ABSTRACT

A BAYESIAN APPROACH TO WIRELESS LOCATION PROBLEMS

by
Ivan Zorych

Several approaches for indoor location estimation in wireless networks are proposed. We explore non-hierarchical and hierarchical Bayesian graphical models that use prior knowledge about physics of signal propagation, as well as different modifications of Bayesian bivariate spline models. The hierarchical Bayesian model that incorporates information about locations of access points achieves accuracy that is similar to other published models and algorithms, but by using prior knowledge, this model drastically reduces the requirement for training data when compared to existing approaches. Proposed Bayesian bivariate spline models for location surpass predictive accuracy of existing methods. It has been shown that different versions of this model, in combination with sampling/importance resampling and particle filter algorithms, are suitable for the real-time estimation and tracking of moving objects. It has been demonstrated that "plug-in" versions of the bivariate Bayesian spline model perform as good as the full Bayesian version. A combination of two Bayesian models to reduce the maximum predictive error is proposed. Models presented in this work utilize MCMC simulations in directed acyclic graphs (DAGs) to solve ill-posed problem of location estimation in wireless networks using only received signal strengths. Similar approaches may be applied to other ill-posed problems.

A BAYESIAN APPROACH TO WIRELESS LOCATION PROBLEMS

by
Ivan Zorych

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology and
Rutgers, The State University of New Jersey – Newark
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Mathematical Sciences

Department of Mathematical Sciences
Department of Mathematics and Computer Science, Rutgers-Newark

August 2005

APPROVAL PAGE

A BAYESIAN APPROACH TO WIRELESS LOCATION PROBLEMS

Ivan Zorych

Manish Bhattacharjee, Ph.D., Dissertation Co-Advisor — Date
Professor of Mathematical Sciences, NJIT

David Madigan, Ph.D., Dissertation Co-Advisor — Date
Professor of Statistics, Rutgers University

Sunil Dhar, Ph.D., Committee Member — Date
Associate Professor of Mathematical Sciences, NJIT

Zoi-Heleni Michalopoulou, Ph.D., Committee Member — Date
Associate Professor of Mathematical Sciences and of Electrical and Computer Engineering, NJIT

Thomas Spencer, Ph.D., Committee Member — Date
Research Professor of Mathematical Sciences, NJIT

BIOGRAPHICAL SKETCH

Author: Ivan Zorych

Degree: Doctor of Philosophy

Date: August 2005

Undergraduate and Graduate Education:

- Doctor of Philosophy in Mathematical Sciences,
  New Jersey Institute of Technology, Newark, NJ, 2005
- Master of Science in Applied Statistics,
  New Jersey Institute of Technology, Newark, NJ, 2002
- Specialist in Mathematics,
  Kyiv State University, Kyiv, Ukraine, 1994

Major: Mathematical Sciences (Applied Probability & Statistics)

Presentations and Publications:

D. Madigan, I. Zorych,
"Bayesian Models for Location Estimation in Wireless Networks,"
Frontiers in Applied and Computational Mathematics, Newark, NJ, 2005.

D. Madigan, I. Zorych,
"A Bayesian Modeling Approach to Location Estimation,"
Frontiers in Applied and Computational Mathematics, Newark, NJ, 2004.

I. Zorich,
"On Informativeness of Experiments in a Bayesian Decision-Making Problem,"
Cybernetics and Systems Analysis, vol. 37, no. 3, pp. 447-452, 2001.

V. Ivanenko, B. Munier, I. Zorich,
"On Construction of a Statistical Regularity of Stochastically Unstable Sequence,"
Journal of Automation and Information Sciences, vol. 32, no. 7, pp. 84-87, 2000.

To my parents

ACKNOWLEDGMENT

I would like to express my sincere gratitude to Manish Bhattacharjee and David Madigan, for their valuable guidance. Without their help, this work would not be possible. I also thank all members of my dissertation committee. Their advice and patience are highly appreciated.

Special thanks goes to Thomas Spencer who greatly enriched my knowledge with his exceptional insights into different areas of modern statistics.

I am very grateful to Roman Voronka for his support, encouragement, and advice.

I would also like to thank the Department of Mathematical Sciences at NJIT for providing financial support for this research.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF TABLES

LIST OF TABLES
(Continued)

Table **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

CHAPTER 1

INTRODUCTION

1.1 Motivation

The growth of wireless networking has generated commercial and research interest in statistical methods to track people and things. Inside stores, hospitals, warehouses, and factories, where Global Positioning System devices generally do not work, Indoor Positioning Systems (IPS) aim to provide location estimates for wireless devices such as laptop computers, handheld devices, and electronic badges. The proliferation of "Wi-Fi" (IEEE 802.11b) wireless internet access in cafes, college campuses, airports, hotels, and homes has generated particular interest in IPS that utilize physical attributes of Wi-Fi signals. Typical applications include tracking equipment and personnel in hospitals, providing location-specific information in supermarkets, museums, and libraries, and location-based access control.

In a standard Wi-Fi setup, one or more access points serve end-users. In what follows we focus on networks with multiple access points (typical of networks in office buildings or large public spaces). Wi-Fi location estimation can employ one or more of several physical attributes of the medium. Typical features include received signal strength (RSS) from the access points, the angle of arrival of the signal, and the time difference of arrival. Among these, RSS is the only feature that reasonably priced hardware can currently measure. There exists a substantial literature on using RSS for location estimation in wireless networks [1, 15, 32]. Related websites include `www.ekahau.com`, `www.bluesoft-inc.com`, and `www.newburynetworks.com`. In a laboratory setting, RSS decays linearly with log distance and a simple triangulation using RSS from three access points could uniquely identify a location in a two-dimensional space.

In practice, physical characteristics of a building such as walls, elevators, and furniture, as well as human activity add significant noise to RSS measurements. Consequently statistical approaches to location estimation prevail.

The standard approach uses supervised learning techniques. The training data comprise vectors of signal strengths, one for each of a collection of known locations. The dimension of each vector equals the number of access points. The corresponding location could be one-dimensional (e.g., location on a long airport corridor), two-dimensional (e.g., location on one floor of a museum), or three-dimensional (e.g. location within a multi-storey office building). Collection of the location data is labor intensive requiring physical distance measurements with respect to a reference object such as a wall. The model building phase then learns a predictive model that maps signal strength vectors to locations. Researchers have applied many supervised learning methods to this problem including nearest neighbor methods, support vector machines, and assorted probabilistic techniques. In this work we explore the use of hierarchical Bayesian graphical models [9, 20, 41] for wireless location. Our objective is to use the hierarchical Bayesian framework to incorporate important prior information and the graphical model framework to facilitate the construction of realistically complex models.

Gathering extensive training data and the requisite physical measures of location ("profiling") involves a steep upfront cost and deployment effort [37]. Furthermore, even in normal office environments, changing environmental, building, and occupancy conditions can affect signal propagation and require repeated data gathering to maintain predictive accuracy [1]. Consequently, minimizing the number of training observations needed to adequately profile a particular site is an important objective. Similarly we seek to minimize data requirements concerning internal wall materials, flooring, occupancy, etc.

Two types of location estimation systems exist. In a client-based deployment, the client measures the signal strengths as seen by it from various access points. The client uses this information to locate itself. The cost to an enterprise for such deployments is the cost of profiling the site, building the model, and maintaining the model. In an infrastructure-based deployment, the administrator deploys so-called sniffing devices that monitor the signal strength from clients. The cost to enterprises in such deployments is the typically modest cost of deploying the necessary hardware and software, and the time and effort to build and maintain the model (if it is not completely automated).

Our key finding is that a hierarchical Bayesian approach, incorporating prior physical knowledge about the nature of Wi-Fi signals, can provide accurate location estimates without *any* location information in the training data. In the context of an infrastructure-based deployment, our proposed model can thus eliminate profiling entirely.

Major part of this work focuses on static location estimation. We investigate different models that estimate location at a particular time point.

Chapter 5 focuses on online location methods and on tracking of moving objects. It utilizes results from chapter 4 in combination with particle algorithms.

1.2 Background

1.2.1 Related Work

Location estimation techniques in wireless networks can be broadly classified based on the methods used to build models and methods used to search the models in the online phase. For building models, most techniques profile the entire site and collect one or more signal strength samples from all visible access points at each sample point. Each point is mapped to either a signal strength vector [1, 15, 27, 36] or a signal strength probability distribution [2, 32, 42, 46]. Such profiling techniques

require considerable investment in data gathering. Alternatively, a parametric model that uses signal propagation physics and calculates signal degradation based on a detailed map of the building, the walls, obstructions and their construction material, has been proposed [1]. Obtaining detailed maps of the building and its changes over time is, however, a hurdle that needs to be overcome for the use of this method.

In [37], the authors emphasize a client-based location model and raise interesting privacy issues in location-based services. We expect that in enterprises, based on current privacy policies used for other electronic transmissions like email and web-access, the preference would be for an infrastructure-based solution. If privacy is desired, in our case, on entering a site a client device could download the model for that site and use it to determine its own location. As mentioned in [37], client-based approaches must also be concerned about the power requirements on the client devices that are inherently power constrained. Sniffing for clients to provide an infrastructure-based system has been proposed [5, 44, 45].

Custom sensors have been used for location estimation in other interesting ways [29, 44, 45]. In [44] and similar systems, infra-red (IR) wireless technology is used; IR technology has limited range and hence has not become very popular. In [29], a decentralized (client-based) approach using time difference of arrival between ultrasound and RF signals from custom sensors is used for location estimation. The system in [45] uses expensive custom RF-based hardware for location estimation, and an approach based on time difference of signal arrival, which is inherently more expensive to measure. In contrast, our approach is easier to bootstrap, is based on RSS and can be built with off-the-shelf components. Recent advances in sensor technology [26] and projected decreases in the manufacturing cost allow us to provide a cost-effective solution in our system.

1.2.2 Radio Frequency Signal Propagation in Wireless Ethernet

In [15], Ladd and coworkers provide an introduction to the behavior of Wi-Fi signals and here we present a brief summary. The IEEE 802.11b High-Rate standard uses radio frequencies in the 2.4 GHz band. Wi-Fi adaptors use spread-spectrum technology that spreads the signal over several frequencies. In this way, interference on a single frequency does not entirely block the signal. The signal itself propagates in a complex manner [22]. Reflection, absorption, and diffraction occur when the signal's waves encounter opaque obstacles resulting in essentially random variations of signal strength. A variety of other factors such as noise, interference from other sources, and interference between channels also affect the signal. The resonant frequency of water happens to be 2.4 GHz so people also absorb the radio waves and impact the signal strength. Other common devices using the 2.4 GHz band include microwave ovens, BlueTooth devices, and 2.4 GHz cordless phones.

The consequence of all this is that received signal strength varies over time at a single location and varies across different locations. However, signal profiles corresponding to spatially adjacent locations are similar as the various external variables remain approximately the same over short distances. Furthermore, the local average of the signal strength varies slowly over time and the signal strength decays approximately in proportion to log distance [25].

Several units of measurement are used to represent signal strength in 802.11, we will use here only dBm ("dee-bee-em" or "dee-bee milliwats"). The "dBm" is a logarithmic measurement connected to mW (milliwats) as follows:

$$dBm = 10 * log(mW). \tag{1.1}$$

For convenience, we will multiply all signal strength measurements by -1, i.e. instead of "-50 dBm", we use "50" value of RSS. We will denote it as "50 -dBm."

CHAPTER 2

INTRODUCTION TO BAYESIAN AND GRAPHICAL METHODS

2.1 The Bayesian Approach

There are two major philosophies in modern Statistics. The classical frequentist approach assumes that probability of events refer to limiting frequencies. Under frequentist paradigm parameters are assumed to be fixed unknown constants, therefore no probability statements can be made about them.

In the Bayesian approach, on the other hand, probability of the event characterizes degree of belief, and there is no fundamental difference between parameters and observations, both are treated as random variables. Statistical information about those variables is summarized in the form of distributions, and the aim of the Bayesian analysis is to get posterior distributions of the parameters of interest.

Bayesian inference is carried through the following steps:

1. Choose the *prior* distribution(s) $f(\theta)$ of the parameter(s) of interest θ. Prior distribution reflects our prior knowledge of the parameter(s).

2. Set up a *full probabilistic model,* a joint probability distribution of all quantities (observable, i.e. data, and unobservable, i.e. parameters): $f(d, \theta)$. Joint density may be expressed as a product of the prior distribution $f(\theta)$ and the *likelihood* $f(d\,|\theta)$:

$$f(d, \theta) = f(d\,|\theta) \times f(\theta) \tag{2.1}$$

3. Having observed data D, we update our belief about θ using the *Bayes rule*:

$$f(\theta\,|D) = \frac{f(d\,|\theta) \times f(\theta)}{f(D)} \tag{2.2}$$

Very often the calculation of the normalized value $f(D)$ is infeasible, that leaves us with the *unnormalized* posterior density:

$$f(\theta\,|D) \propto f(d\,|\theta) \times f(\theta) \tag{2.3}$$

2.2 The Gibbs Sampler

Let's assume that our aim is to draw a sample from the joint density $\pi(x)$, $x \in \mathbf{R}^n$. Let $\pi(x_i\,|x_{-i})$ denote the conditional distribution of x_i given all other variables:

$$\pi(x_i\,|x_{-i}) = \pi(x_i \mid x_1, x_2, ..., x_{i-1}, x_{i+1}, ..., x_n). \tag{2.4}$$

According to [10], Gibbs sampler algorithm is carried on in the following way:

step 0. Choose a starting point $x^0 = (x_1^0, x_2^0, ..., x_n^0)$

step 1. Draw x_1^1 from $\pi(x_1 \mid x_{-1}^0)$

step 2. Draw x_2^1 from $\pi(x_2 \mid x_1^1, x_3^0, ..., x_n^0)$

step 3. Draw x_3^1 from $\pi(x_3 \mid x_1^1, x_2^1, x_4^0..., x_n^0)$

...

step k. Draw x_k^1 from $\pi(x_k \mid x_1^1, x_2^1, ..., x_{k-1}^1, x_{k+1}^0..., x_n^0)$

...

step n. Draw x_n^1 from $\pi(x_n \mid x_1^1, x_2^1, ..., x_{n-1}^1)$

These n steps show the transition from $x^0 = (x_1^0, x_2^0, ..., x_n^0)$ to $x^1 = (x_1^1, x_2^1, ..., x_n^1)$. Repeating those steps M times will generate a sequence $x^0, x^1, ..., x^M$. And it can be shown that if $M \to \infty$, $\pi(x)$ is a stationary distribution of the Markov chain $\{x^M\}$.

2.3 Graphical Models and Conditional Independence

A *graphical model* is a multivariate statistical model embodying a set of conditional independence relationships. A graph displays the independence relationships.

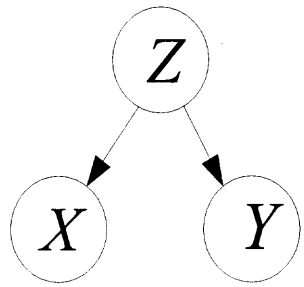


Figure 2.1 This directed acyclic graph (DAG) represents connditional independence of X and Y given Z

Definition. *Let X, Y, and Z be random variables. X and Y are conditionally independent given variable Z, $X \perp Y \mid Z$, if*

$$f_{X,Y|Z}(x, y \mid z) = f_{X|Z}(x \mid z) f_{Y|Z}(y \mid z). \tag{2.5}$$

The vertices of the graph correspond to random variables and the edges encode the relationships. To date, most graphical models research has focused on acyclic digraphs, chordal undirected graphs, and chain graphs that allow both directed and undirected edges, but have no partially directed cycles [17].

Here we focus on acyclic digraphs (ADGs) with both continuous and categorical random variables. In an ADG, *all* the edges are directed and the graph represents them with arrows (see Figure 2.1).

A directed graph is acyclic if it contains no cycles. Each vertex in the graph corresponds to a random variable $X_v, v \in V$ taking values in a sample space $\mathcal{X}_v$. To simplify notation, we use v in place of X_v in what follows. In an ADG, the *parents* of a vertex v, pa(v), are those vertices from which vertices point into v. The *descendants* of a vertex v are the vertices which are reachable from v along a directed path. A vertex w is a *child* of v if there is an edge from v to w. The parents of v are taken to the be the only direct influences on v, so that v is independent of its non-descendants given its parents. This property implies a factorization of the joint

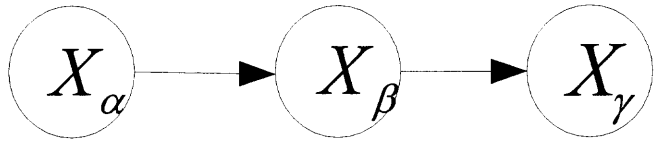


Figure 2.2 A simple acyclic directed graphical model with joint density $p(X_\alpha, X_\beta, X_\gamma) = p(X_\gamma|X_\beta)p(X_\beta|X_\alpha)p(X_\alpha)$

density of $X_v, v \in V$, which we denote by $p(V)$, given by

$$p(V) = \prod_{v \in V} p(v|\text{pa}(v)). \tag{2.6}$$

Figure 2.2 shows a simple example. This directed graph represent the assumption that X_γ and X_α are conditionally independent given X_β. The joint density of the three variables factors accordingly,

$$p(X_\alpha, X_\beta, X_\gamma) = p(X_\gamma|X_\beta)p(X_\beta|X_\alpha)p(X_\alpha).$$

For graphical models where all the variables are discrete, Spiegelhalter and Lauritzen presented a Bayesian analysis and showed how independent Dirichlet prior distributions can be updated locally to form posterior distributions as data arrive [39]. Heckerman and coworkers provided corresponding closed-form expressions for complete-data likelihoods and posterior model probabilities [24]. Madigan and York described corresponding Bayesian model averaging procedures [20]. In the Bayesian framework, model parameters are random variables and appear as vertices in the graph.

When some variables are discrete and others continuous, or when some of the variables are latent or have missing values, a closed-form Bayesian analysis generally does not exist. Analysis then requires either analytic approximations of some kind or simulation methods. Here we consider a Markov chain Monte Carlo (MCMC) simulation method.

As follows from the previous section, the Gibbs sampler starts with some initial values for each unknown quantity (that is, model parameters, missing values, and latent variables), and then cycles through the graph simulating each variable v in turn from its conditional probability distribution, given all the other quantities, denoted $V \backslash v$, fixed at their current values (known as the "full conditional"). The simulated v replaces the old value and the simulation shifts to the next quantity. After sufficient iterations of the procedure one assumes that the Markov chain has reached its stationary distribution, and then future simulated values for vertices of interest are monitored. Inferences concerning unknown quantities are then based on data analytic summaries of these monitored values, such as empirical medians and 95% intervals. Some delicate issues do arise with the Gibbs sampler such as assessment of convergence, sampling routines, etc. Gilks and coworkers provide a full discussion [11].

The crucial connection between directed graphical models and Gibbs sampling lies in expression (2.6). The full conditional distribution for any vertex v is equal to:

$$\begin{aligned} p(v|V\backslash v) &\propto p(v, V\backslash v) \\ &\propto \text{terms in } p(V) \text{ containing } v \\ &= p(v|\text{pa}(v)) \prod_{w \in \text{child}(v)} p(w|\text{pa}(w)), \end{aligned}$$

i.e., a prior term and a set of likelihood terms, one for each child of v. Thus, when sampling from the full conditional for v, we need only consider vertices which are parents, children, or parents of children of v, and we can perform local computations. The *BUGS* language and software (`www.mrc-bsu.cam.ac.uk/bugs`) implements a version of the Gibbs sampler for Bayesian graphical models [40]. As follows from the formula for the full conditional distribution for vertex v, a DAG model usually gives us the full conditional distribution of each node up to proportionality constant. Fortunately, methods of sampling from full conditional distributions used in *BUGS*

(*WinBUGS*) do not require evaluation of proportionality constants. We used *WinBUGS* for the experiments we report below.

2.4 Semiparametric Regression Models

There are two types of regression models: parametric and nonparametric. Parametric models assume a known form of functional relationships between a dependent variable and explanatory variables. And fitting a parametric model usually means evaluating of unknown coefficients as we do in the case of linear regression. Many phenomena provide data that has a complicated structure and an intrinsically nonlinear character. If one is interested in applying regression like techniques in such situation, the natural choice would be an approach which is known as nonparametric regression. The feature that differentiates a nonparametric approach from a parametric one is the absence of knowledge about exact form of functional dependence. Although such dependence is still described by a mathematical equation with many unknown parameters, the data, not our model choice, defines eventual form of the function. This flexibility of nonparametric model often helps to capture features of the data that would be very difficult to describe using other tools.

As pointed out by Ruppert [35], there are several approaches to nonparametric modeling: regression splines, smoothing splines, series-based smoothers (such as wavelets), kernel methods, etc. We will use penalized splines, also known as P-splines or low-rank splines. From methodological point of view penalized splines are close to ridge regression and mixed models. Ruppert [35] states that "Markov chain Monte Carlo provide...the most satisfactory approach to fitting complex semiparametric models as well as the direction that semiparametric regression is most likely to take in the future."

Let's consider a set of points (x_i, y_i), $i = 1, 2, ..., N$, without assuming any hidden probabilistic models. According to [4], an ordinary nonparametric model may

be written as

$$y_i = f(x_i) + \varepsilon_i \tag{2.7}$$

where $f(x)$ is some function of interest, and ε_i are independent $N(0, \sigma^2)$ variables.

