Pacific Institute for the Mathematical Sciences

IMA
Institute for Mathematics and its Applications

Proceedings of the Seventh

PIMS-IMA Industrial Problem Solving Workshop

PIMS-IMA IPSW 7

Co-sponsored by:

**The Natural Science and
Engineering Research Council
of Canada,**

**The Alberta Science and
Research Authority,**

and

The US National Science Foundation

Editor: C. Sean Bohun, Pennsylvania State University

Proceedings of the Seventh Annual PIMS-IMA Industrial Problem Solving Workshop

Editor: C. Sean Bohun, Pennsylvania State University

Co-sponsored by:

The Natural Sciences and Engineering Research Council of Canada

The Alberta Science and Research Authority

The US National Science Foundation

NSERC CRSNG

NSF

Alberta INNOVATION AND SCIENCE

June, 2003

Foreword by the PIMS Director

The Seventh Annual PIMS-IMA Industrial Problem Solving Workshop was hosted by the PIMS at the University of Calgary in Vancouver, May 25–29, 2003. This year it was co-sponsored by the Institute for Mathematics and its Applications (IMA).

Approximately 55 participants worked intensely on seven problems posed by industrial companies from across North America.

The problems were provided by Lalitha Venkataramanan (Schlumberger), Veena B. Mendiratta (Lucent), Bruce McGee (McMillan-McGee Corp.), John R. Hoffman (Lockheed-Martin), Edward Keyes (Orisar), Carlos Tolmasky (Cargill), and Wei Lu (Manifold Data Mining).

PIMS looks forward to the Eighth Industrial Problem Solving Workshop which will be held at the University of British Columbia next year.

Thank you to Sean Bohun from Penn State University who edited these proceedings. I would also like extend my thanks to the organizing committee Rachel Kuske (UBC), Fadil Santosa (IMA), Jack Macki (U. Alberta), Chris Bose (U. Victoria), Huaxiong Huang (York U.), Ian Frigaard (UBC) and Tony Ware (U. Calgary).

Dr. Nassif Ghoussoub, Director
Pacific Institute for the Mathematical Sciences

Contents

List of Figures

List of Tables

Preface

The University of Calgary was the site of the seventh annual PIMS Industrial Problem Solving Workshop (IPSW). Hosted from May 25 through May 29, 2003, this particular workshop was co-sponsored by the Institute for Mathematics and its Applications (IMA) and saw the bringing together of some 55 participants from across North America and Europe.

Many first time participants to these workshops are surprised by the intensity of the work. Based on the Oxford Study Group Model, the morning of the first day consists of the initial problem presentations and the beginning of focussed discussions. Work continues throughout the next few days culminating in a summary presentation on the morning of the fifth day. These presentations form the basis of the proceedings and being one of the principal authors myself, I can appreciate the amount of extra work required to convert the workshop presentation material into a final technical report. I would like to personally thank the other eleven authors for their dedication and prompt response so that these proceedings could appear in a timely fashion. These individuals were:

- Christopher Bose: The Schlumberger Doll Research NMR Inversion Problem;
- Nilima Nigam: Lucent Technologies Warranty Cost Problem;
- Huaxiong Huang, Serguei Lapin & Rex Westbrook: The McMillan-McGee Thermal Remediation Problem;
- Christain Ketelsen, Alfonso Limon, Yuriy Mileyko & Kerianne Yewchuk: Lockheed Martin's Multiple Tracking Problem;
- Fadil Santosa: The Orisar Short Localization Problem;
- Seung Youn Lee: The Cargill Forward Rate Correlation Problem.

To ensure the smooth and efficient operation of the workshop, many individuals are needed behind the scenes. I would like to begin by thanking the organizing committee: Rachel Kuske (University of British Columbia), Fadil Santosa (IMA), Jack Macki (University of Alberta), Chris Bose (University of Victoria), Huaxiong Huang (York University), Ian Frigaard (University of British Columbia) and Tony Ware (University of Calgary), without whose efforts this event would not have been possible. It is a pleasure to mention the support received from NSERC of Canada, the Alberta Science and Research Authority and the NSF.

Two other essential ingredients for the success of the workshop were the industrial representatives and the industrial experts. While the representatives are certainly an asset and provide a grounding for each of the problems, it is the industrial experts that are responsible for each of the groups moving along productive lines. For this year the industrial experts were:

- Rita Aggarwala, University of Calgary;
- C. Sean Bohun, Penn State University;
- Huaxiong Huang, York University;
- Mike Kouritzin, University of Alberta;

- Jack Macki, University of Alberta;
- Nilima Nigam, McGill University;
- Robert Piché, Tampere University of Technology, Finland;
- Juan Restrepo, University of Arizona;
- Fadil Santosa, IMA/University of Minnesota;
- Rex Westbrook, University of Calgary.

A special thanks goes to the industrial contributors and their representatives who included:

- John R. Hoffman, LockHeed-Martin;
- Edward Keyes, Orisar Inc.;
- Wei Lu, Manifold Data Mining;
- Bruce McGee, McMillan-McGee Corp.;
- Veena B. Mendiratta, Lucent Technologies;
- Carlos Tolmasky, Cargill Inc.
- Lalitha Venkatararnanan, Schlumberger Doll Research;

On a personal note, I have attended all but one of the IPSW workshops. My first as a PhD. student, then as an invited participant, more recently as an academic expert, and now where I find myself editing these current proceedings. It is a testament to quality of these workshops and the individuals involved that so many of my colleagues return year after year. At the end of this monograph there is a listing of the workshop participants and the various institutions where they can be found and I would like to take this opportunity to apologize for any mistakes or omissions therein.

In closing I would like to thank Marian Miles at the University of Calgary who dealt with the inevitable day to day administrative minutiae associated with any event of this nature.

C. Sean Bohun, Editor,
Department of Mathematics,
Penn State University

Pimslips

When people are working intensely, they sometimes say things that, in retrospect, are amusing or downright hilarious. Here are some from IPSW 7:

- [Referring to units of pressure] “1 atm, is *approximately* equal to 101.325 kPa.”
- “While true, the equation 0=0 isn’t very useful.”
- “I can’t believe it took us half an hour to show that the time derivative of position is velocity.”
- “We just have to change a couple of things.” [Meaning the entire project.]
- “It takes two days to get eight seconds.”
- “It’s Wednesday, we don’t have a model, we will never have a model, and I don’t care.”
- ”You do not hesitate to take on difficult problems.” [Fortune cookie at dinner.]
- “There’s a talk by a world-renowned speaker today—but I can’t remember his name.”
- “I need *less* support.”
- “So how many numbers *are* there between 5 and 9?”
- “The only shorts we found so far are the ones people wear.”
- “We need the rhos. . . ”
  “You mean the columns!”
  “No, I mean the rhos!”
- “We have to remove artifacts like curved polygons.”

Chapter 1

Inversion of 2D NMR Data

Christopher Bose[1], Zhenlu Cui[2], Xinghua Deng[3], Ying Han[4], Qingguo Li[5], Robert Piché[6],
Lalitha Venkataramanan[7], Qian Wang[3], Lin Zhou[8]

Report prepared by Christopher Bose (cbose@math.uvic.ca)

1.1 Introduction

Schlumberger Limited is a multinational company supplying oilfield and information services to a worldwide energy market. These services include both exploration and production tools ranging through seismic and remote sensing, well-logging and reservoir optimization. The problem described in this report is related to well-logging via Nuclear Magnetic Resonance (NMR), a relatively new and developing tool with potential to reveal a range of reservoir properties including porosity and saturation, as well as physical properties of the petroleum deposit.

In order to recover this information from NMR spectra the company must have an effective, efficient and robust algorithm to perform inversion from the dataset to the unknown probability distribution on magnetic relaxation times. This ill-posed problem is encountered in diverse areas of magnetic imaging and there does not appear to be an ‘off-the-shelf’ solution which the company can apply to its problem. Company scientists have developed a sophisticated algorithm which performs well on some simple test datasets, but they are interested in knowing if there are simpler approaches which could work effectively, or if some limited but useful properties of the density are accessible with a totally different approach.

Our report is organised as follows. In Section 1.2 we present a careful and complete description of the problem and the work already done by the company. In Section 1.3 we discuss Truncated Singular Value Regularisation and Tikhonov Regularisation and show how some ‘off-the-shelf’ Matlab code

[1] University of Victoria
[2] Florida State University
[3] University of Alberta
[4] McGill University
[5] Simon Fraser University
[6] Tampere University of Technology, Finland
[7] Schlumberger Doll Research
[8] New Jersey Institute of Technology

may be used to good effect on the test datasets provided by the company. In Section 1.4 we show that one can incorporate higher order regularisation into the company's existing algorithm, answering one specific question raised at the beginning of the workshop. Finally, in Section 1.5 we record our unsuccessful attempt to establish an iterative algorithm for the positively constrained inversion. Finally in the last section we review our conclusions and make suggestions for future work.

1.2 Problem Description

Schlumberger is interested in using Nuclear Magnetic Resonance (NMR) analysis for exploration in the oil and gas industry. The model problem presented to our group at the workshop involved the recovery of a two-dimensional probability distribution $f(x, y)$ on magnetic field relaxation times in two directions, x, the so-called longitudinal relaxation time and y the transverse relaxation time. The data collected is known to be a convolved image of the relaxation time distribution according to the following formula

$$d(\tau_1, \tau_2) = \iint (1 - 2e^{-\tau_1/x}) e^{-\tau_2/y} f(x, y)\, dx\, dy. \tag{1.1}$$

Other types of data can be collected, involving different convolution kernels, but the forward model, in any case is in the form of a 2-D **Fredholm Integral of the First Kind**. Since the transformation involves a smooth kernel, it is well known that the corresponding inverse problem is ill-posed [5, p.2]. None the less, it is important for the company's program to provide some sort of stable and computationally tractable inversion scheme.

The continuous forward model (1.1) is mainly of theoretical interest since in practice, data is collected at discrete values in the $\tau_1\tau_2$-domain. Therefore, for the rest of this analysis we will assume the data function d is replaced by a data **matrix** D of dimension $m_2 \times m_1$. Convolution kernels are similarly discretized as matrices K_1 and K_2 with dimensions $m_1 \times n_1$ and $m_2 \times n_2$ respectively and a discrete form of the equation (1.1) is rewritten as

$$D = K_2 F K_1^T. \tag{1.2}$$

The discrete density F is now an $n_2 \times n_1$ matrix. K_1 and K_2 are (generally) rank-deficient with infinite condition number and singular values decaying quickly to zero. So, as expected, the ill-posed problem leads to an ill-conditioned finite-dimensional inversion (1.2).

Three sets of test data were provided to our group for use during the workshop. Distribution files F were of size $n_2 \times n_1 = 100 \times 100$. Kernel discretisation led to K_1 and K_2 of size 30×100 and 4000×100 respectively. There are reasonable grounds for the asymmetric choice in the discretisation grid here. For test inversion problems we replace the data D computed from equation (1.2) by $\hat{D} = D + E = K_2 F K_1^T + E$ where E is mean-zero Gaussian noise. The object is to recover F. The choice of signal to noise ratio for the various test files will be discussed later in the numerical results section.

Suppose for the moment we take a completely naive point of view and convert our problem to a standard one-dimensional least squares approximation

$$\text{vec}(\hat{F}) = \arg\min_f \|Kf - \text{vec}(\hat{D})\|^2 \tag{1.3}$$

where $\mathrm{vec}(\cdot)$ represents the operator making a vector from a matrix by stacking columns, $K = K_1 \otimes K_2$ is the Kronecker product of the convolution kernels and $f = \mathrm{vec}(F)$. Since K is huge ($m_1 m_2 \times n_1 n_2 = 120,000 \times 10,000$) and dense, we may have difficulty fitting it into the RAM memory of a PC even if we ignore the computational complexity of the positivity constraint $f \geq 0$ and ill-posed nature of the high dimension inversion! For example, in [3] a similar problem arising in medical imaging was analysed using a CRAY supercomputer. Therefore, we conclude that a numerically reasonable approach of the type required by the company should try to work directly with the factored form (1.2). This observation was known to the company scientists. For this reason, most of our analysis will be centred on the factored problem of the type

$$\hat{F} = \arg\min_{F \geq 0} \|K_2 F K_1^T - \hat{D}\|_{\text{Fro}}^2 \tag{1.4}$$

where $\|\cdot\|_{\text{Fro}}$ denotes the Frobenius matrix norm.

The problem proposor (L.V.) described a three step approach to the the optimisation in (1.4) which the company has found to be effective on the test datasets. First, the problem dimension is significantly reduced by projection. The range of this projection is related to the singular value decomposition (SVD) truncation of the convolution matrices. Next, the ill-conditioned (but lower-dimension) problem is regularised as a positively constrained Tikhonov optimisation in unfactored form

$$\mathrm{vec}(\hat{F}) = \arg\min_{f} \left(\|Kf - \mathrm{vec}(\hat{D})\|^2 + \lambda^2 \|f\|^2 \right) \tag{1.5}$$

where λ is the regularisation parameter. Finally, this constrained problem is solved by the method in Butler, Reeds and Dawson [1] (BRD) which transforms to an unconstrained optimisation with respect to a derived objective function. Details, including methods to choose the regularisation parameter and performance on the test problems may be found in [10].

With this background in place our group was asked to consider three lines of investigation.

First, are there other numerically tractable (and possibly simpler) approaches to the inversion problem (1.2)? In the next section, we describe three answers to this question. First we consider using truncated singular value decomposition (TSVD) and Tikhonov regularisation on the factored form (1.4), greatly reducing the computational and algorithmic complexity of previous methods. Performance on the test datasets is presented. We also consider briefly a direct Galerkin-type approach.

Next, we were asked consider the possibility of extending the BRD-method described by the proposor to higher-order Tikhonov regularization. In particular, can we replace the problem (1.5) with

$$\hat{F} = \arg\min_{F \geq 0} \left\{ \|K_2 F K_1^T - \hat{D}\|_{\text{Fro}}^2 + \lambda^2 (\|LF\|_{\text{Fro}}^2 + \|FL^T\|_{\text{Fro}}^2) \right\} \tag{1.6}$$

where L invokes the discrete first derivatives on the square matrix F? We present mixed results for this second question in that we can transform the problem (1.6) into a standard problem of the type (1.5), to which the BRD method can subsequently be applied, but we cannot arrange that the transformed problem has the desirable Kronecker product structure. In a slightly different direction we consider if the company's idea for an iterative algorithm can be adapted to higher-order regularisation. Unfortunately the same problems which led to the use of the BRD method appear to confound this approach as well.

Finally, the company scientists believe that it may not be necessary to obtain complete inversion of the problem but that some macroscopic information about the distribution of relaxation times (moments or (x_1, x_2)-correlations for example) may be sufficient. While this question may be amenable to a Galerkin approach, without prior information about a restricted class of possible distributions our group saw no tractable way to make progress in this direction during the week of the workshop.

1.3 TSVD and Tikhonov Regularisation

The Theory

Let us first set up a unified framework for these two well known regularisation methods in non-factored problems. Consider the discrete linear system

$$Kf = d \tag{1.7}$$

where K is $m \times n$. Let $K = U\Sigma V^T$ be a SVD where Σ is $m \times n$ diagonal, U is $m \times m$ orthogonal, and V is $n \times n$ orthogonal. The Tikhonov and TSVD regularised solutions of (1.7) are

$$f_{\text{reg}} = V\phi(\Sigma^T)U^T d \tag{1.8}$$

where

$$\begin{array}{cc} \text{Tikhonov} & \text{TSVD} \\ \phi(\sigma) = \dfrac{\sigma}{\sigma^2 + \lambda^2}, & \phi(\sigma) = \begin{cases} 1/\sigma & \text{if } |\sigma| > \lambda \\ 0 & \text{otherwise} \end{cases} \end{array}$$

is applied elementwise. The nonnegative regularisation parameter λ affects the amount of smoothing of the regularised solution. Its value can be selected by minimising the generalised cross validation (GCV) function

$$G(\lambda) = \left(\frac{\|Kf_{\text{reg}} - d\|}{\text{trace}(U\Sigma\phi(\Sigma^T)U^T - I)} \right)^2 = \left(\frac{\|c \circ (U^T d)\|}{\text{sum}(c)} \right)^2$$

where the vector norm is Euclidean, $\circ$ is the Hadamard product (elementwise multiplication), and $c = \text{diag}(\Sigma\phi(\Sigma^T)) - 1$.

Now remember that the coefficient matrix in (1.7) has the Kronecker product structure

$$K = K_1 \otimes K_2$$

where K_1 is $m_1 \times n_1$ and K_2 is $m_2 \times n_2$ with $m_1 m_2 \geq n_1 n_2$. As we have observed, the key to effective algorithms is to rewrite formulae in ways that avoid explicitly forming the Kronecker product of full matrices.

Let $F = \text{reshape}_{n_2 \times n_1}[f]$ and $D = \text{reshape}_{m_2 \times m_1}[d]$. Then the linear system (1.7) is obtained by applying the vec operator to both sides of the matrix equation

$$K_2 F K_1^T = D \tag{1.9}$$

This equation does not involve the Kronecker product. Similar techniques can be used to eliminate expensive Kronecker products from the regularisation formulae, as follows.

Let $K_1 = U_1 \Sigma_1 V_1^T$ and $K_2 = U_2 \Sigma_2 V_2^T$ be SVDs. Then $(U_1 \otimes U_2)(\Sigma_1 \otimes \Sigma_2)(V_1 \otimes V_2)^T$ is an SVD of K [7, Thm 4.2.15]. The formula (1.8) for the regularised solution can therefore be written

$$\begin{aligned} F_{\text{reg}} &= \text{reshape}_{n_2 \times n_1}[f_{\text{reg}}] \\ &= \text{reshape}_{n_2 \times n_1}[(V_1 \otimes V_2)\phi(\Sigma_1^T \otimes \Sigma_2^T)(U_1 \otimes U_2)^T d] \\ &= V_2 \cdot \text{reshape}_{n_2 \times n_1}[\phi(\Sigma_1^T \otimes \Sigma_2^T)\text{vec}(U_2^T D U_1)] \cdot V_1^T. \end{aligned} \tag{1.10}$$

The Kronecker product formation and multiplication in (1.10) only involves diagonal matrices, so the formula can be implemented efficiently with appropriate data structures. Further savings are possible by using the “economy size” version of the SVD of K_2 when $m_2 > n_2$.

Similarly, the formula for the GCV function can be written

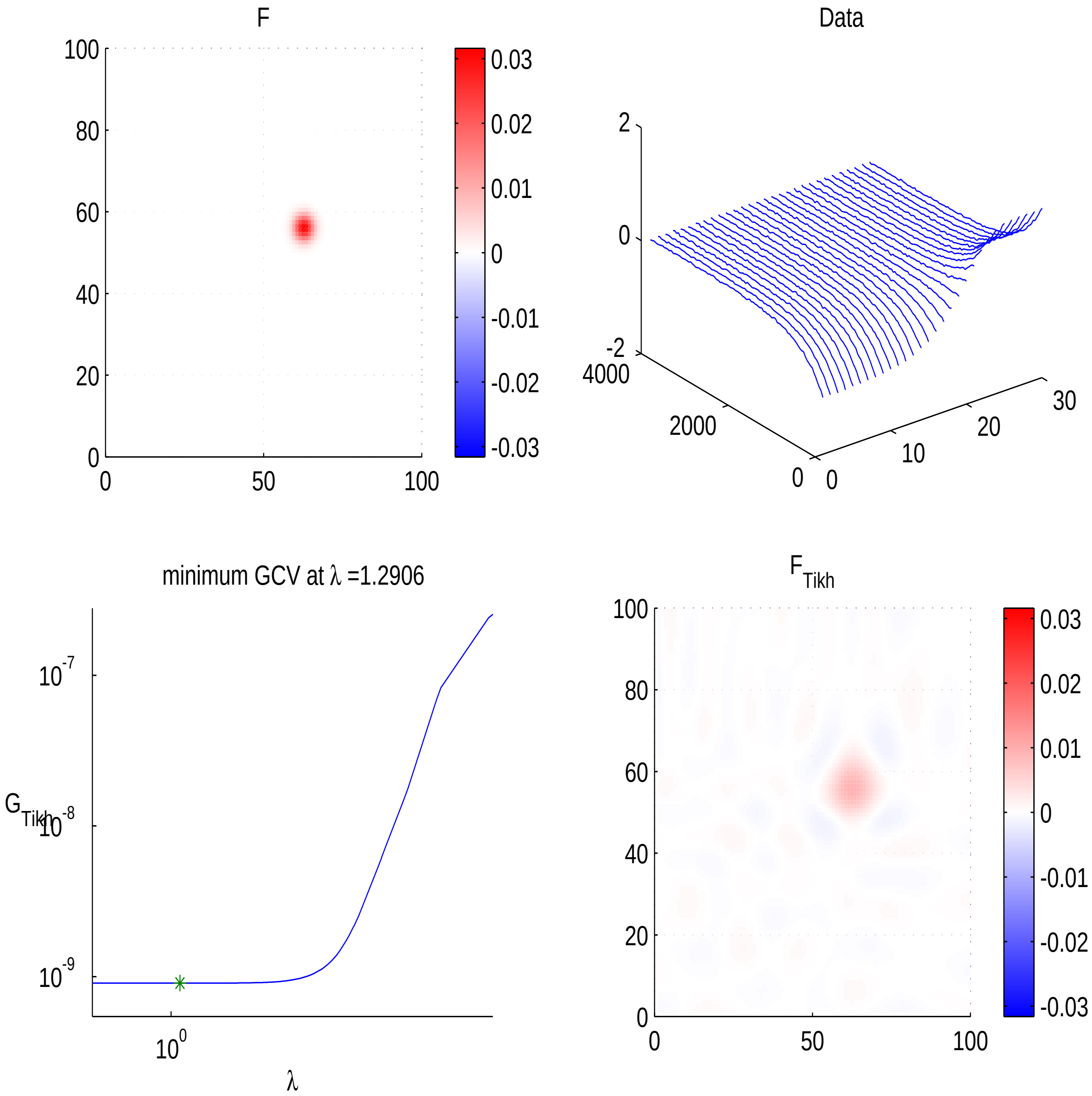
$$G(\lambda) = \left(\frac{\|C \circ (U_2^T D U_1)\|_{\text{Fro}}}{\text{sum}(C)} \right)^2$$

where $C = \text{reshape}_{m_2 \times m_1}[\text{diag}((\Sigma_1 \otimes \Sigma_2)\phi(\Sigma_1^T \otimes \Sigma_2^T))] - 1$. Here again the only Kronecker products are of diagonal matrices.

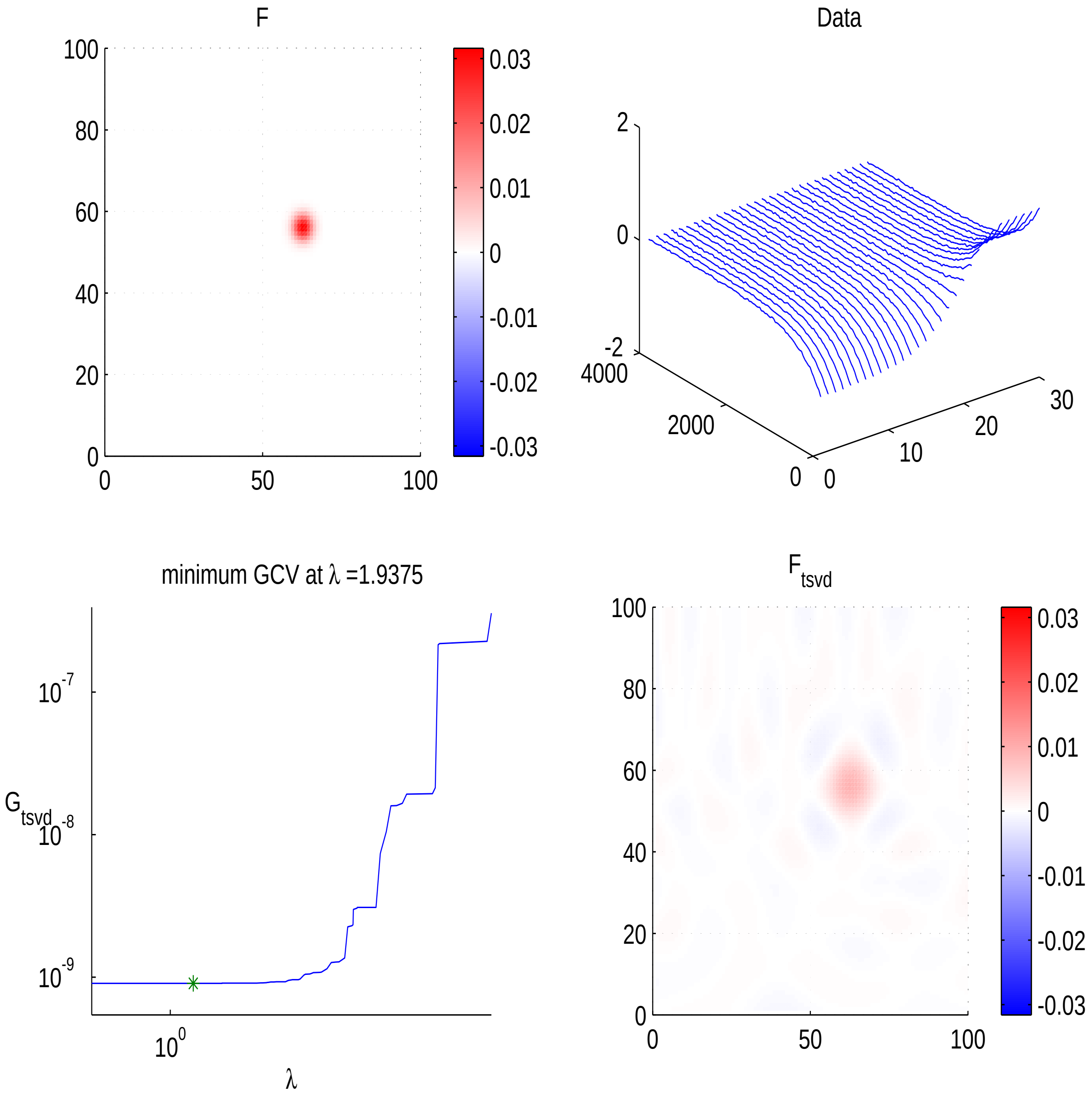
Numerical Results

The problem proposer provided K_1 (of size 30×100), K_2 (of size 4000×100), and three different 100×100 F matrices. Measurement data was generated by adding zero mean pseudorandom noise E to $K_2 F K_1^T$; the noise variance was set so that $\|E\|_{\text{Fro}} = 0.05 \|K_2 F K_1^T\|_{\text{Fro}}$.

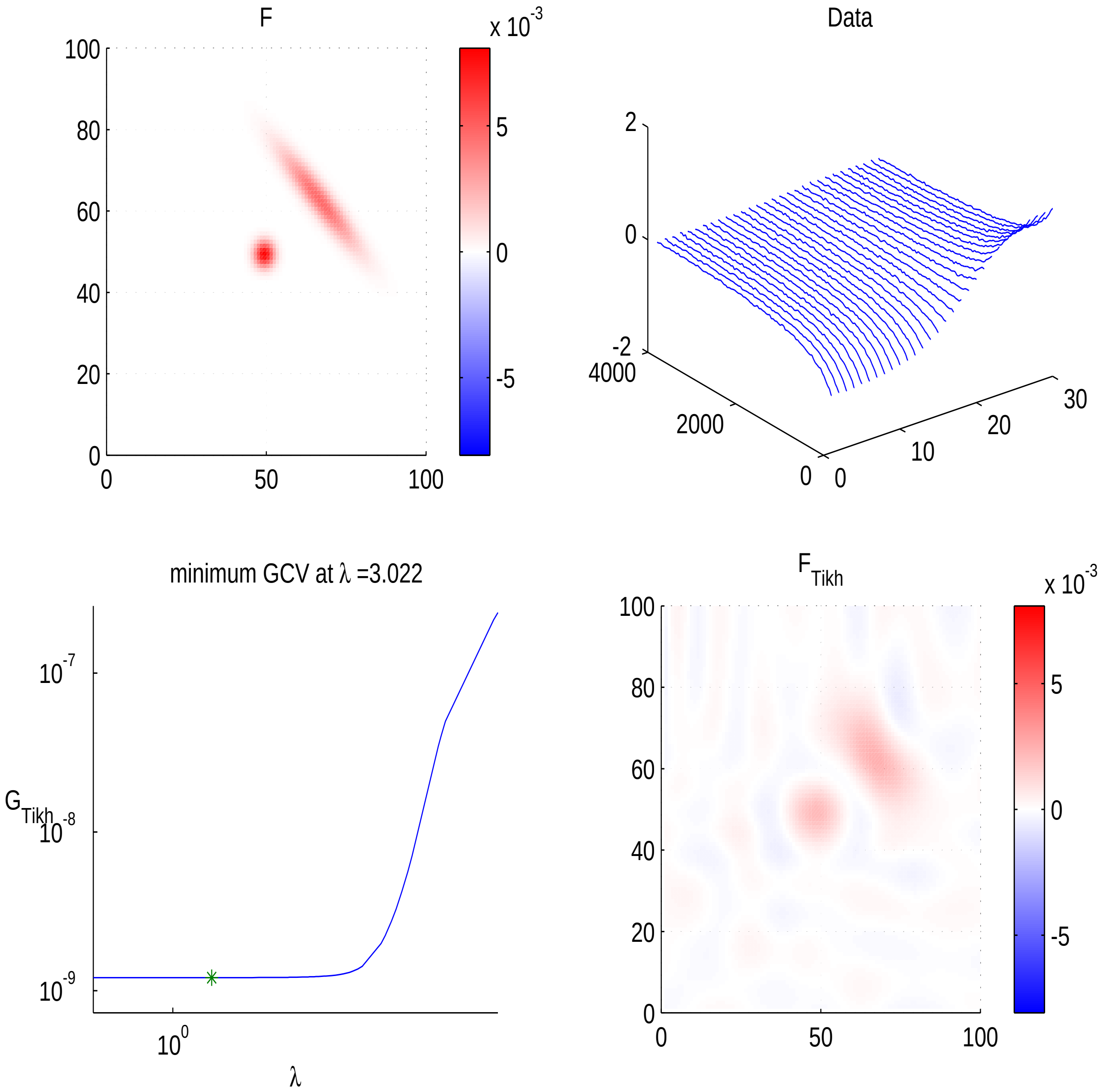
Each of the following regularisations (including SVD and GCV curve computations) took about 12 seconds to compute in Matlab 5.2 on a 30 MB memory partition of a 400 MHz Powerbook. The GCV minimisation appears to select reasonable regularisation parameters, and Tikhonov and TSVD regularisation give about the same results for all three models. The data and Matlab code are available at `http://alpha.cc.tut.fi/~piche/ipsw2003/`



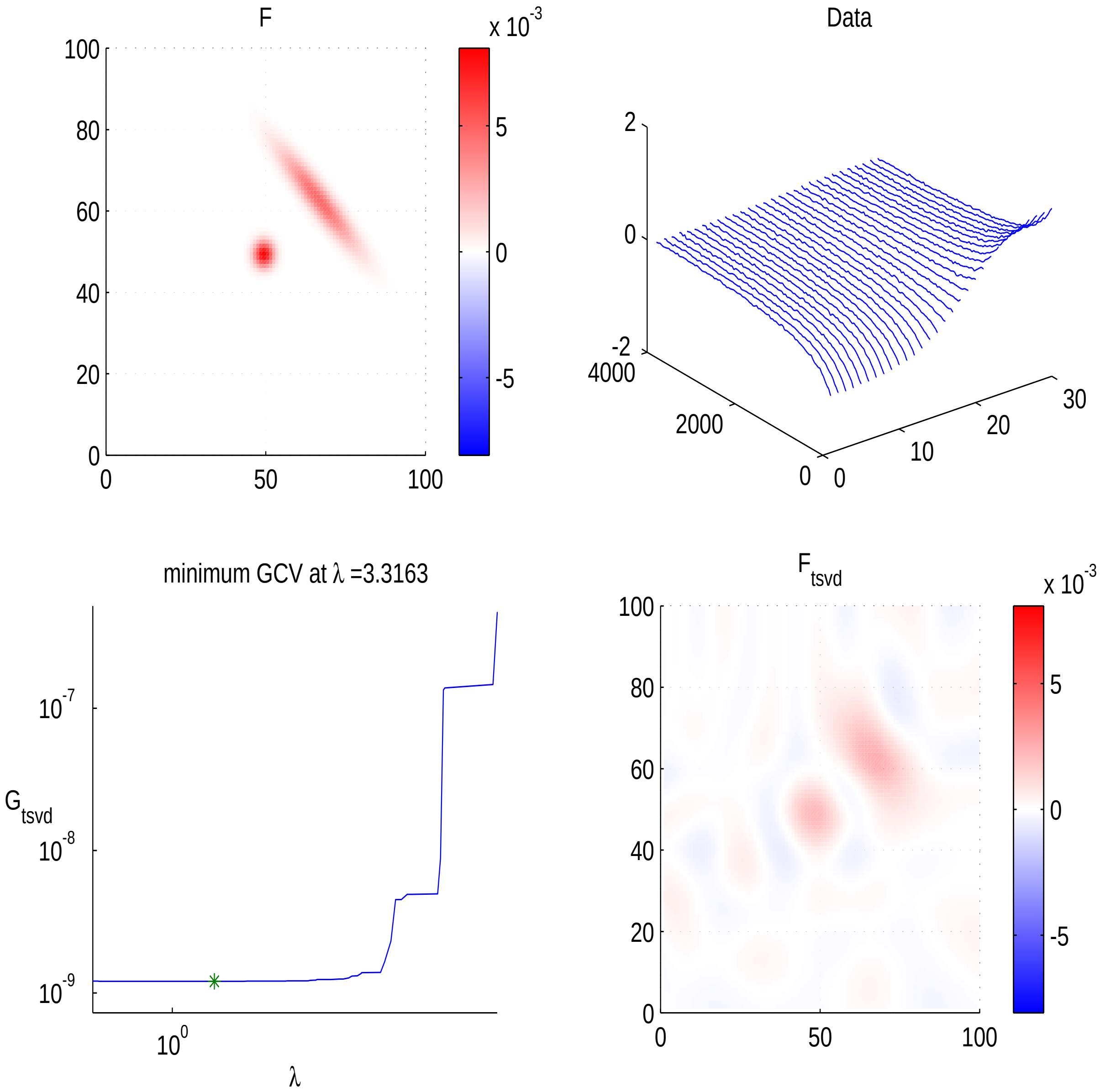
model A, Tikhonov regularisation



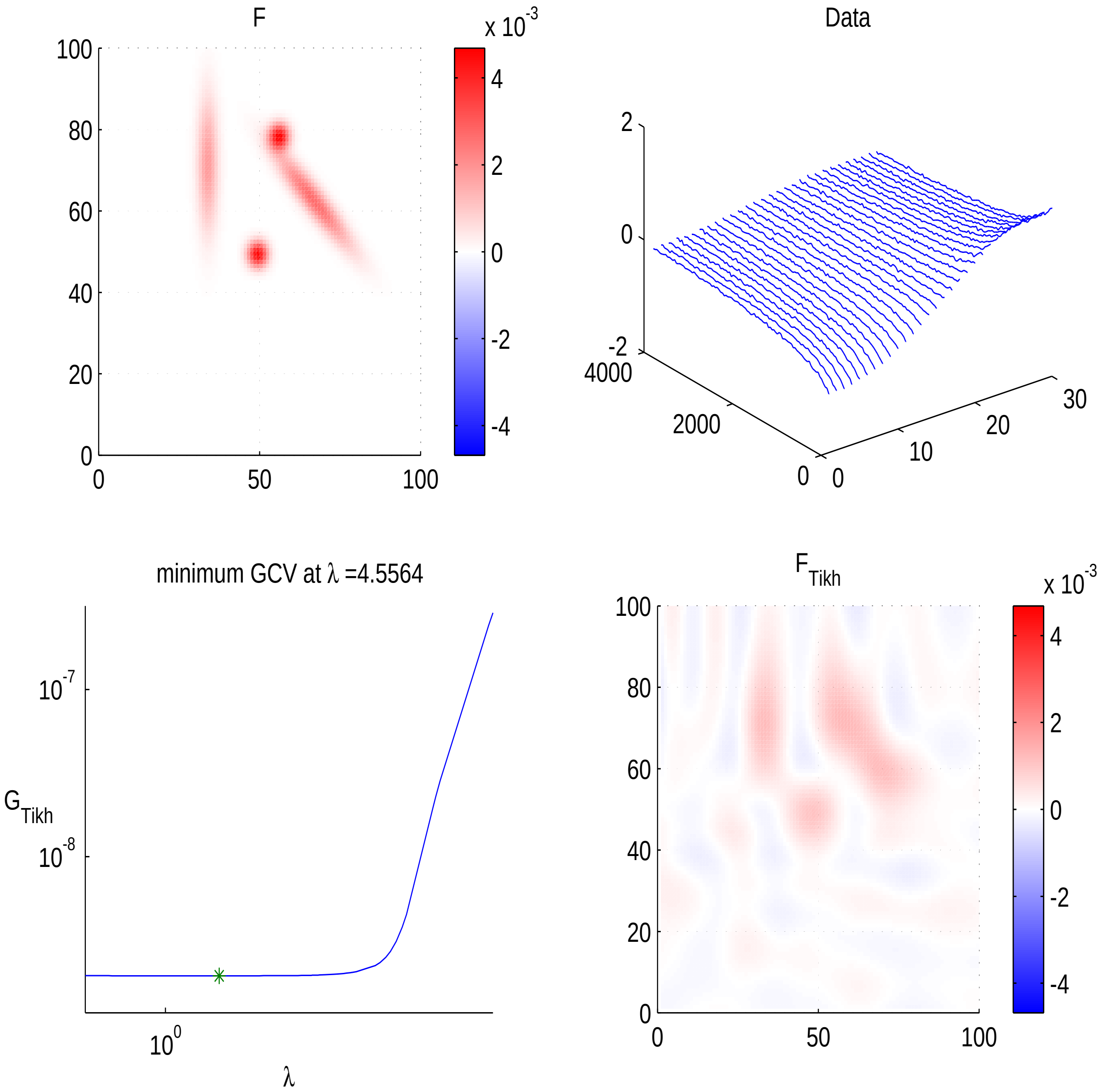
model A, TSVD regularisation



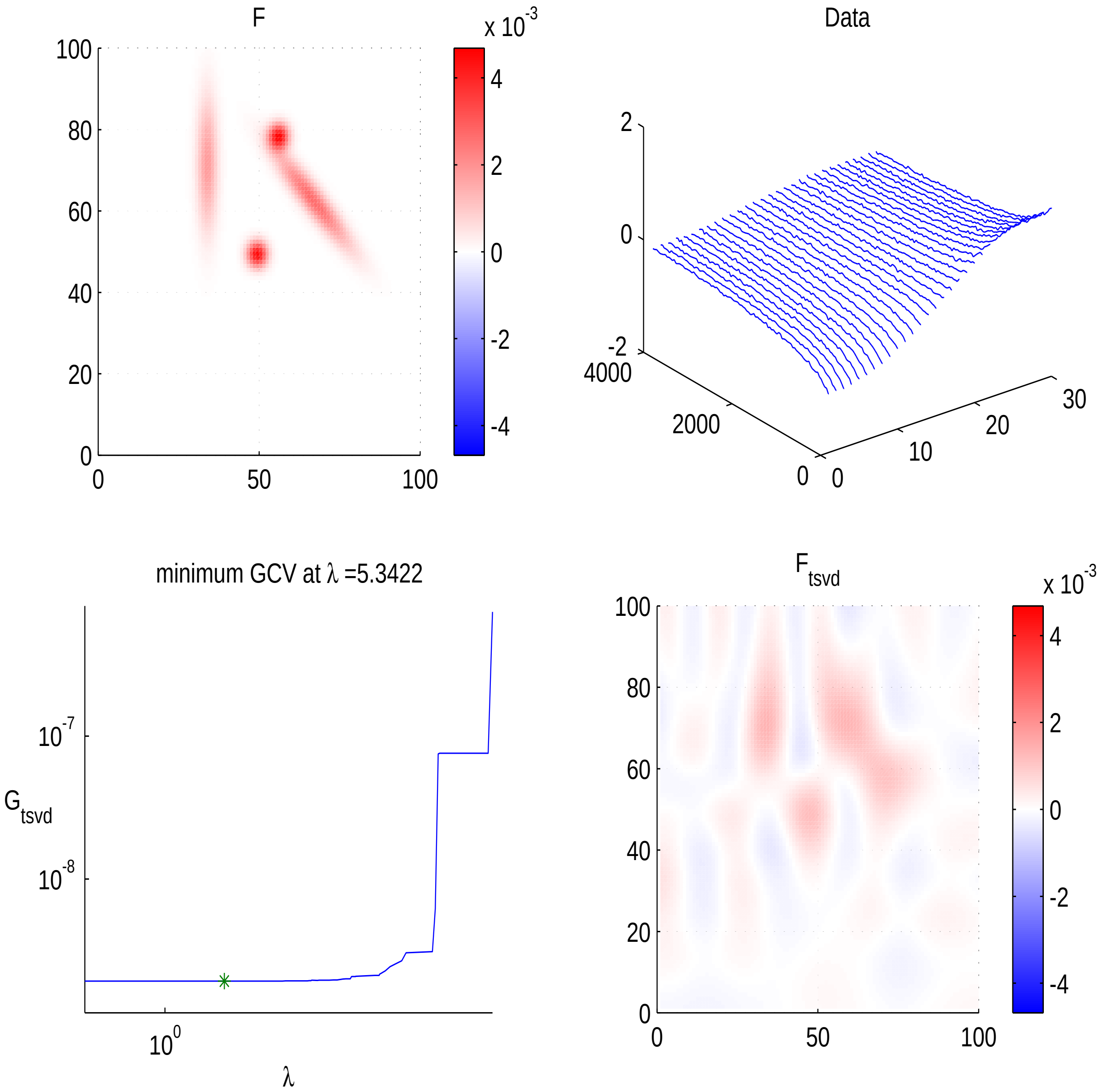
model B, Tikhonov regularisation



model B, TSVD regularisation



model C, Tikhonov regularisation



model C, TSVD regularisation

Parameterised Methods

Our group briefly considered the possibility of using a Galerkin approach to the inversion problem (1.1). Thus, we make the ansatz

$$f(x,y) = \sum_i f(x,y,\mathcal{P}_i) \tag{1.11}$$

where the $f(x,y,\mathcal{P}_i)$ are a finite set of parameterised basis functions with parameter values $\mathcal{P}_i$.

