University of California
Los Angeles

# Scalable, Synthetic, Sensor Network Data Generation

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

**Yan Yu**

2005

The dissertation of Yan Yu is approved.

Greg Pottie

Stefano Soatto

Songwu Lu

Ramesh Govindan

Deborah L. Estrin, Committee Chair

University of California, Los Angeles

2005

*To my mom*

# Table of Contents

# List of Figures

# List of Tables

# Acknowledgments

I would like to take this opportunity to acknowledge all the help and support I have received over the years. No amount of words can do this adequately.

First of all, I am extremely fortunate to have Deborah Estrin and Ramesh Govindan as my thesis advisors. One could not possibly ask for better advisors and mentors.

Deborah is an inspiration in many ways. She has excellent vision and deep insight, incredible enthusiasm for her students' research, and abundant energy. She has selflessly lent her support throughout the years. She has patiently guided me along my research path. She has taught me to think about research problems with broad perspective, and at the same time has encouraged me to study practical problems that impact the real world. I am also grateful for her excellent advice regarding presentations, writings and many other non-technical endeavors. It is an honor and privilege to be her student and to work with her. She has been my role model and will remain so for years to come.

In no less measure, I must thank Ramesh. Ramesh is an excellent researcher who maintains high standards for his work and his students. I would be delighted if after graduation I can still live up to his and Deborah's high standards. Ramesh has handled my progress with great patience and sensitivity. His excellent advice and his attentiveness to details has greatly helped my growth as a researcher. His rigorous approach to research and his incredible insight to many research problems has benefitted me tremendously. I will always be grateful to Ramesh for his guidance, his perspective, and constant encouragement and support.

I wish to thank the other members of my thesis committee: Professors Songwu

Lu, Greg Pottie, and Stefano Soatto for their invaluable feedback and encouragement. I also wish to thank professors Mark Hansen and Yingnian Wu for their very helpful consultation on statistical problems and Professor Richard Guy who has offered me helpful feedback on my defense talk dry run and has managed the projects in our lab.

Professor Bill Kaiser deserves special mention. I appreciate that he trusted me to work on the NIMS project. He is always encouraging along every step of my progress, no matter how small it is and is very responsive to students. He has incredible energy in assuming many roles: from drawing plans for future projects to attending to every detail of existing ones. I have learned a lot from working with him and working on the NIMS projects.

I thank the professors at UCLA who generously allowed me to audit their classes: Professors Coen Bernaards, Eddie Kohler, Mark Hansen, Miodrag Potkonjak, Majid Sarrafzadeh, Rick Schoenberg, Yingnian Wu, Alan Yuille, and Song Chun Zhu. I especially thank professor Coen Bernaards for demonstrating that statistics is interesting. Coen's class is the first statistics class I took at UCLA. I learned from professor Yingnian Wu what an art is to be able to explain statistical problems in a way that is understandable to anyone. From Eddie, I learned that systems can be taught with passion and found depth in systems research. His insight into system problems has shown me that systems research can be fun and has strengthened my interest in the field even though i did not start from there.

At AT&T Research (Shannon Lab), I thank my mentors, Steve Bellovin, Ramon Caceres, Kathleen Fisher, and Anne Rogers for their patient guidance, excellent feedbacks and availability over the course of my summer internship.

I also would like to thank all the professors I have taken classes from and

interacted with while a graduate student at USC. In particular, I would like to thank Professors John Heiderman, Sandeep Gupta and Jon Gratch. I am always grateful to Jon for his generous support for my first two years at USC, and for selflessly encouraging me to pursue my interests and transfer research groups. I also thank Jon for introducing me to the research world and for always having the best interests of his students in mind. I thank Sandeep for his responsiveness and excellent feedback. In collaborating with John in the SCADDS project and an ns-related project, John has offered much excellent feedback, and this thesis is written using his latex thesis template.

Of course, I could not have accomplished anything without the support from all my fellow students and lab-mates:

from USC: Nirupama Bulusu, Alberto Cerpa, Jeremy Elson, Deepak Ganesan, Lewis Girod, Ahmed Helmy, Polly Huang, Chalermek Intanagonwiwat, Satish Kumar, Pavlin Radoslavov, Reza Rejaie, Fabio Silva, Hongsuda Tangamunarunkit, Ya Xu, Wei Ye, Jerry Zhao;

from UCLA: Solomon Bien, Nirupama Bulusu, Naimasaranya Busek, Vladimir Bychkovskiy, David Braginsky, Alberto Cerpa, Deepak Ganesan, Lewis Girod, Benjamin Greenstein, Moshe Golan, Obi Iroezi, Ruhul Kapur, Martin Lukac, Andrew Parker, Denis Perelyubskiy, Mohammad Rahimi, Nithya Ramanathan, Thomas Schoellhammer, Thanos Stathopoulos, Hendra Tjahayadi, Hanbiao Wang, other people in LECS; Richard Pon, Steve Liu, Maxim Batalin, Anitha Vijayakumar, Jason Gordon, Lisa Shirachi, Rachel Scollans and other people in NIMS.

I thank all of them for many useful discussions, help and friendship. I would like to especially thank Lew and Deepak, who have patiently listened to my unbaked ideas and given me excellent feedback. Lew's integrity to research and pragmatic approach rubbed off on me. I also learned enormously from his system

insights. I thank Mohammad for many useful discussions, kind help and encouragement throughout our collaboration on the NIMS projects. I thank Hanbiao for timely consulting on statistical problems. I have bugged many people in the lab for proofreading my papers: Ben, Alberto, Andrew, Ruhul, Nithya, Tom, Lew and others. I appreciate their generous help and always being there for me. Niru, Jeremy, and Deepak had served as examples of LECS graduates. I had referenced Jeremy and Niru's dissertation many times.

I would like to thank all my friends for all the help and fun time together. It helps me remain sane in this long graduate study.

Last, and most importantly, I would like to thank my family. They are the reasons for who I am, and they are always encouraging and there for me in good times and bad times. My mother is the kindest person I have ever known. I am grateful to her for her unselfish love and for always believing in me, even in many occasions when I did not. Hearing that I have passed my defense certainly makes her much happier than even I am, so I dedicate this thesis to my mom. I thank my father for equally unselfish love and for always being concerned about my health and happiness. My brother has become my dearest friend over the years. I thank him for patiently listening to my complaints and always encouraging and cheering. My family has given me more love than I deserve.

I wish I could enumerate all the kind people that have lent their support to me over the years, but i am sure I missed many of them on this short list. I wish I could pass on their spirits and kindness to other people in the future.

# Vita

| | |
|---|---|
| 1997 - 1999 | Graduate Student Researcher, USC/ISI |
| 1999 - 2000 | Graduate Student Researcher, USC |
| Summer 2000 | Summer Research Intern at AT&T Research (Shannon Lab, Florham Park, NJ) |
| 2000 - 2004 | Graduate Student Researcher, Laboratory for Embedded Collaborative Systems and Center for Embedded Networked Sensing, UCLA |
| Jan 2005 | Graduated, UCLA |

# Publications

**Yan Yu**, Deborah Estrin, Ramesh Govindan and Mohammad Rahimi "Using more realistic data models to evaluate sensor network data processing algorithms." *First IEEE Workshop on Embedded Networked Sensors 2004.*

Maxim A. Batalin, Mohammad Rahimi, **Yan Yu**, Duo Liu, Aman Kansal, Gaurav S. Sukhatme, William J. Kaiser, Mark Hansen, Gregory J. Pottie, Mani

Srivastava, Deborah Estrin "Call and Response: Experiments in Sampling the Environment." *Sensys'04.*

**Yan Yu**, Deepak Ganesan, Lewis Girod, Deborah Estrin and Ramesh Govindan "Synthetic data generation to support irregular sampling in Sensor Networks." in *Geo Sensor Networks* 2003 Oct 9-11 2003, Portland Maine.

Deepak Ganesan, Alberto Cerpa, **Yan Yu**, Wei Ye, Jerry Zhao and Deborah Estrin "Networking Issues in Sensor Networks in Sensor Networks." To appear in the *Journal of Parallel and Distributed Computing (JPDC)*, Special issue on Frontiers in Distributed Sensor Networks, Elsevier Publishers.

A. Helmy, S. Gupta, D. Estrin, A. Cerpa, **Yan Yu** "Systematic Performance Evaluation of Multipoint Protocols." Proceedings of *FORTE/PSTV*, IFIP, Kluwer Academic Publication, Pisa, Italy, October 2000.

**Yan Yu**, ZhuoqunXu "History Viewer: an Adaptive and Cooperative Agent in WWW Browsers." *Proceedings of Hong Kong Beijing International Computing Conference'97, Hong Kong 1997.*

**Yan Yu**, Ramesh Govindan and Deborah Estrin "Geographical and Energy Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks." *UCLA Computer Science Department Technical Report* UCLA/CSD-TR-01-0023, May 2001.

**Yan Yu**, Deborah Estrin, Sandeep Gupta "Worst-Case Performance Analysis

of Wireless Ad-hoc Routing Protocols: Case Studies." *USC Computer Science Department Techical Report* 00-730, May 2000.

**Yan Yu**, Deborah Estrin, Ramesh Govindan and Fabio Silva "Region Based Multi-point Data Dissemination in Wireless Sensor Networks." (In submission)

Abstract of the Dissertation

# Scalable, Synthetic, Sensor Network Data Generation

by

**Yan Yu**

Doctor of Philosophy in Computer Science

University of California, Los Angeles, 2005

Professor Deborah L. Estrin, Chair

Sensor networks are a new class of distributed systems composed of a large number of densely deployed sensors, actuators, low power computation, and wireless communication devices. Sensor network research is still in its infancy. There is a large volume of exploratory research; however, few real systems are deployed and little experimental data from sensor networks is available to test proposed protocol design. Due to lack of experimental data and sophisticated models derived from such data, most data processing algorithms from the sensor network literature are evaluated with data generated from simple models.

Through statistical performance analysis and algorithm evaluation using experimental data, we demonstrate the need to evaluate algorithms using realistic data corresponding to a wide range of parameter values and irregular topology input. Motivated out of a concern for data sensitivity and a lack of experimental data, we propose a scalable synthetic data generation framework to support systematic algorithm evaluation and robust algorithm design.

We have two strategies to achieve scalable synthetic data generation. First, using case studies of concrete sensor network algorithms we identify a small number

of parameters to manipulate in our synthetic data generation. This significantly reduces the search space from exponential to a manageable number. Second, we use empirical models derived from experimental data to guide simulation towards those portions of the space that represent real world scenarios. Guided by these two strategies, we implement algorithms to generate synthetic data either with similar characteristics as the experimental data, or exhibiting a wide range of data characteristics. In addition to the irregular topology data generation toolbox, we also provide ready-to-use test suites to encourage people to utilize more realistic data when evaluating their algorithms. Moreover, the sensor data replay addition to EmStar enhances EmStar simulation / emulation with more realistic sensor data traces.

# CHAPTER 1

# Introduction

## 1.1 Data processing algorithms in sensor networks

Due to the advances in miniaturization and MEMS technology, we have witnessed a tremendous amount of interest and progress in sensor networks in the last few years. Wireless sensor networks will bring unprecedented capabilities to many applications, such as environmental sensing, monitoring and control of structural health and seismic activity, and disaster prevention and response. Sensor networks is a new class of extremely distributed systems that are composed of a large number of densely deployed sensors, actuators, low power computation, and wireless communication devices. Potential for large scale, dense deployment enables the possibility for data sampling at a much higher density; while being close to the physical phenomena allows much higher fidelity (*e.g.*, higher SNR) sampling than what was possible before, providing unprecedented opportunities and challenges for many applications.

With large scale, dense deployment in the physical world, huge amounts of sensing data will be available. Instead of transferring and presenting this raw data to the end user, processed information and a high level representation of the phenomena are sometimes more useful. For example, the end users or applications may be more interested in object trajectories [ZSR02], the object iden-

tifications (*e.g.*, as applied to identifications of bird species [WYP04]), or data statistics [NGA04], rather than raw sensor data at each node.

Further, together with untethered and unattended systems the small form factor imposes severe energy and configuration constraints. With the current trend of battery technology, energy will remain a stringent constraint in the foreseeable years to come. Due to the energy constraints, energy efficiency has been an important consideration in designing sensor network algorithms and systems. Therefore, in-network data processing techniques are preferred and necessary even when we are interested in data collection. For example, in order to achieve a long-lived data collection system, there have been enormous interests in designing efficient data sampling and data aggregation techniques, including various types of data compression algorithms [KIR04, WMN04, MN04]. Therefore, data processing algorithms play an essential role in sensor networks.

## 1.2 State of the art in current data processing algorithm evaluation

Sensor network research is still in its infancy. Few real systems are deployed and little experimental data from sensor networks is available to test proposed protocol design. Due to lack of experimental data and sophisticated models derived from such data, most data processing algorithms from the sensor network literature are evaluated with data generated from simple parametric models. For example, it is common that data collection and estimation algorithms are evaluated with uniform or Gaussian data input [DNB04, RN04]. Similarly, a random Gaussian field model is typically used in both analytical and simulation studies to evaluate the compression and source coding algorithms [GCB04].

Many of these data processing algorithms are sensitive to data input. If this sensitivity only produces small perturbations on the algorithm performance, simple parametric models used in the simulation help to clarify the algorithm behavior without the distraction from the details of data input. Unfortunately, the type of input data and node topology used in the evaluation often significantly affect the evaluation results, as elaborated by the following examples.

In the first example, we demonstrate that for a given performance accuracy requirement, the performance of one algorithm may appear to be acceptable for certain data but unacceptable for another input data set.

We evaluate a uniform sampling based median computation algorithm [NG] (Section 4.1.1) against four data sets: data generated from uniform, Gaussian, and Bimodal distributions, and a one-day snapshot from the S-Pol radar data set[1]. Our performance metric here is communication cost, which is proportional to the sampling rate. To achieve the same precision, the particular experimental data set we used and the bimodal distributed data require a sample size of at least one order of magnitude more than Gaussian and uniformly distributed data. Specifically, in order to obtain a similar estimation precision as uniformly distributed data at the cost of a 1% sampling rate, it requires an 8 - 10% sampling rate in the case of the experimental data. Furthermore, to obtain similar precision as a normally distributed data set at the cost of a 1% sampling rate, the experimental data requires a 30% sampling rate.

The significantly higher cost in the case of an experimental data set and a bi-modally distributed data set may suggest that the median computation by uniform sampling algorithm is not feasible for these two data distributions. The

[1]S-Pol (S band polar metric radar) data were collected during the International H2O Project (IHOP; Principal Investigators: D. Parsons, T. Weckwerth, et al.). S-Pol is fielded by the Atmospheric Technology Division of the National Center for Atmospheric Research. We acknowledge NCAR and its sponsor, the National Science Foundation, for provision of the S-Pol data set.

results from using our experimental data suggest that the median is underrepresented, thereby the median estimated from a uniform sampling will not work well in practice.

In the second example we will show that the relative performance order of two algorithms may change depending on different data inputs.

As reported in Section 6.1, we compare two field estimation algorithms, Fidelity Driven Sampling vs. Raster Scan using two types of data: (1) data generated from simple parametric models and (2) data collected from a lab environment. For each data distribution we compare the reconstruction accuracy of Fidelity Driven Sampling vs. Raster Scan as a function of *the number of samples used in the estimation*. In terms of the Mean Squared Error (MSE) in the reconstruction results, for data generated from linear and quadratic models, the MSE achieved by Fidelity Driven Sampling is several orders of magnitude lower than raster scan; whereas for a data set collected from a lab environment the MSE achieved by Fidelity Driven Sampling is higher than raster scan.

In the third example we demonstrate that algorithm evaluations over a regular grid and over an irregular topology may lead to different conclusions.

We use the DIMENSIONS [GEH02] system as our case study and investigate the impact of irregular topologies on algorithm performance. DIMENSIONS provides a unified view of data handling in sensor networks, incorporating long-term storage, multi-resolution data access, and spatio-temporal pattern mining. It is designed to support observation, analysis, and querying of distributed sensor data at multiple resolutions while exploiting spatio-temporal correlations. In evaluating the DIMENSIONS system, we use two types of data: (1) Irregular topology data generated from modeling the spatio-temporal correlation in the experimental data; and (2) Snapshots, in regular grid, from 50km resolution daily precip-

itation data for the Pacific NorthWest from 1949-1994 [WC]. Our evaluation results demonstrate that DIMENSIONS evaluated in an irregular setting exhibit different behavior than in a regular configuration (see Section 8.1 for details). Using data from a regular grid, as the query proceeds down the DIMENSIONS hierarchy, gaining access to more detailed information, as expected, the mean squared error drops gradually. In an irregular configuration, however, there is no consistent error improvement with more levels of drill down.

We would like to clarify that the objective of our case studies is not to criticize a particular algorithm in any way. Rather our objective is to demonstrate some potential problems in the evaluation methodology followed by many sensor network researchers, *i.e.*, evaluating algorithms only based on data generated from simple models.

Given that different data inputs can change algorithm performance dramatically, evaluating the system with data representing real-world scenarios or corresponding to a wide range of conditions is essential for systematic algorithm evaluation and robust design of sensor network systems.

Most existing experimental data are obtained from remote sensing or collected from in-situ instrumentation. Unfortunately, most of them are collected from a regular grid whereas the sensor networks are most likely deployed in an irregular topology. Therefore, they cannot be directly used to evaluate the sensor network algorithms.

Motivated out of a concern for data sensitivity problems and a lack of experimental data described above, we propose to generate irregular topology data from models derived from the experimental data or data across a range of parameters in order to support robust algorithm design.

## 1.3 Key Challenges of the Synthetic Data Generation

In this section, we briefly explain the challenges in synthetic data generation, and discuss how we address them in this thesis.

**Scalability Issues** The parameter space is huge in fully characterizing a physical phenomenon. The following back-of-the-envelop calculation demonstrates that the search space in the synthetic data generation is at least exponential. A spatial data set with $m$ possible sensor readings each at $n$ locations has $m^n$ possibilities in the synthetic data output. Further, sensor data input is often 8 or 16 bits, in which case, $m = 2^8$ or $m = 2^{16}$. Thus, systematic algorithm evaluation by exhaustive search is not practical.

**Reality Check** Ideally algorithm evaluation using our synthetic data will provide insight on system performance in future field deployments; therefore, the synthetic data is desirable to represent real world scenarios. We want to avoid the pathological scenarios in which the synthetic data misguides the simulation.

**Unavailability of Ideal Experimental Data** Most existing experimental data are collected from a regular grid, however, sensor networks are usually deployed in an ad-hoc manner. Evaluating algorithms with irregular topology data versus regular grid data could lead to completely different results. Therefore, data in regular grid topology alone is not sufficient to evaluate algorithm performance.

Most available experimental data has spatial sampling resolution that is below the Nyquist rate. This creates challenges for generating fine-grained synthetic data that can resemble real-world phenomena and match the deployment density of future sensor networks.

Corresponding to the first two challenges identified above, we propose two strategies. First, through concrete case studies, we identify a small number of parameters essential to algorithm performance. This will significantly reduce the search space in synthetic data generation. Second, we use empirical models derived from experimental data to guide simulation towards those portions of the space that represent real world scenarios. Guided by these two strategies, we implement algorithms to generate synthetic data of an arbitrary irregular topology.

In this thesis, we do not directly address the problem of generating fine-grained synthetic data based on models derived from sparse experimental data. As discussed in Section 7.2.4, if the phenomenon itself is smooth (*i.e.*, the spatial correlation function is continuous near the separation distance 0), then our synthetic data can approximate the spatial correlation of the real phenomena at a small scale. Without prior knowledge on the spatial correlation at small scales, we propose to vary the spatial correlation at small scales across a wide range of parameter values to achieve systematic algorithm evaluation.

## 1.4 Contributions of this Thesis

To support algorithm evaluation with realistic data input across a wide range of parameter values, we propose techniques to generate irregular topology data. Data sensitivity and algorithm evaluation requiring better models are not unique to sensor networks. Our major contributions are in identifying the extent of this problem through statistical performance analysis and evaluation in the context of sensor networks. Further, we propose a scalable synthetic data generation framework to support systematic algorithm evaluation and robust algorithm design. Our contributions fall into the following categories:

**Search Space Reduction** Using case studies of concrete sensor network algorithms we identify a small number of parameters to manipulate in our synthetic data generation. This reduces the search space significantly from exponential to a manageable number.

In this thesis, we investigate three classes of data processing algorithms that are potentially sensitive to data input. Through statistical performance analysis and algorithm evaluation using experimental data, we identify a small set of data characteristics essential to algorithm performance. Furthermore, we demonstrate the need to evaluate algorithms using realistic data corresponding to a wide range of parameter values and irregular topology input.

**Systematic performance evaluation techniques** In our case studies, we apply two types of performance evaluation techniques. (1) When the algorithm performance can be attributed to a single parameter, we use standard non-parametric statistical tools to examine how the algorithm performance varies with the changing parameter value. Identifying the parameter essential to algorithm performance often requires knowledge of the algorithm under evaluation. (2) When the algorithm performance depends on multiple parameters, we use synthetically generated scenarios to investigate how each parameter changes the algorithm performance. In the synthetically generated scenarios, we have the flexibility to vary the phenomena along a single dimension while keeping other parameters fixed.

**Techniques to Generate Irregular Topology Data** We propose two types of synthetic data generation techniques. First, we use empirical models derived from experimental data to guide synthetic data generation towards those portions of the parameter space that represent real world scenarios. The synthetic data

generation techniques in this category will create irregular topology data with similar characteristics as the experimental data. Second, when the available experimental data is scarce, we provide methods to generate synthetic traces exhibiting a wide range of data characteristics over the parameter of interest to applications.

**Implementation** Our deliverable is an implementation of a synthetic data generation toolbox, which consists of eight different spatial interpolation algorithms and techniques that allow users to directly adjust the spatial correlation and data distribution in the synthetic data. This toolbox automatically generates irregular topology data from the given experimental data. In addition, we also provide suites of experimental data and irregular topology data created using the aforementioned synthetic data generation techniques. These ready-to-use test suites will encourage people to utilize more realistic data input when evaluating their algorithms.

EmStar [GEC04] is an integrated simulation and runtime environment for developing embedded sensor network systems. EmStar provides a rich library and tool support to facilitate the development of mobile embedded systems. EmStar has been used in multiple institutions and corporations in their sensor network research. In order to incorporate our work into EmStar, we have added sensor data replay support in EmStar and are in the process of adding a sensor noise model to EmSim. This addition supports EmStar simulation / emulation with more realistic sensor data traces.

## 1.5 Dissertation Overview

Chapter 2 reviews related work on data modeling techniques in various contexts: data modeling in environmental science, databases and data mining; and TCP traffic modeling in the Internet. We then examine a proposal by Floyd *et al.* for better models of traffic on the Internet.

In Chapter 3, we identify three classes of data processing algorithms that are potentially sensitive to input data: statistics estimation, data compression, and field estimation. These three classes of algorithms will be studied in detail in the subsequent chapters. Results are summarized here: we identify a small number of parameters essential to algorithm performance. This provides two benefits for a scalable synthetic data generation framework. First, identifying a small number of parameters reduces the search space in the synthetic data generation; second, it makes synthetic data evaluation manageable, because it provides a set of quantitative metrics that allow us to directly evaluate the synthetic data. Motivated by this, we discuss parameterized synthetic data generation and evaluation in Chapter 7.

Chapter 4 discusses the statistical estimation applications. We study two types of percentile computation algorithms. Statistical performance analysis demonstrates that, in both cases, the algorithm performance changes significantly when evaluated using different data input. In particular, this is the case when the corresponding percentile bin size varies across a wide range. Ideally, we would like to demonstrate that a small number of data characteristics are sufficient to determine the algorithm performance which will significantly reduce the search space in the synthetic data generation. In the case of median computation by uniform sampling, we prove that estimation accuracy depends only on a single parameter, namely, the normalized median bin size.

Chapter 5 discusses the data compression algorithms themselves. In particular, we study the joint entropy coding algorithm and an instance of a wavelet compression algorithm. The statistical performance analysis demonstrates a strong correlation between the algorithm performance and spatial correlation in the data.

Chapter 6 discusses an instance of field estimation algorithm, namely, Fidelity Driven Sampling. Compared to using data simulated from simple models, evaluating the Fidelity Driven Sampling algorithm with experimental data changes its relative performance order to Raster Scan. This indicates that evaluation results from data input that are solely based on simple parametric models may be misleading. Further, through synthetic scenarios, we demonstrate that the smoothness and the spatial structure of the phenomena play an essential role in the algorithm performance.

Given the significant performance dependency on data input, an ideal input to algorithm evaluation is experimental data that represent various real-world scenarios. Unfortunately, often, this ideal experimental data is not available; therefore, we propose to generate synthetic data that can approximate the experimental data, as well as data corresponding to a range of parameter values. In Chapter 7, we present our scalable synthetic data generation framework. The scalability is achieved by two strategies: focus on the experimental data, and focus on a small number of essential parameters. Corresponding to these foci, we discuss two types of synthetic data generation techniques: (1) techniques to generate synthetic data based on empirical models of the experimental data; (2) algorithms to generate data sets that are rich in the data characteristics essential to algorithm performance. The latter is useful when the available experimental data is scarce and the scope of the parameter space it covers is not sufficient to evaluate the algorithm performance across a wide range of data input.

In Chapter 8, we examine the utility of the synthetic data through algorithm evaluation using our synthetic data. In the case of Fidelity Driven Sampling and median computation, synthetic data provide similar results as the experimental data. In the case of the DIMENSIONS system, synthetic data allows algorithm evaluation over irregular topology. The performance difference in regular data and irregular topology demonstrates the need to improve the standard wavelet compression algorithm to handle irregular data.

Finally, in Chapter 9, we summarize our work and discuss future directions.

# CHAPTER 2

# Related Work

To the best of our knowledge, before the start of this dissertation, there was no previous attempt in data modeling and synthetic data generation in a sensor network context. However, there exists research on data modeling in other contexts, or modeling other system components with the objective of creating a realistic simulation environment.

In Section 2.1, we briefly review the data modeling and data analysis techniques in environmental science. In particular, we examine the spatial interpolation and modeling techniques, time series analysis, and joint space-time modeling techniques. Our synthetic data generation work greatly benefit from these data modeling techniques. In Section 2.2, we review data modeling and synthetic data generation in the database and data mining research, and discuss why they are not applicable for our purpose. Similarly, in Section 2.3 we discuss why TCP traffic modeling techniques for the Internet do not directly apply to generating data input for a sensor network application.

In Section 2.4 we review related work on modeling various simulation system components in the wireless ad-hoc and sensor networks. They share the same objective of creating a realistic simulation environment with our work; however, they did not directly target modeling data input.

