeat the process for each data point and hence, create a density surface based on the observed point data.

$$\hat{f}(x,y) = \frac{1}{nh_s^2} \sum_{i=1}^{n} k_s\left(\frac{d_{i(x,y)}}{h_s}\right) \quad (1)$$

Equation 1 shows how density estimates for a given grid point $\hat{f}(x,y)$ is calculated. *n* is the number of data points within the study area, h_s is the radius of the circular window. k_s is the kernel function, which characterizes the contribution of each data point *i* as a function of its distance to the grid point $d_{i(x,y)}$. Popular kernel functions are Epanechnikov, Gaussian, or Biweight (Bowman and Azzalini 1997).

3.2.1.2 Kernel density estimation with inhomogeneous background

Computing density as the distance-weighted number of points per unit area is not realistic for many geographical applications (Bithell 1990). For instance, when mapping disease risk, we are interested in the number of (disease) cases (a.k.a data points) per unit population-at-risk, which might exhibit an uneven distribution in space and time. Depending on the phenomenon under study, the population-at-risk may include all population, or only certain strata. It may be a sample or the full population-at-risk and it is also referred to as *background population*, or simply *background*. As a result, an area of elevated disease risk identified by kernel density estimation might merely reflect a large local background population (Bithell 2000). A generic method to deal with inhomogeneous background population is to compute risk ($\hat{r}$) by dividing density of cases (c) by the density of the background population (p), shown in Equation 2 (Davies and Hazelton 2010).

$$\hat{r}(x, y) = \frac{c}{p} \tag{2}$$

When considering an inhomogeneous background population, the distinction between site-side and case-side kernel density estimation becomes important (Shi 2010). From an epidemiological perspective, it makes more sense to assess population around the disease case instead of around a site (grid point), which is what the case-side method does. It is hard to justify that the contribution of a case to disease risk at the site is based on the population around the site, which is the idea behind the site-side method (Shi 2010). Figure 3 illustrates the distinction between site-side and case-side methods, where

the circles represent kernels that both have equal bandwidth. The two cases (red dots) both have equal distances to the two sites (black dots). Using the site-side method, both cases are of equal importance to the sites, but they are equally more important to the left site because of the low population (blue dots) within the left circle, compared to the high population in the right circle (indicated by the thick and thin blue arrows). This results in higher disease risk for the left site. The Case-side method assigns higher importance to the upper case because of low population within its circle. It is key to recognize that each case contributes equal weight to both sites. Hence, the resulting disease risk is the same at both sites. I conclude that site-side and case-side kernel methods might produce different results for a given scenario.

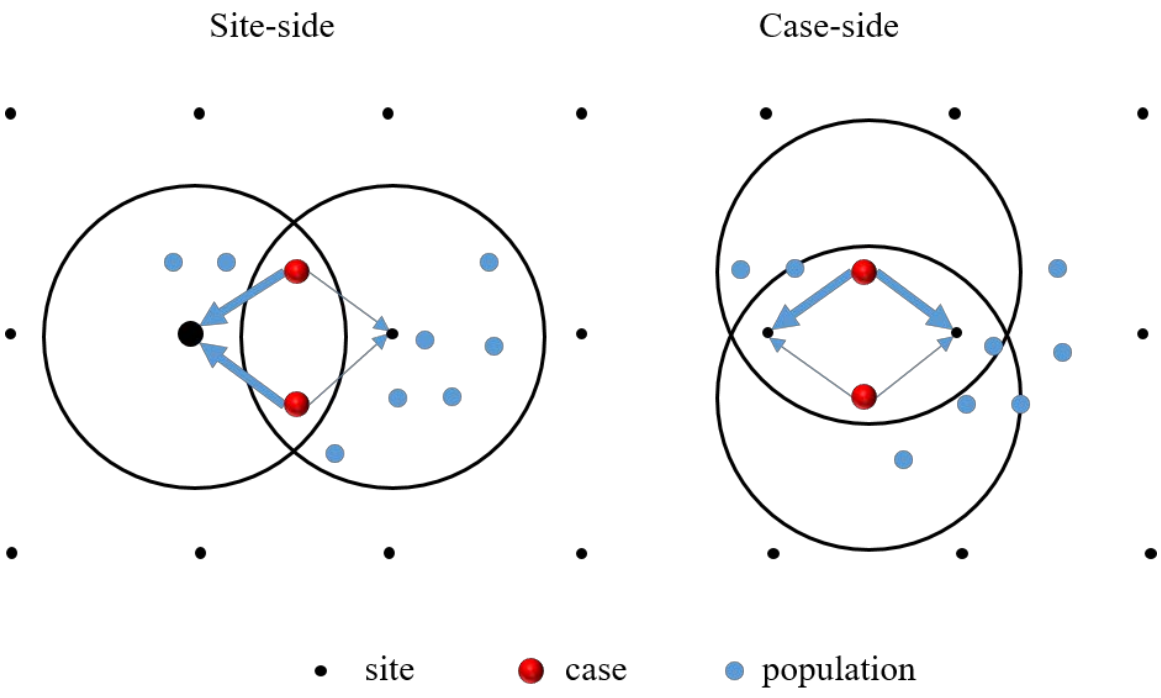


Figure 3: Distinction between Site-side and Case-side kernel density estimation with inhomogeneous background population.

Now that I established the importance of distinguishing site-side and case-side kernel density estimation over inhomogeneous backgrounds, I shift focus on the difference between fixed and adaptive kernels. Fixed kernels have constant bandwidth whereas adaptive kernels allow the bandwidth to adapt to local conditions. A kernel that adapts to the background population is useful to establish constant population support (constant *p* in Equation 2), rather than constant areal support, which is the case with fixed bandwidth kernels. Alternatively, it makes sense to adapt the bandwidth to the surrounding cases when computing risk of communicable disease, such as dengue fever, the local case density is more informative than the population density. I achieve this by imposing the kernel on a disease case and start increasing the bandwidth until a specified (population or disease case) support is reached (Figure 4). Note that as the kernel expands, the case in its center will expand the range of its contribution to disease risk. In other words, as the circle grows outwards, seeking for support, more sites will receive contribution from the disease case in its center.

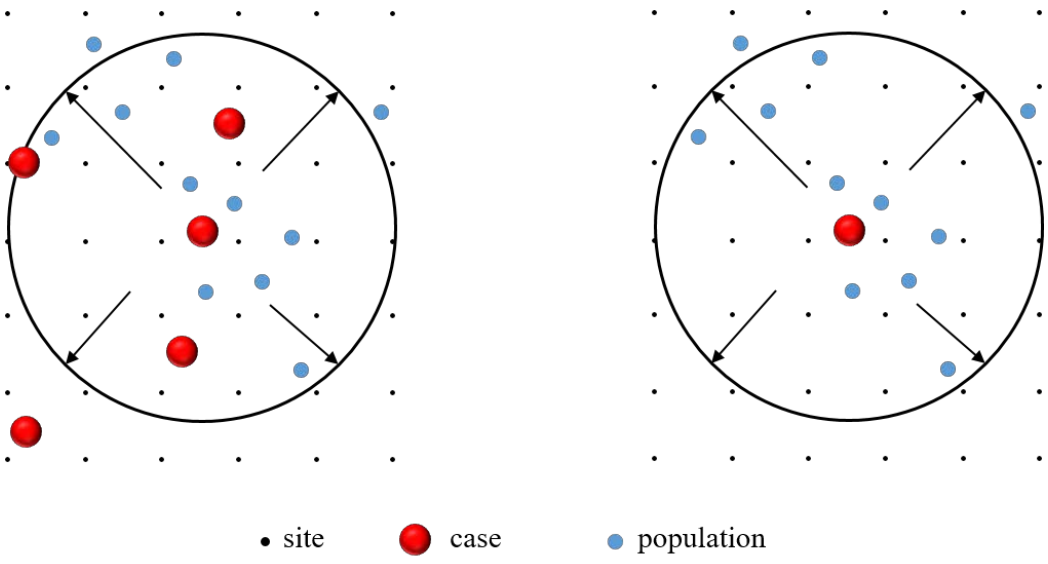


Figure 4: Adaptive bandwidth kernel with inhomogeneous background. Right: Kernel adapts to population. Left: Kernel adapts to cases.

Shi (2010) proposes the case-side adaptive bandwidth kernel density estimator (Equation 3):

$$\hat{r}_{CA}(x,y) = \sum_{i=1}^{n} k_s\left(\frac{d_{i,(x,y)}}{h_s[p(x_i,y_i)]}\right) \quad (3)$$

Where the bandwidth h_s is a function of the local population density at the location (x_i, y_i) of case *i*. This method results in disease risk values that are defensible in health studies, while being more statistically comparable (Carlos et al. 2010; Shi 2010; Shi and Wang 2015).

3.2.1.3 Space-time kernel density estimation

So far, I completely ignored the temporal dimension in the discussion. Many geographic studies employ time-flattening: collapsing the temporal dimension into one single 2D map, which represents the entire study period (Bach et al. 2016). Other approaches discretize time into a number of time slices, which can be displayed as small multiples (Boyandin, Bertini, and Lalanne 2012). However, none of these approaches represent time as a real continuous dimension, which is necessary for depicting spatiotemporal patterns of point events. Space-time kernel density estimation (STKDE) is a temporal extension of KDE, used for identifying spatiotemporal patterns of spatial point events with a timestamp. Different from the concepts of space-time paths and space-time prism to analyze individual movement patterns (Kwan 2000; Kwan 2004; Miller 1991), STKDE considers each point as an independent observation. We can visualize the density

estimates within the space-time cube framework using two spatial (x, y) and a temporal dimension (t). STKDE outputs a regular 3D grid of points (a.k.a. voxels) that hold a density estimate based on the surrounding point data (Delmelle et al. 2014; Brunsdon, Corcoran, and Higgs 2007). The space-time kernel density is estimated by Equation 4:

$$\hat{f}(x,y,t) = \frac{1}{nh_s^2 h_t} \sum_i k_s\left(\frac{d_{i,(x,y)}}{h_s}\right) k_t\left(\frac{d_{i,(t)}}{h_t}\right) \quad (4)$$

Density $\hat{f}(x,y,t)$ of each voxel *s* with coordinates *(x,y,t)* is estimated based on neighboring data points *i*. Each point located within neighborhood of *s* is weighted using the spatial and temporal kernel functions, k_s and k_t, respectively (the closer the data point, the higher the weight. The spatial and temporal distances between voxel and data point are given by $d_{i,(x,y)}$ and $d_{i,(t)}$ respectively. Normalization or scaling of all dimensions to a common range of values would allow for computing true 3D kernel density, which would simplify the procedure if STKDE to Equation (1). Defining a weight, or conversion factor between the spatial and temporal dimension would achieve the same thing. However, this would make the result of our analysis dependent on the distribution and extent of the data or the subjective choice of weighing time versus space, and hence, hurt its comparability and general applicability.

3.2.1.4 Space-time kernel density estimation with inhomogeneous background

Since we are able to compute disease risk for spatially inhomogeneous background populations, we could use STKDE to generate maps of disease risk over time, assuming a temporally homogeneous (static) background. However, we find

ourselves in the age of migration (Castles, De Haas, and Miller 2013), where people move their residential location for many reasons: forced migration due to climate change (Martin 2001), conflicts (Mitchell 2011), or to find labor (Münz 2007). Cities experience waves of urbanization (Meentemeyer et al. 2013), suburbanization (Lang and Simmons 2001), re-urbanization and counter-urbanization (Champion 2001). Hence, the temporally homogeneous background population assumption might no longer hold true. In addition, population data are becoming available at finer spatial and temporal resolutions, and given the current technological advancement, it is foreseeable that this development will continue (Bhaduri et al. 2007), calling for an extension of the current kernel methods for computing disease risk to address temporally inhomogeneous background populations.

Equation 5 denotes a case-side adaptive-bandwidth space-time kernel density estimator for spatially and temporally inhomogeneous background (ST-IB):

$$\hat{r}_{ST-IB}(x, y, t) = \sum_i k_s \left(\frac{d_{i,(x,y)}}{h_s[p(x_i, y_i)]}\right) k_t \left(\frac{d_{i,(t)}}{h_t[p(t_i)]}\right) \quad (5)$$

Here the spatial- and temporal bandwidths h_s and h_t, respectively, are a function of the local population density $p(x_i, y_i)$, $p(t_i)$ at space-time location (x_i, y_i, t_i) of case i. The population is assessed within a half ball moving through 3D space, which means that only the population present before the disease case contributes to the population adjustment. However, as seen in Section 3.2.1.2, it may make sense to adapt the bandwidth to the surrounding cases $c(x_i, y_i)$, $c(t_i)$ instead of the underlying population. For

instance, when computing risk of communicable disease, the local case density is more informative than the population density (Equation 6).

$$\hat{r}_{ST-IB}(x,y,t) = \sum_i k_s\left(\frac{d_{i,(x,y)}}{h_s[c(x_i,y_i)]}\right) k_t\left(\frac{d_{i,(t)}}{h_t[c(t_i)]}\right) \quad (6)$$

Both estimators suffer from the multiway problem: There are multiple ways to achieve the specified support (of either neighboring cases or population). When searching for support, we could either extend the spatial bandwidth (Figure 5a), or the temporal bandwidth (Figure 5b), or both.

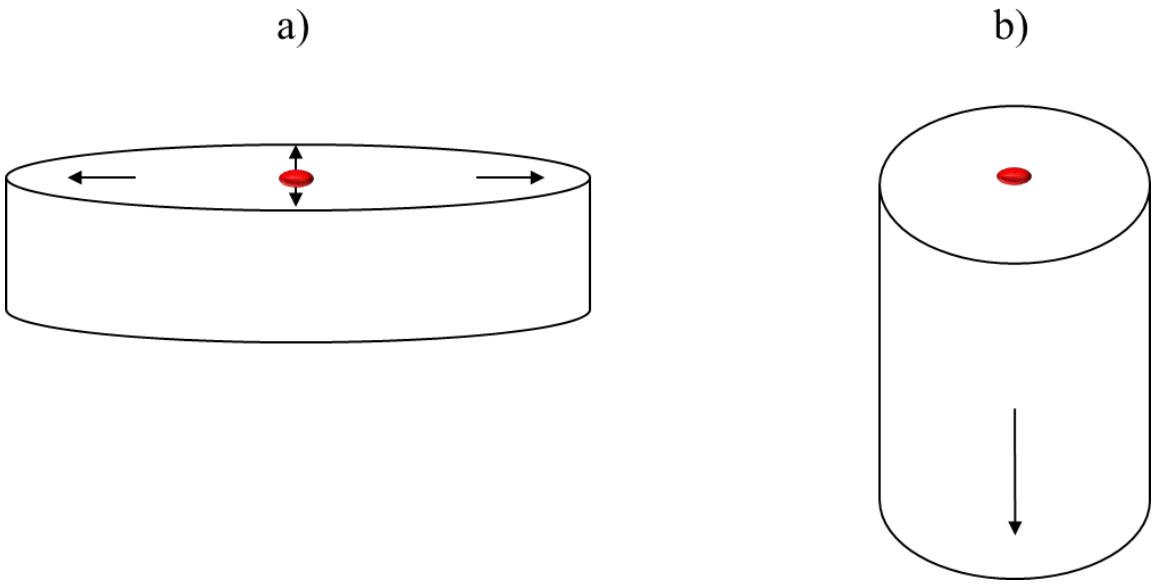
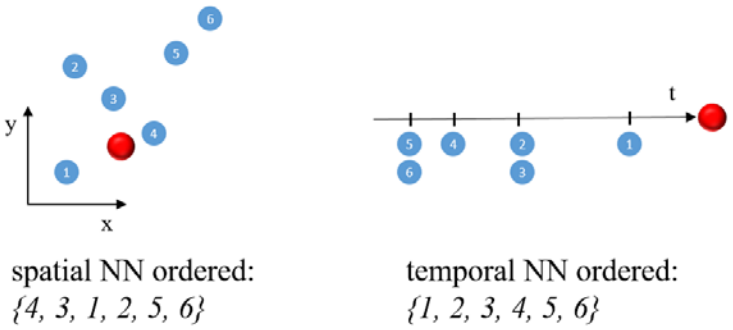


Figure 5: The multiway problem. a) extending the spatial bandwidth, b) extending the temporal bandwidth.

Clearly, we need to overcome the orthogonal relationship between space and time (Nakaya 2013), and unify them into the same space. I chose to use the *k*-nearest neighbors (*kNN*) method to deal with the issue (Jacquez 1996). Figure 6 illustrates the

process: Because we cannot compute a meaningful Euclidean distance between objects using the spatial and temporal dimensions, I generate two ordered sets for each disease case: 1) the spatial *k*-nearest neighbors and 2) the temporal *k*-nearest neighbors of the case. I then increase *k* and compute the cardinality *card()* of the intersection between the two sets. I keep increasing *k* until *card()* equals the specified support. I then compute the spatial and temporal bandwidths h_s, h_t, respectively, as the spatial and temporal distance of the farthest point in the intersection set to the case.



$$1NN: \{4\} \cap \{1\} = \{\emptyset\} \rightarrow card(\{\emptyset\}) = 0$$
$$2NN: \{4,3\} \cap \{1,2\} = \{\emptyset\} \rightarrow card(\{\emptyset\}) = 0$$
$$3NN: \{4,3,1\} \cap \{1,2,3\} = \{1,3\} \rightarrow card(\{1,3\}) = 2$$
$$4NN: \{4,3,1,2\} \cap \{1,2,3,4\} = \{1,3,2,4\} \rightarrow card(\{1,3,2,4\}) = 4$$
$$5NN: \{4,3,1,2,5\} \cap \{1,2,3,4,5\} = \{1,3,2,4,5\} \rightarrow card(\{1,3,2,4,5\}) = 5$$
$$6NN: \{4,3,1,2,5,6\} \cap \{1,2,3,4,5,6\} = \{1,3,2,4,5,6\} \rightarrow card(\{1,3,2,4,5,6\}) = 6$$

Figure 6: Spatiotemporal nearest neighbors.

Using this procedure, I unify the spatial and temporal dimensions, enabling search for support of adaptive-bandwidth kernel density estimation for spatially and temporally inhomogeneous backgrounds. I do so by discretizing the continuous spatial and temporal

dimensions into sets of nearest (case or population) neighbors of disease cases. Therefore, I solve the multiway problem for the kernel density estimator in Equations 5 and 6.

3.2.3 Case data

I illustrate our implementation of ST-IB using a spatiotemporal explicit set of dengue fever cases in the city of Cali, Colombia (Figure 7), for the years 2010-2011. Cali, the third largest metropolitan area in the country with a total population of around 2.3 million and a population density of 4,140/km^2 in 2013, is located in the southwest of Colombia (Cali 2014). Cali experiences two rainy seasons: the first from April to July and the second from September to December. Located at approximately 1,000 m above sea level, it has an average temperature of 26°C and an average precipitation of 1,000 mm over most of the metropolitan area (Cali 2014). The city, as most colonial cities in Latin America, grew from its central core, following the city spine, and towards the periphery. Peripheral neighborhoods are typically characterized by high density and low income since they have been the result of squatter settlements and poor urban planning (Restrepo 2011).

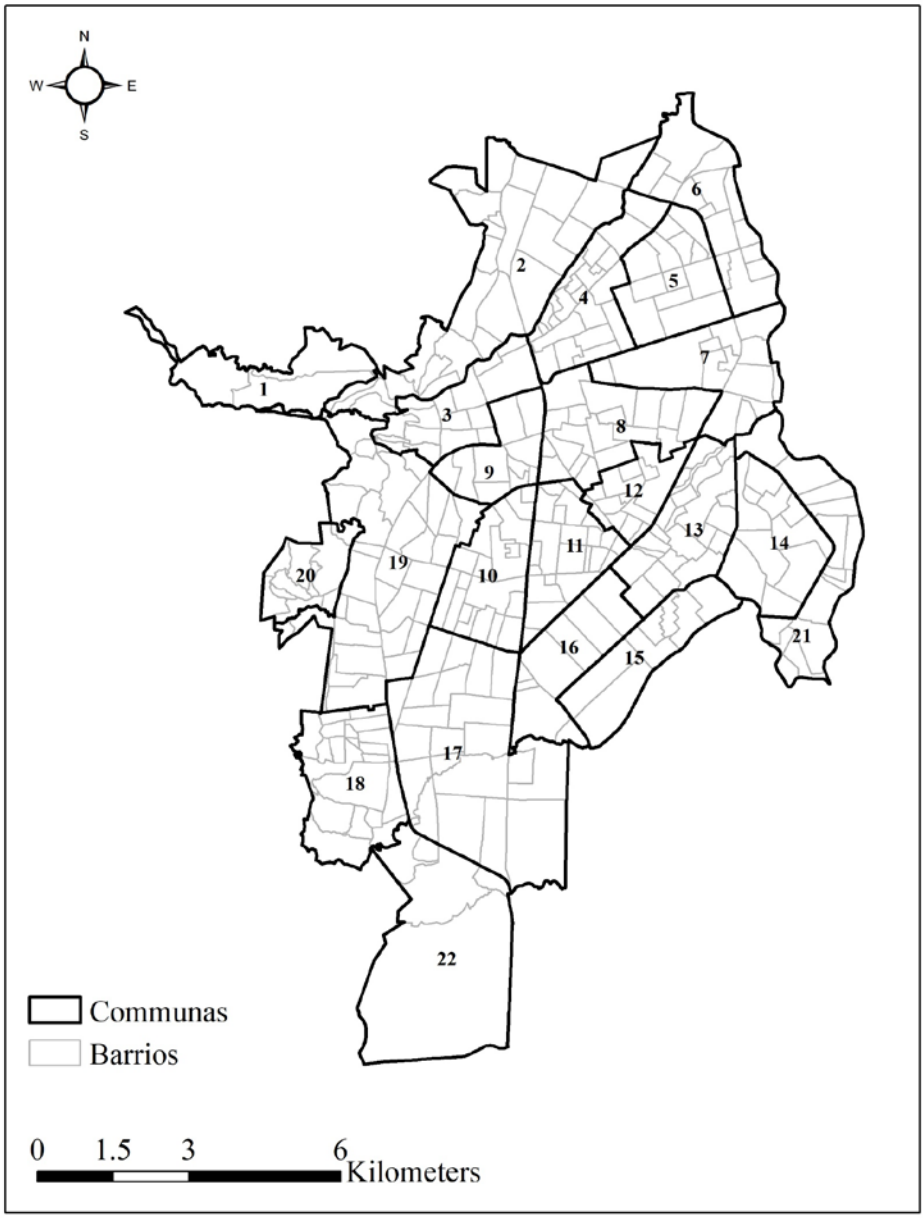


Figure 7: The city of Cali, Colombia.

I use a dataset of dengue fever cases within the city of Cali in this study. The data is extracted from the “Sistema de Vigilancia en Salud Pública (SIVIGILA)” (English: Public Health Surveillance System) for the city of Cali for the years 2010 and 2011. The

SIVIGILA system has as a main responsibility to observe and analyze health events with the objective of planning, follow up, and evaluation of public health practices (Colombia 2017). Reported cases of dengue fever are entered into the system daily. Each case includes personal information about the patient such as their home address and when they were diagnosed. A total of 11,056 cases were geocoded to the closest intersection to guarantee a level of privacy, for both years. There were 9,606 cases in 2010 and 1,562 in 2011. The difference in the number of cases is explained by the fact that 2010 was identified as an epidemic year (Varela, Aristizabal, and Rojas 2010).