We remark that this approach reduces to the analysis of the previous paragraphs by the choice of the basis functions as 'delta functions' $f(\cdot, F_{ij}) = F_{ij}\delta_{(}x_i, y_j)(\cdot)$ centred on the points of the discretisation lattice and the optimal parameter selection is the matrix $\hat{F}$ of the previous analysis. As we have discovered this is a high-dimensional, ill-posed and (because of the positivity requirement) nonlinear problem. Our question then is this: Can a judicious choice of basis functions lead to a significantly smaller parameter space (instead of the 10,000 - dimensional space already encountered)? Of course the ill-posed nature of the problem must reappear in the Galerkin method as the number of basis functions increases, no matter how cleverly this basis is chosen.

The following are a few examples for the basis functions which seem well-suited to the test problems given to the group.

1. Gaussian functions ($\mathcal{P} = \{x_0, y_0, \sigma, P\}$)

   $$g(x,y) = \frac{1}{\sqrt{2\pi\sigma}}\, e^{\mathbf{x}^T P^T P \mathbf{x}}$$

   where $\mathbf{x} = (x - x_0, y - y_0)^T$.

2. Box functions with centre at (x_0, y_0), and dimension of its base are $2a$ and $2b$ and height $1/4ab$.

3. Pyramid functions with centre at (x_0, y_0), and square base dimension of $2a$ and $2b$ and height $3/4ab$.

For example, when using a basis consisting of one box function ($\mathcal{P} = \{c, x_0, y_0, a, b\}$) to approximate $f(x,y)$ the Fredholm integral becomes

$$\begin{aligned} D_{\mathcal{P}}(\tau_1, \tau_2) &= \frac{c}{4ab} \int_{x_0-a}^{x_0+a} \int_{y_0-b}^{y_0+b} (1 - 2e^{-\tau_1/x})(e^{-\tau_2/y})\, dy\, dx \\ &= \frac{c}{4ab} \int_{x_0-a}^{x_0+a} (1 - 2e^{-\tau_1/x})\, dx \int_{y_0-b}^{y_0+b} e^{-\tau_2/y}\, dy. \end{aligned} \tag{1.12}$$

There is no closed form solution to this integral, which can only be solved numerically.

Based on this simple parameterization, the problem of approximating $f(x,y)$ is transformed as a nonlinear optimization problem stated as follows:

For given $\tau_1 \in [0, T_1], \tau_2 \in [0, T_2]$ and data $\hat{D}(\tau_1, \tau_2)$, the objective is to find $\mathcal{P} = \{c, x_0, y_0, a, b\}$, $c \geq 0$ such that

$$\hat{F} = \underset{f(x,y,\mathcal{P})}{\arg\min}\, \|D_{\mathcal{P}} - \hat{D}\|^2_{\text{Fro}}.$$

Unfortunately, due to time constraints we were unable to conduct numerical tests on this optimisation problem during the week of the workshop.

We note that even though this overly simplified approach has no hope of establishing fine structure of the underlying density, it would be interesting to see if macroscopic properties desired by the company scientists could be isolated with such a relatively low-dimensional parameterisation. On the other hand, the method depends on *a priori* information about the the density, likely a fatal flaw for any robust numerical package of the type required by the company.

1.4 Higher-Order Tikhonov Regularisation

The Tikhonov regularised solution described Section 1.3 is the minimiser of the objective function

$$\frac{1}{2}\|Kf-d\|^2+\frac{1}{2}\lambda^2\|f\|^2=\frac{1}{2}\|K_2FK_1^T-D\|^2_{\mathrm{Fro}}+\frac{1}{2}\lambda^2\|F\|^2_{\mathrm{Fro}}. \tag{1.13}$$

A more general regularisation has the objective function's second term in the form

$$\frac{1}{2}\lambda^2\|LF\|^2_{\mathrm{Fro}}$$

where the operator L is chosen to penalise undesired features of the solution. When L has the same Kronecker product structure as K, then it is straightforward to develop efficient regularisation algorithms along the lines of the previous section.

In this section we will consider the more difficult problem of incorporating a non-factored regularisation term. For example, setting

$$L=\begin{pmatrix} 1 & & & & \\ -1 & 1 & & & \\ & -1 & 1 & & \\ & & \ddots & \ddots & \\ & & & -1 & 1 \\ & & & & -1 \end{pmatrix}_{(n+1)\times n}$$

the discrete first derivative, and changing the regularisation term in (1.13) to be

$$\frac{1}{2}\lambda^2(\|LF\|^2_{\mathrm{Fro}}+\|FL^T\|^2_{\mathrm{Fro}})$$

in effect regularises by the **boundary value problem**

$$\Delta_d F=0$$

$$F(1,j)=F(n,j)=0 \quad \text{for all } j$$

$$F(i,1)=F(i,n)=0 \quad \text{for all } i$$

where Δ_d denotes the discrete Laplacian. The boundary conditions ensure that L has trivial kernel which will be useful for us later. Similar considerations would allow regularisation with respect to higher order derivatives, for example replacing L with the discrete Laplacian operator plus appropriate boundary conditions to ensure a trivial kernel.

The estimation of F is equivalent to solving the following problem

$$\hat{F} = \arg\min_{F \geq 0} \left\{ \|K_2...etc...\|^2_{\text{Fro}}) \right\}, \tag{1.14}$$

where K_1, K_2 are the convolution kernels and $\hat{D}$ is noisy data. Note that in this section we are keeping the notation simple by assuming a square, $n \times n$ unknown F but the method trivially extends to rectangular F.

The first term in the two-dimensional problem in (1.14) can be transformed to a one-dimensional problem as before:

$$\|K_2 F K_1^T - \hat{D}\|^2_{\text{Fro}} = \|Kf - \hat{d}\|^2,$$

where the vectors $f = \text{vec}(F)$ and $\hat{d} = \text{vec}(\hat{D})$ are obtained from matrices F and $\hat{D}$, respectively, and $K = K_1 \otimes K_2$.

Next, let

$$LF = LFI = (I \otimes L)f = L_1 f,$$
$$FL^T = IFL^T = (L \otimes I)f = L_2 f,$$

then

$$\begin{aligned}
\|LF\|^2_{\text{Fro}} + \|FL^T\|^2_{\text{Fro}} &= \|L_1 f\|^2 + \|L_2 f\|^2 \\
&= f^T L_1^T L_1 f + f^T L_2^T L_2 f \\
&= f^T (L_1^T L_1 + L_2^T L_2) f \\
&= f^T \widetilde{L}^T \widetilde{L} f \\
&= \|\widetilde{L} f\|^2,
\end{aligned}$$

where $\widetilde{L}$ is the upper triangular Cholesky factor of the positive definite and symmetric matrix $(L_1^T L_1 + L_2^T L_2)$.

Assume for the moment that $\widetilde{L}^{-1}$ is positive in the sense that $\widetilde{L}^{-1} g \geq 0$ whenever $g \geq 0$. Then with $g = \widetilde{L} f$ the objective function in (1.14) becomes one-dimensional as:

$$\begin{aligned}
\min_{f \geq 0} \left(\|Kf - \hat{d}\|^2 + \lambda^2 \|\widetilde{L} f\|^2 \right) &= \min_{\widetilde{L}^{-1} g \geq 0} \left(\|K \widetilde{L}^{-1} g - \hat{d}\|^2 + \lambda^2 \|g\|^2 \right) \\
&\leq \min_{g \geq 0} \left(\|\widetilde{K} g - \hat{d}\|^2 + \lambda^2 \|g\|^2 \right),
\end{aligned}$$

where $\widetilde{K} = K \widetilde{L}^{-1}$. In this case we suggest to take $\text{vec}(\hat{F}) = \widetilde{L}^{-1} \hat{g}$ as an estimate of the minimiser in (1.14).

Regarding the assumptions made in the previous paragraph we note that it is a straightforward calculation to show that $\widetilde{L}^T \widetilde{L} = (L_1^T L_1 + L_2^T L_2)$ is a banded, symmetric, positive definite matrix with non-positive off-diagonal elements. In [9] it is shown that such **Stieltjes** matrices have non-negative (elementwise) inverses. While we have not been able to prove the same thing for the Cholesky factor

$\widetilde{L}$, we believe it to be true for the general class of discrete differentiation operators that we have in mind for applications. In particular, all of our numerical examples have exhibited this property. We suggest that the general fact may already be known in the literature and if not, it would make an interesting problem for future investigation. Perhaps a more interesting and important issue is to show that the value of the optimisation problem above posed in terms of $g \geq 0$ is the same as the value of the $f \geq 0$-problem in order justify our use of $\widetilde{L}^{-1}\hat{g}$ as a rigorous estimate for $\text{vec}(\hat{F})$ above.

Finally, as we have pointed out before, methods which involve unfactorised convolutions are computationally unwieldy and it would be helpful to come up with factored versions of the above transformation.

All of these points merit further investigation.

1.5 Duality

Extending the notation of the previous section we define

$$Q(f) = \frac{1}{2}\|Kf - \hat{d}\|^2 + \frac{1}{2}\lambda^2\|Df\|^2 \tag{1.15}$$

and rewrite the optimisation problem

$$\text{Minimize } Q(f), \text{ subject to } f \geq 0. \tag{1.16}$$

Here we are assuming that the regularisation operator D and regularisation parameter λ have been given to us in advance.

Standard duality analysis and the principle of strong duality implies the Karush-Kuhn-Tucker necessary optimality conditions on f and μ (the Lagrangian dual vector):

$$f \geq 0, \;\; \mu \geq 0, \;\; \sum_i \mu_i f_i = 0, \;\; \nabla Q(f) = \mu.$$

A straightforward calculation gives

$$\nabla Q(f) = K^T(Kf - \hat{d}) + \lambda^2 D^T Df.$$

Substituting this result into the KKT conditions yields our basic optimality conditions

$$\begin{aligned} K_i^T(Kf - \hat{d}) = &\quad -\lambda^2 D_i^T Df \quad \text{if } \; f_i > 0 \\ K_i^T(Kf - \hat{d}) \geq &\quad -\lambda^2 D_i^T Df \quad \text{if } \; f_i = 0. \end{aligned} \tag{1.17}$$

Here $K_i, \; D_i =$ denote the i-th columns respectively. An important point to be made here is that (1.17) are equivalent to the KKT-optimality conditions.

It is possible to write conditions (1.17) as a closed form expression involving f. First we write

$$g = \lambda^{-2}(\hat{d} - Kf) \tag{1.18}$$

after which we find

$$D^T Df = \max[0, K^T g].$$

Consider now the case of first order regularisation where $D = I$. Then

$$f = \max[0, K^T g]$$

and it is tempting to attempt to recover f via an iterative scheme. However, in practice this approach leads to serious convergence problems as described in [10]. It is exactly at this point that the BRD method [1] provides a way to avoid a direct iterative approach. Details are to be found in [10].

The proposor has asked if an iterative method can be salvaged or, failing that can the BRD method be applied when $D \neq I$. We were not able to answer this question clearly during the week of the workshop, however we record here for completeness some observations made by both the workshop members and the problem proposor.

First, suppose we define $\Gamma = \mathrm{diag}(\|D_i\|^2)$, the diagonal elements of $D^T D$. Γ is strictly positive on the diagonal. Writing $D^T D f = (D^T D - \Gamma) f + \Gamma f$ we can rewrite the above closed form expression as

$$f = \max[0, \Gamma^{-1}(K^T g - (D^T D - \Gamma) f)]. \tag{1.19}$$

Here we are using the fact that the $\max$ operator commutes with Γ.

If we denote by $\tilde{f}$ the least squares best (unregularised) solution we can further simplify (1.19) as

$$f = \max[0, \Gamma^{-1}(\lambda^{-2} K^T K(\tilde{f} - f) - (D^T D - \Gamma) f]$$

so the iterative properties of the map

$$f \rightarrow \Gamma^{-1} \lambda^{-2} K^T K \tilde{f} - \Gamma^{-1}(\lambda^{-2} K^T K + D^T D - \Gamma) f$$

need to be explored. In our opinion, the main barrier to convergence is the nonlinear effect invoked by the $\max$ operator in the above iterative scheme.

1.6 Conclusions and Future Work

In this report we have shown how relatively simple, off-the-shelf code can be effectively applied to solve Fredholm Integrals of the first kind through TSVD and Tikhonov regularisation. Higher-order regularisation can also be incorporated with some additional technical difficulties, depending on the nature of the regularising operator. Iterative schemes for solving regularised problems are known in the literature, but work remains in order apply these ideas to the present setting.

Future Work

Bidiagonalisation vs SVD: Eldén's bidiagonalisation algorithm [2] for computing Tikhonov regularised solutions is normally faster than the SVD-based formula (1.8). Developing a version of Eldén's algorithm that exploits the Kronecker product structure would be a good research topic. The work of Faucett and Fulton [4] could be a starting point. However, we expect that a Matlab implementation (without MEX files) of such an algorithm would probably not be any faster than the SVD-based algorithm presented here.

Factored form of higher-order regularisation: It should be straightforward to develop efficient regularisation algorithms when the regularisation operator has factored form. Devising such penalisation operators is an interesting topic for future work. Also, for more general regularisations, the connection between the first order transformed problem and the higher-order problem should be investigated.

Nonnegative constraints: A number of iterative methods are available for regularisation with nonnegative constraints on the solution [11][chapter 9]. It should be straightforward to recode these algorithms to exploit Kronecker product structure. Again, the key to obtaining efficient code is to eliminate expensive Kronecker products from formulae appearing in the algorithm. For example, the gradient projection method involves the objective function and the gradient. The Tikhonov regularisation objective function (1.13) has the gradient

$$K^T(Kf - d) + \lambda^2 f = \mathrm{vec}(K_2^T(K_2 F K_1^T - D)K_1 + \lambda^2 F) \tag{1.20}$$

The right hand side formulae of (1.13) and (1.20) are the ones to use in the iterative algorithm.

Bibliography

[1] J.P. Butler, J.A. Reeds and S.V. Dawson, Estimating solutions of the first kind integral equations with nonnegative constraints and optimal smoothing, SIAM J. Numer. Anal., vol. 18, no. 3, 1981, pp. 381–397.

[2] L. Eldén, Algorithms for the regularization of ill-conditioned least squares problems, BIT, vol. 17, 1977, pp. 134–145.

[3] A.E. English, K.P. Whittall, M.L.G. Joy and R.M. Henkelman, Quantitative two-dimensional time correlation relaxometry, Magn. Reson. Med. vol. 22, pp. 425-434, 1991.

[4] Donald W. Fausett and Charles T. Fulton, Large least squares problems involving Kronecker products, SIAM J. Matrix Anal. Appl. vol. 15, No. 1, pp. 219–227, 1994.

[5] C.W. Groetsch, The Theory of Tikhonov Regularization for Fredholm Equations of the First Kind, Pitman, Boston, 1984.

[6] Per Christian Hensen, Regularization Tools, a Matlab Package for Analysis and Solution of Discrete Ill-Posed Problems. `pch@imm.dtu.dk, http://www.imm.dtu.dk/˜pch`, 2001.

[7] Roger A. Horn and Charles R. Johnson, Topics in Matrix Analysis, Cambridge University Press, 1991.

[8] Julie Kamm and James G. Nagy, Kronecker product and SVD approximations in image restoration, Linear Algebra and Applications vol. 284, 1998, 177–192.

[9] R. Varga, Matrix iterative analysis, Prentice-Hall, 1962, p. 85.

[10] L. Venkataramanan, Y. Song and M.D. Hürlimann, Solving Fredholm integrals of the first kind with tensor product structure in 2 and 2.5 dimensions, IEEE Trans. Sign. Proc., vol. 50, no. 5, 2002, pp. 1017–1026.

[11] Curtis R. Vogel, Computational Methods for Inverse Problems, SIAM, 2002.

Chapter 2

Modelling Quality and Warranty Cost

C. Sean Bohun[1], Olivier Dubois[2], Hongbin Guo[3], Veena B. Mendiratta[4], Nilima Nigam[2],
Kostyantyn Stepankevych[5], Tzvetalin Vassilev[6]
Report prepared by Nilima Nigam (nigam@math.mcgill.ca)

2.1 Problem Description and Methodology

The main aim of this project was to begin a modelling effort directed at optimizing the warranty and quality costs associated with the production of a system with both hardware and software components. This optimization would be constrained by the need to maintain reliability of the product, while staying within an operational budget. For a more detailed problem statement, see [1]. Our aim was to identify important quality attributes, and capture overall trends in costs and warranties. More concretely, our goals were:

- Identifying the major quality-related attributes of interest, denoted by a vector $\mathbf{q}$,
- Modelling the key indicators of the reliability constraint: the failure rate (FR($\mathbf{q}$)) and the severity level (SL($\mathbf{q}$)),
- Modelling the cost of building a product to a certain quality level, $C(\mathbf{q})$,
- Modelling the warranty costs of a product built to a certain quality level, $W(\mathbf{q})$.

The optimization model is to minimize the sum of the quality and warranty costs over the entire class of admissible quality-related attribute vectors. This procedure is accomplished while simultaneously ensuring that the failure rate remains below a specified maximum FR$_{\max}$ and the severity level remains above a given minimum SL$_{\min}$ with a given probability level p. In other words, determine

$$\mathcal{F}_{\textit{opt}} = \min_{\mathbf{q}}(C(\mathbf{q}) + W(\mathbf{q})), \tag{2.1}$$

[1] Pennsylvania State University
[2] McGill University
[3] University of Alberta
[4] Lucent Technologies
[5] University of Calgary
[6] University of Saskatchewan

subject to

$$P(\text{FR}(\mathbf{q}) < \text{FR}_{\max}) < p, \qquad P(\text{SL}(\mathbf{q}) > \text{SL}_{\min}) < p \tag{2.2}$$

over all admissible $\mathbf{q}$. In this project we did not perform this optimization, focusing instead on the modelling of the function involved.

It is important to note at this juncture that no raw data from Lucent was provided for this project, nor did we have specific information about the particular products being built. It was therefore not feasible to use existing hazard/risk models for the various components. Our modelling effort was thus critically dependent on discussions with the industrial contact, Prof. Veena Mendiratta. In the section on future directions we make a series of recommendations which will help refine the models involved.

We systematically identified the key quality-related attributes, described by a quality vector $\mathbf{q}$, which could be measured and quantified. We then developed reliability, warranty and cost models based on these. As our discussions progressed, it became clear that these quality attributes were not all independent. Nor were they all equally important indicators of overall quality. It is thus possible to simplify the models considerably by focusing on the effects of the most important attributes, making the optimization problem (2.1) simpler to solve. In practice, once cost functions and parameters have been picked on the basis of standard hazard models, it will be possible via scaling arguments to achieve further simplification.

With a view to illustrating qualitative trends predicted by our models, we generated some test data (see Subsection 2.8), and ran our models on them. The graphs presented in this report are therefore not linked to any true data, and serve only to provide qualitative information.

2.1.1 A Road Map

The following list details the strategy for this report. Figure 2.1 illustrates how the various sections of the report interconnect.

Section 2: In this section we identify the quality-related variables, $\mathbf{q}$, which drive the various costs associated with a product, and over which the optimization will occur. The fact that many of these variables are not independent will be dealt with later in this report.

Section 3: Here, we develop models for the reliability constraints, the failure rate FR and the severity levels SL. As well as providing some graphical insight into the dependence of these models on the quality $\mathbf{q}$, we also discuss how these failure rates determine the probability of the various modes of failure. These probabilities play a role in determining the warranty costs of a product.

Section 4: At this point in the report a model for $C(\mathbf{q})$, the cost of implementation of a given quality level $\mathbf{q}$ is proposed.

Section 5: This contains the development of the warranty models for hardware W_{hw} and software W_{sw} aspects of a product.

Section 6: Here we combine the models to summarize the total proposed optimization problem.

Section 7: A sensitivity to parameters is discussed, providing insight into the relative importance of terms in the various models that have been introduced.

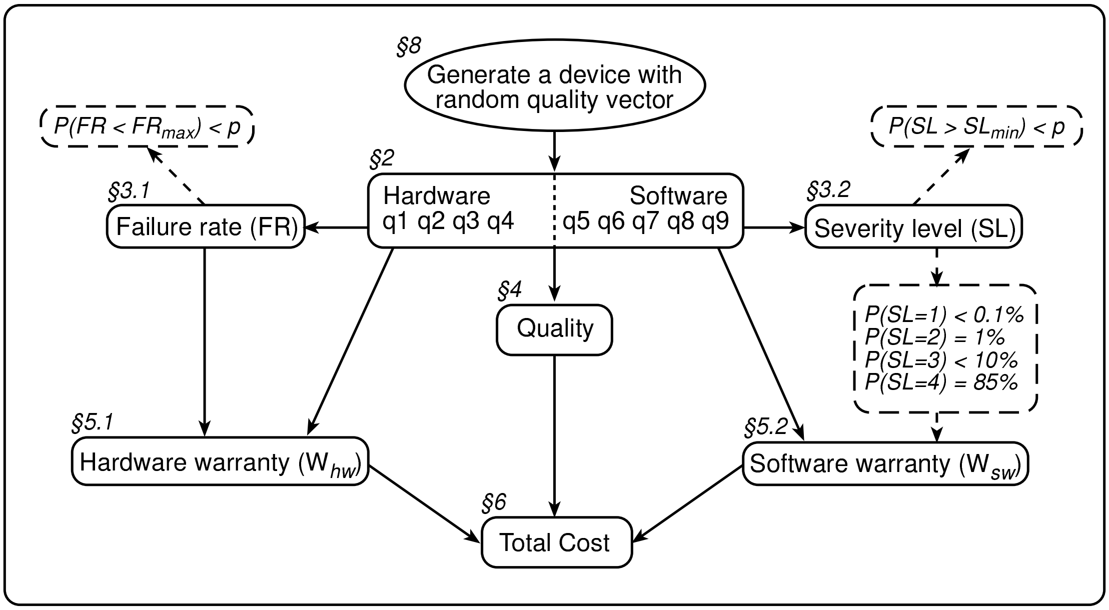


Figure 2.1: Illustrated are the various sections of this report and how they interconnect.

Section 8: Broad trends in the proposed models are generated by drawing the quality-related variables from a given probability distribution. Two cases are considered, each characterized by the probability distribution being used.

2.2 Quality Attributes Vector

We begin by identifying the important quality-related attributes which are both salient and measurable in the context of this project. These attributes fall into two broad categories – hardware-related and software-related – and the optimization of the total cost will be performed over these attributes. In practice, most of these attributes will be measured statistically. In the absence of raw data, we are unable to provide statistical models for these attributes, which will change depending on the product.

Mathematically, these attributes are gathered in a **quality vector**

$$\mathbf{q} = (q_1, q_2, q_3, ..., q_9) \in D := [0, 1]^9.$$

The cost will be optimized as a function of $\mathbf{q} \in D$, subject to certain reliability constraints. We have scaled these attributes q_i to take on values between 0 and 1 for convenience. This enables us to compare, for example, a quantity originally measured as a percentage with one measured as a number between 0 and 10. When using the model in application, it will be important to identify the units used and convert them if necessary.

The various quality attributes are described below.

HARDWARE:

q_1: *Component quality.* In practice measured as a failure rate percentage per year. Here, this rate is converted to a scale from 0 to 1, and is called q_1.

- q_2: *Infant mortality factor* (IMF). Measured as the ratio of the initial failure rate to the steady-state failure rate, this is a number between 1 and 2. In this project, we use the scaling $q_2 =$ measured infant mortality factor -1.
- q_3: *Diagnostics capability*. This attribute is denoted q_3 and lies between 0.8 and 1. In practice, it is measured as a percentage, typically between 80 and 100.
- q_4: *Working environment range*. The variable q_4 is defined as the amount by which the constructed working range exceeds the specifications of the device. For example, suppose the device is intended to operate between 0°C and 100°C, but is built with a working range of -10°C to 125°C. The constructed working range exceeds the operational specifications by 10°C on the lower end, and by 25°C on the higher end. Thus, we would compute $q_4 = (10 + 25)/(\text{working range}) = 35/100 = 0.35$.

From the description of these hardware-related attributes, it is not immediately obvious which are the best indicators of overall quality.

SOFTWARE:

- q_5: *Software development environment* (SDE). Denoted q_5, this describes the overall quality metric of the software development process.
- q_6: *Code complexity*. This metric measures the complexity of a code based on a variety of indicators. Essentially, the more complicated the interactions between different parts of a large code, the harder it is to ensure reliability.
- q_7: *Stability index*. Typically a number between 0.8 and 1, this metric describes the robustness of a code over longer periods of time.
- q_8: *Coverage testing*. This attribute describes how comprehensively each module of the code has been checked.
- q_9: *Fault density*. This measures the number of failures per 1000 lines of code. We express this as a fraction between 0 and 1.

The SDE index clearly seems to include, or be affected by, the other software-related attributes. We expect a good model will therefore be very sensitive to changes in q_5. In particular cases, these quality attributes may be restricted to tighter "operating ranges" by the company's production policies.

2.3 How do we Model the Reliability Constraints?

The optimization of the costs of quality and warranty would be straightforward in the absence of certain *reliability constraints*. These constraints are identified as benchmarks, or standards, which must be met by any product. The quality attributes must be chosen to meet or exceed these standards.

Prior to prescribing the nature of the constraints, we need to model the indicators of reliability which will be used. There are two major indicators, one for hardware and one for software.

HARDWARE:

> **Failure Rate** (FR): this is described by the *system* failure rate per year, and includes the effect of the component failure rate q_1.

SOFTWARE:

> **Severity Levels** (SL): ranges in scale from 1 to 4, where SL $= 1$ is a catastrophic failure, and SL $= 4$ is a minor error.

The reliability constraints will be interpreted in terms of these indicators – the failure rate FR must be below a certain prescribed value with high probability, and the severity level SL must stay away from the catastrophic failures with high probability. This is illustrated in expression (2.2).

2.3.1 Modelling the Failure Rate

The failure rate used in the characterization of reliability combines several factors including the failure rate of the components themselves, the robustness of the overall architecture, the infant mortality factor (IMF) and the working environment range.

We identified the broad trends that the failure rate exhibited in three of the quality attributes: component failure rate q_1, the infant mortality factor q_2 and the working environment range q_4. As the component failure rate q_1 increases, so does the overall failure rate. Likewise, if the IMF q_2 is high, the failure rate is large. The effect of the working range environment q_3 is opposite: if the constructed working range is larger than the specs, the device is more robust and thus the failure rate goes down.

We proposed two models with increasing complexity that exhibit this behaviour. Our discussions revealed that in this specific context the failure rate was described largely in terms of the component quality.

The first model FR$_1(\mathbf{q})$ is a simple one, with 3 free parameters f_1, f_2, and f_3:

$$\text{FR}_1(\mathbf{q}) = \text{FR}_1(q_1, q_2, q_4) = f_1\, q_1 + f_2\, q_2 - f_3(1 - q_1)q_4. \tag{2.3}$$

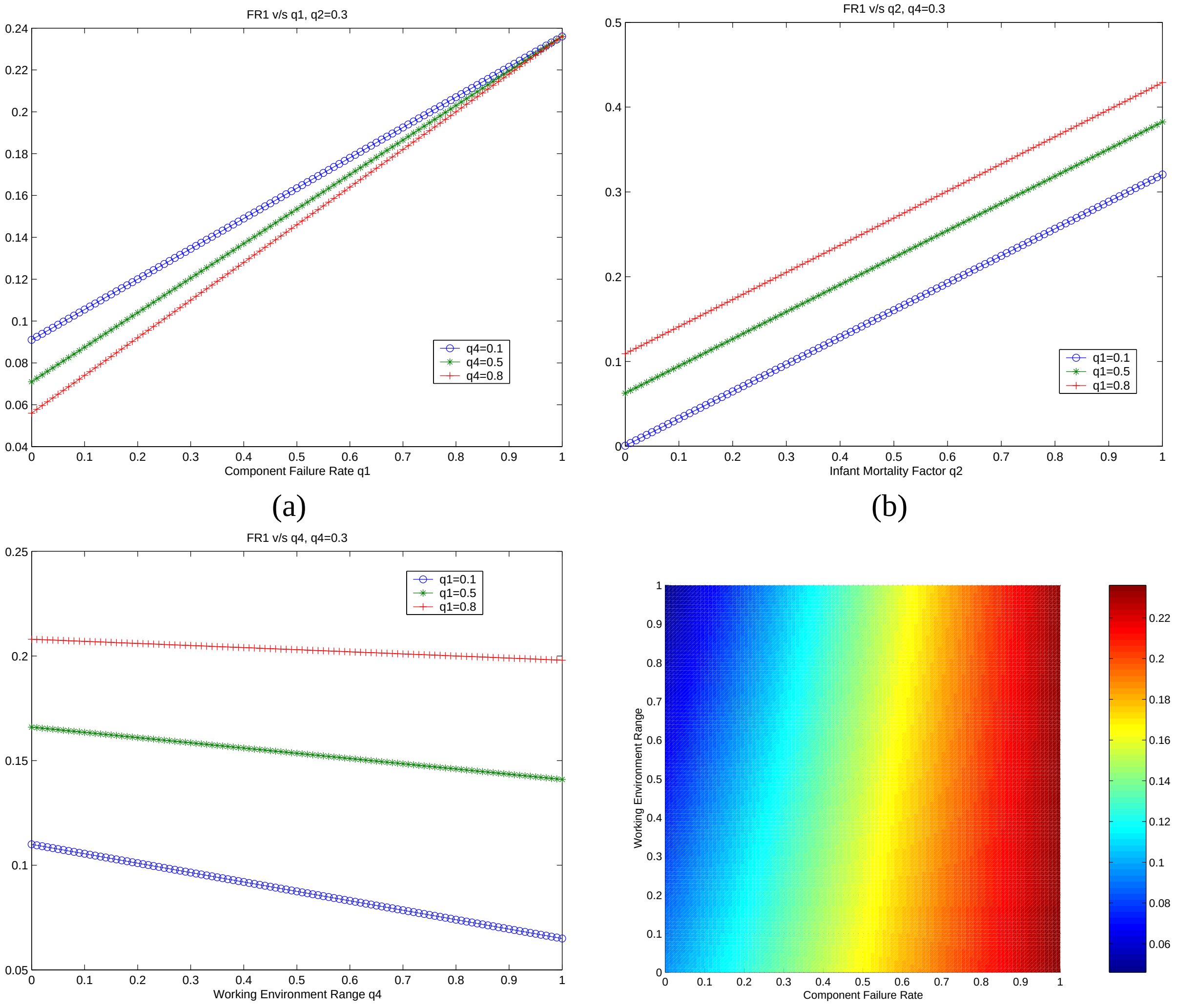
The nonlinear term $-f_3(1 - q_1)q_4$ enters since the failure rate should *decrease* with larger working environment range q_4, however the system will nevertheless be affected by poor component failure rates q_1. These two effects are therefore competing.

Figure 2.2 below shows four graphs related to failure rate model FR$_1$. The first three graphs exhibit the trends of the failure rate with respect to the individual attributes q_1, q_2, q_4. The last graph depicts a surface plot describing failure rate trends when q_1 and q_4 are allowed to change.