In Section 2.5 we discuss related work on identifying the need for better models

for Internet research. Finally, in Section 2.6 we discuss a recently proposed synthetic data generation approach by Jindal *et al.* [JP04].

## 2.1 Data modeling techniques in environmental science

In environmental science or geophysics, various data analysis techniques have been applied to extract interesting statistical features from the data, or estimate sensor values at un-sampled or missing data points. Various spatial interpolation techniques, such as *Voronoi polygons*, *triangulation*, *natural neighbor interpolation*, *trend surface* or *splines* [WO01], have been proposed. Kriging, which refers to a family of generalized least-squares regression algorithms, has been used extensively in various environmental science disciplines. Kriging models the spatial correlation in the data and minimizes the estimation variance under the unbiased constraints of the estimator. In this thesis, we explored Kriging and several non-stochastical interpolation techniques.

In addition, significant research has been devoted to time series analysis. Autoregressive Integrated Moving Average model (ARIMA) [BJ76] explicitly considers the trend and periodic behavior in the temporal data. The wavelet model [FBS02] has been successfully used to model the cyclic, or repeatable behavior in data. In addition, researchers have also explored neural networks [Dor96] and kernel smoothing for time series analysis.

Joint spatio-temporal models have received much attention in recent years [KJ99, SBG, Sch, Oga98] because they inherently model the correlation between the temporal and spatial domain. The joint space-time model used in our data analysis is inspired by and simplified from a joint space-time model proposed by Kyriakidis *et al.* [KMK02]. In [KMK02], co-located terrain elevation values are used

to enhance the spatial prediction of the coefficients in the temporal model constructed at each gauge station. However, this requires the availability of an extra environmental variable, which does not exist in our case.

In our study, to generate synthetic data that can approximate the experimental data in terms of data features of interest to applications, we borrow heavily from geostatistics and spatial interpolation techniques.

## 2.2 Data modeling in Database and Data Mining

Theodoridis *et al.* [TN00] proposes to generate spatio-temporal datasets according to parametric models and user-defined parameters. However, because the parameter space is huge, it is impossible to exhaustively search the entire parameter space, *i.e.*, generate data sets corresponding to every possible set of parameter values. Without additional knowledge, we have no reason to believe that any parameter setting is more realistic or more important than others. Therefore we propose to start with an experimental data set and generate synthetic data that shares similar statistics with the experimental data.

Given a large data set that is beyond computer memory constraints, DuMouchel *et al.* [WCT99] propose data squashing techniques to shrink a large data set to a manageable size. Although sharing the same objective of deriving synthetic data from modeling existing data as ours, they consider non-spatio-temporal data. Under the assumption that the likelihood of a particular value is a relatively smooth function of the rest of the data, [WCT99] applies Taylor expansion on the likelihood functions. Moreover it assumes that a data set is the result of $N$ independent draws from the same probability model. Spatial and temporal stationarity often do not hold for an arbitrary physical random process.

As a result, the spatio-temporal data cannot be assumed to be drawn from the same probability model as assumed by [WCT99].

## 2.3 TCP traffic modeling in the Internet

In a similar attempt to model the input data to a network system in the context of the Internet, researchers have studied TCP traffic modeling. For example, Caceres *et al.* [CDJ91] characterized and built empirical models of wide area network applications. The specific data modeling technique in their study [CDJ91] does not apply to sensor networks due to the following: (a) Sensor networks are closely coupled with the physical world; therefore, data modeling in sensor networks needs to capture the spatial and temporal correlation in a highly dynamic physical environment; (b) the characteristics of wide area TCP traffic are potentially very different from the workload or traffic in sensor networks. Internet TCP traffic is the superposition of many TCP connections. However, sensor networks tend to be specially designed and used for one or a few applications. Sensor network traffic is often triggered by physical phenomena and the output of applying signal processing algorithms to physical phenomena. Due to reasons stated above, the approaches proposed by [CDJ91, AA00] may not be able to capture highly dynamical physical environment in which sensor networks are deployed.

## 2.4 System components modeling in wireless ad-hoc and sensor networks

### 2.4.1 Channel models

Previous research has been carried out on modeling system components in wireless ad-hoc and sensor networks; however, most existing research focuses on modeling communication channels. In modeling wireless channels, Konrad *et al.* [KZJ01] study non-stationary behavior of packet loss in the wireless channel and model the Global System for Mobile (GSM) traces with a Markov based Trace Analysis (MTA) algorithm. In the sensor network context, several experimental studies [CWK04, ZG03, WTC03] investigated the large discrepancy between the experimentally observed communication properties and what had been widely used in the simulation studies, *e.g.*, symmetric, circle disk models. Zhao *et al.* [ZG03] demonstrated a high variability in packet reception rate for a wide range of distances between a transmitter and a receiver. Cerpa *et al.* [CWK04] apply non-parametric statistical techniques to build an empirical pdf of a certain feature value (*e.g.*, packet reception rate) given alternative feature values. Moreover, they develop a series of realistic radio channel models based on comprehensive experimental traces.

### 2.4.2 Mobility models

Among research on topology and mobility models in the context of ad-hoc networks, [BMJ98, DPR00] use regular or uniform topology setups and "random waypoint" models in their protocol evaluations. In a typical "random waypoint" model, each node begins the simulation by remaining stationary for *pause time* period. It then selects a random destination and moves to that destination at a

speed uniformly distributed in a certain range, and pause again before moving to the next destination. In the context of evaluating routing protocols, [JLH99] discuss multiple topology setups and mobility patterns for more realistic scenarios. Specifically, they discuss three types of mobility patterns. (1) Conference scenarios represent low mobility patterns, where only 10% of the nodes are moving at any moment in time. The purpose of this scenario is to test responsiveness to local changes of long-lived routes. (2) Event coverage scenarios represent high mobility patterns, where at any moment 50% of the nodes move with a speed of 1 m/s. The objective of this scenario is to test the ability to respond to fast topology changes and fluctuating traffic. (3) Disaster area scenarios include diverse mobility patterns, where 95% of the nodes have low mobility, and the rest have very high mobility. The purpose of this scenario is to study the protocol behavior with diverse node speeds and with the presence of network partitions.

### 2.4.3 Simulators for wireless ad-hoc and sensor networks

Ns-2[BBE99] and GloMoSim [ZBG98] provide flexibility in simulating various layers of wired networks or wireless ad-hoc networks. However, they do not capture many important aspects of sensor networks, such as sensor models or channel models. In contrast, Sensorsim [PSS00, PSS01] directly targets sensor networks. In addition to a few topology and traffic scenarios, they introduce the notion of a sensor stack and sensing channel. The sensor stack is used to model the signal source, and the sensing channel is used to model the medium which the signal travels through. Their work mostly focuses on point source sensor models and exponential channel loss models. These models may capture point source phenomena, such as contaminant transport monitoring; however, it is not applicable to environmental phenomena in general. Our work could be used as a

new model in Sensorsim.

## 2.5 Proposal on better input models in the context of the Internet

In the context of Internet research, Floyd *et al.* [FK02] use examples drawn from Active Queue Management and TCP variants to illustrate the problems caused by inappropriate models. Inappropriate or incomplete models may cause flaws in the protocol design, or generate simulation results different from what you would see in today's Internet. It identifies the need for a richer understanding of the range of realistic models and the relevance of different model parameters to network performance. It proposes to build models based on existing measurement results, and to generate new empirical results when needed. Although proposed in a different context, they share similarity to our work in that both identify the significant influence that input models have on the algorithm performance; both propose to use empirical models to guide the simulation to focus on scenarios representative of real world situations. Our work differs from theirs in the following two aspects: First, we model different subjects; they propose to model topology and traffic mix patterns in an Internet application; whereas, we model sensor data input in a sensor network system. Consequently, due to reasons given in Section 2.3, their modeling techniques will not apply to sensor network contexts. Secondly, through concrete case studies on a few widely studied data processing algorithms, we identify a small number of parameters essential to algorithm performance. This significantly reduces the search space in synthetic data generation.

## 2.6 Synthetic data generation in sensor networks

The work in [JP04] proposes a mathematical model to capture the spatial correlation in sensor network data, and to generate large synthetic traces from a small experimental trace. This synthetic data generation technique can be easily incorporated into our proposed synthetic data generation framework. However, as pointed out in [YGG03], we lack ground truth data to verify that the synthetic data matches the statistics of the experimental data at fine scales. Consequently, we do not recommend using models derived from a few sensor nodes to generate a large trace of fine granularity unless the phenomena that we are trying to capture is known to be smooth at small scales.

# CHAPTER 3

# Identifying a small number of parameters for a scalable synthetic data generation

In this chapter, we first identify a few widely-studied classes of data processing applications that are potentially sensitive to input data: statistical estimation, data compression, and field estimation applications. We then introduce the methodologies that we will use in the next three chapters and summarize our major findings. In the end, we point out the implications of these findings for a scalable synthetic data generation framework.

We identify three widely-studied classes of applications that are potentially sensitive to data input:

- Statistical estimation of the field data, *e.g.*, median, percentile, or density estimation, of which we examined median and percentile estimation.
- Data compression, *e.g.*, distributed source coding, entropy coding, or coding schemes that exploit the spatial and temporal correlation in the data. We studied an instance of wavelet compression and of joint entropy coding algorithm.
- Field estimation. There has been extensive research in sampling and reconstruction of a physical field [RPE04, WMN04, BKV04, KIR04] in recent sensor network literature. We investigated fidelity driven sampling in our

case study.

In this study, the above algorithms serve as case studies in systematically studying the dependency of algorithm performance on data. Although the above list of sensor network algorithms is not exhaustive, they are selected as our case studies for the following reasons. First, many sensor network systems are deployed to monitor and understand the physical environment. Due to energy constraints, often sensor network applications can not afford to transmit every bit of sensed information back to the base station continuously. Therefore, statistical summary, data compression, and efficient field estimation algorithms are essential to building a long-lived system. Second, the algorithms in the above three categories all need to exploit the statistics and redundancy in the data in some manner and therefore are potentially sensitive to data input.

Next we briefly describe the evaluation methodologies used in our case studies. For the case studies in the first two classes of applications, we identify a single parameter that the algorithm performance depends on. Given multiple experimental data sets with various values for this parameter, we systematically study how the algorithm performance changes. Our statistical analysis consists of **three** steps. Given a specific problem and a particular algorithm, *first*, we define a performance metric for the given algorithm; *second*, we identify a data characteristic relevant to the algorithm. Taking median computation as an example, we define the performance metric to be *estimation error* and we identify the relevant data characteristic to be the *normalized median bin size*, which will be explained in the next chapter. *Third*, we apply standard non-parametric statistical techniques in our performance analysis. Evaluated against a S-Pol radar data set (Section 1.2), results from both *scatter plot* and *Pearson's correlation coefficient* indicates a strong correlation between the identified performance met-

ric and the identified data characteristics. At the same time, the scatter plots demonstrate that the *algorithm performance* varies significantly with respect to the identified *data characteristics*.

In the third category of applications, we studied the Fidelity Driven Sampling algorithm [RPE04]. Algorithm evaluation using data simulated from simple models and a few experimental data sets demonstrate how different data input may change its relative performance order compared to a simple alternative, namely, raster scan. This suggests that algorithm evaluation using data generated from simple models may be misleading.

The algorithm performance is affected by multiple data characteristics. Experimental data could arbitrarily change along these multiple dimensions. Fortunately, in the synthetically generated scenarios, we have the flexibility to vary the phenomena along a single dimension. Therefore, using synthetically generated scenarios we identify two parameters important to the Fidelity Driven Sampling algorithm: the *total mean curvature* and *spatial structure* of the phenomena.

In summary, these case studies serve the following purposes:

First, they serve as examples of how to systematically study the dependency of algorithm performance on data. Through afore-mentioned three-step statistical analysis, we can identify the regime in which the algorithm is applicable or perform well.

Second, the evaluation results from these case studies demonstrate that different data input could change the algorithm performance significantly. For example, in the case of median computation, the estimation accuracy was changed by one order of magnitude when evaluated with different data input. Further, the relative performance order among different algorithms even changes. This strongly suggests evaluating algorithms across a range of data input and moti-

vates our synthetic data generation framework.

Last, and most important, through these case studies we identify a small number of parameters essential to algorithm performance. This provides two benefits for a scalable synthetic data generation framework. First, identifying a small number of parameters significantly reduces the search space in the synthetic data generation. Second, it makes synthetic data evaluation manageable because it provides a set of quantitative metrics that allows us to directly evaluate the synthetic data.

Figure 3.1 shows parameters essential to the algorithms in our study. The blocks in the *applications* row list three classes of applications. The blocks in the *algorithms* row list the specific algorithms we investigated in our case studies. We identify a performance metric for each algorithm, listed above the corresponding algorithm block. The identified interesting data characteristics are listed on the top row. A one-way arrow from data characteristic $D$ to algorithm $A$ indicates that $D$ is essential to the performance of the corresponding algorithm. A two-way arrow assures *sufficiency*, *i.e.*, the particular set of data characteristics under study are sufficient to determine the algorithm performance.

In median computation and percentile estimation applications, we identify that data distribution, specifically, its $p$-percentile bin size and nodes spatial topology are essential to the algorithm performance. In the case of Joint entropy coding and Wavelet compression, we identify spatial correlation as an important parameter. In the case of Fidelity Driven Sampling, we identify total mean curvature and spatial structure of the phenomena as essential to the algorithm performance.

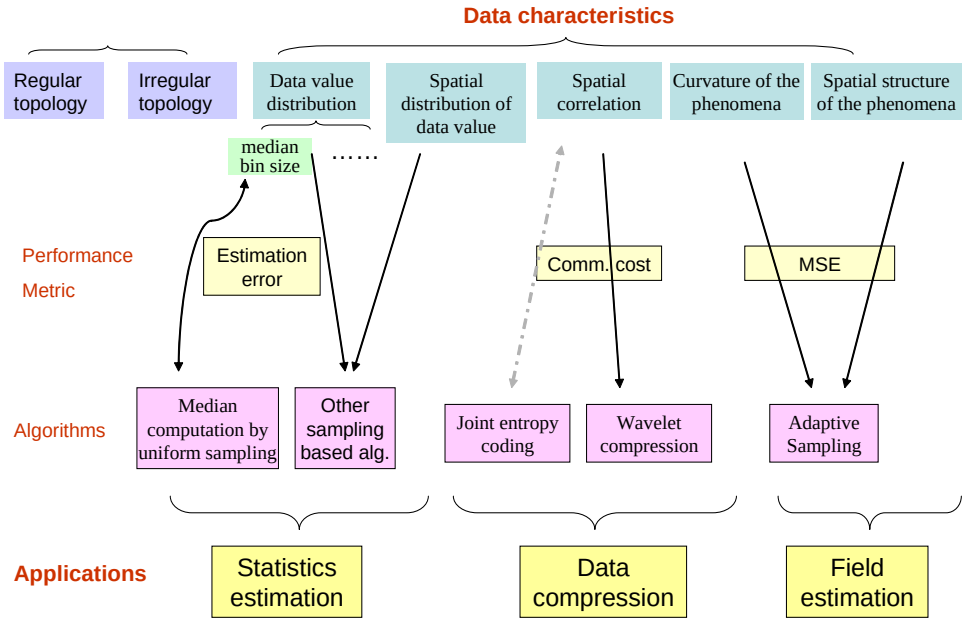



Figure 3.1: Diagram of parameter space

# CHAPTER 4

# Statistics Estimation Applications

In this chapter we consider a type of application in which we are able to identify a single parameter that determines the algorithm performance. In particular we investigate two statistical estimation applications in our case studies, median computation and percentile estimation.

We start with a definition of the statistical estimation problem. A random process, $Z$, which consists of a set of random variables $Z(u)$, one for each sensor location $u$ in the deployment area $A$, denoted as $\{Z(u), \forall u \in A\}$. Assuming $Z$ is a stationary process, the problem is to estimate some statistical measure (*e.g.*, the median) of $Z(u)$ ($u \in A$) given a realization of $\{Z(u)\}$ (*i.e.*, a time snapshot of sensor readings) at locations $u_i$, $i = 1, ..., n$, $u_i \in A$.

We study two types of percentile estimation algorithms. In our case studies, we identify a single parameter essential to the algorithm performance, namely, the *p-percentile* bin size. We investigated how the estimation accuracy is affected by data characteristics across a wide spectrum. Statistical performance analysis demonstrates that, in both cases, the corresponding *p-percentile* bin size is essential to algorithm performance. In the case of *percentile estimation by uniform sampling*, we prove that estimation error is determined by the corresponding *p-percentile* bin size.

In section 4.1.1 and section 4.1.2, we study the median computation and

the *p-percentile* estimation algorithms by random sampling, respectively. Using the experimental data, the statistical analysis demonstrates a strong correlation between the estimation error and the identified data characteristics, *i.e.*, the normalized *p-percentile* bin size. Estimation error decreases with the increasing median bin size and varies significantly across a wide range of data input.

To demonstrate that the performance evaluation approach we used is applicable to the scope beyond percentile computation by random sampling in section 4.1.3 we apply the same performance evaluation techniques to Power-Conserving Computation of Order-Statistics proposed in [GK04], a percentile estimation algorithm not based on uniform sampling methods. Results from our statistical analysis once again confirm the above findings. In particular, the *p-percentile* bin size is an essential parameter to the algorithm performance.

In section 4.2, we provide proof that the *p-percentile* bin size is sufficient to determine the algorithm performance. This sufficiency proof significantly reduces the search space in synthetic data generation.

Motivated by the finding that algorithm performance could change dramatically for different data input, in section 4.3, to explicitly reduce the data sensitivity we design an improvement to the median computation by random sampling algorithm. The evaluation results demonstrate that the performance of our proposed algorithm is not sensitive to data input. It may be difficult to remove data sensitivity from an arbitrary algorithm. Nevertheless, this simple improvement to the median computation by random sampling algorithm exemplifies that systematic evaluation of algorithm performance lends insight to a robust algorithm design.

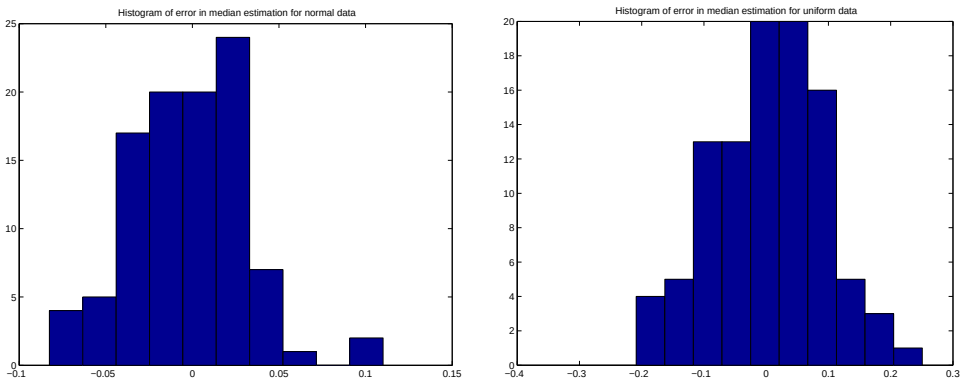
## 4.1 Sensitivity of algorithm performance on data and Identification of essential data characteristics

### 4.1.1 Median computation by uniform sampling

For simplicity of illustration, here we consider an instantiation of random process $Z(u)$ as deterministic. Thus, given a snapshot of sensor readings at each node, $z(i)$, $i = 1$, ...,$n$, the real median $M(Z)$ is defined as $z(\frac{n+1}{2})$, if $n$ is odd, or $(z(\frac{n}{2}) + z(\frac{n}{2} + 1))/2$ if $n$ is even. The estimated median is written as $\hat{M}(Z)$.

There are variations of median estimation by uniform sampling. Without loss of generality, in this paper we consider a specific median computation by uniform sampling algorithm as follows: (1) Suppose there is a single sink in the deployed sensor network and each sensor node has the same probability (*e.g.*, 1%) of sending its reading back to the sink. (2) Within this scenario, the sensor value is forwarded along the shortest path tree from the sensor source to the sink. Further, an intermediate node on the shortest path tree will not prune samples, instead it simply relays these packets back to the sink. (3) Suppose p samples are transmitted back to the sink, s(1),..., s(p), we then compute the median, M(S) of s(1),..., s(p) as an estimate of $M(Z)$.

Before systematically evaluating how the algorithm performance is affected by different data input, we use four example data sets to demonstrate that different data input can dramatically affect the algorithm performance. Specifically, we consider data sets generated from uniform, Gaussian, Bi-modal distributions, and S-Pol radar data sets. Provided by NCAR, the S-Pol radar data records the intensity of reflectivity in dBZ, where Z is proportional to the returned power for a particular radar and a particular range. We selected 259 time snapshots across 2 days in May 2002 in our study. Each snapshot is a 60 x 60 spatial grid data




(a) Results on normally distributed data

(b) Results on uniformally distributed data

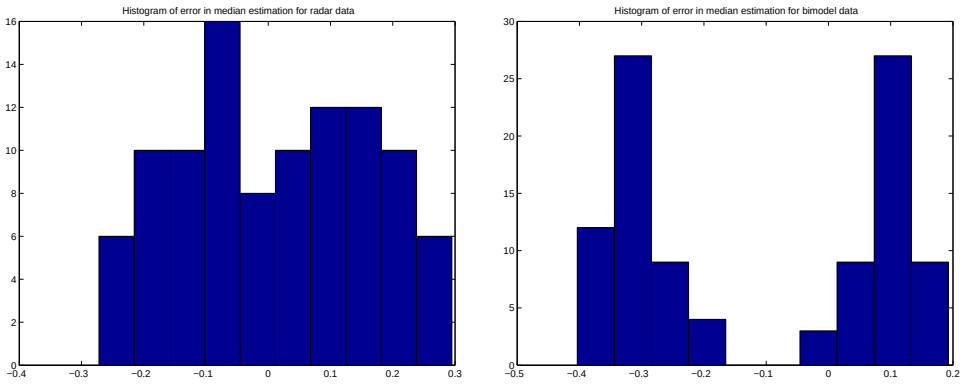
Figure 4.1: Histograms of estimation error on data of normal or uniform data distribution are close to the normal bell shape (x-axis: normalized estimated median error; y-axis: number of occurrence)

with 1 km spacing.

We identify the performance metric of the median computation algorithm to be **normalized estimated median error**. We define this to be *the difference between the estimated median, $\hat{M}(Z)$, and the real median, $M(Z)$*, normalized by *the range of the entire set of sample values*. Normalization is introduced to make comparing results across different data sets meaningful. Median is often used as a robust estimator instead of mean. Therefore, we define the error metric in terms of value, as opposed to position. [1]

The histogram of normalized estimation error derived from using the four

[1] If the estimation error metric is defined in terms of order (*i.e.*, let $p$ denote the real position of the estimated median $\hat{M}(Z)$ in the original data set, $n/2$ is the position of the real median, the estimation error is then defined as $p - n/2$). We can prove (see Section 4.4) that median computation by uniform sampling is not sensitive to data distribution. Intuitively, uniform sampling is applied in the spatial dimension. When the error metric is defined in the same dimension, *i.e.*, position in an ordered list or space, the estimation error will not be sensitive to the underlying data distribution.




(a) Results on S-Pol radar data

(b) Results on data of bimodal distribution

Figure 4.2: Histograms of estimation error for radar data and bimodal distribution are widely spread out or bi-modal. The average estimation error is also larger than that from Gaussian and uniform distributions. (x-axis: normalized estimated median error; y-axis: number of occurrence)

data sets is presented in Figures 4.1 and 4.2. For normal data (Figure 4.1(a)) and uniform data input (Figure 4.1(b)), the estimation error distribution is close to a normal distribution. Whereas for the S-Pol radar data (Figure 4.2(a)), and bimodal data (Figure 4.2(b)) both the mean and variance of estimation error are higher than that from the other two distributions. Above we use four data sets as an example to illustrate to what extent the median computation by uniform sampling algorithm can be sensitive to data distribution. Next, using statistical tools we systematically investigate how the algorithm performance varies across a wide range of data distributions.

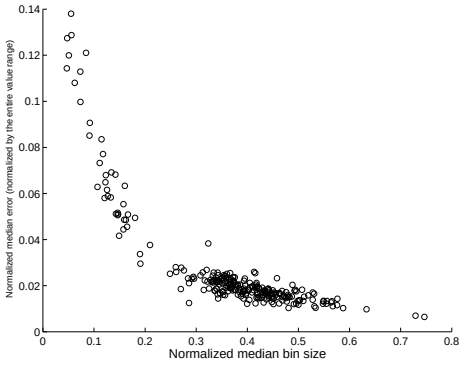
Our statistical analysis consists of three key steps. First, we define our performance metric to be *normalized median estimation error*. Second, we identify the data characteristic that is most relevant to the median computation algo-

rithm to be *normalized median bin size*. We bin the entire set of samples into a fixed number (*e.g.*, 10) of equally spaced containers. *Median bin* is defined as the container that includes the median. Let $n$ denote the total number of samples, and $m$ denote the number of samples in the *median bin*, then the **normalized median bin size** is defined as $m/n$, *i.e.*, the ratio of the size of the median bin relative to the size of the entire data set.

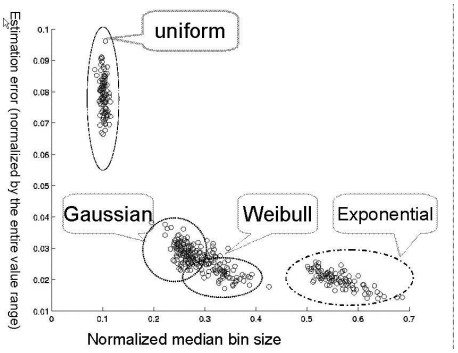
As a final step, we study how the algorithm performance changes with data input across a wide range of parameter values. We evaluate the algorithm using two types of data: data simulated from Gaussian, Exponential, and Weibull distributions; and 259 snapshots of S-Pol radar data. Note that data generated from simple models have been widely used in previous algorithm evaluations. We vary the parameters in the above models across a wide range: in the Gaussian distribution, we fix the mean and vary the standard deviation from 1 to 100; in the Weibull distribution, we fix the scale parameter and vary the shape parameter from 1 to 100; and in the Exponential distribution, we vary the exponential changing rate $\lambda$ from 0.1 to 10.

Figure 4.3 shows the scatter plot of *normalized estimated median error* vs. *normalized median size*. Each point in the graph corresponds to results from one data set. The x-axis is the *normalized median bin size of the data* and the y-axis is the *normalized estimation error* averaged from 100 runs of algorithm on the corresponding data set.