Let's define x_+ as $\max(x, 0)$, in this case for some number λ,

$$(x - \lambda)_+ = \max(x - \lambda, 0) = \begin{cases} x - \lambda, \ if \ x \geq \lambda; \\ 0, \ otherwise. \end{cases} \tag{2.8}$$

If $\lambda_1, \lambda_2, ..., \lambda_K$ is a set of different numbers from the range of x, then any linear combination of $1, x, (x - \lambda_1)_+, ..., (x - \lambda_2)_+, ..., (x - \lambda_K)_+$ is called a linear *spline,* numbers $\lambda_1, \lambda_2, ..., \lambda_K$ are called *spline knots.* The spline model for $f(x)$ from (2.7) is

$$f(x) = a[1] + a[2] * x + \sum_{k=1}^{K} b[k] * (x - \lambda_k)_+ \tag{2.9}$$

One may try to use a standard least square approach for fitting model (2.7)-(2.9). But as number of knots increases, "overfitting" becomes a real problem. "Overfitting" means excessive flexibility when the fitted function, instead of only capturing main features of the data, catches its random changes. Shrinkage of the spline coefficients, that impose penalty on their size, may help to avoid overfitting [23]. There are different forms of constraints that may be used here, but from technical point of view, if one is interested in using a least square approach, the most suitable form is the spherical restrictions:

$$\sum_{k=1}^{K} b^2[k] < C^2, \tag{2.10}$$

where C is some constant. Following [35] we can define vector of all unknown coefficient B as

$$B = \begin{bmatrix} a[1] \\ a[2] \\ b[1] \\ \dots \\ b[K] \end{bmatrix}, \tag{2.11}$$

and $(K+2) \times (K+2)$ matrix D as

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 \\ 0 & 0 & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0_{2\times 2} & 0_{2\times K} \\ 0_{K\times 2} & I_{K\times K} \end{bmatrix}, \tag{2.12}$$

and the $N \times (K+2)$ matrix X as

$$X = \begin{bmatrix} 1 & x_1 & (x_1-\lambda_1)_+ & (x_1-\lambda_2)_+ & \dots & (x_1-\lambda_K)_+ \\ 1 & x_2 & (x_2-\lambda_1)_+ & (x_2-\lambda_2)_+ & \dots & (x_2-\lambda_K)_+ \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & x_N & (x_N-\lambda_1)_+ & (x_N-\lambda_2)_+ & \dots & (x_N-\lambda_K)_+ \end{bmatrix}. \tag{2.13}$$

It can be easily proved that minimization of $\sum_{i=1}^{N}(y_i - f(x_i))^2$ by B, subject to restriction (2.10) is equivalent to minimization of

$$\|y - XB\|^2 + \alpha B^T D B \tag{2.14}$$

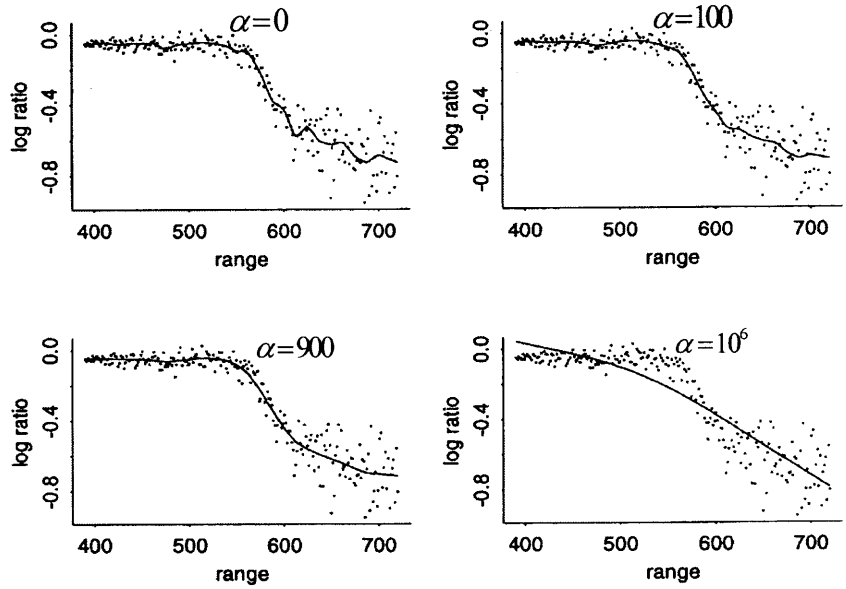


Figure 2.3 Linear penalized spline regression fits to the LIDAR data (see [35]), 24 knots are used.

as a function of B, for some fixed positive value of α; $\|\cdot\|$ is an Euclidian norm, and $'T'$ represents transpose of a matrix. The value of the smoothing parameter α in (2.14) is determined by the value of the constant C in (2.10). Figure 2.3 from page 67 of [35] shows the influence of a smoothing parameter on the spline fit. When $\alpha = 0$, model overfits the data being too sensitive to random changes in the dataset. Model fits data well when $\alpha = 900$. When $\alpha = 10^6$, the influence of the spline knots decreases, and we observe the least-squares line.

Several recent works [35, 47] pointed out a close relationship between penalized splines and *mixed models.*

We will use the Linear Mixed Effects model in the form developed by Laird and Ware [16]:

$$y = X\beta + Zu + \varepsilon, \tag{2.15}$$

where y is a $n \times 1$ response vector, X is a $n \times m$ matrix of explanatory variables, β is a $m \times 1$ vector of fixed effect coefficients, Z is a $n \times k$ design matrix of random effects, ε is a $n \times 1$ error vector with independent components, u is a $k \times 1$ vector of random effects, and $E \begin{bmatrix} u \\ \varepsilon \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}$, $cov \begin{bmatrix} u \\ \varepsilon \end{bmatrix} = \begin{bmatrix} \mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{R} \end{bmatrix}$, where $\mathbf{G} = \sigma_u^2 \mathbf{I}$, $\mathbf{R} = \sigma_\varepsilon^2 \mathbf{I}$.

According to Henderson's approach (see *Henderson's Justification,* p. 18 in [31]), β^* and u^*, estimations of β and u in (2.15), may be found as follows:

$$(\beta^*, u^*) = \arg\min_{(\beta,u)}[(y - X\beta - Zu)^T \mathbf{R}^{-1}(y - X\beta - Zu) + u^T \mathbf{G}^{-1} u]. \tag{2.16}$$

The criterion (2.16) may be re-written as

$$\frac{1}{\sigma_\varepsilon^2} \|y - X\beta - Zu\|^2 + \frac{1}{\sigma_u^2} \|u\|^2 . \tag{2.17}$$

If one put $\alpha = \frac{\sigma_\varepsilon^2}{\sigma_u^2}$, then (2.17) takes the following form:

$$\frac{1}{\sigma_\varepsilon^2} \|y - X\beta - Zu\|^2 + \frac{\alpha}{\sigma_\varepsilon^2} \|u\|^2 . \tag{2.18}$$

Penalized spline (2.9)–(2.10) may be represented in the mixed model form. We put

$$X = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ & \cdots \\ 1 & x_N \end{bmatrix}, \tag{2.19}$$

$$\beta = \begin{bmatrix} a[1] \\ a[2] \end{bmatrix}, u = \begin{bmatrix} b[1] \\ b[2] \\ \dots \\ b[K] \end{bmatrix}, \tag{2.20}$$

$$Z = \begin{bmatrix} (x_1 - \lambda_1)_+ & (x_1 - \lambda_2)_+ & \dots & (x_1 - \lambda_K)_+ \\ (x_2 - \lambda_1)_+ & (x_2 - \lambda_2)_+ & \dots & (x_2 - \lambda_K)_+ \\ \dots & \dots & \dots & \dots \\ (x_N - \lambda_1)_+ & (x_N - \lambda_2)_+ & \dots & (x_N - \lambda_K)_+ \end{bmatrix}, \tag{2.21}$$

$$E \begin{bmatrix} u \\ \varepsilon \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}, Cov \begin{bmatrix} u \\ \varepsilon \end{bmatrix} = \begin{bmatrix} \sigma_u^2 \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \sigma_\varepsilon^2 \mathbf{I} \end{bmatrix}. \tag{2.22}$$

The linear mixed model $y = X\beta + Zu + \varepsilon$ represents a penalized spline with the criterion (2.18), where $X\beta$ corresponds to the polynomial part of (2.9), and Zu stands for the spline basis functions part. One can easily see that criteria (2.14) and (2.18) lead to the same estimations of the spline coefficients.

2.5 Bayesian View of the Linear Mixed Effects Model

Mixed model approach to penalized splines allows a Bayesian interpretation. Conditional distributions for the Bayesian version of model (2.15) may be defined in the following way:

$$y \sim \text{Normal}(X\beta + Zu, \sigma_\varepsilon^2 \mathbf{I}), \tag{2.23a}$$

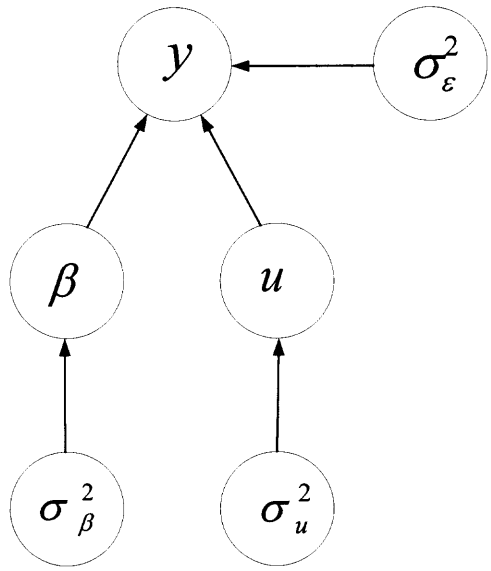


Figure 2.4 Graphical representation of the Bayesian linear mixed model.

$$\beta \sim \text{Normal}(0, \sigma^2_\beta \mathbf{I}), \tag{2.24}$$

$$u \sim \text{Normal}(0, \sigma^2_u \mathbf{I}), \tag{2.25}$$

appropriate prior distributions should be assigned to hyperparameters σ^2_ε, σ^2_β, and σ^2_β.

Figure 2.4 shows a graphical representation of the Bayesian linear mixed model. An example of one-dimensional spline model in *BUGS* is given in [6].

Mixed model approach to penalized splines, and, in particular, its Bayesian implementation, allows us to avoid the difficult problem of choosing the smoothing parameter α in (2.14) and (2.18), or choosing of the constant C in (2.10). The parameter α will be determined automatically as the ratio $\frac{\sigma^2_\varepsilon}{\sigma^2_u}$. Figure 2.5 shows a Bayesian spline fit (red line) with 30 knots for the part of the *BR* data (see section 3.1 of chapter 3), access point 5.

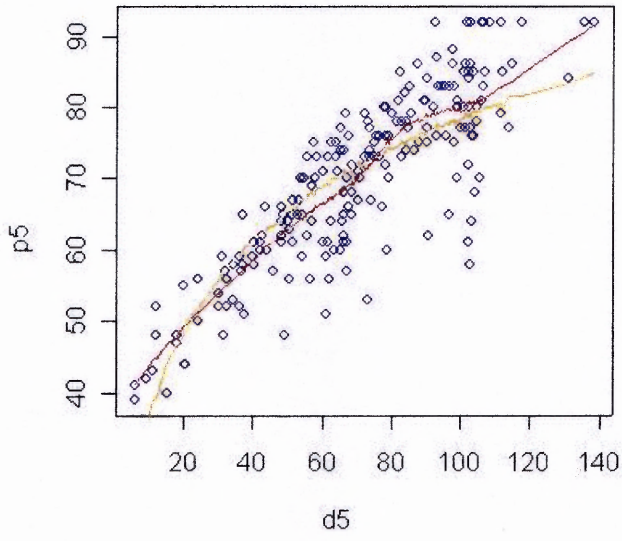


Figure 2.5 Red line is a Bayesian spline fit (signal strength versus distance to the access point) for the 5^{th} access point (BR data). Yellow line shows the hierarchical Bayesian model fit (chapter 3).

2.6 Sampling/Importance Resampling (SIR)

From the first section of this chapter we know that very often a Bayesian model is specified up to the constant (see formula 2.3), i.e. we know a likelihood $L(\theta, x)$ and a prior distribution $p(\theta)$, and therefore a posterior is proportional to the product of the likelihood and the prior: $p(\theta\,|x) \propto L(\theta, x)p(\theta)$. But the calculation of the normalizing constant $\int L(\theta, x)p(\theta)d\theta$ poses a real challenge. An elegant way to deal with such problem is presented in [38].

Suppose one can easily generate sample from a continuous density $g(\theta)$, and our ultimate aim is to generate a sample from a density $h(\theta)$ that is absolutely continuous with respect to $g(\theta)$. Let's assume that $h(\theta)$ is known up to the constant, i.e. we know the functional form of a positive function $f(\theta)$, and $h(\theta) \propto f(\theta)$. If there exists constant $M > 0$, such that $\frac{f(\theta)}{g(\theta)} \leq M$ for any θ, then rejection sampling [30] allows to generate sample from $h(\theta)$:

1. Generate θ from the distribution $g(\theta)$;

2.Generate u from uniform distribution on $[0, 1]$;

3. If $u \leq \frac{f(\theta)}{g(\theta)M}$, accept θ, otherwise discard θ and go to the step 1.

The result this procedure will be a sample from the target distribution $h(\theta)$.

When M is unknown, one can get a sample from the target distribution using the following procedure that is known as *bootstrap resampling* [8] or SIR (*sampling/importance resampling*) algorithm [33]:

1. Take a sample of size n from the distribution $g(\theta) : \theta_1, \theta_2, ..., \theta_n$;

2. Calculate weights

$$w_i = \frac{f(\theta_i)}{g(\theta_i)}, q_i = \frac{w_i}{\sum_{j=1}^{n} w_i}, i = 1 \div n; \tag{2.26}$$

3. Draw a new sample from multinomial distribution on $\{\theta_1, \theta_2, ..., \theta_n\}$ with respective probabilities $(q_1, q_2, ..., q_n)$.

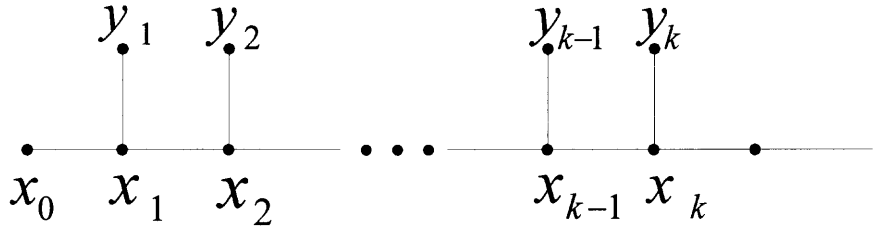


Figure 2.6 A graphical representation of the state-space model

The new sample is approximately distributed as $h(\theta)$, and the approximation gets better as the sample size n increases.

2.7 The Particle Filter

For dynamic tracking we are going to use a Bayesian method that is commonly known as a *particle filter or sequential Monte Carlo method.* The comprehensive reviews of Bayesian methods for dynamic tracking are given in by Liu, Chen, and Doucet [19, 7].

Particle filter uses a state-space model (figure 2.6) that contains two parts:

a) State vector that evolves according to the state equation:

$$x_{k+1} = q_k(x_k, \xi_k), \tag{2.27}$$

where $q_k : \mathbf{R}^n \times \mathbf{R}^m \longrightarrow \mathbf{R}^n$, and ξ_k is a noise vector with the known density.

b) Observational equation:

$$y_k = h_k(x_k, \eta_k), \tag{2.28}$$

where $h_k : \mathbf{R}^n \times \mathbf{R}^p \longrightarrow \mathbf{R}^r$, and η_k is a noise vector with the known density.

The initial density of the state vector $p(x_0)$ is assumed to be known. One can not observe the state vector x_k, but instead at time k the set of observations $\{y_1, y_2, ..., y_k\}$ is available, we denote it as D_k.

Suppose that at time $k-1$ we have a particle cloud (sample) $\{x_{k-1}^1, x_{k-1}^2, ..., x_{k-1}^M\}$ of size M, that approximates the posterior distribution $p(x_{k-1} \,|D_{k-1})$. Gordon et al. [13] proposes the following algorithm to estimate x_k when y_k is observed:

(1) Pass the sample via the system model, i.e. draw x_k^{j*} using the state equation $q_{k-1}(x_{k-1}^j, \xi_{k-1})$, $j = 1 \div M$. This transforms the cloud $\{x_{k-1}^1, x_{k-1}^2, ..., x_{k-1}^M\}$ into the cloud $\{x_k^{1*}, x_k^{2*}, ..., x_k^{M*}\}$, each particle in the latter sample has weight $1/M$.

(2) Having received the observation y_k, re-evaluate the weight of each particle:

$$w_k^j = \frac{p(y_k \,|x_k^{j*})}{\sum_{j=1}^{M} p(y_k \,|x_k^{j*})}, \tag{2.29}$$

where conditional density $p(y_k \,|x_k^{j*})$ is defined by the observation equation (2.28).

(3) Define a discrete distribution on $\{x_k^{1*}, x_k^{2*}, ..., x_k^{M*}\}$ with probabilities $\{w_k^1, w_k^2, ..., w_k^M\}$. Resample M times from this discrete distribution to generate the sample $\{x_k^1, x_k^2, ..., x_k^M\}$. This sample is a particle representation of the conditional density $p(x_k \,|D_k)$.

CHAPTER 3

GRAPHICAL MODELS FOR LOCATION

3.1 Datasets

We collected RSS data from three floors at two sites, referred as *BR, CA Up* and *CA Down.* Both the *BR* and *CA* sites are office buildings and have deployed 802.11b wireless networks. Figure 3.1 shows the floor plans for the both sites.

To make our RSS measurements, we used a Linux IPAQ with a modified driver updated to scan for access points. The IPAQ had a custom client and a standard Konqueror web browser. The user making RSS measurements clicked on their current location in an image of the floor as displayed on the browser. The posting of this information triggered an RSS measurement request at the client from the web server on a separate TCP channel. The web server then recorded the coordinate and RSS vector information at that location. We did not specifically orient the IPAQ in any way while taking measurements.

The *BR* site has 5 access points and measures 225 ft. $\times$ 144 ft. We made 254 RSS measurements along the corridors of this site. The measurements were made over different sessions spanning several days.

The *CA Down* floor has 4 access points, three of which are collinear, and measures 250 ft. $\times$ 175 ft., with a "slice" removed. Due to the collinearity of the three access points, we installed two temporary access points. The *CA Up* floor has 4access points. At the *CA* site, a colleague took 146 measurements on the "Down" floor and 56 measurements on the "Up" floor.

3.2 Models and Experiments

Our goal is to construct a model that embodies extant knowledge about Wi-Fi signals as well as physical constraints implied the target building. We present a series of

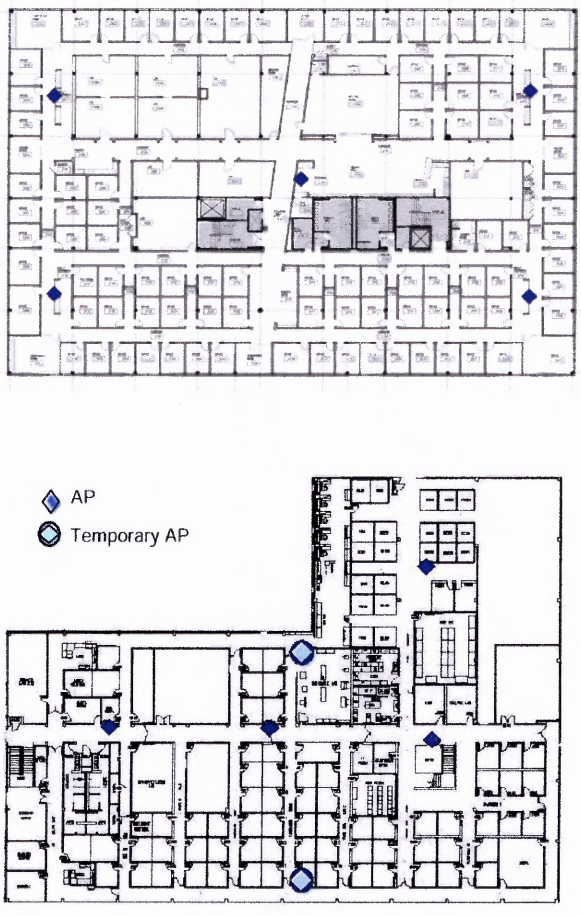


Figure 3.1 Floor plans for the *BR* and *CA* sites showing the access points.

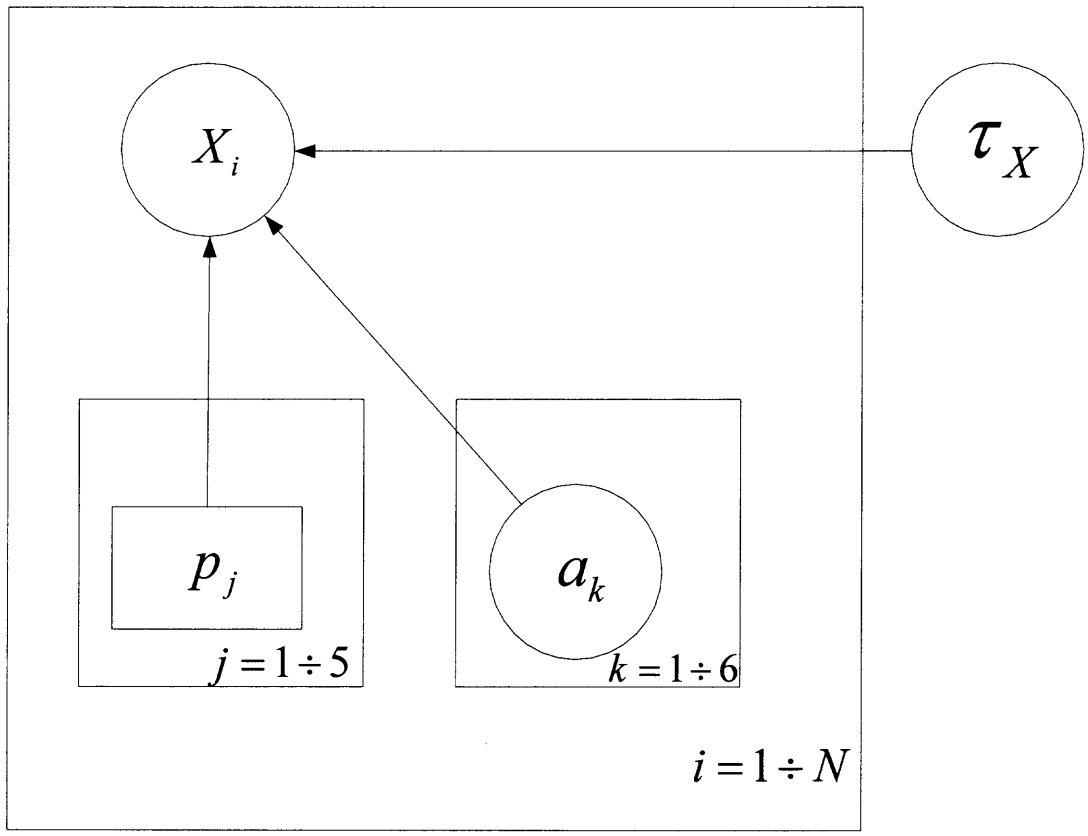


Figure 3.2 A simple regression model for X.

models of increasing complexity, in each case showing results with varying training dataset sizes. We focus throughout on predictive accuracy.

3.2.1 Independent Regressions (IR model)

Figures 3.2 and 3.3 show graphical models, using *BUGS* plate notation, for a two-dimensional location estimation problem in a building with five access points. We will refer to the combination of these two models as IR. The vertices X and Y represent location, p_j $(j = 1 \div 5)$ are signal strengths, and a_k $(k = 1 \div 6)$ and b_k $(k = 1 \div 6)$ are regression coefficients of two independent regression models. The first model regresses X on the vector of signal strengths, another does the same for Y.