The next failure rate model we propose is manifestly nonlinear, and aims to better capture the importance of the component failure rate q_1 on the system failure rate FR. The free parameters are denoted f_1, f_2, f_3 and f_4. As before, the failure rate FR depends on the quality vector $\mathbf{q}$, but in particular on the attributes q_1, q_2, q_4.

$$\text{FR}_2(q_1, q_2, q_4) = f_1\, e^{f_2 q_1} + f_3\, q_2 q_1^2 - f_4(1 - q_1)q_4. \tag{2.4}$$

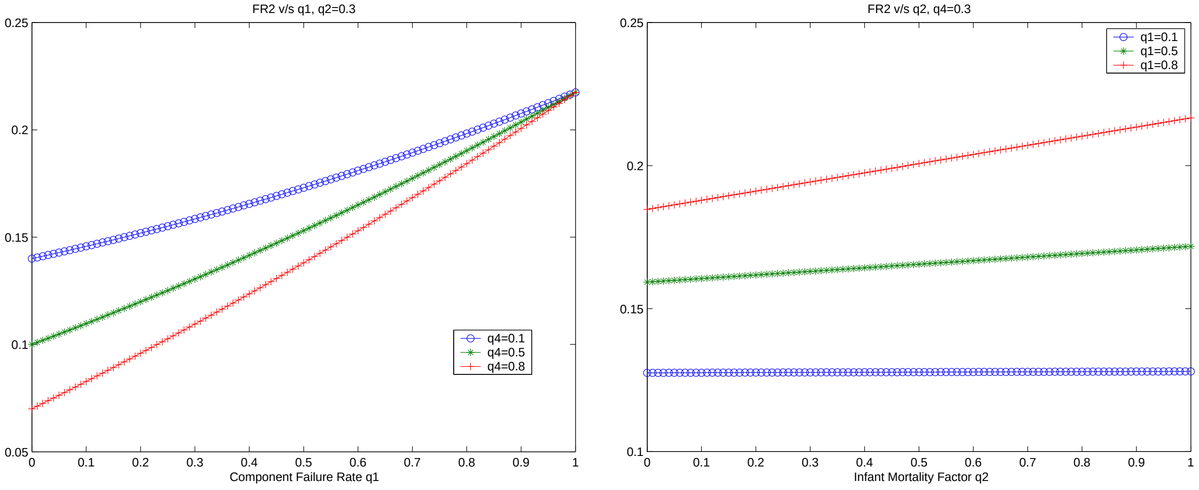
The rationale for picking this model is as follows: first, the system failure rate FR$(\mathbf{q})$ increases with poorer component quality, with this rate of change depending on q_1. Therefore the dependence of



(a) (b)

(c) (d)

Figure 2.2: Failure Rate model FR_1 as a function of (a) component quality q_1, (b) infant mortality q_2, (c) working environment range q_4, and (d) both q_1 and q_4 together, $q_2 = 0.3$.



(a) (b)

Figure 2.3: Failure Rate model FR_2 as a function of (a) component quality q_1, and (b) infant mortality q_2.

FR_2 on q_1 is modelled by an exponential. Second, the initial mortality rate q_4 impacts the overall failure rate, but even if this IMF is low, a poor-quality component will impact the failure rate adversely.

The two graphs in Figure 2.3 use the failure model (2.4) for FR to describe the broad trends in the model with component failure rate q_1 and infant mortality factor q_2 and can be compared to Figure 2.2. In Section 2.8 we show the effect of inputting several instances of $\mathbf{q}$, drawn from test data, into the model FR_2.

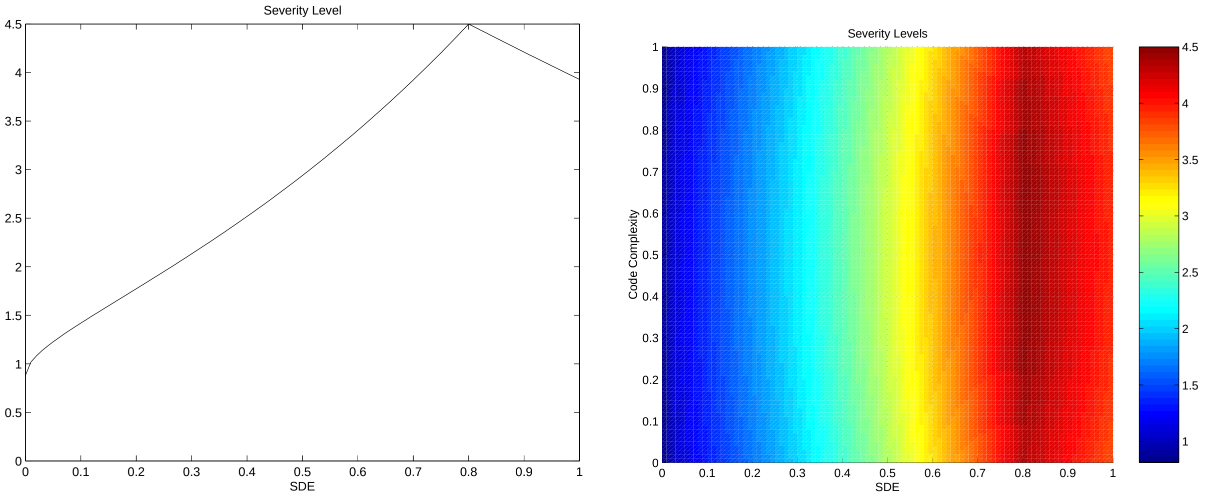
2.3.2 Modelling the Severity Level

Software failures are characterized in terms of varying *severity levels* (hereafter denoted SL), where an SL $= 1$ is a catastrophic failure, while an SL $= 4$ is a minor failure. In this section we present some models describing the relationship between the quality vector $\mathbf{q}$ and the SL.

In the context of this specific project, we determined that the severity levels of software failure were impacted by the software development environment q_5, the code complexity q_6, the stability index q_7, the coverage testing q_8 and the fault density q_9. The model we propose for the severity levels is not an additive/linear one. We believe that the chosen functional form captures well the trends in severity levels as functions of the individual attributes, as well as the relative importance amongst these factors. There are some free (nonnegative) parameters in the model, s_1, s_2, s_3, s_4, s_5. The severity level SL as a function of $\mathbf{q}$ is:

$$\mathrm{SL}_1(\mathbf{q}) = \mathrm{SL}_1(q_5, q_6, q_7, q_8, q_9) = s_1 e^{s_2|q_9-0.8|-s_3|q_8-0.1|}\left[\sqrt{q_5} + (1 + q_6 - q_6^2)^{s_4} q_7^2\right]. \tag{2.5}$$

To describe the effects of coverage testing q_8, we noted that as q_8 increases, the likelihood of catastrophic software error decreases since more of the software is validated. Similarly, as the number of faults per 1K lines, q_9, increases, so does the risk of catastrophic error. Keeping in mind the scale



(a) (b)

Figure 2.4: Severity levels as a function of the various components of the quality-related vector: (a) SDE q_5, (b) SDE and code complexity (q_5, q_6).

on which we measure SL, the dependence on q_8 and q_9 is modelled by exponentials with appropriate signs, penalizing deviations from high-quality.

Based on discussions, we modelled the dependence of SL on the stability index q_7 by a quadratic, since a more stable code is less prone to severe software failures.

As the software development environment indicator q_5 increases, the types of software failures get less severe and the SL increases. Poor quality development environment impacts the severity level more. That is, $\partial(\mathrm{SL})/\partial q_5$ should be larger for small values q_5. This behaviour is captured well by the square root function.

The opposite trend is exhibited as a function of code complexity q_6. When the code complexity is low, the overall software is less prone to severe errors, putting the SL index in the high range. After a certain threshold complexity is exceeded, the effect of complexity on the severity levels becomes less dramatic. To capture this behaviour, the dependence of SL on q_6 is described by $(1 + q_6 - q_6^2)^{s_4}$ where $s_4 < 1$.

While discussing SL it appeared that the attributes q_8, q_9, the coverage testing and fault density, were well-predicted by the software development environment, q_5. Therefore, we assumed that at least for the purpose of modelling severity levels as a function of $\mathbf{q}$, we could write

$$q_8 = K_8' q_5, \qquad q_9 = K_9' q_5. \tag{2.6}$$

This suggests a possible simplification to the severity level model:

$$\mathrm{SL}_2(\mathbf{q}) = \mathrm{SL}_2(q_5, q_6, q_7) = s_1 e^{-s_5|q_5-0.8|}\left[\sqrt{q_5} + (1 + q_6 - q_6^2)^{s_4} q_7^2\right]. \tag{2.7}$$

where s_1 and s_4 are as in model (2.5), while $s_5 = s_2 K_9' - s_3 K_8'$. The trends in the severity level are graphically described in the Figure 2.4.

Failure	(s_1, s_4, s_5)			
type	(2.46,0.4,1)	(2.46,0.6,1)	(2.46,0.4,2)	(2.46,0.4,3)
SL $\in (0, 1.5)$	0.1 %	0.01 %	0.01 %	0.02 %
SL $\in (1.5, 2.5)$	0.9 %	0.1 %	0.1 %	0.8 %
SL $\in (2.5, 3.5)$	12.7 %	14 %	29 %	43.5 %
SL $\in (3.5, 4.5)$	87.2 %	85.8 %	70.4 %	55.6 %

Table 2.1: Predicted distribution of severity levels for various sets of (s_1, s_4, s_5).

Even with this simplification, the severity model described by equation (2.7) is highly nonlinear. How is one to choose the exponent s_4? Does this model actually capture the observed behaviour of software systems when they are built within a given range of quality?

To answer these questions, we first determined the heuristic trend: if the software development environment q_5, the code complexity q_6 and the stability index q_7 were in the high-end, then the number of software failures classified as SL $= 1$ (catastrophic) should be less than 0.1%, SL $= 2$ failures should be about 1%, SL $= 3$ failures should be less than 10% and SL $= 4$ failures should be about 85%. A good reality check for our SL model (2.7) is to draw (q_5, q_6, q_7) from a given set of distributions. For our first simulation we take (q_5, q_6, q_7) from normal distributions with means $\mu_5 = 0.8$, $\mu_6 = 0.4$, $\mu_7 = 0.8$ and a common variance $\sigma^2 = 0.05$ so that $q_5 \sim N(0.8, 0.05)$, $q_6 \sim N(0.4, 0.05)$ and $q_7 \sim N(0.8, 0.05)$. The probability of each SL failure type can be computed through

$$\text{Probability of an SL type } i \text{ failure} = \frac{\mu\left\{(q_5, q_6, q_7) \in \Omega \,|\, \text{SL}(q_5, q_6, q_7) = i\right\}}{\mu\left\{(q_6, q_6, q_7) \in \Omega\right\}},$$

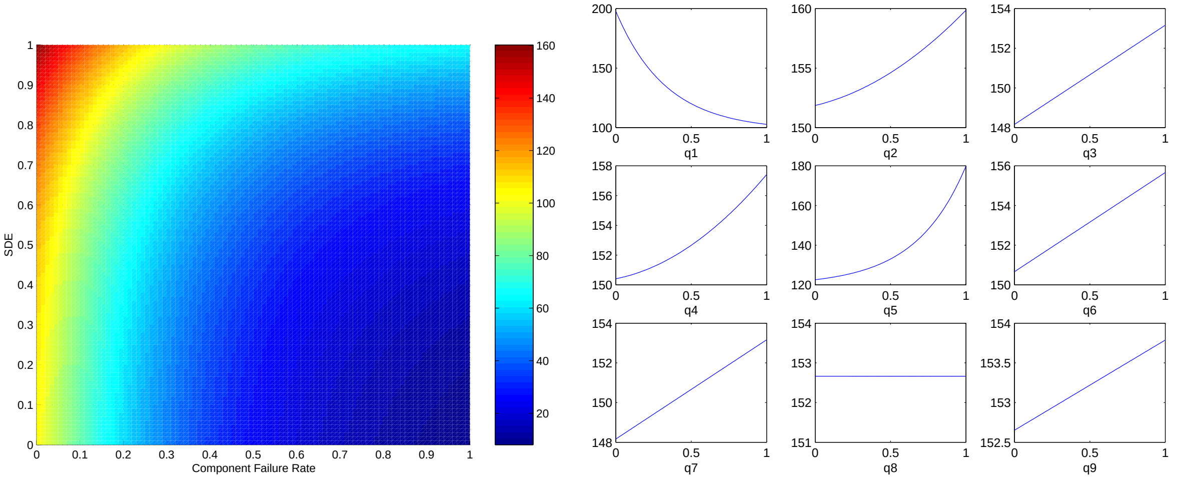
where $\Omega := \{(q_5^i, q_6^i, q_7^i)_{i=1}^{1000} | q_5 \sim N(0.8, 0.05), q_6 \sim N(0.4, 0.05), q_7 \sim N(0.8, 0.05)\}$ is a set of 1000 i.i.d. test data points drawn from the appropriate normal distributions, and $\mu(S)$ is the volume of a set S. We show in Table 2.1 these (approximate) percentages for a few choices of s_1, s_5 and, critically, s_5. We note that these ranges are not obtained from Lucent, but are used because they seem consistent with a high-end product. The code-complexity q_6 was set to be mid-range since a marketable system would have a certain minimal level of complexity, but high complexity was undesirable.

2.4 The Cost of Quality Implementation

Having identified the constraints in the previous section, we now describe the costs associated with building a product with given quality vector $\mathbf{q}$. Our discussion revealed that the largest effects on the cost were due to maintaining a high software development environment q_5, and a low component failure rate q_1.

The model for the cost of quality, $C(\mathbf{q})$ which is proposed in this project is

$$\left.\begin{aligned} C(\mathbf{q}) &= C_{hw} + C_{sw} \\ &= C_1 e^{-C_1' q_1} + c_2(q_2 + c_2')^2 + c_3 q_3 + c_4(q_4 + c_4')^2 + \text{fixed hardware costs} \\ &\quad + C_5 e^{C_5' q_5} + c_6 q_6 + c_7 q_7 + \frac{c_9 q_9}{q_6 + c_9'} + \text{fixed software costs}. \end{aligned}\right\} \quad (2.8)$$



(a) (b)

Figure 2.5: Behaviour of the quality function. (a) Trend in cost of quality as a function of component failure rate q_1 and SDE q_5. (b) Trend of cost of quality in individual attributes q_i.

Here C_1, C_1', C_5, C_5', c_2, c_2', c_3, c_4, c_4', c_6, c_7, c_9, c_9' are constant parameters which need to be determined by fitting actual data to the model.

The first term of the hardware and software portions in this model capture the importance of the component failure rate q_1 and the software development environment q_5 in the overall cost model. As q_1 decreases, the overall likelihood of failure decreases. This improvement costs more, especially after a certain threshold is achieved. Improvements beyond this level are increasingly expensive, as captured by an exponential function with the negative exponent. On the other hand, as SDE q_5 increases, the software development environment becomes better and thus costs more. These trends are captured in Figures 2.5(a) and (b).

The terms collected in equation (2.8) under *hardware* describe the effects of the infant mortality rate q_2, the diagnostics capability q_3 and the working environment range q_4. In terms of the overall hardware quality costs, these are higher-order effects in the sense that their contribution may not be as significant as that of the component failure rate, q_1. This reasoning dictated the functional relationships as being at best quadratic. Similarly, the costs collected under *software* describe the effects of controlling the code complexity q_6 and the stability index q_7. These costs contribute less significantly to the overall quality costs for the software than the software development environment q_5. Indeed, the effects of increasing the coverage testing and decreasing the fault density are captured (to a large extent) by the cost of q_5, and are therefore ignored in this cost model. The trends of the cost as each of these attributes vary is pictured graphically in Figure 2.5. The results using the test data are described in Section 2.8.

2.5 The Warranty Costs

Warranty costs can be broadly broken up into the hardware and software costs, and we thus modelled each of these separately. As in the cost of quality, the dominant factors involved are the component failure rates q_1 and the software development environment q_5. We now present a model for the warranty costs $W(\mathbf{q})$ expressed as

$$W(\mathbf{q}) = W_{hw}(\mathbf{q}) + W_{sw}(\mathbf{q}).$$

In this project we do not attempt to present a model for finding an optimal warranty policy. Our attempt is to model the effect of changing quality on warranty costs for a *given, fixed* warranty policy. This distinction is an important one. Clearly, the policies themselves will change considerably if the average product quality attributes change a lot. This model does not account for this, at least in the hardware costs. However, as a first approximation, if we assume the $\mathbf{q}$ vector stays within a certain range, the warranty policy may be considered fixed, and we can describe the effects on the warranty *costs* of changing $\mathbf{q}$ within this range.

2.5.1 The Hardware Warranty Costs

Hardware warranty costs are characterized by the four major types of hardware failures seen: *no trouble found* (NTF), *repaired*, *junked*, and *further failure modes analysis* (FMA). Of these, the NTF costs are the smallest, but the supplier may wish to penalize these. The model should be flexible enough so that an optimization will ensure that most of the errors fall into the second (repaired) category.

Empirical data will be able to describe the observed probabilities p_1, p_2, p_3 and p_4 of seeing the various warranty-related costs (NTF, repair, junk, and FMA, respectively). These probabilities are computed based on the assumption that the quality vector $\mathbf{q}$ lies within a particular range, but are not sensitive to variations within the range. There is also a standardized dollar amount w_1, w_2, w_3, w_4 associated with each of these.

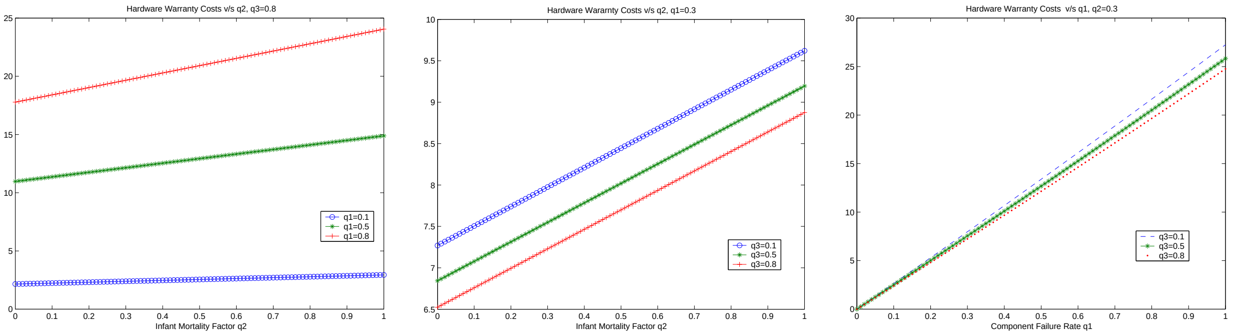
The standard warranty cost model simply computes the expected cost $E(C) = \sum_{i=1}^{4} p_i w_i$. As a result this standard model fails to capture the most important trend, that of changing component failure rate q_1, on the hardware warranty costs.

To include the effects of quality, we *penalize* these four kinds of failures to various degrees. This will allow the user, for example, to explicitly adjust the model so that NTF failures are reduced. This allows for greater flexibility in optimization. For example, NTF failures are inexpensive, and thus optimizing a standard warranty cost model may result in choosing $\mathbf{q}$ values which *increase* NTF failures, while reducing the expensive FMA failures. By contrast, the proposed hardware warranty model (2.9) allows the user to choose the penalty parameters k_1, k_2, k_3, k_4 so that the optimal $\mathbf{q}$ values result in a small number of NTF.

$$W_{hw}(\mathbf{q}) = w_0 q_1 \Big\{ \underbrace{p_1[w_1 + k_1(1-q_3)]}_{\text{NTF costs}} + \underbrace{p_2(w_2 + k_2 q_2)}_{\text{repair costs}} + \underbrace{p_3(w_3 + k_3 q_1)}_{\text{junk costs}} + \underbrace{p_4[w_4 + k_4(1-q_3)]}_{\text{FMA}} \Big\} \tag{2.9}$$

where $p_1 = 0.1$, $p_2 = 0.8$, and $p_3 = p_4 = 0.05$.

These probabilities are based on how many items which are returned to the supplier (and are ad hoc), while w_0 is used to scale costs appropriately by the number of items in service. We notice that the standard warranty cost model has been multiplied by q_1, the component failure rate. In the proposed model if the component failure rates q_1 are at the high end of the operating range, then the warranty costs are higher. Figure 2.6 depicts the various dependencies.



(a) (b) (c)

Figure 2.6: (a) Trends in W_{hw} with component quality q_1 and IMF q_2, (q_3 fixed). (b) Trends in W_{hw} with IMF q_2 and diagnostics capability q_3, (q_1 fixed). (c) Trends in W_{hw} with component quality q_1 and diagnostics capability q_3, (q_2 fixed).

2.5.2 The Software Warranty Costs

The software warranty costs are characterized in terms of the severity levels of the software failures, SL (see Section 2.3.2). Therefore, given the operating range of the quality vector $\mathbf{q}$, we can calculate from our SL model the probabilities p_5, p_6, p_7, p_8 of failures of SL $= 1$ (catastrophic) through SL $= 4$ (minor) respectively (see Table 2.1). Associated with each of these types of failures is a warranty cost, w_5, w_6, w_7 and w_8 respectively. The warranty cost for software is expressed by:

$$W_{sw}(\mathbf{q}) = \widetilde{W}(1 - q_5)(p_5 w_5 + p_6 w_6 + p_7 w_7 + p_8 w_8). \tag{2.10}$$

The SDE most significantly impacts the warranty cost of the software. A higher SDE means fewer catastrophic failures. The probabilities p_i, $i = 5, \ldots, 8$ are *fixed* for a given operational range of q_5, q_6, q_7, and are chosen according to our model of the Severity Level function (2.7). Recall that q_8 and q_9 are given by expression (2.6). More precisely,

$$p_{i+4} = \frac{\mu \left\{(q_5, q_6, q_7) \in \Omega \,|\mathrm{SL}(q_5, q_6, q_7) = i\right\}}{\mu \left\{(q_6, q_6, q_7) \in \Omega\right\}} + \epsilon_i$$

where $\Omega := (0.8, 1) \times (0.4, 0.7) \times (0.7, 1)$. The quantities ϵ_i are introduced to account for other lower-order effects, which are not accounted for during the production. These include effects such as a product being returned due to incorrect usage by the customer. One may also use a model allowing more flexibility, such as (2.9). Unfortunately, in such a model the probabilities will need to be computed as functions of q_i.

2.6 The Complete Model

We now summarize the models of the previous sections. Recall that the goal was to identify the various functions in the optimization problem (2.1) repeated here for convenience

$$\mathcal{F}_{opt} = \min_{\mathbf{q}}(C(\mathbf{q}) + W(\mathbf{q})),$$

subject to

$$P(\mathrm{FR}(\mathbf{q}) < \mathrm{FR}_{\mathrm{max}}) < p, \qquad P(\mathrm{SL}(\mathbf{q}) > \mathrm{SL}_{\mathrm{min}}) < p$$

over all admissible $\mathbf{q}$ and some preset probability p. The objective function consists of the combined costs of quality (2.8) and warranty (2.9), (2.10):

$$\left.\begin{aligned} \mathcal{F} &= C(\mathbf{q}) + W(\mathbf{q}) = C(\mathbf{q}) + W_{hw}(\mathbf{q}) + W_{sw}(\mathbf{q}) \\ &= C_1 e^{-c_1' q_1} + C_5 e^{c_5' q_5} + w_0 q_1 \sum_{i=1}^{4} p_i w_i + \widetilde{W}(1-q_5) \sum_{i=5}^{8} p_i w_i \\ &\quad + c_2(q_2 + c_2')^2 + c_3 q_3 + c_4(q_4 + c_4')^2 + c_6 q_6 + c_7 q_7 + c_9 \frac{q_9}{q_6 + c_9'} \\ &\quad + w_0 q_1 \left[p_1 k_1 (1-q_3) + p_2 k_2 q_2 + p_3 k_3 q_1 + p_4 k_4 (1-q_3) \right] + \text{fixed costs} \end{aligned}\right\} \tag{2.11}$$

where the terms have been reordered. The reliability constraints are given by (2.4) and (2.7):

$$\begin{aligned} \mathrm{FR}_2(q_1, q_2, q_4) &= f_1\, e^{f_2 q_1} + f_3\, q_2 q_1^2 - f_4(1-q_1) q_4 < \mathrm{FR}_{\mathrm{max}} \\ \mathrm{SL}_2(q_5, q_6, q_7) &= s_1 \sqrt{q_5} e^{s_5 q_5} (1-q_6)^{s_4} q_7^2 > \mathrm{SL}_{\mathrm{min}} \end{aligned}$$

where $\mathrm{FR}_{\mathrm{max}}$ and $\mathrm{SL}_{\mathrm{min}}$ are specified by the user. One may also use other models proposed in Section 2.3. This objective function is quite complicated. However, by retaining only terms to leading order, we propose a simpler model $\widetilde{\mathcal{F}}$ which captures most of the behaviour:

$$\widetilde{\mathcal{F}}(\mathbf{q}) = c_1 e^{-c_1' q_1} + c_5 e^{c_5' q_5} + \widetilde{W}_1 q_1 + \widetilde{W}_2 (1-q_5).$$

This simplification is justified numerically in Section 2.8.2.

2.7 Model Justification: Sensitivity to Parameters

Consider a function $\mathcal{G}(\mathbf{q}; f_1, f_2, \ldots, f_N)$ depending on N continuously varying parameters on some domain Ω. We assume that $\mathcal{G}$ is sufficiently regular to ensure that $\partial\mathcal{G}/\partial f_j$ exists throughout Ω. By considering the sequence of parameters $\{f_j\}_{j=1}^N$ as a vector $\vec{f} \in \mathbb{R}^N$ the total differential of $\mathcal{G}$ can be written as

$$d\mathcal{G} = \nabla_{\vec{f}} \cdot d\vec{f} = \sum_{j=1}^{N} \frac{\partial \mathcal{G}}{\partial f_j}\, df_j.$$

As a consequence, assuming $\mathcal{G}$ is positive valued[7], one has

$$\frac{d\mathcal{G}}{\mathcal{G}} = \sum_{j=1}^{N} |f_j| \frac{\partial \log \mathcal{G}}{\partial f_j} \frac{df_j}{|f_j|} \tag{2.12}$$

illustrating that the proportion of the relative change in the $\mathcal{G}$ due to the relative change in the parameter f_j is $f_j \partial \log \mathcal{G}/\partial f_j$. This simple formalism is used in the analysis that follows.

[7] The general expression is $\frac{d\mathcal{G}}{|\mathcal{G}|} = \sum_{j=1}^{N} |f_j| \mathrm{sgn}(\mathcal{G}) \frac{\partial \log |\mathcal{G}|}{\partial f_j} \frac{df_j}{|f_j|}$.

2.7.1 Failure Rate Model

Recall that the second model for overall failure rate that we proposed in Section 2.3 was

$$\mathrm{FR}_2(\mathbf{q}; f_1, f_2, f_3, f_4) = f_1 e^{f_2 q_1} + f_3 q_2 q_1^2 - f_4(1-q_1)q_4$$

where f_1, f_2, f_3, f_4 are positive constants. Because we have explicitly specified the dependence of FR on the parameters f_i expression (2.12) allows one to estimate which of these parameters have the most impact on the model. Indeed,

$$\frac{d\mathrm{FR}_2}{|\mathrm{FR}_2|} = \frac{(df_1 + f_1 q_1\, df_2)e^{f_2 q_1} + q_2 q_1^2\, df_3 - (1-q_1)q_4\, df_4}{|f_1 e^{f_2 q_1} + f_3 q_2 q_1^2 - f_4(1-q_1)q_4|}.$$

We claim that the effect of changing f_1 or f_2 by an amount δ has a larger effect than a similar change in f_3, f_4. To see this, note that perturbations to f_1 and f_2 are amplified by a factor of $e^{f_2 q_1} > 1$ and $q_1 e^{f_2 q_1} \geq q_1$ respectively, whereas perturbations to f_3 and f_4 are only amplified by factors $q_2 q_1^2 \leq q_1 < 1$ and $(1-q_1)q_4 \leq q_1 < 1$.

2.7.2 Severity Level Model

In order to gauge which of the free parameters most significantly affect the second SL model (2.7), we fix $\mathbf{q}$, and compute $\log(\mathrm{SL}_2) = \log s_1 + (s_5 q_5 + \frac{1}{2}\log q_5) + s_4 \log(1-q_6) + 2\log q_7$, which helps us identify the relative sensitivity of the model to the parameters s_1, s_4 and s_5:

$$\frac{d\mathrm{SL}_2}{\mathrm{SL}_2} = \frac{ds_1}{s_1} + s_5 q_5 \frac{ds_5}{s_5} + s_4 \log(1-q_6)\frac{ds_4}{s_4}.$$

Thus, if all other parameters are fixed, a 1% change in s_5 will result in a $s_5 q_5$% change in the SL value. If s_4 is changed by 1%, the resultant percentage change in the SL values is $s_4 \log(1-q_6)$. Heuristically the q_6 values range between 0.4 and 0.7, and therefore $\log(1-q_6)$ is negative. Increasing s_4 thus results in a decreasing SL. This also explains our findings regarding Table 2.1.

2.7.3 Cost of Quality Implementation Model

Following the same techniques as in the previous two subsections, we examine the model $C(\mathbf{q})$, given by equation (2.8) for the relative importance of the parameters

$$\vec{C} = \langle C_1, C_1', C_5, C_5', c_2, c_2', c_3, c_4, c_4', c_5, c_6, c_7, c_9, c_9' \rangle.$$

Continuing with our prescription yields

$$\begin{aligned} dC(\mathbf{q}; \vec{C}) \;\; = \;\; & (dC_1 - q_1 C_1\, dC_1')e^{-C_1' q_1} + (dC_5 + q_5 C_5\, dC_5')e^{C_5' q_5} \\ & + (q_2 + c_2')\,[2c_2\, dc_2' + (q_2 + c_2')\, dc_2] + (q_4 + c_4')\,[2c_4\, dc_4' + (q_4 + c_4')\, dc_4] \\ & + q_3\, dc_3 + q_6\, dc_6 + q_7\, dc_7 + \frac{c_9 q_9}{(q_6 + c_9')^2}\left[(q_6 + c_9')\frac{dc_9}{c_9} - dc_9'\right]. \end{aligned}$$

2.8 Test Data, and Model Trends

The validation of a proposed model is an important step in any modelling effort. In the absence of real data from Lucent, we were unable to specify the nature of distributions from which to generate test data. Indeed, test data should be created on the basis of *hazard models* appropriate for the products. In the absence of these models, any test data used is for illustration purposes only.

As part of this project, we provide two test sets of data, drawn from a normal distribution (the mathematical interpretation of the popular Six-Sigma model) and from a Beta distribution. To illustrate the broad trends of our models, we generated several instances of $\mathbf{q}$, with individual attributes picked as independent random variables drawn from these two models.

2.8.1 Test Data Drawn from a Normal Distribution

Test data of 1000 instances of $\mathbf{q}$ was created by considering the attributes q_i as independent random variables drawn from appropriate normal distributions $N(\mu_i, \sigma_i^2)$ (mean μ_i, variance σ_i^2), with the distribution parameters chosen to reflect a high quality product. Figure 2.7 illustrates a particular instance of this process. Note that the attributes q_i are scaled to reflect the *natural* quantities, e.g., component failure rate q_1 is scaled by 100 to yield a failure rate percentage. We input our simulated data $\mathbf{q}$ into the failure rate model FR. The results are described in Figure 2.8.

We expect that with most products being built to high quality specifications, the warranty costs will be low, while the cost of implementation will be high. Figure 2.9 illustrates the histograms of the warranty costs W_{sw}, W_{hw}, and the implementation costs $C(\mathbf{q})$ when this instance of test data is applied to each of the relevant models.

2.8.2 Test Data Drawn from a Beta Distribution

A beta distribution was chosen because it is a two parameter distribution defined on the interval $[0, 1]$. The probability distribution function is given by

$$p(x; \alpha, \beta) = \begin{cases} \dfrac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1} & 0 \le x \le 1 \\ 0 & \text{otherwise} \end{cases}$$

where the mean and variance are

$$\mu = \frac{\alpha}{\alpha+\beta}, \qquad \sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}.$$

So as to match with the corresponding normal distribution $N(\mu_i, \sigma_i^2)$ one chooses the parameters α and β as

$$\alpha = \frac{\mu_i}{\sigma_i^2}\left(\mu_i - \mu_i^2 - \sigma_i^2\right), \qquad \beta = \left(\frac{1}{\mu_i} - 1\right)\alpha.$$

Once again test data of 1000 instances of $\mathbf{q}$ was created and the simulated data applied to the failure rate model FR. The results are described in Figure 2.10. Since we assumed the attributes were drawn from a beta distribution (most instances are high quality), it is reasonable that the failure rate is skewed

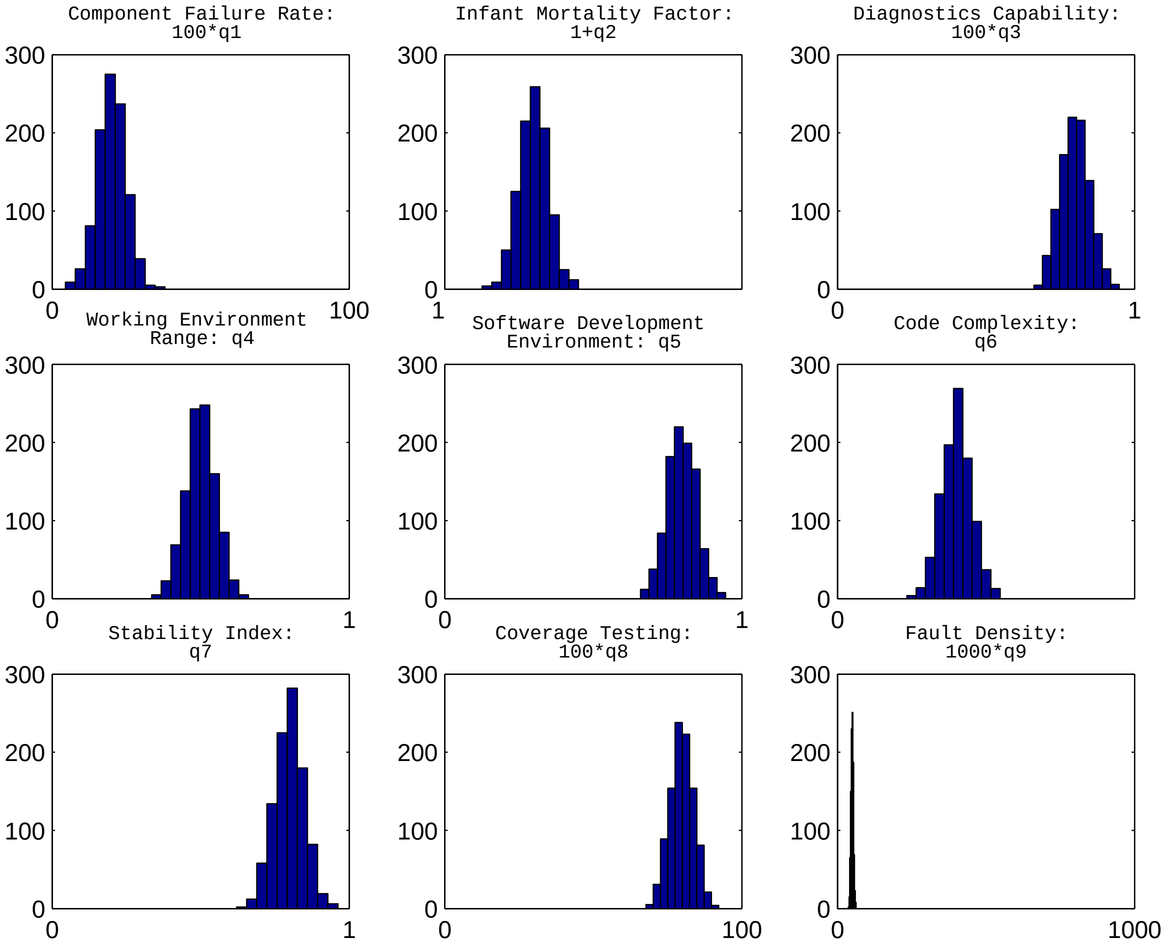


Figure 2.7: Simulated data drawn from normal distributions. In detail, $q_1 \sim N(0.2, 0.05)$, $q_2 \sim N(0.3, 0.05)$, $q_3 \sim N(0.8, 0.05)$, $q_4 \sim N(0.5, 0.05)$, $q_5 \sim N(0.8, 0.05)$, $q_6 \sim N(0.4, 0.05)$, $q_7 \sim N(0.8, 0.05)$, $q_8 \sim N(0.8, 0.04)$, $q_9 \sim N(0.05, 0.004)$. Any normalization to the respective variables is indicated.

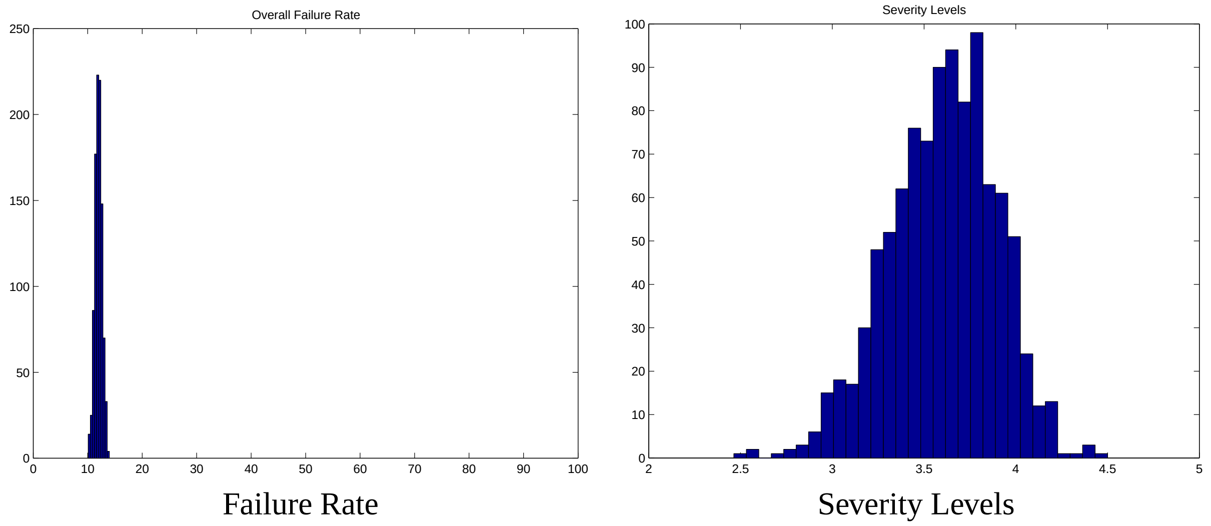


Figure 2.8: Effect of FR_2 and SL_2 on simulated data using a normal distribution: most instances of the product have a low failure rate.

towards the lower end. We expect that with most products being built to high quality specifications the warranty costs will be low while the cost of implementation will be high. This is borne out in Figure 2.11.