As shown in Figure 4.3, for both experimental data and data generated from parametric distributions, the algorithm performance (*i.e.*, the normalized median error) is well correlated with our defined data characteristic, namely, *the normalized median bin size*. With increasing normalized median bin sizes, the estimation error decreases. Intuitively, under uniform sampling with increasing normalized




(a) Results on experimental data, Correlation Coefficient= -0.8239




(b) Results on data generated from Gaussian, uniform, Weibull and exponential distributions; Correlation Coefficient= -0.7818

Figure 4.3: Scatter plot of normalized estimated median error vs. normalized median size. The normalized median error is well correlated with the normalized median bin size. With increasing normalized median bin size, the estimation error decreases. Further, the experimental data covers a super set of all 4 families of data distributions.

median bin size, more samples from the median bin will appear in the final sample set at the sink. Therefore, there is a higher chance that a sample from the *median bin* will be selected as the estimated median at the sink as opposed to samples from other bins. Because samples from the same bin are close in value, the *estimated median* will be close to the real median in value.

We also compute the correlation coefficient of *the normalized median estimation error* and *the normalized median bin size*. It is $-0.8239$ and $-0.7818$ for the data in Figure 4.3(a) and 4.3(b) respectively. The fact that the absolute value of the correlation coefficients are close to 1 also indicates the strong correlation between them.

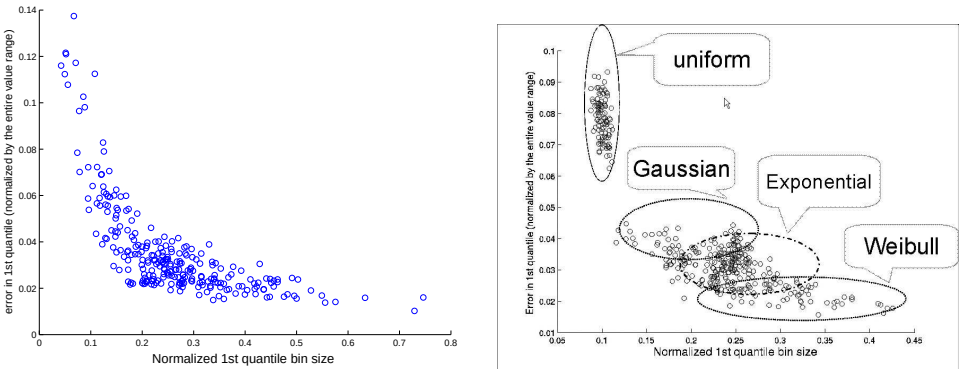
We would like to point out another interesting observation from Figure 4.3: In terms of normalized median bin size, experimental data encompasses a super set of all four families of data distributions. This may suggest that it would be difficult to cover a wide range of data characteristics if we only use data generated from simple parametric distributions in our algorithm evaluation. The experimental data sets cover a wide range of data characteristics not covered by any single distribution. We believe this result strongly suggests the importance of realistic data in algorithm evaluations.

### 4.1.2 Percentile computation by uniform sampling

Without loss of generality, we define percentile as follows: for an ordered data set, $z(i)$, $i = 1$, ..., $n$, its $p$ percentile is defined as $z(\lfloor p * n \rfloor)$. In percentile computation by uniform sampling, each sensor node has the same probability of sending its reading back to the sink. An intermediate node simply relays the sample along the shortest path tree back to the sink. If $t$ samples are eventually transmitted back to the sink, $s(1)$,..., $s(t)$, we compute the $p$ percentile of $s(1)$,..., $s(t)$ as an estimate of the $p$ percentile of the original sample set, $z(1)$, ..., $z(n)$.

Similar to median computation by uniform sampling, we identify the data characteristics as *normalized p-percentile bin size*: the entire sample set is divided into a fixed number of equally spaced containers. The *p-percentile bin* is defined as the container that includes the *p-percentile*. Let $n$ denote the total number of sensor readings and $n_p$ denotes the number of samples in the *p-percentile bin*; then the *normalized p-percentile bin size* is defined as $n_p/n$.

Here we presented results on *1st-quartile* (*i.e.*, 25-percentile). We evaluate the algorithm using the same sets of data as in the median computation. Figure 4.4 shows the scatter plot of normalized estimation error vs. normalized




(a) Results on experimental radar data, Correlation Coefficient=-0.6796

(b) Results on data generated from Gaussian, Uniform, Weibull and Exponential distributions, Correlation Coefficient=-0.8729

Figure 4.4: *1st*-quartile computation results: the scatter plot of normalized estimation error vs. normalized *1st*-quartile bin size. The estimation error increases when the *1st*-quartile bin size decreases. Again, experimental data covers a super set of all four families of data distributions.

1st-quartile bin size. We observe the same performance trend as in the case of median computation (see Figure 4.3), *i.e.*, the estimation error increases when the corresponding quartile bin size decreases. This can be explained by the fact that the corresponding quartile is less represented. We also studied the *3rd*-quartile estimation and obtained similar results as in Figure 4.4. We leave out the details here due to the space limitations.

### 4.1.3 PCCOS: an order-statistics estimation algorithm not based on random sampling

Through the above case studies, we demonstrated the key steps to systematically study how the algorithm performance changes with various data input:

- Define a performance metric for the particular algorithm under evaluation, *e.g.*, normalized estimation error in the above case study.
- Identify the set of data characteristics most relevant to the algorithm in the study, *e.g.*, normalized $p$-percentile bin size in the case study above.
- Statistically study how the algorithm performance varies with changing data characteristics. Scatter plots and correlation coefficients were used to identify the correlation between the performance metric and data characteristics in the study.

The above systematic performance evaluation technique may not be applicable to all statistical estimation algorithms. However, to demonstrate that it is applicable to the scope beyond the statistical estimation by uniform sampling algorithm, we apply the statistical analysis above to Power-Conserving Computation of Order-Statistics proposed in [GK04]. We used the same problem definition, performance metric, and data characteristics as defined in Section 4.1.1, to evaluate PCCOS with the S-Pol radar data. Similar to the case of the median computation by uniform sampling, we observed a strong correlation between the estimation accuracy and the normalized median bin size (Figure 4.5). It is evident that the variance of estimation error is larger when the normalized median bin size is small, in the range between 0.1 and 0.3.

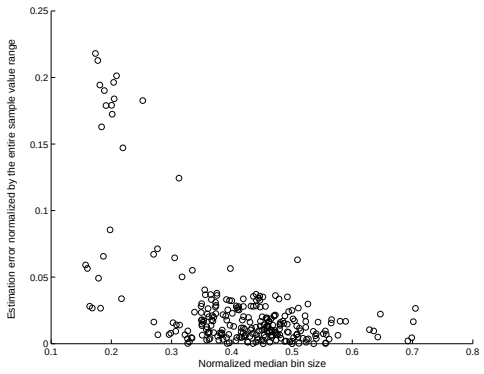



Figure 4.5: Median computation results from PCCOS algorithm on radar data: scatter plot of estimation error vs. normalized median bin size indicates a strong correlation between them; Correlation coefficient = -0.58

## 4.2 Parameter space reduction: identifying a small set of characteristics sufficient to the algorithm performance

Through statistical performance analysis we have demonstrated that the corresponding *p-percentile* bin size is an essential parameter to the algorithm performance. Our ideal goal is to demonstrate that a few data characteristics are sufficient to determine the algorithm performance. This could significantly reduce the search space in synthetic data generation. To generate a spatial data set consisting of sensor readings at $n$ locations and each sensor reading has $m$ possible outcomes, there are $m^n$ possibilities in the synthetic data output. Further, if a sensor reading is quantized in 8 or 16 bits, $m = 2^8$ or $m = 2^{16}$. Therefore, identifying a small number of parameters sufficient to determine the algorithm performance reduces this exponential search space to a tractable number. Thus our synthetic data generation can safely vary only in a few dimensions sufficient

to determine the algorithm performance. As a result, this sufficiency proof provides algorithm designers with confidence to sufficiently evaluate algorithm using our synthetic data.

In the case of median computation by uniform sampling, we prove that estimation accuracy depends only on a single parameter, namely, the normalized median bin size. The proof is as follows:

Assume that the original sample population is of size $n$, we take $l$ samples out of $n$. Let $x_p$ denote the median from these $l$ samples and $m$ denote the real median from the original population. $P(|x_p - m| = \epsilon R) = P((x_p - m) = \epsilon R) + P((m - x_p) = \epsilon R)$ where $\epsilon$ is the normalized estimation error and $R$ is the range of the entire samples.

$P((x_p - m) = \epsilon R) = P($ there exists $\frac{l}{2}$ samples $s_i$ satisfying $s_i \geq (m + \epsilon * R))$

Let $Y_i$ denote a random variable and $Y_i = 1$ if sample $s_i \geq m + \epsilon * R$

Thus, $P(Y_i = 1) = \frac{1}{2} - \delta_R$ where $\delta_R = \frac{\text{number of samples } s_i, s_i \in [m, m+\epsilon*R)}{\text{total number of samples}}$.

$P(Y_i = 0) = \frac{1}{2} + \delta_R$

The event "there exists $\frac{l}{2}$ samples $s_i$ satisfying $s_i \geq (m + \epsilon * R)$" is a bernoulli trial. Thus we have

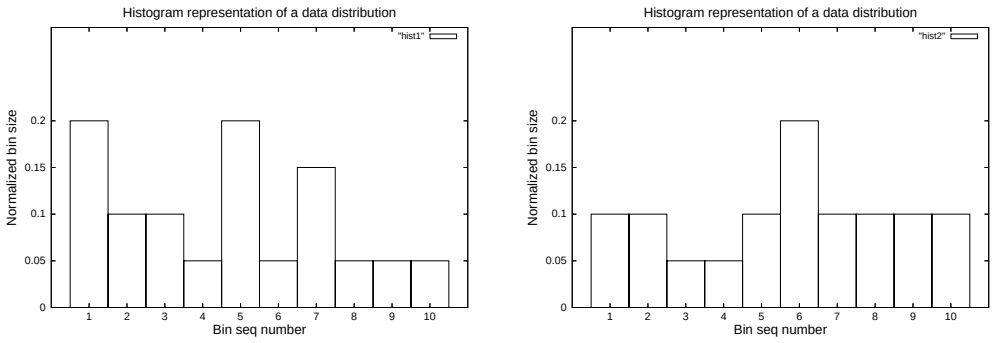
$P($ there exists $\frac{l}{2}$ samples $s_i$, satisfying $s_i \geq (m + \epsilon * R)) =$

$$\binom{l}{\frac{l}{2}} * (\frac{1}{2} - \delta_R)^{\frac{l}{2}} * (\frac{1}{2} + \delta_R)^{(l - \frac{l}{2})} \tag{4.1}$$

Similarly, $P((m - x_p) = \epsilon * R) =$

$$\binom{l}{\frac{l}{2}} * (\frac{1}{2} - \delta_L)^{\frac{l}{2}} * (\frac{1}{2} + \delta_L)^{(l - \frac{l}{2})} \tag{4.2}$$

where $\delta_L = \frac{\text{number of samples } s_i, s_i \in [m - \epsilon*R, m]}{\text{total number of samples}}$.




(a) The median falls in the 5th bin (b) The median falls in the 6th bin

Figure 4.6: Two data distributions with the same normalized median bin size.

$$P(|x_p - m| = \epsilon R) =$$

$$\binom{l}{\frac{l}{2}} * (\frac{1}{2} - \delta_R)^{\frac{l}{2}} * (\frac{1}{2} + \delta_R)^{(l-\frac{l}{2})} + \binom{l}{\frac{l}{2}} * (\frac{1}{2} - \delta_L)^{\frac{l}{2}} * (\frac{1}{2} + \delta_L)^{(l-\frac{l}{2})} \quad (4.3)$$

Note that Equation 4.3 only depends on the number of samples $l$, $\delta_L$ and $\delta_R$. If $[m - \epsilon * R, m + \epsilon * R]$ is considered as the bin that contains the median, then $\delta_L + \delta_R$ can be considered as the normalized median bin size.

Equation 4.3 demonstrates that the probability distribution of the normalized estimation error depends only on the sample density, relative to the entire population, surrounding the median. This *normalized sample density* is measured by the *normalized median bin size*. The estimation error distribution does not depend on how data are distributed in other portions of the sample space. For example, for the two data distributions shown in Figure 4.6, despite their different distributions as illustrated in their histogram representations, statistically they deliver similar estimation accuracy in median computation with the same number of samples.

Even though these two distributions are dramatically different, the normalized median bin size are the same in both; therefore, the error distribution of the

estimated median in these two data sets are similar. Note that the above proof can be easily generalized to *p-percentile* estimation by replacing $\frac{1}{2}$ with $p$.

The above proof demonstrates that the normalized median bin size is sufficient to determine the performance of *median computation by random sampling*. Thus, to generate synthetic data for evaluating *median computation by random sampling*, we can safely vary the data along a single parameter, namely, the normalized median bin size. As a result, this significantly reduce the search space in the synthetic data generation.

## 4.3 Case study of systematic evaluation leading to a robust algorithm design: Median computation by selective sampling

Whenever possible, it is desirable to reduce the dependency of algorithm performance on data. As demonstrated in Section 4.1.1, *median computation by uniform sampling* is sensitive to data input. In this section we introduce *median computation by selective sampling*. We are *not* trying to propose the best available median computation algorithm. Instead, this approach serves as an example algorithm that explicitly takes data distribution into consideration and reduces the dependency of the algorithm performance on data distribution.

We first describe a primitive - called *selective sampling* - used in the algorithm. This procedure selects $m$ samples from a sample set $S$, where $m \leq |S|$. The *selective sampling* procedure is described in Algorithm 1. The output of this procedure are $m$ pairs, ($W_1$, $M_1$), ($W_2$, $M_2$), ..., ($W_m$, $M_m$), as an representative of $S$. Next we briefly describe the median computation by *selective sampling* algorithm:

**Algorithm 1** Selective Sampling($S$, $m$)

1: $S$ : sample set
2: $m$ : integer ▷ number of samples selected from $S$
3: Sort the samples in $S$;
4: Divide $S$ into $m$ equal-ranged containers: $S_1$, $S_2$, ..., $S_m$;
5: **for** $i = 1$ to $m$ **do**
6: $M_i$ = median of $Si$;
7: $W_i = |S_i|$ ▷ the number of elements in $Si$
8: **end for**
9: Output $< M_i, W_i >$: $i = 1, ..., m$

- At the leaf node, each node reports its sensor reading, $s$. This sensor reading is sent along the shortest path tree back to the sink.

- At an intermediate node $A$, suppose $A$ has $k$ children. Each child reports a list of ($W_i$, $M_i$). Each list has at most $m$ tuples, where $m$ is determined by the communication budget. Combine the inputs from all child nodes together, treat each ($W_i$, $M_i$) as $W_i$ copies of $M_i$. Use the primitive *selective sampling* described in Algorithm 1 to output a new list of ($W_i'$, $M_i'$). Note that the *sort* operation in *selective sampling* is *not* computationally intensive since it is a *merge sort* from $k$ sorted list, as opposed to a sort procedure from scratch where $k$ is the number of children at an intermediate node.

- At the sink, combine inputs from all its children. Suppose that there are $m$ tuples in total, ($W_1$, $M_1$), ($W_2$, $M_2$), .... ($W_m$, $M_m$). When computing the median from these $m$ tuples, treat each ($W_i$, $M_i$) as $W_i$ copies of $M_i$.

We evaluate *median computation by selective sampling* using S-Pol radar data.

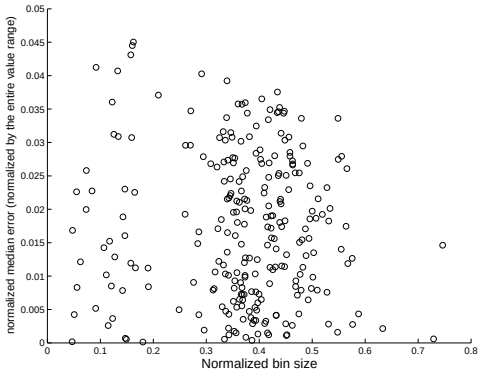



Figure 4.7: Results from evaluating Selective Sampling algorithm with the S-Pol radar data: the scatter plot of normalized estimation error *vs.* normalized median bin size indicates no correlation between them

The scatter plot of the normalized estimation error vs. the normalized median bin size (see Figure 4.7) indicates no correlation between them. This may suggest that the algorithm is not sensitive to data distribution.

We demonstrated that a simple improvement to the approach based on uniform sampling reduces the algorithm's sensitivity to data distribution. In general it may be difficult, or even impossible, to design such an improvement for many algorithms. When it is difficult to design an algorithm that can explicitly remove its sensitivity to data input we recommend evaluating algorithms with data across a range of parameter values and investigate how the algorithm's performance changes with different data characteristics. The parameter of interest could be data distribution, spatial correlation, or other data characteristics. The synthetic data generation approach discussed in Chapter 7 can be used to generate realistic data sets corresponding to a wide range of parameter values.

## 4.4 Further discussion on the data sensitivity problem

The significance of the data sensitivity problem not only depends on the specific algorithm in the study, but also depends on the performance metric under consideration. For example, in the case of median computation by uniform sampling, when the estimation error is defined in terms of value, it is sensitive to data distribution. However, when the estimation error is defined in terms of order, the following proof proved that the estimation error does not depend on the underlying data distribution.

Suppose the entire sample population $S$ includes $n$ elements and the median is computed based on $l$ samples randomly selected from $S$. The median of these $l$ samples, $x$, will be returned as an estimate of the real median; Let $x_p$ denote the position of $x$ in $S$. The error is then defined as:

$$x_p - n/2 \tag{4.4}$$

Next, we consider the *probability distribution of this error*: $P((X_p - n/2) = \epsilon n)$ where $X_p$ is a random variable denoting the position of the median estimated using $l$ samples. Without loss of generality, assume that $l$ is odd and the elements in $S$ are distinct. The event "$(X_p - n/2) = \epsilon n$" is equivalent to "*exactly* $l/2$ *of* $l$ *samples are located in positions less than* $n/2 + \epsilon n$":

$$P((X_p - n/2) = \epsilon n) = P(S_l = l/2) \tag{4.5}$$

where $S_l$ denotes the number of samples out of $l$ samples having positions less than $n/2 + \epsilon n$. Let $p_i$ denote the position for the $i^{th}$ sample. $Y_i (i = 1, ..., l)$ is a random variable, *s.t.*, $Y_i = 1$ if $p_i < (n/2 + \epsilon n)$; otherwise, $Y_i = 0$. Since we apply uniform sampling, we have:

$P(Y_i = 1) = 1/2 + \epsilon$ and $P(Y_i = 0) = 1/2 - \epsilon$

Independently drawing $l$ uniform samples is a $n$ *Bernoulli trials with success probability* $p = 1/2 + \epsilon$. Thus we have

$$P(S_l = l/2) = \binom{l}{l/2} * (1/2 + \epsilon)^{l/2} * (1/2 - \epsilon)^{l-l/2} \tag{4.6}$$

From Equation 4.5, 4.6, we have:

$$P((X_p - n/2) = \epsilon n) = \binom{l}{l/2} * (1/2 + \epsilon)^{l/2} * (1/2 - \epsilon)^{l-l/2} \tag{4.7}$$

Note that equation 4.7 only depends on $l$ and $\epsilon$, not on the original sample distribution.

Assuming that the estimation error is measured in terms of order difference (as defined in Equation 4.4), We prove that the error statistics of median computation by uniform sampling does not depend on the underlying data distribution but only on the proportion of samples taken.

## 4.5 Summary

In this chapter we discussed the statistical estimation applications. We study two types of percentile computation algorithms and identify a single parameter essential to the algorithm performance. In both cases results from our statistical performance analysis demonstrate that the algorithm performance changes significantly when evaluated using different data input. Using non-parametric statistical analysis tools, we identify *p-percentile* bin size to be an essential parameter to the percentile estimation algorithms. Ideally, we would like to identify a small number of data characteristics sufficient to determine algorithm performance.

This will significantly reduce the search space in synthetic data generation. In the case of percentile estimation by uniform sampling, we prove that estimation accuracy depends on a single parameter, namely, the normalized *p-percentile* bin size.

# CHAPTER 5

# Data Compression Applications

In the case studies of *p-percentile estimation* we identify a single parameter essential to the algorithm performance, namely, *p-percentile* bin size, which partially characterize the data distribution. In this chapter we examine another parameter essential to many data processing algorithms: *spatial correlation*. In particular, we investigate two data compression algorithms and demonstrate that *spatial correlation* are essential to their performance.

## 5.1 Data compression

Since it is prohibitively expensive to transfer raw sensor data in sensor networks, compression is often performed in the temporal domain or in the spatial domain to reduce the communication cost. In this paper we consider spatial compression. Given a snapshot of sensor readings, $\{Z(u_i)\}$, at locations $u_i$, $i = 1, ..., n$, $u_i \in A$, a compressed representation of $\{Z(u_i)\}$, instead of $\{Z(u_i)\}$ itself, were transfered to the sink. Depending on whether we can reconstruct the original sensor readings $\{Z(u_i)\}$ without error at the sink, compression algorithms can be classified into lossless and lossy compression. In this chapter we consider two compression algorithms that are intuitively sensitive to data, an instance of wavelet compression algorithm and joint entropy coding algorithm. They fall into lossy compression and lossless compression algorithms category respectively.

Using the same performance analysis technique in Chapter 4, we quantitatively demonstrate how the algorithm performance changes with data of various spatial correlations. Through statistical performance analysis, we demonstrate that spatial correlation is an essential parameter in The wavelet compression and joint entropy coding algorithms.

### 5.1.1 Wavelet Compression Algorithm

In this section we study an instance of wavelet compression algorithms. As illustrated in Figure 5.1, it is composed of three steps: (1) we apply wavelet decomposition to the 2D spatial signal. Details on how to implement this in a distributed fashion is out of the scope of this paper. For example, [Ser03] provides a distributed algorithm of wavelet decomposition in the spatial dimension. The results presented here are acquired from applying the Matlab built-in wavelet decomposition function to a 2D spatial data set. Without loss of generality we use the "sym2" filter. (2) After wavelet decomposition most of the energy concentrates on small percentage of coefficients; therefore, we use threshold cutting to discard insignificant coefficients. In the reconstruction phase these insignificant coefficients were set to 0. (3) We apply run-length coding to the wavelet coefficients that are above a certain threshold.

Among these three steps wavelet coefficients thresholding are lossy compression; the other two steps are lossless transformation. The diagrams on the bottom of Figure 5.1 illustrate the decompression steps. In the reconstruction phase the discarded wavelet coefficients are set to a default value of 0. Therefore, the reconstructed wavelet coefficients are likely to be different from the original ones. We define the **MSE** to be the **mean squared difference** between the reconstructed data and the original data.

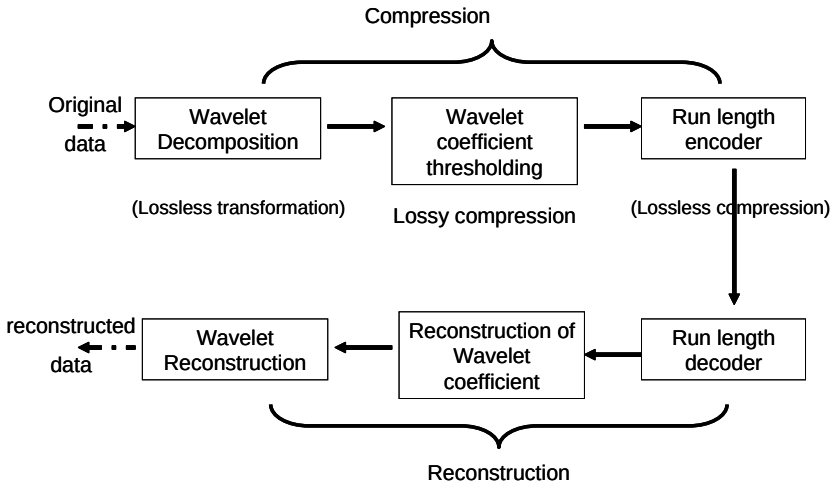



Figure 5.1: Illustration of wavelet compression and decompression procedure

In step (2), the reconstruction mean squared error is proportional to the threshold used to select the wavelet coefficients. Thereby, we can use this threshold to adjust the reconstruction fidelity (*i.e.*, mean squared error). Assuming that uniform quantization is applied to the output of the run-length encoding, the amount of output data of the run-length encoding is then proportional to the communication cost measured in bit-hops count. The communication cost in transferring the compressed data also depends on the distance from the sensor source to the sink. Thus, our communication cost metric, the amount of the output data of the run-length encoding, is measured in terms of spatial bit rate. In our study we fix the threshold used in selecting the wavelet coefficients and examine the amount of output of the run-length encoding.

Following the statistical analysis described in Chapter 4, we first define the performance metric of the wavelet compression algorithm to be the **communication cost**. This is measured by *the number of elements in the output from the run-length encoding.*

Second, we define the relevant data characteristic to be the *spatial correlation*. In geostatistics, *Variogram values* have been used to characterize the spatial correlation in the data. Briefly, the variogram value of a pair of points $x_i$ and $x_j$ is defined as $\gamma(x_i, x_j) = \frac{1}{2}\{Z(x_i) - Z(x_j)\}^2$ where a random process, $Z$, consists of a set of random variables $Z(u)$, one for each sensor location $u$. To characterize the spatial correlation between data points separated by various distances, assuming an isotropical data set a *variogram value* can be defined as a function of the *separation distance* between two points. To simplify our analysis, we use the variogram value at a unit separation distance to represent the spatial correlation in the data. The S-Pol radar data under study is collected from a grid topology; thus, the variogram value at unit separation distance is equivalent to the expected difference between neighboring nodes. This variogram value is identified to be the relevant data characteristic in our study.

As a final step we study how the algorithm performance changes with data input across a wide range of spatial correlations. We evaluate the wavelet compression algorithm using 259 snapshots of S-Pol radar data.

In the scatter plot of *communication cost* vs. *spatial correlation*(Figure 5.2), each point in the graph corresponds to one snapshot of S-Pol radar data. The x-axis plots its *variogram value at unit separation distance*, which is used to characterize the spatial correlation of the data. A small variogram value indicates a strong spatial correlation in the data, or vice versa. The resulting y-axis is the number of elements in the output from the run-length encoding. This value is directly proportional to the communication cost in transferring the compressed data to the sink.