Here is the specification of the IR model:

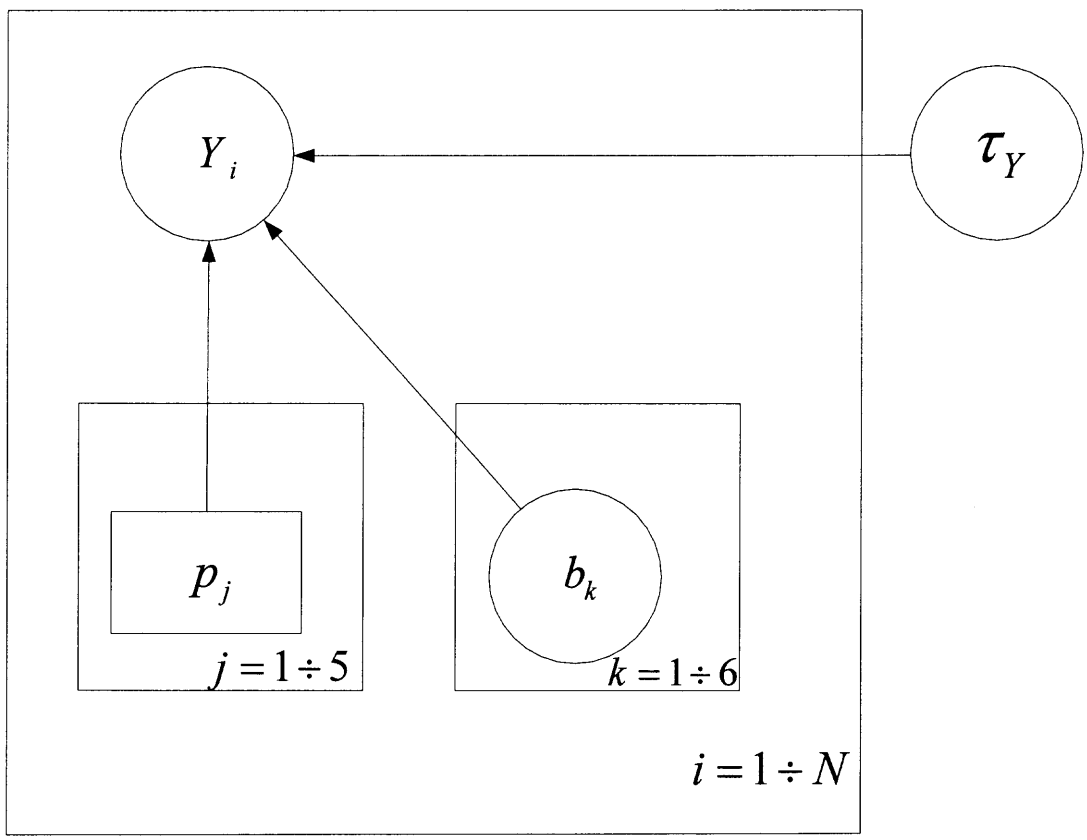


Figure 3.3 A simple regression model for Y.

$$
\begin{aligned}
X[i] &\sim \text{Normal}(mx[i], \tau_X), \\
mx[i] &= a[1] + a[2]p_1[i] + a[3]p_2[i] + a[4]p_3[i] + a[5]p_4[i] + a[6]p_5[i], \\
a[k] &\sim \text{Normal}(0, 0.001), \\
\tau_X &\sim \text{Gamma}(0.001, 0.001), \\
Y[i] &\sim \text{Normal}(my[i], \tau_Y), \\
my[i] &= b[1] + b[2]p_1[i] + b[3]p_2[i] + b[4]p_3[i] + b[5]p_4[i] + b[6]p_5[i], \\
b[k] &\sim \text{Normal}(0, 0.001), \\
\tau_Y &\sim \text{Gamma}(0.001, 0.001),
\end{aligned}
$$

Following the *WinBUGS* notation, 'Normal(μ, τ)' denotes a Gaussian distribution with mean μ and precision τ ($\tau = \frac{1}{\sigma^2}$). *WinBUGS* code for this model is given in Appendix A.1. Table 3.1 contains simulation results for the IR model. For each value of the training data set (10, 25, 50, 100, 150, and 200 points), we sampled from the BR data a training sample of the given size and 30 additional points to create a testing data set. The total number of iterations was 50,000 with a burn-in period of 40,000. For each size of the training data set, the procedure was repeated 10 times.

Table 3.2 contains simulation results for the second data set (*CA Down*). The sampling procedure was the same as for the BR data.

Figure 3.4 shows scatterplots that characterize possible application of IR models. The data output from one run of the IR model with 100 training data points was used. Blue dots represent last 1000 simulations of the IR model for the third, tenth, twelfth, seventieth, twenty fifth, and thirtieth point from the test data set, big red circles show the actual location of the appropriate test point. The fact that each simulated blue

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	0.642	12.620	21.130	26.360	33.960	104.700
25	0.988	11.890	17.360	19.320	24.520	71.380
50	0.762	11.320	17.370	18.980	24.340	54.390
100	0.397	10.470	16.270	17.120	22.010	45.480
150	0.776	10.220	16.420	17.340	23.090	65.670
200	1.208	10.400	16.180	17.490	22.620	63.590

Table 3.1 Predictive performance of the *IR* model on the *BR* data.

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	2.180	24.790	41.660	57.210	67.940	417.100
25	1.510	16.140	25.150	29.690	38.990	88.650
50	2.202	15.970	23.900	29.080	36.430	100.300
100	1.217	15.310	23.530	28.470	35.980	88.680
120	2.340	15.550	24.600	29.750	40.950	75.650

Table 3.2 Predictive performance of the *IR* model on the *CA Down* dataset.

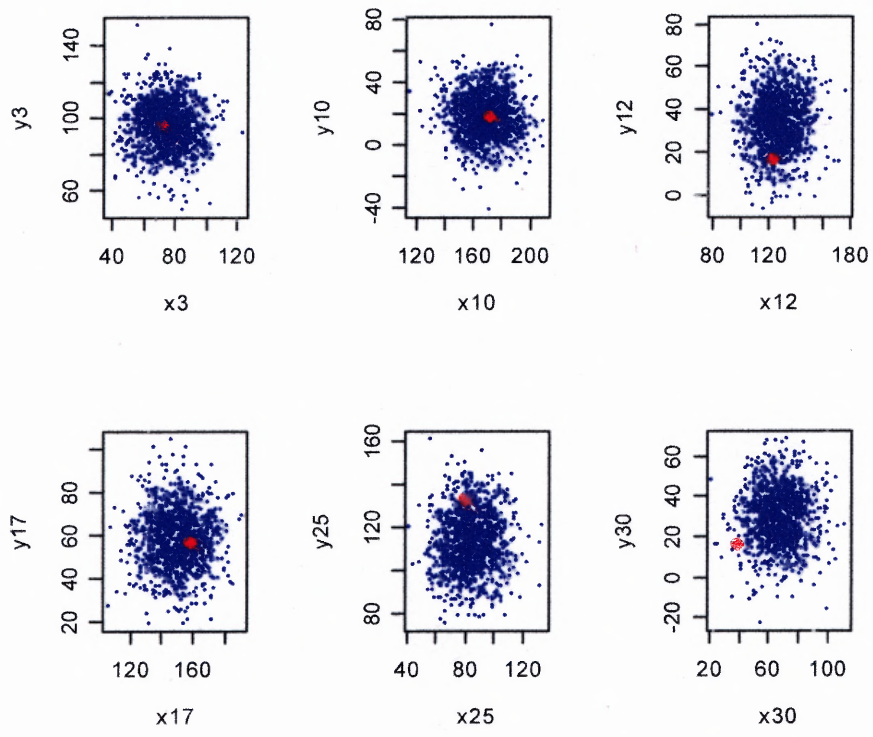


Figure 3.4 Scatterplots for the IR model.

cloud contains a red point, the actual location, may be used in combination with more sophisticated models.

3.2.2 Multivariate Regression Model (MR model)

Unlike IR model, that comprises of two independent regression models, the multivariate regression models the relationship between p response variables and a set of q explanatory variables [28]. In case of the BR data, there are two response variables: X and Y, and five predictor variables: p_1, p_2, ..., p_5. Bayesian model MR for the BR data is specified below:

$$
\begin{aligned}
\begin{pmatrix} X[i] \\ Y[i] \end{pmatrix} &\sim \text{Normal}\left(\begin{bmatrix} mx[i] \\ my[i] \end{bmatrix}, T \right) \\
mx[i] &= a[1] + a[2]p_1[i] + a[3]p_2[i] + a[4]p_3[i] + a[5]p_4[i] + a[6]p_5[i], \\
my[i] &= b[1] + b[2]p_1[i] + b[3]p_2[i] + b[4]p_3[i] + b[5]p_4[i] + b[6]p_5[i], \\
a[k] &\sim \text{Normal}(0, 1000), \\
b[k] &\sim \text{Normal}(0, 1000), \\
T^{-1} &\sim \text{Wishart}(D[,], 2).
\end{aligned}
$$

This Bayesian model uses the multivariate (bivariate in this case) normal distribution and Wishart distribution with 2 degrees of freedom (the rank of T) for the prior of the inverse of the covariance matrix of normal distribution. As pointed out by Lindley [18] , the choice of the scale matrix D has small effect on the posterior assessment of T^{-1}. We specified D as $\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. *WinBUGS* code for this model is given in Appendix A.2. Table 3.3 shows the simulation results for the MR model. The simulation procedure is identical to the one for the IR model, i.e. for each value of the training data set (10, 25, 50, 100, 150, and 200 points), the sample of the appropriate size was drawn from the BR data, and 30 additional points were drawn from BR data to create a testing data set. The total number of iterations for each run of the MR model was 50,000, with a burn-in period of 40,000. The procedure was repeated 10 times for the each sample size. As one can see, IR and MR models performed very similarly for every sample size. Table 3.4 shows simulation results for the *CA Down* dataset.

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	0.996	12.310	21.240	27.240	35.400	121.400
25	1.760	12.060	17.130	19.380	25.170	70.170
50	1.767	11.120	17.560	18.930	23.520	55.850
100	0.1972	10.210	15.930	17.090	21.920	43.680
150	0.892	10.110	16.270	17.270	23.120	66.870
200	0.528	10.640	15.890	17.510	22.87	65.060

Table 3.3 Predictive performance of the *MR* model on the *BR* dataset.

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	0.265	26.930	45.210	62.370	73.670	347.900
25	0.391	16.330	25.480	29.540	39.200	88.810
50	2.289	16.180	23.810	29.170	36.720	101.600
100	1.358	16.050	23.880	28.610	36.320	88.480
120	1.968	15.660	25.050	29.920	40.310	77.370

Table 3.4 Predictive performance of the *MR* model on the *CA Down* dataset.

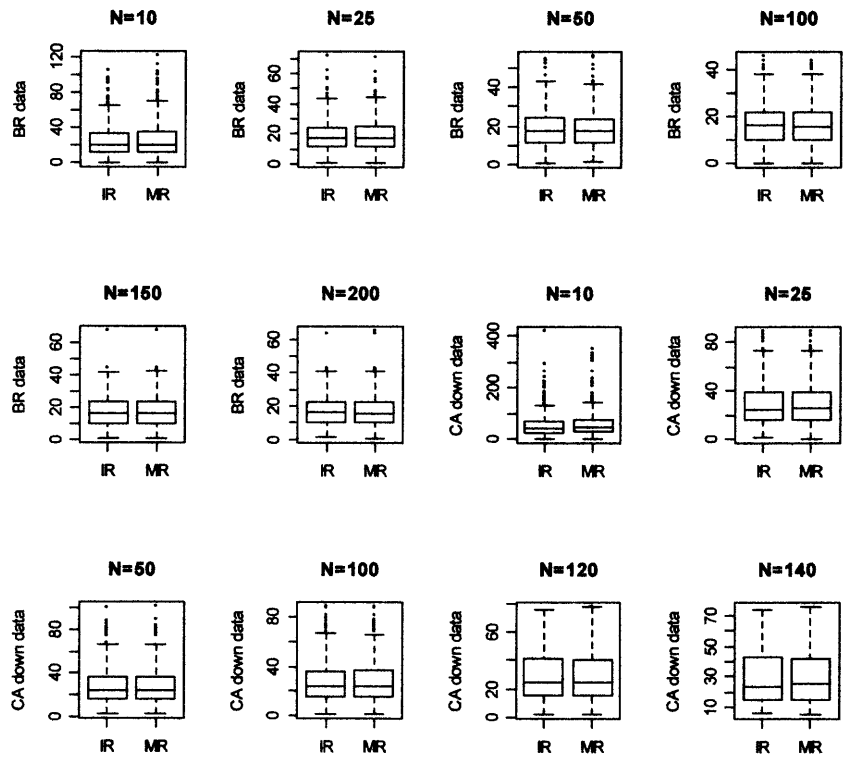


Figure 3.5 Predictive accuracy of the Independent Regressions model (IR) versus the Multivariate Regression model (MR) for the BR data and the CA *Down* data.

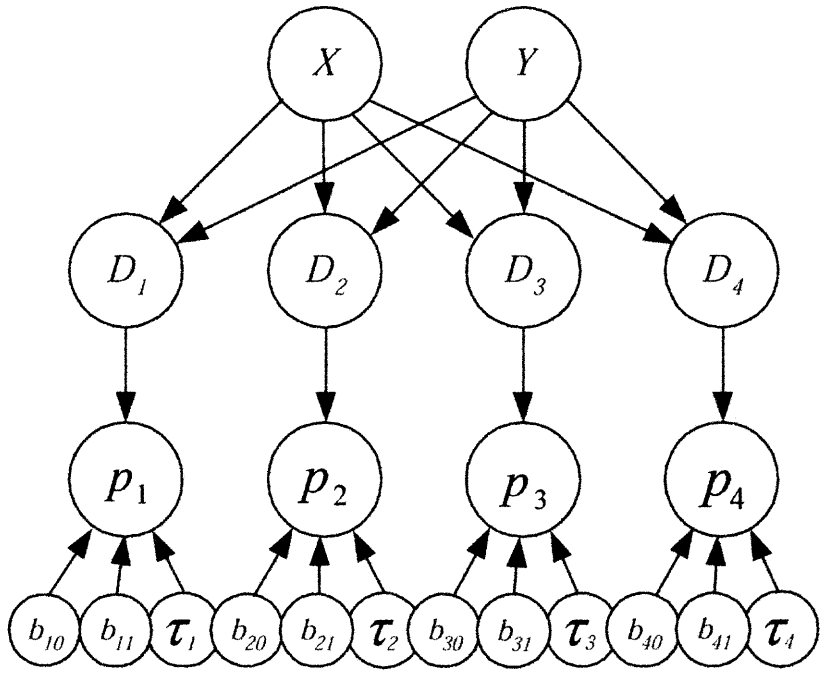


Figure 3.6 A Bayesian graphical model for location estimation. This is model M_1.

3.2.3 A Non-Hierarchical Bayesian Graphical Model that Uses Information about AP's Locations

Figure 3.6 shows a particular graphical model for a two-dimensional location estimation problem in a building with four access points. In what follows we refer to this model as M_1 (although the number of access points varies).

The vertices X and Y represent location. The vertex D_1 (respectively D_2, D_3, and D_4) represents the euclidean distance between the location specified by X and Y and the first (respectively second, third, and fourth) access point. Since we assume the locations of the access points are known, the D_i's are deterministic functions of X and Y. The vertex p_i represents the signal strength measured at (X, Y) with respect to the ith access point, $i = 1, \ldots, 4$. The model assumes that X and Y are marginally independent.

Specification of the model requires a conditional density for each vertex given its parents as follows:

$$\begin{aligned} X &\sim \text{Uniform}(0, L), \\ Y &\sim \text{Uniform}(0, B), \\ p_i &\sim N(b_{i0} + b_{i1} \log D_i, \tau_i), i = 1, 2, 3, 4, \\ b_{i0} &\sim N(0, 0.001), i = 1, 2, 3, 4, \\ b_{i1} &\sim N(0, 0.001), i = 1, 2, 3, 4. \end{aligned}$$

Here L and B denote the length and breadth of the building respectively. The distributions for X and Y reflect the physical constraints of the building. The model for p_i reflects the fact that signal strength, decays approximately linearly with log distance. Note that we use $N(\mu, \tau)$ to denote a Gaussian distribution with mean μ and precision τ so that the prior distributions for b_{i0} and b_{i1} have large variance.

Figure 3.7 shows a more compact representation for M_1 using the *BUGS* plate notation for replicated sub-models and with d denoting the number of access points. *WinBUGS* code for model M_1 is given in Appendix A.3.

Figure 3.8 shows the predictive performance of model M_1 on the *BR* data, as a function of training set size. Specifically, for each training set size N, we plot the average performance for 30 replications of a random test-training split, using N observations for training and one observation for testing. The red solid curve shows the results for M_1 (i.e., the model of Figure 3.7). In each case, and throughout this chapter, the estimates resulted from 110,000 MCMC iterations, discarding the first 10,000. This seemed to provide adequate convergence in most cases, according to standard *BUGS* diagnostics. For comparison purposes, the blue dotted curve shows the equivalent results for the smoothed nearest-neighbor "*SmoothNN*] model of Krishnan and coworkers [14]. The *SmoothNN* model proved highly competitive in comparison with two other benchmark systems and hence we use it for comparison

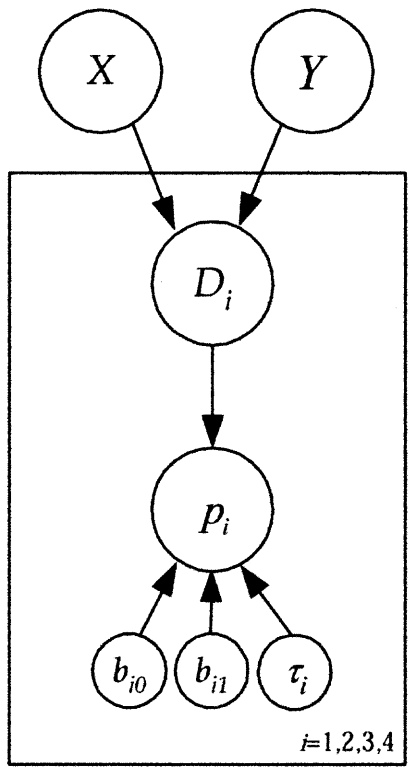


Figure 3.7 A Bayesian graphical model in plate notation (model M_1).

	Training Sample Size					
Model	5	10	20	50	100	253
Bayesian M_1	20.1	18.1	15.2	14.7	15.2	14.8
SmoothNN	39.7	18.7	17.5	16.2	12.3	13.00

Table 3.5 Leave-one-out average accuracy in feet for the *BR* data. Results are averaged over 30 replications.

	Training Sample Size				
Model	5	10	20	50	145
Bayesian M_1	28.8	27.4	21.6	18.2	19.9
SmoothNN	46.6	26.7	24.3	17.1	17.4

Table 3.6 Leave-one-out average accuracy in feet for the *CA Down* data. Results are averaged over 30 replications.

purposes in this paper. The Figure 3.8 shows that M_1 outperforms the *SmoothNN* model with smaller training sample sizes but underperforms the *SmoothNN* model at the larger sample sizes.

Figure 3.9 provides shows more detail and also shows results for the other two datasets. The results for the three different datasets are qualitatively similar. Tables 3.5, 3.6, and 3.7 provide corresponding summary statistics. Note that predictive accuracy does tend to improve with training sample size, although not in every case.

	Training Sample Size			
Model	5	10	20	55
Bayesian M_1	35.4	31.7	30.5	28.5
SmoothNN	59.9	36.3	25.2	28.2

Table 3.7 Leave-one-out average accuracy in feet for the *CA Up* data. Results are averaged over 30 replications.

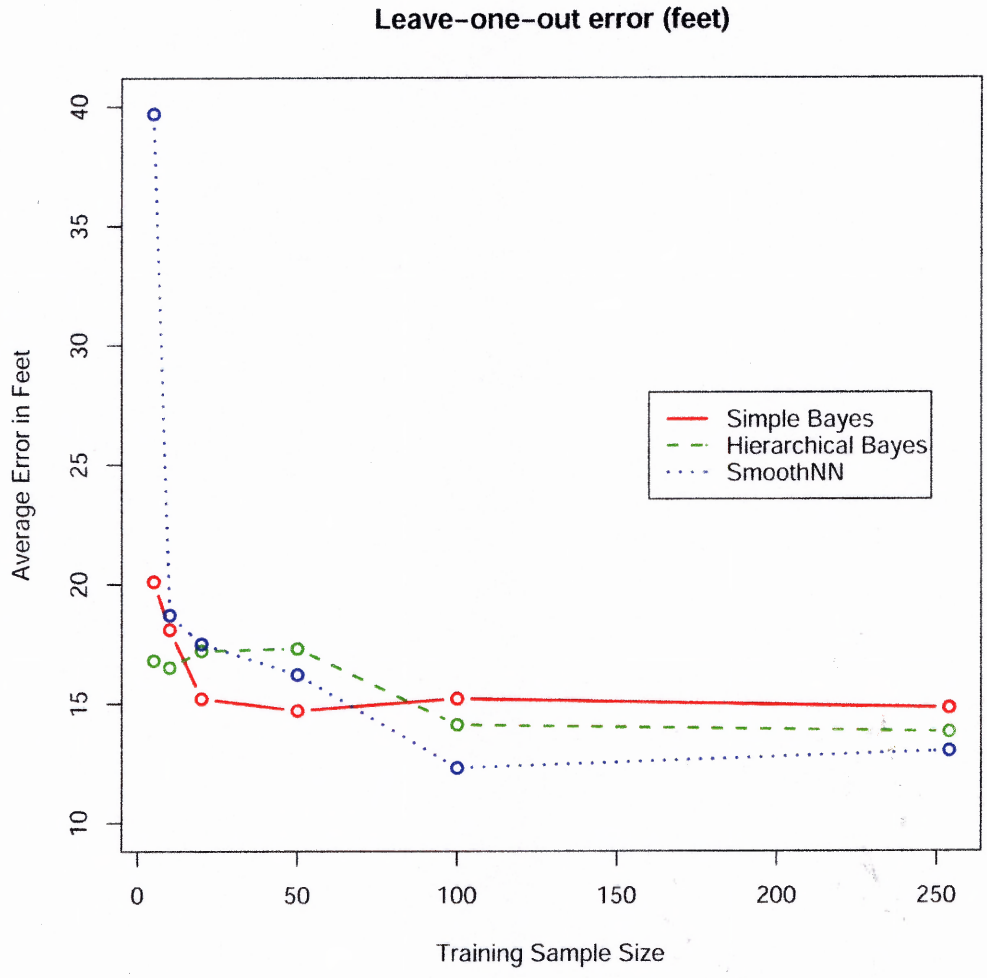


Figure 3.8 Average predictive accuracy of the non-hierarchical Bayesian graphical model M_1, the hierarchical model M_2, and the *SmoothNN* model on the BR data.