3.2.4 Population data

Population information at fine spatial and temporal scales might be available in different formats and conceptualizations. Apart from census data, scientists have used tweets as a proxy for population (Malleson and Andresen 2015), trajectories of individuals created through retrospective activity diaries (Chen et al. 2011), migration history datasets (Shaw, Yu, and Bombom 2008), or multi-dimensional dasymetric modelling approaches (Bhaduri et al. 2007). These population datasets are profoundly different: In social media, a tweet is a point in space and time and although it can suggest that the person was at location *(x,y)* at time *t*, we have no information regarding his/her whereabouts at any other time than *t*. On the other hand, activity diaries allow for knowing a person's whereabouts at any *t*, which permits drawing the space-time path. Besides availability, the following principle should guide the choice of population data: The level of detail of the population data should match the level of detail of the case data.

For instance, if we use patient residential locations, possibly geomasked for privacy, activity diaries population information might be too detailed.

I obtained annual population counts on the neighborhood level of Cali from 2000 to 2014, as well as the geometries as ESRI shapefile (Cali 2014). The city is administratively divided into 22 communes covering 120.9 km^2, and composed of 340 neighborhoods (see Figure 7). A commune is a grouping of neighborhoods based on homogenous demographic and socioeconomic characteristics. Neighborhoods are classified using a stratification system composed of six classes, one being the lowest and six the highest. The strata are developed by evaluating the type of housing, urban environment and context. However, communes and years are coarse spatial and temporal units, but we need population data at fine resolution to achieve meaningful disease estimates. I draw inspiration from studies that disaggregate disease data, for instance, Jacquez and Jacquez (1999), who introduce a procedures to disaggregate areal data by assigning random locations within the area. This approach is extended to the restricted and controlled Monte Carlo (RCMC) process by Shi (2009). Luo, McLafferty and Wang (2010) use Monte Carlo simulation to disaggregate cancer data and identify risk factors using a hierarchical logistic model. I do so in a similar way for the Cali population data using the following steps (see Figure 8):

1. I distribute the population of the first year (2010) within each of the 340 neighborhoods as random points. Every neighborhood receives a number of points equal to its total population of 2010.

2. Using the yearly neighborhood population counts 2010 – 2011, I compute annual population change (increase/decrease) for each neighborhood. I scale down the annual change to daily values, assuming linear change. Hence, I compute the daily change of population by dividing the annual change by 52.
3. For each day within 2010 – 2011 (which amounts to 730 timesteps), I replicate the population points from the previous day. For each neighborhood, I add as many random points as there is population increase. In case of population decline, I randomly delete as many existing points as there is population decline.

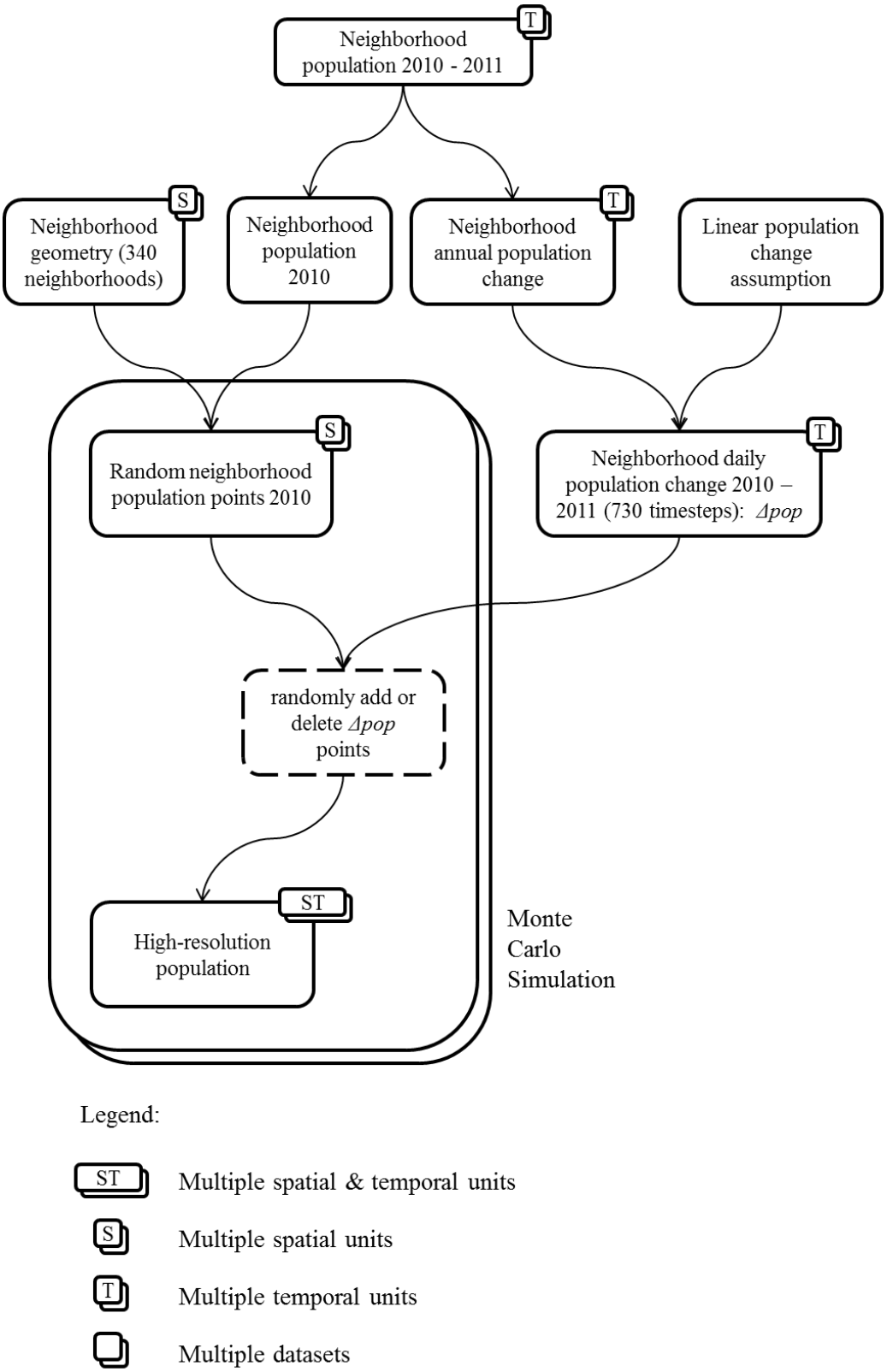


Figure 8: Flowchart for population disaggregation.

Hence, I create a structured spatiotemporal grid of population points. Due to the random elements of the procedure, I use Monte Carlo approach to obtain 100 simulated population datasets. This results in variance of the resulting disease risk estimates, which is a measure of uncertainty resulting from aggregation of population data.

3.2.5 Research objective 1 analysis

I assess whether ST-IB, which challenges the temporally homogeneous background assumption of Shi's method (Shi 2010), actually improves the ability to detect high disease risk areas/periods. Therefore, I need to define what a good performance is: Disease risk maps allow for identifying areas/periods that have increased disease risk, provide measures of risk differences between regions, and enable targeting areas for intervention, or for allocating resources (Charras-Garrido et al. 2012). The general public might benefit from such maps by knowing which areas to avoid. Therefore, disease risk within the identified and clearly delineated areas/periods (a.k.a. "clusters") should be substantially higher than outside. In a case-control scenario, risk is the ratio of cases to controls. The ratio between risk inside and outside of the identified areas is called *odds ratio* (Bland and Altman 2000). A high odds ratio means the disease cluster has been delineated well, as the ratio between cases and controls inside the cluster is much higher than outside. I employ odds ratios to compare ST-IB with other kernel methods using the same data: Method *A* produces a better disease risk estimates and cluster delineations than method *B* if it identifies risk areas/periods that produce a higher odds ratio. Hence, *A* performs better than *B* if it produces higher odds ratios. Odds ratios

are to some extent similar to *likelihood ratios* of the SaTScan method (Kulldorff 1997). However, while SaTScan uses expected disease counts, I use the underlying population directly. Also, while SaTScan uses likelihood ratios to determine the most likely cluster, I use them to compare two different methods. In addition to using odds ratios to measure cluster strength, I use Monte Carlo simulation to measure their statistical significance. Hence, method *A* is only better than *B* if it produces higher odds ratios that are statistically significant.

3.2.5.1 Uncertainty from population simulation

Before comparing ST-IB and any other method, I quantify uncertainty that stems from the population simulation: I perform kernel density estimation for spatially and temporally inhomogeneous backgrounds using the case dataset (Section 3.2.3) and each of the 99 simulated population datasets (Section 3.2.4). Therefore, I compute 99 grids of density estimates, which allows us to extract upper and lower envelopes as the maximum and minimum value for each site (a.k.a. grid point, voxel). The spatiotemporal resolution of the grid is 100m / 1 day. To quantify the uncertainty, 1) I compute a histogram of the differences between upper and lower envelope for each site as a measure of how far they spread, and 2) visualize them within the space-time cube. If the histogram indicates that the difference is mostly small, I conclude that uncertainty from population simulation is small. In addition, the depiction within the space-time cube enables for detecting patterns of where and when the results may be subject to high uncertainty.

3.2.5.2 Benefit of considering time and cluster significance

Since I quantified uncertainty from population simulation (Section 3.2.5.1), I acknowledge that any subsequent result is subject to variability according to the range of values we observed in the previous step. Hence, I can pick one of the 100 population simulations, and am now able to produce random simulations of the disease cases for significance testing.

In a this step, I am interested in comparing ST-IB with with it's purely spatial equivalent S-IB (kernel density estimation for spatially inhomogeneous backgrounds), to answer the question whether including the temporal dimension in our analysis achieves more realistic disease risk estimates. S-IB is inspired by the work of (Shi 2010), and corresponds to Equation 3 in section 3.2.1.2 with the exception that the bandwidth does not adapt to the local population, but to neighboring disease cases. Hence, S-IB centers the kernel (circle) on each disease case and increases the radius until it contains a chosen number of nearest disease neighbors.

Such comparison is feasible and meaningful, as I run both methods with the same data, but collapse the temporal dimension for S-IB. Hence, I pick the population from week 26, the most central time step during our study period. While ST-IB produces a 3D grid of risk estimates, S-IB produces a 2D grid. S-IB Therefore, I are not comparing apples to oranges because both methods are fed with the same data. I pick the n^{th}-percentile of sites that exhibit the highest risk and label the corresponding areas as disease cluster. I compute the odds ratios of disease risk inside versus outside the cluster

area, and do so for each combination of percentiles [90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 99.9, 99.99] and disease case support values [5, 10, 15, 20, 25, 30, 35, 40, 45] (see Section 3.2.1.2), resulting in a total of 108 treatments. This allows us to draw odds ratio surfaces in dependence of the parameter values, and to compute the difference between ST-IB and S-IB.

It is important to note that the "winner" of the comparison between ST-IB and S-IB is not better in a universal sense as it may merely result in different conclusions. Explicitly simulating space-time clusters may be the only way to determine whether method A is truly better than B. In addition, the reader should note that ST-IB refers to the kernel density estimator in Equation (6), and does not include the procedure of delineating clusters by picking the n^{th}-percentile highest disease risk sites and calculating odds ratios, as outlined above. This procedure merely serves for validation and comparison purposes.

In addition to knowing which method produces higher odds ratios, we may want to determine the statistical significance of the delineated clusters. With the goal of illustrating the utility of my approach, I pick one parameter combination (percentile: 95, support: 45) for significance testing, and I do so for ST-IB only, even though it would be feasible and interesting to do so for both algorithms and for all 108 treatments. I am using odds ratio as a statistic to measure the strength of a cluster, and report statistical significance of the odds ratios as p-values. I simulate 99 datasets by the following procedure: I take the observed dengue cases, which are a dataset of 11056 [x, y, t] tuples.

Maintaining the temporal intensity of the observed dengue cases, I randomize their spatial locations. In other words, I leave the t-value untouched and instead, randomize the x- and y-values for each observation. The randomization is restricted to the city limits of Cali, and I use the R-packages rgeos, spatstat, and maptools to perform the necessary point-in-polygon operation. The point-in-polygon operation provided by the R-packages creates a specified (=11056) number of random points within a given polygon, which corresponds to the city limits of Cali. Every location has an equal probability of "receiving" a point, therefore resulting in a random uniform scenario. In summary, our reference scenario is complete spatial randomness (CSR) with a temporal trend. For each of the 99 simulations, as well as for the observed dataset, I compute kernel density using ST-IB, delineate disease clusters using the parameter combination (percentile: 95, support: 45), and compute the corresponding odds ratios. The rank of the observed odds ratio among the simulated ones is considered as its p-value. For instance, if the observed odds ratio is among the top 5% of simulated odds ratios, the p-value is <= 0.05, which indicates strong evidence against the null hypothesis of CSR with a temporal trend. In addition to assessing statistical significance, I visualize significant clusters within the space-time cube using Voxler (Golden Software, Colorado), an interactive 3D modeling environment.

3.3 Research objective 2 methodology

In this section, I develop a method (ST-FLEX-D) to accelerate spatiotemporal analysis algorithms, such as space-time kernel density estimation (STKDE). I improve upon an existing methodology (Hohl, Delmelle, and Tang 2015), which consists of the

following two stages: 1) spatiotemporal domain decomposition of the input point set (ST-STATIC-D), 2) parallel computing of any spatiotemporal analysis that fits the domain decomposition strategy. I solve one of the key problems of ST-STATIC-D and develop a flexible splits heuristic to minimize domain replication for parallel processing of STKDE. I make the following key assumption: The spatiotemporal analysis in stage 2 uses a known, fixed bandwidth. Otherwise, the bandwidth has to be determined prior to the decomposition procedure. This also applies to point patterns that are characterized by anisotropy, which results in an elliptic base of the search cylinder at stage 2. It requires finding the equation of the ellipse using the quantification of anisotropy provided by the data (angle, length). Then, the maximum search distance in X, Y, and T directions can be found easily, which dictates the buffer distance during decomposition.

3.3.1 The existing method

ST-STATIC-D decomposes the spatiotemporal domain of a set of points for subsequent distribution of the resulting subdomains to processor queues for concurrent processing. It creates subdomains of similar computational intensity, which promotes equal workloads among CPUs. Computational intensity of spatiotemporal analysis algorithms may depend on 1) the number of data points within the subdomain, 2) the number of voxels, given by subdomain size, as voxels are structured within a regularly spaced 3D grid. Recursive spatiotemporal domain decomposition accounts for input data structure, which might exhibit a heavily clustered distribution. Recursion is a method where the solution to a problem depends on solutions to smaller instances of the same problem (Graham 1994).

The ST-STATIC-D decomposition algorithm works as follows (see Figure 9): First, I compute the minimum and maximum *x, y* and *t* values of the point set (a.k.a its spatiotemporal domain, or bounding box). Second, I bisect the domain midway of every of the three dimensions, resulting in 8 subdomains of equal size and cuboid shape. Third, I decompose each of the 8 cuboids recursively until either one of the two thresholds T_1 and T_2 are crossed for every subdomain I create. T_1 is the number of points within the subdomain, whereas T_2 is the proportion of the subdomain volume within the buffer volume (see next paragraph). Low thresholds yield a fine-grained decomposition and a reduced search space, as empty subdomains are discarded from further processing. A fine-grained decomposition is desirable because a high number of small tasks rather than a low number of large tasks will likely balance workloads among processors. On the other hand, low thresholds increase recursion depth, causing the program to terminate if the maximum depth is surpassed.

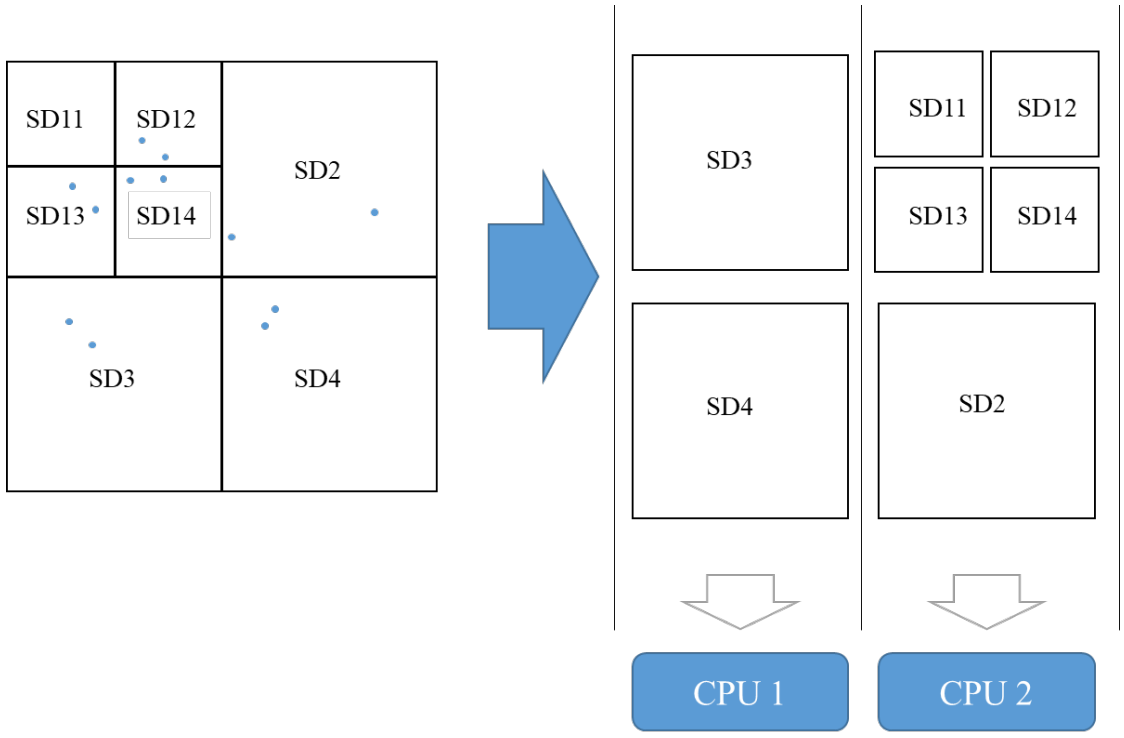


Figure 9: Octree-based recursive spatiotemporal domain decomposition. Note that this is a 2D representation of a 3D problem. Same concepts apply in 3D.

The decomposition procedure introduces new (subdomain-) boundaries, which may result in edge effects that degrade the results of the subsequent spatiotemporal analysis, due to spatial and temporal neighborhood search. I cope with the issue by creating buffers around each subdomain of distance equal to the spatial and temporal search radius (Figure 10). Therefore, a point located within a buffer $b_h(sd_1)$ of subdomain sd_1 is assigned to sd_1, even if it does not fall inside it. As subdomains share borders with others, their buffer zones overlap, causing multiple replications of data points (up to 8), which are assigned to different subdomains, and therefore, data redundancy.

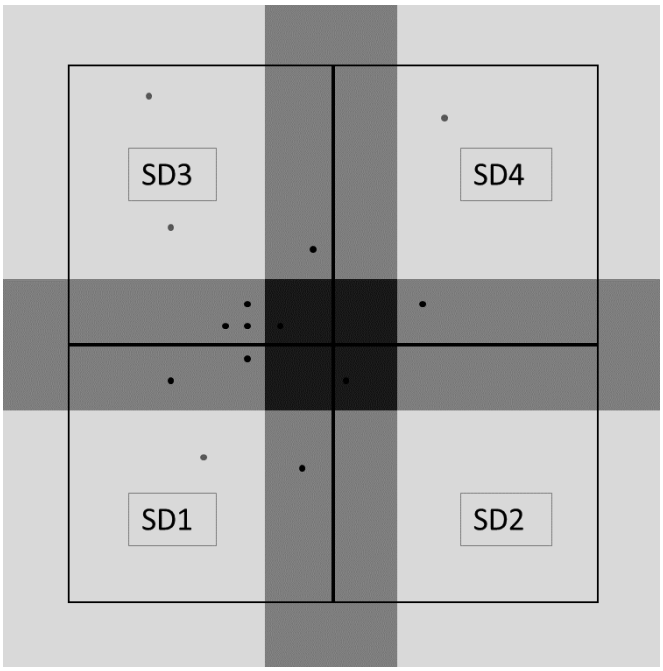


Figure 10: Buffer implementation for handling edge effects. Note that this is a 2D representation of a 3D problem. Same concepts apply in 3D.

For each subdomain SD_i that results from the decomposition, I quantify computational intensity *CI* (Wang 2008) as a function of the product of 1) the number of data points $N_p(SD_i)$ and 2) the number of voxels $N_v(SD_i)$ that are contained in the corresponding subdomain (Equation 7).

$$CI(SD_i) = f\left(N_p(SD_i) * N_v(SD_i)\right) \tag{7}$$

To ensure balanced workloads, I distribute the sequence of subdomains (SD_1, SD_2, …, SD_i), resulting from 3D to 1D mapping by space filling curve (Bader 2012), to the processors by equalizing the cumulative *CI*. Therefore, processors receive variable numbers of subdomains but similar workloads. The importance of accurately quantifying

CI for our endeavor cannot be stressed enough, as failure of doing so results in failure of balancing workloads.

3.3.2 The ST-FLEX-D approach

Here, I present ST-FLEX-D_base, ST-FLEX-D_uneven, and ST-FLEX-D_alternate, three improved versions of ST-STATIC-D, which focus on minimizing the redundancy caused by replication of points within the subdomain buffers. The improvement is based on the observation that ST-STATIC-D bisects domains at the midpoint in each dimension.

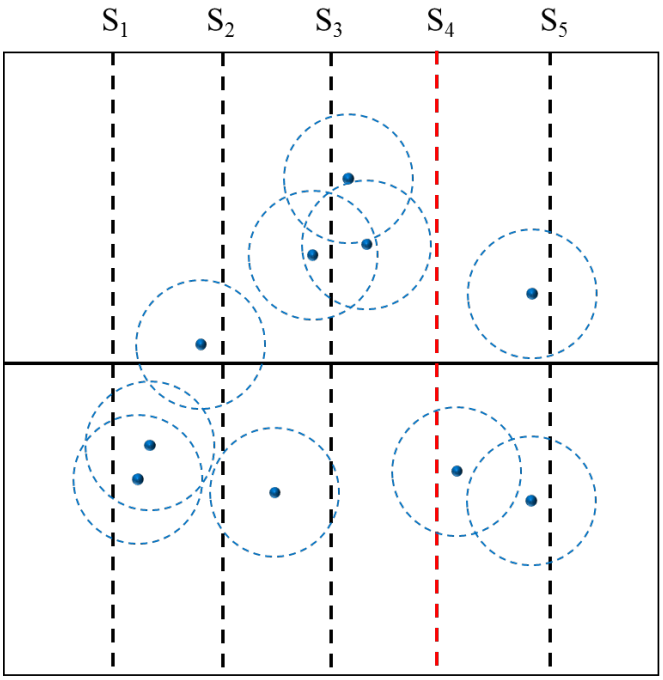
3.3.2.1 ST_FLEX_D_base

For the ST-FLEX-D_base implementation, I relax the midway bisection dictate and allow for multiple candidate split positions. I define candidate split positions by regular increments along each axis (see Figures 11 – 13), and pick one split for bisection according to the following rules:

- Rule 1 – Pick the *minimum replication split*: the candidate split that results in the lowest number of replicated points. A point is replicated if the splitting line/plane cuts the circle/cylinder centered on a data point that has radius equal to the kernel bandwidth(s) (Figure 11).
- Rule 2 – In case of a tie (two candidate splits have the lowest number of replicated points), pick the *most even split*: split that bisects the set of points most evenly among splits in consideration (Figure 12).