As shown in Figure 5.2 the communication cost increases with the increasing variogram value, which corresponds to less spatial correlation. In wavelet

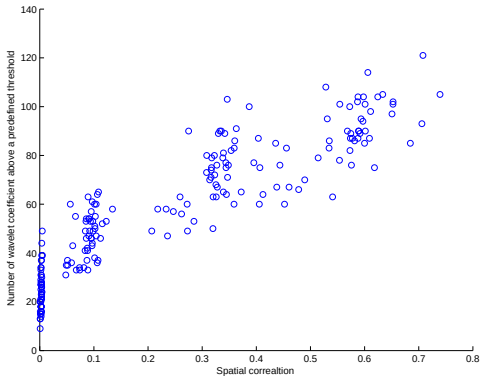



Figure 5.2: Scatter plot of the communication cost in wavelet compression versus spatial correlation measured in terms of variogram value at a unit distance. Communication cost increases with increasing variogram values. Correlation coefficient = 0.9069

compression the wavelet decomposition basically removes the spatial redundancy. Stronger spatial correlation will lead to higher spatial redundancy; thereby, wavelet compression will have a better chance to compress the data. As a result, the communication cost will decrease. We also compute Pearson's correlation coefficient of the *communication cost* and *the variogram values*. The correlation coefficient is 0.9069, which is close to 1. This indicates a strong correlation between them.

### 5.1.2 Joint entropy coding algorithm

Compared to coding each sample independently joint entropy coding improves the coding efficiency by exploiting the correlation between sensor readings at neighboring nodes, thus coding samples from a local neighborhood together. In our case study we compare two coding policies: (1) apply joint entropy coding to two neighboring nodes; (2) apply entropy coding independently at each node. To evaluate the gain of joint entropy coding over marginal entropy coding at each

node, we define the performance metric of a joint entropy coding algorithm to be *the joint entropy of sensor readings at two neighboring nodes*. This joint entropy value is normalized by the *the corresponding marginal entropy*. This normalized communication cost metric is inverse to the *efficiency of the joint entropy coding*.

The *joint entropy* of two variables is equivalent to the sum of the marginal entropy of those two variables deducted by their mutual information (*i.e.*, $H(X,Y) = H(X) + H(Y) - I(X;Y)$ [TC91]). Because mutual information can be captured by spatial correlation in the data, we define *spatial correlation* to be the relevant data characteristic in this study. We define the *normalized spatial correlation* (or called *spatial correlation coefficient*) to be the *variogram value at unit separation distance normalized by the variance of the data*. The normalization introduced here removes the effect of data being measured in different units; thus, making data sets of different magnitudes comparable.

Given the performance metric and relevant data characteristic identified above, we study how the algorithm performance changes with different data input. Again we use 259 snapshots of S-Pol radar data in our evaluation. In the scatter plot of *normalized communication cost* vs. *normalized spatial correlation* (Figure 5.2) the x-axis plots its *variogram value at unit separation distance normalized by the variance of the data*. These values represent the spatial correlation in the data. The y-axis plots the *joint entropy normalized by the marginal entropies*, This value is directly proportional to the spatial bit rate in transmitting the compressed data back to the sink.

As shown in Figure 5.3 the communication cost of joint entropy coding increases with the increasing normalized spatial correlation; this indicates less spatial correlation. Both the scatter plot and the Pearson's correlation coefficient (0.9355) indicates that the efficiency of the joint entropy coding algorithm is well

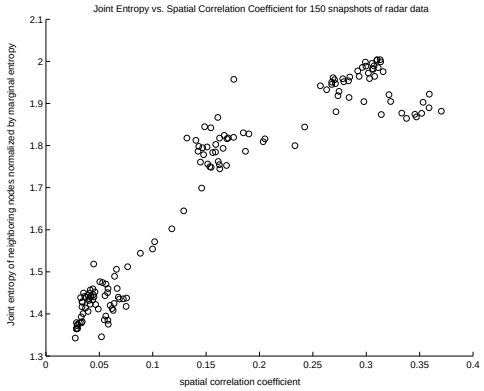



Figure 5.3: Scatter plot of the normalized communication cost versus normalized spatial correlation. Communication cost increases with increasing variogram values. Correlation coefficient = 0.9355

correlated with the spatial correlation in the data.

## 5.2 Sufficiency in joint entropy coding algorithm

The above statistical performance analysis demonstrates that in the joint entropy coding and wavelet compression algorithms, spatial correlation is an essential parameter to the algorithm performance. As mentioned in Section 4.2, our ideal goal is to demonstrate that a small number of parameters is sufficient to determine algorithm performance. In the case of the joint entropy coding algorithm, algorithm performance is measured by the joint entropy of two random variables normalized by their marginal entropy. This metric is used to assess the benefits of joint entropy coding over marginal entorpy coding. Ideally we would like to demonstrate that spatial correlation is sufficient to determine algorithm performance. If this ideal goal can be achieved, then *spatial correlation* is the single parameter needed to vary in the synthetic data generation process for the purpose

of evaluating the joint entropy coding algorithm.

For data following a joint Gaussian distribution, we prove that the joint entropy of two random variables at neighboring nodes is a function of their spatial correlation. Without prior assumptions regarding the data distribution, we demonstrate that the joint entropy and spatial correlation of these two random variables achieve their extreme values simultaneously.

### 5.2.1 Bivariate Gaussian Distribution

We start with stating our assumptions and notations. Let $X$ and $Y$ denote two random variables generated by a random process $Z$ at two neighboring nodes. Assume that this random process $Z$ is stationary in a local neighborhood. Consequently, the mean and variance of $X$ and $Y$ are constants, *i.e.*, $E(X) = E(Y) = \mu$; and variance(X) = variance(Y) = $\sigma$.

We first examine the scenario in which $X$ and $Y$ follows a bivariate Gaussian distribution. Further, under the assumption of local stationary, the bivariate Gaussian probability density function of $X$ and $Y$ is:

$$p(x,y) = \frac{1}{2\pi\sigma^2\sqrt{(1-\rho)}} * e^{[-\frac{1}{2(1-\rho^2)}[(\frac{x-\mu}{\sigma})^2 - \frac{2*\rho*(x-\mu)(y-\mu)}{\sigma^2} + (\frac{y-\mu}{\sigma})^2]]} \tag{5.1}$$

Under the local stationary assumption, $\mu$ and $\sigma$ can be considered constants. Therefore, the value of Equation 5.1 depends on a single parameter $\rho$, where $\rho$ is Pearson's correlation coefficient of $X$ and $Y$.

The joint entropy of $X$ and $Y$ is

$$H(X,Y) = -\sum_{x\in X}\sum_{y\in Y}\log p(x,y) \tag{5.2}$$

On the other hand, the spatial correlation coefficient of $X$ and $Y$ is defined

as: $\frac{E((Y-X)^2)}{\sigma^2}$, where

$$\begin{aligned} E((Y-X)^2) &= E(((Y-\mu)-(X-\mu))^2) \\ &= E((Y-\mu)^2) + E((X-\mu)^2) - 2 * E((Y-\mu)*(X-\mu)) \\ &= \sigma^2 + \sigma^2 - 2 * Cov(X,Y) \end{aligned} \tag{5.3}$$

From Equation 5.3, we have:

$$\frac{E((Y-X)^2)}{\sigma^2} = 2 - 2 * \rho_{X,Y} \tag{5.4}$$

where $\rho_{X,Y}$ is the Pearson's correlation coefficient of $X$ and $Y$. Slightly changing Equation 5.4, we have

$$\rho_{X,Y} = \frac{1}{2}(2 - \frac{E((Y-X)^2)}{\sigma^2}) \tag{5.5}$$

Under the assumption of bivariate Gaussian distribution from Equation 5.1, 5.2, and 5.5, the joint entropy of $X$ and $Y$, $H(X,Y)$, can be determined by a single parameter, their spatial correlation coefficient, $\frac{E((Y-X)^2)}{\sigma^2}$. Note that $\mu$ and $\sigma$ are constants under the assumption of local stationarity. Note that the above proof also holds when $X$, $Y$ is wide-sense stationary in the temporal dimension and $Z$ is wide-sense stationary in a local neighborhood.

### 5.2.2 Extreme conditions in the general case

In this section we investigate those scenarios in which the assumption of the joint Gaussian distribution does not hold. In the following example, we use the same notations as in Section 5.2.1. Two random variables $X$, $Y$ are located at neighboring nodes. The joint entropy of $X$, $Y$ is normalized by their marginal entropy.

$$\frac{H(X,Y)}{H(X)} = \frac{H(X) + H(Y|X)}{H(X)} = 1 + \frac{H(Y|X)}{H(X)} \tag{5.6}$$

We derive Equation 5.4 in Section 5.2.1, which we rewrite below for convenience. The spatial correlation coefficient of $X$ and $Y$ is defined as:

$$\frac{E((Y-X)^2)}{\sigma^2} = 2 - 2 * \rho_{X,Y} \tag{5.7}$$

where $\rho_{X,Y}$ is the Pearson's correlation coefficient of $X$ and $Y$.

Intuitively, a stronger correlation (*i.e.*, larger $\rho_{X,Y}$ value) between the two variables indicates less uncertainty if one of the two random variables $X$ and $Y$ is known. This results in a smaller conditional entropy value $H(Y|X)$. In general, $\rho_{X,Y}$ is a metric for a linear correlation of $X$ and $Y$. In contrast, $H(X,Y)$ incorporates a higher order correlation between $X$ and $Y$. As a result, $H(X,Y)$ generally depends on more than a single parameter, $\rho_{X,Y}$.

Next, we demonstrate that under two extreme conditions the joint entropy of $X$ and $Y$ and their spatial correlation coefficient simultaneously achieve their extreme values.

1. $X$ and $Y$ are perfectly correlated. In equation 5.6, $H(Y|X) = 0$. Thus the normalized joint entropy of $X$ and $Y$ ($\frac{H(X,Y)}{H(X)}$) achieves its minimum value, 1. At the same time, $\rho_{X,Y} = 1$. Therefore, Equation 5.7 achieves its minimum value, 0.

2. $X$ and $Y$ are independent from one another. Under this condition, $H(Y|X) = H(Y)$. Further we have $H(X) = H(Y)$ due to the assumption of local stationary. This, in turn, implies that $\frac{H(Y|X)}{H(X)} = 1$. Therefore, the normalized joint entropy of $X$ and $Y$ in equation 5.6 achieves its maximum value, 2. Under this same condition, $\rho_{X,Y} = 0$; and Equation 5.7 achieves its maximum value, 2.

# CHAPTER 6

# Field Estimation Applications

For each case study in previous chapters we identify a single parameter essential to algorithm performance. In general, algorithm performance is potentially affected by multiple data characteristics. In this chapter we investigate the significance of data sensitivity and identify parameters essential to algorithm performance in the context of those more complex data processing algorithms. In particular we examine a field estimation algorithm, namely, Fidelity Driven Sampling [BRY]. We use it as an example to illustrate how to apply our systematic performance analysis technique to a fairly sophisticated algorithm. Intuitively, the algorithm performance is not determined by any single data metric. In Section 6.1, we demonstrate that compared to data simulated from simple models, evaluating Fidelity Driven Sampling using experimental data changes its relative performance compared to Raster Scan. This may suggest that results from algorithm performance evaluation using data derived from simple models may be misleading and we need to evaluate algorithms using realistic data with various features. Motivated by the performance contrast results in Section 6.1, using synthetically generated scenarios, in Section 6.2 we identify *total mean curvature* as a quantitative metric important to the algorithm performance.

## 6.1 Evaluating Fidelity Driven Sampling with simulated and experimental data

Densely deployed sensor networks provide unprecedented capabilities for environmental monitoring. The objective of the *field estimation* is to reconstruct a map of the environmental field at the sink. Due to the energy constraints in sensor networks, field estimation algorithms resort to efficient sampling and data transportation techniques to reduce the communication cost while providing a high-fidelity reconstruction of the field at the same time.

**Fidelity Driven Sampling** (proposed in [RPE04, BRY]) exploits mobile sampling to first stratify the environment into regions requiring varying degrees of sample density, then samples in these regions. Fidelity Driven Sampling (FDS) maintains an estimate of the field being observed. Using this estimate the Fidelity Driven Sampling identifies regions or strata exhibiting a high degree of misfit. At each step in the sampling process Fidelity Driven Sampling adds points to that stratum with the largest error. In so doing Fidelity Driven Sampling attempts to reduce the mean squared error at each sampling point by adjusting point density and location. The algorithm continues adding points to poor fitting strata until either an overall sample budget is exhausted or a desired fidelity limit is achieved. A simple alternative to Fidelity Driven Sampling is to Raster Scan the field with a fixed resolution.

Following the Fidelity Driven Sampling operation (or raster scanning data acquisition), the returned variable field with its distribution of sample points is then supplied to a local polynomial interpolation algorithm and returns a reconstruction of the environmental field. We define the performance evaluation metric as the Mean Squared Error (MSE) between this reconstructed field map and the

ground truth. We evaluate the algorithm using data simulated from simple models and data collected from a lab environment. We plot the Mean Squared Error achieved in Fidelity Driven Sampling or Raster Scan against the total number of samples used in the field estimation. (Figures 6.1 and 6.2). The MSE represents the quality of the reconstructed field map. The number of samples is proportional to the cost or delay to achieve this reconstruction. Thus the performance curves in Figures 6.1 and 6.2 reflect the trade-offs between the quality and delay or cost. The lower the curve, the more desirable the performance is.

### 6.1.1 Evaluation results on the simulated data

The Fidelity Driven Sampling algorithm can be evaluated using data simulated from linear, quadratic, and cubic models [RPE04]. As shown in Figure 6.1 when evaluated with data simulated from simple models, both Fidelity Driven Sampling and Raster Scan deliver a very small MSE. Further, in most cases the MSE generated from Fidelity Driven Sampling is several orders of magnitudes smaller than that from Raster Scan. However, this conclusion does not hold when evaluated with experimental data collected from a lab environment.

### 6.1.2 Evaluation results on the experimental data

As discussed in [BRY], Fidelity Driven Sampling is evaluated by subjecting the algorithm to environmental variable fields having two extremes in their "curvature" characteristics. For one limit the environmental variable field was created by placing many obstacles in the illumination field (Figure 6.2(b)). This emulates the most complex patterns observed in the natural environment. In addition, we created a low curvature field by casting a diffuse shadow on the transect (Figure 6.2(a)). This latter case is characteristically similar to the least complex fields

observed under clear forest canopy structure. In both cases the *ground truth* was obtained by measurements from exhaustively moving the node at its highest resolution through the variable field. In contrast to the results from Section 6.1.1, when evaluated with the experimental data (Figure 6.2), the MSE obtained from Fidelity Driven Sampling is very close to or higher than the MSE obtained from Raster Scan. Specifically, under smooth phenomena, the MSE achieved by Fidelity Driven Sampling and Raster Scan are comparable. However, under data with rough curvature Fidelity Driven Sampling delivers a larger MSE than Raster Scan. This may suggest that smoothness of the data input may significantly affect algorithm performance.

### 6.1.3 Summary

In summary, when evaluated with data simulated from simple models and experimental data, the relative performance order between Fidelity Driven Sampling and Raster Scan changes. This indicates that the evaluation results from data input solely based on simple models may be misleading. Further, due to the complexity of the physical phenomena, it is impossible to represent all possible data input using only parametric models. Experimental data guides our efforts to these portions of the parameter space that represents real-world scenarios.

The dramatical performance change caused by different data inputs may not be unique to the Fidelity Driven Sampling algorithm. For example, the Backcasting algorithm proposed in [WMN04] shares a similar idea with the Fidelity Driven Sampling because both algorithms adjust their sampling densities based on the initial coarse model of the field map. Therefore, the Backcasting algorithm might be subject to the same problem, *i.e.*, sensitivity to the environmental field. In [WMN04] the algorithm was evaluated using a simulated piecewise smooth field

with a single edge. Evaluating algorithms using experimental data with various features (*e.g.*, Figure 6.2(a) and 6.2(b)) helps identify the regime of the parameter space where the algorithm may perform well compared to other alternatives.

## 6.2 Identifying parameters important to Fidelity Driven Sampling algorithm performance

In this section, we investigate what data features cause a change in relative performance order when evaluating Fidelity Driven Sampling and Raster Scan using data simulated from simple models and experimental data (Section 6.1). We conjecture that the smoothness and geometrical shape of the phenomena are two important data features. First we provide intuition on why the smoothness and geometrical shape of the phenomena are important to Fidelity Driven Sampling. Then we provide a quantitative metric, *total mean curvature of a surface*, to characterize these data features.

The estimation algorithm used by Fidelity Driven Sampling may contribute to performance degradation when the phenomena under evaluation has sharp edges (*e.g.*, the results shown in Figure 6.2(b) and 6.2(d)). Fidelity Driven Sampling uses the *locfit* [LOC] function in R [Sta99] for its estimation. The locfit function has a side effect of local smoothing. Therefore, Fidelity Driven Sampling may perform better when the phenomena is mostly smooth. On the other hand, the algorithm performance might degrade if the phenomenon has sharp edges. As for the geometric shape of the phenomenon, Fidelity Driven Sampling uses a quad tree and rectangular regions in its models of the physical phenomena. We conjecture that when the spatial structure of the physical phenomenon matches a rectangular shape, Fidelity Driven Sampling will perform well; otherwise its

performance may degrade.

In searching for a quantitative metric that incorporates both the smoothness and geometrical shape of the phenomena, we define *total mean curvature* for a regular surface $S$. Four metrics have been conventionally proposed to measure the curvature at a point $p$ on a surface $S$: *normal curvature*, *principal curvature*, *Gaussian curvature*, and *mean curvature*. Based on the definitions given in [Car76],

**Definition 1 (Normal curvature).** *Let $C$ be a regular curve in surface $S$ passing through $p \in S$, $k$ is the curvature of $C$ at $p$, and $cos\theta =< n, N >$, where $n$ is the normal vector to $C$ and $N$ is the normal vector to $S$ at $p$. $k_n = kcos\theta$ is then* called *the* normal curvature *of $C \subset S$ at $p$.*

**Definition 2 (Principal curvature).** *The maximum normal curvature $k_1$ and minimum normal curvature $k_2$ are called the principal curvatures at $p$; the corresponding directions, that is, the directions given by the eigenvectors $e_1$ and $e_2$,* are called *principal directions at $p$.*

**Definition 3 (Mean curvature).** *Let $p \in S$ and let $dN_p : T_p(S) \longrightarrow T_p(S)$ be the differential of the Gauss map. The negative of half of the trace of $dN_p$* is called *the mean curvature $H$ of $S$ at $p$. In terms of principal curvatures $k_1$ and $k_2$, $H$ can be written as $H = \frac{k_1+k_2}{2}$.*

**Definition 4 (Total mean curvature).** *For a* Monge patch *$S$ with $z = f(x, y)$, where $(x, y)$ is the sensor location and $z$ is the sensor value at location $(x, y)$, let $H(x, y)$ denotes its mean curvature at $(x, y)$. The total mean curvature of $S$ is defined as: $\int_{(x,y,z)\in S} H^2(x, y)$.*

Note that we use *mean curvature* in the above definition of *total mean curvature* of a surface. To compute the *mean curvature* of a point on a Monge patch

defined by a discrete data set $\{z(x, y)\}$, we used a discrete version of the formula provided in [Gra97]:

$$H(x, y) = \frac{(1 + h_y^2)h_{xx} - 2h_x h_y h_{xy} + (1 + h_x^2)h_{yy}}{2(1 + h_x^2 + h_y^2)^{3/2}} \tag{6.1}$$

Next we use synthetically generated data input to test whether there is a strong correlation between the algorithm performance and the *total mean curvature* metric. In the synthetically generated scenarios we have the flexibility to vary the phenomena along a single dimension while keeping other parameters fixed.

In our simulations, we use data generated from bivariate Gaussian pdf:

$$p(x, y) = \frac{1}{2\pi(\sigma_1\sigma_2)\sqrt{1 - \rho^2}} \exp(-\frac{(\frac{x-\mu_1}{\sigma_1})^2 - \frac{2\rho(x-\mu_1)(y-\mu_2)}{\sigma_1\sigma_2} + (\frac{y-\mu_2}{\sigma_2})^2}{2(1 - \rho^2)}) \tag{6.2}$$

In Equation 6.2, we fix the mean and correlation coefficient, and use variance to adjust the smoothness (consequently, the *curvature metric*) in the data. Specifically, we fix $\mu_1$ and $\mu_2$ to be 50, correlation coefficient $\rho$ to be 0.2; and we vary the *variance* parameter $\sigma_1$ and $\sigma_2$ in the range from 1 to 50. We compute the *total mean curvature* for each data set. As demonstrated in Figure 6.4(a), the *total mean curvature* value is smaller for a larger variance value, which corresponds to smoother data. Figure 6.3 shows a couple of example data sets generated from the above bivariate Gaussian pdf model, with variances 7 and 37, which represent data with high curvature and relatively smooth data respectively.

Figure 6.4(b) shows the average MSE achieved by Fidelity Driven Sampling vs. *total mean curvature* of the data, both of which are in log scale. Each point in Figure 6.4(b) corresponds to evaluating Fidelity Driven Sampling using a data set generated by a certain variance value. Given a single phenomenon, with more samples available, Fidelity Driven Sampling tends to produce more accurate predictions. For example, in Figure 6.2(c) and 6.2(d), the mean squared error is

decreasing with increasing number of samples. To acquire a single quantitative metric, the mean squared errors are averaged over different numbers of samples. We then plot this averaged MSE *vs.* total mean curvature of the phenomenon. Furthermore, to make performance comparisons between data sets with different value ranges meaningful, we normalize MSE by the mean signal magnitude of each data set. Figure 6.4(b) clearly indicates that Fidelity Driven Sampling performs better with smooth phenomena than with high curvatured data. Moreover, it demonstrates the wide range of performance for data input with different curvatures.

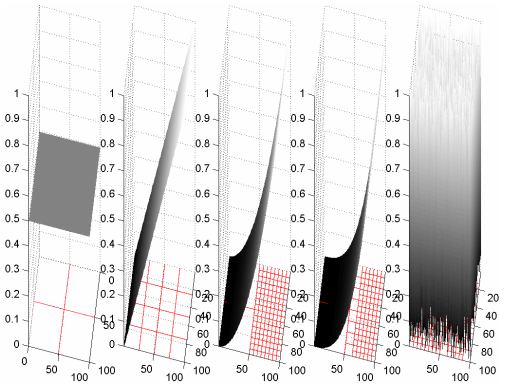
The above illustrations evaluated results from the simulated Gaussian model. Here we revisit the previous experimental evaluation results discussed in Section 6.1.2 and compute the *total mean curvature* metric for each data set. Visually the data in Figure 6.2(a) is smoother than the data in Figure 6.2(b). The computed total mean curvature metrics are consistent with this, they are 3638.2 and 19923 respectively. As demonstrated in Figure 6.2, when evaluating Fidelity Driven Sampling using data in Figure 6.2(a), MSE from Fidelity Driven Sampling is comparable or slightly smaller than that from Raster Scan (Figure 6.2(c)); whereas for data in Figure 6.2(b), MSE from Fidelity Driven Sampling is slightly higher than Raster Scan (Figure 6.2(d)). The evaluation results using the experimental data again confirms our conjecture that Fidelity Driven Sampling performs better with smooth phenomena than with high curvatured data.

*Total mean curvature* is intended to incorporate both the smoothness and geometrical structure of the data. However, partial information is lost when integrating the curvature information at each point to a single scalar metric. Furthermore, due to the algorithm complexity, it is unlikely that a single quantitative metric will determine the algorithm performance. Next we provide an

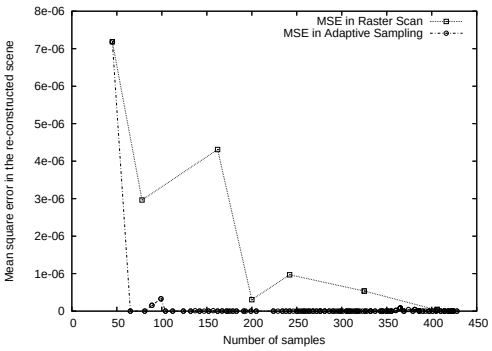
example in which *total mean curvature* alone does not provide an accurate indication on the algorithm performance. In this example, geometrical shape of the phenomenon plays an important role in the algorithm performance.

For data with rectangular edges (Figure 6.5(a)) and data with circular edges (Figure 6.5(b)), their *total mean curvature* are $3.57 * e + 6$ and $1.27 * e + 6$ respectively. If solely based on this curvature metric, the MSE generated from evaluating Fidelity Driven Sampling using data in Figure 6.5(a) should be higher than that generated from evaluating Fidelity Driven Sampling using data in Figure 6.5(b). However, we observe the opposite results in Figure 6.5(c) and 6.5(d). When evaluated using data with rectangular edges (Figure 6.5(a)) the MSE achieved by Fidelity Driven Sampling (Figure 6.5(c)) is comparable to that achieved by Raster Scan. Whereas, when evaluated using phenomena with circular edges (Figure 6.5(b)) the MSE achieved by Fidelity Driven Sampling (Figure 6.5(d)) is higher than that achieved by Raster Scan.

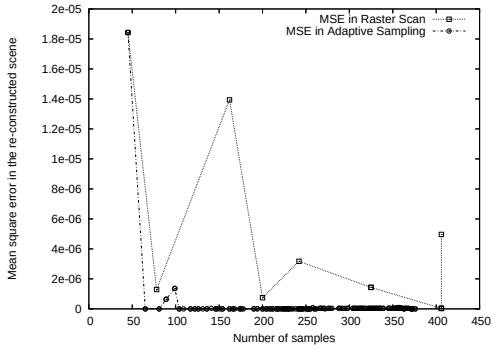
In summary, we have demonstrated that *total mean curvature* is an important parameter to the Fidelity Driven Sampling algorithm. In addition, the spatial structure of a physical phenomenon could also affect the algorithm performance significantly. Furthermore, the contrast between algorithm evaluation using simulated data and experimental data also demonstrates the necessity to evaluate algorithms using data with various features. However, both the curvature and the spatial structure of a real physical phenomenon could potentially take on many different values; therefore, it is impractical to represent all possible data input using only parametric models. The synthetic data generation discussed in the next chapter will guide simulation efforts to the portions of the space that represents real world scenarios.