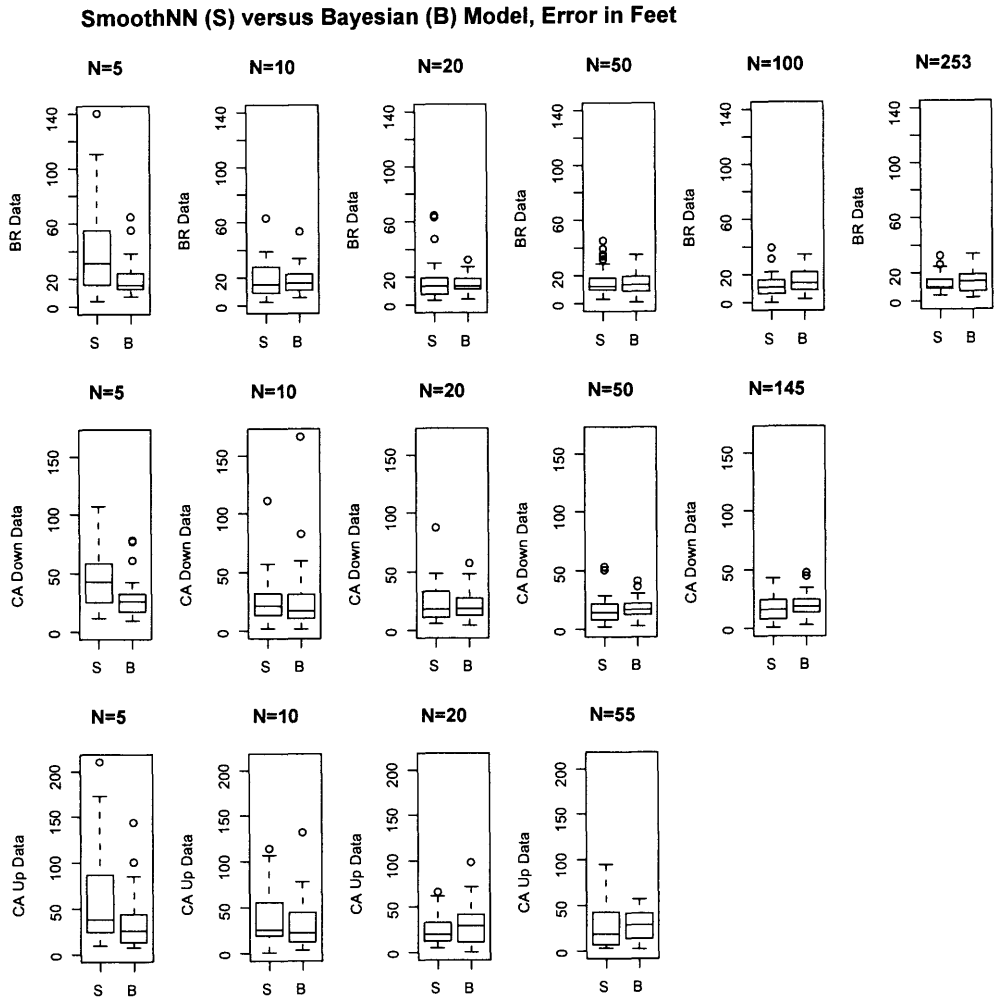


Figure 3.9 Predictive accuracy of the *SmoothNN* model vs. the non-hierarchical Bayesian model (M_1), for the *BR* data, *CA Down* data, and *CA Up* data.

3.2.4 A Hierarchical Bayesian Graphical Model

Next we seek to incorporate the knowledge that the coefficients of the linear regression models corresponding to each of the access points should be similar since the similar physical processes are in play at each access point. Physical differences between locations of the different access points will tend to mitigate the similarity but nonetheless, borrowing strength across the different regression models might provide some predictive benefits.

Figure 3.10 shows the hierarchical model M_2. The conditional densities for this model are (see Appendix A.4 for the *WinBUGS* code):

$$
\begin{aligned}
X &\sim \text{Uniform}(XL, XR), \\
Y &\sim \text{Uniform}(YL, YR), \\
p_i &\sim N(b_{i0} + b_{i1} \log D_i, \tau_i), i = 1, \ldots, d, \\
b_{i0} &\sim N(b_0, \tau_{b0}), i = 1, \ldots, d, \\
b_{i1} &\sim N(b_1, \tau_{b1}), i = 1, \ldots, d, \\
b_0 &\sim N(0, 0.001), \\
b_1 &\sim N(0, 0.001), \\
\tau_{b0} &\sim \text{Gamma}(0.001, 0.001) \\
\tau_{b1} &\sim \text{Gamma}(0.001, 0.001)
\end{aligned}
$$

The green dashed curve in Figure 3.8 shows the predictive accuracy of M_2 on the *BR* data. A comparison of M_1 and M_2 shows that the hierarchical model performs similarly to its non-hierarchical counterpart, although M_2 does provide improvement in average error for the smallest training sample size of 5.

Figure 3.11 provides shows more detail and also shows results for the other two datasets. Again, the results for the three different datasets are qualitatively

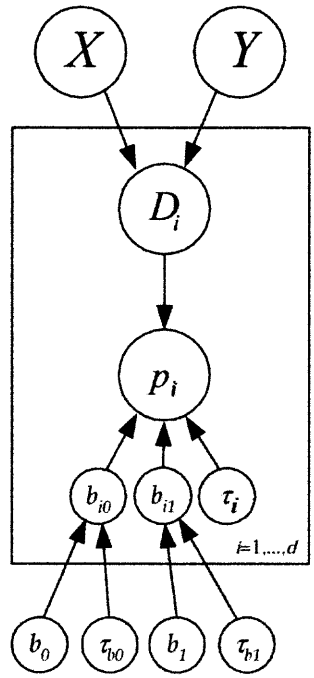


Figure 3.10 A Bayesian hierarchical graphical model using plate notation (model M_2).

similar. Tables 3.8, 3.9, and 3.10 provide corresponding summary statistics. In general, the results show small differences between the non-hierarchical model M_1 and the hierarchical model M_2.

3.2.5 Training Data With No Location Information

Model M_2 incorporates two sources of prior knowledge. First, M_2 embodies the knowledge that signal strength decays approximately linearly with log distance. Second,

	Training Sample Size					
Model	5	10	20	50	100	253
Bayesian M_1	20.1	18.1	15.2	14.7	15.2	14.8
Bayesian M_2	16.8	16.5	17.2	17.3	14.1	13.8

Table 3.8 Leave-one-out average accuracy in feet for the *BR* data. Results are averaged over 30 replications.

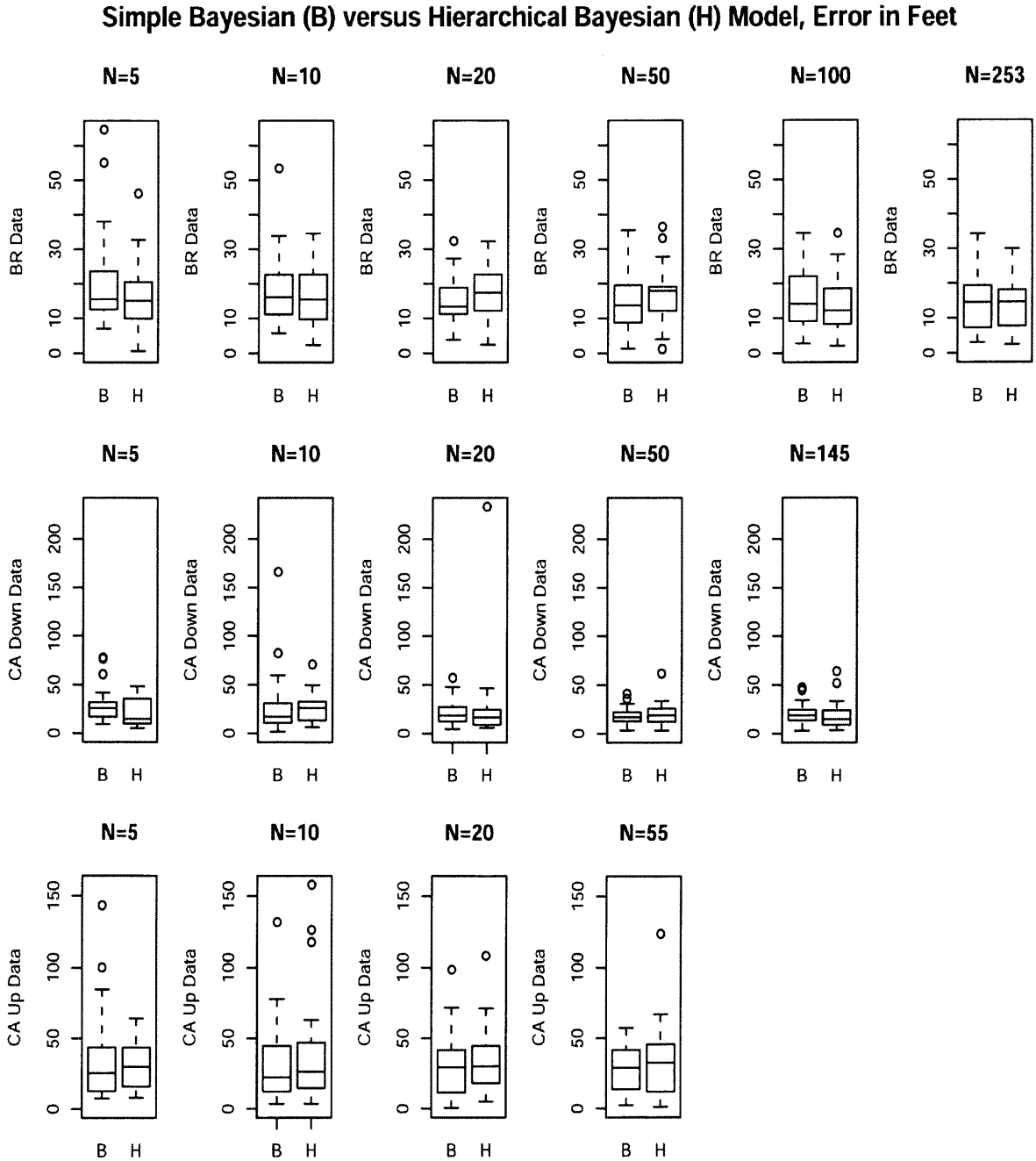


Figure 3.11 Predictive accuracy of the non-hierarchical Bayesian graphical model (M_1) vs. the hierarchical Bayesian graphical model (M_2) for the BR data, $CA\ Down$ data, and $CA\ Up$ data.

	Training Sample Size				
Model	5	10	20	50	145
Bayesian M_1	28.8	27.4	21.6	18.2	19.9
Bayesian M_2	21.3	26.3	25.0	20.3	18.7

Table 3.9 Leave-one-out average accuracy in feet for the *CA Down* data. Results are averaged over 30 replications.

	Training Sample Size			
Model	5	10	20	55
Bayesian M_1	35.4	31.7	30.5	28.5
Bayesian M_2	30.6	37.9	33.0	33.5

Table 3.10 Leave-one-out average accuracy in feet for the *CA Up* data. Results are averaged over 30 replications.

the hierarchical portion of M_2 reflects prior knowledge that the different access points behave similarly. Here we pursue the idea that perhaps this prior knowledge provides sufficient constraints to obviate the need to know the actual locations of the training data observations. Specifically, the training data now comprise vectors of signal strengths with *unknown* locations; X and Y in M_1 and M_2 become latent variables.

Figure 3.12 shows the average predictive performance for the *BR* data with different numbers of randomly signal strength vectors. In each case the results shows averages over 30 replications, except for the maximal case (254 for *BR*, 146 for *CA Down*, 56 for *CA Up*) which uses all the signal strength vectors in the training data. The red solid curve corresponds to M_1 and the green dashed curve to M_2. The results for the *SmoothNN* model are reproduced from Figure 3.8 and reflect training data *with known locations.* These results show some striking features. With no location information, M_1 performs poorly and shows no improvement with increasing numbers of signal strength vectors. Model M_2, however, from about 10 training

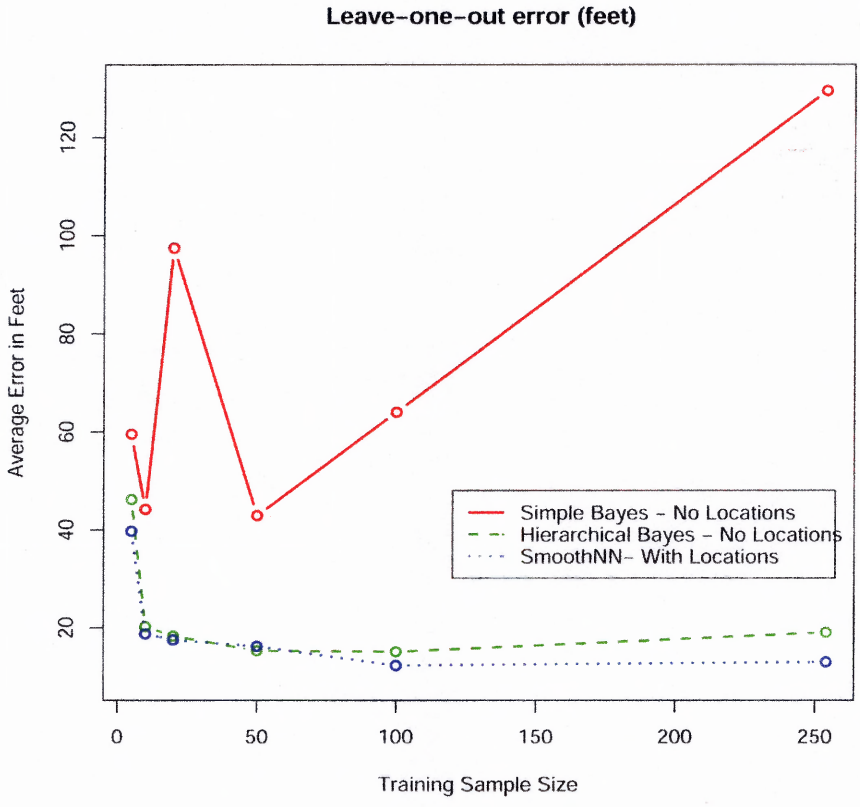


Figure 3.12 Average predictive accuracy of the non-hierarchical Bayesian graphical model M_1, the hierarchical model M_2, and the *SmoothNN* model on the BR data with no location information.

vectors onwards, performs almost as well as the *SmoothNN* model trained on data with complete location information for each signal strength vector!

Figure 3.13 provides shows more detail and also shows results for the other two datasets. Once again, the results for the three different datasets are qualitatively similar. Tables 3.11, 3.12, and 3.13 provide corresponding summary statistics. In each case the hierarchical model, even with no location information, provides predictive performance that is close to, although not as good as, the state-of-the-art *SmoothNN* model (that uses location information).

Dropping the location data requirement affords significant practical benefits. As discussed in chapter 1, the location measurement process is slow and human-intensive.

	Training Sample Size					
Model	5	10	20	50	100	253
Bayesian M_1, No Locations	59.5	44.2	97.4	42.9	64.0	129.5
Bayesian M_2, No Locations	46.2	20.2	18.3	15.3	15.1	19.0
SmoothNN, With Locations	39.7	18.7	17.5	16.2	12.3	13.0

Table 3.11 Leave-one-out average accuracy in feet for the *BR* data. No location information in the training data. Results are averaged over 10 replications.

	Training Sample Size				
Model	5	10	20	50	145
Bayesian M_1, No Locations	66.7	46.9	52.5	58.1	67.5
Bayesian M_2, No Locations	23.9	29.4	29.2	29.8	21.9
SmoothNN, With Locations	46.3	26.7	24.3	17.1	17.4

Table 3.12 Leave-one-out average accuracy in feet for the *CA Down* data. No location information in the training data. Results are averaged over 10 replications.

	Training Sample Size			
Model	5	10	20	55
Bayesian M_1, No Locations	46.4	33.2	59.1	89.0
Bayesian M_2, No Locations	30.6	37.9	33.0	33.5
SmoothNN, With Locations	59.9	36.3	25.2	28.2

Table 3.13 Leave-one-out average accuracy in feet for the *CA Up* data. No location information in the training data. Results are averaged over 10 replications.

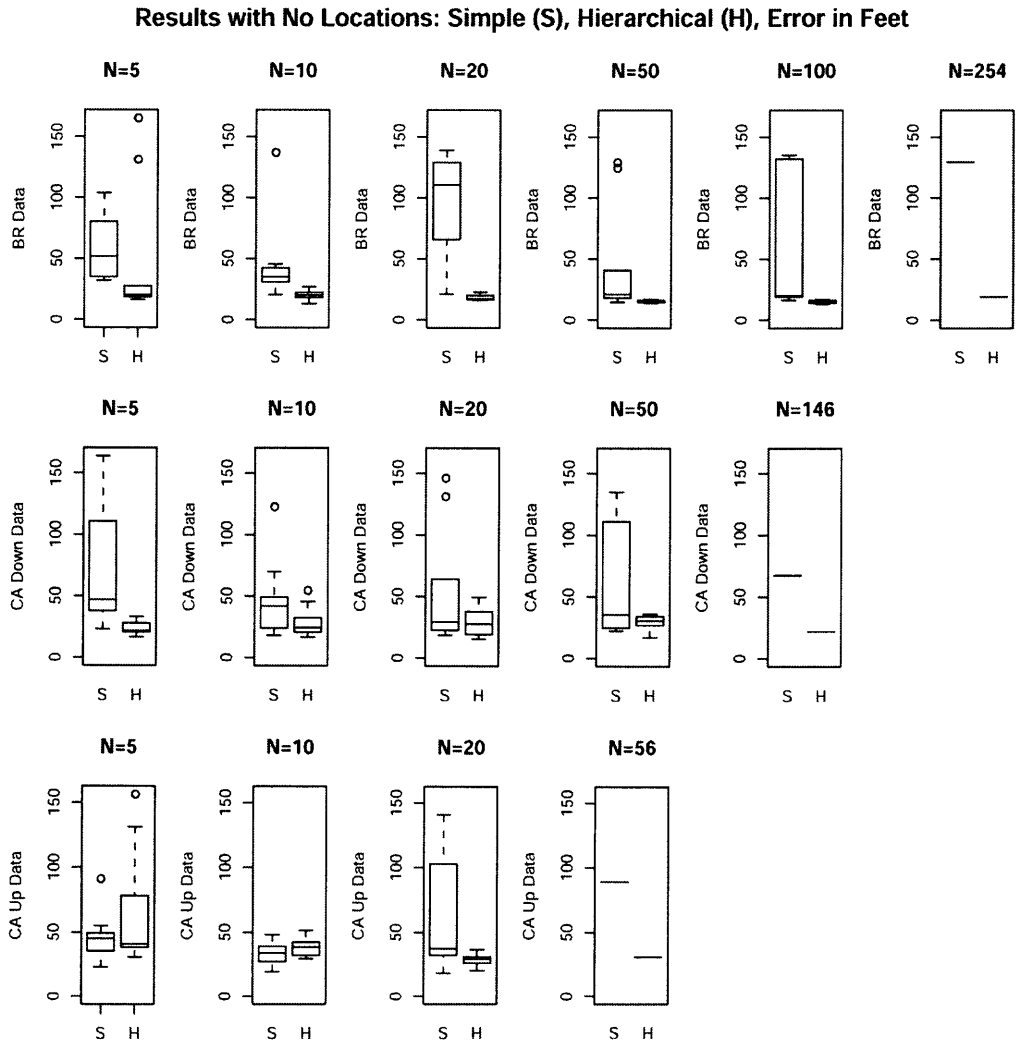


Figure 3.13 Predictive accuracy of the Bayesian graphical models with no location information. Non-hierarchical model (M_1) vs. the hierarchical Bayesian graphical model (M_2) for the BR data, $CA\ Down$ data, and $CA\ Up$ data.

By contrast, gathering signal strengths vectors without the corresponding locations does not require human intervention; in the infrastructure approach, suitably instrumented access points or sniffing devices can solicit signal strength measurements from existing Wi-Fi devices and can do this repeatedly at essentially no cost.

We note the existing location estimation algorithms that we are aware of all require location information in the training data to produce *any* estimates.

CHAPTER 4

SPLINE MODELS

All models that we have considered so far used the fact that the signal strength depends on the distance between a wireless device and an access point. Such dependence allowed us to develop several models for predicting the location of a wireless device using RSS. In the ideal world the signal strength would attenuate evenly in all directions, and contours of constant signal response would be concentric circles with an access point as the center. In the real life the situation is different, every floor of each building has a complicated structure. Walls and objects of various nature distort a signal significantly, whereas corridors allow better signal transmission along them. One can see that from the contour plots on [fig.4.1]. There were some works that proposed a model for WI-FI signal propagation inside buildings. The general idea is the following: radio waves, generated by a transmitter, take multiple paths to a receiver. If one can accurately model these paths, it will enable him to calculate the average signal strength at every point of a building, and, therefore, use this information for prediction purposes. But to do that, a very accurate building model is required, and the model must be updated with every major change in the floor configuration, etc.

We would like to consider the whole situation from another point of view. Instead of describing the behavior of "each particle in the Universe," i.e. instead of using some kind of propagation model, we would like to create a "signal map" for each access point, and then to use these maps in our general graphical model framework to predict the location of the transmitter. Different modifications and extensions of the approach will be discussed.

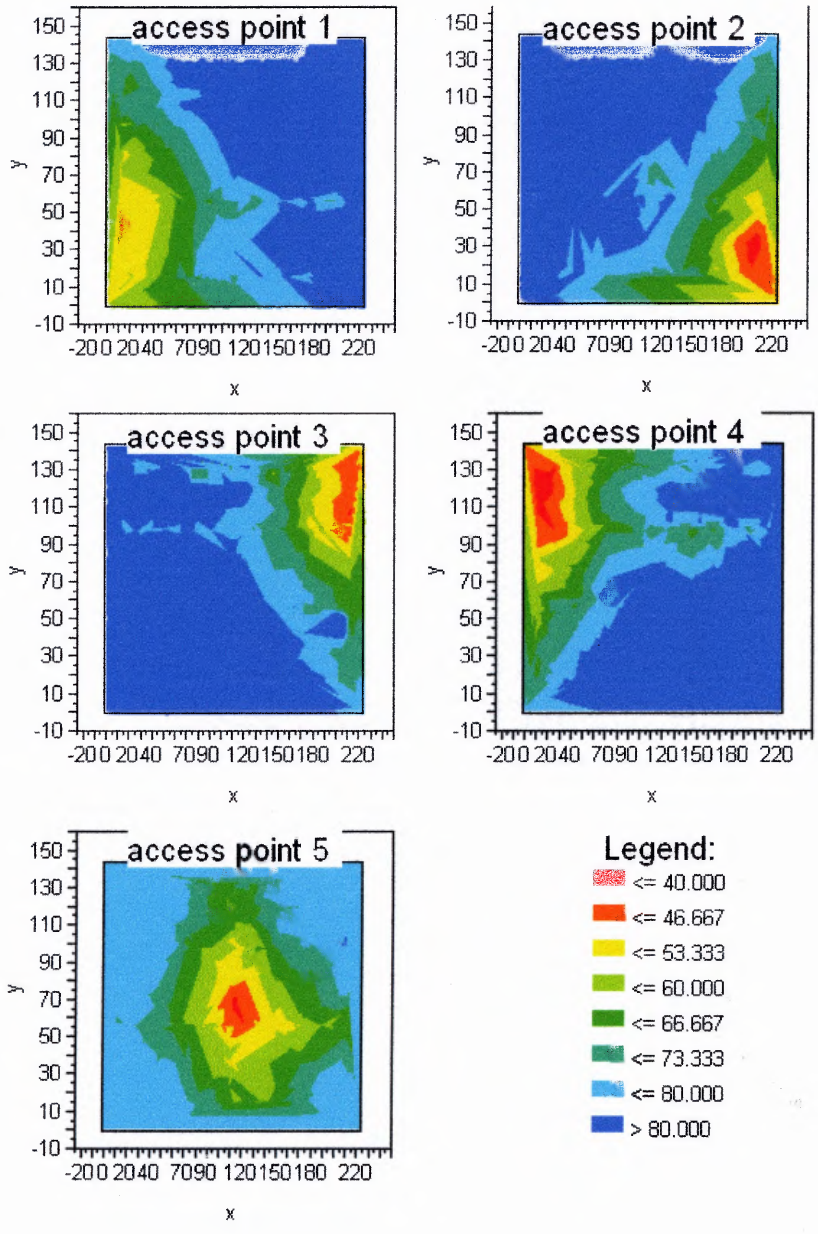


Figure 4.1 Contour plots for the *BR* data set (JMP output).