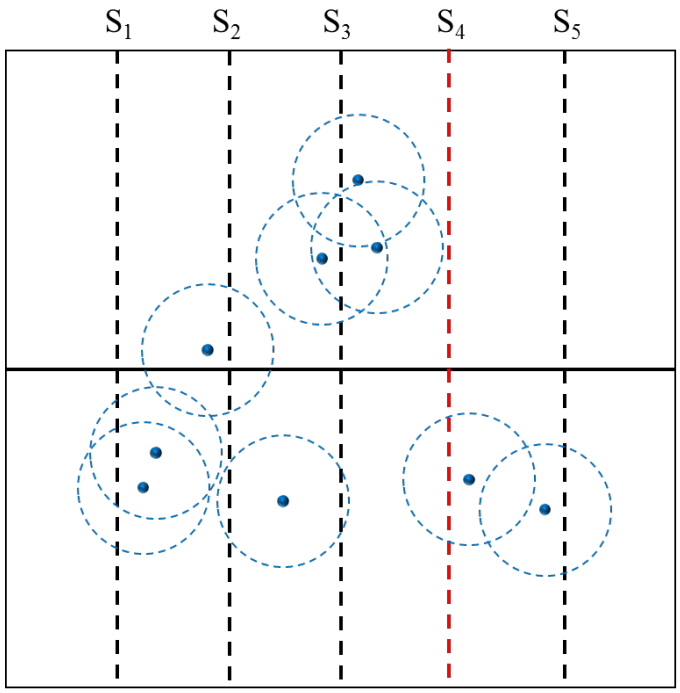
- Rule 3 – If still tied, pick the *most central split*: candidate split that is most central among splits in consideration (Figure 13).

Figure 14 shows an illustrative example of the entire process. Note that all illustrations related to ST_FLEX_D are 2D, whereas the actual application is 3D (2D + time). It is much easier to explain the concepts on paper medium in 2D, but the procedure is the same for the added temporal dimension. First, I focus on the x-axis, where the minimum number of cuts is tied between two candidate splits: SX_1 and SX_5. Hence, I apply Rule 2 and pick SX_5 because its evenness (9/1) is higher than SX_1 (0/10). I then focus on the y-axis, where the minimum number of cuts is again tied between SY_1 and SY_5. I pick SY_1 by applying Rule 2 (evenness of 1/9 over evenness of 10/0).



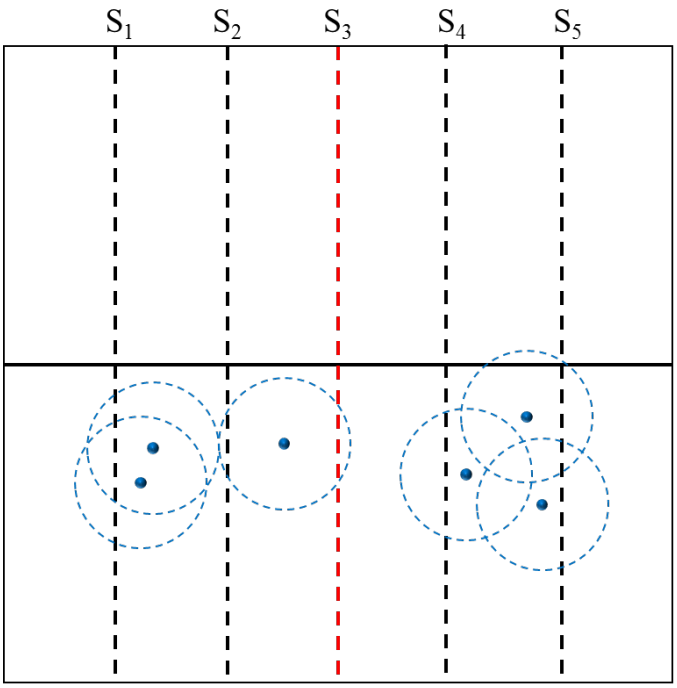
Candidate Split	S_1	S_2	S_3	**S_4**	S_5
Number of cuts	2	2	4	2	1

Figure 11: Rule 1 of ST-FLEX-D. In this example, I choose S_4 because it minimizes the number of circles cut by the bisection line.



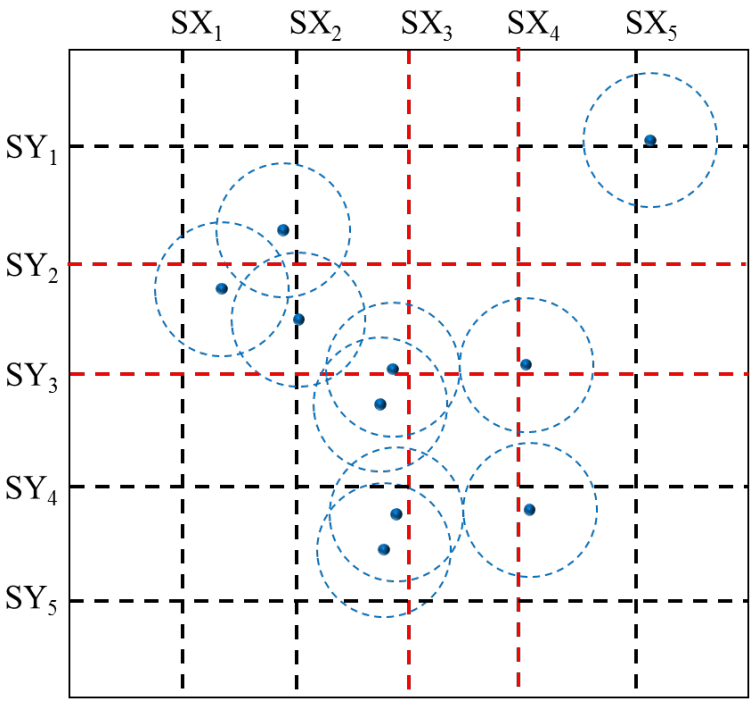
Candidate Split	S_1	S_2	S_3	**S_4**	S_5
Number of cuts	2	2	4	1	1
Split evenness	0/9	3/6	5/4	7/2	9/0

Figure 12: Rule 2 of ST-FLEX-D. Here, the minimum number of cut circles ties between S_4 and S_5 (Rule 1). Hence, I pick S_4, which bisects the set of points more evenly.



Candidate Split	S_1	S_2	**S_3**	S_4	S_5
Number of cuts	2	1	1	1	2
Split evenness	0/6	2/4	3/3	3/3	6/0
Centrality	1	2	3	2	1

Figure 13: Rule 3 of ST-FLEX-D. Here, the minimum number of cut circles ties between candidate splits S_2, S_3 and S_4 (Rule 1). Split evenness ties between S_3 and S_4 (Rule 2). I pick S_3, which is more central than S_4.



Candidate Split	SX_1	SX_2	SX_3	SX_4	**SX_5**
Number of cuts	1	2	4	2	1
Split evenness	0/10	2/8	7/3	7/3	9/1
Centrality	1	2	3	2	1

Candidate Split	SY_1	SY_2	SY_3	SY_4	**SY_5**
Number of cuts	1	3	4	3	1
Split evenness	1/9	2/8	4/6	7/3	10/0
Centrality	1	2	3	2	1

Figure 14: Example of ST-FLEX-D.

3.3.2.2 ST_FLEX_D_uneven

The implementation of ST_FLEX_D_base brings the danger of picking “bad splits”. Bad splits do not advance the decomposition procedure at all and the issue arises by picking the outermost split (SX_1, SX_5, SY_1, SY_5) when points are distributed more

centrally within the domain. While bad splits may cut zero circles (and therefore are chosen by our procedure), all points potentially lie on the same side of the split. This is does not advance the decomposition and is therefore not desired. ST_FLEX_D_uneven attempts to solve the problem by candidate split locations that are not evenly distributed along the axis, do not cover the entire range of values within that dimension, but congregate around the midway split (Figure 15). This regime maintains flexible split locations while reducing the odds of choosing bad splits. Rules 1-3 of ST_FLEX_D_base for picking the bisection split still apply for ST_FLEX_D_uneven.

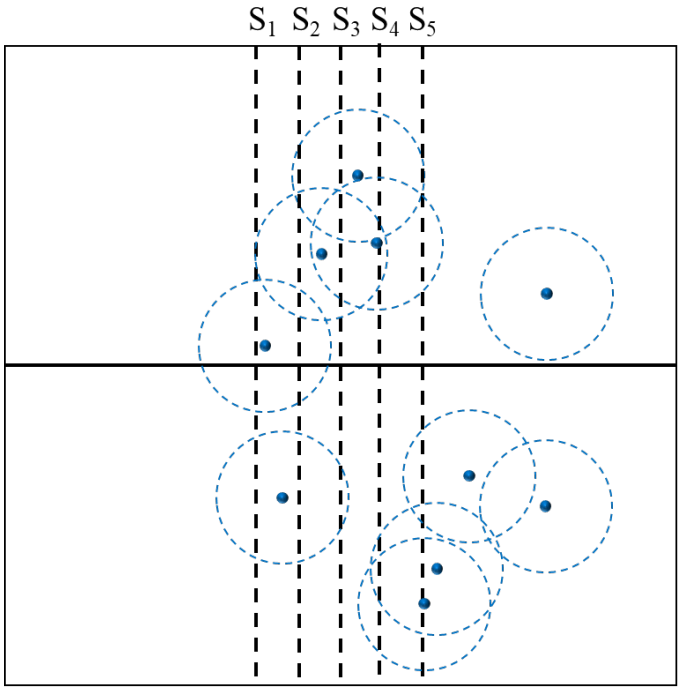


Figure 15: Uneven candidate splits.

3.3.2.3 ST_FLEX_D_alternate

So far, I assumed that our domain is perfectly square, hence bisecting all dimensions simultaneously made sense. However, when using real data, such as disease cases, we may face less compact (elongated) rectangular domains, and the decomposition procedure may even decrease the compactness (increase elongation) for the subdomains it produces. For instance, the domain of the dengue fever dataset (see Section 3.2.3) is elongated in N-S direction. Subdomain compactness may have effect on the efficiency of the decomposition and the subsequent parallel processing of spatiotemporal statistics because compact subdomains can foster workload balance and reduce the overall computational intensity of the applications. ST_STATIC_D, ST_FLEX_D_base or ST_FLEX_D_uneven may result in subdomains that are highly elongated, especially if I choose bad candidate splits from the beginning for (more so for ST_FLEX_D_base and less so for ST_FLEX_D_uneven).

With ST_FLEX_D_alternate, I address the issue by dropping the requirement of bisecting all dimensions simultaneously. At each node of the tree, I chose one dimension for bisection (instead of bisecting all three of them). Hence, I no longer perform octree decomposition, as our procedure results in a binary tree. At each node, I pick the dimension that exhibits the largest range (i.e. difference between minimum and maximum value), i.e. the “longest” dimension for bisection. That way, I balance the ranges for each dimension and hence, achieve more compact subdomains (see Figure 16). Rules 1-3 of ST_FLEX_D_base for picking the bisection split still apply for ST_FLEX_D_alternate.

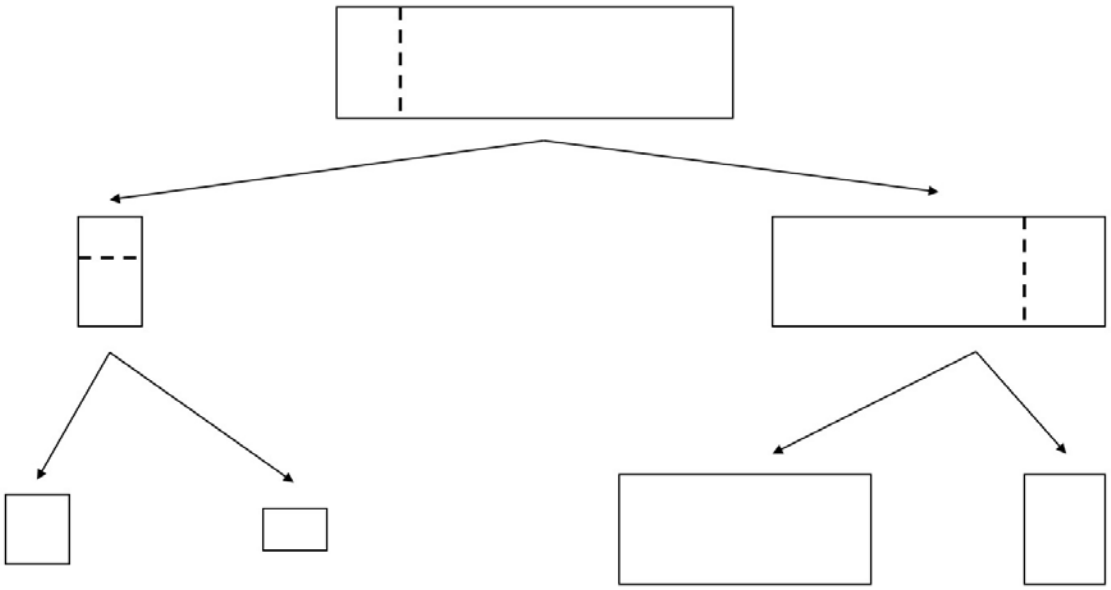


Figure 16: ST-FLEX-D-alternate.

3.3.3 Research objective 2 analysis

I compare the performance of ST_STATIC_D with all three implementations of ST-FLEX-D (ST_FLEX_D_base, ST_FLEX_D_uneven, ST_FLEX_D_alternate) using the following metrics:

1) execution time of decomposition

2) total number of cut circles

3) average leaf node depth

4) average leaf node size

The *execution time of decomposition* is the total amount of time the computer needs to decompose the dataset, disregarding I/O. The *total number of cut circles* is equal to the number of replicated data points that result from the decomposition. It is a measure of the redundancy within the decomposition procedure and our goal is to minimize it. The decomposition procedure is inherently hierarchical, where a domain splits into multiple subdomains. Therefore, it is common to illustrate the procedure as a tree, where the root is the initial domain to be decomposed, and the subdomains resulting from the first split are children nodes linked to the root node (see Figure 17 for illustration and example, Figure 16 is another example). Since the recursion does not go equally deep in all of its branches, I compute the *average leaf node depth*, which measures how many times on average I split the initial domain to form a particular subdomain. The *average leaf node size* is just the number of data points it contains and measures the granularity of the decomposition. The largest leaf node ultimately determines the parallel performance as it is the largest chunk of workload.

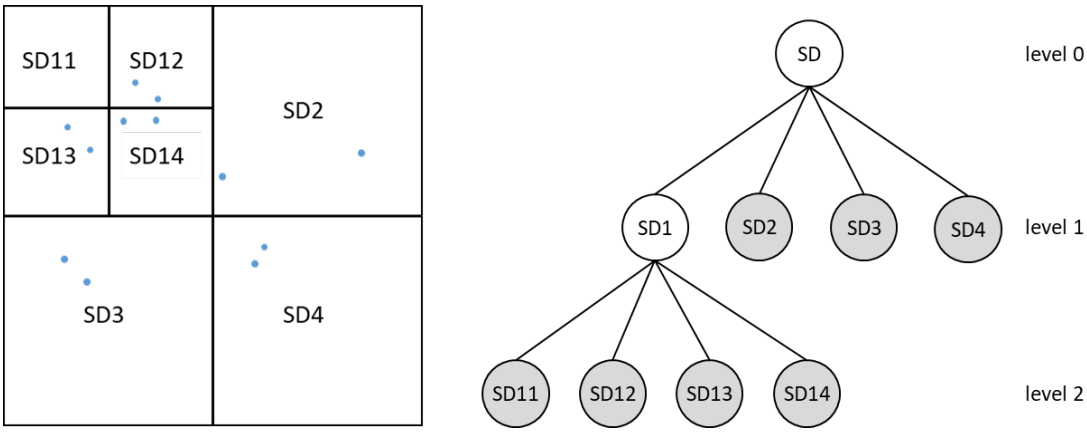


Figure 17: Domain decomposition. Spatial depiction (left), tree (right). Leaf nodes of the tree are denoted by grey color. The average leaf node depth is (1+1+1+2+2+2+2)/7=1.57.

The following parameters determine the outcomes of our implementations of spatiotemporal domain decomposition: 1) the maximum number of points per subdomain (threshold T_1, see paragraph 3.2.1), 2) the buffer ratio (threshold T_2 paragraph 3.2.1), 3) spatial and temporal bandwidths, 4) output grid resolution, 5) number of data points. I set parameters 1 - 5 to the values given in Table 1, where all values are kept steady but values for spatial and temporal bandwidth vary (spatial: 200m – 2500m in steps of 100m; temporal: 1 day – 14 days in steps of 1 day). Hence, I have 336 different parameter configurations (treatments) for which I compute the metrics introduced above for all implementations (ST_STATIC_D, ST_FLEX_D_base, ST_FLEX_D_uneven, ST_FLEX_D_alternate).

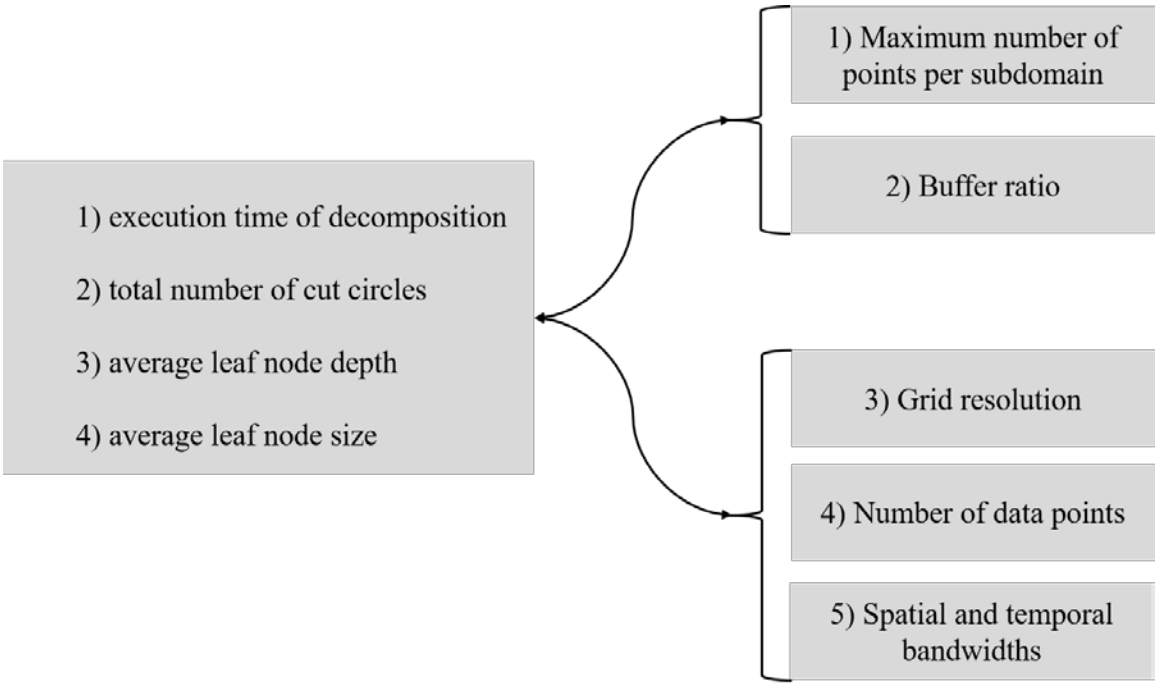


Figure 18: Performance metrics and their influencing factors.

Table 1: Parameter values for ST-STATIC-D and ST-FLEX-D.

Parameter	Name	Value
1	Maximum number of points per subdomain	50
2	Buffer ratio	0.01
3	Grid resolution	100m, 1day
4	Number of data points	11056
5	Spatial and temporal bandwidths	[200m, 300m, 400m, 500m, 600m, 700m, 800m, 900m, 1000m, 1100m, 1200m, 1300m, 1400m, 1500m, 1600m, 1700m, 1800m, 1900m, 2000m, 2100m, 2200m, 2300m, 2400m, 2500m], [1 day, 2 days, 3 days, 4 days, 5 days, 6 days, 7 days, 8 days, 9 days, 10 days, 11 days, 12 days, 13 days, 14 days]

3.4 Research objective 3 methodology

In this section, I conduct computational sensitivity analysis (ST-SA) on ST-STATIC-D. I used a single set of parameters to assess the computational performance of ST-STATIC-D in section 3.3. Hence, I now focus on assessing the sensitivity of computational performance to various parameter values in a systematic way.

3.4.1 Global sensitivity analysis

Spatiotemporally explicit data, as well as statistical models and their outputs are often nonlinear and nonmonotonic. Therefore, analyzing the sensitivity to input parameters requires approaches that handle nonlinear, nonmonotonic, and spatiotemporally explicit characteristics while not relying on model structure (Saltelli et al. 2008; Saltelli et al. 2004). Sobol's approach is a variance-based approach for global sensitivity analysis, and provides support for analyzing spatiotemporal statistical models (Lilburne and Tarantola 2009). Essentially, Sobol's approach is based on the decomposition of variance in model output (V; see Equation 8) into first-order effects due to single input parameters (V_i) and higher order effects contributed by interactions among input parameters (e.g., V_{ij}, V_{ijk})

$$V = \sum_i V_i + \sum_{i<j} V_{ij} + \sum_{i<j<k} V_{ijk} + V_{12\ldots m} \tag{8}$$

where V_i is the first-order effect of input parameter i on output variance, m the number of input parameters, and V_{ij} the second-order effect of interactions between input i and j on output variance. V_{ijk} is the third-order effect of interactions among input variables i, j, and k. $V_{12\ldots m}$ is the highest order effect on output variance, explained by interactions among all input variables. For each model input parameter i, I compute first- and total-order sensitivity indexes based on the decomposition of output variance (see Equation 9):

$$\begin{cases} S_i = V_i/V \\ S_{Ti} = 1 - V_{\sim i}/V \end{cases} \tag{9}$$

where S_i is the first-order sensitivity index and S_{Ti} is the total-order sensitivity index for input parameter *i*. $V_{\sim i}$ is the conditional variance explained by all input parameters except parameter *i*. Jointly using first- and total-order sensitivity indexes allows us to identify the singular and compounded effects of model inputs on variance in model outputs. Therefore, Sobol's approach is a good choice for sensitivity analysis and is preferred over local approaches or regression, which often have no support for high-order effects. The numerical derivation of Sobol's sensitivity indexes requires Monte Carlo integration (Saltelli et al. 2004; Saltelli et al. 2008) and a random sampling of multidimensional parameter space. I use quasi-random sequences, where the determination of a random number depends on previously generated numbers (Gentle 2006), to sample our model input parameters (Saltelli et al. 2010). Quasi-random sequences have better performance for multidimensional sampling than pseudo-random sequences because of their low discrepancy and fast convergence (Sobol 1967; Niederreiter 1978). There exist alternative approaches to generate quasi-random sequences (see Gentle 2003). I used Sobol's quasi-random sequences (see Sobol 1967) in this study.

Based on Sobol's method, the number of required samples *N* is computed by Equation 10:

$$N = (2 * k + 2) * N_{mc} \tag{10}$$

where N_{mc} is the number of Monto Carlo runs, and *k* is the number of input parameters. For relatively simple linear models, N_{mc} is set to a value within the range of [20, 100]. For

sophisticated nonlinear models, N_{mc} is usually larger than 100, typically suggested within the range of [100, 500] or higher. I generate the first $2*N_{mc}$ samples using Sobol's quasi-random method (Sobol 1967): For each input parameter sample, the initial quasi-random number q is chosen between 0-1. I compute the final number f based on the given range of the parameter (Equation 11):

$$f = p_{min} + (p_{max} - p_{min}) * q \tag{11}$$

where p_{min} is the minimum and p_{max} the maximum parameter value as specified and justified by the analyst. Depending on the data type of the input parameter, f might have to be rounded to the next integer. Samples are organized in matrices A and B, of dimension $N_{mc} \times k$, hence A contains samples 0 to N_{mc} and B contains samples $N_{mc} + 1$ to $2*N_{mc}$. Then, I generate another $2*N_{mc}$ samples for parameter i, which are denoted as matrices C_i and D_i. The relationship between C_i, D_i and A, B is illustrated in Figure 19 (adapted from Lilburne and Tarantola 2009): All the values in D_i are the same as in A, except those for the ith parameter, which are the same as in B. All the values in C_i are the same as in B, except those for the ith parameter, which are the same as in A. Then I have all the $k*2*N_{mc}$ samples for all parameters. Adding the initial $2*N_{mc}$ samples, we totally have N samples (see Equation 10).