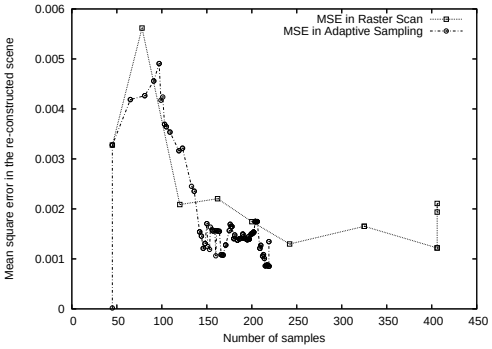
(a) Data simulated from simple models, including linear, quadratic and cubic models




(b) Result on linear data

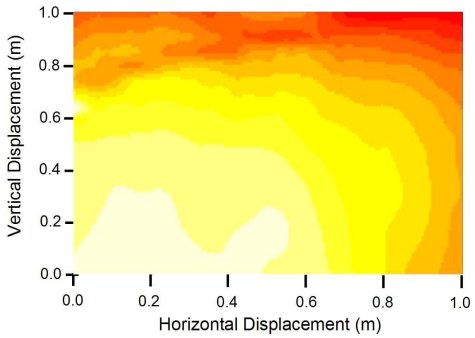



(c) Result on quadratic data

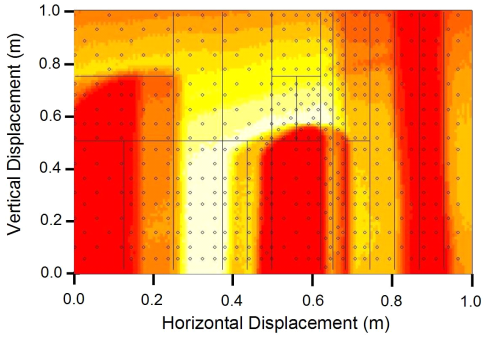



(d) Result on cubic data

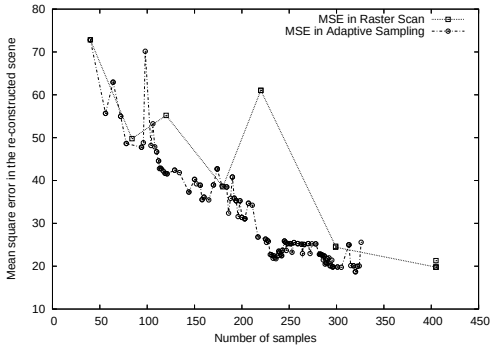
Figure 6.1: Comparison of Fidelity Driven Sampling vs. Raster Scan evaluated with data generated from linear, quadratic, and cubic models. Both Fidelity Driven Sampling and Raster Scan deliver very small MSE; however, MSE generated from Fidelity Driven Sampling is several orders of magnitudes smaller than that from Raster Scan for the same number of samples.



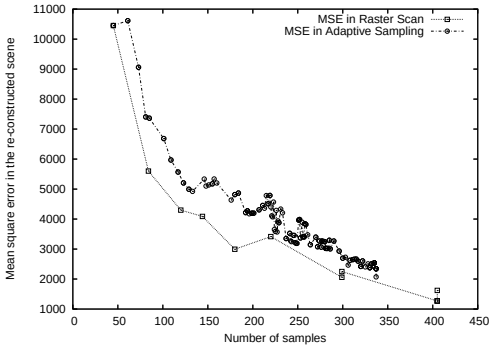

(a) Experimental data with smooth curvature




(b) Experimental data with rough curvature

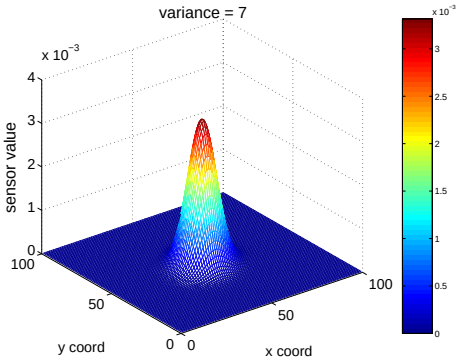



(c) Result on data with smooth curvature: MSE generated from FDS is comparable to that from RS.

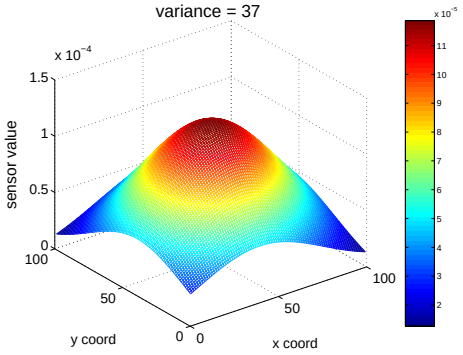



(d) Result on data with rough curvature: MSE generated from FDS is higher than that from RS.

Figure 6.2: Comparison of Fidelity Driven Sampling vs. Raster Scan when evaluated using data collected from a lab environment. The MSE achieved by Fidelity Driven Sampling is comparable to or higher than Raster Scan.

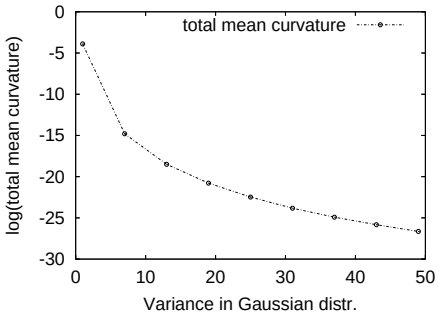



(a) variance in bivariate Gaussian pdf = 7

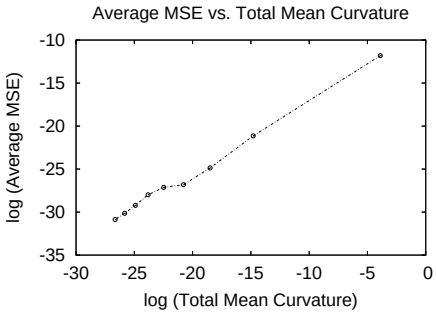



(b) variance in bivariate Gaussian pdf = 37

Figure 6.3: Different variance values in bivariate Gaussian pdf generate data of various smoothness

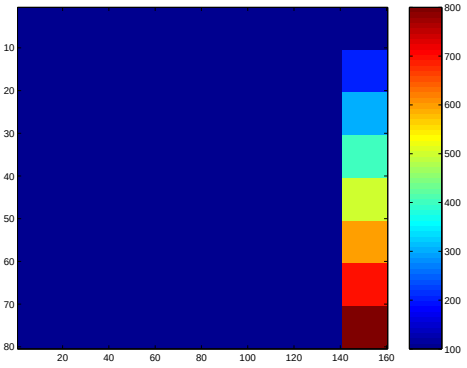



(a) Total mean curvature vs. Variance parameter in bivariate Gaussian pdf

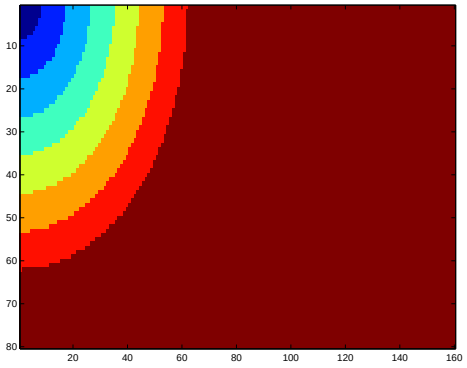



(b) Average MSE vs. Total mean curvature

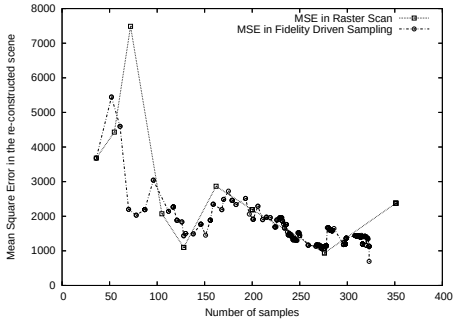
Figure 6.4: FDS demonstrates a wide range of algorithm performance when evaluated using data with different curvatures



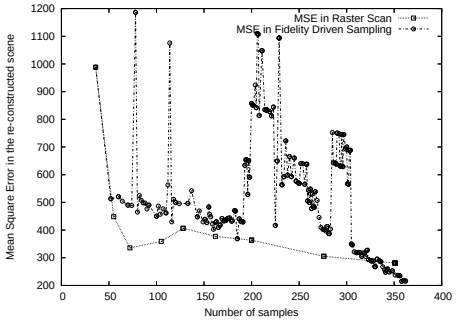

(a) Phenomena in rectangular shape, total mean curvature = 3.57e+06




(b) Phenomena in circular shape, total mean curvature = 1.27e+06




(c) Evaluation result over phenomena in rectangular shape: FDS is comparable to RS




(d) Evaluation result over phenomena in circular shape: MSE achieved by FDS is much higher than RS

Figure 6.5: In contrast to what is indicated by the total mean curvature metric, MSE achieved by FDS is comparable to that by RS when the edge is in rectangular shape; whereas, the MSE achieved by FDS is larger than that by RS when the edge is in circular shape.

# CHAPTER 7

# Scalable synthetic data generation

Through statistical performance analysis, the previous chapters have demonstrated that different data input could potentially change the algorithm performance dramatically. In some cases these inputs even change the relative performance order among multiple alternative algorithms. Therefore, we recommend evaluating algorithms against experimental data that represent various real-world scenarios, or data input corresponding to a range of parameter values.

In this chapter, we discuss techniques to generate data input that is required for this systematic algorithm evaluation. In particular, we present our scalable synthetic data generation framework, which will generate irregular topology data based on models of the experimental data and data that spans across a wide range along a dimension of interest to the algorithm.

The organization of this chapter is as follows. In section 7.1, we introduce our scalable synthetic data generation framework, briefly stating the design principles, motivation and objectives of two types of synthetic data generation techniques. In section 7.2, we describe those techniques used to generate data based on empirical models derived from the experimental data. In section 7.3, we describe algorithms to generate data sets corresponding to a wide range of parameter values.

## 7.1 Scalable synthetic data generation framework: design principles and objectives

Due to the complexity of the physical environments surrounding us, accurately describing a physical phenomenon often requires a large number of parameters. This huge parameter space of data input makes exhaustive exploration of parametric models impractical. In order to address the dimension explosion problem, we propose the following. First, by driving simulations from previously collected experimental data, we focus our testing on the part of the space that matters in real-world applications. Second, we identified a small number of data characteristics that are essential to the algorithm performance (as discussed in chapters 3 $\sim$ 6). Combining these strategies, the experimental data that are rich along these important dimensions represents the ideal input for the algorithm evaluation. Unfortunately, collecting new experimental data sets is expensive, time-consuming, and often presents technical challenges in itself. Thus, it is advantageous to develop methods to generate synthetic data based on models derived from previously collected data.

This application of previously collected data to new problems presents several challenges. First, existing experimental data is often collected from regular grids, whereas real deployments may have an irregular topology. Second, the available experimental data from relevant fields may be scarce, and therefore, not sufficient to evaluate algorithms against data across a wide range of parameter values. Third, the important data features are application and sensing modality dependent.

Leveraging the existing experimental data from relevant applications, we propose to generate irregular topology data from models of the experimental data.

Our proposed synthetic data generation techniques attempt to approximate the experimental data in terms of distribution, spatial correlation, or other features of interest. This allows algorithm evaluation with realistic data of arbitrary topology. Our design principles are listed as follows.

First, our approach is experimentally oriented. Due to reasons stated above, we recommend generating synthetic data from models driven by the experimental data, rather than from purely parametric models.

Second, data features to be modeled should be driven by the relevant user applications. The underlying physical phenomena the data models represent are inherently different, data from different application domains, or data from different sensing modalities may dramatically differ from each other in terms of distribution, spatial correlation, frequency spectrum, *etc.* For example, data from environmental weather applications could be very different from seismic waveforms or chemical contamination plumes in many relevant dimensions.

Last, the specific synthetic data generation technique used should adapt to the application and algorithm. Here we propose a framework to generate synthetic data, as opposed to recommending a single synthetic data generation algorithm or a single synthetic data set. Currently, our synthetic data generation toolbox includes nine different algorithms, based on spatial interpolation techniques and joint spatiotemporal modeling techniques. This will generate multiple data sets from a single experimental data set. The one that is most desirable depends on the specific application and algorithm in the study. For example, in evaluating a wavelet compression algorithm [YGG03], *spatial correlation* was identified as an essential data characteristic. We then select the synthetic data set that can best match the experimental data in terms of its *spatial correlation*. Identifying a small set of parameters essential to the algorithm performance provides quantita-

tive metrics that allow us to directly evaluate the synthetic data sets. Otherwise, we would have to rely on subjective eyeballing or indirect evaluation from comparing the algorithm performance when evaluated against the experimental data and against the synthetic data. The caveat with the latter approach is that synthetic data is often used when no experimental data is directly available, thereby, making it difficult to obtain a single basis for comparison.

The synthetic data generated based on the above principles will cover a similar range as the experimental data, since it tries to approximate the original data as closely as possible. If the existing experimental data from the relevant fields is scarce, the synthetic data generated from empirical models will have the same limited range as the experimental data. Thus, to address those scenarios where we do not have sufficient experimental data, we design algorithms to generate data sets corresponding to a wide range of parameter values. For quantitative data metrics (*e.g.*, data distribution, spatial correlation), we provide knobs to directly adjust these parameters. In contrast, for features that are difficult to characterize quantitatively, *e.g.*, the spatial structure of the phenomena or the spatial distribution of data values, we provide test data suites that can cover a good percentage of parameter space.

In the next two sections, we discuss two types of synthetic data generation techniques: (1) techniques to generate irregular topology data based on empirical models, whose output approximate experimental data as close as possible; and (2) algorithms to generate data sets rich in data characteristics essential to the algorithm performance.

The techniques in the first category are preferred when a rich set of experimental data is available since it will guide the simulation to the part of the parameter space representing real-world scenarios. Techniques in the second cat-

egory serve as a complement of those in the first category when the scope of the parameter space that the experimental data covers is not sufficient to evaluate the algorithm performance over a wide range of data input.

## 7.2 Synthetic data generation based on empirical models of experimental data

In this section, we discuss how to generate irregular topology data based on empirical models of experimental data. Briefly, the proposed irregular data generation procedure consists of two steps: (1) generate ultra fine-grained synthetic data from modeling the experimental data; (2) derive synthetic data of the specified topology by applying the nearest neighbor re-sampling method to the fine-grained data obtained from the first step.

We start with the problem of generating fine-grained synthetic data. Our proposed synthetic data generation includes both spatial and spatio-temporal data types. To generate spatial data, we start with an experimental data set which is a collection of data measurements from a study area. Assuming that the data is a realization of an ergodic and local stationary random process, we use spatial interpolation techniques to generate synthetic data at unmonitored locations.

Similarly, to generate synthetic spatio-temporal data, again we start with an experimental space-time data set, which includes multiple snapshots of measurements from a study area at various times. If we were only interested in data at a particular recording time, we could apply our proposed spatial interpolation techniques to each snapshot of data separately, then generate a collection of spatial data sets at each recording time. However, this does not allow us to generate synthetic data at times other than the recording times. In addition, the

joint space-time correlation is not fully modeled and exploited if each snapshot of spatial data is modeled separately. Therefore, we propose to model the joint space-time dependency and variation in the data. Inspired by a joint space-time model in [KMK02], we model our data as a joint realization of a collection of space indexed time series, one for each spatial location. Time series model coefficients are space-dependent, and so we further spatially model them to capture these space-time interactions. Synthetic data are then generated at both unmonitored time and location derived from the joint space-time model. This allows us to generate synthetic data at arbitrary spatial and temporal configurations.

A brief outline for the remainder of section 7.2 is as follows: Section 7.2.1 discusses spatial interpolation techniques and presents the results of applying them to a S-Pol radar data set. Section 7.2.2 then discusses a joint spatio-temporal model and the result of applying it to the same radar data set. In Section 7.2.3, we briefly describe how to generate data of an arbitrary topology by applying the nearest neighbor re-sampling method to the fine-grained data obtained from applying the spatial interpolation or joint spatio-temporal modeling techniques.

### 7.2.1 Spatial Data Generation

#### 7.2.1.1 Problem definition

We start with an experimental data set, which is typically sparsely sampled. To generate a large set of samples at much finer granularity, a spatial interpolation algorithm is used for prediction at unsampled locations. The spatial interpolation problem has been extensively studied. Both stochastic and non-stochastic spatial interpolation techniques exist, depending on whether we assume the observations are generated from a stochastic random process. In general, the spatial interpolation problem can be formulated as: Given a set of observations

$\{z(k_1), z(k_2), ..., z(k_n)\}$ at known locations $k_i$, $i = 1, ..., n$, spatial interpolation is used to generate prediction at an unknown location $u$. On the other hand, if we take a stochastic approach, the above spatial interpolation problem can be formulated as the following estimation problem. A random process, $Z$, is defined as a set of dependent (here spatially dependent) random variables $Z(u)$, one for each location $u$ in the study *area* $A$, denoted as $\{Z(u), \forall u \in A\}$. Assuming $Z$ is an ergodic process, the problem is defined to estimate some statistics (*e.g.*, mean) of $Z(u)$ ($u \in A$) given a realization of $\{Z(u_i)\}$ at locations $u_i$, $i = 1, ..., n$, $u_i \in A$. $A$ lies in a one dimensional or high dimensional space. We can use the mean of $Z(u)$ as an estimate of sensor readings at un-sampled locations.

#### 7.2.1.2 Spatial Interpolation Techniques

Among many existing stochastic interpolators, we studied Kriging in particular, partially because it has been widely used in the environmental science and mining industry, which are important sensor network applications. Kriging [Goo97] is a widely used geostatistics technique to address the above estimation problem. Kriging, which is named after D. G. Krige [Kri66], refers to a range of least-squares based estimation techniques. It has both linear and non-linear forms. Ordinary Kriging, which is a linear estimator, has provided good results in our spatial interpolation and joint spatio-temporal modeling example.

Assuming that the underlying random process is locally stationary, Kriging uses a variogram to model the spatial correlation in the data. Before going into the estimation theory in Kriging, we first briefly introduce the notion of *variogram* [IS89].

A variogram is used to characterize the spatial correlation in the data. The

variogram (also called semivariance) of a pair of points $x_i$ and $x_j$ is defined as

$$\gamma(x_i, x_j) = \frac{1}{2}\{Z(x_i) - Z(x_j)\}^2 \tag{7.1}$$

Variogram can also be defined as a function of lag, $h$ (*i.e.*, the separation between two points can be either separation distance or a vector with components of distance and direction, both of which can occur in two and three dimensions):

$$\gamma(h) = \frac{1}{2}E[\{Z(x) - Z(x+h)\}^2] \tag{7.2}$$

For a set of samples, $z(x_i)$, i=1, 2, ..., $\gamma(h)$ can be estimated by

$$\hat{\gamma}(h) = \frac{1}{2m(h)} \sum_{i=1}^{m(h)} \{z(x_i) - z(x_i + h)\}^2 \tag{7.3}$$

where $m(h)$ is the number of samples separated by the lag distance $h$.

Data in high dimensions might add complexity in modeling variograms. If data lie in a high dimensional space, variograms are first computed in different directions separately. If variograms in different directions turn out to be more or less the same, the data under study are isotropic, then sample variograms can be averaged together. Otherwise, data in different directions need to be modeled separately.

In Ordinary Kriging, at unmonitored locations, the data is estimated as a weighted average of the neighboring samples,

$$\hat{Z}(x_0) = \sum_{i=1}^{N} \lambda_i z(x_i), \tag{7.4}$$

where $\sum_{i=1}^{N} \lambda_i = 1$.

There are different ways to determine the weights. One method is to assign all of the weight to the nearest data, as used in the nearest neighbor interpolation approach; Another is to assign the weights inversely proportional to the distance

from the location being estimated among others available. In Kriging, the weights are determined by minimizing the estimation variance, which is written as a function of the variogram (or covariance),

$$var[\hat{Z}(x_0) = E[\hat{Z}(x_0) - Z(x_0)^2] = 2\sum_{i=1}^{N} \lambda_i \gamma(x_i, x_0) - \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i \lambda_j \gamma(x_i, x_j) \quad (7.5)$$

where $\gamma(x_i, x_j)$ is the variogram value of $Z$ between the sample points $x_i$ and $x_j$, and $\gamma(x_i, x_0)$ is the variogram value of $Z$ between the sample point $x_i$ and the target data point $x_0$.

Minimizing the estimation variance (*i.e.*, Equation 7.5) under the constraint that $\sum_{i=1}^{N} \lambda_i = 1$ (which is a necessary condition for an unbiased estimator) is a constrained optimization problem. It can be converted to an unconstrained optimization problem using the method of *Lagrange multiplier*. Specifically, we add a *Lagrange parameter* to the Equation 7.5:

$$\begin{aligned} var[\hat{Z}(x_0) &= E[\hat{Z(x_0)} - Z(x_0)^2] \\ &= 2\sum_{i=1}^{N} \lambda_i \gamma(x_i, x_0) - \sum_{i=1}^{N}\sum_{j=1}^{N} \lambda_i \lambda_j \gamma(x_i, x_j) - 2\mu(\sum_{i=1}^{N} \lambda_i - 1) \end{aligned} \quad (7.6)$$

The unconstrained minimization problem expressed in Equation 7.6 can be solved by setting its partial derivative with respect to each $\lambda_i$ and $\mu$ to be 0. It is a system of $N+1$ of linear equations involving $N+1$ unknowns, and can be solved by methods for solving systems of linear equations, *e.g.*, Gaussian Elimination, or through matrix inversions. We rewrite the above linear equations in the matrix

format as: $A\lambda = b$, where

$$\mathbf{A} = \begin{bmatrix} \gamma(x_1, x_1) & \gamma(x_1, x_2) & \cdots & \gamma(x_1, x_N) & 1 \\ \gamma(x_2, x_1) & \gamma(x_2, x_2) & \cdots & \gamma(x_2, x_N) & 1 \\ \vdots & \vdots & \cdots & \vdots & \vdots \\ \gamma(x_N, x_1) & \gamma(x_N, x_2) & \cdots & \gamma(x_N, x_N) & 1 \\ 1 & 1 & \cdots & 1 & 0 \end{bmatrix} \tag{7.7}$$

$$\lambda = \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \\ \mu \end{bmatrix} \tag{7.8}$$

$$\mathbf{b} = \begin{bmatrix} \gamma(x_1, x_0) \\ \gamma(x_2, x_0) \\ \vdots \\ \gamma(x_N, x_0) \\ 1 \end{bmatrix} \tag{7.9}$$

The weights and the Lagrange parameter can be obtained from

$$\lambda = A^{-1}b. \tag{7.10}$$

In addition to providing the least squares based estimate, Kriging also provides an estimation variance, which is a function of the variogram values and weights. This is one of the important reasons that Kriging has been popular in geostatistics.

However, as we will explain shortly, minimizing the estimation variance is not our ultimate goal. Our goal is to generate synthetic data which can approximate the original data set and be used to effectively evaluate sensor network

protocols. Therefore we also study other non-stochastic spatial interpolation algorithms, including: Nearest neighbor interpolation, Delaunay triangulation interpolation, Inverse-distance-squared weighted average interpolation, BiLinear interpolation, BiCubic interpolation, Spline interpolation, and Edge directed interpolation [al01].

We do not think one single interpolation algorithm fits all application scenarios due to the following reasons: First, interpolation algorithms usually perform differently when evaluated against different data input, just as what have been demonstrated earlier that data processing algorithms are sensitive to data input. From our experience, there is no consistent ranking between different interpolation algorithms across multiple data input. Second, when considering different evaluation metrics, interpolation algorithms rank differently among themselves. For example, considering original signal and its spatial correlation, we have two corresponding metrics: Mean squared difference between the spatial correlation of the synthetic data and that of the experimental data; and Mean squared error between the synthetic data and the experimental data. In our experiments, the ranks of eight interpolation algorithms change significantly when we compare the synthetic data to the original data using these two different metrics.

Therefore, as opposed to suggesting one or optimizing one single interpolation algorithm for a single metric or specific type of data, we experimented with and provide a suite of spatial interpolation algorithm implementations. With a new spatial interpolation or synthetic data generation algorithm, *e.g.*, the data generation technique proposed in [JP04] can be easily integrated into our scalable synthetic data generation framework.

In our study, we explored eight previously-studied interpolation algorithms. Next we briefly describe their interpolation scheme. Most interpolation algo-

rithms estimate the value at an un-monitored location as a weighted average of the neighboring observations. They differ in the following aspects: how many neighbors are involved to interpolate a point and how to determine the weight of each neighbor. *Nearest neighbor* interpolation assigns all the weight to the nearest neighbor. In *Delaunay triangulation*, the entire network is triangulated based on the locations of all observations and, on any triangle, the predictor is a planar interpolant of the surrounding three data values. In *Inverse-distance-squared weighted average*, neighboring data values are weighted according to how far they are away from the location being estimated. In *Bilinear interpolation*, at an unmonitored location, variables are estimated as a linear average of its 4 closest neighbors. Procedurely, first interpolates along each row, then along each column. In contrast, *Bicubic interpolation* is a weighted average of its 16 nearest neighbors, weights are a cubic function of the relative location from the unknown point to each neighbor.

Edge-directed interpolation [al01] was originally designed to increase an image's resolution. Its basic case is to increase the resolution of an image in each spatial dimension by a factor of 2, and it can be applied recursively. It uses 4th-order linear interpolation, *i.e.*, the weighted sum of its 4 diagonal neighbors. Under the assumption of local stationary Gaussian process, according to the classical Weiner filtering theory, the MSE linear interpolation coefficients are given by $\alpha = inv(R) * r$. where $\alpha$ is a vector of weights, $R$ (4x4 matrix) and $r$ (1x4 vector) are local covariance of high resolution image. Again, under the assumption of local stationary Gaussian process, high resolution covariance is a function of its corresponding low resolution covariance, which in turn can be computed from the observation.

Spline is a piecewise polynomial function that can have a locally very simple

form, yet at the same time be globally smooth. Cubic Spline, which is used in our study, is a piecewise cubic function, and the first and second derivative of the piecewise cubic functions exist at internal sample points (*i.e.*, the piecewise function is continuous and smooth).

Without knowledge on the phenomena structure, it is not obvious that which interpolation algorithm will more accurately capture the original data characteristics. We do not favor any single synthetic data generation algorithm, instead, we recommend generating multiple data sets by applying different interpolation algorithms, and select the one that best matches the experimental data in terms of the data characteristics that we are interested in.