4.1 A Bayesian Spline Model (BS)

Let's consider scatterplots of the received signal strengths $p1 - p5$ versus location coordinates (x and y). Figure 4.2 shows such scatterplots for the BR data. Due to the technical restrictions, standard WI-FI equipment can detect a signal in the -92..-32 dBm range. If signal becomes stronger than -32 dBm or weaker than -92 dBm, it will be truncated to -32 and -92 respectively.

If we divide the whole $X - Y$ domain into smaller regions, the dependence between signal strength and variables x and y is close to linear for each of these subdivisions. This observation is valid for every access point. On the other side, the overall relationships between signal strengths and (x, y) coordinates are quite complicated. They can not be adequately described by means of usual (parametric) regression of signal strengths on x, y, and their interactions, even if one decides to use nonlinear terms. Instead we can think about some kind of response surface model [21] that utilizes nonparametric (semiparametric) spline approach. One-dimensional penalized splines from the Bayesian point of view were described in section 2.4. To deal with three-dimensional scatterplots (two-dimensional spline), we have to extend the set of basis functions to include x-terms, y-terms, and their interactions. Instead of using all possible pairwise products of basis functions, we will use points $\{(x_k, y_k)\}_{k=1,2,...,Kxy}$ from the training data set to form paired basis functions:

$$xy, (x - x_k)_+(y - y_k)_+, k = 1, 2, ..., Kxy, \tag{4.1}$$

where Kxy is the total number of points in the set. Sets $\{\xi_i\}_{i=1,2,...,Kx}$ and $\{\eta_i\}_{i=1,2,...,Ky}$ of the unique values of x and y, respectively, are the knots of x-terms and y-terms:

$$1,\ x,\ (x - \xi_1)_+,\ (x - \xi_2)_+, ...,\ (x - \xi_{Kx})_+$$

$$1, y, (y - \eta_1)_+, (x - \eta_2)_+, ..., (x - \eta_{Ky})_+ \tag{4.2}$$

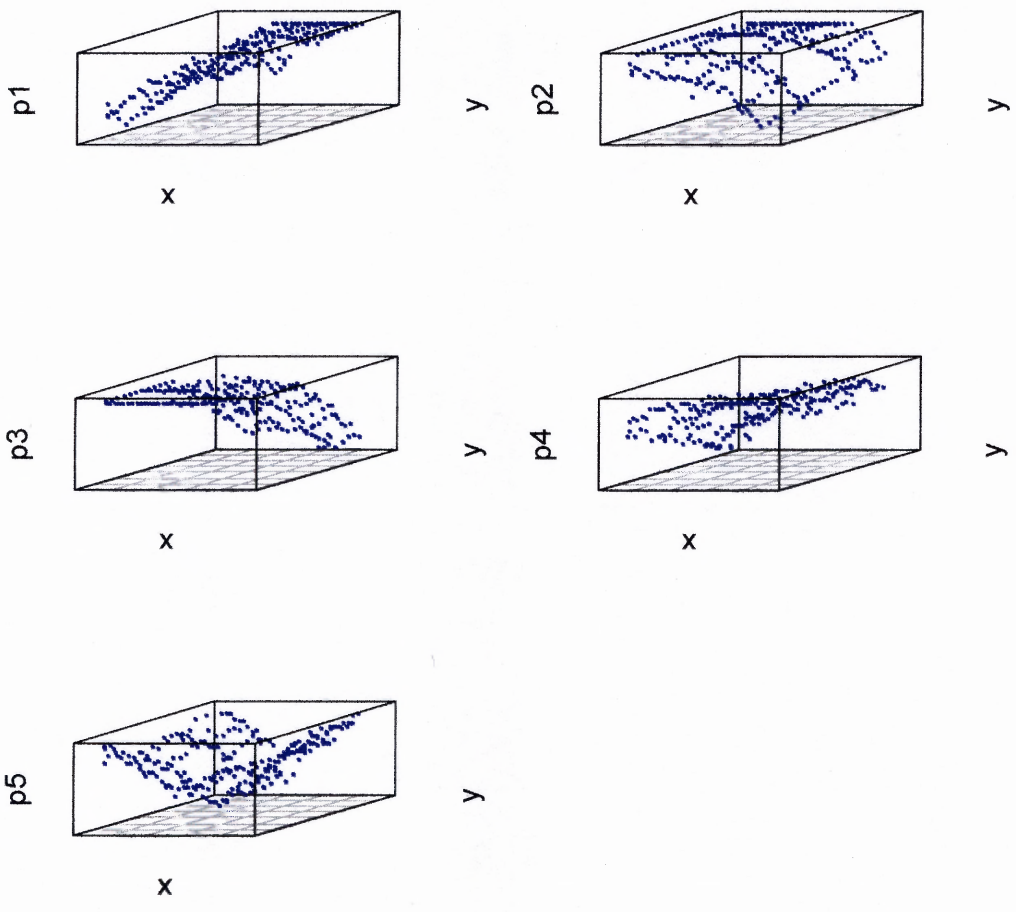


Figure 4.2 3D scatterplots for the *BR* data.

, Kx is a number of the unique values of x, and Ky – a number of the unique values of y in the training data. Functions (4.1) and (4.2) form a set of bivariate basis functions.

This leads to the following model for the average signal strength with respect to a particular access point:

$$m(x,y) = \beta[1] + \beta[2]x + \beta[3]y + \beta[4]xy + \sum_{k=1}^{Kx} b[k](x-\xi_k)_+ +$$

$$+ \sum_{k=1}^{Ky} d[k](y-\eta_k)_+ + \sum_{k=1}^{Kxy} s[k](x-x_k)_+(y-y_k)_+ \qquad (4.3)$$

Knot specification for spline models is a non-trivial problem. If there are K potential knots, there will be 2^K possible knot sets. But as pointed out by many researches [34, 43], in the case of penalised splines knot specification is a minor issue.

We will use here a modification of the approach proposed by Ruppert [34]. He suggests to use the sample quantiles of the independent variables as the spline knots. Taking into consideration the fact that, due to the cost and labor reasons, training data sets for the location problem usually contain not so many observations, typically a hundred or two for each floor of a building, we are going to use every (x, y) coordinate of each training point as a knot in the spline model. Note that instead of using a *tensor product basis* that includes all pairwise products of x-terms and y-terms from (4.2), we use only (x, y) points from the training data set as knots in the two-dimensional part of (4.3). Here is the specification of the spline model (BS model) for the BR data set (XL, XR, YL, and YR are constants that characterize geometry of the floor):

$$
\begin{aligned}
p1[i] &\sim \text{Normal}(m1[i], \tau_{p1}), \\
p2[i] &\sim \text{Normal}(m2[i], \tau_{p2}), \\
p3[i] &\sim \text{Normal}(m3[i], \tau_{p3}), \\
p4[i] &\sim \text{Normal}(m4[i], \tau_{p4}), \\
p5[i] &\sim \text{Normal}(m5[i], \tau_{p5}), \\
x[i] &\sim \text{Uniform}(XL, XR), \\
y[i] &\sim \text{Uniform}(YL, YR),
\end{aligned}
$$

$$
\begin{aligned}
m1(x[i],y[i]) &= \beta1[1]+\beta1[2]x[i]+\beta1[3]y[i]+\beta1[4]x[i]y[i]+\sum_{k=1}^{Kx} b1[k](x-\xi_k)_+ + \\
&\quad +\sum_{k=1}^{Ky} d1[k](y-\eta_k)_+ +\sum_{k=1}^{Kxy} s1[k](x-x_k)_+(y-y_k)_+, \\
m2(x[i],y[i]) &= \beta2[1]+\beta2[2]x[i]+\beta2[3]y[i]+\beta2[4]x[i]y[i]+\sum_{k=1}^{Kx} b2[k](x-\xi_k)_+ + \\
&\quad +\sum_{k=1}^{Ky} d2[k](y-\eta_k)_+ +\sum_{k=1}^{Kxy} s2[k](x-x_k)_+(y-y_k)_+, \\
m3(x[i],y[i]) &= \beta3[1]+\beta3[2]x[i]+\beta3[3]y[i]+\beta3[4]x[i]y[i]+\sum_{k=1}^{Kx} b3[k](x-\xi_k)_+ + \\
&\quad +\sum_{k=1}^{Ky} d3[k](y-\eta_k)_+ +\sum_{k=1}^{Kxy} s3[k](x-x_k)_+(y-y_k)_+, \\
m4(x[i],y[i]) &= \beta4[1]+\beta4[2]x[i]+\beta4[3]y[i]+\beta4[4]x[i]y[i]+\sum_{k=1}^{Kx} b4[k](x-sx_k)_+ + \\
&\quad +\sum_{k=1}^{Ky} d4[k](y-\eta_k)_+ +\sum_{k=1}^{Kxy} s4[k](x-x_k)_+(y-y_k)_+, \\
m5(x[i],y[i]) &= \beta5[1]+\beta5[2]x[i]+\beta5[3]y[i]+\beta5[4]x[i]y[i]+\sum_{k=1}^{Kx} b5[k](x-\xi_k)_+ + \\
&\quad +\sum_{k=1}^{Ky} d5[k](y-\eta_k)_+ +\sum_{k=1}^{Kxy} s5[k](x-x_k)_+(y-y_k)_+,
\end{aligned}
$$

$$
\begin{aligned}
\beta1,\beta2,\beta3,\beta4,\beta5 &\sim \text{Normal}(\mu_{beta},\tau_{beta}), \\
b1,b2,b3,b4,b5 &\sim \text{Normal}(\mu_b,\tau_b), \\
d1,d2,d3,d4,d5 &\sim \text{Normal}(\mu_d,\tau_d), \\
s1,s2,s3,s4,s5 &\sim \text{Normal}(\mu_s,\tau_s), \\
\mu_{beta},\mu_b,\mu_d,\mu_s &\sim \text{Normal}(0,0.001), \\
\tau_{beta},\tau_b,\tau_d,\tau_s, &\sim \text{Gamma}(0.001,0.001), \\
\tau_{p1},\tau_{p2},\tau_{p3},\tau_{p4},\tau_{p5}^2 &\sim \text{Gamma}(0.001,0.001).
\end{aligned}
$$

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	0.827	17.050	27.130	36.390	48.810	160.400
25	1.048	10.290	16.180	21.540	24.250	225.100
50	0.182	5.490	9.396	11.250	14.280	78.970
100	0.617	4.718	7.976	9.699	12.280	60.690
150	0.657	4.570	7.886	10.210	11.710	67.530
200	0.637	3.526	6.101	9.182	10.310	88.190

Table 4.1 Predictive accuracy in feet for the *BR* data (bivariate spline model).

WinBUGS code for the *BS* model is given in Appendix A.5. Tables 4.1 shows predictive accuracy of the *BS* model for the *BR* data. For each training set size N, we average performance for 5 replications of a random test-training split, using N observations for training and 30 observations for testing. We run 4,000 MCMC iterations for each replication of the experiment, discarding the first 3,000. Model performs quite well with a resolution (50th percentile) of under 2 meters for the training set of 200 points. On the training data sets of the same size, the *BS* model outperforms models from chapter 3, non-hierarchical and hierarchical Bayesian models, as well as the *SmoothNN* model that was used for comparison.

Figure 4.3 gives an interesting graphical illustration of the *BS* model. It shows surfaces (for 4 access points, a picture for AP 5 is given in Appendix A.6) that are generated by splines (green surfaces), training data set (blue points), and a test data set (red points). Table 4.2 shows predictive accuracy on the *CA Down* data. Although the resolution (50^{th} percentile) and 75^{th} percentiles look very good, mean error of $21-22$ feet is rather big. As mentioned above, access points at the *CA Down* site located in a way that makes our task harder to fulfil. Although the *BS* model shows smaller 25^{th}, 50^{th}, and 75^{th} percentiles than any model from chapter 3, those models outperform the *BS* model on the *CA Down* data if the mean predictive error

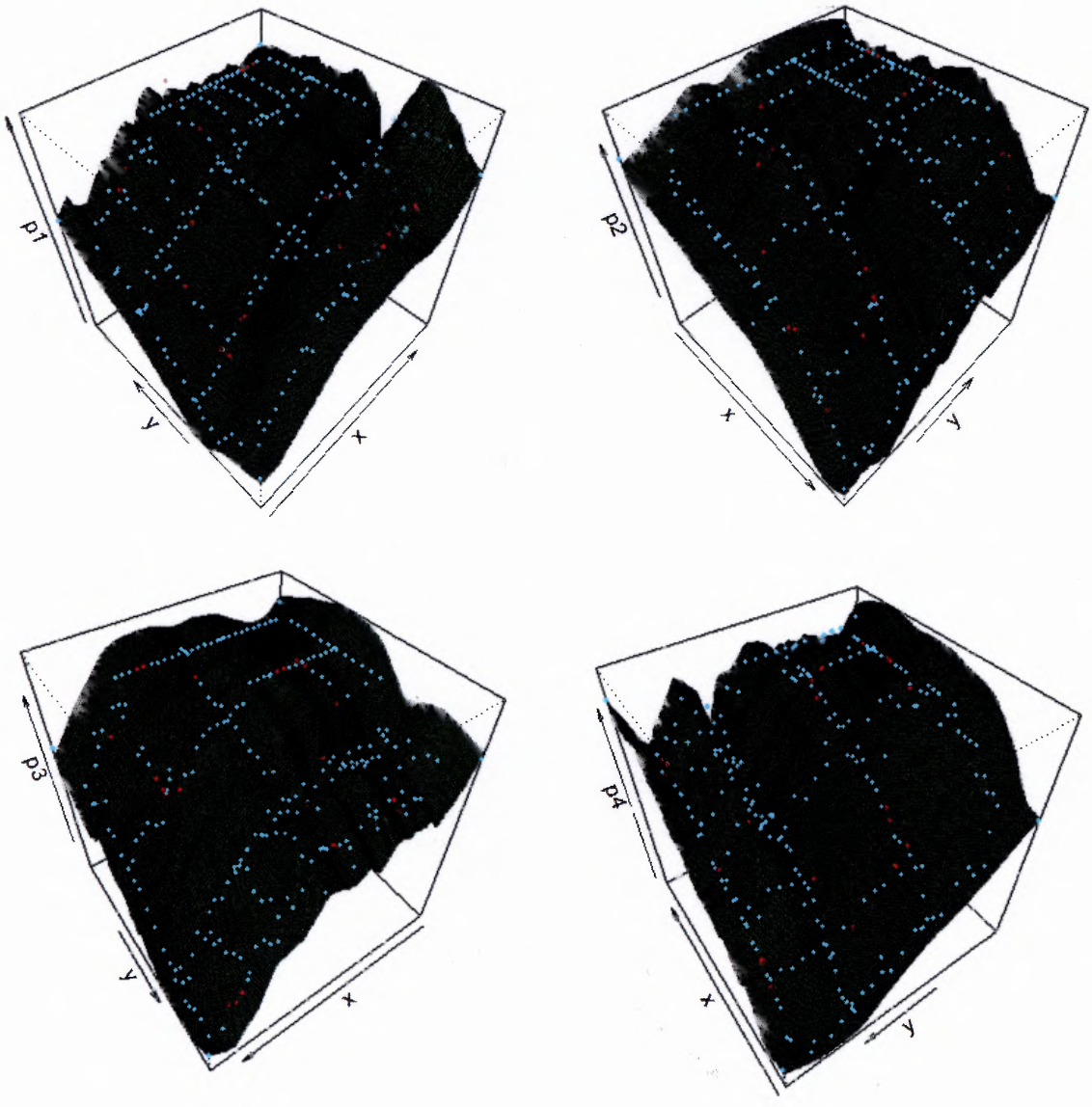


Figure 4.3 Graph illustrates how the *BS* model works on the *BR* data (4 APs are shown). Green surface is generated by splines. Blue points represent the training data set, red points belong to the test data.

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	1.432	14.140	31.200	51.590	56.950	237.500
25	0.785	9.537	14.980	27.330	22.750	237.500
50	0.698	7.940	12.400	22.120	19.080	237.200
100	0.635	6.334	9.986	22.170	13.680	227.500
120	1.032	7.064	9.707	21.200	14.800	229.900

Table 4.2 Predictive accuracy in feet for the *CA Down* data set (bivariate spline model).

is considered. Several points with big predictive errors draw the mean up. We will address this issue in the next section.

4.2 A Bayesian Spline Model with Informative Priors (BSI)

Here we seek to incorporate the knowledge about location coordinates, X and Y, that one may get from the simple models like IR or MR (chapter 3). We hope that such knowledge, in the form of prior distributions on X and Y (new, tighter, bounds on the uniform priors), will reduce the maximum predictive error and improve the overall prediction accuracy. We will consider two-step model BSI that combines models IR and BS as follows:

1. Use the IR model to calculate rough priors on X and Y;
2. Incorporate prior information into the BS model.

BSI model executes IR algorithm at step one, with $50,000$ iterations and a burn-in period of $40,000$. The last 1000 simulations of X and Y are used to get prior information for the BS model. Minimum of these 1000 iterations of X becomes a low bound on X in the BS model, and maximum of the last 1000 iterations creates

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	1.718	17.310	26.270	31.520	36.980	115.200
25	0.678	9.079	15.050	16.970	21.970	53.650
50	0.380	6.387	10.460	11.640	14.350	61.280
100	0.327	4.268	7.586	9.453	12.250	47.440
150	0.460	4.320	6.928	9.008	10.440	65.070
200	0.272	3.672	6.557	9.390	10.840	65.040

Table 4.3 Predictive accuracy in feet for the *BR* data set (*BSI* model with informative priors on X and Y).

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	0.767	14.080	24.770	37.840	44.590	192.400
25	2.518	8.760	12.990	16.240	22.370	59.680
50	1.334	7.727	11.800	13.720	16.690	65.890
100	0.671	5.977	9.529	11.760	13.230	141.800
120	0.955	6.695	9.501	12.000	14.140	65.010

Table 4.4 Predictive accuracy in feet for the *CA Down* data set with informative priors (*BSI* model).

an upper bound on X, the same procedure applies to Y. Table 4.3 shows predictive performance for the *BSI* model on the *BR* data. Both, *BS* and *BSI*, models show a resolution below 2 meters for training sets of size 200. As we expected, informative priors reduce the maximum predictive error.

Table 4.4 shows the predictive model accuracy on the *CA Down* data. On this data, the *BSI* model performs better than *BS*. Maximum predictive error is effectively bounded, and so as the mean error. *BSI* also outperforms all models from chapter 3 (but, unlike the hierarchical Bayesian model M_2, *BSI* can not be used without knowing location information for the training data set.)

4.3 Off-line/Online Version of the BSI Model ($Off-line/BSI$)

It takes approximately 5 hours (on 4.2 Gh Pentium IV PC with 1Gb of RAM) to calculate one run (4000 MCMC iterations) of BS or BSI with 150 training and 30 test points. To deal with this situation, we would like to investigate whether it is possible to use in the BSI model spline coefficients that were identified off-line (via the BS model).

This approach involves four stages:

1. Run the BS model only for splines coefficients;
2. Run the IR model to calculate model coefficients;
3. Use the IR model with coefficients from stage 2 to calculate rough priors on X and Y;
4. Use coefficients from step 1 and information from step 3, and
   run the BSI model only to identify unknown location (x, y).

At stage 1 we ran 10000 MCMC iterations with burn-in period of 8000, at stage 2 we ran 50000 MCMC iterations with 40000 burn-in, at stage 3 we ran 1100 MCMC iterations (100 burn-in) of the IR model with known (from stage 2) coefficients, then, at stage 4, we applied the BSI algorithm with total number of MCMC iterations of 2000, with 1000 burn-in period. Table 4.5 shows the predictive accuracy results for such an approach. This method allows to identify the location in seconds rather than in hours when using BS or BSI approaches. The $off-line/BSI$ approach performs on the BR data similarly to BS and BSI.

Table 4.6 gives the results of the third ($off-line/BSI$) approach for the *CA Down* data. Predictive accuracy of this approach is comparable to the predictive accuracy of the BSI method.

Figure 4.4 helps us to compare all three models (BS, BSI, and $off-line/BSI$). It shows that informative priors, derived from a simple regression model, improve

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	2.194	20.940	31.400	35.710	46.620	124.100
25	1.185	9.290	15.420	19.530	27.610	62.830
50	0.326	6.958	12.330	14.710	18.040	57.600
100	0.431	4.659	7.452	10.510	13.210	66.020
150	0.441	4.483	7.068	10.690	11.960	63.610
200	1.009	3.457	5.520	8.641	10.050	66.830

Table 4.5 Predictive accuracy in feet for the BR data set (off-line identification of spline coefficients and informative priors on X and Y).

predictive performance of the spline model. The $off-line/BSI$ model with "plug-in" coefficients shows performance that is comparable to the full Bayesian model with informative priors.

Convergence is an important issue for models that, as BS or BSI, contain hundreds of parameters. Instead of focusing on every single parameter in the model, an infeasible task when one is dealing with so many parameters, we will direct our attention to the estimations of xs and ys, as the most important proof of the validity of these models (BS, BSI, $off-line/BSI$) is their predictive performance.

Figure 4.5 shows important information about x and y estimations. There are three graphs in each column, the first column contains graphs that characterize $x[1]$ and $y[1]$ estimations on the data split of 200 training data points and 30 test points (BR data), the second columns shows graphs related to the estimations of $x[4]$ and $y[4]$. Time series of x and y (we used 1000 MCMC iterations) show that Markov chains mix well. Autocorrelations plots (figure 4.6) does not reveal any problems either. Although estimations only for two locations are given here, graphs for others (traceplots, time-series plots, and ACF plots) look similar.