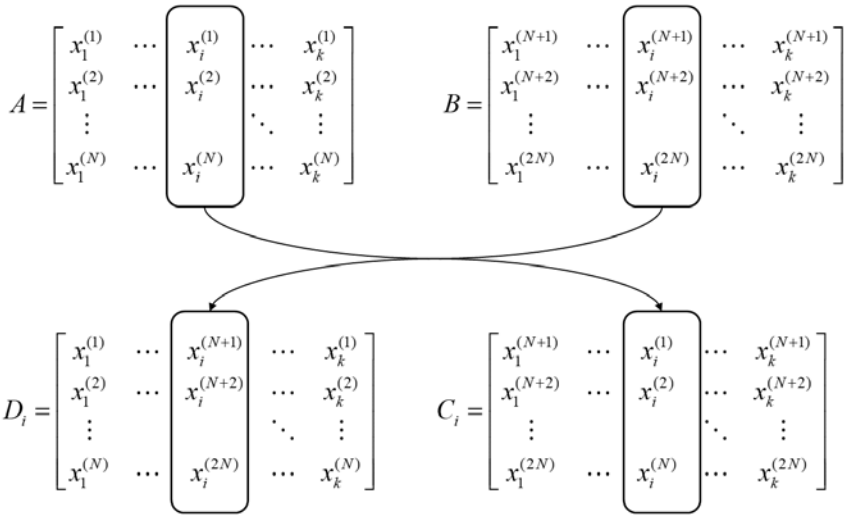


Figure 19: Matrices A, B, C_i and D_i.

To obtain first- and total-order sensitivity indexes (S_i and S_{Ti}, respectively), I evaluate our model for the parameter values in A, B, C_i and D_i, which results in four vectors of model output values (Y_A, Y_B, Y_{Ci}, Y_{Di}), each of dimension $N \times 1$. Then, I use the following estimators to compute the sensitivity indexes (see Lilburne and Tarantola 2009):

$$S_i = \frac{\frac{1}{N}\sum_{j=1}^{N} Y_A(j)Y_{Ci}(j) - \left(\frac{1}{N}\sum_{j=1}^{N} Y_A(j)\right)\left(\frac{1}{N}\sum_{j=1}^{N} Y_B(j)\right)}{\frac{1}{N}\sum_{j=1}^{N} Y_A^2(j) - \left(\frac{1}{N}\sum_{j=1}^{N} Y_A(j)\right)\left(\frac{1}{N}\sum_{j=1}^{N} Y_B(j)\right)} \tag{12}$$

$$S_{Ti} = \frac{\frac{1}{N}\sum_{j=1}^{N} Y_A(j)Y_{Di}(j) - \left(\frac{1}{N}\sum_{j=1}^{N} Y_A(j)\right)\left(\frac{1}{N}\sum_{j=1}^{N} Y_B(j)\right)}{\frac{1}{N}\sum_{j=1}^{N} Y_A^2(j) - \left(\frac{1}{N}\sum_{j=1}^{N} Y_A(j)\right)\left(\frac{1}{N}\sum_{j=1}^{N} Y_B(j)\right)} \tag{13}$$

The estimators presented in Equations 12 and 13 are part of a group of eight estimators each for first- and total-order sensitivity indexes, which have been developed by Saltelli (2002), and Tarantola et al. (2006). Computing the average of all eight estimators leads to better accuracy than only using one of them (Lilburne and Tarantola 2009). S_i indicates the average potential reduction of output variance if a given model input parameter *i* could be fixed, regardless of interactions. By the same token, $S^c_{i_1,i_2,\ldots,i_s}$ indicates the average potential reduction of output variance if we fixed model input parameters i_1, i_2, …, i_s. S_{Ti} is greater or equal to S_i if model input parameter *i* is not involved in any interaction. The difference $S_{Ti} - S_i$ allows for estimating the degree by which parameter *i* is involved in interactions with other input parameters. $S_{Ti} = 0$ means that *i* has no influence on output variance. $\sum_i S_i$ is 1 for additive models and less than 1 for non-additive models. The difference $1 - \sum_i S_i$ indicates the presence of interactions in the model. The sum $\sum_i S_{Ti} = 1$ means that the model is perfectly additive, $\sum_i S_{Ti} > 1$ if not. Negative values of S_i and S_{Ti} are possible, however, they are usually close to zero and hence, can be set to zero. Increasing the number of Monte Carlo runs (N_{mc}) might reduce the occurrence of negative indexes (Saltelli et al. 2004; Saltelli et al. 2008).

3.4.2 Research objective 3 analysis

Here, I conduct computational sensitivity analysis (ST-SA) on the computational model that consists of the two-stage procedure of: 1) spatiotemporal domain decomposition of a set of points (ST-STATIC-D), 2) parallel computing of STKDE. I use the dengue fever dataset (Section 3.2.3) to compute STKDE of dengue fever cases in Cali, Colombia. From a sensitivity analysis perspective, I input the dengue fever data and

parameters into our computational model, which outputs density estimates and computational performance metrics (Figure 20). ST-SA aims for assessing the sensitivity of variance in computational performance due to uncertainty of input parameters. I are aware that the data, its size and distribution are additional important sources of variance in the computational performance metrics but focus on the uncertainty that stems from parameters of the two-stage procedure (ST-STATIC-D, STKDE) for now. ST-SA does not assess sensitivity for standard model outputs, such as regression coefficients, or a prediction, or as is the case here, density estimates resulting from STKDE. ST-SA solely focuses on analyzing variance in computational performance of a model.

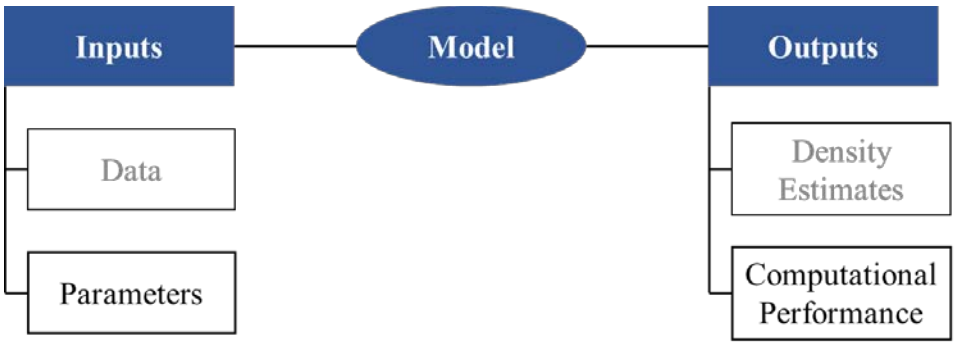


Figure 20: Inputs, Model and Outputs.

Our computational model has six parameters (Table 2): 1) Spatial bandwidth, 2) temporal bandwidth, 3) spatial voxel resolution, 4) temporal voxel resolution, 5) the maximum number of points per subdomain threshold (T_1 in section 3.2.1), 6) the buffer ratio threshold (T_2 in section 3.2.1). The parameter ranges given in Table 2 are based on domain knowledge and computational feasibility. For instance, the minimum spatial and temporal voxel resolutions are informed by the positional accuracy of the dengue fever data. On the output side, I focus on execution time and speedup of parallel STKDE the

second step of the two-stage procedure outlined above. Hence, I analyze the sensitivity of the variance in execution time and speedup to the six input parameters. I do so for multiple treatments, for which I vary the number of concurrent processors used from 1 (sequential scenario, only for execution time), to 10 to 100 in increments of 10 (equals 11 separate treatments for execution time, 10 for speedup). I choose 700 Monte Carlo runs which, together with the number of parameters, results in (2*6+2)*700 = 9,800 samples (see Equation 11). Hence, I need to evaluate the model 9,800 times, which is a massive computational burden.

Apart from Sobol's method, I use multivariate linear regression to quantify the effects of model parameter values on model outputs. I contrast and compare the two approaches and keep in mind that the linear model I use cannot account for interaction effects among factors. I use Box-Cox transform of variables to stabilize variance and transform non-normal predictors into normal shape (Box and Cox 1964). I develop a separate model for each treatment outlined above and report significance values, as well as coefficient of determination (R^2).

Table 2: Input parameters and ranges.

PID	Parameter	Relevance	Range	Data type
1	Spatial bandwidth	Decomposition & STKDE	250 - 2500	float
2	Temporal bandwidth	Decomposition & STKDE	3 - 14	integer
3	Spatial voxel resolution	Decomposition & STKDE	50 - 500	float
4	Temporal voxel resolution	Decomposition & STKDE	1 - 14	integer
5	Maximum #points threshold, T_1	Decomposition	5 - 1000	integer
6	Buffer ratio threshold, T_2	Decomposition	0.000013 - 0.17	float

I run all computations on the Copperhead high-performance computing cluster at the University of North Carolina at Charlotte, which has 59 nodes connected through an infiniband network switch (Pfister 2001), and 708 CPUs that are dual Intel Xeon 2.93 GHz 6 core X5670 processors with 36 GBs of RAM. When varying the number of CPUs in several treatments, I varied the number of nodes, choosing one CPU per node to exclude overhead through memory usage by other jobs. Copperhead is a Linux-based cluster that runs TORQUE resource and queue managing software.

I employ Copperhead in a shared-nothing architecture, where each job is executed on one processor within a self-sufficient node, which has no single point of contention across the system. Therefore, I use following fixed task scheduling method: First, I compute the ideal target workload, which is the same for each concurrent processor, by dividing the cost of the entire computation by the number of processors. Second, I use

space filling curve (Bader 2012) to map the subdomains resulting from decomposition to a 1D sequence. Third, I assign subdomains from the sequence to the first processor until I reach the target load, and employ the same procedure for the remaining processors. Although the number of assigned subdomains may vary among processors, the workload is similar (Hohl, Delmelle, and Tang 2015; Hohl, Delmelle, et al. 2016). Note that dynamic task scheduling methods may result in substantially different performance (Casavant and Kuhl 1988).

There are three points to mention about choices we made for ST-SA: First, we used the “naïve” algorithm for STKDE with complexity O(n*m) (n = number of voxels, m = number of data points) for simplicity. Second, ST-SA means choosing experimentation over theory to establish a relationship between parameter values and uncertainty in computational performance. In other words, we ran computation and measured execution time rather than making inference using computational performance models based on complexity theory, a seemingly valid and efficient alternative to ST-SA. However, it is the cost and outcome of decomposition that to our best knowledge are not easily captured by a model, especially because they are dictated by two exit conditions (number of points threshold, buffer ratio threshold), as well as the distribution of the observed points. With that uncertainty in our workflow, predicting execution time (especially parallel execution time) of STKDE based on parameter values seems very hard. In addition, due to our static scheduling procedure, the effects of subdomains that are outliers in computational cost (extremely high cost) are hard to foresee, as they could be bigger than the target workload for each processor, especially at high levels of

parallelization. Third, in support of our second point, we argue that complexity theory may inadequately capture computational requirements of spatial analysis because the spatial characteristics of data and operations, which have profound effects on computational intensity, are not sufficiently represented in complexity theory (Wang 2008).

CHAPTER 4: RESULTS

4.1 Research objective 1 results

4.1.1 The uncertainty from population simulation

The 99 population simulations result in 99 density estimates for each site. Therefore, I compute the difference between maximum and minimum value for each site (upper and lower envelope) and plot their frequency within a histogram (Figure 21). The vast majority of differences lies within a range of 0.0 – 0.005 (first column), which is a very small deviation, considering a range of density values within 0.0 – 0. 27 (Figure 22). All differences are below 0.0037, which means that the envelopes are very close to each other. Therefore, the uncertainty from population simulation is rather small.

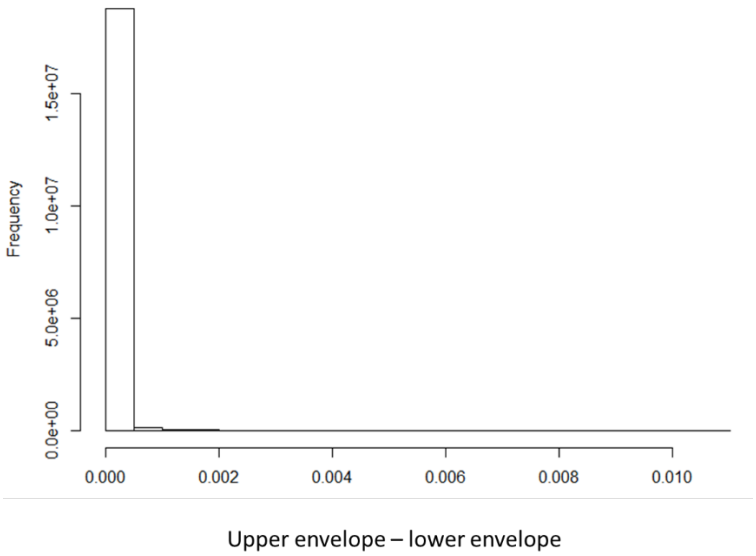


Figure 21: Histogram of differences between upper and lower envelope.

Visualizing both, the upper and lower envelopes within the space-time cube made no sense because the envelopes may not be distinguishable due to their small separating distance. Hence, I plotted the upper envelope within the space-time cube (Figure 22) to provide a spatiotemporal depiction of the density estimates. We can clearly see the two clusters of increased disease risk within the southwestern part of the city (Figure 2, 1 & 2), commensurate with the findings of Delmelle et. al (2014) and Hohl et al. (2016). These clusters are active from the very beginning of the study period and remain so for the first quarter of the study period. We also see another risk zone within the more central part of the city (Figure 2, 3) which exhibits elevated disease risk estimates for about the first half of the study period.

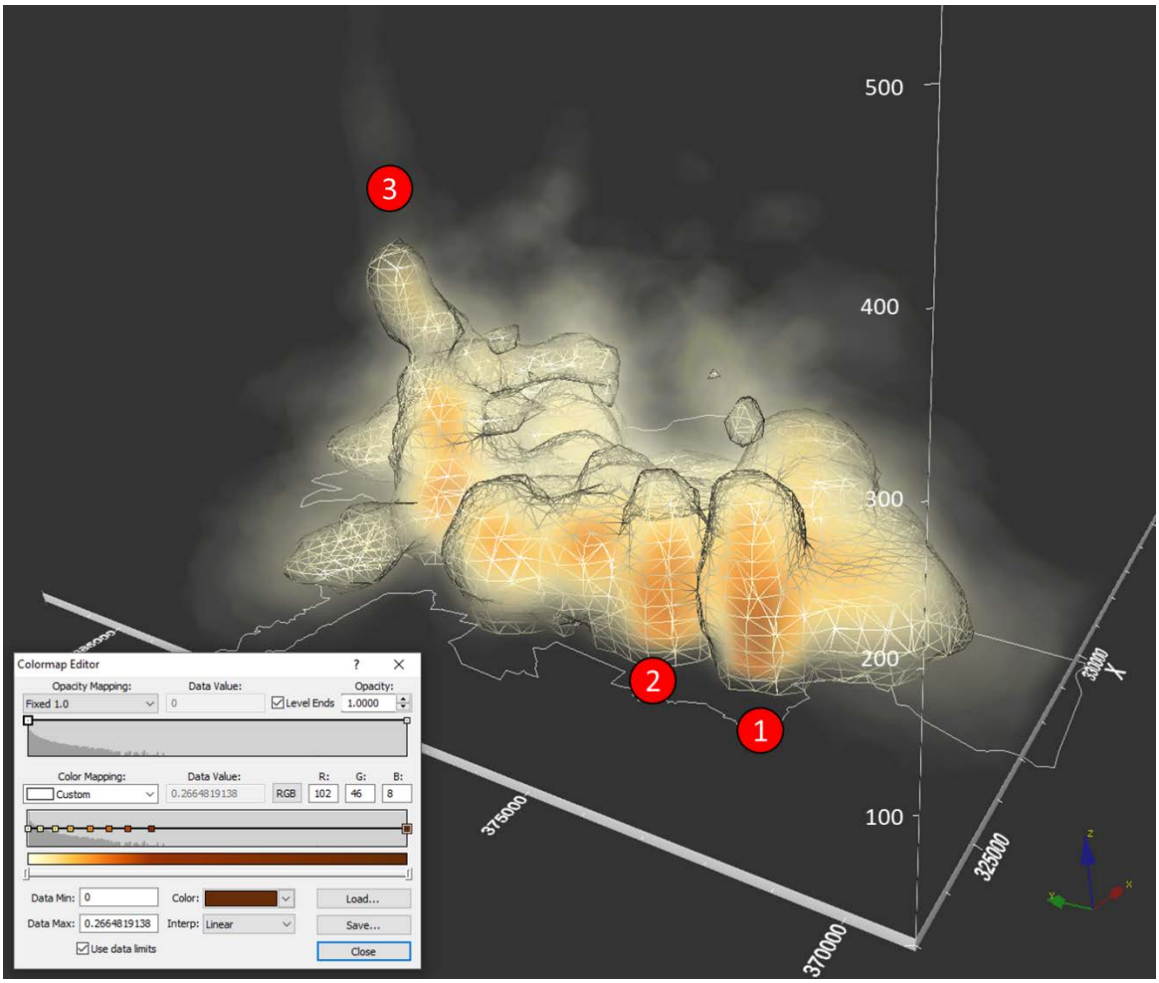


Figure 22: The upper simulation envelope (population simulation).

The spatiotemporal distribution of the difference between upper and lower envelope reveals the differences are relatively large where density estimates are large as well (Figure 23). Hence, the differences redraw the distribution of kernel density estimates. This result is expected and confirms that the uncertainty from population simulation is relatively small while following the spatiotemporal distribution of density estimates (Section 4.1.1).

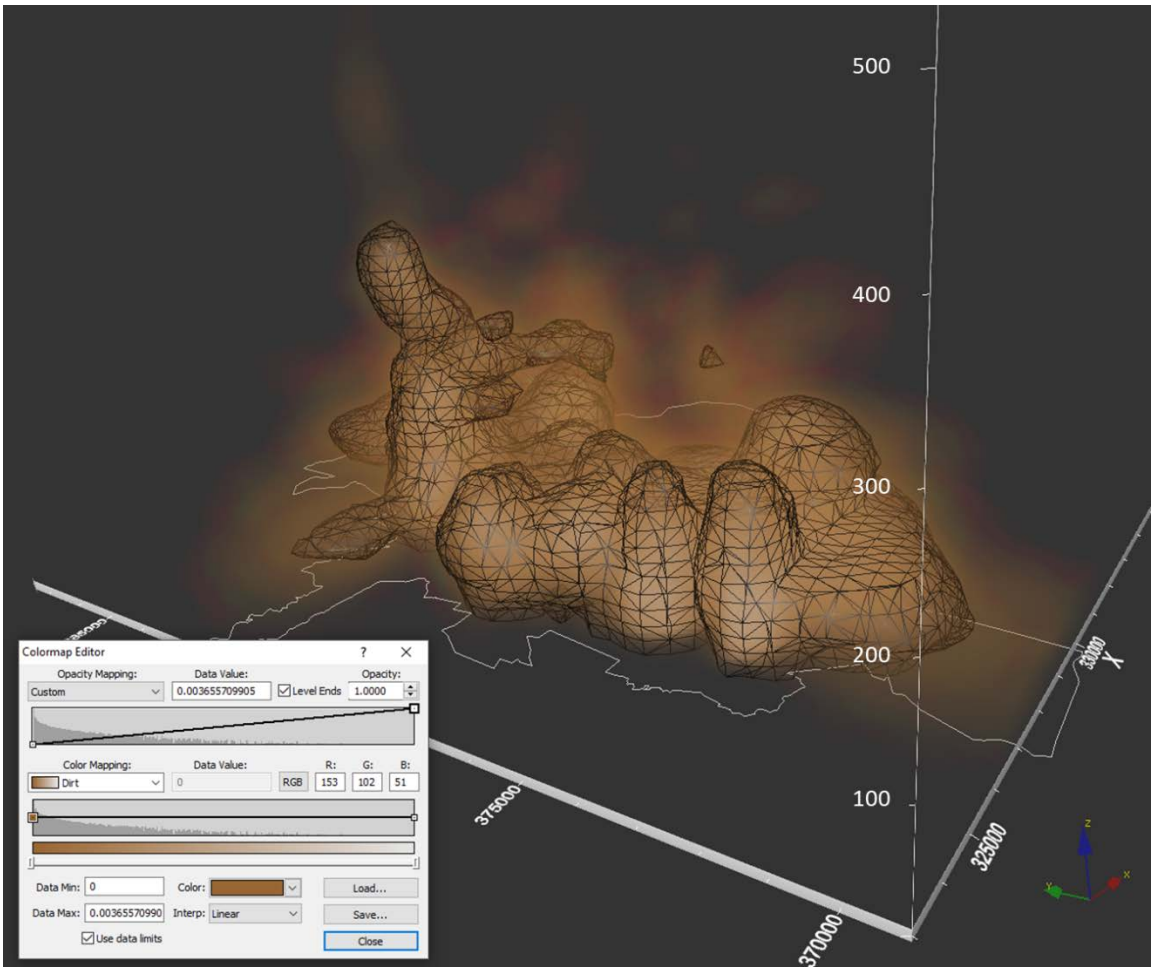


Figure 23: Difference between upper and lower simulation envelope.

4.1.2 Benefit of adding the temporal to our analysis

Comparing the S-IB and ST-IB reveals that there are parts of the parameter space assessed, where ST-IB performs better. Figure 24 shows the difference between odds ratios produced by S-IB and ST-IB. The blue region of the parameter space in Figure 24 shows a negative differences, i.e. ST-IB outperforms S-IB. The blue region stretches between a case support parameter value of 30 – 45 and from a percentile threshold of 90 – 97. In other words, if I use the 30 – 45 nearest case neighbors to determine kernel bandwidths, and if I delineate disease clusters by choosing voxels that exhibit densities

above the 90 – 97 percentile, ST-IB outperforms S-IB. S-IB outperforms ST-IB for the rest of the parameter space assessed. The difference in odds ratios are mostly small, but with percentile threshold values of 99 and above, they increase to a maximum of 58.38, indicating substantial superiority of S-IB. This superiority seems to decrease with an increasing case support, as the difference is around 0 – 19.31 at high values (35 - 45) of this parameter.

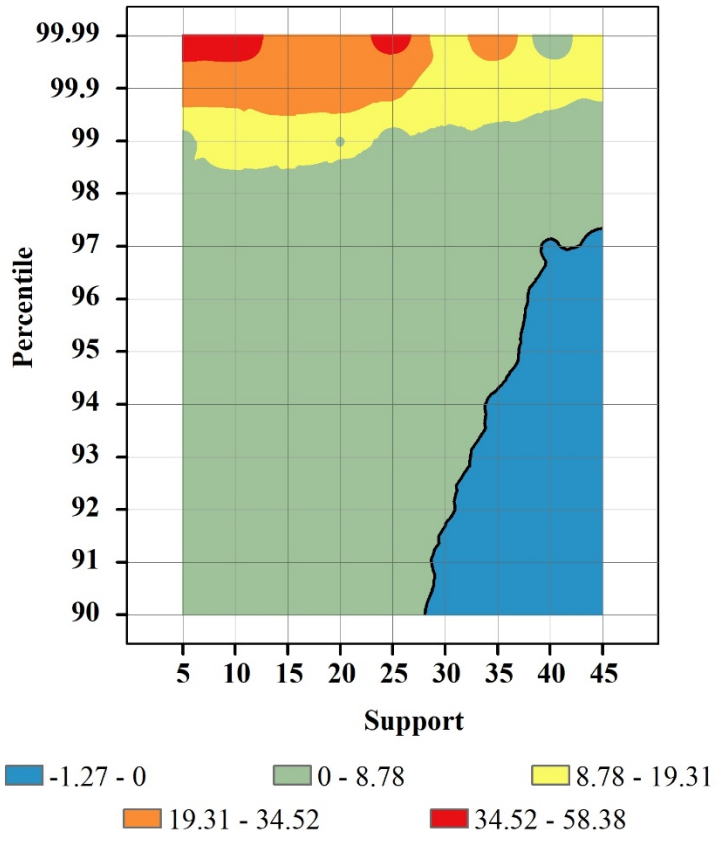


Figure 24: Difference between odds ratios S-IB - ST-IB. X-axis: Percentile of highest density sites selected for cluster delineation. Y-axis: Support parameter. Number of neighboring cases to search for bandwidth selection.