#### 7.2.1.3 Evaluation of synthetic data generation

**Data set description** To apply the spatial interpolation techniques described above, we consider the resampled S-Pol radar data provided by NCAR[1], which records the intensity of reflectivity in dBZ, where Z is proportional to the returned power for a particular radar and a particular range. The original data were recorded in the polar coordinate system. Samples were taken at every 0.7 degrees in azimuth and 1008 sample locations (approximately 150 meters between neighboring samples) in range, resulting in a total of 500 x 1008 samples for each 360 degree azimuthal sweep. They were converted to the Cartesian grid using the nearest neighbor resampling method. A grid point is only assigned a value from a neighbor when the neighbor is within 1km and 10 degree range. If none of its neighbors are within this range, the grid point is labeled as missing value, *e.g.*,

[1]S-Pol (S band polar metric radar) data were collected during the International H2O Project (IHOP; Principal Investigators: D. Parsons, T. Weckwerth, et al.). S-Pol is fielded by the Atmospheric Technology Division of the National Center for Atmospheric Research. We acknowledge NCAR and its sponsor, the National Science Foundation, for provision of the S-Pol data set.

the NaN value is assigned. Resampling, instead of averaging, was used to retain the critical unambiguous and definitive differences in the data. In this paper, we select a subset of the data that has no missing values to perform our data analysis. Specifically, each snapshot of data in our study is a 60 x 60 spatial grid data with 1 km spacing.

**Spatial interpolation algorithms implementation** We apply the aforementioned eight interpolation algorithms to the selected spatial radar data sets. We use the *spatial* package in R [R u] to achieve Kriging. Nearest neighbor, Bilinear, Bicubic, Spline interpolation results were obtained from the interp2() function in Matlab. Since Bilinear and Bicubic interpolation functions in Matlab provide no prediction for edge points, we use results from Nearest Neighbor interpolation for edge points in bilinear or bicubic interpolation results. Edge directed interpolation is from [al01]. Inverse-distance-squared weighted average interpolation, and Delaunay triangulation interpolation were implemented in Matlab following the interface of interp2(). The *s*patial package in R and the interp2() function in Matlab requires input data to be collected from a grid region. This motivates us to use the resampled grid data, instead of the raw data from the polar coordinate system.

**Evaluation metrics** The synthetic data is desired to closely approximate the experimental data. This can be examined in an indirect or direct manner. Comparing the algorithm performance when evaluated against the experimental data and against the synthetic data is an indirect evaluation, which will be discussed in Section 8. A visual comparison after plotting both synthetic data and experimental data is a direct evaluation. In this section, we focus on direct evaluation through quantitative metrics.

For our synthetic data generation, it is desirable that the synthetic data can closely capture the interesting statistical features of the original data. The set of statistical features selected as evaluation metrics should be the important parameters that directly affect the algorithm performance. It is hard to define a statistical feature set that is generally applicable to most algorithms and data sets. Indeed, we argue that the evaluation metrics should be selected based on the application and algorithm under study. In chapter 3 - 6, we identified three classes of data processing applications in sensor networks that are potentially sensitive to data inputs. For each class of applications, we identify a small number of data characteristics that directly contribute to the algorithm performance.

Identifying a small set of parameters essential to the algorithm performance provides quantitative metrics that allow us to directly evaluate the synthetic data sets. Otherwise, we would have to rely on subjective criteria or indirect evaluations.

For example, a large percentage of existing data compression algorithms (including joint entropy coding and wavelet compression, which is used in the DIMENSIONS system [GEH02]) are sensitive to the spatial correlations in the data. In general, sensor networks are envisioned to be deployed in the physical environment and deal with data from the geometric world, thus, we believe that many sensor network protocols will exploit spatial correlation in the data. Therefore, besides visual comparison, here we use spatial correlation (which is measured by its variogram values) of the synthetic data versus original data to assess the applicability of this synthetic data generation technique to the sensor network algorithm being evaluated. In general, the evaluation metrics for the synthetic data generation are application and algorithms specific. When given a new class of data processing algorithms, identifying relevant data characteristics (or in other

words, the corresponding evaluation metric for synthetic data generation techniques) requires knowledge of the algorithm being evaluated.

To evaluate the synthetic data, we examine whether the synthetic data can approximate the interesting features of the experimental data. If a quantitative metric is defined for the identified data characteristics, the difference between this quantitative metric of the synthetic data and that of the original data is used as the evaluation metric for our synthetic data generation.

In this section, we use variogram values to measure the spatial correlation in the data, and define the evaluation metric for the synthetic data as follows: Two data sets $A$ and $B$ (*e.g.*, a synthetic data set and an experimental data set), and their variogram values are denoted as $\{\hat{\gamma}_1(h_i)\}$ and $\{\hat{\gamma}_2(h_i)\}$ respectively, where $h_i$ are sample separation distances between two observations; i =1, ..., m. The Mean Square Difference of variogram values of two data sets is defined as: $\sum_{i=1}^{m}(\hat{\gamma}_1(h_i) - \hat{\gamma}_2(h_i))^2$.

**Interpolation resolution** We studied two extremes of interpolation resolutions: (1) Coarse grained interpolation, in which case, we start from the down-sampled data (which reduces the data size in half in each dimension), increase the interpolation resolution by 4, compare the variogram value of the interpolated data with that of the original data. Note that the original data can be considered as ground truth in this case. The coarse grained interpolation is used to evaluate how the synthetic data generated by different interpolation algorithms approximate the spatial correlation of the experimental data. (2) Fine grained interpolation. Starting with a radar data set with 1km spacing, we increase the resolution by 10 times in each dimension, resulting in a 590x590 grid with 100m spacing. Fine grained interpolation is an essential step in generating irregular

topology data.

**Evaluation Results** First we visually present how the spatial correlation (*i.e.*, variogram values) of the synthetic data approximates that of the original data in the case of coarse-grained interpolation. For the spatial dataset shown in Figure 7.1, Figure 7.2 shows the variogram plot of several synthetic data sets (generated from various interpolation algorithms) *vs.* that of the original data. It demonstrates that the variogram curves of most synthetic data (except the one from Inverse-distance-squared weighted average interpolation) closely approximate that of the original one. At the long lag distances, the synthetic data may appear slightly under-estimating the long-range dependency in the original data. The source of this under-estimate may be due to the smoothing effect of the interpolation algorithms.

Further, we use the Mean Square Difference between the variogram values of the original data and the synthetic data as a quantitative measure of how closely the synthetic data approximates the original data in terms of variogram values. Table 7.1 lists the Mean Square Difference results averaged over 100 snapshots of radar data in increasing order. For the S-Pol radar data set, in both coarse grained interpolation and fine-grained interpolation, the Nearest neighbor interpolation best matches with the original variogram, the Inverse-distance-squared weighted averaging appeared the worst in preserving the original variogram. However, in general, most interpolation algorithms rank differently in the case of fine-grained interpolation and coarse grained interpolation. We observe the same inconsistency with another precipitation data set [WC]. Note that in the case of fine-grained interpolation, we verify whether the spatial correlation in the synthetic data matches that of the experimental data at the extent of coarse granularity, not at the scale of fine granularity, where we do not have ground

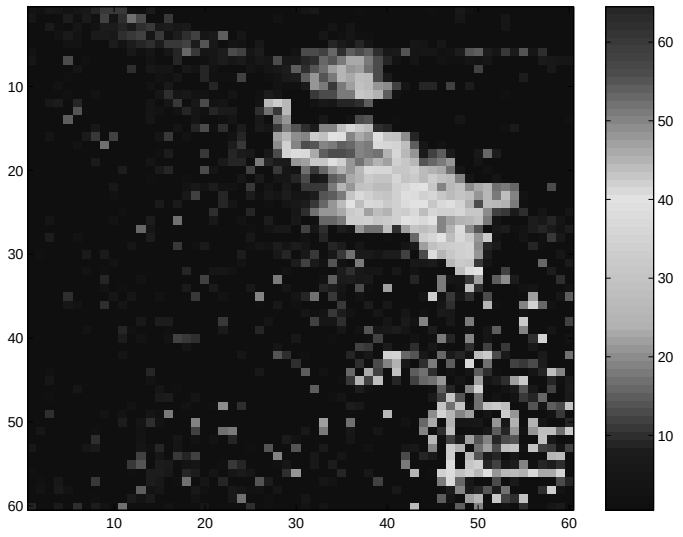



Figure 7.1: Spatial modeling example: original data map (60x60)

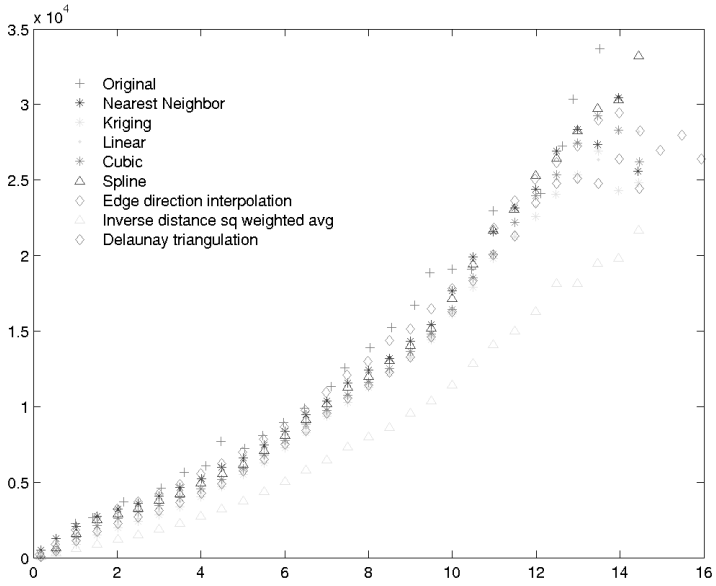



Figure 7.2: MSD of variogram values: Coarse grained interpolation results on a snapshot of radar data

truth data.

Based on these results we do not recommend one single interpolation algorithm over others; instead, we propose to use spatial correlation as the evaluation metric for our synthetic data generation purpose and a suite of interpolation algorithms. Given a new synthetic data generation task, we would test with different interpolation algorithms, select one that can best suit the algorithm and experimental data set under study. Note that although the Nearest neighbor interpolation appears best matching with the original variogram model, it is not appropriate in the case of ultra-fine grained interpolation, since it assigns all nodes in a local neighborhood the same value from the nearby sample. However, most physical phenomena have some degree of variation even in a small local neighborhood, thereby, we would not expect all sensors deployed in a local neighborhood report the same sensor readings as in the case of the nearest neighbor interpolation.

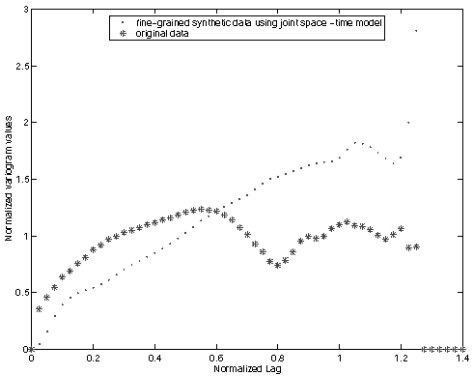



Figure 7.3: Spatial modeling example: Variogram of the fine-grained synthetic data and the original data

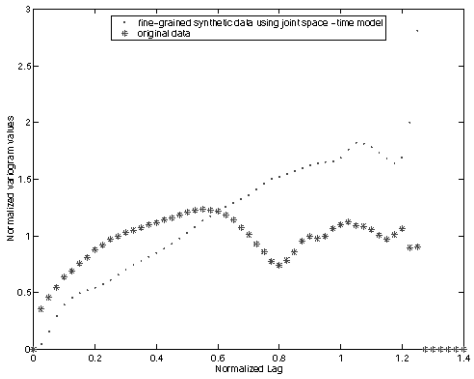



Figure 7.4: Joint modeling example: Variogram of the fine-grained synthetic data and the original data

| Name of method | MSD for coarse-grained interpolation |
|---|---|
| Nearest neighbor | 8.354218e+01 (1.836358e+01) |
| Edge directed | 1.970850e+02 (2.129320e+01) |
| Cubic | 2.000790e+02 (1.694163e+01) |
| Delaunay triangulation | 3.406270e+02 (4.795614e+01) |
| Linear | 3.941510e+02 (2.876476e+01) |
| Spline | 7.148526e+02 (5.5949e+01) |
| Kriging | 1.469954e+03 (1.913371e+04) |
| Inverse-dist.-squared-weighted avg. | 1.682726e+03 (3.617214e+02) |

Table 7.1: Mean Square Difference of variogram values for different interpolation algorithms in the increasing order of MSD for coarse-grained interpolation. Here we use median from 100 snapshots instead of mean to get rid of outliers, and list 95% confidence interval in the brackets.

**Summary:** As shown above, most interpolation algorithms can approximate the original variogram models. However, it can only be used to interpolate at unsampled locations, not unsampled time. Furthermore, spatial interpolation algorithms, including Kriging, is not able to characterize the correlation between the spatial domain and temporal domain of the data, such as how the time trend varies at each location and how the spatial correlation changes as time progresses. Next, we wish to use the joint space-time model to address the limitations of the spatial interpolation techniques alone.

### 7.2.2 Joint space-time modeling

When considering the time and space domains together, a spatial-temporal random process is often decomposed into a mean component modeling the trend, and a random residue component modeling the fluctuations around the trend in both the time and space domains. Formally,

$$Z(u_\alpha, t_i) = M(u_\alpha, t_i) + R(u_\alpha, t_i) \tag{7.11}$$

where $Z(u_\alpha, t_i)$ is the attribute value under study, $u_\alpha$ is the location, $t_i$ is the time, $M(u_\alpha, t_i)$ is the trend, and $R(u_\alpha, t_i)$ is the stationary residual component.

For the trend component, we borrowed the model from [KMK02] where Kyriakidis *et al.* built a space-time model for daily precipitation data in northern California coastal region. $M(u_\alpha, t_i)$, in Equation 7.11 is further modeled as the sum of $(K+1)$ basis functions of time, $f_k(t_i)$: $M(u_\alpha, t_i) = \sum_{k=0}^{K} b_k(u_\alpha) f_k(t_i))$ where

$f_k(t_i)$ is a function solely dependent on time $t_i$, with $f_0(t_i) = 1$ by convention. $b_k(u_\alpha)$ is the coefficient associated with the k-th function, $f_k(t_i)$, and is solely dependent on location $u_\alpha$. $B(u_\alpha)$ and $F(t_i)$ can be computed as follows.

We first describe the guidelines to compute $f_k(t_i)$. [KMK02] suggested that any temporal periodicities in the data should be incorporated in $f_k(t_i)$. Alternatively, $f_k(t_i)$ could also be identified as a set of orthogonal factors from Empirical Orthogonal Function (EOF) analysis of the data, or the spatial average of data at a time snapshot. In this paper, we use two basis functions: $f_0(t_i) = 1$ by convention; for $f_1(t_i)$, we take the spatial average of each time snapshot of the data. Formally, $F(t_i)$ (for illustration convenience, we write $f_k(t_i)$ and $b_k(u_\alpha)$ in matrix formats) can be written as:

$$\begin{pmatrix} 1 & \frac{1}{n}\sum_u z(u,t_1) \\ 1 & \frac{1}{n}\sum_u z(u,t_2) \\ .. & ... \\ 1 & \frac{1}{n}\sum_u z(u,t_i) \end{pmatrix} \tag{7.12}$$

Next let us see $B(u_\alpha)$. If we ignore the residue component in Equation 7.11 for now, $z(u_\alpha, t_i)$ can be written as

$$Z(u_\alpha, t_i) = B(u_\alpha) \cdot F(t_i) \tag{7.13}$$

The vector of coefficients $B(u_\alpha)$ can be written as a weighted linear combination of the data vector $Z(u_\alpha, t_i)$: $B(u_\alpha) = H(u_\alpha) \cdot Z(u_\alpha)$, where $H(u_\alpha)$ is a matrix of weights assigned to each data component of $Z(u_\alpha)$ and $Z(u_\alpha)$ is a vector consisting of a time series data at location $u_\alpha$. If the matrix F is of full rank, we have $H = (F' \cdot F)^{-1} \cdot F'$ from the ordinary least squares analysis (OLS).

The joint spatio-temporal trend model is constructed at each monitored location. The resulting trend parameters, $\{b_k(u_\alpha)\}$, are spatially correlated since they are derived from the same realization of the underlying spatio-temporal random process. Therefore, we spatially model and interpolate the trend parameters, $\{b_k(u_\alpha)\}$, using Kriging (or alternatively, other spatial interpolation techniques)

to obtain the value of $\{b_k(u_\alpha)\}$ at unsampled location $u_\alpha$. Similarly, $\{F_{t_i}\}$ can be modeled and interpolated to obtain the value of $F_t$ at unsampled time point $t$.

As mentioned above, a spatio-temporal random process was decomposed into a trend and a random residue component. The decomposition is subject to the particular model we used;

thereby, the residue component is not guaranteed to be stationary and free of any spatio-temporal trend. It is possible to further construct a spatio-temporal model on the residue. However, for simplicity, the residue is often modeled either as a function of time or as a function of location, such that interpolation can be done either in the spatial or in the temporal domain alone. For example, the residue could be modeled as the spatial average at each time instant. In the example presented next in Section 7.2.2.1, for simplicity, we treat the residue component as stationary noise.

#### 7.2.2.1 Evaluation of joint space-time modeling

To apply the joint space-time model described above, we considered a subset of the S-Pol radar data provided by NCAR. We selected a 70 x 70 spatial subset of the original data with 1km spacing, and 259 time snapshots across 2 days in May 2002. As mentioned above, the synthetic data is desired to have similar spatial correlation as the original data. Here we presented results from one snapshot to shed some light on how the synthetic data generated from the joint space-time model captures the spatial correlation of the original data. Figure 7.4 shows the variogram plot of the synthetic data (which is based on the joint space-time model) *vs.* original data, where the variogram value is normalized by the variance of the data, and the lag distance[2] is normalized by the maximum distance in

[2]lag distance is defined as the separation distance between two points

horizontal or vertical directions. For comparison, we also show the results from spatial Kriging for the same snapshot in Figure 7.3.

In Figure 7.4, the variogram of the synthetic data approximates that of the original one in the range [0, 0.6], which is less than half of the maximum distance between any two nodes; while in the spatial interpolation case (Figure 7.3), similar trends between two variogram curves (of synthetic vs. original data) are observed except in lags with range [1.1, 1.3], where the last few points in the variogram may be less accurate due to the fact that it is based on less data compared to other portions of the variogram. The larger discrepancy between the synthetic and original data in Figure 7.4 suggests that the joint space-time model does not capture the spatial correlation in the original data as precisely as the purely spatial interpolation does.

**Spatial modeling vs. Joint space-time model** A joint spatio-temporal model can capture the correlation between the temporal trend and spatial variation and generate synthetic data at unmonitored times and locations. Further, compared with spatial interpolation at each time snapshot, joint space-time modeling is faster when we need to generate large numbers of snapshots of spatial data. For example, the precipitation data set [WC] to be introduced next in section 8.1, includes daily precipitation data from the Pacific Northwest for approximately *45* years, or *16801* days. If we interpolate each daily snapshot separately, we would have to apply spatial interpolation technique 16801 times. However, using the joint space-time model described above, we need only to interpolate the coefficients $\{B_k(u_\alpha)\}$ $(k = 1, 2)$ in the spatial domain. This reduces the application of spatial interpolation to only twice and interpolating $F(t)$ in the temporal domain to only once.

The joint space-time model comes at a cost. It combines spatial and temporal domains that have completely different characteristics. Time is a dimension with completely different characteristics and consequently is non-trivial to combine with space. Models that are meaningful in one domain are sometimes meaningless in the other. For example, useful characteristics of temporal data can often be identified by looking at the spectrum of the data, but in a spatial setting, cyclic behavior is rarely observed, and is difficult to define. Therefore, a joint space-time model is usually more complicated and typically results in greater error than a space model. When we compare the prediction accuracy of spatial interpolation to the joint space-time model at the same time instant, a spatial interpolation technique usually can more closely capture the original data than a joint space-time model can.

In addition, spatial interpolation is often used as a component in the joint spatio-temporal models. Therefore, even though a joint space-time model naturally captures both the spatial and temporal characteristics in the data, the spatial modeling and interpolation is still important and popular in practice.

### 7.2.3 Generating synthetic data in an arbitrary topology

Next we briefly state how to generate irregular topology data by applying the nearest neighbor re-sampling method to the fine-grained data obtained from above. The methods we use are as follows:

First, we generate ultra fine-grained data using the spatial modeling and interpolation techniques discussed in section 7.2. The objective in this step is to create a grid topology at a much finer granularity than our target topology.

Second, we create irregular topology data. Our objective in this step is to down-sample the fine-grained data set from Step 1 and generate a data set for an

arbitrary topology. Among synthetic data sets generated from different interpolation and modeling techniques in step 1, we select the one that can best match the original experimental data in terms of the specific characteristics that we are interested in. We overlay the target topology on this ultra fine-grained grid data. Each node in the target topology is assigned a value from the nearest grid data. Note that the spatial interpolation techniques or joint space-time modeling techniques introduced above could be used to directly generate synthetic data at an arbitrary location and time. However, providing an ultra fine-grained data set allows the protocol designers to derive arbitrary topology data as they wish from the fine-grained data, thereby simplifying the generation of synthetic data in their chosen configuration.

### 7.2.4 Discussion on generating data of fine granularity from empirical models

In this section, we discuss whether the spatial interpolation and modeling techniques discussed in Section 7.2 apply to generating synthetic data of ultra-fine granularity. Due to lack of deployed sensor network systems, most existing experimental data are obtained from remote sensing or in-field instrumentation where phenomena are sparsely sampled, especially in the spatial domain. Coarse sampling resolution (*i.e.*, the sampling resolution is below the Nyquist rate) creates challenges for generating fine-grained synthetic data that can match the deployment density of future sensor networks, and at the same time, resemble the real-world phenomena.

By fine-grained interpolation, we refer to cases where we increase the scale by at least one order of magnitude in each dimension. In our experiment, we downsample the experimental data at 1/10th of the original sampling resolution in each

dimension; then we interpolate it back to the original scale using the techniques introduced in Section 7.2. We compare the spatial correlation of the interpolated data with that of the original data. As shown in Figure 7.5, the spatial correlation of the fine-grained interpolated data aptly approximates the experimental data well at larger scales, while tending to under-estimate the variogram values (*i.e.*, over-estimate the spatial correlations) at smaller scales where no samples are available.

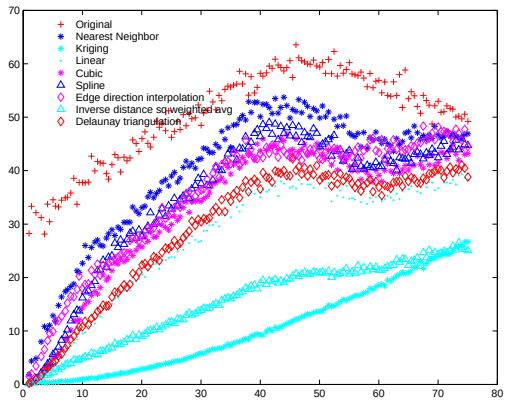



Figure 7.5: The spatial correlation of the synthetic data from fine-grained interpolation vs. that of the experimental data. The synthetic data approximates the experimental data better at a larger scale than a smaller scale (*i.e.*, near the origin) where we do not have sample data.

In the case of ultra fine-grained interpolation, we increase the sampling resolution by more than one order of magnitude. Because we do not have an empirical spatial correlation model at small scales, it is difficult to estimate the spatial correlation of the real phenomena at fine granularity. One reason is that most interpolation algorithms tend to smooth the existing samples, thereby smoothing out the non-continuity of the spatial correlation in real phenomena. Due to this smoothing effect, if the phenomena that we are trying to capture is smooth, then

we may be able to achieve reasonable approximation by applying the aforementioned interpolation techniques. However, if we priorly knew that a sensor field is smooth, a high density deployment is not necessary. Therefore, synthetic data of fine granularity is not indispensable in those deployment scenarios. On the other hand, if we have domain knowledge of the spatial correlation or other data characteristics at small scales, we can directly plug that into our model. The synthetic data based on empirical models at large scales together with domain knowledge at smaller scales may be able to capture the real phenomenon. Unfortunately, most of the time, we have no prior knowledge as to the smoothness of the phenomena nor a physical model at small scales. In general, we recommend applying the techniques introduced in section 7.3.1 to vary the spatial correlation model at small scales, thereby generating data with a wide range of spatial correlations at the appropriate scale.

## 7.3 Generating synthetic data with a wide range of parameters

When the experimental data spans across a wide range of parameter values it is sufficient to apply the techniques introduced above to generate synthetic data that can match the original experimental data. However, the available experimental data collected from relevant applications may be scarce and the range of the parameter space it covers may be limited. In order to evaluate the algorithm performance over a wide range of data input we provide techniques to generate synthetic traces exhibiting a wide range of data characteristics.

In this thesis we provide methods to directly adjust the spatial correlation and data distribution in the synthetic data.

### 7.3.1 Adjusting the spatial correlation

In general, sensor networks are going to be deployed in the physical environment and process data collected from the geometric world. Consequently, many data processing algorithms exploit spatial correlation in the data. Our study confirms that *spatial correlation* is an essential parameter to the algorithm performance in sensor networks. In Chapter 5 through two instances of data compression algorithms, namely, joint entropy coding and wavelet compression, we have identified a strong relationship between the spatial correlation of the data input and the algorithm performance. This relationship is demonstrated by a high correlation coefficient.

Next we describe a synthetic data generation method which provides a knob to directly vary the spatial correlation in the data. This allows algorithm evaluation against data that contain a wide range of spatial correlations.

#### 7.3.1.1 Generate synthetic data with spatial correlation different from what is in the experimental data

As described in Section 7.2.1.2, Kriging directly models the spatial correlation in the data and uses it in the estimation of sensor values at unmonitored locations. In Ordinary Kriging at unmonitored locations the data is estimated as a weighted average of the neighboring samples. Under the criteria of minimizing the estimation variance, the weights parameter can be derived from Equation 7.10, 7.7 and 7.9 applied here in this order. Variogram values are required in matrix 7.7 and 7.9. Based on Equation 7.1, variogram values $\gamma(x_i, x_j)$ can be obtained from sample data sets. However, $\gamma(x_i, x_0)$ is not necessarily covered by the sample data set, *i.e.*, the distance between $x_i$ and $x_0$ may not appear in the separation distance between any pair of nodes $x_i$ and $x_j$ in the sample data set. Instead,

$\gamma(x_i, x_0)$ can be derived from sample data in the following manner: First, compute sample variograms from the sample data: for a set of samples, $z(x_i)$, i=1, 2, ..., $\gamma(h)$ can be estimated by

$$\hat{\gamma}(h) = \frac{1}{2m(h)} \sum_{i=1}^{m(h)} \{z(x_i) - z(x_i + h)\}^2 \tag{7.14}$$

where $m(h)$ is the number of samples separated by the lag distance $h$.