In Figure 2.12 we see that at least for this test data, the simplified objective function $\widetilde{\mathcal{F}}$ described in Section 2.6 captures most of the behaviour of the complicated objective function (2.11).

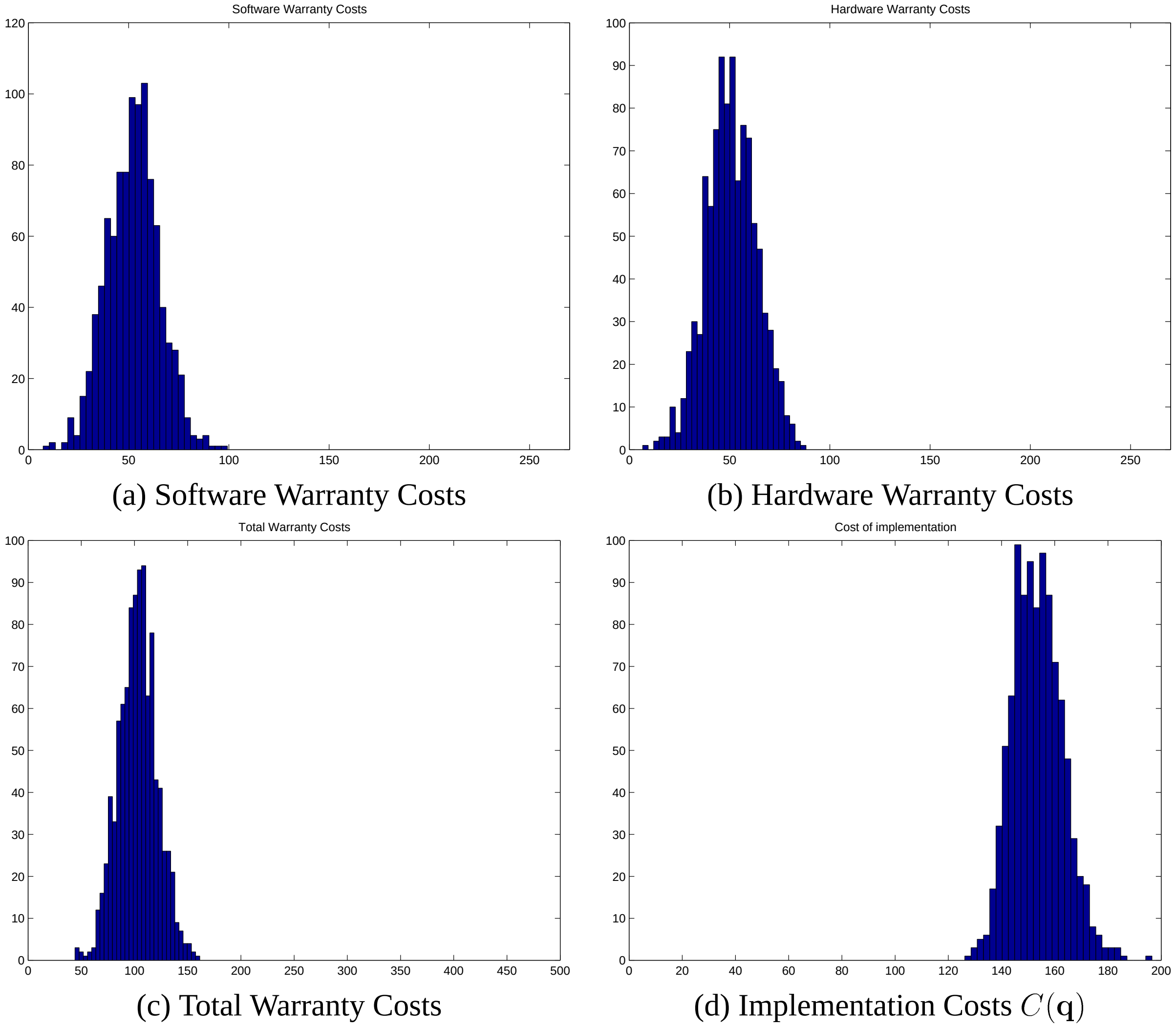
2.9 Summary, Future Directions and Suggestions

We conclude this report by noting again that the models developed were based solely on discussions and heuristic arguments. In the absence of data, survival and hazard models, indeed even product information from Lucent, this report should not be interpreted as representative. Instead, we hope that the arguments will provide the basis for a more careful modelling effort by Lucent.

We note that despite the identification of nine quality attributes q_i, not all of these attributes are equally important. This is a crucial step in any modelling process: identifying the key elements. Based on the preceding discussion we can conclude that the most significant *independent* attributes are the *component failure rates* q_1 and the *software development environment* q_5. These indices seem to outweigh the others. In fact, most of the other attributes are affected by these two. Thus, any further work should focus on the careful estimation of these attributes. The attributes associated with diagnostics capability q_3, coverage testing q_8 and fault density q_9 are the least significant. Indeed, these do not even appear in the constraints.

Neither the constraint functions FR and SL nor the objective function contain complicated functional forms; the resultant model is nonlinear and awkward, but none of the individual components is more complicated than a quadratic or an exponential. These forms are deliberately chosen since the associated parameters can be easily fit, using real data and standard statistical software.

We suggest that the parameters be located based on true data which may be available to the industry.



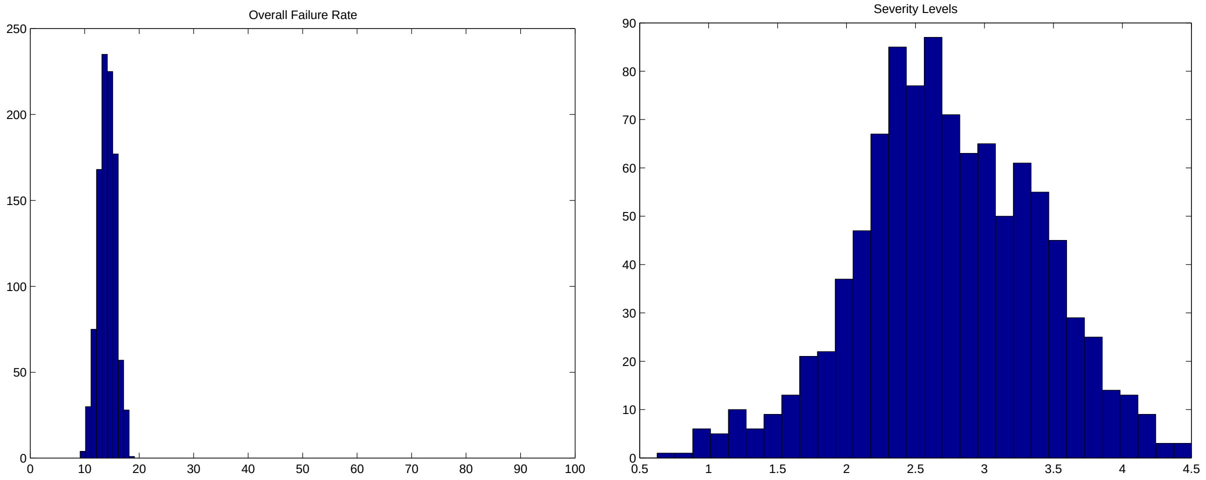
(a) Software Warranty Costs

(b) Hardware Warranty Costs

(c) Total Warranty Costs

(d) Implementation Costs $C(\mathbf{q})$

Figure 2.9: Warranty Costs and Implementation Costs using normally distributed test data. Scales range from 0 to maximum possible cost in each case.

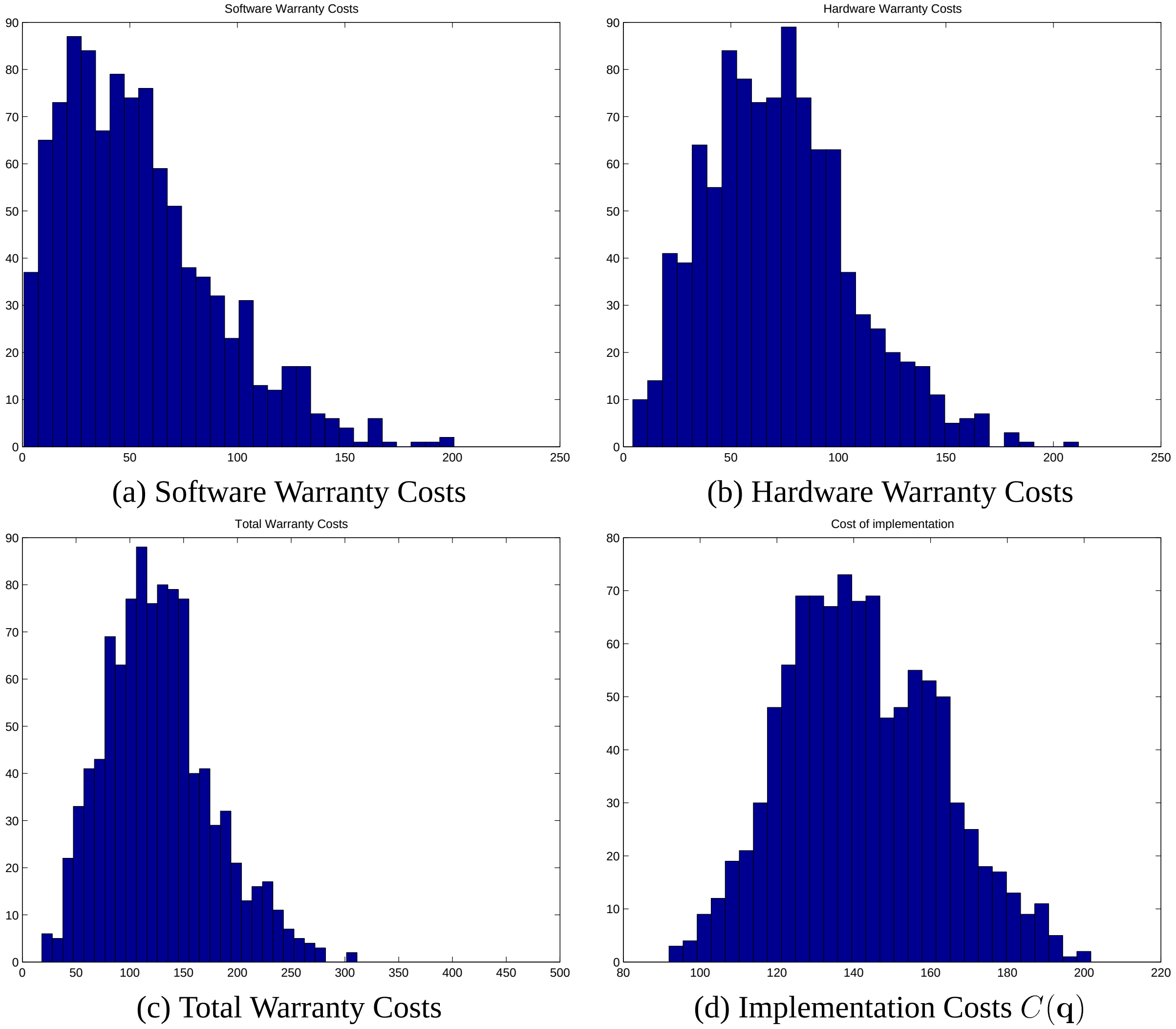


Failure Rate

Severity Levels

Figure 2.10: Effect of FR_2 and SL_2 on simulated data using a beta distribution: most instances of the product have a low failure rate.

These parameters will vary with various product and warranty policies. Simultaneously, test data drawn from hazard models appropriate to the specific product should be used as a reality check. The final optimization can be carried out using standard packages.



(a) Software Warranty Costs

(b) Hardware Warranty Costs

(c) Total Warranty Costs

(d) Implementation Costs $C(\mathbf{q})$

Figure 2.11: Warranty Costs and Implementation Costs using beta distributed test data. Scales range from 0 to maximum possible cost in each case.

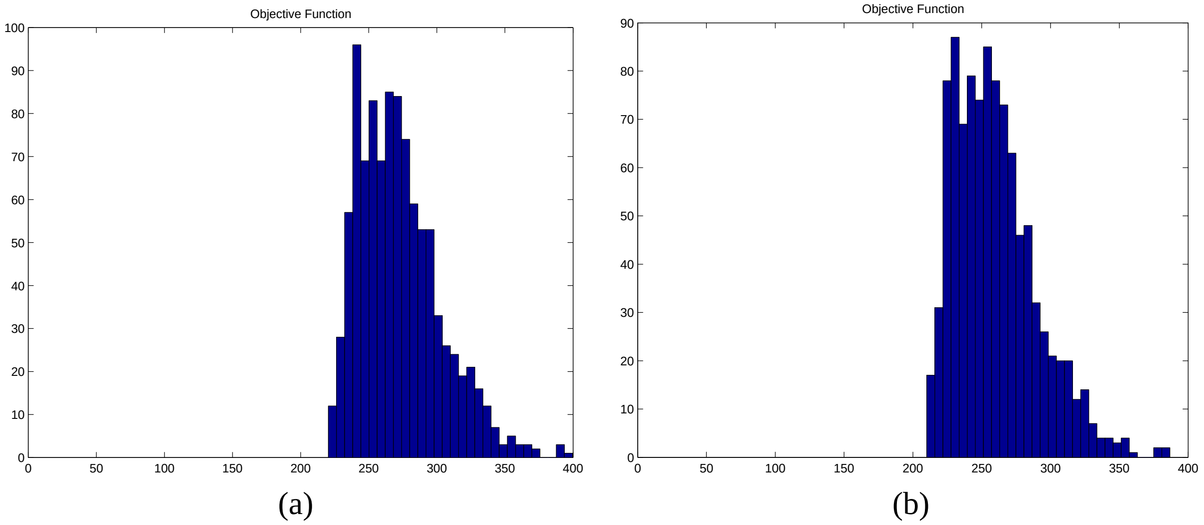


Figure 2.12: (a) Objective function $\mathcal{F}(\mathbf{q})$. (b) Simplified objective function $\widetilde{\mathcal{F}}(\mathbf{q})$.

Bibliography

[1] Mendiratta, V.B. (2002). Modeling Quality and Warranty Costs. http://www.pims.math.ca/industrial/2003/ipsw/Lucent.pdf

Chapter 3

In-Situ Thermal Remediation of Contaminated Soil

Katharina Baamann[1], Charles Bergeron[2], Thalya Burden[3], Huaxiong Huang[4], Samet Kadioglu[5], Serguei Lapin[6], Bruce McGee[7], Anuj Mubayi[8], Juan Restrepo[9], Andrew Taylor[10], Rex Westbrook[10]

Report prepared by Huaxiong Huang (hhuang@yorku.ca),[11] Serguei Lapin (slapin@math.uh.edu) and
Rex Westbrook (westbroo@math.ucalgary.ca)

3.1 Background

Recently, a method for removing contaminants from soil (several meters under the ground) has been proposed by McMillan-McGee Corp. The process can be described as follows. Over a period of several weeks, electrical energy is introduced to the contaminated soil using a multitude of finite length cylindrical electrodes. Current is forced to flow through the soil by the voltage differentials at the electrodes. Water is also pumped into the soil via the injection well and out of the ground at the extraction well. The soil is heated up by the electrical current and the contaminated liquids and vapours are produced at the extraction well. The temperature of the contaminated soil, during the process, is believed to reach the maximum value (the boiling temperature of water). Normally, the electrodes are placed around the contaminated site and the extraction well is located in the centre of the contaminated region. The distance between the electrodes is usually seven to eight meters. The distance between the

[1] Georgia Institute of Technology

[2] École Polytechnique de Montréal

[3] University of Kentucky

[4] York University

[5] Florida State University

[6] University of Houston

[7] McMillan-McGee Corp.

[8] University of Texas-Arlington

[9] University of Arizona

[10] University of Calgary

[11] HH wishes to thank Sean Bohun for his help when the report was prepared. He read through our first draft and cleaned up the non-dimensionalization procedure. HH also wishes to thank NSERC and PIMS for their financial support.

extraction well and an electrode is about four meters. The diameter of the electrodes is 0.2 meter and the extraction well is 0.1 meter in diameter.

The reason for using the electrical current is that "flushing" the soil using water alone is not effective for removing the contaminants. By heating up the soil and vaporizing the contaminated liquid, it is anticipated that rate of extraction will increase as long as the recondensation is not significant. A major concern, therefore, is whether recondensation will occur. Intuitively, one might speculate that liquid phase may dominate near the injection well. Moving away from the injection site towards the extraction well, due to the combined effects of lower pressure and higher temperature (from heating), phase change occurs and a mixture of vapour and liquid may co-exist. There may also be a vapour-only region, depending on the values of temperature, pressure, and other parameters. In the two-phase zone, since vapour bubbles tend to rise due to the buoyancy force, and the temperature decreases along the vertical path of the bubbles out of the heated region, it is possible that the bubbles will recondense before reaching the extraction well. As a consequence, the probability exists that part of the contaminants stay in the soil. Obviously, to predict transition between single-phase and two-phase regions and to understand the transport phenomenon in detail, a thermal capillary two-phase flow model is needed. However, to simplify the problem, here we only consider the case when two-phases co-exist in the entire region.

The main objective of this modelling exercise is to determine the necessary vacuum pressure (pressure drop from the electrodes to the extraction well) so that the chemical bubbles are removed at the extraction well before they rise too high and condense to the liquid state.

3.2 Flow and Temperature Fields

To make the problem tractable, we consider an idealised situation where the extraction well and an electrode are both placed at the centre of a circle and the current as well as the mixture of liquid and vapour are flowing towards the centre. The domain of interest becomes a cylindrical region with the extraction well and an electrode at the centre. To further simplify the problem, we assume radial symmetry and the electrical current is in the radial direction only. Even with these simplifications, the problem at hand is still a complicated one and in principle a multi-phase flow model will be an appropriate starting point. However, we take a simplistic approach in this report by decoupling the complicated process into several sub-processes.

First of all, since the main components in the system are water and water vapour, we will not distinguish various components in the system and treat it as a one-component system with two phases: liquid and vapour. Secondly, we assume that the two-phase flow under consideration falls into the slug flow regime since the flow rates are relatively low - typically in the range of 10^{-2} m^3/s. As a result, in the horizontal direction r, the vapour (generated by the heating) moves with the liquid phase. Therefore, we will not distinguish the two phases and a single-phase model will be used with a common radial mixture discharge velocity u. Furthermore, the mass exchange happens mainly in the r direction due to an applied pressure drop between the extraction well and the injection well. Therefore, conservation of mass will be applied to the horizontal velocity component only. In the vertical z direction the bulk of the liquid phase is at rest, except the part displaced by vapour due to buoyancy force. Following [4], the r-component of the velocity is determined by the Darcy's law

$$u = -\frac{k}{\mu}\frac{\partial P}{\partial r} \tag{3.1}$$

where k is the effective permeability, μ is the effective viscosity, and P is the effective pressure. The mass conservation (assuming that the vertical z component is small) can be written as

$$\frac{\partial \rho_f}{\partial t} + \frac{1}{r}\frac{\partial}{\partial r}(ru\rho_f) = 0. \tag{3.2}$$

The mixture density ρ_f and r-velocity u are defined as

$$\rho_f = \alpha\rho_l + (1-\alpha)\rho_v, \quad \rho_f u = \alpha\rho_l u_l + (1-\alpha)\rho_v u_v$$

where α is the liquid volume fraction (saturation), u_l and u_v are the liquid and vapour velocity components in the r-direction, ρ_l and ρ_v are the liquid and vapour densities.

The temperature of the soil, liquid and vapour mixture is determined by the energy conservation law

$$\rho c\frac{\partial T}{\partial t} + \rho_f c_f u\frac{\partial T}{\partial r} = \frac{1}{r}\frac{\partial}{\partial r}\left(\kappa r\frac{\partial T}{\partial r}\right) + \sigma|E|^2 - LM, \tag{3.3}$$

where $\sigma|E|^2$ is the Joule heating of the soil with an the electrical resistivity σ and electric field E. The term LM represents the heat lost in the formation of the bubbles with L is the latent heat of vapourization of the fluid and M the mass rate of vapourization. The soil fluid mixture is characterized by a heat capacity ρc and thermal conductivity κ. Note that we have assumed that there is no temperature variation in the vertical z direction.

Now, we use dimensional analysis to further simplify (3.1)-(3.3), by keeping the dominant terms. With this in mind we make the following assignments:

$$P = P_0 + \Delta P\hat{P}, \;\; r = x\hat{r}, \;\; t = \beta\hat{t}, \;\; u = \frac{x}{\beta}\hat{u}$$

where the quantities with hats are dimensionless and the collection $\{\Delta P, x, \beta\}$ are representative values. Under this assignment the expression for the pressure becomes

$$\frac{\partial\hat{\rho}_f}{\partial\hat{t}} - \frac{\beta k}{\mu}\frac{\Delta P}{x^2}\frac{1}{\hat{r}}\frac{\partial}{\partial\hat{r}}\left(\hat{r}\hat{\rho}_f\frac{\partial\hat{P}}{\partial\hat{r}}\right) = 0.$$

Representative values for the various quantities can be found in Table 3.1. Typical orders of magnitude in SI units [8] are $x \sim 10$, $\beta \sim 10^5$, $k \sim 10^{-9}$, $\mu \sim 10^{-3}$ and $\Delta P \sim 10^5$ yielding $\beta k\Delta P/(\mu x^2) \sim 10^2 \gg 1$. Consequently the temporal variations of $\hat{\rho}_f$ can be ignored to first approximation giving a pressure field expression[12]

$$\frac{1}{\hat{r}}\frac{\partial}{\partial\hat{r}}\left(\hat{r}\hat{\rho}_f\frac{\partial\hat{P}}{\partial\hat{r}}\right) = 0. \tag{3.4}$$

Turning to the expression for the thermal energy we make the further assignment that

$$T = T_0 + \Delta T\hat{T}.$$

[12] For the purposes of the dimensional analysis the relative permeability has been taken as a constant.

The expression for the temperature becomes

$$\rho c \frac{\partial \hat{T}}{\partial \hat{t}} + \rho_f c_f \hat{u} \frac{\partial \hat{T}}{\partial \hat{r}} - \frac{\beta \kappa}{x^2} \frac{1}{\hat{r}} \frac{\partial}{\partial \hat{r}} \left(\hat{r} \frac{\partial \hat{T}}{\partial \hat{r}} \right) - \frac{\sigma |E|^2}{\Delta T} \beta + \frac{LM}{\Delta T} \beta = 0.$$

The first two terms have coefficients of $\rho c \sim 10^6$ and $\rho_f c_f \sim 10^6$. Since the thermal conductivity $\kappa \sim 10$, the coefficient of the third term has a magnitude of $\beta\kappa/x^2 \sim 10^4 \ll 10^6$ indicating that diffusion of the temperature field can be ignored to first order. For the last two terms we use [8] $\Delta T \sim 10^2$, $\sigma|E|^2 \sim 10^3$, $L \sim 10^6$ and $M \sim 10^{-4}$ giving $\sigma|E|^2\beta/\Delta T \sim 10^6$ and $LM\beta/\Delta T \sim 10^5$ and to first order the temperature field satisfies

$$\rho c \frac{\partial \hat{T}}{\partial \hat{t}} + \rho_f c_f \hat{u} \frac{\partial \hat{T}}{\partial \hat{r}} - \frac{\sigma |E|^2}{\Delta T} \beta = 0. \tag{3.5}$$

We should mention that the phase-change term may become important when the mass rate M increases. In that case, the temperature and the flow fields will be coupled and numerical or asymptotic methods have to be used.

Dropping the hat notation and assuming ρ_f is spatially uniform allow one to express (3.4) and (3.5) as

$$\frac{1}{r} \frac{\partial}{\partial r} \left(r \frac{\partial P}{\partial r} \right) = 0, \tag{3.6}$$

$$\rho c \frac{\partial T}{\partial t} + \rho_f c_f u \frac{\partial T}{\partial r} = \frac{\sigma |E|^2}{\Delta T} \beta. \tag{3.7}$$

Finally, we use the motion of a long gas bubble inside a small channel to describe the relative vapour rising velocity in the vertical z direction due to buoyancy force. This problem was first investigated by Taylor [11] and studied subsequently by many researchers as a model to gain insights into slug multi-phase flows (in oil and gas recovery) [2, 7, 9, 10, 12].

For low viscosity and high surface tension systems such as the water-vapour two phase flows in a moderate-sized circular tube, Tung and Parlange [12] proposed that the terminal velocity v_b of the rising bubble is given by the phenomenological expression

$$v_b = \sqrt{0.272gd} - 0.472 \frac{\gamma}{\Delta \rho d}, \quad d \geq d_{\min} = 0.936 \left[\frac{\gamma^2}{g(\Delta\rho)^2} \right]^{1/3} \tag{3.8}$$

where d is the diameter of the tube, γ is the surface tension coefficient and $\Delta\rho = \rho_l - \rho_v$ is the density difference of the liquid and vapour. Note that for a sufficiently small tube, $d = d_{\min}$, this formula predicts the vapour slug velocity becomes zero.

On the other hand, experimental investigations in micro non-circular channels have shown that the elongational bubbles always rise even for a channel with effective diameter as small as 0.866$\times 10^{-3}$ meter [1]. More recently, numerical simulation of long gas bubbles rising through micro channels with triangular and rectangular cross section filled with stagnation liquid has been carried out [7]. The terminal velocity of the rising bubble as a function of the effective diameter is given in a non-dimensional form as

$$\mathrm{Ca} = c_1 \mathrm{Eo}^{d_1} + c_2 \mathrm{Eo}^{d_2} \tag{3.9}$$

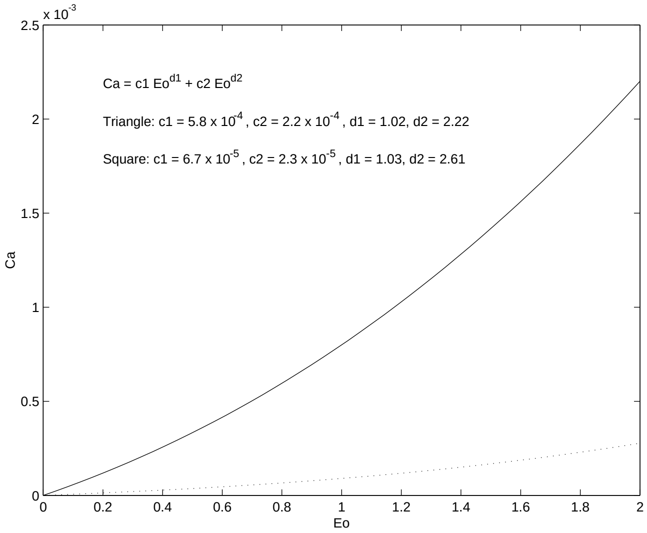


Figure 3.1: Illustrated is the capillary number as a function of the Eötvös number for channels with a triangular (solid) and square (dotted) cross section.

where the capillary number Ca and the Eötvös number Eo are defined as

$$\text{Ca} = \frac{\mu v_b}{\gamma}, \quad \text{Eo} = \frac{\Delta \rho g d^2}{\gamma}.$$

The parameters c_i and d_i are fit using numerical results. Figure 3.1 illustrates this relationship for channels with a triangular and rectangular cross section.

3.2.1 Pressure

Now let us consider the following equation for the pressure distribution in the system, assuming that the relative permeability is a constant one has equation (3.6)

$$\frac{\partial}{\partial r}\left(r\frac{\partial P}{\partial r}\right) = 0$$

with boundary conditions

$$P(r_w) = P_w, \quad P(r_e) = P_e,$$

where r_w is the radius of the domain (where the injection well is placed) and r_e is the radius of the extraction well, P_w is the pressure at the extraction well and P_e is the pressure at the electrode. The solution to this equation is

$$P(r) = (P_w - P_e)\frac{\ln r - \ln r_e}{\ln r_w - \ln r_e} + P_e. \tag{3.10}$$

3.2.2 Velocity

We have two equations for velocity distribution: lateral velocity and vertical velocity. Let us first consider equation for lateral velocity field:

$$u = -\frac{k}{\mu}\frac{dP}{dr}. \tag{3.11}$$

Using expression (3.10) we find

$$u = -\frac{k}{\mu}\frac{P_w - P_e}{\ln r_w - \ln r_e}\frac{1}{r}. \tag{3.12}$$

For the vertical component of the vapour velocity, we use the dimensional form of (3.9), which gives

$$v_b = \frac{\gamma}{\mu_l}\text{Ca} = \frac{\gamma}{\mu_l}\left(c_1\text{Eo}^{d_1} + c_2\text{Eo}^{d_2}\right). \tag{3.13}$$

From which we can obtain the (average) discharge velocity in the vertical direction

$$v_v = \tau\phi(1-\alpha)v_b = \tau F V_\infty \tag{3.14}$$

where $V_\infty = \Delta\rho g d^2/(3\mu_l)$ is a representative terminal velocity of the rising vapour bubbles, and τ is the tortuosity factor of the bubble in the porous media relative to the terminal velocity of a vapour bubble in straight vertical channel with diameter d. The factor

$$F = \phi(1-\alpha)\frac{\gamma}{\mu_l V_\infty}\left(c_1\text{Eo}^{d_1} + c_2\text{Eo}^{d_2}\right) \tag{3.15}$$

where ϕ is the porosity of the medium

3.2.3 Temperature

Assuming constant electrical current inside the electrode I_t, the magnitude of the electrical field in the porous medium can be written as $E = I_t/(2\pi r\sigma H)$ where H is the height of the electrode. The energy equation (3.7) can be written in the form

$$\frac{\partial T}{\partial t} + \frac{b}{r}\frac{\partial T}{\partial r} = \frac{a}{r^2} \tag{3.16}$$

with initial and boundary conditions

$$\begin{cases} T(r,0) = T_0(r), \\ T(r_e,t) = T_e \end{cases} \tag{3.17}$$

with $T_0(r) = T_e$ for $0 < r \leq r_e$ and where

$$a = \frac{10^3 I_t^2}{4\sigma\pi^2 H^2\rho c}, \quad b = -\frac{q}{2\pi H}\frac{\rho_f c_f}{\rho c}.$$

Here we have replaced u by the volume flow rate q using the relationship $q = -2\pi r H u$. Unlike pressure, the temperature of the system is time dependent. In order to solve (3.16)-(3.17) we use method of characteristics which gives us:

$$T(r,t) = \begin{cases} \frac{a}{b}\ln\left(\frac{r}{\sqrt{r^2-2bt}}\right) + T_0(\sqrt{r^2-2bt}), & r \geq \sqrt{r_e^2+2bt} \\ \frac{a}{b}\ln\left(\frac{r}{r_e}\right) + T_e, & r < \sqrt{r_e^2+2bt}. \end{cases} \tag{3.18}$$

3.3 Transport of Contaminants

Let C_v and C_l be the mass concentration of the contaminant in the vapour and liquid phase. The total amount of contaminant is conserved and satisfies

$$\frac{\partial}{\partial t}(C_v + C_l) + \nabla \cdot (\vec{u}_v C_v + \vec{u}_l C_l) = 0$$

with $\vec{u}_i = \langle u_i, v_i \rangle$ being the radial and vertical components of the velocity vector of either the vapour $(i = v)$ or liquid $(i = l)$ phase.

Fast reaction assumption yields

$$C_l = K C_v$$

where

$$K = P \times 10^{-A+B/(T+C)}$$

with [8] $A = 7.098$, $B = 1238.71$ and $C = 217$. Here T and P are temperature and pressure, respectively. Eliminating C_l gives the expression

$$\frac{\partial}{\partial t}(1+K)\,C_v + C_v \nabla \cdot (\vec{u}_v + K\vec{u}_l) + (\vec{u}_v + K\vec{u}_l)\,\nabla C_v = 0. \tag{3.19}$$

Therefore, assuming that K is independent of time, we have

$$\frac{\partial}{\partial t}C_v + \frac{C_v}{1+K}\nabla \cdot (\vec{u}_v + K\vec{u}_l) + \left(\frac{\vec{u}_v + K\vec{u}_l}{1+K}\right)\nabla C_v = 0. \tag{3.20}$$

Solving by the method of characteristics, we have

$$\frac{dC_v}{dt} + \frac{C_v}{1+K}\nabla \cdot (\vec{u}_v + K\vec{u}_l) = 0 \tag{3.21}$$

where the characteristic base curves are determined by the two ODEs

$$\begin{aligned} \frac{dr}{dt} &= \frac{u_v + K u_l}{1+K}, \\ \frac{dz}{dt} &= \frac{v_v + K v_l}{1+K}. \end{aligned}$$

From these we have

$$\frac{dr}{dz} = \frac{u_v + Ku_l}{v_v + Kv_l}. \tag{3.22}$$

From the flow and temperature model we have

$$u_l = u_v = -\frac{q}{2\pi rH}, \quad v_l = 0, \quad v_v = \tau F V_\infty$$

where q is the volume flow rate at the extraction well, F is given by (3.15) and τ is a free parameter.

Assume that a parcel of contaminant at the base of the injection electrode ($z = 0$, $r = r_w$) moves towards the extraction electrode located at $r = r_e < r_w$. According to equation (3.22) the vertical displacement of this parcel is given by the monotonically increasing function

$$z = \frac{2\pi H v_v}{q} \int_r^{r_w} \frac{\xi}{1+K} d\xi.$$

If we let h be the height above the contaminant region which has been heated up by the electrodes then the parcel will be successfully extracted provided it reaches $r = r_e$ before $z = h$. Let $z_e = z(r_e)$ denote the height of this characteristic when it reaches the extraction well $r = r_e$. By assuming that the factor $1 + K$ does not vary significantly over the interval $r_e \leq r \leq r_w$ the condition that $z_e \leq h$ can be converted into a lower bound on the volumetric flux of

$$q \geq \frac{\pi\tau F H V_\infty (r_w^2 - r_e^2)}{h(1+K)}.$$

With $H = 5$ m, $h = 1$ m, $r_w = 5$ m, $r_e = 0$ m, $\Delta\rho = 10^3$ kg/m^3, $\mu_l = 10^{-3}$ Pa s, and $\tau = 0.01$, for a mean throat size of $d = 10^{-3}$ m, we obtain $F = 1.7 \times 10^{-4}$ for a medium with square channels. Based on these values, we can compute the value of $q = 6.8 \times 10^{-4}$ m^3/s. The calculation was done based on the parameter values listed in Table 3.1. We note that there is no physical basis for choosing this value. However, viewing the possibility that bubbles may get trapped in a particular porous medium, it is not unreasonable to expect that it will take a much longer time for the bubbles to travel vertically.

3.4 Conclusions and Recommendations

In this report we have proposed a simple model for estimating the transport of contaminants using thermal remediation. Based on the model, the minimum extraction rate of fluid is calculated and its value is within the practical range. However, many questions remain unanswered. For example, we have not addressed the effect of temperature variation in the vertical direction and near the edge of the heated zone. We have not attempted to examine the effect of possible condensation near the cold/low pressure region. Finally, we have not considered the possibility that bubbles may be trapped in the isolated pore space and the effects of heating and the accumulation of vapour bubbles on the soil.

We should also mention that we have not tried to identify transitions between liquid-only, vapour-only and two-phase regions and the transportation of contaminants in the liquid and vapour only regions. However, this may not be as critical as other issues since it is relatively simpler to determine the velocity of the liquid or gas in the one-phase region.

Data	Symbol	Value
Operating Properties		
Maximum Temperature	$T_{\max}$	100^oC
Initial Temperature	$T_{\min}$	20^oC
Initial Pressure	P_0	101.325 kPa
Target Thickness	H	5 m
Electrode Length	L_e	5 m
Electrode Spacing	Z_e	10 m
Extraction Well Spacing	Z_x	10 m
Physical Properties		
Initial Permeability[13]	k	10.0 mD
Viscosity[14]	μ	1.0 cP
Surface Tension of Water[15]	γ	0.0717 N m^{-1}
Producing Pressure Drop	ΔP	500 kPa
Total Heat Capacity	ρc	2.8×10^6 J m^{-3}K^{-1}

Table 3.1: Input Data for the sample calculations.

Evidently further improvements are needed before the model can be used for prediction and to answer the other issues raised. In particular, a proper flow and temperature model must be derived and reference [5] should provide a good starting point where thermal two-phase flow in porous media is discussed. Secondly, the effects of the phase change (vapourization and condensation), capillary pressure and the existence of vapour only regions, which have been neglected, may be important in certain domains, especially near the boundary of the heated zone where the change of soil temperature is more significant. Models analyzing condensation and vapourization in porous media have been studied for other applications which may be helpful, see for example [3] and references therein. Finally, the effect of different types of porous media and the impact of heating and vapour bubbles may be important as well. Models of gas penetrating elastic media have been studied [6], which may be useful if fracturing of the media needs to be considered.

[13] 1 Darcy = 9.87×10^{-12} m^2.

[14] 1 centipoise = 1×10^{-3} kg m^{-1}s^{-1} $= 1 \times 10^{-3}$ Pa s.

[15] at 25^oC.

Bibliography

[1] Q.C. Bi & T.S. Zhao. Taylor bubbles in miniaturized circular and noncircular channels. *Int. J. Multiphase Flow*, 27, pp. 561-570 (2001).

[2] F.P. Bretherton. The motion of long bubbles in tubes. *J. Fluid Mech.*, 10, pp. 166-188 (1961).