4.1.3 Significant clusters

Significance testing using Monte Carlo simulation (see Section 3.2.5.3) yields a cluster of elevated dengue fever risk, using the parameter values of 45 for case support and 95 percentile threshold value. As I ran 99 Monte Carlo simulations, the cluster is significant at the 0.01-level. The simulated odds rations range from 5.365 to 5.373, whereas the observed odds ratio was 5.501. The clustered voxels are distributed within the center of the city and within the first 314 days of the study period. The cluster has a large base at the beginning of the study period, which becomes thinner as time progresses. Therefore, distinct patterns of cluster shape are visible towards the upper end of the 314 period. For instance, the cluster seems to consist of a southern (Figure 25, 1) and a northern (Figure 25, 2) part. The northern part is substantially higher, meaning the cluster has a longer duration than the southern part. We are also able to make out detached “clouds” of voxels that have been chosen as cluster (Figure 25, 3). These “clouds” indicate regions that experienced a resurgence of dengue cases after a period of little activity.

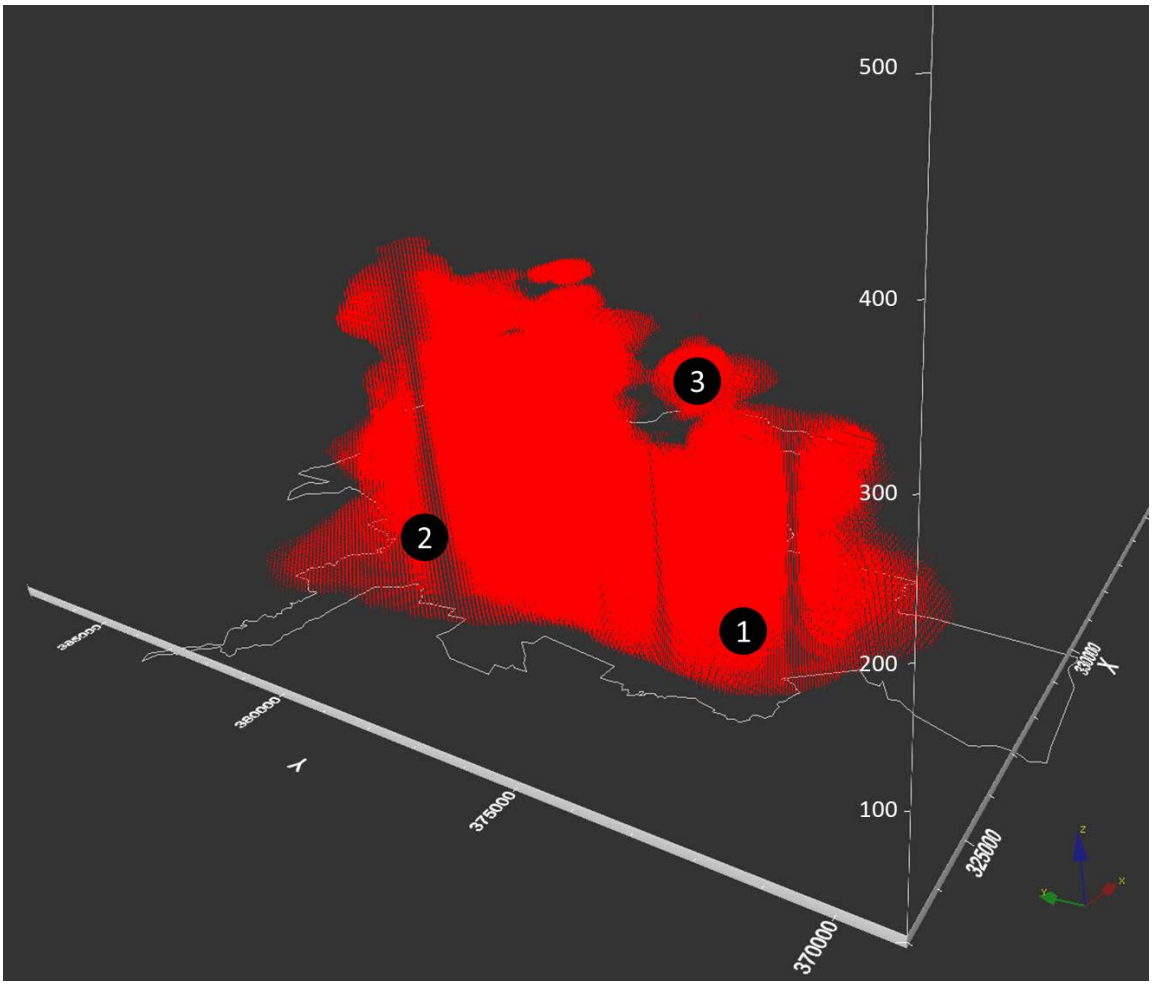


Figure 25: Voxels that form a significant cluster at the 0.01-level.
95-percentile highest densities. Support parameter = 45.

4.2 Research objective 2 results

4.2.1 Execution time of decomposition

Figure 26 shows decomposition execution times in seconds for each implementation (ST_STATIC_D, ST_FLEX_D_base, ST_FLEX_D_uneven, ST_FLX_D_alternate). Due to the varying parameter configurations, I have 336 treatments for which I recorded execution times for all implementations. Figure 26

clearly illustrates that ST_STATIC_D is the fastest in terms of decomposition time, followed by ST_FLEX_D_uneven, then ST_FLEX_D_base, and with ST_FLEX_D_alternate being the slowest implementation with the largest spread of values. The explanation is straightforward: ST_FLEX_D_alternate is a binary tree decomposition, whereas all other implementations are octree-based. It means that ST_FLEX_D_alternate splits each node into two children, whereas the other implementations split into eight, which causes the relatively slow execution times. Hence, ST_FLEX_D_alternate is profoundly different from the other implementations and any comparison between them requires caution.

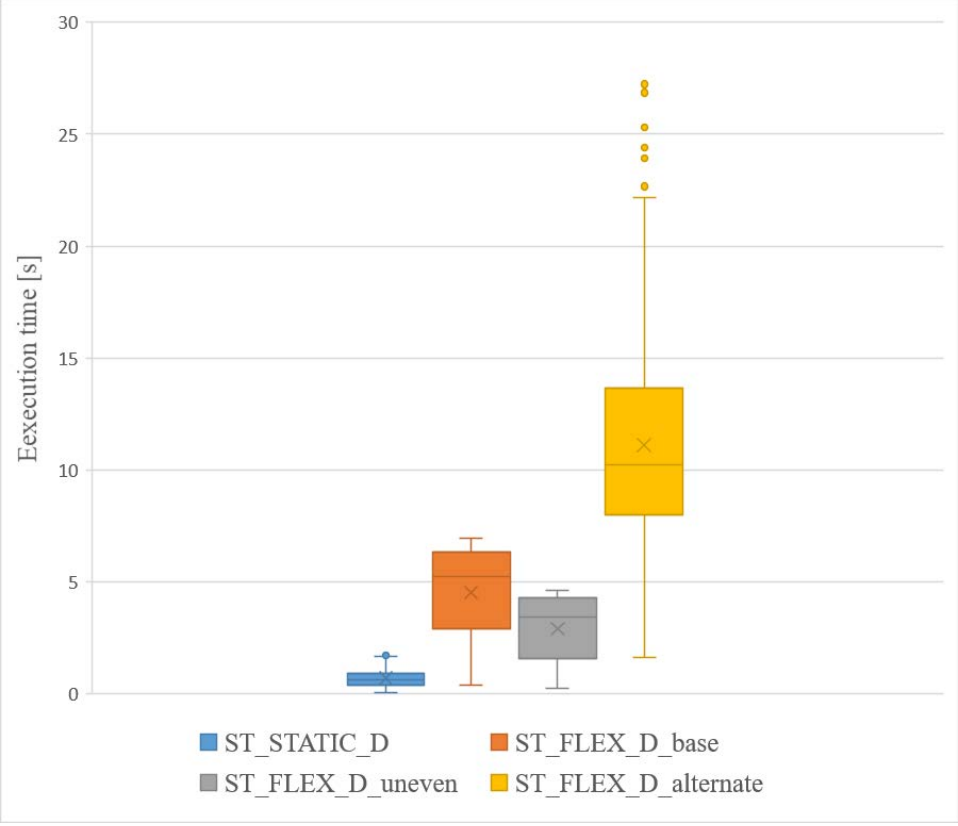


Figure 26: Average execution time in seconds.

4.3.2 Total number of cut circles

Figure 27 shows that the number of cut circles in very high in general, as it ranges from around 55,000 to 5,500,000. What initially seems to be a absurdly high number, especially keeping in mind that the initial number of data points is only 11,056. ST_STATIC_D is not necessarily the worst performing implementation when it comes to the total number of cut circles (Figure 27). Although it exhibits the largest spread, its median is lower than ST_FLEX_D_base and ST_FLEX_D_uneven, whereas ST_FLEX_D_alternate clearly performs best. Why do ST_FLEX_D_base and ST_FLEX_D_uneven perform worse than the initial implementation, even though I created them exactly to reduce that redundancy? The answer lies in sections 4.3.3 and 4.3.4, which compare the average leaf node depth and size.

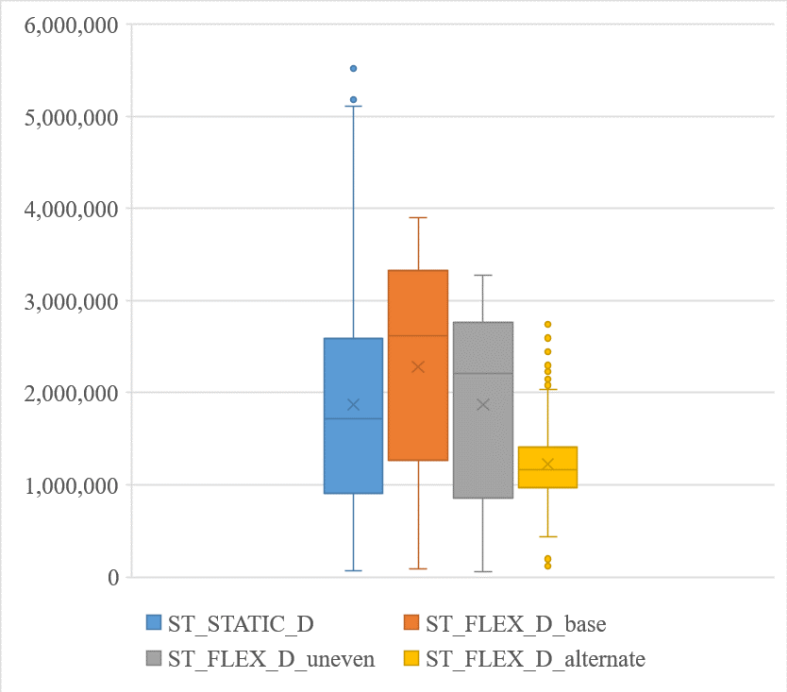


Figure 27: Number of cut circles.

Figure 28 shows the number of cut circles for each implementation across all parameter configurations. I varied the spatial and temporal bandwidths and calculated the number of cut circles for each of the 336 treatments. For ST_STATIC_D, the number of cut circles is mainly driven by the spatial bandwidth, at least more so than the other implementations. The maximum number of cut circles is achieved by a spatial bandwidth of around 1,400m-2,000m. Further increase of the spatial bandwidth results in a sharp drop of the number of cut circles and a subsequent increase (sawtooth pattern). I speculate that the interplay between various decomposition parameters (bandwidths, maximum number of points per subdomain, buffer ratio) causes the pattern. For instance, as the spatial bandwidth increases beyond the 1,400m -2,000m range, the buffer ratio threshold may kick in to prevent further decomposition (and cutting circles). ST_FLEX_D_base and ST_FLEX_D_uneven exhibit a similar pattern, where the number of cut circles grows with increasing bandwidths. ST_FLEX_D_alternate exhibits a distinct pattern, where the number of cut circles does not further grow with increasing bandwidths beyond 600m-1,200m. Again, this is not surprising as its decomposition mechanism is very different from the rest.

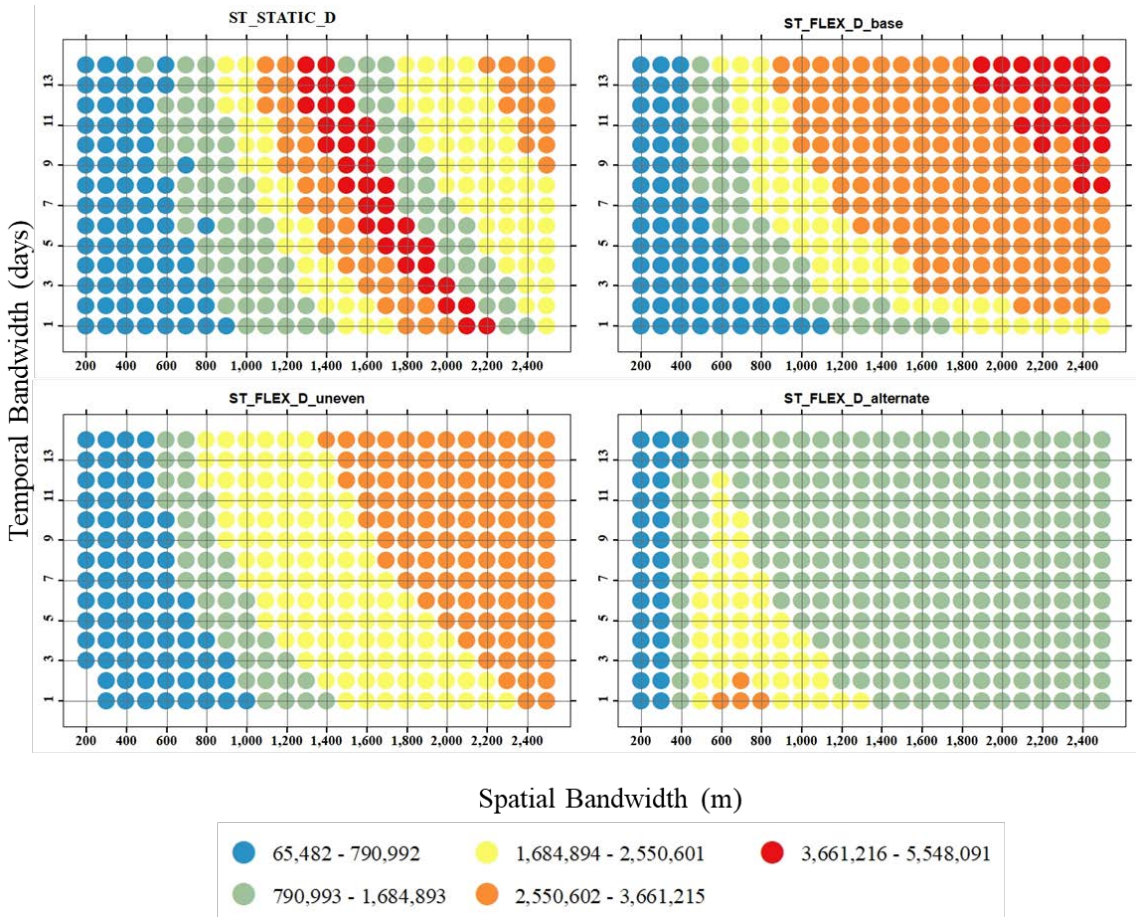


Figure 28: Number of cut circles vs. bandwidths.

4.3.3 Average leaf node depth

Here we see that ST_STATIC_D produces the shallowest tree out of all implementations (Figure 29). On average, the subdomains resulting from the decomposition procedure are created by splitting the initial domain 5.4 times. ST_FLEX_D_base has a substantially deeper tree, which explains the higher number of cut circles as compared with ST_STATIC_D (more splits lead to more cut circles), as seen in section 4.2.2. ST_FLEX_D_uneven has a similar average depth than ST_STATIC_D because the candidate split locations do not cover the entire the entire ranges. Therefore, the split locations are more similar to those if ST_STATIC_D than to

those of ST_FLEX_D_base. ST_FLEX_D_alternate naturally has a higher leaf node depths because it is binary tree.

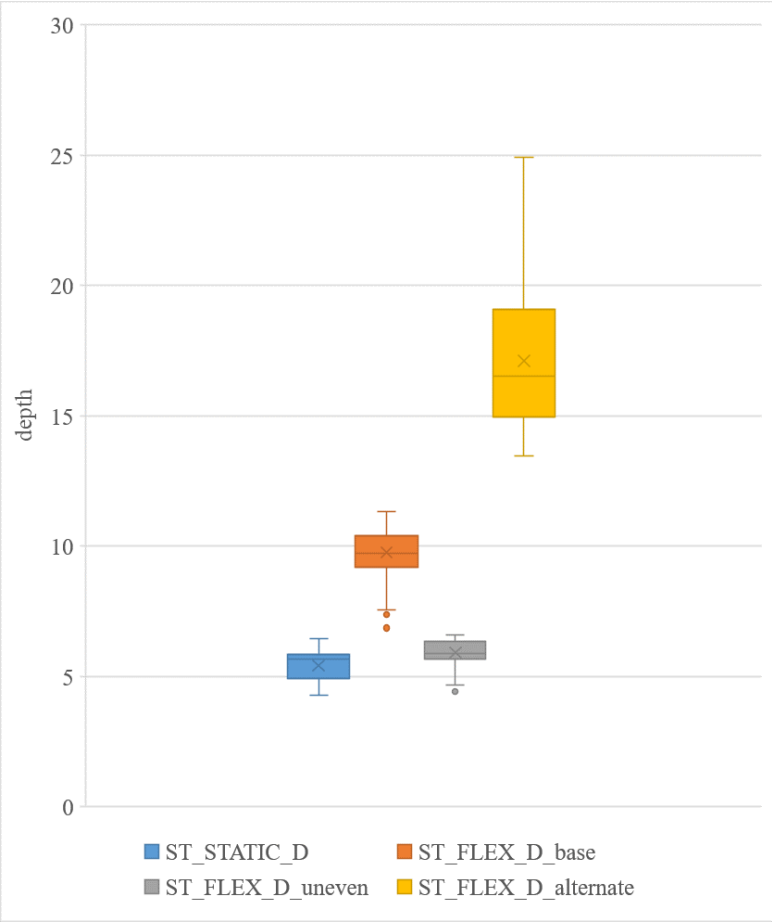


Figure 29: Average leaf node depth.

4.3.4 Average leaf node size

This indicator is relevant for parallel processing performance. Figures 30 and 31 show that ST_FLEX_D_base and ST_FLEX_D_uneven result in slightly smaller subdomains than ST_STATIC_D, which is beneficial for computational performance, given the parallel resources at hand. ST_FLEX_D_alternate results in much larger subdomains. I think this is due to the different, more compact shape of the resulting

subdomains, which calls for an adjustment of the buffer ratio parameter (which I hold steady in all our treatments).

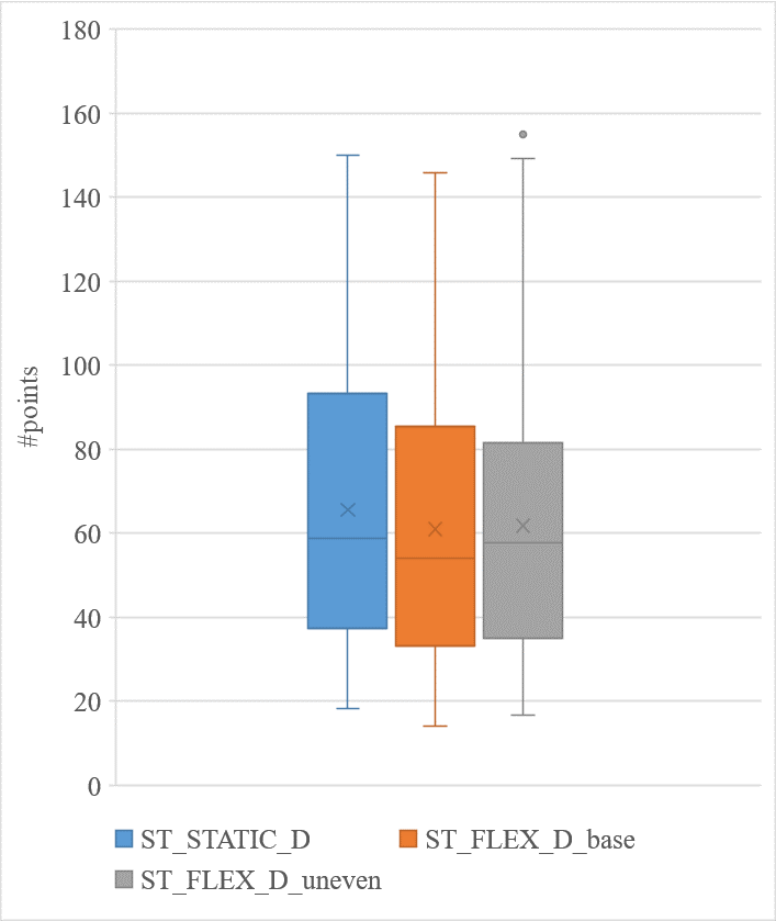


Figure 30: Average leaf node size ST_STATIC_D, ST_FLEX_D_base, ST_FLEX_D_uneven.

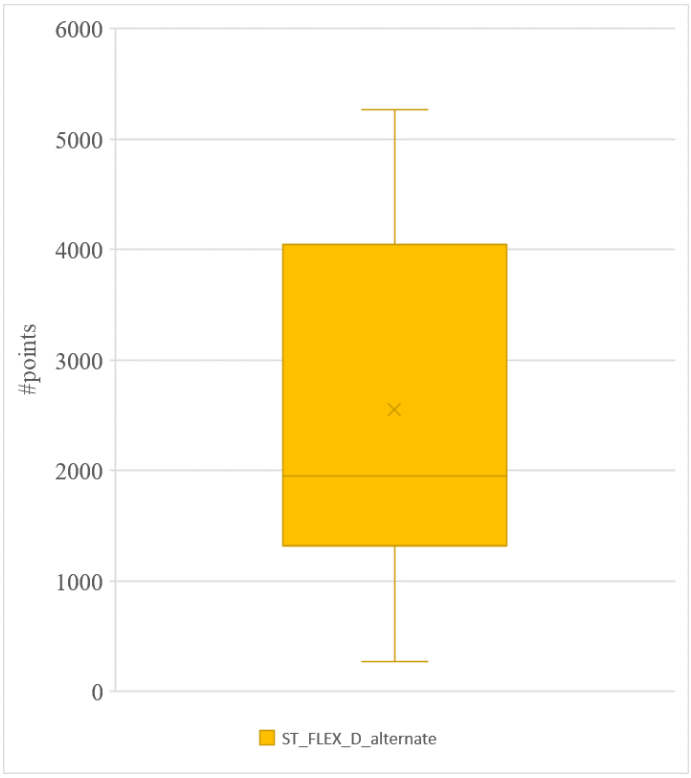


Figure 31: Average leaf node size. ST_FLEX_D_alternate.