Second, use a random function (usually in the form of a certain parametric model) to fit the sample variograms computed from the first step. The family of the parametric model can be determined based on domain knowledge. In the absence of domain knowledge, a common practice is to plot the sample variograms *vs.* separation distances then determine the family of the parametric functions from the shape of the original sample variograms. The unknown parameters in the function can be estimated using various function estimation techniques, *e.g.*, the ordinary least squares method.

The synthetic data generated from the above procedure will approximate the spatial correlation in the original experimental data since the variogram values are either derived directly from the sample data or from a parametric function obtained from fitting the original sample variograms.

Alternatively, assigning $\gamma(x_i, x_j)$ and $\gamma(x_i, x_0)$ to a different value will generate synthetic data with a different spatial correlation feature from that of the experimental data. Consequently, by varying the variogram values in Equations 7.7 and 7.9 across a wide range, we will be able to generate synthetic data with a broad range of spatial correlations.

#### 7.3.1.2 Implementation and results in adjusting the spatial correlation

We slightly modify the Kriging procedure introduced in section 7.2.1.2. In Equations 7.7 and 7.9, instead of using a function derived from fitting the sample variograms, we plug in different families of variogram models and vary the parameters in each parametric function. We implemented this procedure using a publicly available statistical software package, *gstat* [Peb03] in *R* [Sta99]. Across different families of parametric models, there are three common parameters that we adjust to vary the variogram models used in the weights estimation: partial sill, nugget, and range. In summary, we provide the following knobs in adjusting the spatial correlation: (a) family of parametric function; and (b) partial sill, nugget, and range in the variogram model. Next we briefly explain these parameters illustrated in Figure 7.6 [Web].

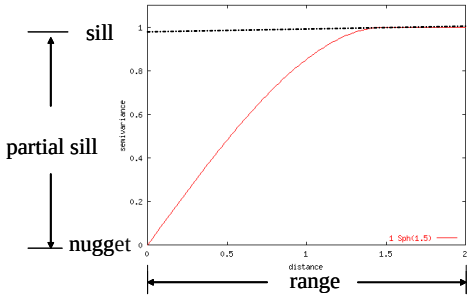



Figure 7.6: Illustration of variogram modeling

**Range:** Initially, as the separation distance between pairs of samples increases the corresponding variogram value will also increase. However, after this separation distance is larger than a certain value, the eventual corresponding variogram value stops increasing. We call the distance at which the variogram value stops

increasing the *range* of this variogram model.

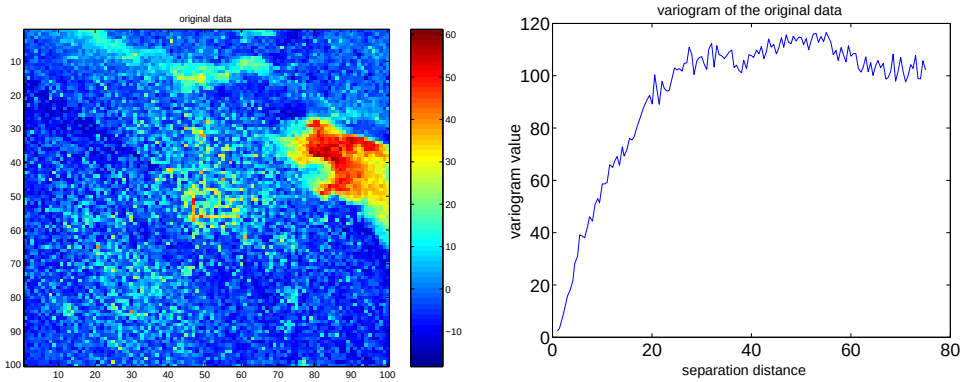
**Sill:** The maximum value that a variogram model allows.

**Nugget:** The variogram value at separation distance 0 is 0, *i.e.*, $V(0) = 0$ where $V(h)$ is the variogram value at separation distance $h$. However, under certain conditions, the variogram value at a very small distance is non-zero, *i.e.*, $\underline{\lim}_{h \to 0} V(h) = v_0$ and $v_0 \neq 0$. This is called the *nugget effect*.

**Partial sill:** This is the difference between the *sill* and the *nugget*.

Next we use an example to illustrate what dynamic range of spatial correlations can be achieved by adjusting the above parameters. In our experiment, we used a snapshot of S-Pol radar data, a 100x100 spatial data set. We downsample it by a 4:1 ratio in each dimension, resulting in a 25x25 spatial data set. Using *the modified Kriging procedure* introduced above, we adjust the variogram models and corresponding parameters to generate multiple 100x100 spatial data sets. In the results presented next, we used a sphere model in which we varied 1) the *sill* from 20 to 180, 2) the *range* from 10 to 70, and 3) the *nugget* from 1 to 100. In total, these variations resulted in 236 combinations.

The original data and its variogram plot is shown in Figure 7.7. Compared to a single spatial correlation value at a certain distance, the variogram plot fully characterizes the spatial correlation of the experimental data in a wider distance range. Since it is impossible to enumerate variogram plots of 236 synthetic data sets, we presented here a few synthetic data sets and their variogram plots. This demonstrates that by adjusting the variogram models based on a single experimental spatial data set, we obtained synthetic data with a wide range of spatial correlations. The image of the synthetic data are shown in Figure 7.9 and their




(a) Original data

(b) Variogram plot of the original data

Figure 7.7: Image and variogram plot of the original data

corresponding variogram plots are shown in Figure 7.8. Note that the variogram values shown in Figure 7.8 tend to be smaller than the original variogram values at the same distance as shown in Figure 7.7. Our explanation is that, in general, Kriging tends to smooth the original experimental data, thus making the spatial correlation of the synthetic data appear stronger than that of the experimental data.

In addition to adjusting the spatial correlation in the synthetic data, adjusting the parameters in the variogram model also provides us a supplementary benefit of adjusting the smoothness of the data. As shown in Figure 7.10, adjusting the relative nugget effect will change the smoothness of the data. In general, the larger the relative nugget effect, the smoother the data tends to be.

### 7.3.2 Adjusting the data distribution

Data distribution is an essential characteristic to many statistics estimation problems. Specifically, in Section 4 we demonstrated that the corresponding $p$-

*percentile* bin size is an essential and sufficient parameter to determine the performance of *percentile estimation*. The synthetic data generation technique introduced next provides a knob to directly adjust the corresponding *p-percentile* bin size Specifically, the problem of generating synthetic data based on the given percentile bin size is formulated as follows: given the size of the $p$-percentile bin, $s_p$, generate synthetic data whose $p$-percentile will match $s_p$.

We decompose the above problem into the following two sub-problems: (1) Given the size of the $p$-percentile bin, $s_p$, generate the histogram representation of a data distribution, in which the corresponding $p$-percentile will match $s_p$. (2) Generate a synthetic data set whose data distribution will match the given histogram.

A *probability density function* can be used to fully characterize the data distribution of a *random variable* (or a *random process*). However, it is difficult to quantitatively specify a *pdf* function unless that *pdf* function under study can be characterized by a parametric model. As we have shown in Section 6.1, data generated from simple parametric models alone are not sufficient to fully evaluate algorithm performance. On the other hand, the histogram of a *random variable* contains most of the statistics that we are interested in. For example, the *corresponding percentile bin size* can be derived from the histogram of a data distribution. Therefore, instead of *pdf*, a histogram is used as the representation of the underlying data distribution.

Corresponding to the first sub-problem, we describe a procedure to generate the histogram representation of the specified data distribution. More precisely, the inputs of this step includes: (1) the $p$-percentile bin size, denoted as $s_p$; (2) the total number of bins in the histogram representation, denoted as $n$; (3) and the range of the data, denoted as $[a, b]$. The output of this step is a histogram whose

characteristics satisfies the given input parameters. Specifically, the histogram is represented by a set of bins, $[A_i, B_i] : P_i, i = 1 \cdots n$, where $[A_i, B_i]$ is the range of the bin $i$, $A_1 = a$, $B_n = b$, and $P_i$ is the normalized bin size, *i.e.*, the percentage of the samples that falls into bin $i$.

Without loss of generality, we consider equally-spaced bins. As a result, the ranges of the bins are specified as $[a + (i-1) * \frac{b-a}{n}, a + i * \frac{b-a}{n}]$, where $n$ is the total number of bins. The remaining unknown are the normalized bin size $P_i$. Obviously, the solution is not unique. Without sacrificing generality, we propose the following procedure as a solution:

1. Generate a random number $x$ from $U[1, n]$, where $U$ stands for uniform distribution. We assign bin $x$ to be the bin that holds the $p$-percentile. Thus, the size of bin $x$, $P_x = s_p$.

2. Resolve the following system of equalities and inequalities.

$$\sum_{j=1}^{x-1} P_j + \sum_{j=x+1}^{n} P_j = (1 - s_p), \tag{7.15}$$

$$(p - s_p) < \sum_{j=1}^{x-1} P_j < p, \tag{7.16}$$

$$(1 - p - s_p) < \sum_{j=x+1}^{n} P_j < (1 - p), \tag{7.17}$$

   Any set of parameters $P_j$ that satisfies the above system of equalities and inequalities is a solution. Even though it will not affect the correctness of our solution, we again use a stochastic method to avoid any deterministic outcome:

   (a) Generate a random number $k$ from $U(0, s_p)$.

   (b) Select $P_j$, where $j = 1, \cdots, x-1$, *s.t.*, $\sum_{j=1}^{x-1} P_j = p - k$.

Without adding a boundary check, the pseudo code to achieve this is:

**Algorithm 7.3.1:** GENERATELOWHALFBINSIZE$(p, k, x)$

$SumBinSize = 0$;
**for** $j \leftarrow 1$ **to** $(x - 2)$
  $CurCredit = p - k - SumBinSize$;
  generate a random number $l$ from U(0, CurCredit);
  $P_j = l$;
  $SumBinSize+ = l$;
$P_{x-1} = p - k - SumBinSize$;

(c) Select $P_j$, where $j = x + 1, \cdots, n$, *s.t.*, $\sum_{j=x+1}^{n} P_j = 1 + k - p - s_p$.

Similarly, the pseudo code to achieve this is:

**Algorithm 7.3.2:** GENERATEHIGHHALFBINSIZE$(p, k, x)$

$SumBinSize = 0$;
**for** $j \leftarrow (x + 1)$ **to** $(n - 1)$
  $CurCredit = 1 + k - p - s_p - SumBinSize$;
  generate a random number $h$ from U(0, CurCredit);
  $P_j = h$;
  $SumBinSize+ = h$;
$P_n = 1 + k - p - s_p - SumBinSize$;

Next we address the other half of the problem. We first formally define the problem of generating a synthetic data set from a given histogram specification. The histogram is specified by its component bins, $\{[A_i, B_i] : P_i\}, i = 1 \cdots n$, where $[A_i, B_i]$ is the range of bin $i$, and $P_i$ is the normalized bin size, which is defined to be the percentage of samples that falls into bin $i$. Below we describe our

approach based on a modified Monte Carlo simulation. Similar to most Monte Carlo simulations, our sampling procedure consists of three steps:

1. Generate a random number $\epsilon$ from $U[0,1]$, where $U$ stands for uniform distribution,

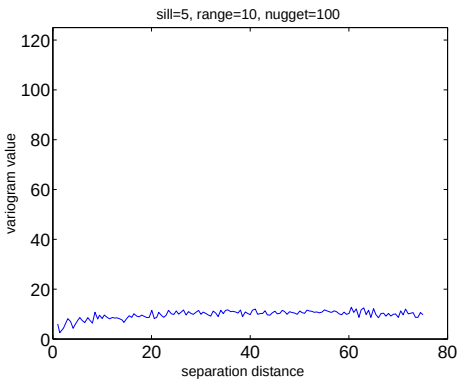
2. Instead of using *cdf* in standard Monte Carlo simulations, we define *cumulative histogram* as follows: $Q_i = \sum_{j=1}^{i} P_j$, where $i = 1 \cdots n$. Find $i$ that satisfies $Q_{i-1} < \epsilon \leq Q_i$

3. Unlike *cdf*, we do not have a unique inversion of the cumulated histogram. *i.e.*, we do not have a one-to-one mapping from a *cumulative histogram* value $Q_i$ to a single realization of the random variable. We slightly modify this *cdf* inversion step in the Monte Carlo simulation as follows. After we determine $i$ that satisfies $Q_{i-1} < \epsilon \leq Q_i$, we output a random number $x$ from $\mathrm{U}[A_i, B_i]$.

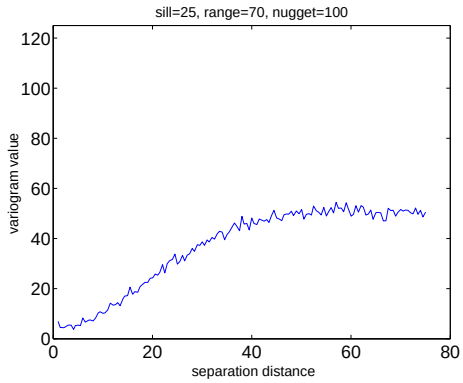
## 7.4 Summary

In this chapter, we present our scalable synthetic data generation framework. We propose two strategies to deal with the huge parameter space in characterizing the data input: first, we use experimental data to guide our simulation; and second, we identify a small number of parameters essential to the algorithm performance. Guided by these two strategies, we provide two types of synthetic data generation techniques: (1) to generate irregular topology data based on models of the experimental data, which will maintain the criteria of verisimilitude in the experimental data; and (2) to generate data that corresponds to a wide range of parameter values which serve as a complement to the algorithms in the first category when the available experimental data is scarce.

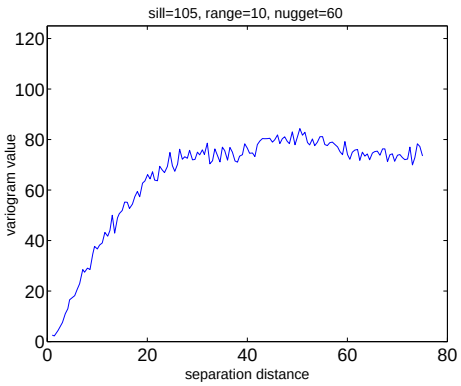
Identifying a small number of parameters essential to the algorithm performance not only reduces the search space in the synthetic data generation, but also allows us to directly evaluate whether the synthetic data captures the important features of the experimental data, or whether the synthetic data cover a wide range of parameter values in the dimension of interest. For example, spatial correlation is identified as a feature essential to many sensor network algorithms. Therefore, the Mean Squares Difference between the spatial correlation of the synthetic data and that of the experimental data is used as a quantitative metric for synthetic data evaluation. Our evaluation results demonstrate that data generated from most interpolation algorithms can closely approximate the spatial correlation of the experimental data.




(a) S=5, R=10, N=100



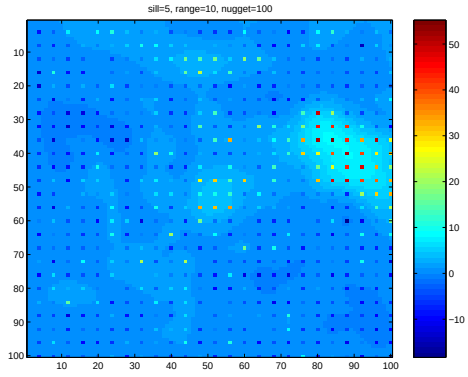

(b) S=25, R=70, N=100



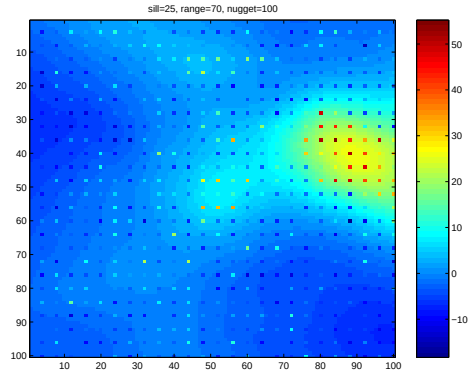

(c) S=105, R=10, N=60


(d) S=165, R=10, N=1

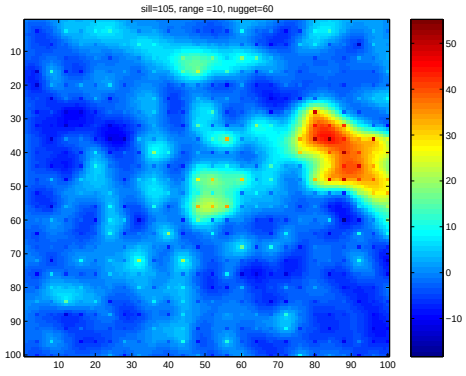
Figure 7.8: variogram plot of the resulting synthetic data by adjusting the range (R), relative nugget effect (N) and partial sill (S) parameters in the variogram model.



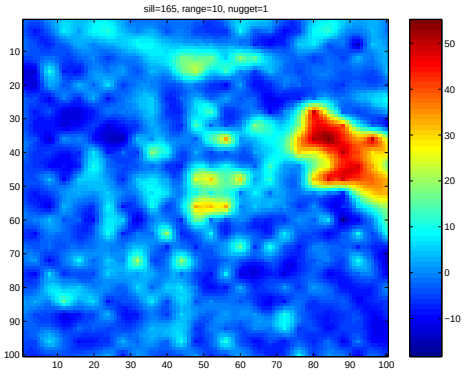

(a) S=5, R=10, N=100




(b) S=25, R=70, N=100

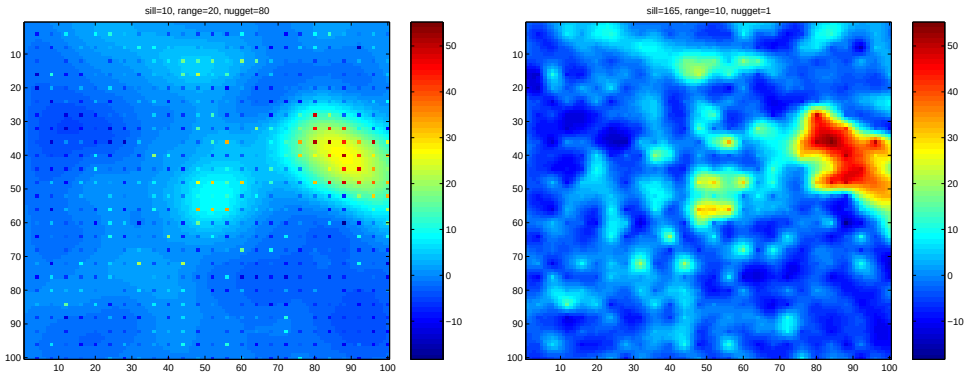



(c) S=105, R=10, N=60




(d) S=165, R=10, N=1

Figure 7.9: Synthetic data from adjusting the range (R), relative nugget effect (N) and partial sill (S) parameters in the variogram model.




(a) rough data where you can see patches clearly:S=10, R=20, N=80

(b) smooth data:S=165, R=10, N=1

Figure 7.10: Smoothness of the synthetic data: Synthetic data from adjusting the range (R), relative nugget effect (N) and partial sill (S) parameters in the variogram model.

# CHAPTER 8

# Algorithm evaluation using our synthetic data

In Chapter 7, the synthetic data was directly evaluated in terms of the important identified parameters. Specifically, we examine whether the synthetic data captures the important features of the experimental data, or whether the synthetic data cover a wide wide range of parameter values in the dimension that we are interested in. In this chapter we will evaluate the utility of the synthetic data through algorithm evaluation using our synthetic data. First, we use a case study of the DIMENSIONS system to demonstrate that synthetic data allows algorithm evaluation over realistic data of an arbitrary irregular topology. The performance difference over regular data and irregular topology data demonstrates the need to improve the standard wavelet compression algorithm to handle irregular data. In addition to its utility on irregular topology data, algorithm evaluation with synthetic data provides similar insights as the experimental data in the case of Fidelity Driven Sampling and median computation.

## 8.1 Algorithm evaluation using synthetic data of irregular topology: DIMENSIONS

In this section, we use DIMENSIONS [GEH02] as an example of how synthetic data generated from empirical models aids in evaluating the performance of an algorithm. In particular, when experimental data sets are available only from

a regular grid topology, we show how synthetic data generation allows us to evaluate DIMENSIONS over irregular topologies. Next, we illustrate how to use the procedure described in Section 7.2.3 to generate irregular topology data sets:

1. Generating Ultra Fine-grained Data. We create a grid topology at a much finer granularity than our target topology. We model the correlation in the experimental data using the joint spatio-temporal model described in Section 7.2.2 and further generate a much finer grained data set based on this empirical model.

   In this case study, we consider a rainfall data set that provides 50km resolution of daily precipitation data for the Pacific NorthWest from 1949-1994 [WC]. The spatial setup comprises a 15x12 grid of nodes where each node recorded daily precipitation values. Since the data set covers 45 years of daily precipitation data, it is rich enough in the temporal dimension for the purpose of this study. Thus, we increase the data granularity only in the spatial dimension leaving the granularity of the temporal dimension unchanged. This fine-grained model was used to interpolate 9 points between every pair of points in both horizontal and vertical dimension, thus resulting in a 140 x 110 grid data.

2. Create Irregular Topology Data. We overlay the target topology on this ultra fine-grained grid data. Each node in the target topology is assigned a value from the nearest neighboring grid data.

   To create a random topology with a predefined number of nodes, we select grid points at random from the fine-grained grid data. In our case study, the nodes in the 140 x 110 fine-grained grid were randomly sampled with a probability of 2%. This results in a network of approximately 14x11x2 nodes.

Next we demonstrate how the synthetic data generated from the above procedure helps to expose problems and gain more insights into the current DIMENSIONS system design.

**DIMENSIONS** We now return to our case study to illustrate how data processing algorithms can be sensitive to topology features. DIMENSIONS [GEH02] proposes wavelet-based multi-resolution summarization and drill-down querying. Summaries are generated in a multi-resolution manner corresponding to different spatial and temporal scales. Queries on such data are posed in a drill-down manner, *i.e.*, they are first processed on coarse, highly compressed summaries corresponding to larger spatio-temporal volumes. The approximate results obtained are used to focus on regions in the network that are most likely to contain the results.

The standard wavelet compression algorithm used in DIMENSIONS assumes a grid topology in the data and cannot directly handle an irregular placement of nodes. However, sensor network topologies are more likely to be irregular. To make the standard wavelet compression algorithm work with irregular topologies without changing the wavelet algorithm itself, DIMENSIONS first convert irregular topology data into regular grid data before applying the standard wavelet compression algorithm. The regularizing procedure works as follows:

- Choosing a coarse grid. Choose a grid size that is coarser than the fine-grained data. In this case, a 14x11 grid is overlaid onto the 140x110 grid acquired in step 1. An average of 2 nodes per grid are expected.

- Averaging. For each such grid generated by step 1, average data from all nodes in the grid. Averaging data from multiple nodes will smooth the data and also reduce the noise in data. If there are N sensors in a certain

grid cell, the original noise per sensor is $\sigma^2$ and the noise distribution at each sensor node is *i.i.d.*, then averaging the data obtains an aggregated measurement with variance $\sigma^2/N$ (from the CLT theorem).

- Interpolation for empty grid cells. If there are no nodes in a cell, we assign to it the value of the nearest non-empty cell. Ties are broken randomly. This can be easily implemented in a distributed setting: a cluster head that is supposed to receive data from four lower level nodes but receives data from only three of them, fills in the empty grid with the interpolated data. Finally, the resulting regular data is passed to the standard grid-based wavelet compression.

However, this data regularizing process may skew the original data and introduce error in the query processing. We use a simple example to illustrate this skew.

Figure 8.1 shows an irregular topology. The solid black circles represent real nodes, each labeled with a sensor reading. To convert the topology into a grid, we overlay a grid on top of the original irregular topology. In general, when we overlay a grid onto an irregular topology, the following may be observed: (1) multiple data may appear in an overlaid grid cell; (2) some grid cells may be empty. The first scenario will most likely occur in a coarse partition of the topology while the second scenario will most likely occur when the region is divided in a very fine-grained manner. In standard grid-based wavelet processing, at the base level of the data hierarchy, one data point is required from each grid. Thus we need to address the cases in which either multiple data points are packed into one cell or a cell is empty.

Using the regularizing approach described above we establish a grid over the irregular topology in Figure 8.1 where the shadow circle represents a virtual node

with values computed from the averaging procedure or nearest neighbor interpolation procedure described above. For a query about *the maximum value in the sensor field*, assuming no compression error, this virtual node configuration will generate a reply of *28*, but the real maximum sensor reading is *38*. The discrepancy is due to smoothing in the interpolation. Using the same data configuration for a query about *average sensor reading*, this virtual node configuration will generate an output of *14*, but the real average is *15*. The discrepancy is again due to interpolation and averaging when converting the original data into a grid.

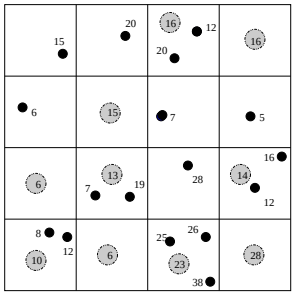



Figure 8.1: Irregular topology: converting an irregular topology to a grid skews the data as shown above

In summary, the procedure to convert an irregular topology to grid data will introduce some error. Different irregular topologies will likely introduce different amounts of error. Further, for the same topology, different procedures used to construct a grid over the original data will also deliver different results.

Next, we compare the algorithm performance under regular and irregular topologies. For regular topologies, we use the experimental precipitation data. For irregular topologies, we use synthetic irregular data sets generated from the joint spatio-temporal model explained in section 7.2.2.

### 8.1.1 Evaluation of DIMENSIONS using the grid data set

We evaluate DIMENSIONS [GEH02] using a rainfall data set that provides 50km resolution daily precipitation data for the Pacific NorthWest from 1949-1994 [WC]. The spatial setup comprises a 15x12 grid of nodes with 50-km spacing, each node recording daily precipitation values. While presumably this data set is at a significantly larger scale than what we would envision in a densely deployed sensor network, the data exhibits spatio-temporal correlations providing a useful algorithm performance case study. A variety of queries can be posted on such a data set, such as range-sum queries, *e.g.*, total precipitation over a specified period from a single node or a region; or drill-down extreme value queries. In our comparison we use maximum value queries as an example to demonstrate the algorithm performance. We present performance results for two queries: (a) GlobalDailyMaxQuery: that investigates which node receives max precipitation for a day in year X? (b) GlobalYearlyMax Query: that investigates which node receives the max precipitation for year X? The evaluation metric we used in this paper is mean square error. The error is defined as the difference between the measured query answer by dimensions and the real answer as calculated by an optimal global algorithm with the same data input, normalized by the real answer. If we consider 15x12 grid data input as a coarse sample of the original phenomena, this mean square error can be deemed as the error in the system output compared with an approximate ground truth. Figure 8.2 and 8.3 show the mean square error vs. the level at which the drill-down query processing terminates.