[3] L. Bridge, R. Bradean, M.J. Ward & B.R. Wetton. The analysis of a two-phase zone with condensation in a porous medium. *J. Eng. Math.*, 45, pp. 247-268 (2003).

[4] H.I. Ene. Thermal flow. In *Homegenization and Porous Media*, U. Hornung Ed., Chapter 7, pp. 147-162, Springer-Verlag, NY, 1997.

[5] H.I. Ene & D. Polisevski. *Thermal Flow in Porous Media*, D. Reidel, Dordrecht, 1987.

[6] O.E. Jensen, M.K. Horsburg, D. Halpern & D.P. Graver III. The steady propagation of a bubble in a flexible-walled channel: asymptotic and computational models. *Phys. Fluids*, 14, pp. 443-457 (2002).

[7] Q. Liao & T.S. Zhao. Modeling of Taylor bubble rising in a vertical mini moncircular channel filled with a statgnnat liquid. *Int. J. Multiphase Flow*, 29, pp. 411-434 (2003).

[8] B. McGee, private communication.

[9] D.A. Reinelt & P.G. Saffman. The penetration of a finger into a viscous fluid in a channel and tube. *SIAM J. Sci. Stat. Comput.*, 6, pp. 542-561 (1985).

[10] P.G. Saffman & G.I. Taylor. The penetration of a fluid into a porous medium or Hele-Shaw cell containing a more viscous liquid. *Proc. Roy. Soc. A.*, 245, pp. 312-329 (1958).

[11] G.I. Taylor. Deposition of a viscous fluid on the wall of a tube. *J. Fluid Mech.*, 10, pp. 161-165 (1961).

[12] K.W. Tung & J.Y. Parlange. Note on the motion of long bubbles in closed tube: influence of surface tension. *Acta Mech.*, 24, pp. 313-317 (1976).

Chapter 4

Tracking and Identifying of Multiple Targets

John Hoffman[1], Christian Ketelsen[2], Michael Kouritzin[3], Alfonso Limon[4], Yuriy Mileyko[5],
Kerianne Yewchuk[3]

Report prepared by Kerianne Yewchuk (kyewchuk@ualberta.ca), Christian Ketelsen (cketelsen@wsu.edu),
Alfonso Limon (alfonso.limon@cgu.edu) and Yuriy Mileyko (ym4@njit.edu)

4.1 Introduction

There are many statistical methods of tracking single and multiple targets; this manuscript will focus on the state estimation problem. Ideally, a generalization of the recursive Bayes non-linear filter would track and resolve the state(s) of single or multiple targets, but that is currently computationally intractable. The Probability Hypothesis Density (PHD) makes the tracking problem computationally feasible by propagating only the first-order multi-target statistical moments by using a particle filter implementation for the PHD. The problem then becomes one of estimating the targets' state based on the output of the PHD when using a particle filter implementation.

This paper describes one heuristic method for obtaining a state estimator from the PHD. The approach used in this paper, based on the Expectation-Maximization (EM) algorithm, views the PHD distribution as a mixture distribution, and the particles as an i.i.d. sampling from the mixture distribution. Using this, a maximum likelihood estimator for the parameters of the distribution can be generated. The EM seems to work fairly well, particularly when targets are well spaced.

4.2 Problem Description

The problem is one of tracking and identifying a finite set of multiple targets by means of data collected from a set of multiple sensors. The motion of each target is modelled as a discrete time, continuous space Markov process; it is also assumed that targets move independently. The exact number of targets

[1] Lockheed Martin
[2] Washington State University
[3] University of Alberta
[4] Claremont Graduate University
[5] New Jersey Institute of Technology

is unknown and may change with time depending on the corresponding birth/death model. At each time step k, the observations $Z_1^k, \dots, Z_M^k$ are gathered from the sensor suite. These observations are affected by clutter; the amount of clutter is modelled using a Poisson distribution.

The PHD is a computationally efficient means of solving the tracking problem; however, it does not provide an estimate of the targets' states. Our approach for solving this part of the problem is described in section 4.4.

4.3 Target Tracking and the PHD

Consider a single-sensor, single-target problem. Let x_k be the target state variable at time step k, and z_k the observation at time step k. Assume that x_k is a Markov process with initial distribution $p_0(x_0)$ and transition equation $p_{k+1|k}(x_{k+1}|x_k)$. Also assume that the observations z_k are conditionally independent given the process x_k and of marginal distribution $p(z_k|x_k)$. In this case, the recursive equations for the posterior distribution can be written as

$$\begin{aligned} p_{k+1|k}(x_{k+1}|Z^k) &= \int p_{k+1|k}(x_{k+1}|x_k) p_{k|k}(x_k|Z^k) dx_k \\ p_{k+1|k+1}(x_{k+1}|Z^{k+1}) &= \frac{p(z_{k+1}|x_{k+1}) p_{k+1|k}(x_{k+1}|Z^k)}{\int p(z_{k+1}|x_{k+1}) p_{k+1|k}(x_{k+1}|Z^k) dx_{k+1}}, \end{aligned}$$

where[6] $Z^k = \{z_1, \dots, z_k\}$. The target state estimators can be given by

$$\hat{x}_{k+1|k+1}^{MAP} = \arg\sup_x p_{k+1|k+1}(x|Z^{k+1}), \hat{x}_{k+1|k+1}^{EAP} = \int x p_{k+1|k+1}(x|Z^{k+1}) dx,$$

where $\hat{x}_{k+1|k+1}^{MAP}$ and $\hat{x}_{k+1|k+1}^{EAP}$ are the Bayes-optimal maximum *a posteriori* and expected *a posteriori* estimators, respectively.

The generalization of the above equations for the multiple-sensor, multiple-target system is not quite obvious. The target state is not just one random vector anymore. The number of targets, as well as their positions, velocities, identities, etc., are all unknown and should be treated as random variables. To deal with this, the tool of finite-set statistics (FISST) can be used. It is based on the following ideas [1]:

- Introduce a notion of a single "global sensor" that encompasses the whole sensor suite.
- Introduce a notion of a "global target" with multi-target state $X = \{x_1, \dots, x_n\}$.
- Regard the set of observations, $Z = \{z_1, \dots, z_m\}$, as a "global measurement" of the "global target."
- Use the *multi-sensor/multi-target measurement model*, which is a randomly varying set $\Sigma = T(X) \cup C(X)$, to model multi-target multi-sensor data. Here $T(X)$ denotes targets' data and $C(X)$ denotes clutter.

[6] Conditioning on Z^k is by definition the same as conditioning on $\mathcal{Y}_k = \sigma\{Z^k\}$.

- Use the *multi-target motion model*, which is a random set $\Gamma_{k+1} = \Phi_k(X_x, V_k)$, to model the motion of multi-target systems.

These ideas make it possible to reformulate multi-sensor, multi-target problems as single-sensor, single-target problems. The process of reformulation relies heavily on the notion of the belief-mass function, which is a generalization of the probability-mass function (see [2]). The FISST multi-sensor multi-target differential and integral calculus introduces the *set integral* and the *set derivative*, which are used to state the problem in rigorous mathematical terms. Omitting all the details (in [1] and [2]), the resulting recursive equations are

$$\begin{aligned} p_{k+1|k}(X_{k+1}|Z^{(k)}) &= \int p_{k+1|k}(X_{k+1}|X_k)p_{k|k}(X_k|Z^{(k)})\delta X_k \\ p_{k+1|k+1}(X_{k+1}|Z^{(k+1)}) &= \frac{p(Z_{k+1}|X_{k+1})p_{k+1|k}(X_{k+1}|Z^{(k)})}{\int p(Z_{k+1}|Y)p_{k+1|k}(Y|Z^{(k)})\delta Y}, \end{aligned}$$

where X_k is a multi-target state, $f_{k|k}(X_k|Z^{(k)})$ is a multi-target posterior density, and $f(Z|Y)$ is a multi-sensor, multi-target likelihood function.

Note that the integrals in the above formulae are set integrals, the computation of which is very expensive. Actually, even in single-target problems, the recursive equations are too complicated to calculate explicitly. Therefore, some approximation should be found. It turns out that, when signal-to-noise ratio (SNR) is high enough, the first-order moment, $\hat{f}_{k|k} = \int x f_{k|k}(z|Z^k)dx$, is a good approximation for single-target problems. Unfortunately, it is not possible to use this straightforward generalization for multiple target problems, because the integral $\int X f_{k|k}(X|Z^{(k)})\delta X$ cannot be defined. The problem is resolved by using some function h that maps the state-set X into a vector space. Then, the first-order moment is computed indirectly as

$$E[h(\Gamma_k)] = \int h(X)f_{k|k}(X|Z^{(k)})\delta X.$$

One of the possible choices for the function h is $h(\Gamma_k) = \delta_{\Gamma_k}$, where $\delta_{\Gamma_k}(x) = \sum_{w \in \Gamma_k} \delta_w(x)$. This makes the first-order moment, denoted by $D_{k|k}(x|Z^{(k)})$, the *probability hypothesis density* (PHD). It has the property that $\int_S D_{k|k}(x|Z^{(k)})dx$ is the expected number of targets contained in the region S (for further information about the PHD see [2]). If the SNR and the signal to clutter ratio are high enough, and the multi-target system has a zero covariance, then the PHD is a good approximation for the unnormalized multi-target posterior density, and an explicit recursive equation can be derived for $D_{k|k}(x|Z^{(k)})$ (see [1]). In general, the time update equation depends on the motion model, which normally includes birth and death of targets.

$$D_{k+1|k}(y|Z^{(k)}) = \int (d_{k+1}(x)f_{k+1|k}(y|x) + b_{k+1|k}(y|x))D_{k|k}(x|Z^{(k)})dx$$

where $d_{k+1}(x)$ is the probability that a target with a state x at time-step k will disappear at time-step $k+1$, and $b_{k+1|k}(y|x)$ is the PHD of the multi-target density $\bar{b}_{k+1|k}(X|x)$ that describes the likelihood that a target with a state x at time-step k will generate a set X of new targets at time step $k+1$.

Assuming that there are no births and deaths, the above equation reduces to

$$D_{k+1|k}(y|Z^{(k)}) = \int f_{k+1|k}(y|x)D_{k|k}(x|Z^{(k)})dx. \tag{4.1}$$

The approximate formula for the observation update is

$$
\begin{aligned}
D_{k+1|k+1}(x|Z^{(k+1)}) \;\simeq\; & \sum_{z\in Z_{k+1}} \frac{p_D f(z|x) D_{k+1|k}(x|Z^{(k)})}{\lambda_{k+1}c_{k+1} + p_D \int f(z|x) D_{k+1|k}(x|Z^{(k)})dx} + \\
& +(1-p_D)D_{k+1|k}(x|Z^{(k)}). \qquad (4.2)
\end{aligned}
$$

Thus, an approximation of the distribution of the whole set of targets at each time step can be computed quite efficiently. The question then arises as to how the state of each separate target can be estimated. A new approach to this problem was proposed and is described in the next section along with other existing approaches.

4.4 State Estimation and the EM Algorithm

Using the results of the previous section, it is assumed that at time step k the function $D_{k|k}(x|Z^{(k)})$ is given. Integrating it over the whole space, the expected number of targets can be computed. The question then becomes: how can this information be used to find the state of each target?

One of the existing approaches is based on peak detection. The idea is very simple. When the expected number of targets is M and the function $D_{k|k}(x|Z^{(k)})$ has M local maxima, it is reasonable to assume that targets' states are located at the points where those maxima are achieved (see Figure 4.1). There are some difficulties when targets are located very close to each other; for example, the number of local maxima may be smaller than the expected number of targets (see Figure 4.2).

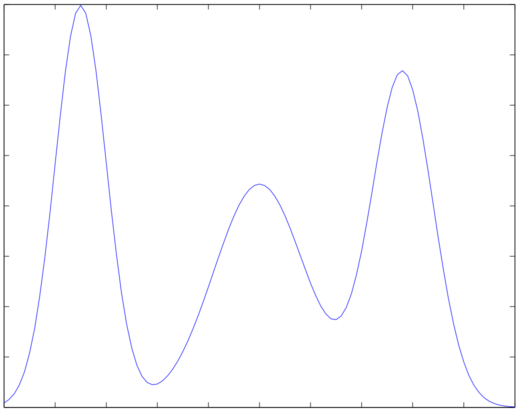


Figure 4.1: Expected number of targets is 3, so the targets are assumed to be located under each peak.

Another approach is based on clustering algorithms, which are ubiquitous in the pattern recognition community. Since the PHD, $D_{k|k}(x|Z^{(k)})$, is represented as a set of particles (see section 4.5.2), it is reasonable to expect that particles are clustered around each target. Applying a clustering algorithm, the loci for the targets can be computed. One of the most popular clustering algorithms is Nearest Neighbour Clustering. It usually gives good results, but the time needed to construct the loci is quadratic in the number of particles. Also, non-intuitive clusters are constructed in some cases.

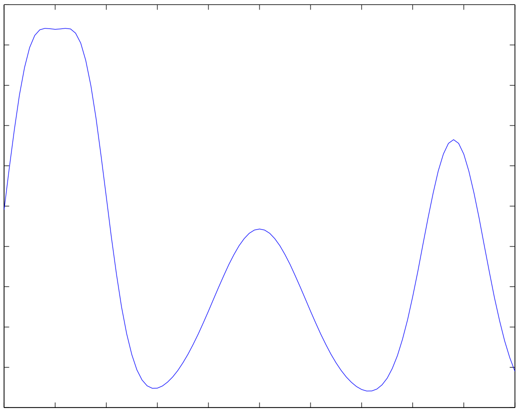


Figure 4.2: Expected number of targets is 4, so there is no natural way to determine targets' states using just the location of the relative maxima.

The new approach is based on the following idea. Suppose that the PHD, $D_{k|k}(x|Z^{(k)})$, is a mixture of Gaussian densities with unknown means and covariance matrices, i.e.,

$$D_{k|k}(x|Z^{(k)}) = \sum_{i=1}^{M} \alpha_i N_i(x|\mu_i, \Sigma_i) = p(x|\Theta),$$

where M is the expected number of targets, $\Theta = \{\alpha_i, \mu_i, \Sigma_i\}_{i=1}^{M}$, and α_i satisfy $\sum_{i=1}^{M} \alpha_i = 1$. The likelihood of these parameters given data $\mathcal{X}$ is

$$\mathcal{L}(\Theta|\mathcal{X}) = \prod_{i=1}^{N} p(x_i|\Theta).$$

It is reasonable to expect that maximization of the likelihood, using the set of particles representing the PHD as the data $\mathcal{X}$, should give the values of the parameters, $\Theta^* = \arg\max_{\Theta} \mathcal{L}(\Theta|\mathcal{X})$, which will provide a good approximation for $D_{k|k}(x|Z^{(k)})$. Then the means, μ_i^*, should be good estimators for the states of targets.

The difficult part of this process is the maximization of the likelihood. The usual algorithms for non-linear optimization may not be efficient in this case, since the number of parameters is quite large. Therefore, the need for more robust techniques arises. One such technique is the Expectation-Maximization (EM) algorithm.

4.4.1 Expectation-Maximization Algorithm

Suppose the data $\mathcal{X}$ for maximum-likelihood estimation is incomplete or has missing values[7]. Assume that there exists a complete data set $\mathcal{Z} = (\mathcal{X}, \mathcal{Y})$, where $\mathcal{Y}$ is missing information. Let

$$p(z|\Theta) = p(x, y|\Theta) = p(y|x, \Theta)p(x|\Theta)$$

[7] Sometimes it is convenient to assume that there are some missing (or hidden) parameters to simplify the expression for the likelihood function.

be a joint density function. Then a new likelihood function is $\mathcal{L}(\Theta|\mathcal{Z}) = \mathcal{L}(\Theta|\mathcal{X}, \mathcal{Y}) = p(\mathcal{X}, \mathcal{Y}|\Theta)$. Note that it is a random variable, since $\mathcal{Y}$ is supposed to be unknown and random. So, it can be assumed that $\mathcal{L}(\Theta|\mathcal{X}, \mathcal{Y}) = h_{\mathcal{X},\Theta}(\mathcal{Y})$ for some function $h_{\mathcal{X},\Theta}$, where $\mathcal{X}$ and Θ are constant and $\mathcal{Y}$ is a random variable.

The EM algorithm consists of two steps.

1. Compute the expected value of the log-likelihood, i.e.

   $$Q(\Theta, \Theta^{(i-1)}) = E[\log p(\mathcal{X}, \mathcal{Y}|\Theta)|\mathcal{X}, \Theta^{(i-1)}],$$

   where $\Theta^{(i-1)}$ are current parameter estimates, $\mathcal{X}$ is the observed data, and $\mathcal{Y}$ is the random variable with respect to which the expected value is computed. Note that Θ is a normal variable, so $Q(\Theta, \Theta^{(i-1)})$ is a normal function of Θ.

2. Maximize the expectation computed in the first step and find

   $$\Theta^{(i)} = \arg\max_{\Theta} Q(\Theta, \Theta^{(i-1)}).$$

These two steps are repeated until some stopping criterion is satisfied. It can be demonstrated that each additional iteration increases the log-likelihood; therefore, the algorithm is guaranteed to converge to a local maximum of the log-likelihood function.

4.4.2 Mixture-Density Parameter Estimation

To return to the state estimation problem, recall that the PHD is supposed to be a mixture of Gaussian distributions with unknown parameters. The log-likelihood function is in this case

$$\log(\mathcal{L}(\Theta|\mathcal{X})) = \sum_{i=1}^{N} \log(p(x_i|\Theta)) = \sum_{i=1}^{N} \log\left(\sum_{j=1}^{M} \alpha_j N(x_i|\mu_j, \Sigma_j)\right).$$

Suppose that the data $\mathcal{X}$ is incomplete and that there exists an unobserved datum $\mathcal{Y} = \{y_i\}_{i=1}^{N}$. Let $y_i \in \{1, \ldots, M\}$ and $y_i = k$ if the i^{th} sample, x_i, was generated by the k^{th} Gaussian component, i.e. the component with the mean μ_k and the covariance matrix Σ_k. Then the expression for the log-likelihood can be rewritten as

$$\log(\mathcal{L}(\Theta|\mathcal{X}, \mathcal{Y})) = \log(P(\mathcal{X}, \mathcal{Y}|\Theta)) = \sum_{i=1}^{N} \log(P(x_i|y_i)P(y)) = \sum_{i=1}^{N} \log(\alpha_{y_i} N(x_i|\mu_{y_i}, \Sigma_{y_i})).$$

Obviously, the problem is that values of $\mathcal{Y}$ are unknown. Assuming that y_i are random variables, it is possible to apply the EM algorithm. Let $\Theta^g = (\alpha_1^g, \ldots, \alpha_M^g, \mu_1^g, \ldots, \mu_M^g, \Sigma_1^g, \ldots, \Sigma_M^g)$ be some guess.

Omitting a few details (which can be found in [3]) the expressions for Θ^{g+1} are

$$\begin{aligned}
\alpha_l^{g+1} &= \frac{1}{N}\sum_{i=1}^{N} p(l|x_i,\Theta^g) \\
\mu_l^{g+1} &= \frac{\sum_{i=1}^{N} x_i p(l|x_i,\Theta^g)}{\sum_{i=1}^{N} p(l|x_i,\Theta^g)} \\
\Sigma_l^{g+1} &= \frac{\sum_{i=1}^{N} p(l|x_i,\Theta^g)(x_i-\mu_l^{g+1})(x_i-\mu_l^{g+1})^T}{\sum_{i=1}^{N} p(l|x_i,\Theta^g)},
\end{aligned}$$

where

$$p(l|x_i,\Theta^g) = \frac{\alpha_l^g N(x_i|\mu_l^g,\Sigma_l^g)}{\sum_{k=1}^{M}\alpha_k^g N(x_i|\mu_k^g,\Sigma_k^g)}.$$

Only some distributions allow derivation of analytic expressions of Θ^{g+1}, and the Gaussian distribution is one of these. In general, some numerical procedure should be used to maximize the expectation $Q(\Theta,\Theta^g)$. Using the above formulae and some reasonable stopping criterion, the estimates for the targets' states, μ_l^*, $l=1,\ldots,M$, can be found efficiently at each time step.

4.5 Numerical Simulation

To test the state estimation method based on the Gaussian mixture assumption and the EM algorithm, several numerical tests were done. The model used for the tests was a racing car model, and the PHD method was implemented using a simple particle filter.

4.5.1 Model

Suppose cars are racing on a circle of length L with velocities in the range $[-b,-a]\cup[a,b]$, where the sign defines a direction of motion. The initial positions and velocities of the cars are distributed uniformly on intervals $[0,L)$ and $[-b,-a]\cup[a,b]$, respectively. The motion of the cars is described by

$$\begin{aligned}
x_{k+1} &= x_k + v_k, \\
v_{k+1} &= (v_k + N(0,\sigma_v^2))(2B(p)-1),
\end{aligned}$$

where x_k is a position of a car at time step k, v_k is the velocity of a car at time step k, $N(0,\sigma_v^2)$ denotes Gaussian noise with a variance σ_v^2, and $B(p)$ is a binomially distributed random variable (at each time step, it is 1 with probability p and 0 with probability $1-p$). Since the track is a circle, when a car crosses one of the boundaries (i.e., its position becomes greater than L or less than 0) it continues its motion from another boundary. Also note that this motion model is highly non-linear, since velocities randomly change not only in magnitudes, but also in directions.

The observation part of the model is defined as follows: The probability of observing a target (car) is P_D, the amount of clutter has Poisson distribution with a mean λ, and the clutter itself is distributed uniformly on the interval $[0,L)$. The observed data is

$$z_k = x_k + N(0,\sigma_{ob}^2),$$

where $N(0, \sigma_{ob}^2)$ is Gaussian noise with a variance σ_{ob}^2. To simplify computations, there is also a no births, no deaths assumption.

The computations were done using the following values for the parameters. The length of the track, $L = 100$; velocities are confined to the range $[-3, -2] \cup [2, 3]$, so $a = 2, b = 3$; maximum number of targets equals 6; variance of Gaussian noise for velocities, $\sigma_v^2 = 0.1$; probability of retaining the same direction, $p = 0.98$; probability of detecting a target, $P_D = 0.9$; variance of Gaussian noise for observations, $\sigma_{ob}^2 = 3$; mean of Poisson distribution for the amount of clutter, $\lambda = 10$.

4.5.2 Interactive Particle Filter

Note that equations (4.1) and (4.2) contain an integral, whose computation requires a lot of effort. To deal with this problem, a sequential Monte Carlo method can be used. One of these methods leads to the Interactive Particle Filter. At each time step k, the PHD is represented as

$$D_{k|k}(x|Z^{(k)}) = \frac{M_{k|k}}{N} \sum_{i=1}^{N} \delta_{\xi_{k|k}^i}(x),$$

where $\xi_{k|k}^i$ are simulated random samples, also named particles. Initially, $\xi_{0|0}^i$ are i.i.d random samples uniformly distributed in $[0, L) \times ([-b, -a] \cup [a, b])$. The number of particles, N, has to be sufficiently large, so for the presented model $N = 5000$ was used. The initial number of targets can be regarded as a random variable T such that $k = 1, \ldots, M$, where M is the maximum number of targets. Thus, the initial mass of the system can be computed as the expected value of T, i.e. $M_{0|0} = E[T]$. For the presented model $M_{0|0} = 3.5$.

The motion update is done by setting $M_{k+1|k} = M_{k|k}$ and moving each $\xi_{k|k}^i$ according to the motion of the targets, i.e. $\xi_{k+1|k}^i = \xi_{k|k}^i + v_k$. The observation update is more complicated. First, the following integral should be computed

$$I(z) = \int f(z|x) D_{k+1|k}(x|Z^{(k)}) dx = \frac{M_{k+1|k}}{N} \sum_{i=1}^{N} f(z|\xi_{k+1|k}^i).$$

Then, the total mass is

$$M_{k+1|k+1} = \int D_{k+1|k+1}(x|Z^{(k+1)}) dx = \sum_{z \in Z_{k+1}} \frac{P_D I(z)}{\lambda_{k+1} c_{k+1} + P_D I(z)} + (1 - P_D) M_{k+1|k}.$$

The next step is computing weights

$$\omega_{k+1|k+1}^i = \sum_{z \in Z_{k+1}} \frac{P_D f(z|\xi_{k+1|k}^i)}{\lambda_{k+1} c_{k+1} + P_D I(z)} + (1 - P_D) M_{k+1|k}.$$

These weights are used to resample the particles using the following rule

$$P(\xi_{k+1|k+1}^i = \xi_{k+1|k}^j) = \frac{\omega_{k+1|k+1}^j}{\sum_{l=1}^{N} \omega_{k+1|k+1}^l}.$$

Therefore, particles with smaller weights should eventually disappear. The particle filter as well as the EM algorithm was implemented using the Matlab software package. The code for the main routines is given in the Appendix, and results of the computations are presented below.

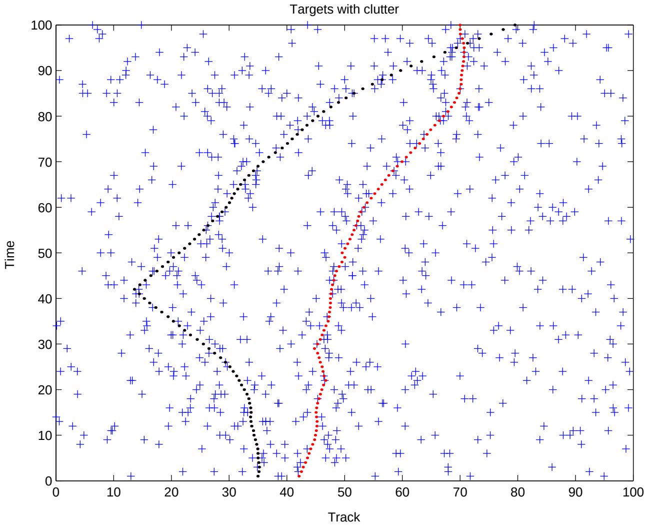


Figure 4.3: Motion of two targets.

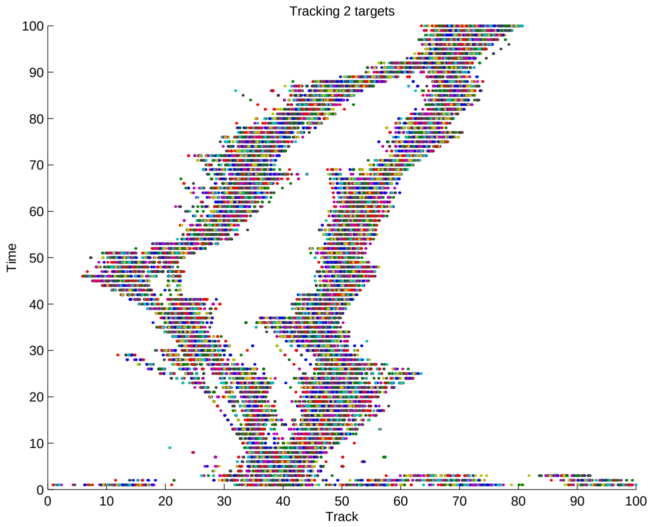


Figure 4.4: Tracking of two targets.

4.5.3 Results

First, the work of the Interactive Particle Filter is presented. In Figure 4.3, observations are shown; trajectories of targets are indicated by dots, while clutter is marked by crosses. The output of the particle filter is illustrated in Figure 4.4. Note how particles eventually cluster around the targets. Taking into account a relatively small number of particles, the PHD gives a good result. Another proof of the efficiency of the PHD is the graph of the total mass of the system, shown in Figure 4.5. Notice that the estimation of the number of targets is quite accurate, since, although the graph oscillates wildly, the average is always close to 2, which is the actual number of targets.

The EM algorithm turns out to be very sensitive to the output of the PHD. In general, it works well when trajectories of targets do not intersect and do not cross the boundaries. This case is shown in Figure 4.6. The difficulties arise when the trajectories intersect or cross the boundaries, the reason being a 'bad' output of the PHD.

The first case is illustrated in Figure 4.7. Note how the inaccurate estimation occurs at the point of intersection. The output of the particle filter for this case is shown in Figure 4.8, and in Figure 4.9, the approximation of the PHD as an unnormalized density function is presented. While the former figure shows no possible obstruction to state estimation, the later figure clarifies the poor performance of the EM algorithm. As the time draws close to the time of intersection, the density function has only one local maximum, a situation which makes identification of two targets very difficult for the EM algorithm.

Figures 4.10, 4.11, and 4.12 illustrate the targets' states estimation, the output of the particle filter, and the unnormalized density function, respectively, for trajectories which cross the boundaries. Notice that the density function has three local maxima that cause the EM algorithm to fail. In all cases, when the particles are not clustered well around the targets, the estimation is not accurate.

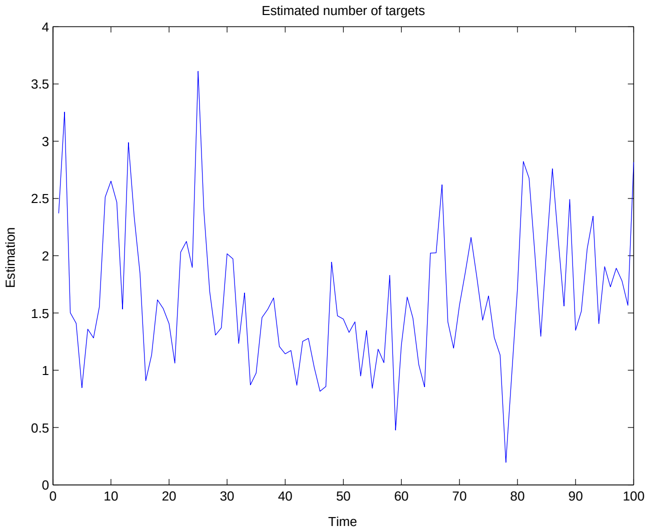


Figure 4.5: Total mass of the system with two targets.

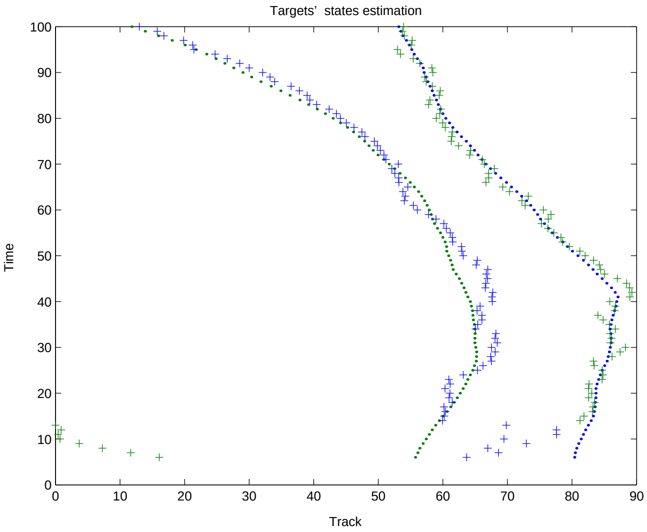


Figure 4.6: State estimation when targets' trajectories do not intersect and do not cross the boundaries. 'Dots' denote exact positions, 'crosses' denote estimations.

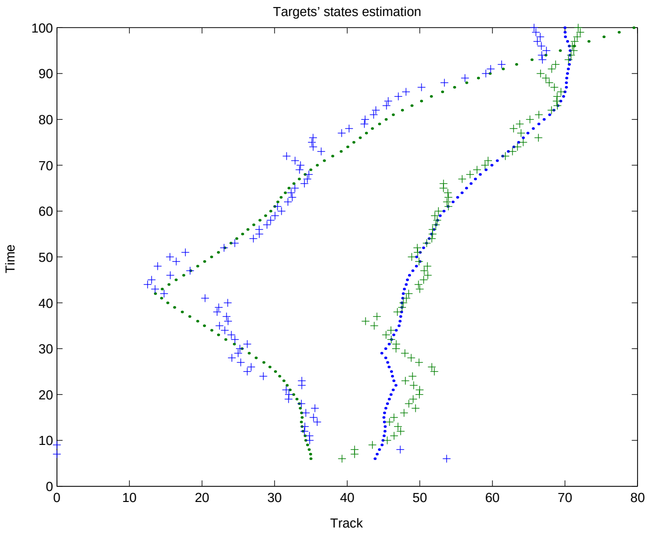


Figure 4.7: State estimation when targets' trajectories intersect. 'Dots' denote exact positions, 'crosses' denote estimations.

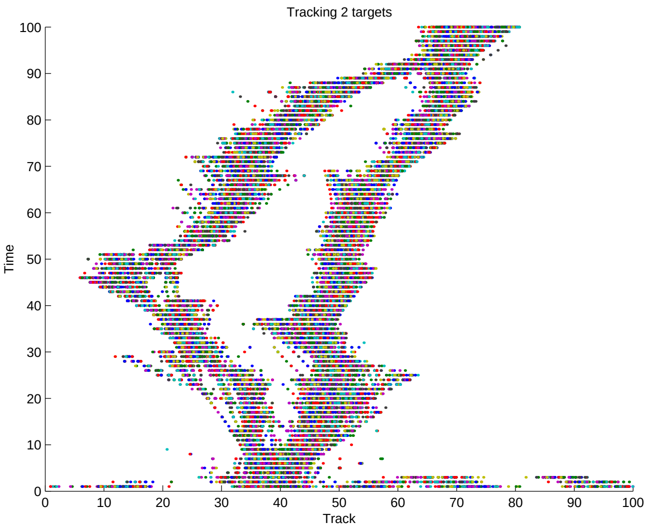


Figure 4.8: Tracking of two targets when their trajectories intersect.

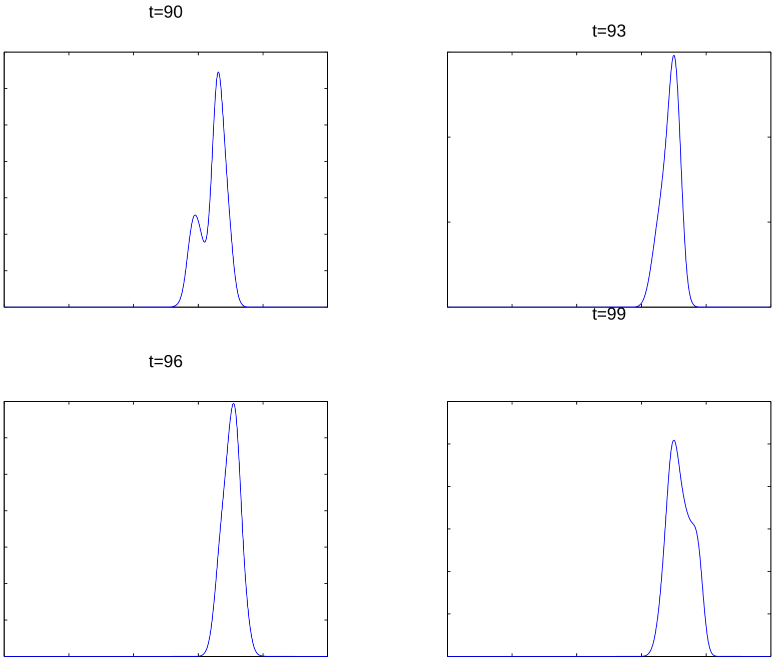
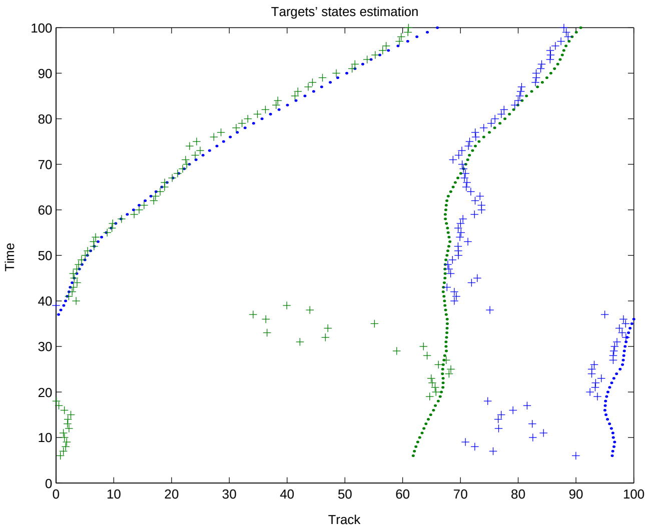


Figure 4.9: Approximation of the PHD as an unnormalized density function for times close to the intersection time.



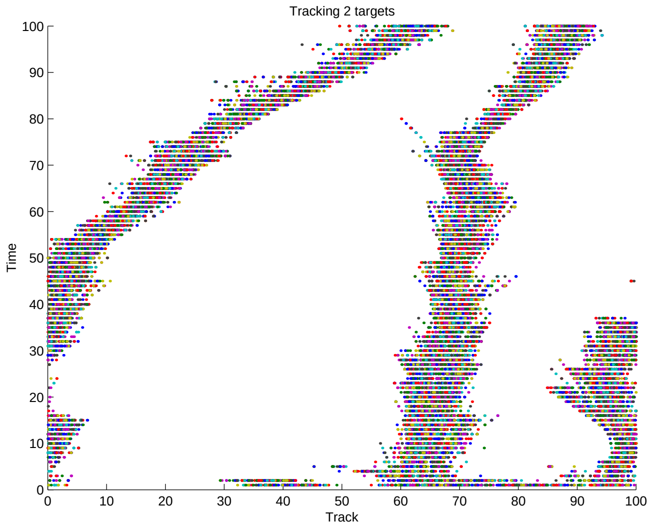


Figure 4.10: State estimation when targets' trajectories cross the boundaries. 'Dots' denote exact positions, 'crosses' denote estimations.