4.3 Research objective 3 results

4.3.1 Logistic regression

The results of the regression modelling using boxcox transform indicates a good model fit (R^2 ~.9, Table 3) for execution time, but less for Speedup (R^2 ~ 0.6-0.7). All parameters are significant at the 0.001-level for execution time, but some of them become insignificant for Speedup and Efficiency.

From Table 3, the number of CPUs does not have any substantial impact on regression result for execution time. Sequential time (R^2 ~ 0.94) has a slight better

performance than parallel time (maximum R^2 ~ 0.93). The shape of R^2's distribution of speedup likes bell-shape, the highest adjusted R^2 is around 0.69 when the number of CPUs is 30. All situations of P1, P3 and P5 for speedup are significant at the 0.001 level, P6 are insignificant when #CPUs = 90. Meanwhile, P4 only significant when 30 and 40 CPUs was involved. P2 are significant at 0.001 level (#CPUs: 10), 0.05 level (#CPUs: 20, 40, 80, 100) and 0.1 level (#CPUs: 90). As the model coefficients are generated for Box-Cox transformed predictors, their actual value does not provide much information in our case.

Table 3. Logistic regression results. Significance codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

		#CPUs	Adjusted R^2	Parameters					
				P1	P2	P3	P4	P5	P6
Execution time	Sequential	1	0.9374	***	***	***	***	***	***
	Parallel	10	0.9282	***	***	***	***	***	***
		20	0.9267	***	***	***	***	***	***
		30	0.9288	***	***	***	***	***	***
		40	0.9279	***	***	***	***	***	***
		50	0.9262	***	***	***	***	***	***
		60	0.926	***	***	***	***	***	***
		70	0.9255	***	***	***	***	***	***
		80	0.925	***	***	***	***	***	***
		90	0.9249	***	***	***	***	***	***
		100	0.9236	***	***	***	***	***	***
Speedup		10	0.6185	***	***	***		***	***
		20	0.655	***	*	***		***	***
		30	0.6924	***		***	.	***	***
		40	0.6795	***	*	***	*	***	***
		50	0.6742	***		***		***	***
		60	0.6682	***		***		***	***
		70	0.6644	***		***		***	***
		80	0.6303	***	*	***		***	**
		90	0.6301	***	.	***		***	
		100	0.6125	***	*	***		***	***

4.3.2 Sensitivity Indexes

With regard to the interpretation of sensitivity indexes (see section 3.4.1), the results (Tables 3, 4, and 5) indicate that: Fixing P5 reduces the variance in the Execution time output group (sequential and parallel). However, the effect is very small, as indicated by the low levels of S_i. The remaining parameters have no such effect. All parameters reduce variance in Speedup and Efficiency, although there are no apparent patterns with varying levels of parallelization. Again, the levels of S_i are very low (0.0185 – 0.0952). The difference values $S_{Ti} - S_i$ are close to 1 across all parameters and treatments, which means that all parameters are clearly and heavily involved in interaction. However, there is some variability, although small, that is worth noticing: 1) The values are greater for execution times (sequential and parallel) than for Speedup and Efficiency, 2) Among the output group of Execution time, P5 has the smallest values, which are consistently below 1, whereas P1 – P4 are consistently above 1, while P6 exhibits mixed values. All Total-order sensitivity indexes S_{Ti} are above 0, suggesting that all of the parameters have an influence on output variance.

The sum of first-order sensitivity indexes $\sum_i S_i$ is below 1 across all parameters and treatments, which strongly suggests that our computational model is non-additive. The difference $1 - \sum_i S_i$, which is above 1 for all parameters for execution time and around 0.6 for speedup.

Table 4. First-order Sensitivity Indexes.

		#CPUs	Parameters					
			P1	P2	P3	P4	P5	P6
Execution time	Sequential	1	-0.0205	-0.0249	-0.0255	-0.0202	0.0498	-0.0279
	Parallel	10	-0.0229	-0.0298	-0.0284	-0.0228	0.036	-0.0305
		20	-0.0205	-0.0284	-0.0306	-0.0214	0.041	-0.0304
		30	-0.0215	-0.0308	-0.0293	-0.0213	0.0358	-0.0322
		40	-0.0212	-0.0287	-0.0301	-0.0218	0.0289	-0.0319
		50	-0.0222	-0.0292	-0.0289	-0.0224	0.028	-0.0317
		60	-0.0211	-0.0297	-0.0279	-0.0221	0.0298	-0.0308
		70	-0.0208	-0.0306	-0.0281	-0.0218	0.0295	-0.0305
		80	-0.0232	-0.0304	-0.0281	-0.0216	0.0319	-0.0314
		90	-0.0208	-0.0252	-0.0255	-0.0231	0.0307	-0.029
		100	-0.0216	-0.0285	-0.0281	-0.0225	0.0291	-0.029
Speedup		10	0.0599	0.0828	0.0713	0.0185	0.06	0.0813
		20	0.0762	0.0892	0.0701	0.0418	0.0946	0.0905
		30	0.0658	0.0645	0.0387	0.0438	0.0713	0.0702
		40	0.0603	0.0721	0.0524	0.0559	0.0528	0.0926
		50	0.0718	0.0643	0.0657	0.0392	0.0721	0.0768
		60	0.0677	0.0615	0.0699	0.0502	0.0806	0.0769
		70	0.068	0.0626	0.0657	0.0647	0.0636	0.0952
		80	0.0443	0.0482	0.0394	0.0635	0.0519	0.074
		90	0.0595	0.0517	0.0459	0.0668	0.0633	0.0747
		100	0.0567	0.0426	0.0303	0.0469	0.0706	0.0558

Table 5. Total-order Sensitivity Indexes.

		#CPUs	Parameters					
			P1	P2	P3	P4	P5	P6
Execution time	Sequential	1	1.0301	1.0157	1.0229	1.0179	1.0209	0.9666
	Parallel	10	1.0307	1.0194	1.0227	1.0179	1.0233	0.9694
		20	1.03	1.019	1.0228	1.0174	1.0234	0.964
		30	1.0324	1.0212	1.0221	1.0169	1.0223	0.9713
		40	1.032	1.02	1.022	1.0167	1.0238	0.965
		50	1.0319	1.0207	1.0218	1.018	1.0212	0.9776
		60	1.0325	1.0219	1.021	1.0171	1.024	0.9726
		70	1.032	1.0212	1.0214	1.0192	1.0218	0.9654
		80	1.0356	1.0225	1.0228	1.0186	1.0201	0.9801
		90	1.0307	1.023	1.0212	1.0181	1.0205	0.9769
		100	1.0324	1.021	1.0226	1.0167	1.0222	0.9747
Speedup		10	0.933	0.9184	0.9888	0.9011	0.9513	0.913
		20	0.9165	0.9229	0.9589	0.9246	0.9179	0.8988
		30	0.9378	0.9312	0.9867	0.9746	0.9293	0.9316
		40	0.9378	0.9089	0.9755	0.9717	0.9377	0.8998
		50	0.9285	0.9263	0.9741	0.9708	0.9285	0.9288
		60	0.9346	0.924	0.9745	0.9616	0.9218	0.9218
		70	0.9391	0.9136	0.9783	0.9573	0.9271	0.9002
		80	0.9565	0.9354	0.9898	0.978	0.9455	0.9125
		90	0.9435	0.9344	0.9906	0.97	0.9343	0.9218
		100	0.9417	0.9516	0.9949	0.9718	0.9409	0.9225

Table 6. Difference between Total-order and First-order Sensitivity Indexes.

		#CPUs	Parameters					
			P1	P2	P3	P4	P5	P6
Execution time	Sequential	1	1.0506	1.0406	1.0484	1.0381	0.9711	0.9945
	Parallel	10	1.0536	1.0492	1.0511	1.0407	0.9873	0.9999
		20	1.0505	1.0474	1.0534	1.0388	0.9824	0.9944
		30	1.0539	1.052	1.0514	1.0382	0.9865	1.0035
		40	1.0532	1.0487	1.0521	1.0385	0.9949	0.9969
		50	1.0541	1.0499	1.0507	1.0404	0.9932	1.0093
		60	1.0536	1.0516	1.0489	1.0392	0.9942	1.0034
		70	1.0528	1.0518	1.0495	1.041	0.9923	0.9959
		80	1.0588	1.0529	1.0509	1.0402	0.9882	1.0115
		90	1.0515	1.0482	1.0467	1.0412	0.9898	1.0059
		100	1.054	1.0495	1.0507	1.0392	0.9931	1.0037
Speedup		10	0.8731	0.8356	0.9175	0.8826	0.8913	0.8317
		20	0.8403	0.8337	0.8888	0.8828	0.8233	0.8083
		30	0.872	0.8667	0.948	0.9308	0.858	0.8614
		40	0.8775	0.8368	0.9231	0.9158	0.8849	0.8072
		50	0.8567	0.862	0.9084	0.9316	0.8564	0.852
		60	0.8669	0.8625	0.9046	0.9114	0.8412	0.8449
		70	0.8711	0.851	0.9126	0.8926	0.8635	0.805
		80	0.9122	0.8872	0.9504	0.9145	0.8936	0.8385
		90	0.884	0.8827	0.9447	0.9032	0.871	0.8471
		100	0.885	0.909	0.9646	0.9249	0.8703	0.8667

Table 7. 1 – sum of First-order Sensitivity Indexes.

		#CPUs	$1 - \sum S_i$
Execution time	Sequential	1	1.0692
	Parallel	10	1.0984
		20	1.0903
		30	1.0993
		40	1.1048
		50	1.1064
		60	1.1018
		70	1.1023
		80	1.1028
		90	1.0929
		100	1.1006
Speedup		10	0.6262
		20	0.5376
		30	0.6457
		40	0.6139
		50	0.6101
		60	0.5932
		70	0.5802
		80	0.6787
		90	0.6381
		100	0.6971

CHAPTER 5: DISCUSSION AND CONCLUSIONS

5.1 General discussion

In this study, I investigate the computational aspects detecting and analyzing space-time patterns under non-stationary backgrounds. The objectives are met by implementing and applying methodologies that allow for 1) visualizing and delineating space-time clusters of disease, 2) accelerating and scaling the computation necessary for spatiotemporal analysis, and 3) characterizing the sensitivity of computational performance to varying parameters. The three objectives have in common: 1) the use of the dengue fever dataset, study area and period, 2) the focus on space-time statistics, point pattern analysis, kernel density estimation, 3) introduction of methods that are generally applicable, but focus on the domain of spatial epidemiology.

Objective 1 concentrates on an analytical method that incorporates spatially and temporally inhomogeneous backgrounds for kernel density estimation (ST-IB). Objective 2 focuses on spatiotemporal domain decomposition accelerating and scaling kernel density estimation, allowing for big data processing. As the methods introduced in objective 2 (ST-STATIC-D, ST-FLEX-D-base, ST-FLEX-D-uneven, ST-FLEX-D-alternate) make use of overlapping subdomains, the connection to objective 1 is the bandwidth. All subdomains created by the decomposition methods introduced here are of cuboid shape and so are their overlapping (buffer) zones. The bandwidth of the underlying spatiotemporal analysis method (i.e. STKDE) determines the amount of overlap (the side length of the overlapping zone of cuboid shape). For objective 2, I make

the assumption that the bandwidth is static and known beforehand. As this is not the case for the kernel density method from objective 1 (ST-IB), I would need to find the bandwidth for each data point (procedure described by Figure 6) before decomposing, should I use spatiotemporal domain decomposition to accelerate ST-IB. So far, I implemented this step in a sequential manner, and I would need to find ways to parallelize it for handling big data. Objective 3 again focuses on the two-step procedure of domain decomposition for parallel space-time analysis. More specifically, the two-step procedure of ST-STATIC-D and STKDE is used to assess the sensitivity of computational performance to input parameters. While Objective 2 aimed for acceleration and improving scalability of the procedure, objective 3 asks the deeper question of how acceleration is influenced by parameter choices one inevitably needs to make for the two-step procedure. It is the parameters, such as space-time bandwidth and resolution of the regular grid of sites that connects objective 3 with objective 1. Whereas the resolution is held constant for objective 1 (and 2), all parameters vary in objective 3. Objective 3 shares two threshold parameters with objective 2: 1) the number of points and 2) the buffer ratio threshold. Variation of these has shown to cause variation in computational performance.

The research presented here contributes to the current body of knowledge in several ways: First, ST-IB is an important addition to the collection of analytical methods within the field of GIScience. It is an improvement over existing kernel density estimators because of its explicit consideration of the temporal dimension for population adjustment. It allows to discover patterns in geographic phenomena within the context of

spatiotemporally dynamic background populations, which is a necessity in today's age of migration and urbanization. Hence, it fills a gap where there is no such spatiotemporal statistics approach to characterize the distribution of point data under non-stationary background, despite an abundance of methods for point data without consideration of either the temporal component of the background or without acknowledgement of its existence at all (Shi 2010; Delmelle et al. 2014; Hohl, Delmelle, et al. 2016; Hazelton 2017; Zhang, Zhu, and Huang 2017; Marcon and Puech 2009; Ruckthongsook et al. 2018). ST-IB inherits the strengths of kernel density estimation methods to characterize a of variety of patterns, including changing point densities, seasonal cycles, and diffusion to new areas. It easily exposes clusters hierarchies by visualization, and allows for a certain level of subjectivity due to parameter choice. To our best knowledge, ST-IB is the first method to delineate space-time clusters under non-stationary background, measure their strength as well as statistical significance. Second, ST-FLEX-D focuses on computational aspects of spatiotemporal statistics and therefore, represents an advancement within the Geocomputation domain. Previous efforts have tackled the challenge of preventing edge effects from domain decomposition for spatial analysis by replicating overlapping domains (Hohl, Delmelle, and Tang 2015; Zheng et al. 2018) or by interprocessor communication (Shashidharan et al. 2016; Shepard 2000; Deveci et al. 2018; Liu et al. 2017). While either approach is limited in terms of scalability, ST-FLEX-D explicitly raises and analyzes the issue, and introduces a novel way of minimizing computational overhead from overlapping spatiotemporal domain decomposition by flexible partitioning. Third, ST-SA bridges the domains of sensitivity analysis and high-performance computing in a unique and novel way. The explicit focus on directly

measurable performance metrics (execution time and speedup) instead of proxys like the number of simulation runs (Tang and Jia 2014; Şalap-Ayça et al. 2018) is a new take on sensitivity analysis for computational performance of spatiotemporal analytics. ST-IB is a valid and important way for assessing the sensitivity of computational performance to input parameters. Alternatively, I could employ computational complexity in a prognostic manner to predict execution time. However, this may result in inaccurate predictions, as computational complexity theory inadequately captures the computational requirements for spatial and spatiotemporal analysis (Wang 2008). Sobol's method presents a diagnostic way to understand the laws about the computational cost of such analyses. Our focus on variance forms a clear distinction to other approaches which may not be able to capture the full range of variation while appreciating interaction effects among groups of input factors (Huang et al. 2010).

Specific plans to expand on the research presented here include developing a parallel version of ST-IB, the population-adjusted kernel density estimation approach. A parallel version of ST-IB would allow harnessing the processing power of high-performance computing, which benefits the general applicability of the method, besides having the advantages of scalability. As the domain decomposition strategy (i.e. ST-FELX-D) might not be feasible to solve this problem (see section 3.3), I intend to search for parallel algorithms developed to accelerate k-nearest neighbor (kNN) search in computer science. In order to increase its applicability to a broad range of scientific domains I aim for conducting sensitivity analysis on parallel ST-IB to gain an understanding of: 1) The factors that influence its computational performance (ST-SA),

and 2) The contribution of model parameters (i.e. support parameter and density threshold) on uncertainties in the ST clusters detected by KDE. In addition, I would like to address current limitations, which include spatial and temporal isotropy assumption of ST-IB and other kernel density estimators and developing methods following an anisotropic model. In addition, exploring the portability of parallel strategies (domain decomposition, kNN) to other space-time tests, such as the LISA statistic or Ripley's *K* function is key for increasing its application domain.

Further, I have a strong interest in generating population data at high spatial and temporal resolutions for use in the applications presented here. The field of population modelling might provide valuable insight on how to incorporate fine-scale human activity data that has the potential to account for daily movement of individuals in spatiotemporal models. Hence, I propose to integrate population information from heterogeneous sources, such as social media, location-aware technology, surveillance, census, and very high resolution satellite data, often differing in data model and conceptualization of space and time, to capture the underlying population structure. In addition to knowing how many people live in a given area, being able to derive their background information, such as demographics and socioeconomic status is critical, i.e. for disease cluster detection. If I can overcome these hurdles, I achieve a tighter definition of the "population at risk" and therefore, more reliable risk estimates.

Lastly, an investigation of the bias (e.g. uncertainty) that arises by deriving population structure from such novel data sources is useful for their applicability.

Simulation approaches have the potential to produce an estimate of the uncertainty due to the integration of heterogeneous data, thereby tackling an existing research objective within GIScience. In conclusion, I have the ambition to develop and apply advanced techniques for tackling significant challenges related to the wellbeing of our society. These challenges are semantic, computational, and methodological, and their solution might lead us to the forefront of an applied data science for practitioners.

I expect to publish three papers in high-impact peer-reviewed journals related to this dissertation. All publications include Alexander Hohl as first and author, as well as Eric Delmelle and Wenwu Tang. A first paper bases on a collaboration with Dr. Xun Shi from Dartmouth College, NH, draws from research objective 1 and fits outlets focusing on spatiotemporal analysis and modelling for health applications, such as the International Journal of Health Geographics (IJHG), or Epidemiology. Tentative title of the paper is “Kernel Density Estimation for Spatially and Temporally Inhomogeneous Backgrounds”. In collaboration with Dr. Erik Saule from the Department of Computer Science of University of North Carolina at Charlotte (UNCC), we plan to submit the second paper, titles “A Flexible Splits Algorithm for Spatiotemporal Domain Decomposition” which extends the concepts introduced in the second dissertation chapter. We have identified suitable outlets that focus on spatial informatics, algorithms and high-performance computing, such as ACM Transactions on Spatial Algorithms and Systems (TSAS), or Parallel Computing. A group effort among colleagues at the Center for Applied GIScience at UNCC to submit a paper that corresponds to objective 3 to the International Journal of Geographic Information Science (IJGIS) is currently ongoing.

The paper is tentatively titled “Computational Sensitivity Analysis: Establishing a Relationship between Model Parameters and Computational Cost” and collaborators include Minrui Zheng and Meijuan Jia.

5.2 Research objective 1 discussion

ST-IB outperforms S-IB for certain parameter configurations. This is a positive result, which means that there is a benefit of adding the temporal dimension to such analyses. In other words, adding the temporal dimension yields higher odds ratios for certain parameter configurations, which means that it improves our ability to delineate clusters of disease occurrence under spatial and temporal inhomogeneous backgrounds. The choice of parameters is admittedly arbitrary, but I confirmed the validity of the resulting clusters by significance testing. Therefore, I created two measures of describing clusters: 1) I quantify the strength of a cluster by its odds ratio. The higher the ratio, the greater the difference in odds pf contracting the disease inside vs. outside the cluster. 2) I quantify the significance of the cluster by its p-value. Therefore, the clustering of observed dengue cases by admittedly choosing arbitrary parameter values generates higher odds ratios compared to all of the randomly simulated datasets.

The results obtained here point towards the following weaknesses and discussion points, some of which need to be addressed in the future. The following passages provide a collection of these points, as well as comments on them: First, I determine kernel bandwidth by the *n*-nearest neighbors in space and time. I only search for past neighboring cases, which is reasonable in a disease setting. However, depending on the

parameter setting, this procedure creates a boundary effect at the beginning of the study period, as I have no case data prior to the start of the data collection. I handled this issue by ignoring the first 5 days of data collection. More specifically, I only used the first 5 days of case data for determining bandwidths of kernels centered on cases that appear later than the 5 first days. This cut-off point is arbitrary and a data driven solution has yet to be found. If the cut-off is not properly chosen, the following scenario might play out: The search for spatial and temporal neighbors cannot go further into the past for additional temporal support, so I expand the search spatially, which leads to larger spatial kernel bandwidths at the beginning of the study period. This mechanism is evident in Figures 22 and 23, where high density values seem to form a large base at the beginning of the study period, indicative of large spatial bandwidths. Second, I like to clearly state that our goal and main contribution is kernel density estimation for spatially and temporally inhomogeneous backgrounds. I merely use the procedure of delineating clusters and quantifying their strength and significance by odds ratios as a means of validation and comparison against S-IB. With that said, an alternative to our clustering procedure would be to assess statistical significance of the kernel density estimates directly. Therefore, for each site, I would rank the density estimates of the observed dataset among the simulated ones and denote significant sites as part of the cluster. Third, we clearly saw that uncertainty from population simulation was low. This result is due to the small rates of population change within the study period. The simulation procedure (Figure 8) may be improved through dasymetric mapping to better reflect population concentration. Fourth, in order to apply ST-IB to domains other than spatial epidemiology, it may be of interest to handle more than the three dimensions (X, Y, T)

used here. While adding dimensions is rather straightforward computationally, visualization of the resulting density estimates becomes more difficult. Hohl et al. (2018) discuss visualization of 3D point data over time, thus offering an approach to solve the challenge. Fifth, ST-IB assumes that cases and population are distributed on an infinitely continuous planar space, which justifies the use of Euclidean distance. However, as people and goods move along the road network, it is necessary to adapting ST-IB towards network distance, drawing from existing research about kernel density estimation for networks (Okabe, Satoh, and Sugihara 2009), space-time hotspot detection for street-level incidents (Shiode and Shiode 2013), and local indicators of network-constrained clusters (Yamada and Thill 2007). Sixth, I may implement a more sophisticated population modelling approach, which omits the admittedly very strong assumption of linear population change. In addition, a dasymetric mapping approach (Wright 1936; Eicher and Brewer 2001; Mennis 2003) would certainly help further increase the accuracy of spatiotemporal population distribution. Seventh, I use epidemiological data under the assumption that people contracted the disease at their residential location. However, this is not necessarily true, as people move around the city for daily commutes or leisure time activities. Therefore, I should address the question whether uncertainty in the spatiotemporal location of disease cases has an effect on our results. In a first step, I may quantify this uncertainty by a measure of the offset of our case locations in space and time through analysis of activity diary data (Chen et al. 2011). Achieving an estimate of people's space-time prism (Miller 1991) then allows us to perturb the case data and compute the effect of locational uncertainty on our results. Eighth, it is evident from Equation (6) that spatial and temporal components of kernel density are of equal

importance. However, for certain applications, a bi-weighted approach might be warranted, where the importance of space is weighted against the importance of time, guided by domain expertise. Li et al. (2014) present a more general approach to weigh space against time, which scales the temporal extent of a spatiotemporal dataset to achieve an equal range to the average spatial extent.

5.3 Research objective 2 discussion

I implemented and compared four different spatiotemporal domain decomposition methods with respect to replication of data points that fall within buffer zones around subdomains. The results indicate that I am able to reduce the number of replicated points during decomposition. This is a positive result because the redundancy stemming from replication is what ultimately limits scalability. However, a more detailed look at the results reveals that reducing redundancy comes at the price of increased execution time and/or decreased granularity of the decomposition. Both of these results are troublesome: I apply domain decomposition to accelerate subsequent spatiotemporal analysis and slow decomposition defeats its own purpose. In addition, if the granularity of the decomposition is not fine enough, subsequent parallel processing performance will be limited by the largest subdomain resulting from decomposition.