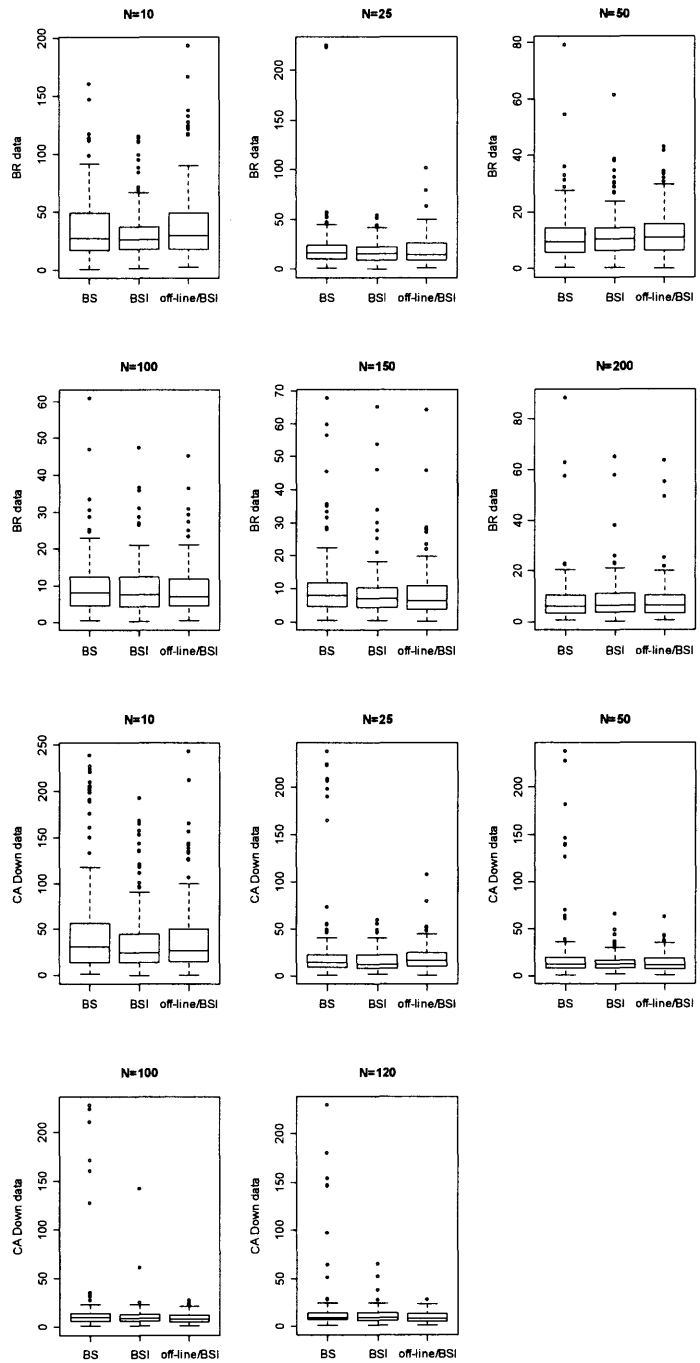


Figure 4.4 Comparison of all three models (BS, BSI, and $off-line/BSI$).

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	0.196	20.100	36.910	51.710	68.840	192.500
25	1.449	11.630	20.340	23.560	30.790	89.600
50	0.564	7.163	12.310	15.610	21.070	156.700
100	1.277	5.016	8.156	9.706	11.980	60.710
120	0.812	5.779	8.222	10.590	14.060	52.150

Table 4.6 Predictive accuracy in feet for the *CA Down* data set (bivariate spline model with off-line identification of spline coefficients and informative priors).

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	2.288	17.510	29.780	39.500	48.840	193.500
25	0.986	8.938	14.450	18.860	25.720	101.800
50	0.099	6.436	10.750	11.870	15.710	42.720
100	0.633	4.534	6.995	8.931	11.590	45.130
150	0.225	3.886	6.379	8.504	10.670	63.910
200	0.676	3.441	6.401	8.364	10.260	63.480

Table 4.7 Predictive accuracy in feet of the *BS* model ("plug-in"coefficients) with the exact starting points on the *BR* data.

The fact that the $off - line/BSI$ version of the BSI model shows predictive performance that is comparable to accuracy of the full Bayesian models (BS and BSI) increases out confidence in the validity of our models.

Another indirect way to test the model validity is to run it starting from the exact locations, i.e. to start MCMC sequences for X and Y in the exact locations (test data). Tables 4.7 and 4.8 show summaries of such simulations. For the both data sets results are close to the obtained above, and they do not reveal any discrepancies.

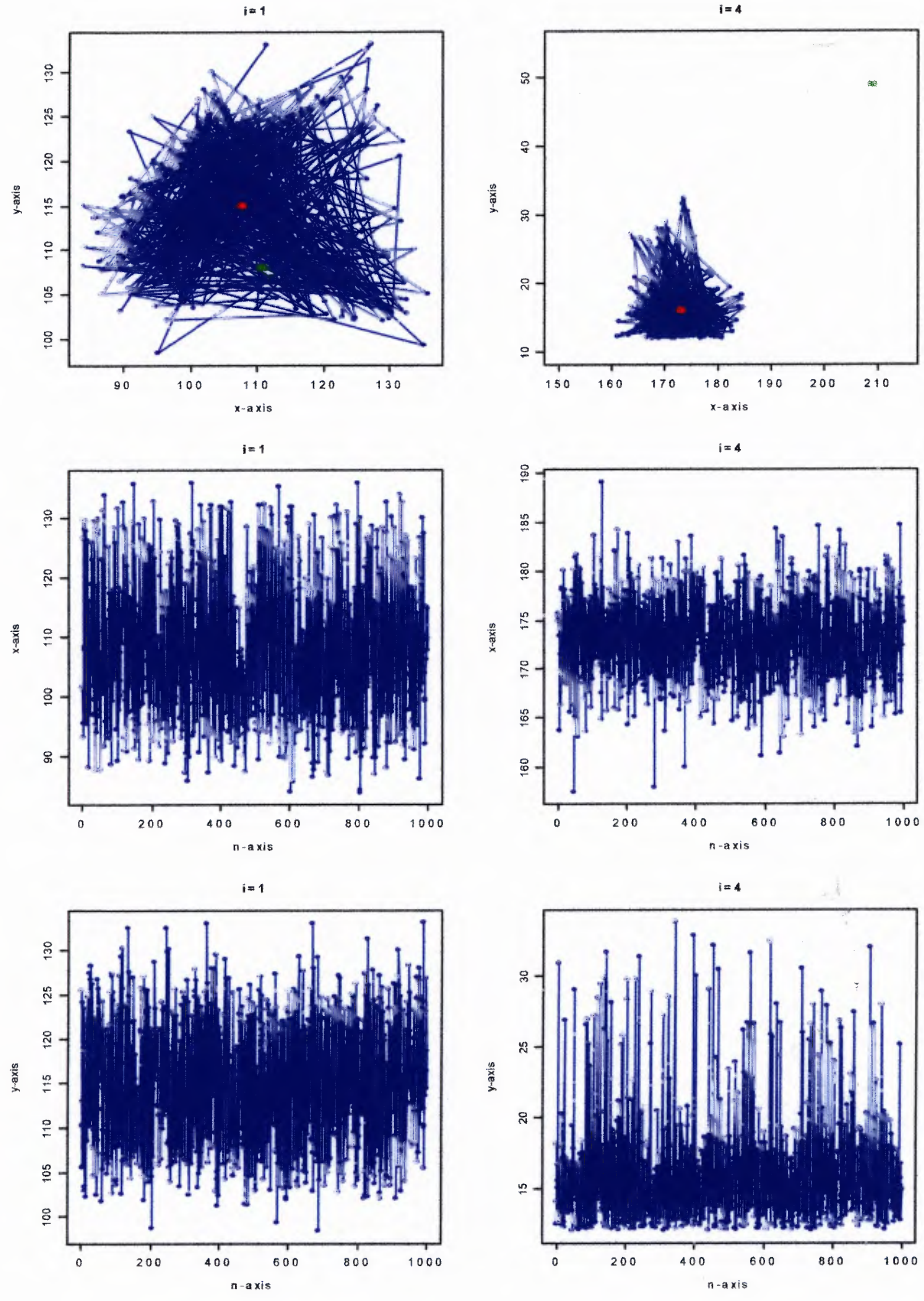


Figure 4.5 Last 1000 MCMC iterations 'track' plots and time-series plots for x and y. Red points represent estimated locations, green rectangles are actual locations.

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
10	0.236	15.690	26.620	41.140	49.710	241.000
25	0.919	10.510	16.640	19.270	25.060	107.100
50	0.470	7.196	11.060	13.930	18.620	62.320
100	0.918	5.187	8.453	9.168	11.930	26.740
120	0.882	5.654	8.339	9.726	12.870	27.850

Table 4.8 Predictive accuracy of the *BS* model ("plug-in" coefficients) on the *CA Down* site.

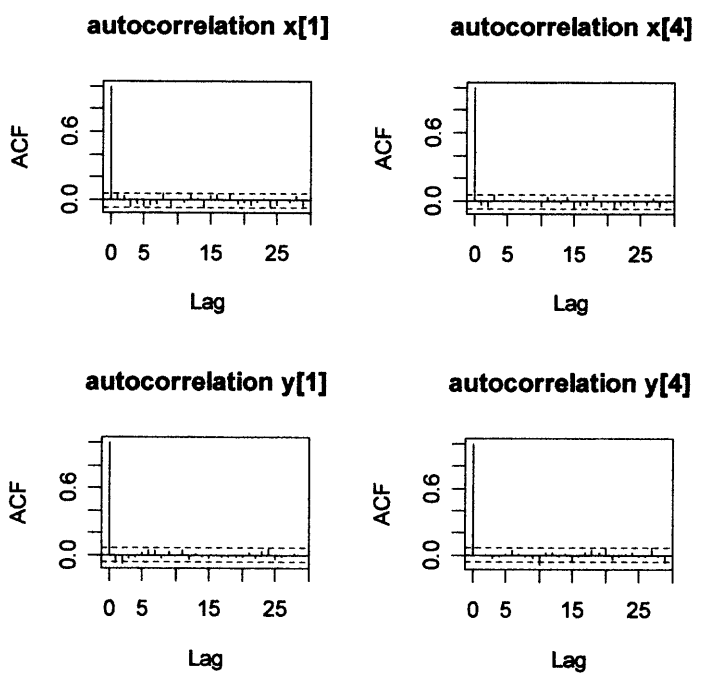


Figure 4.6 ACF plots for $x[1], y[1], x[4], y[4]$ (the *BR* data).

CHAPTER 5

ONLINE ESTIMATIONS AND MOVING OBJECTS

5.1 A Hybrid MCMC – Sampling Importance Resampling Method (*MCMC-SIR*)

In the previous chapter we introduced several versions of the spline graphical model for location. Although the latest version of this model allows us to find the location of a wireless device in seconds, rather than in hours, we would like to propose a particle implementation of the spline model. Particle version is suitable for the real-time calculations and, as one can see from the results below, it is comparable to or even surpasses all other methods (of course, one need to know the locations of all points in the training data set).

Let's consider a hybrid *MCMC-SIR* approach for the BR data. As in the case with the BSI model from chapter 4, we used the BS model for off-line identification of the spline coefficients (MCMC part to get parameter estomations), therefore all of them ($\beta 1 - \beta 5$, $b1 - b5$, $d1 - d5$, $s1 - s5$) are assumed to be known.

Then we are going to apply Sampling Importance Resampling algorithm for location estimation. We used particles sets (clouds) of different sizes M, $M \in \{50, 100, 500, 1000, 2000, 3000, 4000, 5000\}$.

Here is the $MCMC - SIR$ method for the BR data:

1. Generate a particle cloud of size M : $\{(x_i, y_i) \mid i = 1, 2, ..., M\}$,

where x_i~Uniform(XL, XR) and y_i~Uniform(YL, YR).

2. Using coefficients of the BS model from the off-line stage, calculate expected signal strengths $m1(x_i, y_i)$, $m2(x_i, y_i)$, $m3(x_i, y_i)$, $m4(x_i, y_i)$, $m5(x_i, y_i)$ for each of M particles.

3. Let's denote a vector of the received signal strength from the wireless object as q, $q = (q_1, q_2, q_3, q_4.q_5)$. For each particle j $(j = 1, 2, ..., M)$ from the particle cloud we calculate five weights w_{1j}, w_{2j}, w_{3j}, w_{4j}, w_{5j}, where

$$w_{kj} = \frac{1}{(2\pi)^{1/2}\sigma_k} \exp\left[-\frac{1}{2}\left(\frac{q_k - mk(x_i, y_i)}{\sigma_k}\right)^2\right], \tag{5.1}$$

σ_k, $k = 1 \div 5$, is a standard deviation calculated during the off-line stage of the algorithm. Expression 5.1 is derived using likelihoods from the BS model (see chapter 4).

4. Let's define w_j as a product of all five weights (by the number of access points), here we assume that received signal strengths for different access points are independent:

$$w_j = \quad w_{1j} \times w_{2j} \times w_{3j} \times w_{4j} \times w_{5j}, \tag{5.2}$$

and normalized weight $\overline{w}_j$ as

$$\overline{w}_j = \frac{w_j}{\sum_{j=1}^{M} w_j}. \tag{5.3}$$

5. Draw a sample of size M from the discrete distribution over $\{(x_1, y_1), (x_2, y_2), ..., (x_M, y_M)\}$ placing mass $\overline{w}_j$ on (x_j, y_j).The resampled cloud $\{(x_1^*, y_1^*), (x_2^*, y_2^*), ..., (x_M^*, y_M^*)\}$ represents an approximation to the posterior distribution of a location of the wireless device.

6. A point estimation of the location may be found by averaging x and y coordinates over the whole particle cloud:

$$\widehat{x} = \frac{\sum_{j=1}^{M} x_j^*}{M}, \widehat{y} = \frac{\sum_{j=1}^{M} y_j^*}{M}. \tag{5.4}$$

Figure 5.1 shows the predictive performance of model SIR on the BR data as a function of the size of particles cloud. Graphs of different colors show the performance of the algorithm on the training data sets of different sizes. As one might expect, predictive accuracy improves, as the training data size grows. The $M = 1000$ or 2000 is an 'optimal' number of particles to use, in the sense that with such M we can reach the maximum model accuracy still doing all calculations online.

Table 5.1 provides detailed results of the method performance when the cloud of 2000 particles is used.

Figures 5.3 shows boxplot summaries of all simulation results for the BR data. "N" denotes the number of points in the training data set, "M" is the number of particles in the $MCMC - SIR$ method.

Figure 5.2 shows the predictive performance (resolution) of model $MCMC - SIR$ on the *CA Down* data. As for the BR data, we investigated the model on the training data sets of different sizes (25, 50, 100, and 120 points) using different numbers of particles. As one can see from the graph, as in the case of the BR data, the model shows the best performance, when the number of particles in the $MCMC - SIR$ model equals to 1000 or 2000. Table 5.2 shows results for the cloud of 2000 particles. Figure 5.4 shows boxplot summaries for all combinations of the training data sizes and particles clouds on the *CA Down* data.

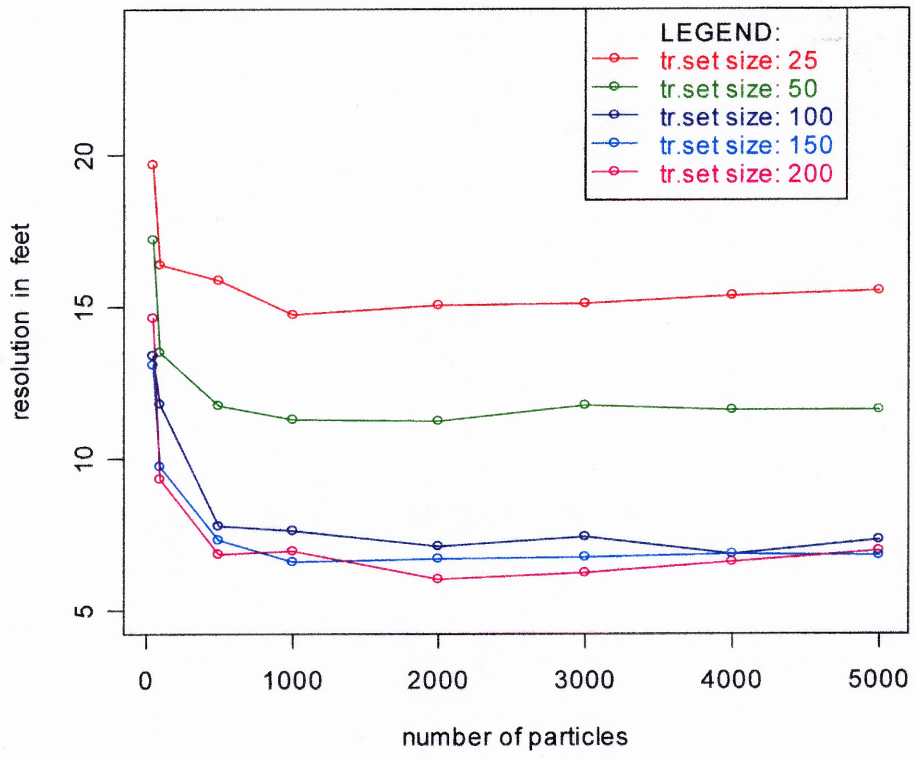


Figure 5.1 Resolution (50^{th} percentile) in feet of the $MCMC-SIR$ model for the BR data set.

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
25	0.517	8.782	15.060	19.140	27.450	77.820
50	0.532	6.764	11.240	13.130	16.850	56.640
100	0.7143	4.570	7.091	9.072	11.730	47.240
150	0.550	4.007	6.691	8.892	11.260	63.640
200	0.704	3.497	6.023	8.428	10.240	72.920

Table 5.1 Predictive accuracy in feet of the *MCMC-SIR* model on the BR data, M (number of particles)=2000.

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
25	2.284	11.390	18.440	21.310	27.530	83.000
50	0.559	7.239	11.044	14.380	18.980	60.280
100	0.5756	5.872	9.077	10.290	14.140	32.480
120	0.499	5.844	8.465	9.807	13.270	29.120

Table 5.2 Predictive accuracy of the *MCMC-SIR* model on the *CA down* site, M (number of particles)=2000.

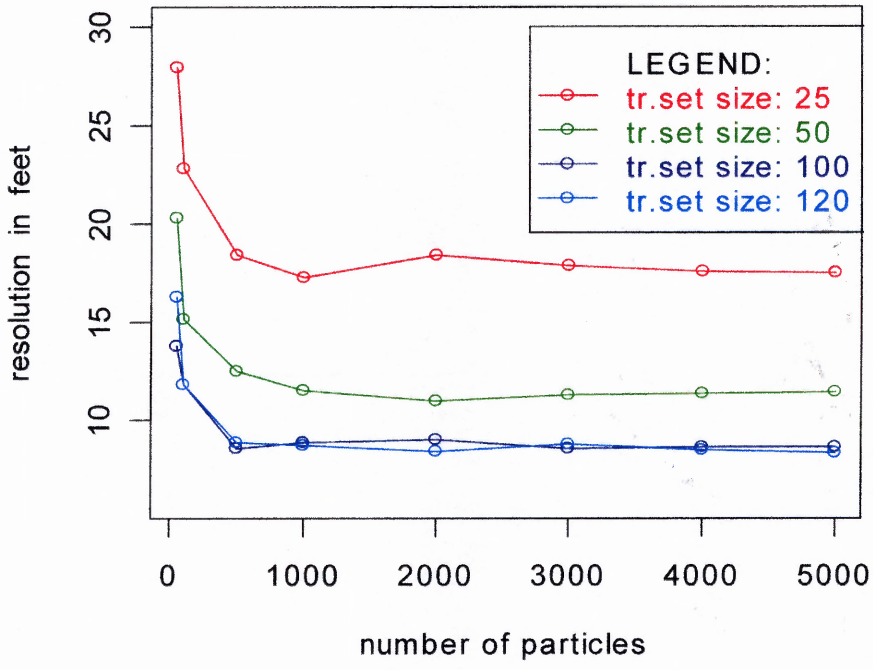


Figure 5.2 Resolution (50^{th} percentile) in feet of the $MCMC-SIR$ model for the *CA Down* data set

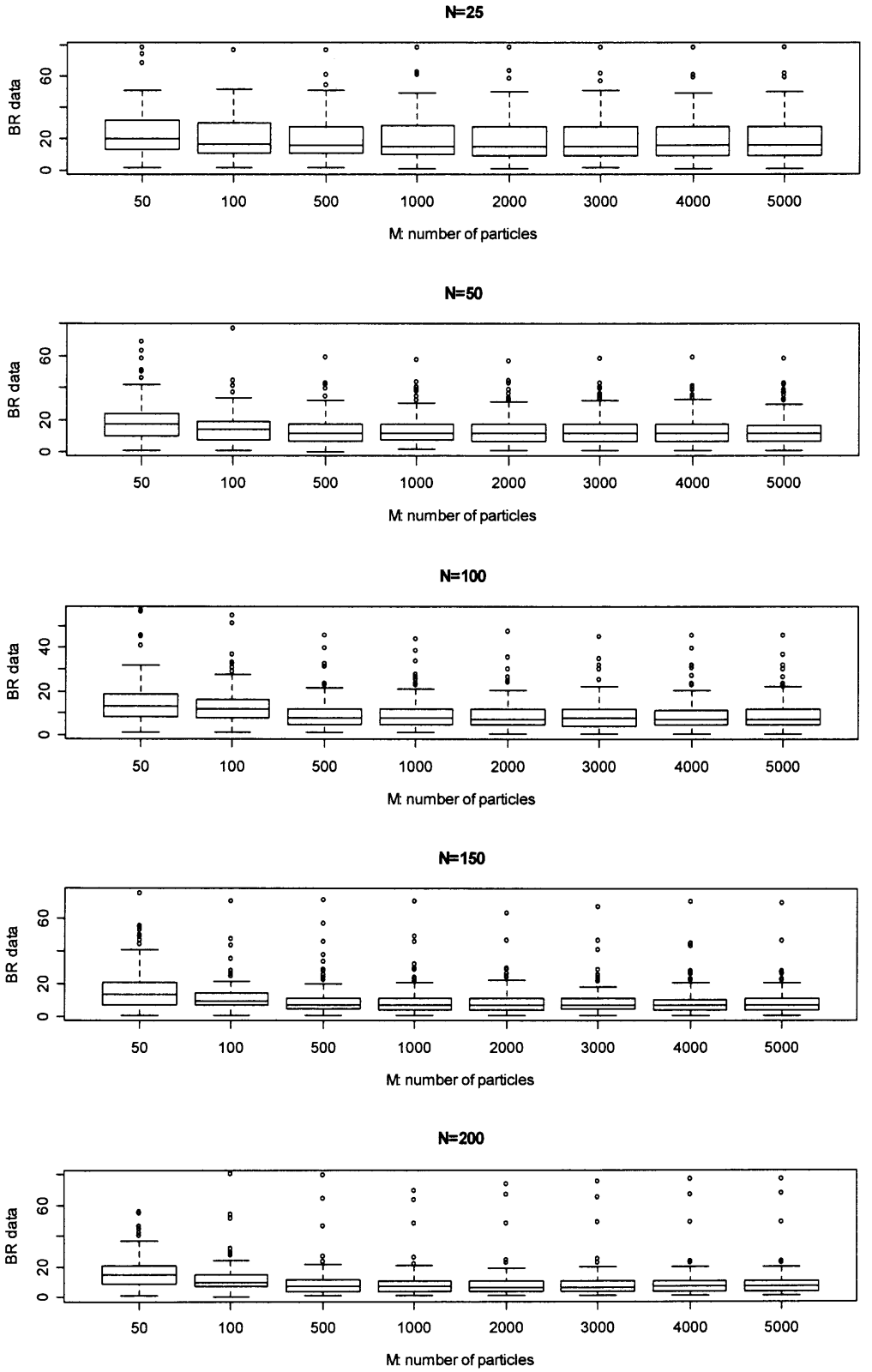


Figure 5.3 Predictive accuracy of the $MCMC - SIR$ model on the BR data.