Figure 4.11: Tracking of two targets when one crosses the boundary.

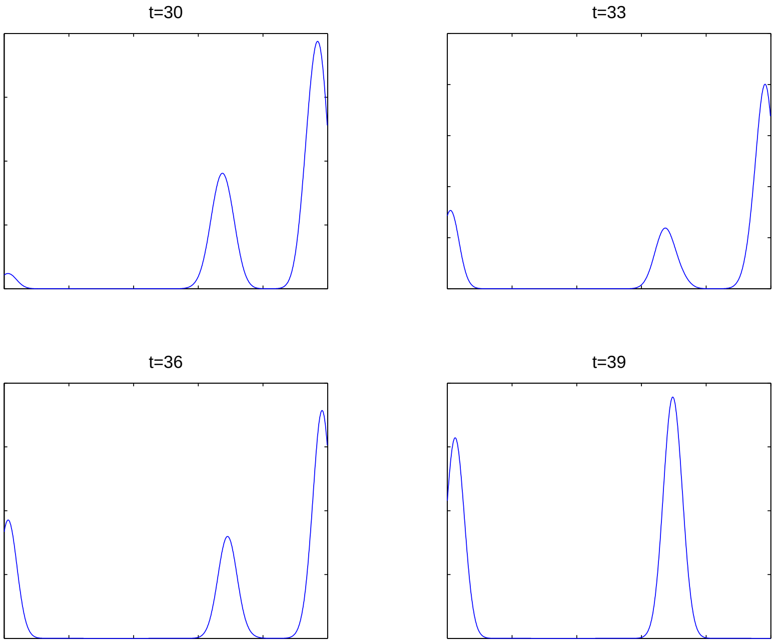


Figure 4.12: Approximation of the PHD as an unnormalized density function for times close to the crossing time.

4.6 Conclusion

In this paper, a new method for targets' states estimation was proposed as a part of a solution to the problem of tracking and identifying multiple targets. The recently proposed Probability Hypothesis Density (PHD) was used to track multiple targets when observations are gathered from multiple sensors. This new method for estimating targets' states is based on the assumption that the PHD can be represented as a mixture of Gaussian densities with unknown means and covariance matrices. The number of Gaussian densities in this mixture is the expected number of targets. Using this representation, the likelihood function can be constructed, and maximization of the likelihood gives the values of the means (for the Gaussian densities) that can serve as estimates for the targets' states. To maximize the likelihood, the Expectation-Maximization (EM) algorithm was employed. Being widely used for the mixture-density parameter estimation problem, the EM algorithm is particularly attractive when the mixture is Gaussian, since analytical expressions for the expectation step can be derived in this case.

The proposed method was tested on a simple racing car model. The results of this simple model provided examples of specific cases in which the targets' states fail to be resolved. Estimations are poor in cases where targets' trajectories either intersect, get very close to each other or cross the boundary. This failure is due to the fact that the EM algorithm is highly sensitive to the output of the PHD. In the case where targets are close to each other, the unnormalized density function (ie., the output of the PHD) collapses to a single peak, making it impossible for the EM algorithm to resolve more than one target's state. In a similar fashion, when a target crosses the boundary, an additional local maximum of the unnormalized density function comes into existence, which again obstructs the work of the

EM algorithm. Additional analysis of the coupling between the PHD and targets’ states estimation algorithm should be done in order to overcome these obstacles.

4.7 Future Work

There are a number of possible ways of extending the work that has been presented in this paper. Updating the motion model is a simple way to eliminate the boundary conditions that cause problems with estimating the targets’ states. This would eliminate one of two ways that the state estimation algorithm can fail. Aside from this extension to the motion model, there has been some interest in comparing various state estimation methods and testing the ways in which those methods interact with the PHD. In addition, further exploration remains to be made of the PHD.

Updating the motion model is a relatively simple way to overcome the problem posed by the extra maximums of the unnormalized density function that appear when targets cross the boundaries. A variant of the racing car problem, which eliminates the looping at the endpoints, could be used to confirm that the EM algorithm resolves the targets’ states correctly when the states are not near each other. Tracking failures of the EM algorithm due to the cyclic boundary conditions are presented in the Results section and illustrated in Figures (4.7) and (4.10).

There are other methods of estimating the targets’ states which are not discussed in this manuscript and which can serve to extend this preliminary research. The most common of these is the Nearest Neighbour Clustering method, which was briefly mentioned in the State Estimation and the EM Algorithm section. However, there are also more modern approaches, such as the Cheeseman Clustering and the Fourier/Wavelet based metric minimization methods. A careful and unbiased comparison of these competing methods would be of interest to tracking practitioners in both industry and academia.

4.8 Appendix

The following code is the main routine that is used to track and estimate the targets’ state.

```
%----------------------------------------------------------------------------------------------------
% Main Script
%----------------------------------------------------------------------------------------------------

clear;                            % clear all variables
global N;                         % number of particles
global M;                         % number of targets
global PD;                        % probability of detecting a target
global ClM;                       % mean for Poisson distribution; used to generate clutter

% assigning values to the global variables
ClM=5;
PD=1;
N=5000;

M=floor(5*rand)+1;                % the number of targets
Tar=init_states(M);               % getting initial positions of the targets
[Mk,Xk]=init;                     % getting particles and guessing the number of targets
TS = 100;                         % number of time steps

MM=zeros(1,TS);                   % array for storing the mass of the system at each time step
XX=zeros(N,TS);                   % two dimensional array for storing particles at each time step
TT=zeros(M,TS);                   % two dimensional array for storing targets’ states at each time step
```

```
stop_flag=-1;                     % a stopping flag

for t=1:TS
  Tar=move(Tar);                  % moving targets
  Obs=observe(Tar);               % observing targets
  [Mk,Xk]=update1(Mk,Xk);         % updating mass and particles,  k|k   -> k+1|k
  [Mk,Xk]=update2(Mk,Xk,Obs);     % updating mass and particles,  k+1|k -> k+1|k+1
  MM(t)=Mk;                       % saving the mass of the system
  XX(:,t)=Xk(1,:)’;               % saving particles
  TT(:,t)=Tar(1,:)’;              % saving targets

  % EM is applied only after the 6th time step; this ‘if’ statement provides initial guess
  if(t==6)
    Me=round(mean(MM(1:t)));      % system mass is estimated as an average value of the previous masses
    Mo=round(mean(MM(1:t-1)));    % old estimation for the mass of the system
    Me=2;                         % we fixed the mass, estimation failed when mass varied
    [a,m,d]=get_init_guess1d(Me); % initial guess for EM algorithm

  % this was written to handle the case when the mass changes
  elseif(t>6)
    Me=round(mean(MM(1:t)));      % system mass is estimated as an average value of the previous masses
    Mo=round(mean(MM(1:t-1)));    % old estimation for the mass of the system
    Me=2;                         % we fixed the mass, estimation failed when mass varied
    Mo=2;                         % we fixed the mass, estimation failed when mass varied

    % ‘if’ statement below will never be called, since we keep Mo and Me equal to 2
    % it was written to handle the case when the mass of the system changes

    if(Me<Mo)                               % if mass becomes less we eliminate elements from a,m,d
      ind=1:length(a);                      % getting indices for the array a
      [s1,s2]=sort(a);                      % sorting a
      keep=setdiff(ind,s2(Mo-Me));          % getting indices for the elements that will be kept
      a=a(keep);                            % updating a
      a=a/sum(a);                           % sum of the elements of a should be 1
      m=m(:,keep);                          % updating m
      d=d(:,keep);                          % updating d
    elseif (Me>Mo)                          % if mass becomes greater
      [mv,mi]=max(a);                       % find the maximum element of array a
      a=a([1:Mo mi*ones(1,Me-Mo)]);         % resampling this elements
      a=a/sum(a);                           % sum of the elements of a should be 1
      m(:,Mo+1:Me)=m(:,mi*ones(1,Me-Mo));   % resampling the corresponding element of m
      d(:,Mo+1:Me)=d(:,mi*ones(1,Me-Mo));   % resampling the corresponding element of d
    end;
  end;

  % EM is applied only after the 6th time step
  if(t>=6)
    [a,m,d]=get_it1d(a,m,d,Xk(1,:));        % applying EM
    RES(1,1:M,t)=Tar(1,:);                  % saving targets’ positions
    RES(2,1:Me,t)=m;                        % saving estimated targets’ positions
  end;
end;
```

Next, the functions implementing the motion and observation models are presented.

```
function y=move(x)
%---------------------------------------------------------------------------------------------------
% implements the following motion model:
% X_k+1=X_k+V_k
% V_k+1=(V_k+N(0,0.1))*(2B(p)-1)
%---------------------------------------------------------------------------------------------------
% x is a collection of two dimensional vectors (X_k,V_k) represented as a 2xN matrix

N=size(x);                                                 % size of x
N=N(2);                                                    % number of vectors
```

```
y(1,:)=x(1,:)+x(2,:);                                      % updating position
y(2,:)=(x(2,:)+0.1*randn(1,N)).*(2*(rand(1,N)>0.02)-1);    % updating velocity; here p=0.02

% we have wrapping boundary condition, i.e. 100=0
y(1,:)=y(1,:)-100*(y(1,:)>=100)+100*(y(1,:)<0);


function y=observe(T)
%----------------------------------------------------------------------------------------------------
% implements the following observation model:
% X_o=X+N(0,3)
% plus clutter
%----------------------------------------------------------------------------------------------------
% T is a set of targets represented as a 2xM matrix

global PD;                                       % probability of detecting a target
n=size(T);                                       % size of T
n=n(2);                                          % number of targets
Pd=(rand(1,n)>(1-PD));                           % Pd indicates what targets are observed
cl=clutter;                                      % generating clutter
cn=length(cl);                                   % number of clutter points
ot=sum(Pd);                                      % number of observed targets
y=zeros(1,cn+ot);                                % number of observations (clutter+observed targets)
y(1:cn)=cl;                                      % storing clutter

% making sure we observe at least one target
if(ot˜=0)                                        % storing observed targets
  y(cn+1:cn+ot)=T(1,find(Pd>0))+3*randn(1,ot);
end;
```

The following functions implement the motion and observation update steps.

```
function [M,X]=update1(m,x)
%----------------------------------------------------------------------------------------------------
% first update
%----------------------------------------------------------------------------------------------------
M=m;                                            % mass remains the same
X=move(x);                                      % particles move according to the motion mod


function [M,X]=update2(m,x,Z)
%----------------------------------------------------------------------------------------------------
% second update
%----------------------------------------------------------------------------------------------------
% m is the mass of the system
% x contains particles
% Z contains observations

global N;                                       % number of particles
global PD;                                      % probability of detecting a target
global ClM;                                     % mean for Poisson distribution; used for clutter
cc=ClM/100;                                     % useful constant

Ii=I(Z,x(1,:),m);                               % computing integral
M=sum(Ii./(cc+Ii))+(1-PD)*m;                    % updating the mass of the system

% the following code generates weights used for particles resampling
tmp=zeros(length(Z),N);
for i=1:length(Z)
  dif=min(Z(i)-x(1,:),100-Z(i)+x(1,:));         % accounting for wrapping boundary condition
  tmp(i,:)=PD*normpdf(0,dif,3)/(cc+Ii(i));      % main part of the weights formula
end;

S=size(tmp);
% need to handle the case of one target separately
```

```
if(S(1)>1)
  w=sum(tmp)+(1-PD)*m;
else
  w=tmp+(1-PD)*m;
end;

w=w/sum(w);                            % normalizing weights
reg=cumsum(w);                         % generating intervals of lengths corresponding to the weights
samp=rand(1,N);                        % just random numbers

% we use the following algorithm for resampling:
% if the k-th random number (from ‘samp’) hits the j-th interval (in ‘reg’)
% then new k-th particle is the old j-th particle
X=zeros(size(x));
for i=1:N
  X(:,i)=x(:,bisearch(reg,samp(i)));   % resampling
end;


function m=bisearch(x,x0)
%----------------------------------------------------------------------------------------------------
% binary search
% searching for the element of array x that is the closest to x0
%----------------------------------------------------------------------------------------------------

N=length(x);                           % length of x
flag=1;                                % useful variable
l=1; r=N;                              % initially, left position is 1 and right position is N

while(l<=r & flag==1)
  m=floor((r+l)/2);                    % computing the middle point
  if(x(m)==x0)                         % if we find x0 exactly
    flag=0;                            % stop the loop
  elseif (x0 < x(m))                   % if x0 lies in the left half
    r=m-1;                             % update the right position
  else
    l=m+1;                             % update the left position
  end;
end;

if(flag==1)                            % a little adjustment
    m=l;
end;


function y=I(z,x,m)
%----------------------------------------------------------------------------------------------------
% computes integral
%----------------------------------------------------------------------------------------------------

global N;                              % number of particles
global PD;                             % probability of detecting a target
y=zeros(size(z));

for i=1:length(z);
    y(i)=sum(normpdf(z(i),x,3));
end;
y=PD*m*y/N;
```

Note, the above group of functions implements the particle filter. Here are functions implementing the EM algorithm. The first is the function that computes the initial guess.

```
function [a,m,d]=get_init_guess1d(M)
%----------------------------------------------------------------------------------------------------
% provides initial guess for EM algorithm; 1-dimensional case
% M - estimated number of targets
%----------------------------------------------------------------------------------------------------

a=ones(1,M)/M;                         % each a_i is just 1/M
m=100*rand(1,M);                       % means are distributed uniformly; only position is estimated
d=5*ones(1,M);                         % variance is 5ˆ2
```

The following function implements one iteration of the EM algorithm.

```
function [a,m,d]=update1d(a_i, m_i, d_i, x)
%----------------------------------------------------------------------------------------------------
% updates values of a,m,d (1-dimensional case) using the corresponding formulae
%----------------------------------------------------------------------------------------------------

N=length(x);                           % number of particles
Pm=P1(a_i, m_i, d_i, x);               % computing P_l_i
Sp=sum(Pm’);
a=Sp/N;                                % computing a
mn=length(a_i);
m=zeros(1,mn);

% computing means
for L=1:mn
  m(L) = dot(x,Pm(L,:))/Sp(L);
end;

d=zeros(size(d_i));

% computing variances
for L=1:mn
  d(L)=dot(Pm(L,:),(x-m(L)).ˆ2)/Sp(L);
end;
```

And the last function implements the EM algorithm itself.

```
function [a,m,d]=get_it1d(a,m,d,x)
%----------------------------------------------------------------------------------------------------
% computes optimal a,m,d (1-dimensional case)
%----------------------------------------------------------------------------------------------------

a_o=a;m_o=m;d_o=d; % storing old values
flag=1;
epsilon=0.0001;

% iterating while distance is greater than epsilon; flag is used only for the first step
while(dist(a,m,d,a_o,m_o,d_o)>epsilon | flag==1)
 if flag==1
    flag=0;
 end;
    a_o=a;m_o=m;d_o=d;
    % updating values of a,m,d
    [a,m,d]=update1d(a_o,m_o,d_o,x);
end;


function y=dist(a1,m1,d1,a2,m2,d2)
%----------------------------------------------------------------------------------------------------
% distance function (it’s just a sum of norms)
%----------------------------------------------------------------------------------------------------

y=norm(a1-a2)+norm(m1-m2)+norm(d1-d2);
```

Bibliography

[1] RONALD MAHLER, *Approximate Multisensor-Multitarget Joint Detection, Tracking, and Identification Using a First-Order Multitarget Moment Statistic*, Technical Report, Lockheed Martin, 2000.

[2] I.R. GOODMAN, R.P.S. MAHLER, AND H.T. NGUYEN, *Mathematics of Data Fusion*, Kluwer Academic Publishers, 1997.

[3] JEFF A. BILMES, *A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models*, Technical report, International Computer Science Institute, Berkeley California, 1998.

[4] BA-NGU VO, SUMEETPAL SINGH, ARNAUD DOUCET, *Sequantial Monte Carlo Implementation of the PHD Filter for Mulit-Target Tracking*. To appear in the proceedings of Fusion 03.

Chapter 5

Methods to Localize Shorts Between Power and Ground Circuits

Mark Braverman[1], Shengyuan Chen[2], Marcio Gameiro[3], Nadine Gärtner[4], Yasong Jin[5], Edward Keyes[6], Fadil Santosa[7], Haris Widjaya[8]

Report prepared by Fadil Santosa (santosa@ima.umn.edu)

5.1 Introduction

In the competitive world of microprocessor design and manufacturing, rapid advancements can be facilitated by learning from the components made by one's closest rivals. To make this possible, Orisar Inc. (formerly Semiconductor Insights) provides reverse engineering services to integrated circuit (IC) manufacturers. The process produces a circuit diagram from a chip and allows the manufacturer to learn about a competitor's product. These services are also used to determine if any intellectual property infringements have been committed by their competitor.

Reverse engineering of integrated circuit is made difficult by the shrinking form factor and increasing transistor density. To perform this complex task Orisar Inc. employs sophisticated techniques to capture the design of an IC. Electron microscope photography captures a detailed image of an IC layer. Because a typical IC contains more than one layer, each layer is photographed and physically removed from the IC to expose the next layer. A noise removal algorithm is then applied to the pictures, which are then passed to pattern recognition software in order to transfer the layer design into a polygonal representation of the circuit. At the last step a knowledgeable human expert looks at the polygonal representation and inputs the design into a standard electronic schematic with a CAD package.

[1] University of Toronto
[2] University of British Columbia
[3] Georgia Institute of Technology
[4] Clemson University
[5] University of Kansas
[6] Orisar Inc.
[7] University of Minnesota
[8] Simon Fraser University

At the first few stages of the process, errors can be introduced into the polygonal representation of the design. Errors can exist in the form of noise or simply be the result of a physical obstruction on the layer being photographed. These errors can introduce invalid components or connections within the device, and the most catastrophic of these errors results when the power and ground networks of an IC are tied directly together. Since the typical first step in determining component functionality is the determination of power and ground, these short circuits or *shorts* must be identified and corrected before further analysis can occur. It should be emphasized that it is a very rapid process to determine the existence of a short however finding the *location* of a short is another matter.

In the current work flow, workers at Orisar Inc. visually inspect the polygonal representation of the circuit in order to find errors. To completely remove all the shorts one person needs to spend one day inspecting the polygonal data by running signal tracing. To make the search faster, the search area is narrowed down by one quarter by running signal tracing on a quarter of the IC at a time, the shorts can then be quickly narrowed down to small sub-regions.

However this process, performed manually, is very time consuming. We propose a method which can be easily automated thereby saving valuable worker time and accelerating the process of reverse engineering.

5.2 Problem Description

A modern IC design contains in the order of millions logic gates packed into a very small area. The gigantic number of devices, together with minimal space, create a problem for designers to connect their logic gates efficiently. This is the main reason for the multi layered approach to IC manufacturing. A typical IC has one layer dedicated to the placement of logic gates and multiple layers of wiring. This is necessary because all the gates on the gate layer are packed as densely as possible, leaving no room for interconnecting wires and connection to the power supply.

In a modern IC there are many more wiring layers than logic gate layers, and typically adding a layer to an IC translates directly into an increase in the cost of manufacturing. Thus any manufacturer tries to pack the wires as densely as possible within the layer without violating the design rules. Because an IC consists predominantly of wiring, it is crucial to get the wiring design transfered properly. There are three types of wiring we can find within an IC.

- Power lines which we will denote as V_{dd}. These lines carry the positive charge into the components of the IC.
- Ground lines which we will denote as V_{ss}. These lines carry the negative charge.
- Signal line which interconnect the various components.
- Vias, this is the inter-layer wiring that connects wires to wires, and wires to logic gates.

The two main wires that we will focus on are the power lines and the ground lines. On a normal IC, the power lines and the ground lines form two disconnected components. Unfortunately nature is against us in this process. The errors mentioned in the introduction can add wires that are not part of the real IC design, and create inadvertent connections, or shorts, between the power lines and the ground lines.

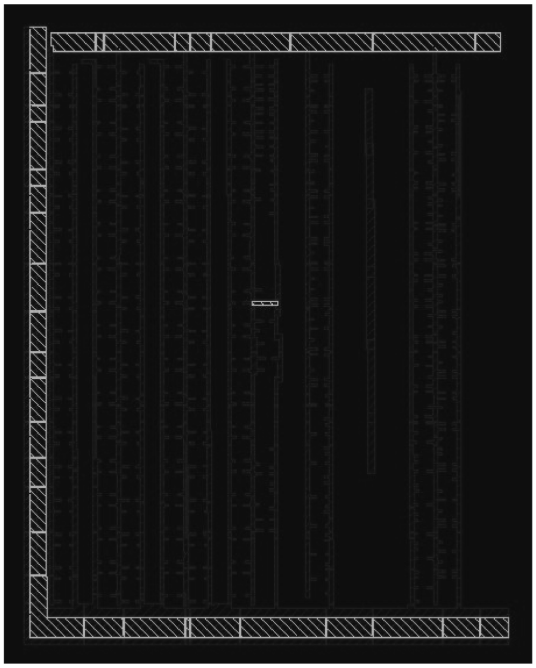


Figure 5.1: A polygon representation of wires in an IC.

The wires in the IC are represented as a set of overlapping polygons. Each polygon contains a list of vertices and a signal number. The number of vertices in each polygon can be very large although many of the polygons are quadrilaterals. The polygons are not guaranteed to be convex, and they can vary greatly in size. The wires are typically laid out in a regular fashion, that is wires run either parallel or or orthogonal to other wires.

In the next section we will explain the approaches that we have considered to solve this problem, and the choices we made concerning the final algorithm.

5.3 Approaches to the Problem

5.3.1 A Network Flow Method

If current can flow from a V_{ss} pin to a V_{dd} pin, it must go through one of the shorts. Normally a short is much narrower than a legitimate wire. Given the fact that the wider a wire is, the more capacity it bears, shorts become the bottleneck of the network. This idea is already used in manual inspections where power is applied between two pins and a thermal image of the circuit is made to identify hot regions.

We would like to simulate this process in a computer. The first step would be to convert the circuit into a network. Our thought is to translate a given circuit into a capacitated network where each edge in the network is a piece of wire. This leave the problem of estimating the capacity of a wire. Several ideas were proposed:

1. Starting from the scanned image, we may regard every black pixel as a node of the network, and assign each edge capacity one. This approach leads to a massive network, which may be too

expensive to compute.

2. Starting from the polygons representation of a wire, determine the direction of flow and estimate the width of the wire. This process can be very complex with many special cases as a polygon could be as simple as a quadrilateral and as complicated as a nonconvex n-gon.

3. Another idea is to extract the network from the scanned image by scanning through all pixels vertically and locating junctions and edges. Even though the idea is very attractive, the group felt that it was unlikely that an algorithm to do this can be completed in the time given.

We remark that upon transforming the circuit into a finite capacity network, we can use network flow techniques to locate the shorts. The max-flow of such a network is limited by the shorts, and thus the min-cut will be on the edges of these shorts. We assume that there are only a few shorts and the sum of the capacities of all these shorts are less than that of a narrow wire.

5.3.2 Signal Processing to Locate H-Junctions

The observation that shorts tend to make **H**-junctions between two legitimate wires lead us to consider pattern recognition algorithms. It was felt that such an algorithm, though it can identify false positives, could be valuable in localizing the search. After some consideration, it was felt that such a method is not sufficiently general.

5.3.3 Exploiting the Incidence Matrix

Since the incidence matrix encodes the connectivity of the circuit, it seems possible that we can devise a way by which the matrix is manipulated so that it separates out into power and ground parts, with the shorts connecting them. To be able to do this, we must develop an objective functional such that when it is minimized, the matrix segregates as desired. This appealing idea deserves further study.

5.3.4 A Monte Carlo Approach

We convert a circuit into a graph by letting the polygons representing the circuit be vertices. If two polygons are connected, then the vertices representing them are connected by an edge. With this description, we can define the length of a path between a pair of polygons. The nice feature of shorts is that it must appear on a path starting from a polygon on the V_{dd} (power) side to a polygon on the V_{ss} (ground) side or vice versa. To utilize this observation, we randomly pick a pair of starting and end points, and compute the shortest path between them using the breadth first algorithm. We repeat this procedure a large number of times, and we count the number of times each polygon appeared in all of these paths.

We claim that the polygons from a short will have a higher number of visits that other polygons, thus allowing us to isolate potential shorts. This approach, as well as an implementation, are described in the next section.

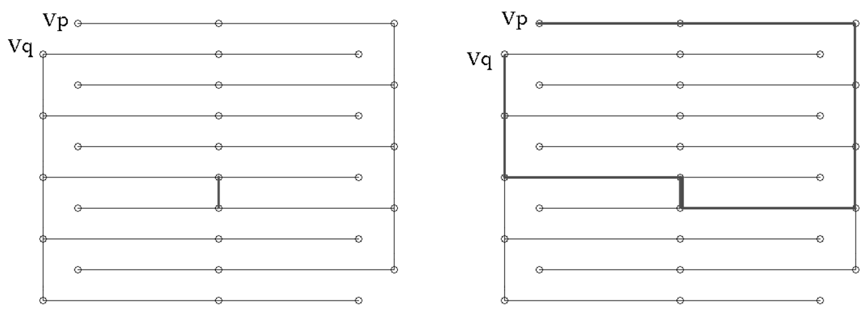


Figure 5.2: The shortest path between v_p and v_q in a simple circuit.

5.4 A Monte Carlo Algorithm

5.4.1 Description of the Algorithm

To make it possible to apply graph theory algorithms to the problem, we represent the circuit as an undirected graph G. The vertices of the graph correspond to the polygons representing the circuit and the edges represent the connection between adjacent polygons. For two vertices v_p and v_q, corresponding to polygons P and Q, there is an edge connecting v_p and v_q if and only if the polygons P and Q touch each other. A path between two vertices v_p and v_q in G is a path between the polygons P and Q in the circuit.

If a circuit has a short, then any path going from a polygon in V_{dd} (power) wire to one in V_{ss} (ground) wire must pass through a short. Since there are only few shorts in the circuit, we would expect many paths to go through each of the shorts. Our algorithm randomly chooses pairs of vertices in the circuit, connects them through a shortest path, and keeps track of the number of visits to each vertex of the circuit. While any path between these vertices would work, we choose the shortest path (see Figure 1.2). We expect that the shorts would be the most visited areas in the circuit.

We keep a counter for the number of visits to each of the vertices. At each iteration of the algorithm we randomly choose two vertices v_p and v_q. We then find a randomized shortest path from v_p to v_q using a randomized version of the **breadth first search** (BFS) algorithm. Then we increase the counter of each of the visited vertices by 1.

We make a large number of iterations, and output a colour map highlighting the most visited areas. Then we manually search for the shorts in the highlighted areas.

```
The Algorithm
1) N = number of trials;
2) For each $p$, $counter(v_p) = 0$;
3) For i = 1 to N
        3.1) randomly choose vertices $v_p$ and $v_q$;
        3.2) find one of the shortest paths from $v_p$ to $v_q$,
             $v_p := v_0, v_1, v_2, \ldots, v_K =: v_q$ using a randomized BFS;
        3.3) For j=0 to K
                 3.3.1) $counter(v_j) = counter(v_j) + 1$;
        Endfor
Endfor
4) Output a colour map with the values of $counter(v)$ for each $v$, the polygons with high counter
values are the primary suspects to be the shorts.
```

Randomized BFS is an algorithm which randomly chooses one of the shortest paths from V_P to V_Q. Details will be explained in the end of the section.

5.4.2 Analysis of the Algorithm

The main idea is that every time we choose v_p from V_{dd} and v_q from V_{ss} or v_p from V_{ss} and v_q from V_{dd}, the path from v_p to v_q must pass through one of the shorts of the circuit. Next we estimate the probability that the chosen v_p and v_q are in different wiring nets? Suppose that the portion of V_{ss} polygons in the net is s, and the portion of V_{dd} polygons is d, $s + d = 1$. Then the probability that we choose v_p from V_{dd} and v_q from V_{ss} is ds and the probability that we choose v_p from V_{ss} and v_q from V_{dd} is sd, so the probability that v_P and v_q are chosen from different wiring nets is $2ds$. This probability is almost 50% if s and d are relatively close to each other. We expect this to be the case most of the time. The following table summarizes the estimated outcomes

The portion s of V_{ss} in the net	The portion d of V_{dd} in the net	The probability that v_p and v_q are chosen from different nets.
10%	90%	18%
20%	80%	32%
30%	70%	42%
40%	60%	48%
50%	50%	50%
60%	40%	48%
70%	30%	42%
80%	20%	32%
90%	10%	18%

If there are k shorts in the circuit, then each of the short points gets visited

$$\frac{2 \cdot d \cdot s \cdot N}{k}$$

times on average, where N is the number of trials. We have

$$\frac{2 \cdot d \cdot s \cdot N}{k} \approx \frac{N}{2 \cdot k}$$

when s and d are close to each other. This number is much larger than the number of visits to an average polygon in the circuit. So the shorts should get highlighted by the algorithm.

A possible improvement to the algorithm would be discarding particularly short paths. A typical path connecting a point in V_{ss} with a point in V_{dd} going through one of the shorts and is generally longer than a path connecting two points on the same side of the net. Hence discarding short paths would discard more paths which connect vertices on the same side, thus yielding a better performance of the algorithm.

Note that the algorithm is extremely parallelizible. All the trials can run completely independently. So the algorithm can be executed simultaneously by arbitrarily many machines in parallel, and the visits statistics can be collected at the end of the run.

5.4.3 The Randomized BFS Algorithm

We use the classical BFS (breadth first search) algorithm for finding a shortest path in an undirected, unweighted graph. It is a classical algorithm described in many introductory algorithms books. A good exposition of the algorithm can be found in Cormen [1]. We modify this algorithm in order to choose a random shortest path.

After scanning the graph from the source s and labelling all the vertices according to their distance from s, we backtrack from the target t in order to find the shortest path from t to s. Whenever we have more than one way to backtrack, a random choice is made amongst the possibilities. This procedure diversifies the possible routes in congested areas and thus avoids false positives.

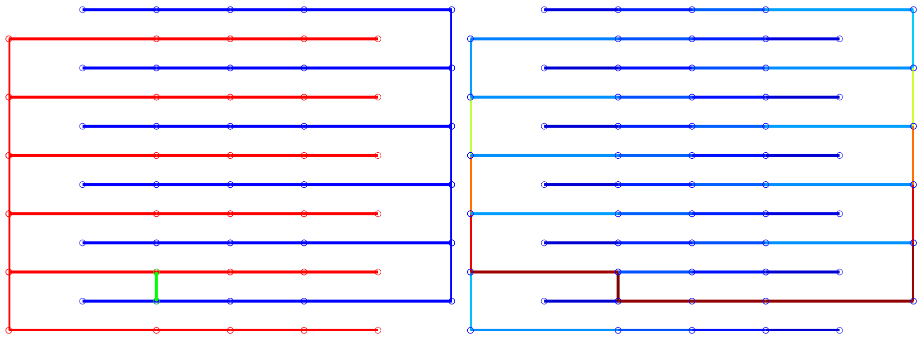
5.5 Performance of the Randomized Algorithm

On a typical circuit we will have approximately 10^5 nodes in the graph, with the number of shorts in the order 10^1. The testing is done in increasing node density to approximate real cases, this is important because our method works under the assumption that the number of shorts are a few orders of magnitude smaller, thus the probability of it being hit by our randomized tracing algorithm is higher than a real node. The following are the results.

In the first experiment we have one short in between the two interdigitated power and ground wires as shown in Figure 5.3. The power wires are coloured red and the ground coloured blue. The short is shown as the green line in Figure 5.3a. The result of our algorithm, with 5000 trials, is shown in Figure 5.3b, it is evident that our algorithm has succeeded in localizing the area of the probable short shown by the red wires in the picture.

In the next experiment we tested our algorithm on a circuit with a higher node density. The circuit contains two shorts. It is clear that our algorithm is able to find a local area where the short occurs (see Figures 5.4). Here, $N = 5000$ trials as well.

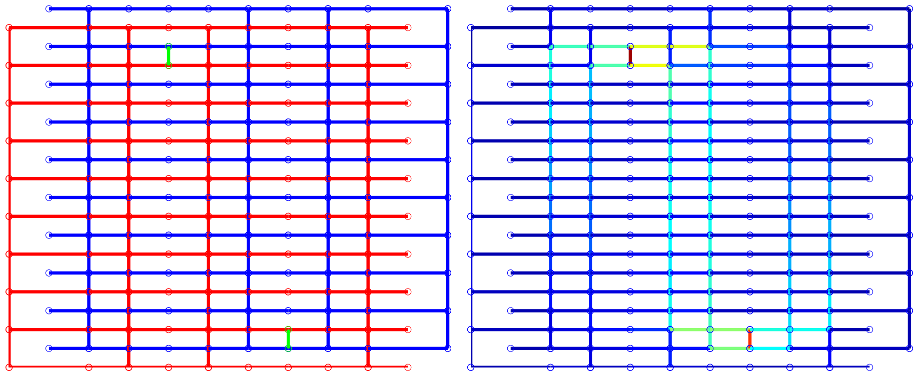
In the next set of figures we ran the algorithm on a circuit that has 3 shorts. After a first run involving 5000 trials, only one short was found (Figure 5.5b). We repeated the run after removing the



(a) Initial figure with one short.

(b) Final figure.

Figure 5.3: Localization of 1 short.



(a) Initial figure with two shorts.

(b) Final figure.

Figure 5.4: Localization of 2 shorts in 1 run.

short found in the first run and identified a second short in Figure 5.5d. A final run, after removing the second short was able to determine the location of the third and final short (Figure 5.5f).

We have numerical evidence that our algorithm coupled with simple heuristics is capable of removing all the shorts that exist within a given circuit model. We have also tested our algorithm on real data that the company uses.

Figure 5.6a shows a case from a real circuit that has been annotated and identified by a knowledgeable human expert. The white lines in Figure 5.6b are areas our algorithm has identified as possibly containing shorts. This illustrates that our algorithm correctly identified the local areas where shorts occur.

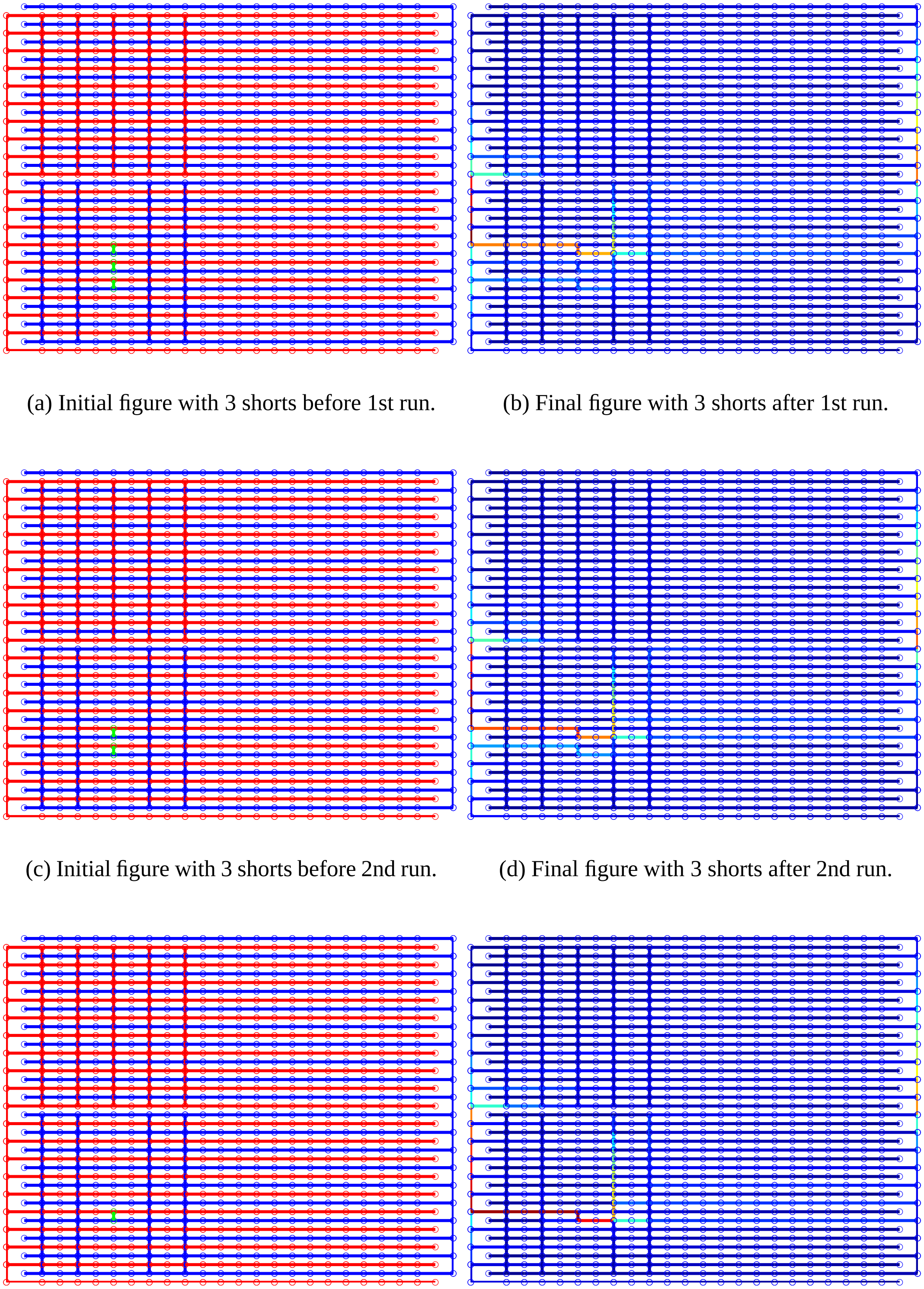
5.6 Conclusions

The team has implemented and tested a Monte Carlo approach capable of localizing power to ground short circuits. Any valid wiring network consists of two unconnected power and ground networks. The key observation is that given any two randomly chosen points in the wiring network, there exists a relatively high probability that the shortest path joining these two points will contain a short circuit if it exists. Our algorithm exploits this relatively high probability of traversing a short and has been shown to be capable of detecting multiple shorts of both simulated and actual IC power wiring networks.