I found that our metrics to analyze decomposition (number of cut circles, leaf node depth, leaf node size) exhibit substantial variation across the bandwidths assessed (Figure X), especially for ST_STATIC_D. The all implementations of ST_FLEX_D exhibit less variation and are therefore more predictable in their behavior. Future efforts

include assessing sensitivity of ST_FLEX_D implementations to variations of parameters I held steady, such as the spatial and temporal grid resolutions (P3 & P4), the number of data points threshold (P5) and the buffer ratio threshold (P6). In addition, I identified considerable potential to accelerate the decomposition procedure: First, using a fast programming language, such as C++ may improve the overall performance (I currently use Python). Second, I could replace our extensive use of the “append” method in Python by initializing arrays of fixed size (maximum number of elements is known), then populating them by indexing. Third, the procedure of finding the best split creates redundancy because I iterate over the array twice (once for finding the best split, once for assigning the data points to their respective subdomains).

5.4 Research objective 3 discussion

Sobol’s sensitivity indexes are a reliable quantitative measurement of evaluating the impact of factors (e.g., buffer distance, grid resolution) on uncertainty of computational model response (e.g., execution time and speedup). It allows for expressing and contrasting the power of interactions among model parameters (by a combination of both, first-order and total-order sensitivity indexes). In this chapter, I assess the robustness of computational performance with respect to uncertainty in parameter values of the two-stage procedure of ST_STATIC_D and parallel STKDE. Our parallel implementation of Sobol’s approach allows for computationally tractable the sensitivity analysis and, therefore has the potential to increase our comprehension of complex spatial algorithms.

REFERENCES

Ang, Q. W., A. Baddeley, and G. Nair. 2012. Geometrically corrected second order analysis of events on a linear network, with applications to ecology and criminology. *Scandinavian Journal of Statistics* 39 (4):591-617.

Anselin, L. 1995. Local indicators of spatial association—LISA. *Geographical analysis* 27 (2):93-115.

Anselin, L. 2011. From SpaceStat to CyberGIS: Twenty years of spatial data analysis software. . *International Regional Science Review* 35:131-157.

Archer, G., A. Saltelli, and I. Sobol. 1997. Sensitivity measures, ANOVA-like techniques and the use of bootstrap. *Journal of Statistical Computation and Simulation* 58 (2):99-120.

Armstrong, M. P. 2000. Geography and computational science. *Annals of the Association of American Geographers* 90 (1):146-156.

Armstrong, M. P., C. E. Pavlik, and R. Marciano. 1994. Parallel processing of spatial statistics. *Computers & Geosciences* 20 (2):91-104.

Assuncao, R. M., and E. A. Reis. 1999. A new proposal to adjust Moran's I for population density. *Statistics in medicine* 18 (16):2147-2162.

Bach, B., P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale. 2016. A Descriptive Framework for Temporal Data Visualizations Based on Generalized Space-Time Cubes. Computer Graphics Forum, 36: 36-61. doi:10.1111/cgf.12804

Bader, M. 2012. *Space-filling curves: an introduction with applications in scientific computing*: Springer Science & Business Media.

Bailey, T., and Q. Gatrell. 1995. *Interactive Spatial Data Analysis*. Edinburgh Gate, England: Pearson Education Limited.

Bentley, J. L. 1975. Multidimensional binary search trees used for associative searching. *Communications of the ACM* 18 (9):509-517.

Bhaduri, B., E. Bright, P. Coleman, and M. L. Urban. 2007. LandScan USA: a high-resolution geospatial and temporal modeling approach for population distribution and dynamics. *GeoJournal* 69 (1-2):103-117.

Bithell, J. 2000. A classification of disease mapping methods. *Statistics in medicine* 19 (17-18):2203-2215.

Bithell, J. F. 1990. An application of density estimation to geographical epidemiology. *Statistics in medicine* 9 (6):691-701.

Bland, J. M., and D. G. Altman. 2000. The odds ratio. *BMJ* 320 (7247):1468.

Boots, B., and A. Okabe. 2007. Local statistical spatial analysis: Inventory and prospect. *International Journal of Geographical Information Science* 21 (4):355-375.

Bowman, A. W., and A. Azzalini. 1997. *Applied smoothing techniques for data analysis: the kernel approach with S-Plus illustrations*: OUP Oxford.

Box, G. E., and D. R. Cox. 1964. An analysis of transformations. *Journal of the Royal Statistical Society. Series B (Methodological)*:211-252.

Boyandin, I., E. Bertini, and D. Lalanne. 2012. A Qualitative Study on the Exploration of Temporal Changes in Flow Maps with Animation and Small-Multiples. Computer Graphics Forum, 31: 1005-1014. doi:10.1111/j.1467-8659.2012.03093.x

Brunsdon, C., J. Corcoran, and G. Higgs. 2007. Visualising space and time in crime patterns: A comparison of methods. *Computers, Environment and Urban Systems* 31 (1):52-75.

Cai, Q., G. Rushton, and B. Bhaduri. 2012. Validation tests of an improved kernel density estimation method for identifying disease clusters. *Journal of Geographical Systems* 14 (3):243-264.

Cali, A. d. S. d. 2014. Cali en cifras, ed. A. d. S. d. Cali. Cali, Colombia: Alcaldía de Santiago de Cali.

Carlos, H. A., X. Shi, J. Sargent, S. Tanski, and E. M. Berke. 2010. Density estimation and adaptive bandwidths: a primer for public health practitioners. *International journal of health geographics* 9 (1):39.

Casas, I., E. Delmelle, and A. Varela. 2010. A space-time approach to diffusion of health service provision information. *International Regional Science Review* 33 (2):134-156.

Casavant, T. L., and J. G. Kuhl. 1988. A taxonomy of scheduling in general-purpose distributed computing systems. *IEEE Transactions on software engineering* 14 (2):141-154.

Castles, S., H. De Haas, and M. J. Miller. 2013. *The age of migration: International population movements in the modern world*: Palgrave Macmillan.

Champion, T. 2001. Urbanization, suburbanization, counterurbanization and reurbanization. *Handbook of urban studies* 160:1.

Charras-Garrido, M., D. Abrial, J. D. Goër, S. Dachian, and N. Peyrard. 2012. Classification method for disease risk mapping based on discrete hidden Markov random fields. *Biostatistics* 13 (2):241-255.

Chen, J., S.-L. Shaw, H. Yu, F. Lu, Y. Chai, and Q. Jia. 2011. Exploratory data analysis of activity diary data: a space–time GIS approach. *Journal of Transport Geography* 19 (3):394-404.

Cheng, T. 2013. Accelerating universal Kriging interpolation algorithm using CUDA-enabled GPU. *Computers & geosciences* 54:178-183.

Colombia, M. d. S. 2017. *Sistema de vigilancia en salud pública*. Ministerio de Salud Colombia 2017 [cited February 2017]. Available from https://www.minsalud.gov.co/salud/Paginas/SIVIGILA.aspx.

Cressie, N. 1990. The origins of kriging. *Mathematical geology* 22 (3):239-252.

Cressie, N., and C. K. Wikle. 2015. *Statistics for spatio-temporal data*: John Wiley & Sons.

Cukier, R., C. Fortuin, K. E. Shuler, A. Petschek, and J. Schaibly. 1973. Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. I Theory. *The Journal of chemical physics* 59 (8):3873-3878.

Cukier, R., H. Levine, and K. Shuler. 1978. Nonlinear sensitivity analysis of multiparameter model systems. *Journal of computational physics* 26 (1):1-42.

Davies, T. M., and M. L. Hazelton. 2010. Adaptive kernel estimation of spatial relative risk. *Statistics in Medicine* 29 (23):2423-2437.

Davies, T. M., K. Jones, and M. L. Hazelton. 2016. Symmetric adaptive smoothing regimens for estimation of the spatial relative risk function. *Computational Statistics & Data Analysis* 101:12-28.

Daya, A. A., and H. Bejari. 2015. A comparative study between simple kriging and ordinary kriging for estimating and modeling the Cu concentration in Chehlkureh deposit, SE Iran. *Arabian Journal of Geosciences* 8 (8):6003-6020.

de Oliveira Martins, L., A. C. Silva, A. C. De Paiva, and M. Gattass. 2009. Detection of breast masses in mammogram images using growing neural gas algorithm and Ripley's K function. *Journal of Signal Processing Systems* 55 (1-3):77-90.

Dean, J., and S. Ghemawat. 2008. MapReduce: simplified data processing on large clusters. *Communications of the ACM* 51 (1):107-113.

Delmelle, E., C. Dony, I. Casas, M. Jia, and W. Tang. 2014. Visualizing the impact of space-time uncertainties on dengue fever patterns. *International Journal of Geographical Information Science* 28 (5):1107-1127.

Desjardins, M., A. Hohl, A. Griffith, and E. Delmelle. 2017. Fine-scale visualization of pollen concentrations across the Eastern United States: A space-time parallel approach. *Proceedings of the 2017 International Conference on GeoComputation*, Leeds, UK.

Deveci, M., K. D. Devine, K. Pedretti, M. A. Taylor, S. Rajamanickam, and U. V. Catalyurek. 2018. Geometric Partitioning and Ordering Strategies for Task Mapping on Parallel Computers. *arXiv preprint arXiv:1804.09798*.

Difallah, D. E., P. Cudre-Mauroux, and S. McKenna. 2013. Scalable anomaly detection for smart city infrastructure networks. *Internet Computing, IEEE* 17 (6):39-47.

Diggle, P. J. 2013. *Statistical analysis of spatial and spatio-temporal point patterns*: CRC Press.

Ding, Y., and P. J. Densham. 1996. Spatial strategies for parallel spatial modelling. *International Journal of Geographical Information Systems* 10 (6):669-698.

Dixon, P. M. 2002. Ripley's K function. *Encyclopedia of environmetrics*.

Draper, N. R., and H. Smith. 2014. *Applied regression analysis*: John Wiley & Sons.

Eicher, C. L., and C. A. Brewer. 2001. Dasymetric Mapping and Areal Interpolation: Implementation and Evaluation. *Cartography and Geographic Information Science* 28 (2):125-138.

Eisen, L., and R. Eisen. 2011. Using geographic information systems and decision support systems for the prediction, prevention, and control of vector-borne diseases. *Annual Review of Entomology* 56 (1):41-61.

Fachada, N., V. V. Lopes, R. C. Martins, and A. C. Rosa. 2017. Parallelization Strategies for Spatial Agent-Based Models. *International Journal of Parallel Programming* 45 (3):449-481.

Fernando, W. S., and M. L. Hazelton. 2014. Generalizing the spatial relative risk function. *Spatial and spatio-temporal epidemiology* 8:1-10.

Gao, S. 2015. Spatio-temporal analytics for exploring human mobility patterns and urban dynamics in the mobile age. *Spatial Cognition & Computation* 15 (2):86-114.

Gao, Y., S. Wang, A. Padmanabhan, J. Yin, and G. Cao. 2018. Mapping spatiotemporal patterns of events using social media: a case study of influenza trends. *International Journal of Geographical Information Science* 32 (3):425-449.

Gentle, J. E. 2006. *Random number generation and Monte Carlo methods*: Springer Science & Business Media.

Getis, A., and J. Franklin. 1987. Second-Order Neighborhood Analysis of Mapped Point Patterns. *Ecology* 68 (3):473-477.

Getis, A., and J. K. Ord. 1992. The analysis of spatial association by use of distance statistics. *Geographical analysis* 24 (3):189-206.

Goodchild, M. F. 2007. Citizens as sensors: the world of volunteered geography. *GeoJournal* 69 (4):211-221.

Goovaerts, P., and G. M. Jacquez. 2005. Detection of temporal changes in the spatial distribution of cancer rates using local Moran's I and geostatistically simulated spatial neutral models. *Journal of Geographical Systems* 7 (1):137-159.

Graham, R. L. 1994. *Concrete mathematics:[a foundation for computer science; dedicated to Leonhard Euler (1707-1783)]*: Pearson Education India.

Grubesic, T. H., R. Wei, and A. T. Murray. 2014. Spatial Clustering Overview and Comparison: Accuracy, Sensitivity, and Computational Expense. *Annals of the Association of American Geographers* 104 (6):1134-1156.

Gu, W. W., Jishui; Shi, Hao; Liu Yongshan. 2011. Research on a Hybrid Spatial Index Structure. *Journal of Computational Information Systems* 7 (11):3972-3978.

Guan, X., and H. Wu. 2010. Leveraging the power of multi-core platforms for large-scale geospatial data processing: Exemplified by generating DEM from massive LiDAR point clouds. *Computers & Geosciences* 36 (10):1276-1282.

Guo, L., S. Du, R. Haining, and L. Zhang. 2013. Global and local indicators of spatial association between points and polygons: a study of land use change. *International Journal of Applied Earth Observation and Geoinformation* 21:384-396.

Hazelton, M. L. 2017. Testing for changes in spatial relative risk. *Statistics in medicine*.

Heuvelink, G., and D. A. Griffith. 2010. Space–Time Geostatistics for Geography: A Case Study of Radiation Monitoring Across Parts of Germany. *Geographical analysis* 42 (2):161-179.

Hey, A. J., S. Tansley, and K. M. Tolle. 2009. *The fourth paradigm: data-intensive scientific discovery*: Microsoft research Redmond, WA.

Hohl, A., I. Casas, E. Delmelle, and W. Tang. 2016. Hybrid Indexing for Parallel Analysis of Spatiotemporal Point Patterns. *9th International Conference on GIScience Short Paper Proceedings*.

Hohl, A., A. D. Griffith, M. C. Eppes, and E. Delmelle. 2018. Computationally Enabled 4D Visualizations Facilitate the Detection of Rock Fracture Patterns from Acoustic Emissions. *Rock Mechanics and Rock Engineering*.

Hohl, A., E. Delmelle, W. Tang, and I. Casas. 2016. Accelerating the discovery of space-time patterns of infectious diseases using parallel computing. *Spatial and spatio-temporal epidemiology* 19:10-20.

Hohl, A., E. M. Delmelle, and W. Tang. 2015. Spatiotemporal domain decomposition for massive parallel computation of space-time kernel density. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2 (4):7.

Homma, T., and A. Saltelli. 1996. Importance measures in global sensitivity analysis of nonlinear models. *Reliability Engineering & System Safety* 52 (1):1-17.

Hu, H., and H. Shu. 2015. An improved coarse-grained parallel algorithm for computational acceleration of ordinary Kriging interpolation. *Computers & Geosciences* 78:44-52.

Huang, L., J. Jia, B. Yu, B.-G. Chun, P. Maniatis, and M. Naik. 2010. Predicting execution time of computer programs using sparse polynomial regression. Advances in neural information processing systems, pp. 883-891.

Jacquez, G., D. Greiling, and A. Kaufmann. 2005. Design and implementation of a space-time intelligence system for disease surveillance. *Journal of Geographical Systems* 7 (1):7-23.

Jacquez, G. M. 1996. A k nearest neighbour test for space–time interaction. *Statistics in medicine* 15 (18):1935-1949.

Jafari-Mamaghani, M., M. Andersson, and P. Krieger. 2010. Spatial point pattern analysis of neurons using Ripley's K-function in 3D. *Frontiers in neuroinformatics* 4.

Jones, A. P., I. H. Langford, and G. Bentham. 1996. The application of K-function analysis to the geographical distribution of road traffic accident outcomes in Norfolk, England. *Social Science & Medicine* 42 (6):879-885.

Karin, S., and S. Graham. 1998. The high-performance computing continuum. *Communications of the ACM*:32-33.

Kauhl, B., J. Schweikart, T. Krafft, A. Keste, and M. Moskwyn. 2016. Do the risk factors for type 2 diabetes mellitus vary by location? A spatial analysis of health insurance claims in Northeastern Germany using kernel density estimation and geographically weighted regression. *International Journal of Health Geographics* 15 (1):38.

Kelly, M., and R. K. Meentemeyer. 2002. Landscape dynamics of the spread of sudden oak death. *Photogrammetric Engineering and Remote Sensing* 68 (10):1001-1010.

Kulldorff, M. 1997. A spatial scan statistic. *Communications in Statistics-Theory and methods* 26 (6):1481-1496.

Kulldorff, M. 2010. SaTScan-Software for the spatial, temporal, and space-time scan statistics. *Boston: Harvard Medical School and Harvard Pilgrim Health Care*.

Kwan, M.-P. 2000. Interactive geovisualization of activity-travel patterns using three-dimensional geographical information systems: a methodological exploration with a large data set. *Transportation Research Part C: Emerging Technologies* 8 (1):185-203.

Kwan, M.-P. 2004. GIS methods in time-geographic research: Geocomputation and geovisualization of human activity patterns. *Geografiska Annaler B* 86:205-218.

Kwan, M.-P., and T. Neutens. 2014. Space-time research in GIScience. *International Journal of Geographical Information Science* 28 (5):851-854.

Lang, R. E., and P. A. Simmons. 2001. Boomburbs: The Emergence of Large, Fast-Growing Suburban Cities. *Fannie Mae Foundation, Washington, DC*, 14.

Lemke, D., V. Mattauch, O. Heidinger, E. Pebesma, and H.-W. Hense. 2015. Comparing adaptive and fixed bandwidth-based kernel density estimates in spatial cancer epidemiology. *International Journal of Health Geographics* 14 (1):15.

Li, L., T. Losser, C. Yorke, and R. Piltner. 2014. Fast inverse distance weighting-based spatiotemporal interpolation: a web-based application of interpolating daily fine particulate matter PM2. 5 in the contiguous US using parallel programming and kd tree. *International journal of environmental research and public health* 11 (9):9101-9141.

Lilburne, L., and S. Tarantola. 2009. Sensitivity analysis of spatial models. *International Journal of Geographical Information Science* 23 (2):151-168.

Lin, G., and T. Zhang. 2007. Loglinear residual tests of Moran's I autocorrelation and their applications to Kentucky breast cancer data. *Geographical Analysis* 39 (3):293-310.

Liu, H., Z. Huang, Q. Zhan, and P. Lin. 2008. A database approach to very large LiDAR data management. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Beijing, China* 37 (B1):463-468.

Liu, H., K. Wang, B. Yang, M. Yang, R. He, L. Shen, H. Zhong, and Z. Chen. 2017. Load Balancing using Hilbert Space-filling Curves for Parallel Reservoir Simulations. *arXiv preprint arXiv:1708.01365*.

Liu, Y., K. Wu, S. Wang, Y. Zhao, and Q. Huang. 2010. A MapReduce approach to G i*(d) spatial statistic. *Proceedings of the 2010 ACM SIGSPATIAL International Workshop on High Performance and Distributed Geographic Information Systems*. San Jose, CA, USA, 11–18.

Malleson, N., and M. A. Andresen. 2015. Spatio-temporal crime hotspots and the ambient population. *Crime Science* 4 (1):1-8.

Marcon, E., and F. Puech. 2009. Generalizing Ripley's K function to inhomogeneous populations. Mimeo: New York, NY, USA.

Martin, S. F. 2001. Forced migration and the evolving humanitarian regime. *New Issues in Refugee Research*, Working Paper 20, UNHCR, Geneva.

McMichael, A. J. 2004. Environmental and social influences on emerging infectious diseases: past, present and future. *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 359 (1447):1049-1058.

Meentemeyer, R. K., W. Tang, M. A. Dorning, J. B. Vogler, N. J. Cunniffe, and D. A. Shoemaker. 2013. FUTURES: multilevel simulations of emerging urban–rural landscape structure using a stochastic patch-growing algorithm. *Annals of the Association of American Geographers* 103 (4):785-807.

Mennis, J. 2003. Generating Surface Models of Population Using Dasymetric Mapping. *The Professional Geographer* 55 (1):31-42.

Miller, H. J. 1991. Modelling accessibility using space-time prism concepts within geographical information systems. *International Journal of Geographical Information Systems* 5 (3):287-301.

Mitchell, M. I. 2011. Insights from the cocoa regions in Côte d'Ivoire and Ghana: Rethinking the migration–conflict nexus. *African Studies Review* 54 (2):123-144.

Moran, P. A. 1950. Notes on continuous stochastic phenomena. *Biometrika* 37 (1/2):17-23.

Münz, R. 2007. Migration, labor markets, and integration of migrants: An overview for Europe (No. 3-6), HWWI policy paper.

Nakaya, T. 2013. Analytical Data Transformations in Space–Time Region: Three Stories of Space–Time Cube. *Annals of the Association of American Geographers* (ahead-of-print).

Nakaya, T., and K. Yano. 2010. Visualising Crime Clusters in a Space-time Cube: An Exploratory Data-analysis Approach Using Space-time Kernel Density Estimation and Scan Statistics. *Transactions in GIS* 14 (3):223-239.

Okabe, A., T. Satoh, and K. Sugihara. 2009. A kernel density estimation method for networks, its computational method and a GIS-based tool. *International Journal of Geographical Information Science* 23 (1):7-32.

Okabe, A., and I. Yamada. 2001. The K-Function Method on a Network and Its Computational Implementation. *Geographical analysis* 33 (3):271-290.

Padmanabhan, A., S. Wang, G. Cao, M. Hwang, Z. Zhang, Y. Gao, K. Soltani, and Y. Liu. 2014. FluMapper: A cyberGIS application for interactive analysis of massive location-based social media. *Concurrency and Computation: Practice and Experience*.

Pfister, G. F. 2001. An introduction to the infiniband architecture. *High Performance Mass Storage and Parallel I/O* 42:617-632.

Restrepo, L. D. E. 2011. El plan piloto de cali 1950. *Bitácora Urbano Territorial* 1 (10):222-233.

Ripley, B. D. 1976. The second-order analysis of stationary point processes. *Journal of applied probability* 13 (2):255-266.

Robertson, C., and T. A. Nelson. 2010. Review of software for space-time disease surveillance. *Int J Health Geogr* 9 (16):10.1186.

Rogerson, P., and I. Yamada. 2008. *Statistical Detection and Surveillance of Geographic Clusters*. Boca Raton, Florida: CRC Press.

Rokos, D.-K. D., and M. P. Armstrong. 1996. Using Linda to compute spatial autocorrelation in parallel. *Computers & Geosciences* 22 (4):425-432.

Ruckthongsook, W., C. Tiwari, J. R. Oppong, and P. Natesan. 2018. Evaluation of threshold selection methods for adaptive kernel density estimation in disease mapping. *International Journal of Health Geographics* 17 (1):10.

Rushton, G., and P. Lolonis. 1996. Exploratory spatial analysis of birth defect rates in an urban population. *Statistics in medicine* 15 (7-9):717-726.

Şalap-Ayça, S., P. Jankowski, K. C. Clarke, P. C. Kyriakidis, and A. Nara. 2018. A meta-modeling approach for spatio-temporal uncertainty and sensitivity analysis: an application for a cellular automata-based Urban growth and land-use change model. *International Journal of Geographical Information Science* 32 (4):637-662.

Saltelli, A. 1999. Sensitivity analysis: Could better methods be used? *Journal of Geophysical Research: Atmospheres* 104 (D3):3789-3793.

Saltelli, A. 2002. Making best use of model evaluations to compute sensitivity indices. *Computer Physics Communications* 145 (2):280-297.

Saltelli, A., and R. Bolado. 1998. An alternative way to compute Fourier amplitude sensitivity test (FAST). *Computational Statistics & Data Analysis* 26 (4):445-460.

Saltelli, A., M. Ratto, T. Andres, F. Campolongo, J. Cariboni, D. Gatelli, M. Saisana, and S. Tarantola. 2008. *Global sensitivity analysis: the primer*: John Wiley & Sons.

Saltelli, A., S. Tarantola, F. Campolongo, and M. Ratto. 2004. *Sensitivity analysis in practice: a guide to assessing scientific models*: John Wiley & Sons.

Saltelli, A., S. Tarantola, and K. P. S. Chan. 1999. A Quantitative Model-Independent Method for Global Sensitivity Analysis of Model Output. *Technometrics* 41 (1):39-56.