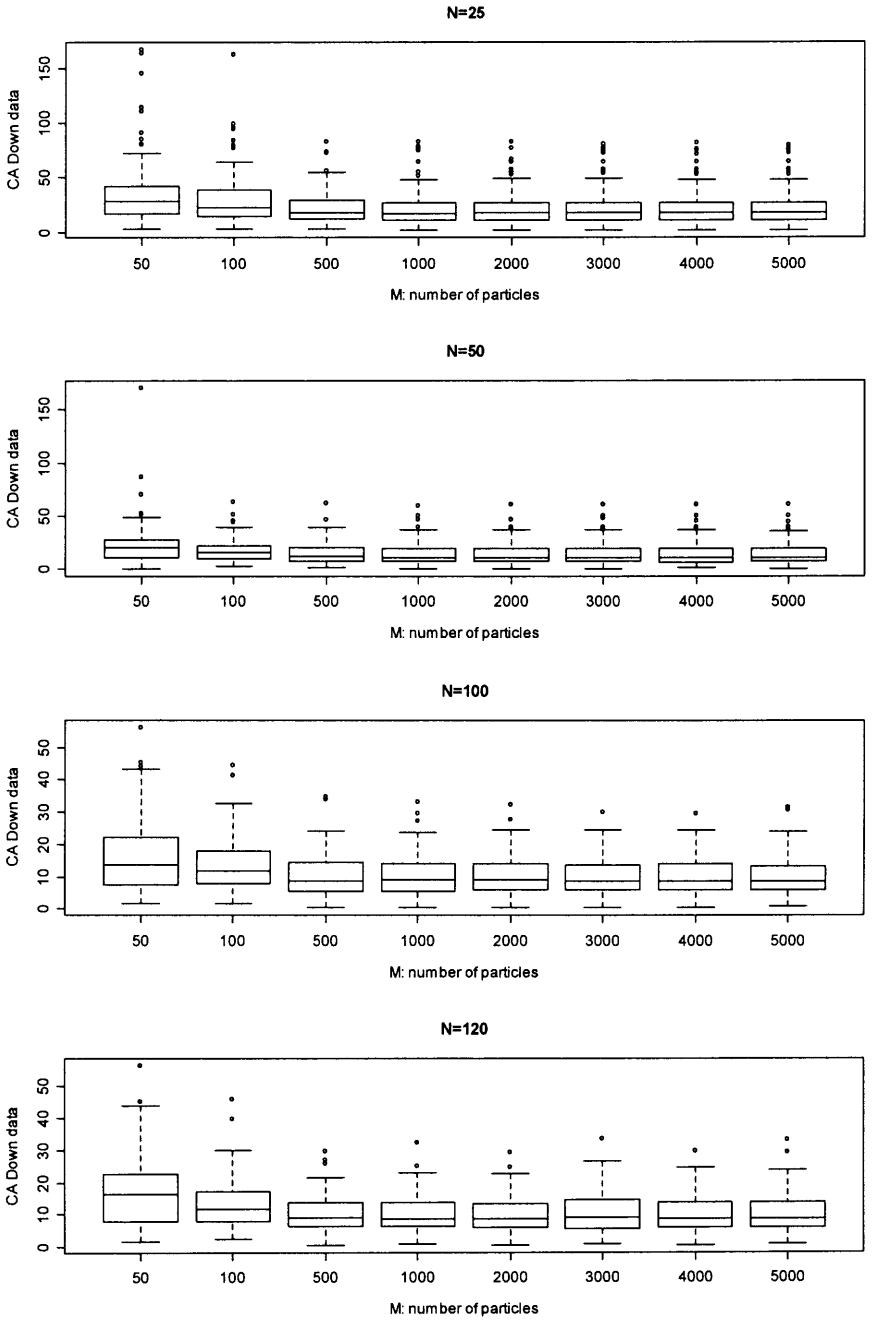


Figure 5.4 Predictive accuracy of the $MCMC-SIR$ model on the $CA\ Down$ data

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
25	0.498	8.062	13.980	17.580	26.780	70.660
50	0.910	6.833	11.140	13.240	16.650	56.670
100	0.698	4.270	7.253	8.601	11.510	39.070
150	0.150	3.950	6.004	9.468	11.350	61.720
200	0.430	3.167	5.228	7.912	9.337	61.870

Table 5.3 Model *MCMC-SIRI* (MCMC Sampling Importance Resampling with informative particle clouds for the *BR* data, M (number of particles)=2000.

5.2 *MCMC-SIR* Method with Informative Prior Clouds (*MCMC - SIRI* Method)

In the previous chapter we introduced the BSI model that used simple regressions (IR model) to put informative priors on distribution of location parameters X and Y. Here we present the particle version of this method.

In the $MCMC - SIR$ method from the previous section of this chapter, prior cloud was distributed uniformly in the interior of the building floor. For the *CA Down* data set it meant that the prior distribution followed a quite complicated geometry of the building (see figure 5.5).

To generate an informative particle cloud we used the IR model with regression coefficients that had been identified off-line (beforehand), then the $MCMC - SIR$ method was applied to an informative cloud of particles. Figure 5.6 shows the performance of this method on the *BR* data, figure 5.7 shows resolution graphs for the *CA Down* data. One can see that the smaller number, $M = 500$, of particles is needed to reach the maximum level of accuracy.

Table 5.3 shows detailed results for the *BR data* when $M = 2000$, table 5.4 gives the same information for the *CA Down* data set. Comparing to the SIR method, informative priors allow to reach the highest level of accuracy using the

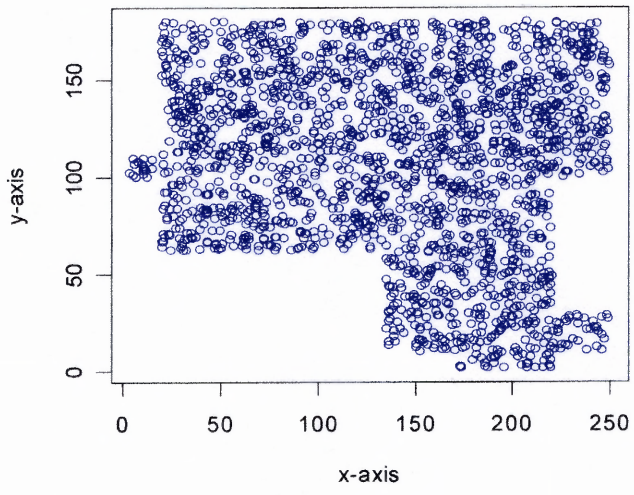


Figure 5.5 Particle cloud that represents a noninformative prior distribution (*CA Down* data).

Size of the Tr. Set	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
25	1.641	11.250	17.440	20.430	26.150	73.640
50	0.507	7.374	12.060	15.210	20.090	53.920
100	0.200	5.148	9.746	10.020	13.240	30.390
120	0.360	6.169	8.656	10.500	14.060	38.040

Table 5.4 Model *MCMC-SIR* with informative particle clouds for the *CA Down* data, M (number of particles)=2000.

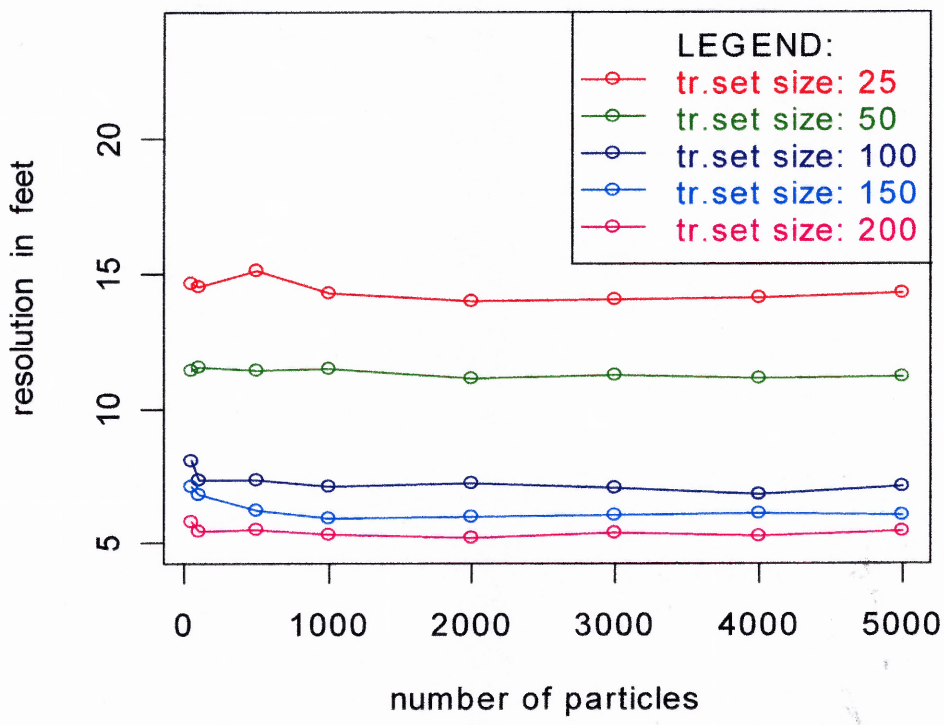


Figure 5.6 Resolution (50^{th} percentile) in feet of the $MCMC - SIRI$ model (MCMC & Sampling Importance Resampling with Informative Prior Clouds) for the *BR* data set.

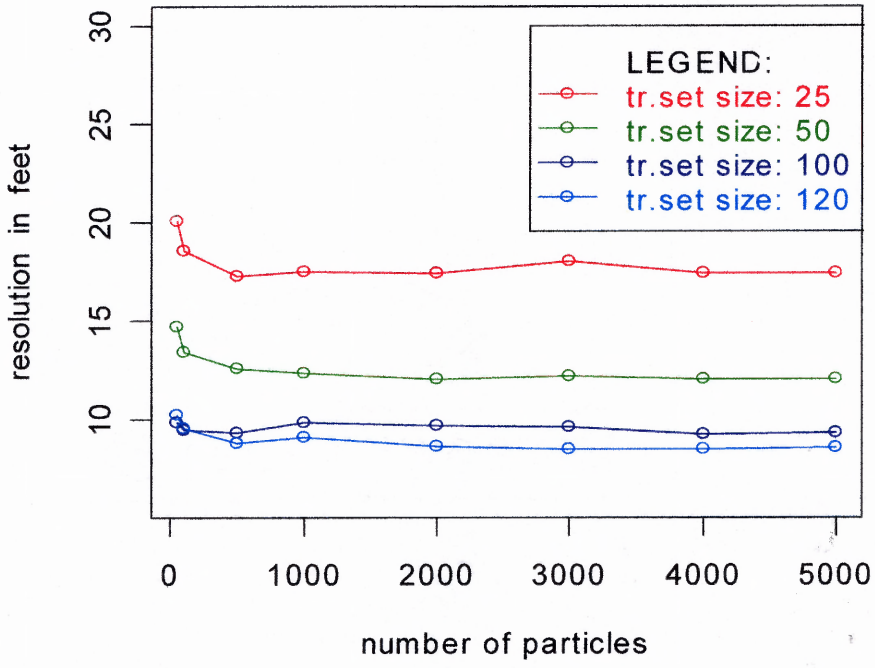


Figure 5.7 Resolution (50^{th} percentile) in feet of the $MCMC - SIRI$ model (Sampling Importance Resampling with Informative Prior Clouds) for the *CA Down* data set.

smaller number of particles, they also effectively restrict maximum predictive error. Figures 5.8 and 5.9 show boxplot summaries that characterize the performance of the $MCMC - SIRI$ approach on the *BR data* and *CA Down* data respectively.

5.3 Moving Objects

So far we have considered a problem of locating stationary (non-moving) objects. Although we do not have data that describes real moving object, we can create such a data set from the existing data. From a subset of the *BR* data of size 240, we picked up 22 points that form a path. Figure 5.10 shows the training data set (blue points) and a selected path (red). There are several ways to track a moving object in the real time. We will explore two of them: the *MCMC-SIR* approach and the combination of $MCMC$ (off-line estimation of the spline coefficients) and a *Particle Filter*.

One may apply the $MCMC - SIR$ method to track a mobile object. Table 5.5 shows the performance of the SIR method fir the selected path. Green line on Figure 5.11 shows the $MCMC - SIR$ estimation of the path (depicted in red).

Unlike the $MCMC - SIR$ method, that estimates every new point independently from all others, *MCMC–Particle Filter* is getting information from the estimations made in previous moments.

The state equation, that describes the motion pattern, is playing an important role in the *Particle Filter* part. We used the following:

$$x[k] = x[k-1] + velocity_x[k] \times \xi[k], \tag{5.5}$$

$$y[k] = y[k-1] + velocity_y[k] \times \eta[k], \tag{5.6}$$

$$velocity_x[k] = x[k-1] - x[k-2],\ velocity_x[0] = 15, \tag{5.7}$$

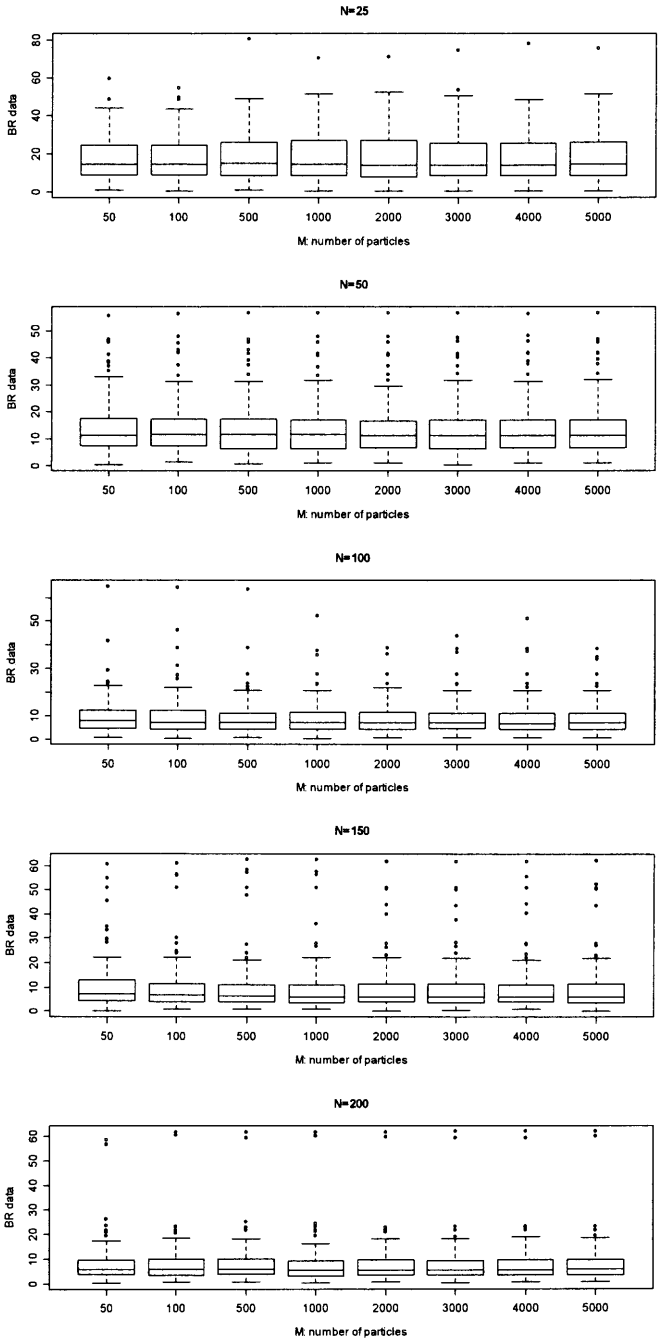


Figure 5.8 Predictive accuracy of the $MCMC - SIRI$ model ($MCMC - SIR$ with informative prior cloud) on the BR data.

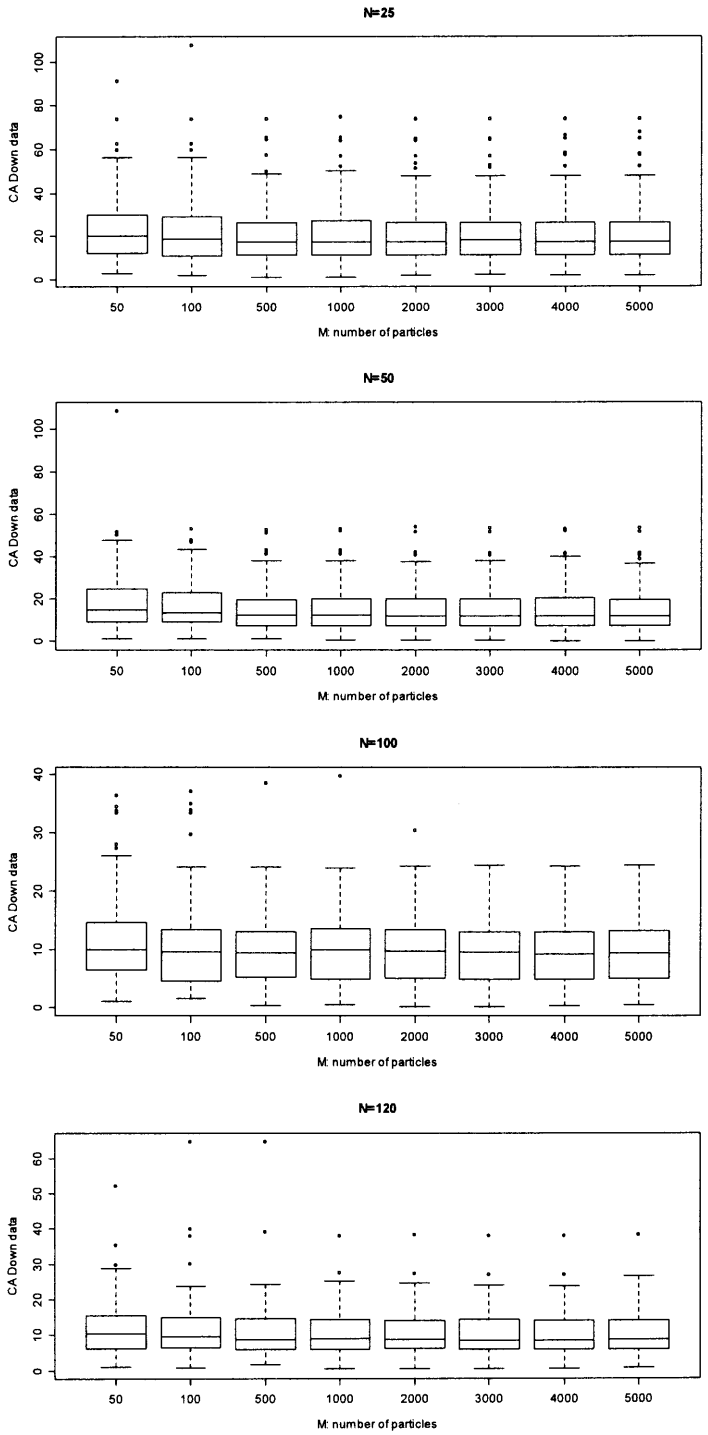


Figure 5.9 Predictive accuracy of the $MCMC - SIRI$ model on the *CA Down* data.

Method	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
$MCMC - SIR$	1.022	4.407	7.851	9.407	13.510	28.450
$MCMC - Particle\ Filter$	1.136	5.706	8.605	8.200	11.220	14.110

Table 5.5 Comparison of the *MCMC-SIR* method and the *MCMC-Particle Filter*, M (number of particles in the particles cloud)=2000.

$$velocity_y[k] = y[k-1] - y[k-2],\ velocity_y[0] = 15, \tag{5.8}$$

$$\xi[k] \sim \text{Normal}(0, \sigma_x^2),\ \sigma_x = 0.4, \tag{5.9}$$

$$\eta[k] \sim \text{Normal}(0, \sigma_y^2),\ \sigma_y = 0.4. \tag{5.10}$$

Table 5.5 shows the results for the *MCMC–Particle Filter* approach. Blue line in Figure 5.11 shows the estimated path. One can see that blue line is a quite good approximation to the actual path, the *MCMC– Particle Filter* approach overperforms the $MCMC - SIR$ method.

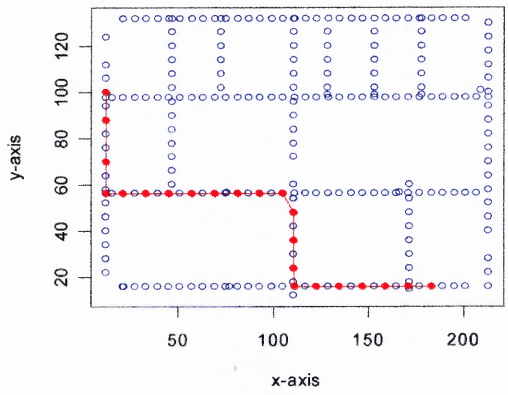


Figure 5.10 Locating a moving object. Blue points belong to the training data set of 218 points. Red points belong to the test set and form a path of the moving object (red points).

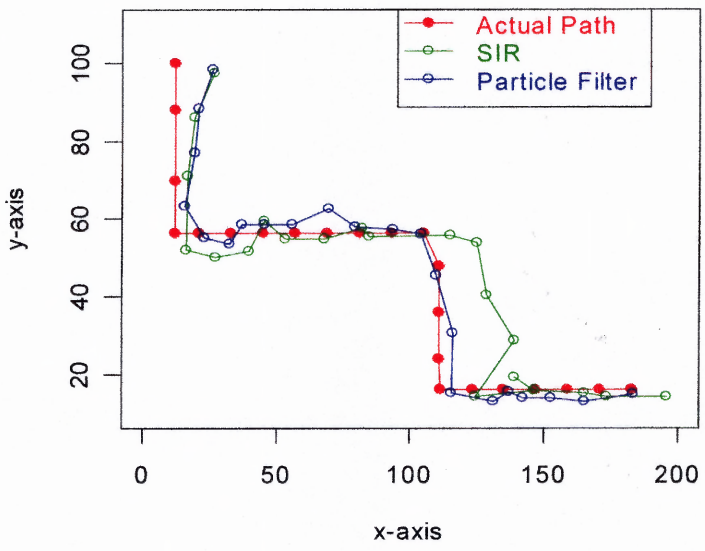


Figure 5.11 Comparison of the $MCMC-SIR$ method and the $MCMC-Particle\ Filter$ for mobile tracking.