There are several ways in which the proposed method can be improved. These include:

- Make better use of path information.
- Incorporate knowledge embodied in the incidence matrix of the graphs.
- Exploit properties of the circuits, such as line widths.

We believe that other ideas explored in this report deserve further investigation.



(a) Initial figure with 3 shorts before 1st run.

(b) Final figure with 3 shorts after 1st run.

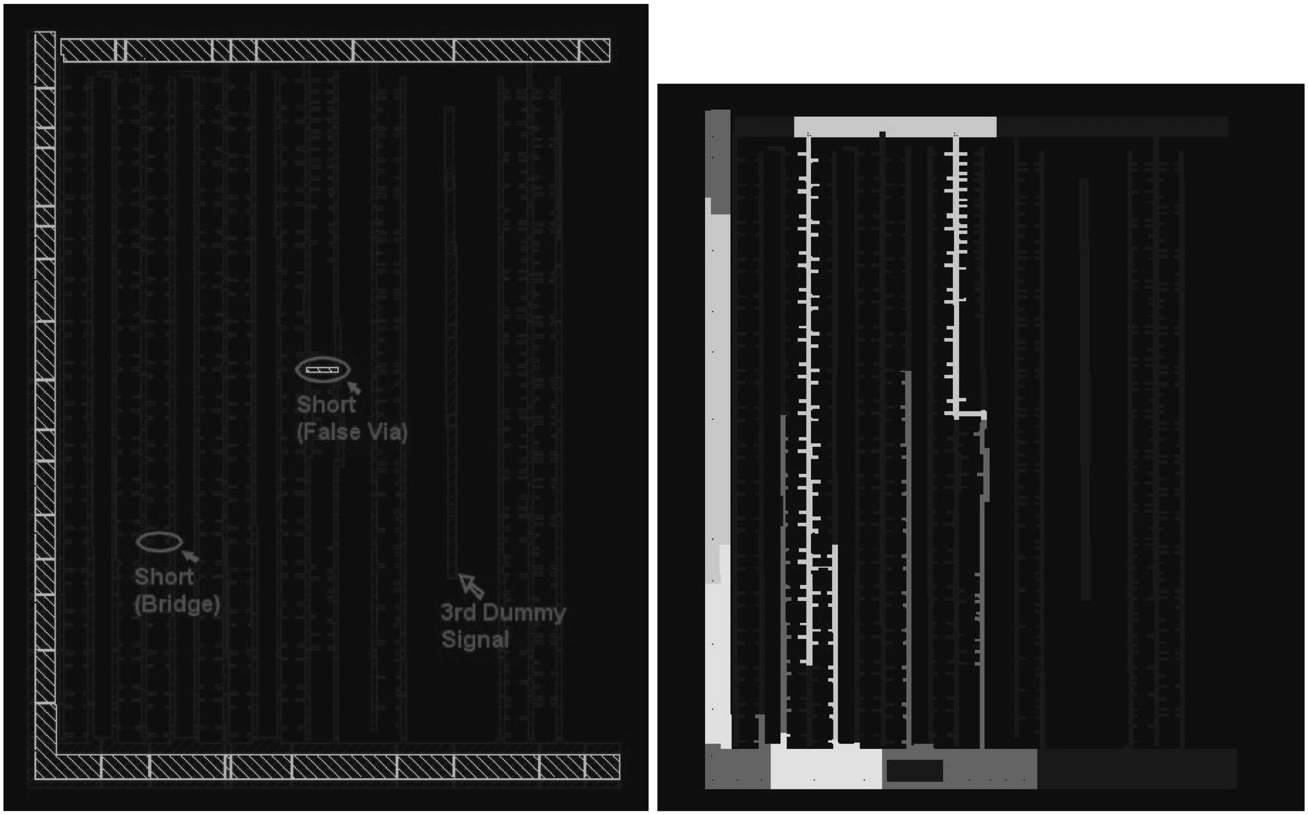
(c) Initial figure with 3 shorts before 2nd run.

(d) Final figure with 3 shorts after 2nd run.

(e) Initial figure with 3 shorts before 3rd run.

(f) Final figure with 3 shorts after 3rd run.

Figure 5.5: Localization of 3 shorts in 3 steps.



(a) The areas where shorts occur.

(b) The highlighted by our algorithm after 5 iterations.

Figure 5.6: The result of our algorithm with real data.

Bibliography

[1] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, *Introduction to algorithms*, McGraw-Hill, New York, (1997).

Chapter 6

Correlation Structures Corresponding to Forward Rates

Ilona Kletskin[1], Seung Youn Lee[2], Hua Li[3], Mingfei Li[4], Rongsong Liu[5], Carlos Tolmasky[6], Yujun Wu[7]

Report prepared by Seung Youn Lee (seunglee@math.ohio-state.edu)

6.1 Introduction

In finance, there is a constant effort to model future prices of stocks, bonds, and commodities; the ability to predict future behaviour provides important information about the underlying structure of these securities. While it has become common to model a single stock using the Black-Scholes formulation, the modelling of bond prices requires one to simulate the change of interest rates as a function of their maturity, which requires one to model the movement of an entire yield curve. If one studies the spectral decomposition of the correlation matrix corresponding to the spot rates from this curve, then one finds that the top three components can explain nearly all of the data; in addition, this same structure is observed for any bond or commodity. In his 2000 paper, Ilias Lekkos [4] proposes that such results are an artifact due to the implicit correlation between spot rates, and that the analysis should instead be performed using forward rates. In this paper, we discuss the results obtained for the spectral structure of the correlation matrices of forward rates, and investigate a model for this associated structure. The paper is divided into four parts, covering forward rates background material, principal components analysis, yield curve modelling, and conclusions and research extensions.

[1] University of Toronto
[2] Ohio State University
[3] University of Calgary
[4] Michigan State University
[5] York University
[6] Cargill Inc.
[7] University of Kentucky

6.2 Background: Forward Rates

6.2.1 Spot Rates

Let us begin with a few definitions and concepts from financial mathematics that will be referred to throughout the paper. To model bond prices, one must know the yields for various maturities. These interest rates, as a function of maturity, constitute the yield curve and are referred to as spot rates. The spot rate $R(T)$ gives the rate that must be paid when money is borrowed (or loaned) today for a time T years. Since each spot rate changes with time, we are interested in knowing the movement of the entire yield curve as time proceeds. When one studies a single stock, and assuming efficient markets, its movement may be predicted using the Black-Scholes formulation:

$$\frac{dS}{S} = \mu \, dt + \sigma dW$$

with S referring to the stock price, μ its expected return, and σ its volatility. W is a Brownian motion representing the random movement of the stock. To study the movement of an entire yield curve, we may assume that each point moves as a Brownian motion. Since the correlation structures, and hence primary movements, of spot rates are well known and will be briefly mentioned in the next section, and since we are interested in studying the correlation structures of forward rates in this paper, let us now adapt the above formulations to focus on forward rates.

6.2.2 Forward Rates

A forward rate is the rate applied to borrow (or loan) money between two dates, T_1 and T_2, determined today at time t; we denote this as $f(t, T_1, T_2)$. In order that no-arbitrage conditions hold, we must have the following relationship between forward and spot rates:

$$e^{R_i T_i} e^{f(T_i, T_{i+1})(T_{i+1} - T_i)} = e^{R_{i+1} T_{i+1}}.$$

The formula simply states that the rate to borrow money starting from today to time T_{i+1} must be the same as the rate if one borrows from today until time T_i, and then from T_i to T_{i+1}. If this equation did not hold, one could borrow money at one rate and lend at another with no risk, thereby creating an arbitrage opportunity. For completeness, let us also define the instantaneous forward rate, which is the rate applied to borrow or lend money for an instant at time T_1, determined at time t, denoted $f(t, T_1)$.

In our work, we are interested in following the approach of Heath, Jarrow, and Morton [2] to model the entire forward rate curve directly. As an example of the type of changes that have taken place in forward rates historically, Figure 6.1 illustrates the movements of various forward rates as a function of time using data from the US.

As previously mentioned, while Black-Scholes is used to model a single stock, the modelling of an entire curve of forward rates will require more work. The formulation proposed by Heath, Jarrow, and Morton is a generalization of Black-Scholes; it is given by the formula:

$$\frac{df(t,T)}{f(t,T)} = \mu dt + \left(\sum_{i=1}^{\nu} \sigma_i(t,T) dW_i \right),$$

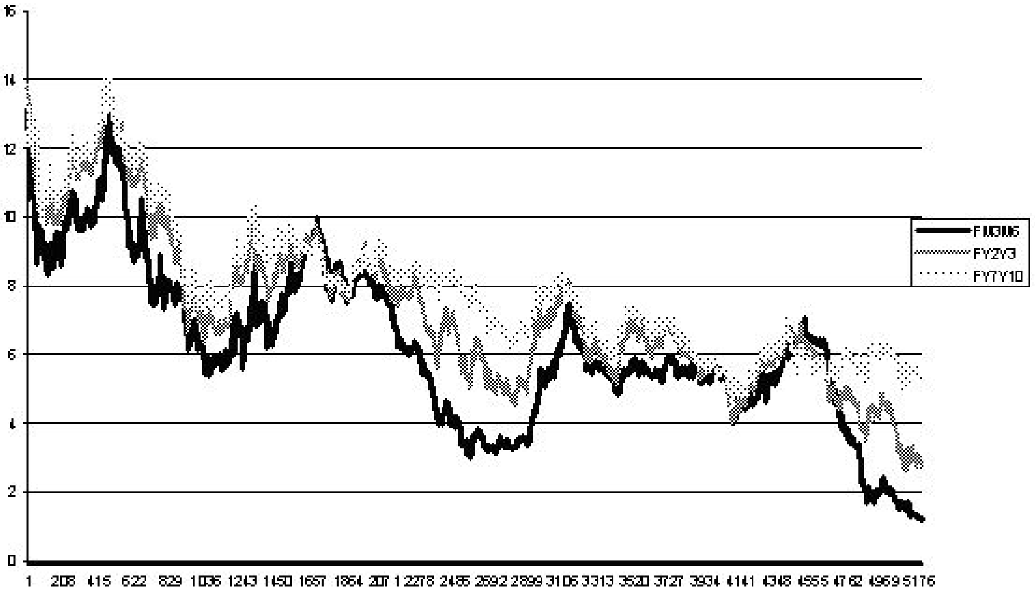


Figure 6.1: Historical Forward Rate Data for the US (x-axis : days; y-axis : percentile).

where the differential is taken with respect to time, so that

$$df(t,T) = f(t+h,T) - f(t,T).$$

The main question now arises as to what value of ν should be used in the summation. Clearly, if we select $\nu = 1$, then we return to modelling a single quantity, which would incorrectly imply that the forward rates are completely correlated. If, however, we allow ν to be the number of points on the curve, then we find that this computation is too costly, and we are not taking into account the fact that rates do indeed have a non-zero correlation. Our goal is to reduce the dimensionality by recovering most of the variances and covariances of the forward rates with a minimal number of components, ν. This can be accomplished using principal components analysis.

6.3 Principal Components Analysis (PCA)

PCA is a statistical procedure that aims at taking advantage of the possible redundancy in multivariate data. It achieves that by transforming p (possibly) correlated variables into ν uncorrelated ones. If the original variables are correlated, then the data is redundant and the observed behaviour can be explained by just ν components of the original variables with $\nu < p$.

This procedure performs PCA on the selected dataset. A principal component analysis is concerned with explaining the variance-covariance structure of a high dimensional random vector through a few linear combinations of the original component variables. Consider a p-dimensional random vector $X = (X_1, X_2, ..., X_p)$. ν principal components of X are ν (univariate) random variables Y_1, Y_2, . . .,

Y_ν which are defined by

$$\left.\begin{array}{lcl} Y_1 & = & l_1X = l_{11}X_1 + l_{12}X_2 + \ldots + l_{1p}X_p, \\ Y_2 & = & l_2X = l_{21}X_1 + l_{22}X_2 + \ldots + l_{2p}X_p, \\ \vdots & & \\ Y_\nu & = & l_\nu X = l_{\nu 1}X_1 + l_{\nu 2}X_2 + \ldots + l_{\nu p}X_p, \end{array}\right\}$$

where the coefficient vectors $l_1, l_2, \ldots, l_\nu$ are chosen such that they satisfy the following conditions:

> First Principal Component = Linear combination l_1X that maximizes Var(l_1X) and $\|l_1\| = 1$.
>
> Second Principal Component = Linear combination l_2X that maximizes Var(l_2X) and $\|l_2\| = 1$ and Cov(l_1X , l_2X)= 0.
>
> j^{th} Principal Component = Linear combination l_jX that maximizes Var(l_jX) and $\|l_j\| = 1$ and Cov$(l_iX, l_jX) = 0$ for all $i < j$.

This says that the principal components are those linear combinations of the original variables which maximize the variance of the linear combination and which have zero covariance (and hence zero correlation) with the previous principal components.

It can be proved that there are exactly p such linear combinations. However, typically, the first few of them explain most of the variance in the original data. So instead of working with all the original variables $X_1, X_2, \ldots, X_p$, one typically performs PCA and uses only the first few principal components in subsequent analysis.

6.3.1 Spot Rates

We are interested in determining which components describing the movement of our curve can be used to explain most of the variance and covariance data while utilizing as few components as possible. In the case of spot rates, from the previous work in principle component analysis in this field, the results are well known. Let R_i denote a vector of yields for the day $0 \leq i \leq N$ and define the matrix A so the column i of A is the vector $R_i - R_{i-1}$. One can then construct $X = \text{cor}(A)$, the correlation matrix formed from A. Note that $[X]_{i,j}$ gives the correlation between the daily changes in rates with maturity T_i and maturity T_j. Calculating the eigenvalues and eigenvectors of this new matrix, one will find that the top three components are level, slope, and curvature. The first eigenvector, referred to as "level" can be interpreted as a parallel shift in the term structure, the second represents a change in the steepness, and the third is interpreted as a change in the curvature of the yield curve.

Using this process and obtaining the corresponding eigenvalues, we can compute the cumulative percentage of the first M eigenvalues, namely

$$\frac{\sum_{i=1}^{M} \lambda_i}{\sum_{i=1}^{\nu} \lambda_i},$$

where ν is the total number of eigenvalues, as shown in Table 6.1. From the result of this principle component analysis process, we can see that the cumulative total of the top three components are already over 95% of original data, where we use US data as an example. These top three components represent the key movements of the yield curve for spot rates, their form is shown in Figure 6.2.

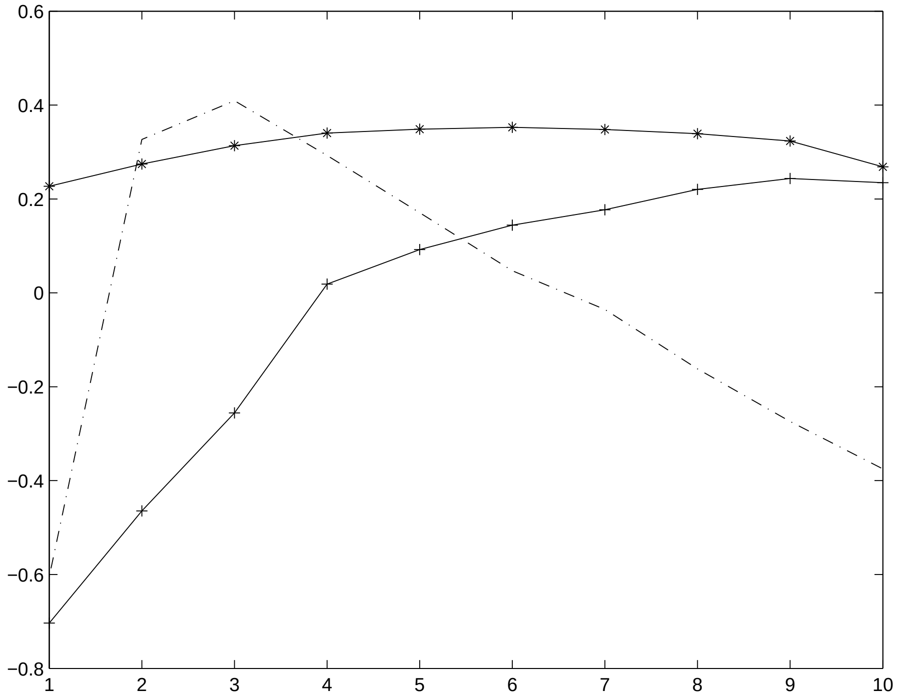


Figure 6.2: Top 3 eigenvectors representing key movements of spot rates in the US (x-axis : maturity; y-axis : eigenvalue component).

While the above graph was generated using US data, in fact we can get the same results regardless of the time period or the market used, and regardless of whether we consider bonds or commodities. In [4], Lekkos argued that such results are an artifact which arises due to the fact that spot rates are highly correlated by construction. He proposes that we should instead be working with forward rates, which although they may be correlated, are not correlated by construction. He claims that the resulting principal component analysis will yield much weaker results.

6.3.2 Forward Rates

As stated above, we are interested in investigating the results when principle component analysis is applied on the correlation matrix for forward rates instead of spot rates.

As before, we calculate the eigenvalues and eigenvectors of the correlation matrix, but for forward rates, we do indeed find that the decay of the eigenvalues is considerably slower, implying that it is not enough to only consider the top three components to adequately explain the movements of the curve. Figure 6.3 is a comparison of the eigenvalues obtained from the correlation matrices of spot and forward rates using 1982–2003 US data.

From this graph, we note that similarly to the top eigenvector for spot rates, the top component for forwards stands out considerably, although it is not as dominant, explaining less than 60% as compared to 80% for spots. If we consider the contribution of the top three components, we find that while these made up over 95% for spots, the total is now less than 80%, owing to the much slower decay of the

Spot Rate Data			
Eigenvalue		Individual variance (%)	Cumulative variance (%)
λ_1	8.0677	80.68	80.68
λ_2	1.1627	11.63	92.31
λ_3	0.2847	2.85	95.16

Forward Rate Data			
Eigenvalue		Individual variance (%)	Cumulative variance (%)
λ_1	5.7776	57.78	57.78
λ_2	0.9425	9.42	67.20
λ_3	0.6609	6.61	73.81

Table 6.1: Principle Component Analysis of US Data.

eigenvalues in the case of forwards.

It is also easy to verify that the first eigenvector in the case of forward rates is still a level movement and that the second still corresponds to slope. Yet, although the first two components can still explain a lot of the total variance, the remaining eigenvectors make up a substantial contribution, and their intuitive meaning, including that of the third eigenvector, is not so clear.

6.4 Yield Curve Modelling

6.4.1 Model Development and Implementation

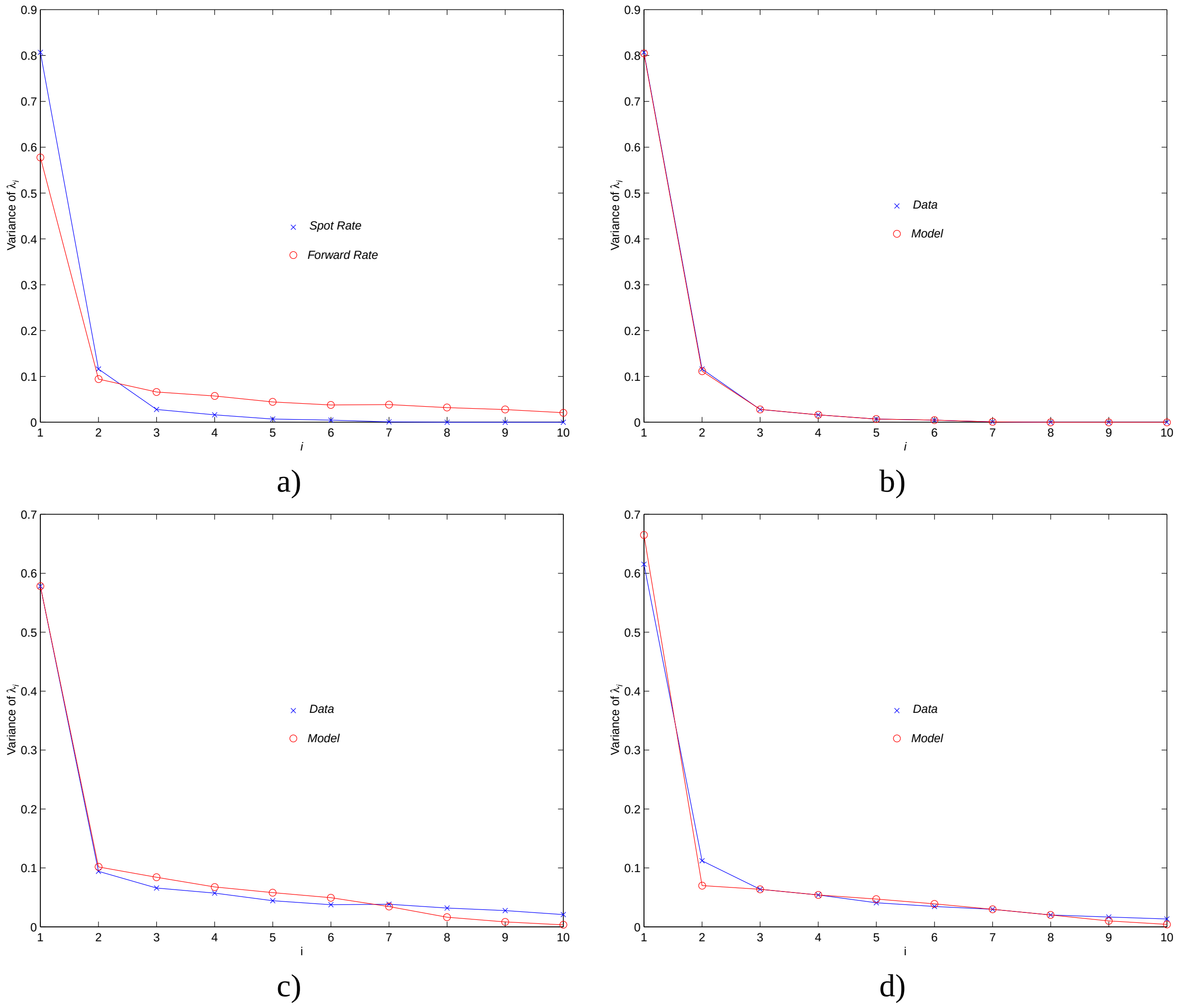
Thus far, we have found that using forward rates instead of spot rates does not produce the same structure for the correlation matrix in which three exceptionally dominant components arise; in fact, the order of the later components may not even be the same as in the case of spot rates. How might we try to model the correlation matrix of the forward rates and its resulting spectral structure? In the case of spot rates, there is an existing model from [1] for the spot rates correlation matrix:

$$[X]_{i,j} = \rho^{|T_i - T_j|}$$

assuming that correlations, ρ, are high enough. Here T is maturity in years.

A comparison of the eigenvalue decay obtained using data and the above model is shown in Figure 6.3b. The circles represent the eigenvalues of the correlation matrix using spot rate data, while the squares stand for the eigenvalues of the modelled spot correlation matrix. We note that the two curves nearly coincide with each other; both of them exhibit a very fast decay and for each of them, the first three eigenvalues are very significant and explain over 95$\%$ of the behaviour of the correlation matrix; the other eigenvalues are insignificant and so the corresponding eigenvectors explain very little about the movement of spot rates. Thus, this model produces a good approximation to the spot rate correlation matrix. To propose a model in the case of forward rates, we can consider the relationship between the covariance matrix for spot and forward rates, namely:

$$\Omega_r = W \Omega_f W^T. \tag{6.1}$$



a)

b)

c)

d)

Figure 6.3: Variance structure of the eigenvalues. a) Spot rates versus forward rates for US data; b) Decay structure for US spot rates (data versus model); c) Decay structure for US forward rates (data versus model); d) Decay structure for European forward rates (data versus model).

Here, Ω_r stands for the covariance matrix for the spot rates, Ω_f stands for the covariance matrix for the forward rates, and W is a matrix of the weights of the forwards to the corresponding spot rates. However, we need to work with the correlation matrix. That means we need to find some way to convert this formula into a relationship between correlation matrices.

Given that the historical variance of the spots is pretty stable across tenors we have assumed constant variance when using formula (6.1) to transform the correlation matrices of the spots into correlation matrices of forwards.

Rearranging the resulting equation, we obtain a model for the forward rate correlation matrix. We may now compare the forwards eigenvalue decay from this modelled correlation matrix with that of the correlation matrix obtained from the data. Figures 6.3c and 6.3d both show such a comparison between model and data; Figure 6.3c illustrates results for 1982-2003 US data while Figure 6.3d presents 1998-2002 European data.

The circles represent the eigenvalues of the correlation matrix for the real data of the forward rates, while the squares stand for the eigenvalues of the modelled forward rates correlation matrix. From these two figures, we observe that for both markets, the model fits the data fairly well, but considerably worse than the fit that was obtained for the spots model earlier. To be specific, it seems that three components are no longer enough to adequately explain the correlation matrix; we may need to use more than five components. Indeed, it is also possible that the spots model, while it seemed to produce a good fit for spots data, is not an adequate foundation for our forwards model, which may be more sensitive to the exact nature of the spots correlation matrix; perhaps a more robust model for the spot rates is necessary when using it as a basis for forwards modelling.

6.4.2 Model Comparison Using Simulations

Since our ultimate goal is to predict forward rates which can then be used to predict bond prices, it is important to perform simulations to determine if using forward rates as we have implemented above, or spot rates (and subsequently computing forward rates) is indeed the best approach. While we know in the case of spot rates that it is sufficient to include the three top components, it still remains to determine how many eigenvectors are necessary when using forward rates. While we have performed some preliminary work for making such a comparison, simulations remain to be done to determine which method best predicts the variance of forward rates, and hence is a better model for predicting future values of forward rates.

6.5 Conclusion

As it is well known, the correlation matrices corresponding to spot rates contain a lot of structure. The fact that this structure is found across markets suggests the possibility that it is due to an artifact and not to any market-specific characteristics. In his work, I. Lekkos argued that forward rates should be the state variables in any such analysis since spot rates are correlated variables by construction. Using interest rate data from the US, Germany, United Kingdom and Japan he showed that the structure present in the correlation matrices when we use forward rates (as opposed to spot rates) is a lot weaker. In this work, we have analysed these type of matrices and found that the forward rate versions of parametric models that have been proposed for spot rate correlations do a fairly good job in describing the data. A lot of work remains to be done as far as understanding these matrices, their commonalities across markets and, of course, their modelling.

Bibliography

[1] L. Forzani, C. Tolmasky, A Family of Models Explaining The Level-Slope-Curvature Effect, International Journal of Theoretical and Applied Finance, 6, 3 May 2003, pp. 239–256.

[2] D. Heath, R. Jarrow, A. Morton, Bond pricing and the term structure of interest rates: a discrete time approximation, Journal of Financial and Quantitative Analysis, 25, 4 December 1990, pp. 419–440.

[3] D. Heath, R. Jarrow, A. Morton, Bond pricing and the term structure of interest rates: a new methodology for contingent claims evaluation, Econometrica, 60, 1 1992, pp. 77–105.

[4] I. Lekkos, A Critique of Factor Analysis of Interest Rates, Journal of Derivatives, Fall 2000, pp. 72–83.

Chapter 7

Product-Driven Data Mining

Rita Aggarwala[1], C. Sean Bohun[2], Rachel Kuske[3], Gerry LaBute[4], Wei Lu[5], Nilima Nigam[6], Fabien M. Youbissi[7]

Report prepared by C. Sean Bohun (csb15@psu.edu)

7.1 Introduction

The behaviour of consumers is believed to be influenced by many factors. Some of these factors include the individuals culture, social status, lifestyle and attitudes. Understanding how these complicated and interrelated factors drive the consumer is the primary goal of Manifold Data Mining. The question posed to the group was to 1) find an algorithm that predicts the likelihood of consumers to respond favourably to a given product. In addition, once this prediction is made for a given consumer the group was also asked to 2) develop a second algorithm that infers other statistical information regarding the consumer.

Manifold Data Mining has developed innovative demographic and household spending pattern databases for six-digit postal codes in Canada. Their collection of information consists of both demographic and expenditure variables which are expressed through thousands of individually tracked factors. This large collection of information about consumer behaviour is typically referred to as a *mine*. Although very large in practice, for the purposes of this report, the data mine consisted of m individuals and n factors where $m \simeq 2000$ and $n \simeq 50$. Ideally, the first algorithm would identify a few factors in the data mine which would differentiate customers in terms of a particular product preference. Then the second algorithm would build on this information by looking for patterns in the data mine which would identify related areas of consumer spending.

To test the algorithms two case studies were undertaken. The first study involved differentiating BMW and Honda car owners. The algorithms developed were reasonably successful at both finding questions that differentiate these two populations and identifying common characteristics amongst

[1] University of Calgary
[2] Pennsylvania State University
[3] University of British Columbia
[4] Manifold Data Mining Inc.
[5] McGill University
[6] Laval University

the groups of respondents. For the second case study it was hoped that the same algorithms could differentiate between consumers of two brands of beer. In this case the first algorithm was not as successful as differentiating between all groups; it showed some distinctions between beer drinkers and non-beer drinkers, but not as clearly defined as in the first case study. The second algorithm was then used successfully to further identify spending patterns once this distinction was made. In this second case study a deeper factor analysis could be used to identify a combination of factors which could be used in the first algorithm. The case studies are discussed in detail in Section 8.

7.2 Latent Variable Models

The initial problem proposal suggested that the method of Projected Latent Spaces could prove fruitful in the first task, that is, in identifying a few factors which could differentiate between consumer preferences on a particular product. In essence this means finding the dominant factors that are closely related to a particular difference between customers, while showing that the remaining factors are not significant. Mathematically, we can view this as trying to represent consumer behaviour in a low dimensional space of factors. This idea is common in many different areas of application, with many different names. In this section and the next we give a discussion of latent variables and related methods. Note that in the remainder of the report we show that this method was not useful for the first algorithm for differentiating between customers with a few factors; however, we also discuss how it could be useful in improving the second algorithm.

The underlying task is to model a set of n continuous variables $T = (t_1, \ldots, t_n)$ that have some joint probability density $f(t; \mu, \Sigma)$ where μ and Σ are the mean and covariance of the underlying distribution. The prime here denotes the transpose. If for example we assume that the components of T satisfy a joint Gaussian process then

$$f(t; \mu, \Sigma) = (2\pi)^{-n/2} |\Sigma|^{-1/2} \exp\left[-\frac{1}{2}(t-\mu)' \Sigma^{-1} (t-\mu) \right]. \tag{7.1}$$

Since Σ is an $n \times n$ symmetric matrix and μ is an n component column vector there are

$$n + \sum_{j=1}^{n} j = \frac{1}{2} n(n+3) \tag{7.2}$$

free parameters in this model. If we denote $\{\tau_j\}_{j=1}^{m}$ as m observations (columns) of the n variables (rows) then by maximizing the logarithm of the likelihood function corresponding to (7.1) one obtains the usual maximum likelihood estimates

$$\hat{\mu}_{\text{mle}} = \frac{1}{m} \sum_{j=1}^{m} \tau_j, \qquad \hat{\Sigma}_{\text{mle}} = \frac{1}{m} \sum_{j=1}^{m} (\tau_j - \hat{\mu}_{\text{mle}})(\tau_j - \hat{\mu}_{\text{mle}})'.$$

Note that the maximum likelihood estimator $\hat{\Sigma}_{\text{mle}}$ is a biased estimator of the population covariance matrix.

As n, the number of factors one attempts to model increases, expression (7.2) implies that the number of free parameters grows as n^2. To reduce the number of free parameters one could simply assume that Σ is diagonal. However, this is a very drastic assumption since it is equivalent to assuming that the variables being modelled are independent. On the contrary, it is known from the data mine that some variables are very strongly correlated. One possible way of reducing the number of free parameters while still preserving the main correlations between the various factors is to choose a set of $k < n$ hidden or *latent* variables $x = \{x_1, \ldots, x_k\}$.

For a given latent variable model one specifies a density function $g(x)$ for x and some map from the latent variables into the random variables t as

$$t = y(x, \omega) + \epsilon$$

where ω are the weights that generate t and ϵ is some random variable with zero mean that is independent of x. Typically $h(\epsilon)$, the probability density of ϵ, and $g(x)$ are specified à priori. Knowing these distributions, the density of T is computed by conditioning on the latent variables so that if T is a continuous random variable,

$$u(t) = \int f(t|x)g(x)\,dx. \tag{7.3}$$

In summary, a given latent variable method is determined by specifying $g(x)$, $h(\epsilon)$, the map $y(x, \omega)$ and computing $u(t)$ with (7.3) or its generalization depending on the probability measure involved.

One example of a latent variable method is factor analysis where one specifies that y is a linear map

$$t = y(x, \omega) = \Omega x + \mu + \epsilon$$

from $\mathbb{R}^k$ to $\mathbb{R}^n$. The μ and Ω are parameters, and x, ϵ are assumed to be independent normal random variables with zero mean. For x one assumes unit covariance while for ϵ one assumes that the covariance is a diagonal matrix so that $x \sim N(0, I)$, and $\epsilon \sim N(0, \Gamma)$ where Γ is some diagonal matrix. From the structure of the map one can determine that $T \sim N(\mu, \Gamma + \Omega\Omega')$. As in the case without latent variables, one may estimate μ, Ω and Γ using a maximum likelihood estimate however even in the case of this linear model there is not a closed form for the estimates and they are typically found through an iterative process.

This linear model illustrates the point of using latent variables. In particular for factor analysis, the symmetry of $\Gamma + \Omega\Omega'$ reduces the original kn free parameters of Ω to [2]

$$(n+1)(k+1) - \frac{1}{2}k(k+1)$$

which only grows linearly with n. This is accomplished while still preserving some of the underlying correlation structure. The trade-off is the increase in complexity when faced with the determination of the latent variables x. We now turn to another method closely related to the projection onto latent spaces[7] (PLS). Namely principal component analysis.

[7] In much of the statistical literature, the latent space methods are known as partial least squares methods. Fortunately this yields the same PLS mnemonic.

7.3 Principal Component Analysis

Principal component analysis (PCA) is the particular latent variable method where the k principal components are the leading eigenvectors of the sample covariance matrix

$$\hat{\Sigma} = \frac{1}{m-1}\sum_{j=1}^{m}(\tau_j - \hat{\mu})(\tau_j - \hat{\mu})'.$$

Rather than using the covariance matrix, an alternative choice (not used here) is to base PCA on the correlation (basically standardized covariance) matrix. In either case, PCA can be viewed as a transformation that diagonalizes $\hat{\Sigma}$ thereby reducing correlations between various combinations of factors while simultaneously finding directions in which the variance is a maximum.

Another way to view PCA is in the mean squared error sense [5]. With this viewpoint, the objective is to find a set of k orthonormal basis vector that span a k dimensional subspace such that the mean squared error between x and its projection onto the subspace is a minimum. As before the $n \times m$ matrix T corresponds to m observations of n random variables. If one denotes the orthonormal basis set as $\{\xi_j\}_{j=1}^{k}$ and the projection of T onto this set as T_ξ then the expected value of the mean square error is

$$\begin{aligned} E\left(\|T - T_\xi\|^2\right) &= E\left(\|T - \sum_{j=1}^{k}(\xi_j' T)\xi_j\|^2\right) \\ &= E\left(\|T\|^2\right) - E\left(\sum_{j=1}^{k}(\xi_j' T)^2\right) \\ &= (m-1)\left(\text{Tr}(\Sigma) - \sum_{j=1}^{k}\xi_j'\Sigma\xi_j\right) \end{aligned} \tag{7.4}$$

where we have assumed $E(T) = 0$ and $\Sigma = TT'/(m-1)$ is the covariance of T. By the spectral theorem every symmetric matrix $\Sigma = \Sigma'$ has a factorization $\Sigma = VDV'$ with D real diagonal and V an orthogonal matrix [8]. Consequently, Σ has n eigenvectors that can be chosen to be orthonormal and moreover, all of the corresponding eigenvalues $\{\lambda_j\}$ are real. From the right hand side of expression (7.4) one can see that the mean squared error is minimized by choosing $\{\xi_j\}$ to be any set of k orthonormal vectors [3].

Representation (7.4) also illustrates the particular advantage of choosing the first k eigenvectors of Σ. In this case the residual of the mean squared error is the sum of the absolute values of the remaining $n - k$ eigenvalues. Because of the relationship, a natural method of choosing the number of latent variables is to fix some acceptable level of error $\delta > 0$ and then choose k so that

$$E\left(\|T - T_\xi\|^2\right) = \sum_{j=k+1}^{n}|\lambda_j| < \delta.$$

It should be emphasized that these results only hold if the error is computed in the mean squared sense.