In the DIMENSIONS hierarchy, each lower level stores twice the amount of data as the higher level. Therefore, as query processing proceeds down the DIMENSIONS hierarchy, consequently gaining access to more detailed information,

as expected, the mean square error drops down gradually (shown in Figure 8.2 and 8.3). Moreover, the marginal improvement decreases with increasing levels. For instance, as shown in Figure 8.3 the marginal improvement in accuracy at the topmost level of information is approximately 66%; at the next level is 15%; followed by 6% and 2% respectively. Figure 8.2 exhibits similar results.

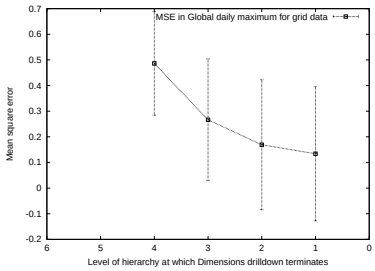



Figure 8.2: Error vs. Query termination level: Global daily maximum over original rainfall data

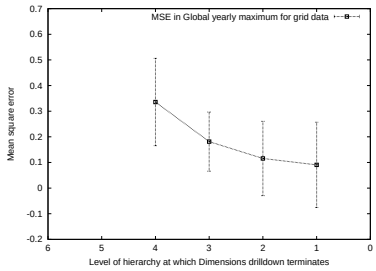



Figure 8.3: Error vs. Query termination level: Global yearly maximum over original rainfall data

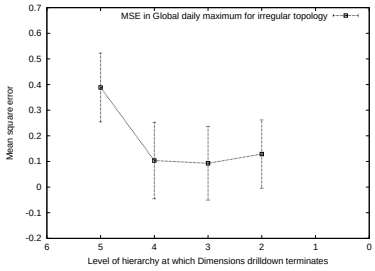



Figure 8.4: Error vs. Query termination level: Global daily maximum over irregular topology

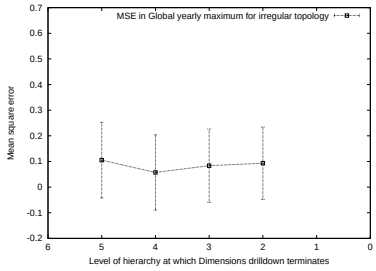



Figure 8.5: Error vs. Query termination level: Global yearly maximum over irregular topology

### 8.1.2 Evaluation of DIMENSIONS using the irregular data set

In this section, we present the algorithm performance over an irregular topology. The precipitation data set we used to evaluate DIMENSIONS in section 8.1.1 is from a grid topology, thus it cannot be directly used here. On the other hand, the performance of the wavelet compression algorithm is sensitive to the correlation pattern in the data and the space of data correlation is too big to simulate exhaustively. Therefore, a randomly generated data set is considered unrealistic and of little value in the DIMENSIONS evaluation. Instead, we use an irregular topology data set derived from models of the precipitation data that we used in section 8.1.1. Following the procedure described in the beginning of section 8.1, we generate a random topology consisting of 14x11x2 nodes from the precipitation data. The current implementation of the wavelet compression algorithm in the DIMENSIONS system works only with a regular grid topology. Thus, the regularizing step is used to convert the irregular data set into regular grid data.

#### 8.1.2.1 Results and comparison

Due to the time consuming nature of wavelet operations, we illustrate our results using a single topology. The results presented in this section are averaged for multiple (*i.e.,* 44) queries over one irregular topology.

Figure 8.4 and 8.5 show the results for the same queries as presented in section 8.1.1, GlobalDailyMax and GlobalYearlyMax, but for the irregular topology data. As in the regular grid case, we present the results in the form of the mean square error vs. the level at which the drill-down query processing terminates. The error here is the difference between the output of DIMENSIONS system with 14x11x2 irregularly placed nodes as input and with the approximate ground truth

as input. Here, the ground truth is approximated in the following way. Since the 14x11x2 irregularly placed nodes were selected from the 140x110 fine grained data, we first overlay a 14x11 grid over the 140x110 fine grained data. For each grid, we average 100 fine grained data points in the grid and obtain a value for each 14x11 grid, and then input these data to the global algorithm. The ground truth was approximated by the output of this global algorithm.

As shown in Figure 8.4 and 8.5, DIMENSIONS behaves differently in an irregular setting from a regular one. In the regular case (as shown in Figure 8.2 and 8.3), we observe gradual error improvement with more levels of drill-down. However, in the irregular case (Figure 8.4 and 8.5), there is no consistent error improvement with more levels of drill down. For the topology that we studied, in some cases (Figure 8.4 and 8.5) the query error actually increases as more levels are drilled down. Such a behavior would not be expected in a regular grid topology. This may be due to the effects caused by holes in topology and our corresponding interpolation procedure. An irregular topology might result in large regions where there is no data available, and as described above, interpolation is used to fill in the empty cells in the regularization process. However, interpolation may smooth and therefore skew the data, which will introduce error in the query processing. Note that if the graphs are averaged over multiple irregular topologies, some of the kinks in Figure 8.4 and 8.5 might be straightened out. However, we do not expect the same consistent improvement or the same order of improvement as with the regular grid data case.

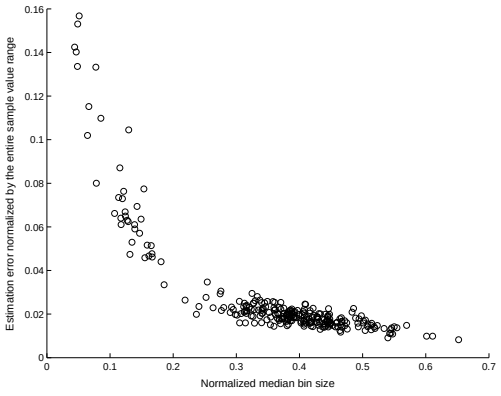
#### 8.1.2.2 Discussions

The error that the user perceived and approximated by our computed mean square error is subject to multiple factors. We just name a few: the physical node

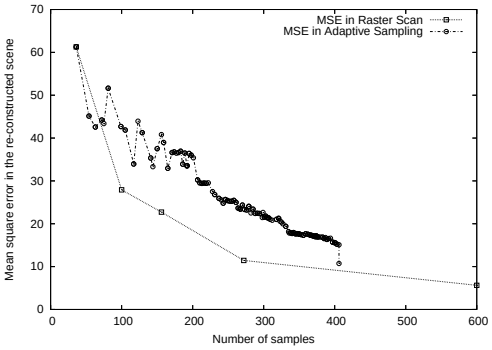
topology, the data correlation statistics, *e.g.,* the spatio-temporal model and its interaction with wavelet compression; and the mechanism used to interpolate (in our case, the nearest neighbor) in the regularizing procedure. If our objective is to study how irregular topology affects system performance, ideally we want to vary only the topology and keep other parameters fixed. One way to achieve this is by modeling the data correlation in the experimental data and then using the same empirical model to generate regular and irregular topologies, which we will use to evaluate the algorithm's performance over regular and irregular data respectively. More specifically, we could use the fine grained data generated from the data modeling process described in section 7.2 as the empirical model. We sub-sample the same fine grained data to obtain the regular and irregular sampled data as input to the DIMENSIONS system. The fine grained data is overly interpolated, thus it is deemed as an approximate model of the phenomena and also used as the approximate ground truth to compute the mean square error. The error is thereby computed as: $\frac{QueryResponseOverDimesions - QueryResponseOverModel}{QueryResponseOverModel}$. We plan to explore this in our future work.

In summary, we used the synthetic data generated from modeling the spatio-temporal correlation in the experimental data to evaluate DIMENSIONS over an irregular data set. DIMENSIONS in an irregular setting exhibited different behavior from the regular setting. This exposes the problem of DIMENSIONS's current regularization scheme. Thus our synthetic data helped to systematically evaluate the algorithm and point out needed improvements.

In our case study, the precipitation data is from a regular grid topology. However, even if we have experimental data from irregular topologies, our proposed synthetic data generation will enable evaluating algorithms over different irregular topologies rather than the particular setting used in the experiment. More im-




(a) Median computation results, similar to Figure 4.3(a), estimation error decreases with increasing normalized median bin size




(b) Fidelity Driven Sampling results: similar to Figure 6.2(d), MSE achieved by Fidelity Driven Sampling is worse than Raster Scan

Figure 8.6: Evaluation results using synthetic data generated from the radar data: the algorithm exhibits similar behavior as the evaluation results using the experimental data.

portantly, it also allows us to evaluate algorithms over the same underlying data correlation model but different topology settings. This allows us to study how the algorithm performance interacts with different parameters independently.

## 8.2 Algorithm evaluation using synthetic data across a wide range of parameters: Median computation by random sampling

Even though we demonstrated that a simple improvement to the percentile estimation by uniform sampling approach reduces the algorithm's sensitivity to data distribution, in general, it is difficult to design an algorithm whose behavior

is not sensitive to data input. Therefore, we recommend evaluating algorithms with data across a wide range of parameter values in order to investigate in which regimes the algorithm performs adequately. The parameter of interest could be data distribution, spatial correlation, or other data characteristics. The synthetic data generation approach discussed in Section 7.3 can be used to generate realistic data sets corresponding to a wide range of parameter values.

The statistical approach introduced in Section 4.1.1 can be used to systematically study how the algorithm behaves in different parts of the parameter space. The challenge of the systematic study is to identify what set of data characteristics best defines the data dependency for a given algorithm. In Chapters 3, 4, 5, 6, we identify a small number of important parameters for the studied algorithms. In general, identifying the relevant set of data characteristics usually requires a fair understanding of the algorithm under evaluation.

We evaluated the median computation by uniform sampling algorithm discussed in Section 4.1.1 using synthetic data generated from the empirical models of the S-Pol radar data. Figure 8.6(a) verifies that the median computation algorithm evaluated with our synthetic data exhibits similar behavior as that evaluated with the experimental data (Figure 4.3(a)): a) a strong correlation between the estimation accuracy and the corresponding *p-percentile* bin size in the data and b) the algorithm performance demonstrates a wide range under various data input. In addition, our synthetic data cover a similar range of data distribution as that of the experimental data.

## 8.3 Algorithm evaluation using synthetic data with realistic data features: Fidelity Driven sampling

We also apply the same synthetic data used above derived from the S-Pol radar data to the Fidelity Driven Sampling algorithm. The evaluation results (Figure 8.6(b)) provide similar insight as the evaluation using data collected from a lab environment (Figure 6.2(d)). Results indicate that the MSE achieved by Fidelity Driven Sampling is slightly higher than Raster Scan; whereas, when evaluated with data simulated from simple models (Figure 6.1), the MSE achieved by Fidelity Driven Sampling is several orders of magnitudes smaller than raster scan.

## 8.4 Summary

In this chapter, we evaluated the utility of our synthetic data through algorithm evaluation. First, in the case study of the DIMENSIONS system, synthetic data allows algorithm evaluation over realistic data of an arbitrary irregular topology. The performance difference over regular data and irregular topology data demonstrates the need to improve the standard wavelet compression algorithm. In addition to its utility on irregular topology data, we also demonstrate the utilities of the synthetic data in the aspects of realistic data features and data spanning across a wide range of parameter values.

# CHAPTER 9

# Conclusion and Future Directions

We conclude this thesis by reviewing our contributions and presenting our future directions.

## 9.1 Contributions of this Thesis

To support algorithm evaluation with realistic data input across a wide range of parameter values, we propose techniques to generate irregular topology data. Data sensitivity and algorithm evaluation requiring better models are not unique to sensor networks. Our major contributions are in identifying the extent of this problem through statistical performance analysis and evaluation in the context of sensor networks. Further, we propose a scalable synthetic data generation framework to support systematic algorithm evaluation and robust algorithm design. Our contributions fall into the following categories:

**Search Space Reduction** Using case studies of concrete sensor network algorithms we identify a small number of parameters to manipulate in our synthetic data generation. This reduces the search space significantly from exponential to a manageable number.

In this thesis, we investigate three classes of data processing algorithms that are potentially sensitive to data input. Through statistical performance analysis

and algorithm evaluation using experimental data, we identify a small set of data characteristics essential to algorithm performance. Furthermore, we demonstrate the need to evaluate algorithms using realistic data corresponding to a wide range of parameter values and irregular topology input.

**Systematic performance evaluation techniques** In our case studies, we apply two types of performance evaluation techniques. (1) When the algorithm performance can be attributed to a single parameter, we use standard non-parametric statistical tools to examine how the algorithm performance varies with the changing parameter value. Identifying the parameter essential to algorithm performance often requires knowledge of the algorithm under evaluation. (2) When the algorithm performance depends on multiple parameters, we use synthetically generated scenarios to investigate how each parameter changes the algorithm performance. In the synthetically generated scenarios, we have the flexibility to vary the phenomena along a single dimension while keeping other parameters fixed.

**Techniques to Generate Irregular Topology Data** We propose two types of synthetic data generation techniques. First, we use empirical models derived from experimental data to guide synthetic data generation towards those portions of the parameter space that represent real world scenarios. The synthetic data generation techniques in this category will create irregular topology data with similar characteristics as the experimental data. Second, when the available experimental data is scarce, we provide methods to generate synthetic traces exhibiting a wide range of data characteristics over the parameter of interest to applications.

**Implementation** Our deliverable is an implementation of a synthetic data generation toolbox, which consists of eight different spatial interpolation algorithms and techniques that allow users to directly adjust the spatial correlation and data distribution in the synthetic data. This toolbox automatically generates irregular topology data from the given experimental data. In addition, we also provide suites of experimental data and irregular topology data created using the aforementioned synthetic data generation techniques. These ready-to-use test suites will encourage people to utilize more realistic data input when evaluating their algorithms.

EmStar [GEC04] is an integrated simulation and runtime environment for developing embedded sensor network systems. EmStar provides a rich library and tool support to facilitate the development of mobile embedded systems. EmStar has been used in multiple institutions and corporations in their sensor network research. In order to incorporate our work into EmStar, we have added sensor data replay support in EmStar and are in the process of adding a sensor noise model to EmSim. This addition supports EmStar simulation / emulation with more realistic sensor data traces.

## 9.2 Future Directions

In this section, we describe possible directions for future work.

**Development and evaluation of the synthetic data against multiple metrics** Synthetic data is currently evaluated based on a single metric; however, the application may be affected by multiple data characteristics (e.g., both the spatial correlation and frequency spectrum). A possible extension to our framework is to develop synthetic data generation algorithms to approximate the

original data based on multiple metrics. As a related problem, we also plan to investigate how to evaluate synthetic data against multiple metrics.

**Fine grained interpolation and extrapolation** Compared to the envisioned scenarios for sensor networks, most currently available experimental data sets are collected at a much lower density. We did not directly address the *fine grained interpolation* issue in this thesis, since we lack ground truth data to evaluate whether the fine grained interpolation results capture the characteristics of the real phenomena. We conjecture that if the phenomena is smooth, we can achieve reasonable approximation by applying the interpolation and space-time modeling techniques proposed in this thesis. On the other hand, if we have domain knowledge of the spatial correlation or other data characteristics at small scales, we can directly plug that into our empirical model. With high density data available from newly deployed sensor networks (e.g., NIMS), we plan to investigate how closely the previously proposed spatial interpolation techniques capture the real phenomena at small scales.

**Combination of irregular topology and non-uniform data distributions** In this thesis, we have studied how irregular topology or non-uniform data distributions affect the algorithm performance. The combination of the irregularity in both topology and data distribution may aggravate the problems caused by each of these scenarios individually. In our future work we plan to study how the interaction between irregular topology and non-uniform data distribution affect the algorithm performance.

**Continual development on toolset and test data suites** We will continue toolset development to facilitate EmStar emulation with more realistic sensor

data input and sensor channel noise models. We also plan to provide more test data traces when more experimental data becomes available.

# References


[AA00] Henrik Abrahamsson and Bengt Ahlgren. "Using empirical distributions to characterize web client traffic and to generate synthetic traffic." In *IEEE Globecom: Global Internet*, San Francisco, USA, Nov. 2000.

[al01] Xin Li et al. "New Edge-Directed Interpolation." In *IEEE Trans. on Image Processing*, Oct. 2001.

[BBE99] Sandeep Bajaj, Lee Breslau, Deborah Estrin, Kevin Fall, Sally Floyd, Padma Haldar, Mark Handley, Ahmed Helmy, John Heidemann, Polly Huang, Satish Kumar, Steven McCanne, Reza Rejaie, Puneet Sharma, Kannan Varadhan, Ya Xu, Haobo Yu, and Daniel Zappala. "Improving Simulation for Network Research." Technical Report 99-702b, University of Southern California, March 1999. revised September 1999, to appear in IEEE Computer.

[BJ76] G. Box and G. Jenkins. "Time series analysis: forecasting and control." Holden-Day, 1976.

[BKV04] B. Beferull-Lozano, Robert Konsbruck, and Martin Vetterli. "Rate-Distortion Problem for physics based distributed sensing." In *IPSN*, 2004.

[BMJ98] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva. "A Performance Comparison of Multi-Hop Wireless Ad-Hoc Network Routing Protocols." In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom'98)*, Dallas, TX, 1998.

[BRY] M. A. Batalin, M. Rahimi, Y.Yu, D.Liu, A.Kansal, G.S. Sukhatme, W.J. Kaiser, M.Hansen, G. J. Pottie, M. Srivastava, and D. Estrin. "Towards Event-Aware Adaptive Sampling Using Static and Mobile Nodes." Technical Report 38, UCLA/CENS.

[Car76] Manfredo Do Carmo. "Differential geometry of curves and surfaces." pp. 141–146. Prentice-Hall, Inc., 1976.

[CDJ91] Ramon Caceres, Peter B. Danzig, Sugih Jamin, and Danny J. Mitzel. "Characteristics of Wide-Area TCP/IP Conversations." In *SIGCOMM*. ACM, 1991.

[CWK04] Alberto Cerpa, Jennifer L. Wong, Louane Kuang, Miodrag Potkonjak, and Deborah Estrin. "Statistical Model of Lossy Links in Wireless Sensor Networks." Technical Report 41, CENS, April 2004.

[DNB04] Ronique Delouille, Ramesh Neelamani, and Richard Baraniuk. "Robust distributed estimation in sensor networks using the embedded polygons algorithm." In *Proceedings of the third international symposium on Information processing in sensor networks*, pp. 405–413. ACM Press, 2004.

[Dor96] Georg Dorffner. "Neural Networks for Time Series Processing." *Neural Network World*, **6**(4):447–468, 1996.

[DPR00] Samir R. Das, Charles E. Perkins, and Elizabeth M. Royer. "Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks." In *INFOCOM*, Israel, March 2000.

[FBS02] P. Fryzlewicz, S. Van Bellegem, and R. von Sachs. "A wavelet-based model for forecasting non-stationary processes." In *Submitted for publication.*, 2002.

[FK02] Sally Floyd and Eddie Kohler. "Internet Research Needs Better Models." In *1st Workshop on Hot Topics in Networks (HotNets-I)*, Oct 2002.

[GCB04] Deepak Ganesan, Rzvan Cristescu, and Baltasar Beferull-Lozano. "Power-efficient sensor placement and transmission structure for data gathering under distortion constraints." In *Proceedings of the third international symposium on Information processing in sensor networks*, pp. 142–150. ACM Press, 2004.

[GEC04] L. Girod, J. Elson, A. Cerpa, T. Stathopoulos, N. Ramanathan, and D. Estrin. "EmStar: a Software Environment for Developing and Deploying Wireless Sensor Networks." In *USENIX*, 2004.

[GEH02] Deepak Ganesan, Deborah Estrin, and John Heidemann. "DIMENSIONS: Why do we need a new Data Handling architecture for Sensor Networks?" In *Proceedings of the First Workshop on Hot Topics In Networks (HotNets-I)*, Oct 2002.

[GK04] M. B. Greenwald and S. Khanna. "Power-Conserving Computation of Order-Statistics over Sensor Networks." In *23rd ACM Symposium on Principles of Database Systems (PODS)*, 2004.

[Goo97] Pierre Goovaerts. "Geostatistics for Natural Resources Evaluation." Oxford University Press, Inc., 1997.

[Gra97] A. Gray. "The Gaussian and Mean Curvatures." In *Modern Differential Geometry of Curves and Surfaces with Mathematica*, pp. 373–380. CRC Press, 1997.

[IS89] Edward Isaaks and R. Srivastava. In *An introduction to Applied Geostatistics*. Oxford University Press, 1989.

[JLH99] Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael Degermark. "Scenario-based performance analysis of routing protocols for mobile ad-hoc networks." In *Proc. ACM Mobicom*, Seattle, Washington, 1999.

[JP04] Apoorva Jindal and Konstantinos Psounis. "Modeling spatially-correlated sensor network data." In *SECON*, 2004.

[KIR04] Animesh Kumar, Prakash Ishwar, and Kannan Ramchandran. "On Distributed Sampling of Smooth Non-Bandlimited Fields." In *IPSN*, 2004.

[KJ99] P.C. Kyriakidis and A.G. Journel. "Geostatistical Space-Time Models." In *Mathematical Geology*, volume 31, 1999.

[KMK02] P.C. Kyriakidis, N. L. Miller, and J. Kim. "A Spatial Time Series Framework for Modeling Daily Precipitation at Regional Scales." In *82nd Annual Meeting of the American Meteorological Society*, January 2002.

[Kri66] D. G. Krige. "Two-dimensional weighted moving average trend surfaces for ore-valuation." In *Journal of the South Africa Institute of Mining and Metallurgy*, volume 66, 1966.

[KZJ01] Almudena Konrad, Ben Y. Zhao, Anthony D. Joseph, and Reiner Ludwig. "A Markov-Based Channel Model Algorithm for Wireless Networks." In *Proceedings of Fourth ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, ACM MSWiM*, July 2001.

[LOC] "LOCFIT: Local Regression and Likelihood." http://cm.bell-labs.com/cm/ms/departments/sia/project/locfit/.

[MN04] Daniel Marco and David Neuhof. "Reliability vs. Efficiency in Distributed Source Coding for Field-Gathering." In *IPSN*, 2004.

[NG] Suman Nath and Phillip Gibbons. "Synopsis Diffusion for Robust Aggregation in Sensor Networks." Technical Report IRP-TR-03-08, Intel Research.

[NGA04] Suman Nath, Phillip Gibbons, Zachary Anderson, and Srinivasan Seshan. "Synopsis Diffusion for Robust Aggregation in Sensor Networks." In *Sensys*, Nov 2004.

[Oga98] Yosihiko Ogata. "Space-time point-process models for earthquake occurences." In *Ann. Inst. Statist. Math.*, volume 50, 1998.

[Peb03] Edzer J. Pebesma. "Gstat: multivariable geostatistics for S." In *DSC*, 2003.

[PSS00] S. Park, A. Savvides, and M. B. Srivastava. "SensorSim: A Simulation Framework for Sensor Networks." In *MSWiM*. ACM, August 2000.

[PSS01] S. Park, A. Savvides, and M. B. Srivastava. "Simulating Networks of Wireless Sensors." In *the 2001 Winter Simulation Conference*. ACM, 2001.

[R u] "The R Project for Statistical Computing." In *http://www.R-project.org/.*

[RN04] Michael Rabbat and Robert Nowak. "Distributed optimization in sensor networks." In *Proceedings of the third international symposium on Information processing in sensor networks*, pp. 20–27. ACM Press, 2004.

[RPE04] Mohammad Rahimi, Richard Pon, Deborah Estrin, William J. Kaiser, Mani Srivastava, and Gaurav S. Sukhatme. "Adaptive Sampling for Environmental Robotics." In *IEEE International Conference on Robotics and Automation*, 2004.

[SBG] F. P. Schoenberg, D. R. Brillinger, and P. M. Guttorp. "Point processes, spatial-temporal." In *Encyclopedia of Environmetrics*, volume 3. Wiley, NY.

[Sch] F. P. Schoenberg. "Consistent parametric estimation of the intensity of a spatial-temporal point process." In *Ann. Inst. Stat. Math. (in review)*. Wiley, NY.

[Ser03] S. D. Servetto. "Distributed Signal Processing Algorithms for the Sensor Broadcast Problem." In *Proceedings of the 37th Annual Conference on Information Sciences and Systems, (CISS)*, March 2003.

[Sta99] The R Foundation for Statistical Computing. "The R Project for Statistical Computing." http://cran.stat.ucla.edu/, 1999.

[TC91] Joy A. Thomas and Thomas M. Cover. "Elements of Information Theory." John Wiley & Sons, Inc., 1991.

[TN00] Y. Theodoridis and Mario Nascimento. "Generating spatiotemporal data sets in the WWW." In *SIGMOD record 29 (3)*, 2000.

[WC] M. Widmann and C.Bretherton. 50 km resolution daily preciptation for the Pacific Northwest, 1949-94, http://tao.atmos.washington.edu/data_sets/widmann/.

[WCT99] DuMouchel W, Volinsky C, Johnson T, Cortes C, and Pregibon D. "Squashing flat files flatter." In *Proc. KDD*, 1999.

[Web] R. Webster. "Geostatistics for environmental scientists.".

[WMN04] Rebecca Willett, Aline Martin, and Robert Nowak. "Backcasting: An Adaptive Approach to Energy Conservation in Sensor Networks." In *IPSN*, 2004.

[WO01] Richard Webster and Margaret A. Oliver. "Geostatistics for Environmental Scientists." John Wiley & Sons, Inc., 2001.

[WTC03] Alec Woo, Terence Tong, and David Culle. "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks." In *Sensys*, 2003.

[WYP04] Hanbiao Wang, Kung Yao, Greg Pottie, and Deborah Estrin. "Entropy-based Sensor Selection Heuristic for Localization." In *IPSN*, April 2004.

[YGG03] Yan Yu, Deepak Ganesan, Lewis Girod, Deborah Estrin, and Ramesh Govindan. "Synthetic data generation to support irregular sampling in Sensor Networks." In *Geo Sensor Networks*. Taylor and Francis Publishers, Oct 2003.

[ZBG98] Xiang Zeng, Rajive Bagrodia, and Mario Gerla. "GloMoSim: a Library for Parallel Simulation of Large-scale Wireless Networks." In *Proceedings of the 12th Workshop on Parallel and Distributed Simulations – PADS '98*, May 1998.

[ZG03] Jerry Zhao and Ramesh Govinda. "Understanding Packet Delivery Performance In Dense Wireless Sensor Networks." In *Sensys*, 2003.

[ZSR02] F. Zhao, J. Shin, and J. Reich. "Information-Driven Dynamic Sensor Collaboration for Tracking Applications." In *IEEE Signal Processing Magazine*, March 2002.