CHAPTER 6

CONCLUSIONS

In this work, several approaches for indoor location estimation in wireless networks are developed. These include non-hierarchical and hierarchical Bayesian graphical models that use information about locations of the access points and knowledge about physics of signal propagation (log-loss model), different modifications of Bayesian bivariate spline models, including models that utilize a "plug-in" approach (substitution a point estimate of the unknown multidimensional parameter into predictive distribution), and combinations of spline models and sampling importance resampling algorithm. We also propose a combination of Bayesian bivariate spline model and a particle filter for tracking of a moving object.

The hierarchical Bayesian graphical model (chapter 3), that utilizes similarities between nodes, possesses a remarkable property. It is capable of predicting location without using any location information in the training data set.

Bivariate spline models (chapter 4) improve the accuracy of predictions and do not use information about location of the access points. As these models show, it often makes sense to fit a Bayesian hierarchical model with more parameters than there are data points in the training set.

Along with the full Bayesian version of the bivariate spline model, we consider "plug-in" model that uses spline coefficients and other model parameters that have been calculated off-line. Such "plug-in" version performs similarly to the full Bayesian model while using hundreds of off-line estimated parameters.

To reduce maximum prediction error, we propose a two-stage approach for location estimation. At stage one, we use a simple Bayesian model (two-independent regressions) to get some information about locations. At the second stage, we use this

information in the form of tighter bounds in the uniform priors of unknown locations in the Bayesian spline model.

For the real-time estimations, we propose a combination of the Bayesian spline model and the sampling importance resampling algorithm (chapter 5). The prediction performance of such a model is comparable or even surpasses the full Bayesian model with informative priors. A modification of such an approach, that incorporates a particle filter, outperforms *MCMC-SIR* model when applied for tracking of a moving object.

It is interesting to note that location problem in wireless networks is an example of ill-posed problems. Different models proposed in this work may be considered as Bayesian graphical solutions for the ill-posed problem. We think that similar Bayesian approaches may be applied to other ill-posed problems, beyond the scope of location estimation in wireless networks.

There are several directions for future work:

- Investigating the splines of higher degree;
- Exploring different spline basis (kriging, etc.);
- Incorporating of other data pertaining to the signal such as angle of arrival;
- Developing Bayesian models capable of tracking several wireless object simultaneously.

APPENDIX A

WinBUGS CODES, PLOTS

A.1 WinBUGS Code for the IR Model

```
# Two independent Regressions: Model IR }
    model
    {
    for( i in 1:N) {
    x[i]~dnorm(mx[i],taux)
    mx[i]<-a[1]+a[2]*p1[i]+a[3]*p2[i]+a[4]*p3[i]+a[5]*p4[i]+a[6]*p5[i]
    y[i]~dnorm(my[i],tauy)
    my[i]<-b[1]+b[2]*p1[i]+b[3]*p2[i]+b[4]*p3[i]+b[5]*p4[i]+b[6]*p5[i]
    }
    for (j in 1:6){
    a[j]~dnorm(0,0.001)
    b[j]~dnorm(0,0.001)
    }
    taux~dgamma(0.001,0.001)
    tauy~dgamma(0.001,0.001)
    }
    #   end of the IR model
```

A.2 WinBUGS Code for the Multivariate Regression Model

```
model # Multivariate Regression: Model MR
    {
    for (i in 1:N) {
    x[i,1:2]~dmnorm(mu[i,],T[,])
```

```
mu[i,1]<-a[1]+a[2]*p1[i]+a[3]*p2[i]+a[4]*p3[i]+a[5]*p4[i]+a[6]*p5[i]
mu[i,2]<-b[1]+b[2]*p1[i]+b[3]*p2[i]+b[4]*p3[i]+b[5]*p4[i]+b[6]*p5[i]
}
T[1:2,1:2] ~dwish(D[,], 2)
for (j in 1:6){
a[j]~dnorm(0,0.001)
b[j]~dnorm(0,0.001)
}
}#  end of the MR model
```

A.3 WinBUGS Code for the non-Hierarchical Bayesian Model (M_1)

```
model
{
        for( i in 1 : N ) {

            p1[i]~dnorm(m1[i],tau1)
            p2[i]~dnorm(m2[i],tau2)
            p3[i]~dnorm(m3[i],tau3)
            p4[i]~dnorm(m4[i],tau4)
            p5[i]~dnorm(m5[i],tau5)

            m1[i]<- b01 + b11*log(1+d1[i])
            m2[i]<- b02 + b12*log(1+d2[i])
            m3[i]<- b03 + b13*log(1+d3[i])
            m4[i]<- b04 + b14*log(1+d4[i])
            m5[i]<- b05 + b15*log(1+d5[i])
```

```
                d1[i]<-sqrt(pow(x[i]-19,2)+pow(y[i]-49,2))
                d2[i]<-sqrt(pow(x[i]-213,2)+pow(y[i]-34,2))
                d3[i]<-sqrt(pow(x[i]-206,2)+pow(y[i]-120,2))
                d4[i]<-sqrt(pow(x[i]-12,2)+pow(y[i]-118,2))
                d5[i]<-sqrt(pow(x[i]-110,2)+pow(y[i]-66,2))
                x[i]~dunif(0,L)
                y[i]~dunif(0,B)
                        }
tau1~dgamma(0.001,0.001)
tau2~dgamma(0.001,0.001)
tau3~dgamma(0.001,0.001)
tau4~dgamma(0.001,0.001)
tau5~dgamma(0.001,0.001)

b01 ~dnorm(0,0.001)
b02 ~dnorm(0,0.001)
b03 ~dnorm(0,0.001)
b04 ~dnorm(0,0.001)
b05 ~dnorm(0,0.001)
b11~dnorm(0,0.001)
b12~dnorm(0,0.001)
b13~dnorm(0,0.001)
b14~dnorm(0,0.001)
b15~dnorm(0,0.001)
}
```

A.4 WinBUGS Code for the Hierarchical Bayesian Model (M_2)

```
model
    {
        for( i in 1 : N ) {
            p1[i]~dnorm(m1[i],tau1)
            p2[i]~dnorm(m2[i],tau2)
            p3[i]~dnorm(m3[i],tau3)
            p4[i]~dnorm(m4[i],tau4)
            p5[i]~dnorm(m5[i],tau5)

            m1[i]<- b01 + b11*log(1+d1[i])
            m2[i]<- b02 + b12*log(1+d2[i])
            m3[i]<- b03 + b13*log(1+d3[i])
            m4[i]<- b04 + b14*log(1+d4[i])
            m5[i]<- b05 + b15*log(1+d5[i])

            d1[i]<-sqrt(pow(x[i]-19,2)+pow(y[i]-49,2))
            d2[i]<-sqrt(pow(x[i]-213,2)+pow(y[i]-34,2))
            d3[i]<-sqrt(pow(x[i]-206,2)+pow(y[i]-120,2))
            d4[i]<-sqrt(pow(x[i]-12,2)+pow(y[i]-118,2))
            d5[i]<-sqrt(pow(x[i]-110,2)+pow(y[i]-66,2))
                }
        tau1~dgamma(0.001,0.001)
        tau2~dgamma(0.001,0.001)
        tau3~dgamma(0.001,0.001)
        tau4~dgamma(0.001,0.001)
        tau5~dgamma(0.001,0.001)
```

```
    b01 ~dnorm(b0,taub0)
    b02 ~dnorm(b0,taub0)
    b03 ~dnorm(b0,taub0)
    b04 ~dnorm(b0,taub0)
    b05 ~dnorm(b0,taub0)

    b0~dnorm(0,0.001)
    taub0~dgamma(0.001,0.001)

    b11~dnorm(b1,taub1)
    b12~dnorm(b1,taub1)
    b13~dnorm(b1,taub1)
    b14~dnorm(b1,taub1)
    b15~dnorm(b1,taub1)

    b1~dnorm(0,0.001)
    taub1~dgamma(0.001,0.001)
}
```

A.5 WinBUGS Code for the Bayesian Spline Model (BS)

```
model
    {
    for( i in 1 : N ) {
    p1[i]~dnorm(m1[i],taueps[1])
    p2[i]~dnorm(m2[i],taueps[2])
```

```
p3[i]~dnorm(m3[i],taueps[3])
p4[i]~dnorm(m4[i],taueps[4])
p5[i]~dnorm(m5[i],taueps[5])
for(k in 1:KX) {
 u[i,k]<-(x[i]-kx[k])*step(x[i]-kx[k])
aa1[i,k]<-b1[k]*u[i,k]
aa2[i,k]<-b2[k]*u[i,k]
aa3[i,k]<-b3[k]*u[i,k]
aa4[i,k]<-b4[k]*u[i,k]
aa5[i,k]<-b5[k]*u[i,k]
}
for(k in 1:KY) {
v[i,k]<-(y[i]-ky[k])*step(y[i]-ky[k])
bb1[i,k]<-d1[k]*v[i,k]
bb2[i,k]<-d2[k]*v[i,k]
bb3[i,k]<-d3[k]*v[i,k]
bb4[i,k]<-d4[k]*v[i,k]
bb5[i,k]<-d5[k]*v[i,k]
}
for(k in 1:KXY) {
z[i,k]<-(x[i]-xs[k])*step(x[i]-xs[k])*(y[i]-ys[k])*step(y[i]-ys[k])
cc1[i,k]<-s1[k]*z[i,k]
cc2[i,k]<-s2[k]*z[i,k]
cc3[i,k]<-s3[k]*z[i,k]
cc4[i,k]<-s4[k]*z[i,k]
cc5[i,k]<-s5[k]*z[i,k]
}
```

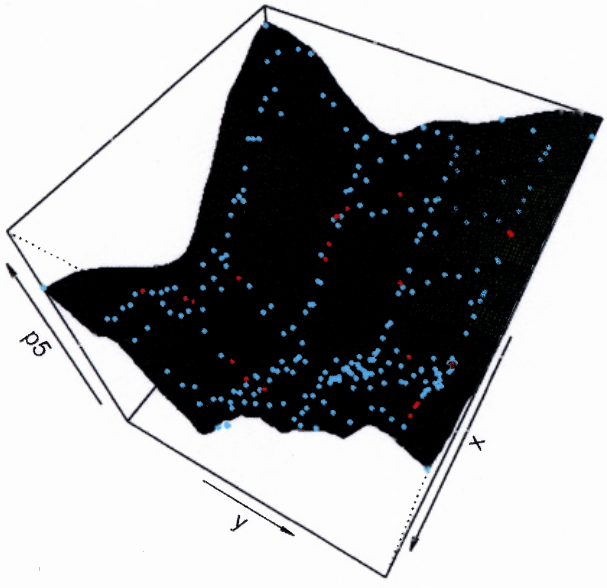
```
aaa1[i]<-sum(aa1[i,])
bbb1[i]<-sum(bb1[i,])
ccc1[i]<-sum(cc1[i,])
aaa2[i]<-sum(aa2[i,])
bbb2[i]<-sum(bb2[i,])
ccc2[i]<-sum(cc2[i,])
aaa3[i]<-sum(aa3[i,])
bbb3[i]<-sum(bb3[i,])
ccc3[i]<-sum(cc3[i,])
aaa4[i]<-sum(aa4[i,])
bbb4[i]<-sum(bb4[i,])
ccc4[i]<-sum(cc4[i,])
aaa5[i]<-sum(aa5[i,])
bbb5[i]<-sum(bb5[i,])
ccc5[i]<-sum(cc5[i,])
# for AP1:
m1[i]<-mfe1[i]+mre1[i]
mfe1[i]<-beta1[1]+beta1[2]*x[i]+beta1[3]*y[i]+beta1[4]*x[i]*y[i]
mre1[i]<-aaa1[i]+bbb1[i]+ccc1[i]
#end of m1
# for AP2:
m2[i]<-mfe2[i]+mre2[i]
mfe2[i]<-beta2[1]+beta2[2]*x[i]+beta2[3]*y[i]+beta2[4]*x[i]*y[i]
mre2[i]<-aaa2[i]+bbb2[i]+ccc2[i]
#end of m2
# for AP3:
m3[i]<-mfe3[i]+mre3[i]
```

```
mfe3[i]<-beta3[1]+beta3[2]*x[i]+beta3[3]*y[i]+beta3[4]*x[i]*y[i]
mre3[i]<-aaa3[i]+bbb3[i]+ccc3[i]
#end of m3
# for AP4:
m4[i]<-mfe4[i]+mre4[i]
mfe4[i]<-beta4[1]+beta4[2]*x[i]+beta4[3]*y[i]+beta4[4]*x[i]*y[i]
mre4[i]<-aaa4[i]+bbb4[i]+ccc4[i]
#end of m4
# for AP5:
m5[i]<-mfe5[i]+mre5[i]
mfe5[i]<-beta5[1]+beta5[2]*x[i]+beta5[3]*y[i]+beta5[4]*x[i]*y[i]
mre5[i]<-aaa5[i]+bbb5[i]+ccc5[i]
#end of m5
x[i] ~dunif(XL, XR)
y[i] ~dunif(YL, YR)
}
#Prior distributions of the random effects parameters
for (k in 1:KX)
{
b1[k]~dnorm(b,taub)
b2[k]~dnorm(b,taub)
b3[k]~dnorm(b,taub)
b4[k]~dnorm(b,taub)
b5[k]~dnorm(b,taub)
}
for (k in 1:KY)
{
```

```
d1[k]~dnorm(d,taud)
d2[k]~dnorm(d,taud)
d3[k]~dnorm(d,taud)
d4[k]~dnorm(d,taud)
d5[k]~dnorm(d,taud)
}
for (k in 1:KXY)
{
s1[k]~dnorm(s,taus)
s2[k]~dnorm(s,taus)
s3[k]~dnorm(s,taus)
s4[k]~dnorm(s,taus)
s5[k]~dnorm(s,taus)
}
#Prior distribution of the fixed effects parameters
for (j in 1:4)
{
beta1[j]~dnorm(bet,taubet)
beta2[j]~dnorm(bet,taubet)
beta3[j]~dnorm(bet,taubet)
beta4[j]~dnorm(bet,taubet)
beta5[j]~dnorm(bet,taubet)
}
taubet~dgamma(0.001,0.001)
bet~dnorm(0,0.001)
b~dnorm(0,0.001)
taub~dgamma(0.001,0.001)
```

```
d~dnorm(0,0.001)

taud~dgamma(0.001,0.001)

s~dnorm(0,0.001)

taus~dgamma(0.001,0.001)

#Prior distributions of the precision parameter

for(i in 1:5)

{

taueps[i]~dgamma(0.001,0.001)

}

} # end of the model
```

A.6 Spline Model, 3D Plot for AP 5 (*BR* Data)



Graph illustrates how the *BS* model works on the *BR* data (AP 5 is shown). Green surface is generated by spline. Blue points represent the training data set, red points belong to the test data.

REFERENCES

[1] P. Bahl and V.N. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," *Proceedings of IEEE Infocom 2000*, Tel Aviv, Israel, March 2000.

[2] R. Battiti, M. Brunato, and A. Villani, "Statistical learning theory for location fingerprinting in wireless LANs," Dipartimento di Informatica e Telecomunicazioni, Universita di Trento, *Technical Report DIT-020086*, 2002.

[3] M. Berna, B. Lisien, B. Sellner, G. Gordon, F. Pfenning, and S. Thurn, "A learning algorithm for localizing people on wireless signal strength that uses labeled and unlabeled data," *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico. 2003.

[4] B. Brumback, D. Ruppert, and M.P. Wand, "Comments on a variable selection and function estimation in additive nonparametric regression using data-based prior by Shively, Kohn, and Wood," *Journal of the American Statistical Association*, vol. 94, pp. 794-797, 1999.

[5] T.W. Christ and P.A. Godwin, "A Prison guard duress alarm location system," *Proceedings of the IEEE International Carnahan Conference on Security Technology*, October 1993.

[6] C.M. Crainiceanu, D. Ruppert, and R.J. Carroll, "Spacially adaptive Bayesian P-splines with heteroscedastic errors," Johns Hopkins University, *Dept. of Biostatistics Working Papers*, paper 61, 2004.

[7] A. Doucet, "On sequential simulation-based methods for Bayesian filtering," *Technical report CUED/F-INFENG/TR 310,* Cambridge University Department of Engineering, 1998.

[8] B. Efron, The Bootstrap, *Jackknife and Other Resampling Plans.* Philadelphia: Society of Industrial and Applied Mathematics, 1982.

[9] A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin, *Bayesian Data Analysis.* Chapman and Hall/CRC, 2003.

[10] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images," *IEEE Trans. Pattn. Anal. Mach. Intell.*, vol. 6, pp.721-741, 1984.

[11] W.R. Gilks, S. Richardson, D.J. Spiegelhalter, *Markov Chain Monte Carlo in Practice.* Chapman and Hall/CRC, 1996.

[12] J. Gill, Bayesian Methods. *A Social and Behavioral Science Approach.* Chapman & Hall/CRC, 2002.

[13] N.J. Gordon, D.J. Salmond, and A.F.M. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," *IEE Proceeding-F,* vol. 140, pp. 107-113, 1993.

[14] P. Krishnan, A.S. Krishnakumar, W. Ju, C. Mallows, and S. Ganu, "A system for lease: system location estimation assisted by stationary emiters for indoor RF wireless networks," *IEEE Infocom,* Hong Kong, March 7-11, 2004.

[15] A.M. Ladd, K.E. Bekris, A. Rudys, G. Marceau, L.E. Kavraki, and S. Dan, "Robotics-based location sensing using wireless ethernet," *Proceedings of the Eighth International Conference on Mobile Computing and Networking (MOBICOM),* Atlanta, GA, September 2002.

[16] N.M. Laird and J.H. Ware, "Random-effects models for longitudinal data," *Biometrics,* vol. 38, pp. 963-974, 1982.

[17] S.L. Lauritzen, *Graphical Models.* Oxford University Press, 1996.

[18] D.V. Lindley, "The estimation of many parameters," in Foundation of Statistical Inference, Toronto, 1970

[19] J.S. Liu and R. Chen, "Sequential Monte Carlo methods for dynamic systems," *Journal of the American Statistical Association,* vol. 93, pp. 1032–1044, 1998.

[20] D. Madigan and J. York, "Bayesian graphical models for discrete data," International Statistical Review, vol. 63, pp. 215-232, 1995.

[21] R. H. Myers and D. C. Montgomery, *Response Surface Methodology. Process and Product Optimization Using Designed Experiments.* Second edition, Wiley-Interscience, 2002.

[22] M. Hassan-Ali and K. Pahlavan, "A new statistical model for site-specific indoor radio propagation prediction based on geometric optics and geometric probability," *IEEE Transactions on Wireless Communications,* vol. 1, no. 1, January 2002.

[23] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer, 2001.

[24] D. Heckerman, D. Geiger, and D. Chickering, "Learning bayesian networks: the combination of knowledge and statistical data," *Machine Learning,* vol. 20, pp. 197-243, 1995.

[25] A. Howard, S. Siddiqi, and G.S. Sukhatme, "An experimental study of localization using wireless ethernet," *The 4th International Conference on Field and Service Robotics,* July 14-16, 2003.

[26] G.T. Huang, "Casting the wireless sensor net," *Technology Review, MIT's Magazine of Innovation,* July/August issue, pp. 51-56, 2003.

[27] P. Prasithsangaree, P. Krishnamurthy, and P.K. Chrysanthis, "On indoor position location with wireless LANs," *The 13th IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC 2002)*, Lisbon, Portugal, September 2002.

[28] S.J. Press, *Subjective and Objective Bayesian Statistics: Principles, Models, and Applications.* J. Wiley, 2003.

[29] N.B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location support system," *Proceedings of the Sixth Annual ACM International Conference on Mobile Computing and Networking,* Boston, MA, August 2000.

[30] B. Ripley, *Stochastic Simulation.* J. Wiley, 1987

[31] G.K. Robinson, "That BLUP is a good thing: the estimation of random effects," *Statistical Science*, vol. 6, pp. 15-51, 1991.

[32] T. Roos, P. Myllymaki, and H. Tirri, " A statistical modeling approach to location estimation," *IEEE Transactions on Mobile Computing*, vol. 1, pp. 59–69, 2002.

[33] D.B. Rubin, "Using the SIR algorithm to simulate posterior distributions," *Bayesian Statistics 3*, Oxford University Press, pp. 395-402, 1988.

[34] D. Ruppert, "Selecting the number of knots for penalized splines," *Journal of Computational and Graphical Statistics*, vol. 11, pp. 735-757, 2002.

[35] D. Ruppert, M.P. Wand, and R.J. Carroll, *Semiparametric Regression.* Cambridge University Press, 2003.

[36] S. Saha, K. Chaudhuri, D. Sanghi, and P. Bhagwat, "Location determination of a mobile device using IEEE 802.11 access point signals," *IEEE Wireless Communications and Networking Conference (WCNC)*, New Orleans, Louisiana, March 16-20, 2003.

[37] A. Smailagic, D.P. Siewiorek, J. Anhalt, D. Kogan, and Y. Wang, "Location sensing and privacy in a context aware computing environment," *Pervasive Computing*, 2001.

[38] A.F.M. Smith and A.E. Gelfand, "Bayesian statistics without tears: a sampling-resampling perspective," *The American Statistician*, vol. 46, pp. 84-88, 1992.

[39] D.J. Spiegelhalter and S. L. Lauritzen, "Local computations with probabilities on graphical structures and their application to expert systems," *Readings in Uncertain Reasoning,* Morgan Kaufmann Publishers Inc., pp. 415 - 448, 1990.

[40] D.J. Spiegelhalter, A. Thomas, and N.G. Best, "Computation on Bayesian graphical models," *Bayesian Statistics 5*, Oxford University Press, pp. 407–425, 1996.

[41] D.J. Spiegelhalter, "Bayesian graphical modeling: a case study in monitoring health outcomes," *Applied Statistics*, vol. 47, pp. 115–133, 1998.

[42] S. Thrun, "Probabilistic algorithms in robotics," *Technical Report CMU-CS-00-126*, Computer Science Department, Carnegie Mellon University, 2000.

[43] M.P. Wand, "A comparison of regression spline smoothing procedures," *Computational Statistics,* vol. 15, pp. 443-462, 2000.

[44] R. Want, A. Hopper, V. Falcao, and J. Gibbons, "The active badge location system," *ACM Transactions on Information Systems*, vol. 10, pp. 91–102, 1992.

[45] J. Werb and C. Lanzl, "Designing a positioning system for finding things and people indoors," *IEEE Spectrum*, pp. 71-78, September 1998.

[46] M. Youssef, A. Agrawala, and U. A. Shankar, "WLAN location determination via clustering and probability distributions," *IEEE International Conference on Pervasive Computing and Communications (PerCom) 2003*, Fort Worth, Texas, March 23-26, 2003.

[47] S.L. Zeger and P.J. Diggle, "Semiparametric models for longitudinal data with application to CD4 cell numbers in HIV seroconverters," *Biometrics,* vol. 57, pp. 689-699, 1994.