Samet, H. 1984. The quadtree and related hierarchical data structures. *ACM Computing Surveys (CSUR)* 16 (2):187-260.

Shashidharan, A., D. B. van Berkel, R. R. Vatsavai, and R. K. Meentemeyer. 2016. pFUTURES: A Parallel Framework for Cellular Automaton Based Urban Growth Models. *International Conference on Geographic Information Science* (pp. 163-177). Springer, Cham.

Shaw, S. L., H. Yu, and L. S. Bombom. 2008. A space-time GIS approach to exploring large individual-based spatiotemporal datasets. *Transactions in GIS* 12 (4):425-441.

Shepard, W. E. 2000. A parallel approach to searching for nearest neighbors with minimal interprocess communication. Doctoral dissertation, University of Georgia.

Shi, X. 2009. A geocomputational process for characterizing the spatial pattern of lung cancer incidence in New Hampshire. *Annals of the Association of American Geographers* 99 (3):521-533.

Shi, X. 2010. Selection of bandwidth type and adjustment side in kernel density estimation over inhomogeneous backgrounds. *International Journal of Geographical Information Science* 24 (5):643-660.

Shi, X., and S. Wang. 2015. Computational and data sciences for health-GIS. *Annals of GIS* 21 (2):111-118.

Shiode, S., and N. Shiode. 2013. Network-based space-time search-window technique for hotspot detection of street-level crime incidents. *International Journal of Geographical Information Science* 27 (5):866-882.

Shook, E., S. Wang, and W. Tang. 2013. A communication-aware framework for parallel spatially explicit agent-based models. *International Journal of Geographical Information Science* 27 (11):2160-2181.

Silverman, B. W. 1986. *Density estimation for statistics and data analysis*: CRC press.

Sobol, I. M. 1993. Sensitivity estimates for nonlinear mathematical models. *Mathematical Modelling and Computational Experiments* 1 (4):407-414.

Sobol, I. M. 2001. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and computers in simulation* 55 (1):271-280.

Sobol, I. M. 1967. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki* 7 (4):784-802.

Soliman, A., K. Soltani, J. Yin, A. Padmanabhan, and S. Wang. 2017. Social sensing of urban land use based on analysis of Twitter users' mobility patterns. *PLOS ONE* 12 (7):e0181657.

Stringer, C. E., C. C. Trettin, S. J. Zarnoch, and W. Tang. 2015. Carbon stocks of mangroves within the Zambezi River Delta, Mozambique. *Forest Ecology and Management* 354:139-148.

Survila, K., A. A. Yıldırım, T. Li, Y. Y. Liu, D. G. Tarboton, and S. Wang. 2016. A Scalable High-performance Topographic Flow Direction Algorithm for

Hydrological Information Analysis. *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale*, 1-7. Miami, USA: ACM.

Tang, W. 2008. Geographically-aware intelligent agents. Doctoral dissertation, University of Iowa.

Tang, W., and D. A. Bennett. 2010. Agent-based Modeling of Animal Movement: A Review. *Geography Compass* 4 (7):682-700.

Tang, W., and D. A. Bennett. 2011. Parallel agent-based modeling of spatial opinion diffusion accelerated using graphics processing units. *Ecological Modelling* 222:3605-3615.

Tang, W., and D. A. Bennett. 2012. Reprint of: Parallel agent-based modeling of spatial opinion diffusion accelerated using graphics processing units. *Ecological modelling* 229:108-118.

Tang, W., W. Feng, and M. Jia. 2015. Massively parallel spatial point pattern analysis: Ripley's K function accelerated using graphics processing units. *International Journal of Geographical Information Science* 29 (3):412-439.

Tang, W., W. Feng, M. Jia, J. Shi, H. Zuo, C. E. Stringer, and C. C. Trettin. 2017. A cyber-enabled spatial decision support system to inventory Mangroves in Mozambique: coupling scientific workflows and cloud computing. *International Journal of Geographical Information Science* 31 (5):907-938.

Tang, W., W. Feng, M. Jia, J. Shi, H. Zuo, and C. C. Trettin. 2016. The assessment of mangrove biomass and carbon in West Africa: a spatially explicit analytical framework. *Wetlands Ecology and Management* 24 (2):153-171.

Tang, W., and M. Jia. 2014. Global sensitivity analysis of a large agent-based model of spatial opinion exchange: A heterogeneous multi-GPU acceleration approach. *Annals of the Association of American Geographers* 104 (3):485-509.

Tang, W., and S. Wang. 2009. HPABM: A Hierarchical Parallel simulation framework for spatially-explicit Agent-Based Models. *Transactions in GIS* 13 (3):315-333.

Tang, W., S. Wang, D. A. Bennett, and Y. Liu. 2011. Agent-based modeling within a cyberinfrastructure environment: a service-oriented computing approach. *International Journal of Geographical Information Science* 25 (9):1323-1346.

Tao, R., J.-C. Thill, and I. Yamada. 2015. Detecting Clustering Scales with the Incremental K-Function: Comparison Tests on Actual and Simulated Geospatial Datasets. In *Information Fusion and Geographic Information Systems (IF&GIS'2015)*, 93-107: Springer.

Tarantola, S., M. Nardo, M. Saisana, and D. Gatelli. 2006. A new estimator for sensitivity analysis of model output: An application to the e-business readiness composite indicator. *Reliability Engineering & System Safety* 91 (10):1135-1141.

Terrell, G. R. 1990. The maximal smoothing principle in density estimation. *Journal of the American Statistical Association* 85 (410):470-477.

Tiwari, C., and G. Rushton. 2005. Using spatially adaptive filters to map late stage colorectal cancer incidence in Iowa. In *Developments in spatial data handling*, 665-676: Springer.

Tobler, W. R. 1970. A computer movie simulating urban growth in the Detroit region. *Economic geography* 46 (sup1):234-240.

Turton, I. 2000. Parallel processing in geography. In: Openshaw, S., Harris, T. and Abrahart, R. (eds), *GeoComputation*, Gordon and Breach.

Vardi, M. Y. 2010. Science has only two legs. *Communications of the ACM* 53 (9).

Varela, A., E. G. Aristizabal, and J. H. Rojas. 2010. Analisis epidemiologico de dengue en Cali. Cali: Secretaria de Salud Publica Municipal.

Wand, M. P., and M. C. Jones. 1994. *Kernel smoothing*: Crc Press.

Wang, S. 2008. Formalizing computational intensity of spatial analysis. *Proceedings of the 5th international conference on geographic information science*. Park City, UT, USA.

Wang, S., and M. P. Armstrong. 2003. A quadtree approach to domain decomposition for spatial interpolation in grid computing environments. *Parallel Computing* 29 (10):1481-1504.

Wang, S. 2009. A theoretical approach to the use of cyberinfrastructure in geographical analysis. *International Journal of Geographical Information Science* 23 (2):169-193.

Wang, S., M. K. Cowles, and M. P. Armstrong. 2008. Grid computing of spatial statistics: using the TeraGrid for G i*(d) analysis. *Concurrency and Computation: Practice and Experience* 20 (14):1697-1720.

Wang, S., and Y. Liu. 2009. TeraGrid GIScience gateway: bridging cyberinfrastructure and GIScience. *International Journal of Geographical Information Science* 23 (5):631-656.

Webber, T., J. F. C. L. Costa, and P. Salvadoretti. 2013. Using borehole geophysical data as soft information in indicator kriging for coal quality estimation. *International journal of coal geology* 112:67-75.

Wei, H., Y. Du, F. Liang, C. Zhou, Z. Liu, J. Yi, K. Xu, and D. Wu. 2015. A kd tree-based algorithm to parallelize Kriging interpolation of big spatial data. *GIScience & remote sensing* 52 (1):40-57.

Wilkinson, B., and M. Allen. 2004. *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers (Second Edition)*. Upper Saddle River, NJ USA: Pearson Prentice Hall.

Wright, J. K. 1936. A Method of Mapping Densities of Population: With Cape Cod as an Example. *Geographical Review* 26 (1):103-110.

Liu, Y.Y., D.R. Maidment, D.G. Tarboton, X. Zheng, and S. Wang. 2018. A CyberGIS Integration and Computation Framework for High-Resolution Continental-Scale Flood Inundation Mapping. *Journal of the American Water Resources Association* 1–15. https://doi.org/10.1111/1752-1688.12660.

Yamada, I., and P. A. Rogerson. 2003. An Empirical Comparison of Edge Effect Correction Methods Applied to K-function Analysis. *Geographical analysis* 35 (2):97-109.

Yamada, I., and J. C. Thill. 2007. Local Indicators of Network-Constrained Clusters in Spatial Point Patterns. *Geographical analysis* 39 (3):268-292.

Ye, S., H.-Y. Li, M. Huang, M. Ali, G. Leng, L. R. Leung, S.-w. Wang, and M. Sivapalan. 2014. Regionalization of subsurface stormflow parameters of hydrologic models: Derivation from regional analysis of streamflow recession curves. *Journal of Hydrology* 519:670-682.

Ye, X., S. Li, X. Yang, and C. Qin. 2016. Use of Social Media for the Detection and Analysis of Infectious Diseases in China. *ISPRS International Journal of Geo-Information* 5 (9).

Yin, J., Y. Gao, and S. Wang. 2017. CyberGIS-Enabled Urban Sensing from Volunteered Citizen Participation Using Mobile Devices. In *Seeing Cities Through Big Data: Research, Methods and Applications in Urban Informatics*, eds. P. Thakuriah, N. Tilahun and M. Zellner, 83-96. Cham: Springer International Publishing.

Zhang, C., L. Luo, W. Xu, and V. Ledwith. 2008. Use of local Moran's I and GIS to identify pollution hotspots of Pb in urban soils of Galway, Ireland. *Science of the total environment* 398 (1):212-221.

Zhang, G., A.-X. Zhu, and Q. Huang. 2017. A GPU-accelerated adaptive kernel density estimation approach for efficient point pattern analysis on spatial big data. *International Journal of Geographical Information Science*:1-30.

Zhang, J., and S. You. 2013. High-performance quadtree constructions on large-scale geospatial rasters using GPGPU parallel primitives. *International Journal of Geographical Information Science* 27 (11):2207-2226.

Zheng, M., W. Tang, Y. Lan, X. Zhao, M. Jia, C. Allan, and C. Trettin. 2018. Parallel Generation of Very High Resolution Digital Elevation Models: High-Performance Computing for Big Spatial Data Analysis. In *Big Data in Engineering Applications*, eds. S. S. Roy, P. Samui, R. Deo and S. Ntalampiras, 21-39. Singapore: Springer Singapore.

Zikopoulos, P., and C. Eaton. 2011. *Understanding big data: Analytics for enterprise class hadoop and streaming data*: McGraw-Hill Osborne Media.

APPENDIX: PSEUDOCODES

```
-----------------------------------------
ALGORITHM ST_IB(sCoord, tCoord, NN)
Inputs: sorted arrays of spatial and temporal coordinates, number of nearest ST neighbors to
search for
BEGIN ALGORITHM
   i = 0
   while i < len(sCoord):
      sNeigh = ordered array of spatial nearest neighbors for sCoord[i]
      tNeigh = ordered array of past temporal nearest neighbors for tCoord[i]
      stNeigh = intersect(sNeigh, tNeigh)
      sMax = max(sCoord[i] - stNeigh[0:NN])
      tMax = max(tCoord[i] - stNeigh[0:NN])
      sDist = sCoord[i] – sMax
      tDist = tCoord[i] – tMax
      i += 1
END ALGORITHM
-----------------------------------------
```

ST_IB algorithm: Computes spatial and temporal bandwidths for each disease case based on a specified number of neighboring cases. This pseudocode assumes knowledge of some widespread general functions, such as len(), intersect(), and max(), which are inspired from Python programming language. In addition, ordered arrays of nearest neighbors are found through K/D-tree indexing.

```
-----------------------------------------
ALGORITHM ST_STATIC_D (inX, inY, inT, xmin, xmax, ymin, ymax, tmin, tmax)
Inputs: arrays of spatiotemporal coordinates, domain boundaries
Global variables: mpt (maximum number of points threshold), brt (buffer ratio threshold), hs
(spatial bandwidth), ht (temporal bandwidth)
BEGIN ALGORITHM
   xDim = xmax - xmin
   yDim = ymax - ymin
   tDim = tmax - tmin
   sdVolume = xDim * yDim * tDim
   bufVolume = (xDim + 2 * hs) * (yDim + 2 * hs) * (tDim + 2 * ht)
   bufRatio = sdVolume / bufVolume
   if len(inX) <= mpt or bufRatio <= brt:
      writeToFile(inX, inY, inT)
   else:
      sXYT, sDom = assign(inX, inY, inT, xmax, xmin, ymax, ymin, tmax, tmin)
      decompose(sXYT[0], sXYT[1], sXYT[2], xmin, sDom[0], ymin, sDom[1], tmin, sDom[2])
      decompose(sXYT[3], sXYT[4], sXYT[5], sXYT[-3], xmax, ymin, sDom[1], tmin, sDom[2])
      decompose(sXYT[6], sXYT[7], sXYT[8], xmin, sDom[0], sDom[1], ymax, tmin, sDom[2])
      decompose(sXYT[9], sXYT[10], sXYT[11], sDom[0], xmax, sDom[1], ymax, tmin,
sDom[2])
      decompose(sXYT[12], sXYT[13], sXYT[14], xmin, sDom[0], ymin, sDom[1], sDom[2],
tmax)
      decompose(sXYT[15], sXYT[16], sXYT[17], sDom[0], xmax, ymin, sDom[1], sDom[2],
tmax)
      decompose(sXYT[18], sXYT[19], sXYT[20], xmin, sDom[0], sDom[1], ymax, sDom[3],
tmax)
      decompose(sXYT[21], sXYT[22], sXYT[23], sDom[0], xmax, sDom[1], ymax, sDom[2],
tmax)
END ALGORITHM
-----------------------------------------
```

ST_STATIC_D algorithm: Octree-based recursive decomposition of the spatiotemporal domain of point-data. Assumes known static spatial and temporal bandwidths. Maximum number of points threshold and buffer ratio threshold guide the granularity of the decomposition. The algorithm uses an ASSIGN function, which is specified below. The writeToFile function writes the spatiotemporal coordinates of the corresponding data points to a text file, therefore, stopping the decomposition procedure for the current branch of the tree.

```
----------------------------------------
ALGORITHM ASSIGN(inX, inY, inT, xmax, xmin, ymax, ymin, tmax, tmin):
Inputs: arrays of spatiotemporal coordinates, domain boundaries
Global variables: mpt (maximum number of points threshold), brt (buffer ratio threshold), hs
(spatial bandwidth), ht (temporal bandwidth)
BEGIN ALGORITHM
   xr2 = split_mid(xmin, xmax)
   yr2 = split_mid(ymin, ymax)
   tr2 = split_mid(tmin, tmax)
   sX1, sX2, sX3, sX4, sX5, sX6, sX7, sX8 = [], [], [], [], [], [], [], []
   sY1, sY2, sY3, sY4, sY5, sY6, sY7, sY8 = [], [], [], [], [], [], [], []
   sT1, sT2, sT3, sT4, sT5, sT6, sT7, sT8 = [], [], [], [], [], [], [], []
   for x, y, t in inX, inY, inZ:       # assign each data point to subdomain
      if x < xr2 - hs:
         if y < yr2 - hs:
            if t < tr2 - ht:
               sX1.append(x), sY1.append(y), sT1.append(t)
            elif t < tr2 + ht:
               sX1.append(x), sY1.append(y), sT1.append(t)
               sX5.append(x), sY5.append(y), sZ5.append(t)
            else:
               sX5.append(x), sY5.append(y), sT5.append(t)
         elif y < yr2 + hs:
            if t < tr2 - ht:
               sX1.append(x), sY1.append(y), sT1.append(t)
               sX3.append(x), sY3.append(y), sT3.append(t)
            elif t < tr2 + ht:
               sX1.append(x), sY1.append(y), sT1.append(t)
               sX3.append(x), sY3.append(y), sT3.append(t)
               sX5.append(x), sY5.append(y), sT5.append(t)
               sX7.append(x), sY7.append(y), sT7.append(t)
            else:
               sX5.append(x), sY5.append(y), sT5.append(t)
               sX7.append(x), sY7.append(y), sT7.append(t)
         else:
            if t < tr2 - ht:
               sX3.append(x), sY3.append(y), sT3.append(t)
            elif  < tr2 + ht:
               sX3.append(x), sY3.append(y), sT3.append(t)
               sX7.append(x), sY7.append(y), sT7.append(t)
            else:
               sX7.append(x), sY7.append(y), sT7.append(t)
      elif x < xr2 + hs:
         if y < yr2 - hs:
            if t < tr2 - ht:
               sX1.append(x), sY1.append(y), sT1.append(t)
               sX2.append(x), sY2.append(y), sT2.append(t)
            elif t < tr2 + ht:
               sX1.append(x), sY1.append(y), sT1.append(t)
               sX2.append(x), sY2.append(y), sT2.append(t)
               sX5.append(x), sY5.append(y), sT5.append(t)
```

```
            sX6.append(x), sY6.append(y), sT6.append(t)
        else:
            sX5.append(x), sY5.append(y), sT5.append(t)
            sX6.append(x), sY6.append(y), sT6.append(t)
    elif y < yr2 + hs:
        if t < tr2 - ht:
            sX1.append(x), sY1.append(y), sT1.append(t)
            sX2.append(x), sY2.append(y), sT2.append(t)
            sX3.append(x), sY3.append(y), sT3.append(t)
            sX4.append(x), sY4.append(y), sT4.append(t)
        elif t < tr2 + ht:
            sX1.append(x), sY1.append(y), sT1.append(t)
            sX2.append(x), sY2.append(y), sT2.append(t)
            sX3.append(x), sY3.append(y), sT3.append(t)
            sX4.append(x), sY4.append(y), sT4.append(t)
            sX5.append(x), sY5.append(y), sT5.append(t)
            sX6.append(x), sY6.append(y), sT6.append(t)
            sX7.append(x), sY7.append(y), sT7.append(t)
            sX8.append(x), sY8.append(y), sT8.append(t)
        else:
            sX5.append(x), sY5.append(y), sT5.append(t)
            sX6.append(x), sY6.append(y), sT6.append(t)
            sX7.append(x), sY7.append(y), sT7.append(t)
            sX8.append(x), sY8.append(y), sT8.append(t)
    else:
        if t < tr2 - ht:
            sX3.append(x), sY3.append(y), sT3.append(t)
            sX4.append(x), sY4.append(y), sT4.append(t)
        elif t < tr2 + ht:
            sX3.append(x), sY3.append(y), sT3.append(t)
            sX4.append(x), sY4.append(y), sT4.append(t)
            sX7.append(x), sY7.append(y), sT7.append(t)
            sX8.append(x), sY8.append(y), sT8.append(t)
        else:
            sX7.append(x), sY7.append(y), sT7.append(t)
            sX8.append(x), sY8.append(y), sT8.append(t)
else:
    if y < yr2 - hs:
        if t < tr2 - ht:
            sX2.append(x), sY2.append(y), sT2.append(t)
        elif t < tr2 + ht:
            sX2.append(x), sY2.append(y), sT2.append(t)
            sX6.append(x), sY6.append(y), sT6.append(t)
        else:
            sX6.append(x), sY6.append(y), sT6.append(t)
    elif y < yr2 + hs:
        if t < tr2 - ht:
            sX2.append(x), sY2.append(y), sT2.append(t)
            sX4.append(x), sY4.append(y), sT4.append(t)
        elif t < tr2 + ht:
            sX2.append(x), sY2.append(y), sT2.append(t)
```

```
                sX4.append(x), sY4.append(y), sT4.append(t)
                sX6.append(x), sY6.append(y), sT6.append(t)
                sX8.append(x), sY8.append(y), sT8.append(t)
            else:
                sX6.append(x), sY6.append(y), sT6.append(t)
                sX8.append(x), sY8.append(y), sT8.append(t)
        else:
            if t < tr2 - ht:
                sX4.append(x), sY4.append(y), sT4.append(t)
            elif t < tr2 + ht:
                sX4.append(x), sY4.append(y), sT4.append(t)
                sX8.append(x), sY8.append(y), sT8.append(t)
            else:
                sX8.append(x), sY8.append(y), sT8.append(t)
   sXYT = [sX1, sY1, sT1, sX2, sY2, sT2, sX3, sY3, sT3, sX4, sY4, sT4, sX5, sY5, sT5, sX6,
sY6, sT6,              sX7, sY7, sT7, sX8, sY8, sT8]
   sDom = [xr2, yr2, tr2]
      return sXYZ, sDom
END ALGORITHM
-----------------------------------------
```

ASSIGN algorithm: Helper function for spatiotemporal domain decomposition. Allocates each point (disease case) to the respective subdomain(s), depending on their location. This particular example uses the SPLIT_MID function (defined below), which performs static midway splits of each dimension for the ST-STATIC-D algorithm. The SPLIT_MID function can be replaced with SPLIT_FLEX (see below) for implementing ST_FLEX_D_BASE, ST_FLEX_D_UNEVEN and ST_FLEX_D_ALTERNATE algorithms.

```
-----------------------------------------
ALGORITHM SPLIT_MID(dmin, dmax)
Inputs: domain boundaries
BEGIN ALGORITHM
return (dmax + dmin)/2
END ALGORITHM
-----------------------------------------
```

SPLIT_MID algorithm: Simply returns the average of two numbers (midway point).

```
-----------------------------------------
ALGORITHM SPLIT_FLEX(inList, max, min, buf, level):
Parameter 1: input coordinates (1D), Parameter 2: maximum of coordinate range, Parameter 3:
minimum of coordinate range, Parameter 4: buffer distance
BEGIN ALGORITHM
   for each point in in List
      for each candidate split
         check whether circle is cut, keep track of count
   if minimum number of cut circles is tied between two or more candidate splits
      pick split that balances partitions more evenly
      if balance tied between two or more candidate splits
         pick split that is more central
END ALGORITHM
-----------------------------------------
```

SPLIT_FLEX algorithm: Finds split that cuts the minimum number of circles among a predefined number of candidate splits. For ST_FLEX_D_BASE and ST_FLEX_D_ALTERNATE (see below) candidate splits are chosen in equal intervals, for ST_FLEX_D_UNEVEN, they are chosen to congregate around the midway split.

```
-----------------------------------------
ALGORITHM ST_FLEX_D_ALTERNATE(inX, inY, inT, dim):
Inputs: arrays of spatiotemporal coordinates, domain boundaries
Global variables: mpt (maximum number of points threshold), brt (buffer ratio threshold), hs
(spatial bandwidth), ht (temporal bandwidth)
BEGIN ALGORITHM
   xDim = dim[1] - dim[0]
   yDim = dim[3] - dim[2]
   tDim = dim[5] - dim[4]

   decompDim = argmax(xDim, yDim, tDim)

   sdVolume = xDim * yDim * tDim
   bufVolume = (xDim + 2 * hs) * (yDim + 2 * hs) * (tDim + 2 * ht)
   bufRatio = sdVolume / bufVolume

   if len(inX) <= mpt or bufRatio <= brt:
      writeToFile(inX, inY, inT)
   else:
      sXYZ = assign(inX, inY, inT, dim, decompDim)
      decomp(sXYZ[0], sXYZ[2], dim)
      decomp(sXYZ[1], sXYZ[3], dim)
END ALGORITHM
-----------------------------------------
```

ST_FLEX_D_ALTERNATE algorithm: Binary tree-based recursive decomposition of the spatiotemporal domain of point-data, promoting compact subdomains. Assumes known static spatial and temporal bandwidths. Maximum number of points threshold and buffer ratio threshold guide the granularity of the decomposition. The ASSIGN function is adapted to the binary decomposition in a straightforward manner.