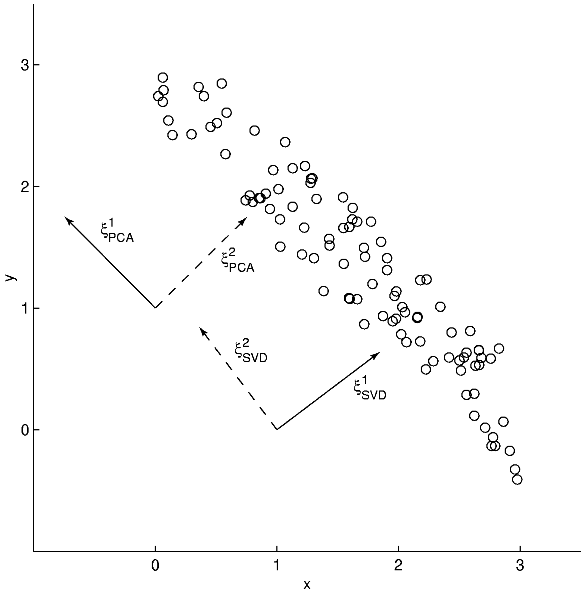


Figure 7.1: Illustrated are the components of a two factor PCA and SVD analysis for a randomly generated set of $m = 100$ points $(x, 3 - x + y)$ where x is uniformly distributed on the interval $[0, 3]$ and y uniformly distributed on $[-0.5, 0.5]$. Since $E(x) = E(y) = 3/2 \neq 0$, the PCA and SVD analysis yield a different principal direction. For this simulation, $\xi^1_{\text{PCA}} = \langle -0.7043, 0.7099\rangle$ and $\xi^1_{\text{SVD}} = \langle 0.7978, 0.6029\rangle$. The other complimentary components are $\xi^2_{\text{PCA}} = \langle 0.7099, 0.7043\rangle$ and $\xi^2_{\text{SVD}} = \langle -0.6029, 0.7978\rangle$.

7.3.1 Contrasting PCA and SVD

The above material shows that PCA corresponds to choosing the k dominant eigenvectors of the covariance matrix Σ. Provided one has $E(T) = 0$ this corresponds to finding the singular value decomposition (SVD) of T. To illuminate the connection, let $T = LSR'$ be the SVD of T where L and R are unitary matrices with columns that span $\mathbb{R}^n$ and $\mathbb{R}^m$ respectively. From the decomposition of T and the fact that $E(T) = 0$, one has $\Sigma = TT'/(m-1) = LSR'RS'L'/(m-1) = LSS'L'/(m-1)$. This should be compared to the eigenvector expansion of $\Sigma = VDV'$ where V is the matrix of orthonormal eigenvectors of Σ and D is the corresponding diagonal matrix of eigenvalues. As a result, one can identify the eigenspace V of Σ with the left singular space L of T. In addition, the matrix of eigenvalues $D = \text{diag}(\lambda_1, \ldots, \lambda_n)$ corresponds to the matrix $SS'/(m-1) = \text{diag}(\sigma_1^2, \ldots, \sigma_n^2)/(m-1)$.

If one does not ensure that $E(T) = 0$ then at least the first component of PCA and SVD indicate different primary directions. Figure 7.1 illustrates this behaviour where $T = \langle x, 3 - x + y\rangle$ with x uniformly distributed on the interval $[0, 3]$ and y uniformly distributed on $[-1/2, 1/2]$. In this case the first component of the PCA points in the direction corresponding to the maximum variance of the data cluster, $\langle -1, 1\rangle/\sqrt{2}$, whereas the first component of the SVD points in the direction of the centroid of the cluster, $\langle 1, 1\rangle/\sqrt{2}$.

7.4 Difficulties with Latent Variables

There are two disadvantages of these initial models when one considers them with respect to the data mine. Firstly, these models typically indicate that while there may be only a few principal directions, these directions may have significant weight in many of their components. This corresponds to the situation where one should ask large numbers of questions to determine which cluster to assign to a given individual. In essence, this analysis does not provide a natural way to determine which of the items is the *best* indicator (or the *best few* indicators).

Secondly, one must deal with the diverse collection of data in the data mine. Responses range from binary information about the ethnicity of an individual to continuous data regarding the market value of their dwelling. These two problems suggest that a robust algorithm is needed to gain a foothold on the structure of the mine before a more sophisticated latent variable analysis is attempted.

7.5 Determining the Best Question(s)

7.5.1 Factor Analysis as a First Look at the Mine

Factor analysis is the means by which we find the covariance relationship among many variables in terms of a few unobservable (or latent) variables. For example, if someone owns a Porsche, we would suspect that the person also has a high-paying job, lives in an upper class neighbourhood, has a six-figure stock portfolio and dines at high-class restaurants on a regular basis. If we were to label a latent variable that encompasses these four variables, we could label it *quality of life*.

In conducting a factor analysis, the basic model as discussed in Section 7.2 is:

$$t = \Omega x + \mu + \epsilon. \tag{7.5}$$

t is the observed random vector at n levels with a corresponding mean vector μ so that μ_i is the expected value of t_i. The vector x consists of the common factors at $k < n$ levels and the $n \times k$ matrix Ω is a matrix of coefficients otherwise known as the factor loadings. The element Ω_{ij} is referred to as the loading of the i-th variable on the j-th factor. In the above example, $n = 4$ and $k = 1$.

The model (7.5) can be rewritten as

$$t - \mu = \Omega x + \epsilon \tag{7.6}$$

and in order to have an orthogonal factor model, the following assumptions are made:

- x and ϵ are independent so that Cov$(x, \epsilon) = 0$,
- $E(x) = 0$ and Cov$(x) = I$,
- $E(\epsilon) = 0$ and Cov$(\epsilon) = \Gamma$ where $\Gamma =$ diag$(\gamma_1, \dots, \gamma_n)$.

Based on these assumptions, equation (7.6) yields an expression for the covariance of t,

$$\text{Cov}(t) = \Omega\Omega' + \Gamma.$$

In particular for the variable t_i one has, $\sigma_{ii}^2 = l_{i1}^2 + \cdots + l_{ik}^2 + \gamma_i$ where $l_{ij} = \mathrm{Cov}(t_i, x_j)$. The sum of l_{i1}^2 through l_{ik}^2 is called the i-th commonality while γ_i is the unique variance.

From here we can determine which common factors contribute the most to the total variability in t_i. The ultimate objective is to be able to group the factor loadings for any one factor and attach some type of label to them as we did with the Porsche example. In particular, we are interested in the loadings which carry a significant amount of the weight.

There are two main methods for estimating the factor loadings: principal component and maximum likelihood. The former uses the eigenvalue/eigenvector pairs of the sample correlation matrix in order to construct Ω. If x and ϵ can be assumed to be normally distributed, then maximum likelihood methods can be used to estimate the covariance matrix of t and thus $\Omega\Omega' + \Gamma$.

In addition, if the initial factor loadings cannot be easily interpreted, various factor rotation methods exist to aid interpretation. The most common method used is the Varimax method which seeks to spread out the squares of the loadings on each factor as much as possible so that the factor loadings can be grouped more easily. It should be noted that in recent years, Bayesian factor analysis has arisen [6]. One of the features of the Bayesian approach is the elimination of the need to rotate factors. Bayesian factor analysis was not attempted in the analysis of the data mine.

7.5.2 Ranked Differences of Means

To deal with the eclectic data in the mine, the most direct method of determining which questions seem to reflect the choice of an individuals product preference is to compute the observed difference in means across the given factors. Moreover, this is easily accomplished when there is a simple choice between two products as in the case studies that follow. The idea can also be generalized to the case when there are multiple products.

If only one product is under consideration the mine Ω is split into two groups, those respondents that have the product and those respondents that do not have the product. Denote these two groups as Ω_1 and Ω_2 consisting of m_1 and $m_2 = m - m_1$ rows (respondents). For each of the n question responses, one computes the test statistic

$$z_j = \frac{\overline{x}_{1j} - \overline{x}_{2j}}{\sqrt{s_{1j}^2/m_1 + s_{2j}^2/m_2}}, \qquad j = 1, \ldots, n.$$

Using this statistic to classify the data is based on the need to find factors which are important to all of the respondents, yet at the same time differentiate the two groups. In terms of the solution to the normal equations, those factors with a large value of z_j correspond to coefficients which are large for both groups, but the *peaks* in the graphs of the coefficients occur at significantly different locations. It is precisely this failure to differentiate between the two groups that demonstrates why latent variable methods do not work easily as a first step.

Due to the large number of samples ($m \gg 50$), the z_j are each approximately normally distributed with zero mean and variance one under the null hypothesis that there is no difference in means between the two groups. Ordering the test statistics from the most negative to the most positive induces a reordering of the questions. In this sense, one can rank the indicators as to their ability to differentiate the two populations with respect to a given product. The factor with the largest observed values of z_j define the starting points of the cluster analysis and because of this, these particular questions form the first steps into the data mine when Ω is viewed in the light of a given product.

Typically the number of questions, n, can be large, and one is likely to find some means which will appear significantly different even when no difference in means exists. To analyse this situation let $\mathcal{U}$ be the number of the of questions with a test statistic that lies in the interval $(-s, s)$. If we assume for simplicity that the questions are independent then $\mathcal{U} \sim \text{Bin}(n, p)$ where $p = 2(1 - \Phi(s))$ and $\Phi(s)$ is the normal cdf. Therefore the probability that $\mathcal{U} \geq u$ and the expected value of $\mathcal{U}$ are

$$P(\mathcal{U} \geq u) = 1 - \sum_{j=0}^{u-1} \binom{n}{j} p^j (1-p)^{n-j}, \qquad E(\mathcal{U}) = np.$$

The case studies at the end of this report use $n = 53$ data factors compiled from census data. For this many factors and $s = 3$ standard deviations, one finds that $P(\mathcal{U} \geq 1) = 0.1289$ and $E(\mathcal{U}) = 0.1378$. Consequently, to eliminate any *false alarm* differences we have chosen to consider statistically significantly differences at three rather than the common two standard deviations from the mean under the null hypothesis.

7.6 Consumer Based Clustering

A simple definition of classification or clustering is using a metric or a set of rules which groups the data, and is also used to classify future data. For example, medical diseases may be classified by the manifesting symptoms which in turn describe each class or subclass of a given disease. In data classification one develops a description or model for each class in a database, based on the features present in a set of class-labelled training data. There have been many data classification methods studied, including decision-tree methods, such as C4.5 algorithm, ID3 algorithm, and SLIQ algorithm, statistical methods, neural networks, rough sets, nearest neighbour method, database-oriented methods, parallel algorithms, etc. The method for classification is in general application dependent, based on the goal of mining the data.

In this paper we have chosen a relatively simple metric to determine the clustering of the data, in particular, correlations between data columns corresponding to the different questions. The choice of the metric is based on the underlying goal that the salesperson has the opportunity to learn about a customer's preferences by asking only a few questions. This metric of clustering then indicates which are the most informative data that one would like to infer from these few questions. This metric is most similar to a nearest-neighbour type rule, where two of the census data are *near* when they are strongly positively correlated.

Another reason for looking at this metric is that it is computationally efficient. In order to look for more complicated classification structures, one could consider classification-rule learning which requires finding rules or decision trees that partition the given data into predefined classes. Of course, there many possible such decision trees; for any realistic problem domain of the classification-rule learning, the set of possible decision trees is too large to be searched exhaustively. In fact, the computational complexity of finding an optimal classification decision tree is NP hard.

Therefore, we have not attempted to find an optimal decision tree; rather, we have shown that the correlations give a fast classification of the mine, which can be readily used in designing questions and conversations with customers.

7.7 Case Study A: BMW/Honda

The first case study considered BMW and Honda owners. Given the census data on BMW and Honda owners grouped by postal code the goal is twofold:

1. Select a *few* questions to ask prospective buyers to infer their BMW/Honda preference.

2. Based upon the indicated preference, infer other information about the consumer.

For the following analysis there are a total of $m = 1995$ respondents which are partitioned into $m_h = 1782$ Honda owners and $m_b = 213$ BMW owners. Corresponding to each of these groups are $n = 53$ census data factors. Those portions of the data mine corresponding to Honda and BMW owners are referred to as Ω_h and Ω_b respectively. As a starting point, we compute a PCA on Ω_h and Ω_b.

7.7.1 PCA: BMW/Honda

The eigenvalue structure of Σ_h and Σ_b, the covariance matrices of Ω_h and Ω_b, are virtually identical, ranging from $\lambda_1 \simeq 2.6 \times 10^{10}$ to $\lambda_{53} \simeq 6.7 \times 10^{-12}$. Figure 7.2(a) illustrates the logarithm of the magnitude of the $\{\lambda_j\}$. What is apparent from the illustration is that λ_1-λ_4 account for much of the variation in the mine. In fact

$$\frac{\sum_{j=1}^{4} \lambda_j}{\sum_{j=1}^{53} \lambda_j} = 0.9996.$$

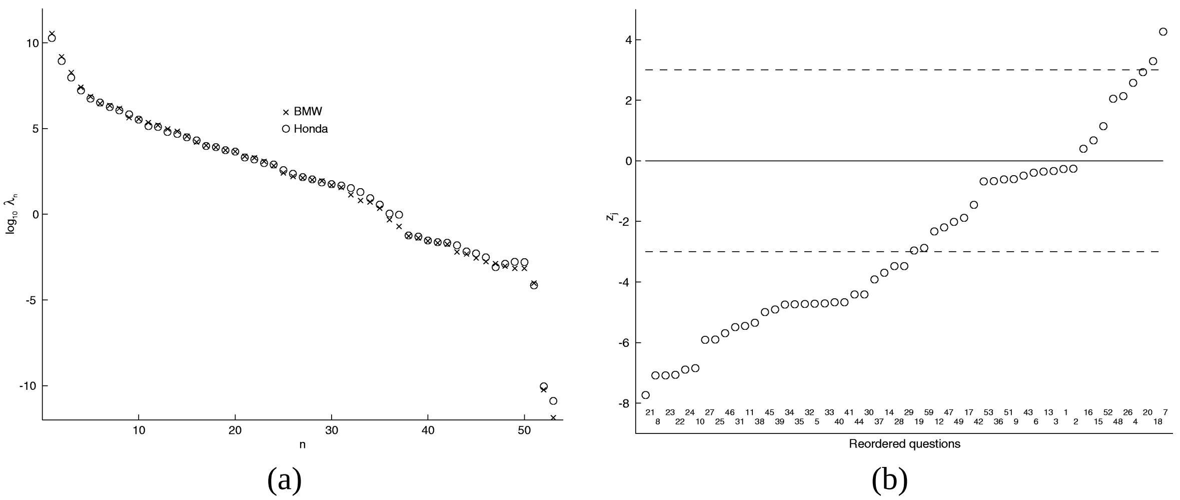
Corresponding to these eigenvalues are eigenvectors focused in the direction of factors 21 to 24. These questions correspond to the average home value, average family income, average household income and total household expenditure. At the other end of the spectrum, λ_{52} and λ_{53} have eigenvectors that identify a strong correlation between factors 32 to 35. These latter indicators correspond to the average amount spent on public transportation, average spent on streetcars and buses, average spent on taxis and average spent on airplanes.

This preliminary analysis indicates that questions that reflect the total income and expenditure of a particular household should be good indicators of whether or not an individual owns a BMW or Honda. In addition, there is a certain amount of redundant information in the data mine with respect to public transportation. The main difficulty with PCA remains in that it does not identify a single question that best identifies BMW owners over Honda owners. Some headway can be made by computing a factor analysis which is attempted next.

7.7.2 A Preliminary Factor Analysis

In order to conduct a factor analysis, the data was again split into Ω_b and Ω_h according to whether the vehicle owned by the respondent was a BMW or Honda. Both principal component and maximum likelihood methods were used with three factors. However, the principal component method accounted for more of the variability than the maximum likelihood method. Varimax rotation was used in both methods. Table 7.1 summarizes the analysis.

As can be seen from the results, the same variables contributed to whether a person would own a BMW or a Honda with some variations. For example, under Factor 3, average home value contributes more to a person being a BMW owner than a Honda owner. The percentage of variability in vehicle



(a) (b)

Figure 7.2: (a) Depicted are the eigenvalues for Σ_h and Σ_b, the covariance matrices for the Ω_h and Ω_b subsets respectively. The similar spectral structure for the BMW and Honda covariance typifies the difficulty encountered when attempting to find differences between these two groups. (b) Displayed is the ranked test statistic for the difference of means for each of the 53 factors. The dashed lines indicate the level of three standard deviations and the reordering of the factors is indicated at the base of the plot.

ownership explained by the complete model is approximately the same for both: 60.7% for BMW and 59.3% for Honda. Further analysis tools, such as discriminant analysis or tree regression can be used to determine which of these variables distinguish between BMW and Honda owners. The main conclusion is that the principal factors are strongly positively correlated and the anti-correlated components are small.

The factor analysis identifies a block of questions that differentiates the two groups. Further identification is possible by considering the difference of means across the 53 factors.

7.7.3 Difference of Means: BMW/Honda

Figure 7.2(b) shows the ordered test statistics for each of the $n = 53$ factors. Detailed explanations for all of the census data can be found in the appendix at the end of this report. This ordering induces a reordering of the factors to $\{21, 8, 23, 22, 24, 10, \ldots, 18, 7\}$, with factor 21 having the most negative and question 7 having the most positive test statistic. This analysis also indicates that any questions related to 21, 8, 23, 22, 24, 10 are equally efficient at identifying BMW owners while factor 7 can be used to identify Honda owners. Factors 21-24 correspond to average home value, average annual family income, average annual household income, and annual household total expenditure, 8 reflects the percentage of individuals in a dwelling with a university education, 10 indicates self employment, 7 indicates the percentage of those subjects in a dwelling with only up to grade nine education and question 18 identifies those individuals living in dwellings with more than five stories. These initially identified factors can now used as starting points for a cluster analysis. Notice that many of these data items appear in the preliminary factor analysis.

Being able to identify a particular individual as a BMW or Honda owner is an important factor

Factor 1	Loading values	
	BMW	Honda
Total adult population	0.996	0.997
Total population	0.995	0.997
Total number of households	0.995	0.991
Total adult labour force	0.994	0.995
Total number of families	0.993	0.996
Total number of dwelling units	0.993	0.989

Factor 3	Loading values	
	BMW	Honda
Ave. home value	0.749	0.579
% self-empl. inc.	0.735	0.580
% univ. degree	0.694	0.714

Factor 2	Loading values	
	BMW	Honda
% homeowners	0.922	0.939
% single-detached house	0.861	0.772
Average owners' major payments	0.794	0.859
% home renters	-0.924	-0.928
% apartment with ≥ 5 floors	-0.746	-0.806
Average gross rent	-0.698	-0.759

Factor	Variability explained	
	BMW	Honda
Factor 1	28.8%	28.5%
Factor 2	20.7%	22.3%
Factor 3	11.2%	8.5%
Total	60.7%	59.3%

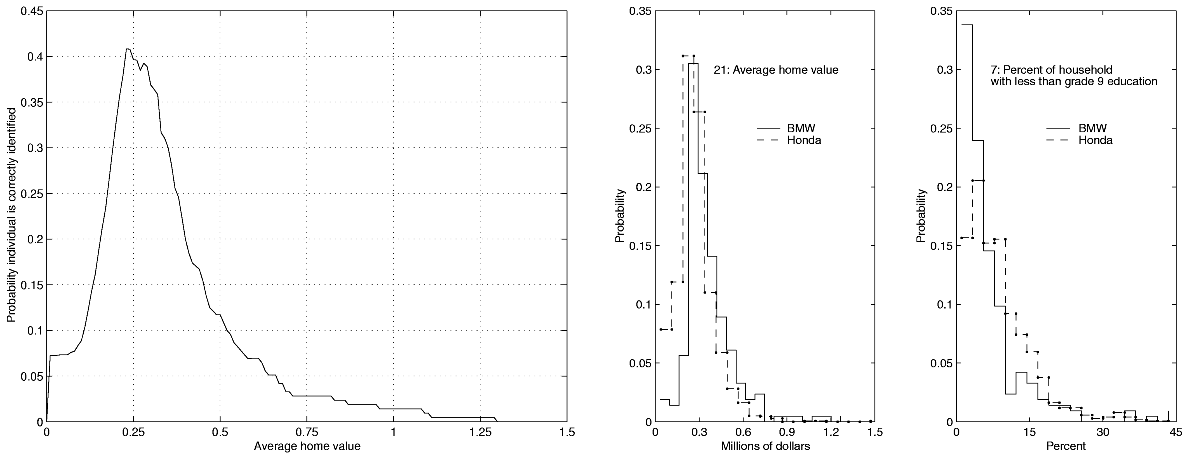
Table 7.1: Listed are the three factors identified in the BMW/Honda data sample and the corresponding loadings. The final table shows that these three factors account for approximately 60% of the observed variability.

for the cluster analysis that follows. To differentiate we choose question 21, the average home value. We can use the data mine to determine the particular house value that should be used as a cutoff value to correctly identify the maximum number of individuals. That is, determine x such that $P(H < x \text{ and } B > x)$ is a maximum where H and B are the responses to question 21 for the Honda and BMW owners. Figure 7.3(a) shows that this probability has a maximum of 0.41 for x chosen in the interval ($230K, $240K). This procedure of choosing an optimal cutoff value from the probability structure encoded in the mine can be repeated for other questions to increase the differentiating power.

Detected differences in the mean response can be quite subtle. As an illustration of this, Figure 7.3(b) contrasts the probability distributions of the response to question 21 and question 7 for the two groups. For the cluster analysis that follows, the first six, 21, 8, 23, 22, 24, 10, and the last six factors, 48, 26, 4, 20, 18, 7, are used to define the initial clusters. By doing this it is hoped that the cluster analysis will be able to identify sequences of questions that link the Honda group to the BMW group. This in turn may help identify characteristics of prospective BMW owners.

7.7.4 Cluster Analysis: BMW/Honda

A cluster analysis was performed for two cases with correlations at the 60% and 75% level indicated. The first analysis was performed by considering only the BMW owners while the second analysis considered the complete data mine. As the number of Honda owners was much larger than the number of BMW owners, a cluster analysis of only Honda owners matches that obtained when using the complete data mine. Figure 7.4 summarizes the results.



(a) (b)

Figure 7.3: (a) Probability of correctly identifying Honda and BMW simultaneously for a given known home value. This distribution has an extreme value of 0.41 for the interval ($230K, $240K). At this cutoff value the probability of correctly identifying a Honda owner is 999/1782 (56%) and that of identifying the BMW owner is 156/213 (73%). (b) On the left is the probability distribution of responses to question 21 (average home value). To the right is the probability distribution to factor 7 (percentage of household with less than a grade nine education). These factors yield the most negative and most positive values of z_j respectively.

What is immediately apparent is the greater resolution one can achieve in the data mine with the BMW group. On the left hand side of each cluster diagram are those questions identified with the most negative test statistic ($+$ BMW) and on the right are those questions corresponding to the most positive test statistic ($-$ BMW). Those traits that identify BMW owners are household value and income, university education and self employment. Characteristics that directly stem from these traits are donations to charity, amount spent on public transportation and amount spent on personal care. From the other end of the data mine, individuals that do not own a BMW are characterized as either renters or having less than a grade nine education. A link between the renters and those with expensive homes is the number of cars per household and the subsequent expenditure on tires, gasoline, food and transportation.

This cluster analysis implies that there are many possible ways to identify a BMW owner. For example, university educated individuals that do not rent and outwardly appear to spend a great deal on personal care. Once identified, the characteristics of this group could be targeted for a broad range of products or services that lie within the identified common interests. Examples of these interests include cosmetics, expensive tires, and perhaps even endowments to universities.

When we consider the entire mine there is a loss of resolution but much of the structure remains. In addition, other characteristics come to the forefront. Two new characteristics are a stronger correlation with the amount spent on computers the loss of the correlation with public transportation. A possible implication here is that Honda owners with an expensive home may be differentiated from BMW owners by the amount that they spend on airlines. Again we point out that being able to accurately classify an individual is an important first step in that the clustering reflects this bias. However, mis-identifying an individual does not have as serious a consequence as one might first expect. The clustering analysis

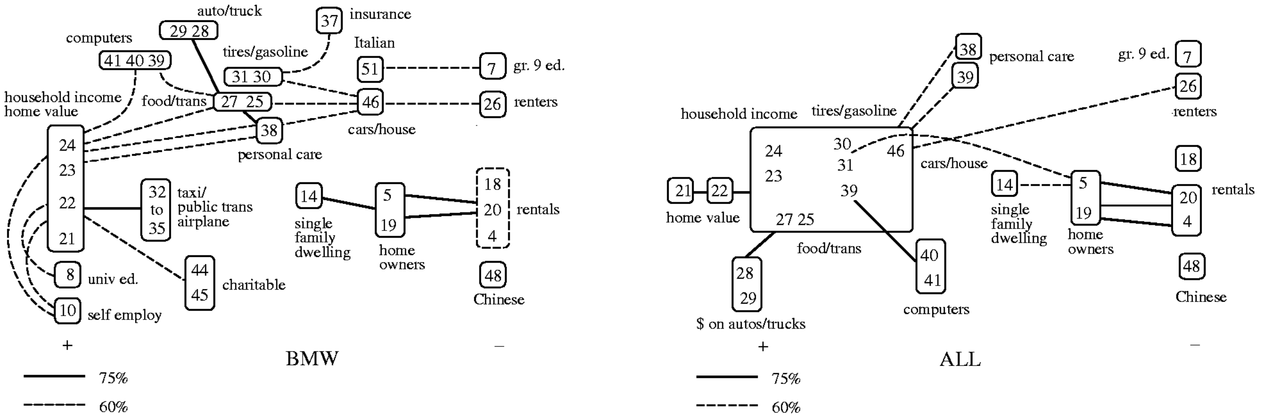


Figure 7.4: To the left are the data clusters considering only the BMW group while to the right is the the same analysis considering the complete data mine. On the + side of each figure, the factors are larger or more likely for BMW owners, while the − side of each figure the factors are smaller or less likely for BMW ownership. Cutoffs at 60% and 75% in the correlation level (either positively or negatively) are indicated. Explanations for all of the data factors can be found at the end of the report.

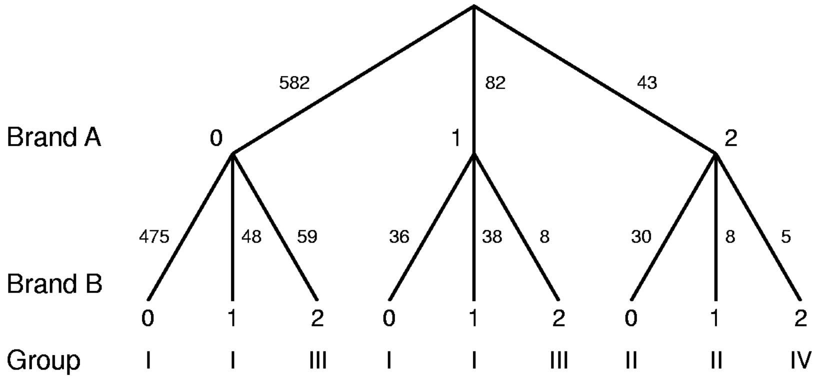


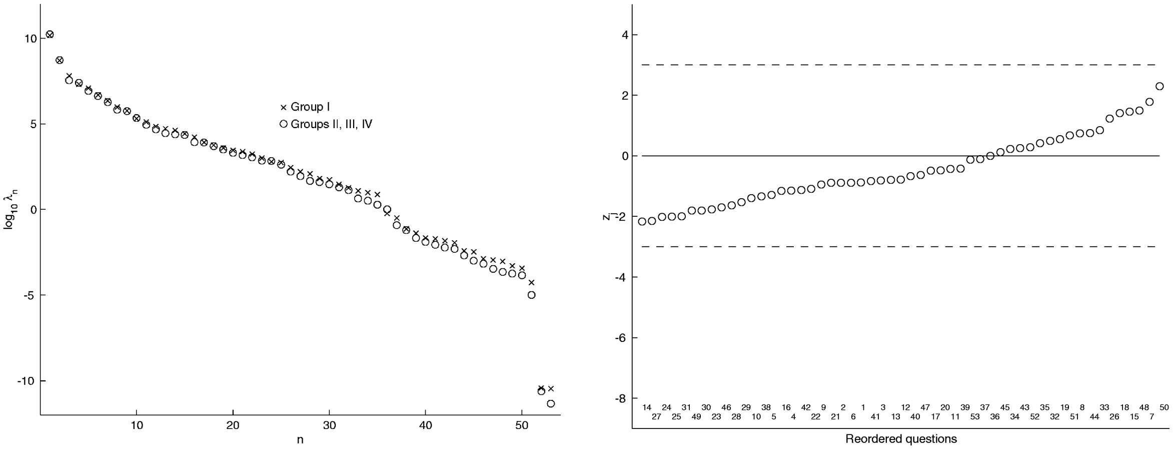
Figure 7.5: The classification tree structure for the beer preference respondents.

supports this by illustrating that much of the structure is preserved when moving from BMW to Honda owners. To contrast with the BMW/Honda data, the second case study considers consumer preference of two brands of domestic beer.

7.8 Case Study B: Beer Preference

Our second case study addresses beer preferences amongst a sample of 707 individuals. Each individual was asked to indicate their preference for two different brands of beer (Brand A and Brand B) according to the four point scale:

0: Don’t drink



(a) (b)

Figure 7.6: (a) Eigenvalues for Σ_b and Σ_1, the covariance matrices for the Ω_b and Ω_1 subsets respectively. (b) Ranked test statistic for the difference of means for each of the 53 factors from the census data . As in Figure 7.2(b), the dashed lines indicate the level of three standard deviations and the reordering of the factors is indicated at the base of the plot.

1: Tried in the past 12 months

2: Becoming usual

3: Usual brand.

As no respondents indicated that either brand was their *usual* brand, the responses broke into nine separate classifications. Figure 7.5 shows the resulting tree structure and the four groups into which the individuals were placed. Group I essentially consists of non-drinkers, group II and III tend to prefer brands A and B respectively, and group IV respondents strongly prefer both brands.

7.8.1 Difference of Means: Beer

No significant differences were detected in a direct comparison of groups II and III since all of the z_j statistics were located within two standard deviations. Large scores were detected when comparing groups I and IV but since group IV consisted of only five individuals our underlying assumptions of normality were no longer valid. Since the 53 characteristics do not seem to be able to clearly differentiate the two brands of beer, it was more appropriate with this data set to compare drinkers of both brands versus those individuals that do not drink either brand. As such, groups II, III and IV were consolidated into a single group which was then compared to group I. The portions of the data mine concerning these two groups will be referred to as Ω_b and Ω_1. Figure 7.6 illustrates the spectral structure of these two classifications and distribution of the difference of means.

Comparing Figure 7.6(a) with Figure 7.2(a) illustrates a striking similarity with the eigenvalue distribution of the beer data and the car data of the previous study. This similarity is also reflected in the preliminary factor analysis which has been omitted because of its similarity to the BMW/Honda analysis. Even though the eigenvalue structure was similar, none of the test statistics lie outside of the

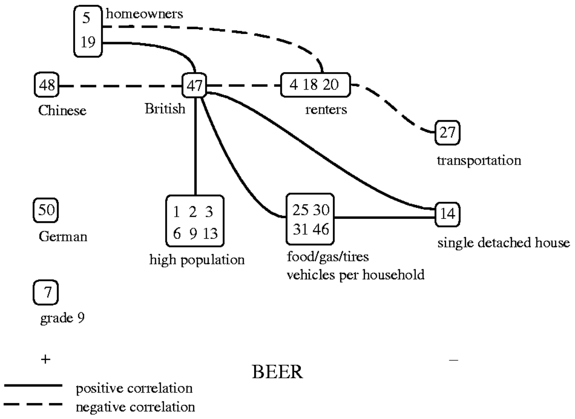


Figure 7.7: Illustrated is the cluster analysis for beer drinkers. The correlation cutoff was set at 50%. Solid lines represent positive correlations and dashed lines represent negative correlations. On the left of the are those indicators whose mean response for beer drinkers was higher than for non beer drinkers.

three standard deviations. Despite this, we begin the cluster analysis starting with factors 50, 7, 48 on the *drink beer* side of the data mine and factors 27 and 14 on the *don't drink beer* side of the mine.

7.8.2 Cluster Analysis: Beer Preference

We again remind the reader that a full explanation of each of the factors from the census data can be found in the appendix. Correlations between the starting factors 7, 14, 27, 48, 50 and the remaining questions were detected once the cutoff level was dropped to 50%. This reduction in the cutoff level was expected given the lack of significant differences detected in the previous section. Figure 7.7 summarizes the results and illustrates that respondents that prefer these brands seem to fall along ethnic lines. The analysis also indicates that beer drinkers are characterized by individuals that rent rather than living in a single detached house. However, with this collection of 53 factors there was no additional product information that could be correlated with these individuals.

Without a clear indication of questions that differentiate between beer drinkers and non beer drinkers, at least for these two brands of beers, we do not attempt to correlate other characteristics. Clearly, some additional analysis is necessary to improve the first part of the analysis, namely, making distinctions according to a particular product preference. Since the difference in means is not significantly large for any one factor, this suggests that a combination of questions would be necessary to make a significant distinction. As mentioned above, a factor analysis has not been done for this data set; however, one could use this analysis to design a combination of a few questions as a first algorithm for differentiating between consumers.

7.9 Conclusion

For a given product two tasks were required. The first being identification of consumers that would react favourably to product and the second being the inference of other characteristics concerning these consumers.

This was accomplished with a twofold strategy. By ranking the difference of means across all of the factors, those questions that best characterize favourable consumers can be identified. Once identified, the data mine can be used to estimate the power of a given strategy to correctly identify a given individual. For case study A the identification algorithm was simply to use an individuals home value. By optimizing the cutoff level this single variable was able to correctly identify 41% of the individuals in the data mine. By using a combination of questions this percentage could be increased. Performing a cluster analysis that is rooted at these key identifying questions allows other characteristics of these consumers to be inferred.

The two case studies show that being able to identify questions that significantly differentiate respondents with respect to a given product is a fundamental part of the process. Failure to make this identification decreases the resolution of the subsequent cluster analysis. Case study A exemplifies the situation when there is a clear separation with respect to a product whereas case study B illustrates the decrease in resolution when no clear separation exists. In general, this first step may be dependent on the type of data and the desired differentiations. A combination of factor analysis and the consideration of differences in basic test statistics proved to be superior to methods based on latent variables or principal components, due to the underlying eigenstructure of the data mine.

To increase the capability of this method future advances should include a more sophisticated clustering algorithm. For example, PLS/SVD could be used on the clustering subgroups after the first step of separating with the difference of the means statistic. An addition, automatic determination of the identification power for a given set of identifying questions should also be addressed.

Appendix: Factors from Census Data

Question	Description	Mnemonic
01	Total population	PP-TOT
02	Total number of families	FM-TOT
Household		
03	Total number of households	HH-TOT
04	Average gross rent	HH-TOTRENT
05	Average owner’s major payments	HH-TOTMAPJ
Education		
06	Total population 15 years old and over	ED-HL
07	Percent education level: less than grade 9	ED-GR-9
08	Percent education level: university with bachelor’s degree or higher	ED-UNIDG
Employment		
09	Total labour force 15 years old and over	EM-TOT
10	Percent employment: self-employed (incorporated)	EM-PSMI
11	Percent employment: self-employed (unincorporated)	EM-PSMU
12	Percent employment: unpaid family workers	EM-UP
Dwelling		
13	Total number of dwelling units	DM-TOT
14	Percent: dwelling type: single-detached house	DW-SINGLE
15	Percent: dwelling: semi-detached house	DW-SEMI
16	Percent: dwelling type: town house	DW-ROW
17	Percent: dwelling type: apartment, detached duplex	DW-DUP
18	Percent: dwelling: apartment building, five or more storeys	DW-APT5
19	Percent: homeowners	DW-OWNED
20	Percent: home renters	DW-RENTED
21	Average home value	DW-TVALUE
Income		
22	Annual average family income	IN-AFM
23	Annual average household income	IN-AHH
Expenditures		
24	Annual household total expenditure	D1000-5230
25	Annual expenditure on food	D1000-1560
26	Annual expenditure on rent	D2000
27	Annual expenditure on transportation	D3000-3260
28	Annual expenditure on purchase of automobiles and trucks	D3000-3004
29	Annual expenditure on automobiles	D3000
30	Annual expenditure on gasoline and other fuels	D3050
31	Annual expenditure on tires, batteries, parts and supplies	D3060
32	Annual expenditure on bus, subway, street car and train	D3200
33	Annual expenditure on public transportation	D3200-3260
34	Annual expenditure on taxi	D3210

35	Annual expenditure on airplane	D3220
36	Annual expenditure on moving, storage and delivery services	D3260
37	Annual expenditure on accident and disability insurance	D3384
38	Annual expenditure on personal care	D3500-3580
39	Annual expenditure on recreation equipment and services	D3700-3830
40	Annual expenditure on computer hardware	D3750-3752
41	Annual expenditure on computer software	D3755
42	Annual expenditure on gifts of money and contributions	D5200-5230
43	Annual expenditure on gifts to persons living outside Canada	D5210
44	Annual expenditure on contributions to charity	D5220-5230
45	Annual expenditure on non-religious charitable organizations	D5230
46	Average number of vehicles owned per household	NMVEHONP
	Ethnicity	
47	Percent ethnicity : British	BRITISH
48	Percent ethnicity : Chinese	CHINESE
49	Percent ethnicity : Dutch	DUTCH
50	Percent ethnicity : German	GERMAN
51	Percent ethnicity : Italian	ITALIAN
52	Percent ethnicity : Polish	POLISH
53	Percent ethnicity : Scandinavian	SCANDINAV

Bibliography

[1] Aha, D.W. (1992). *Tolerating Noisy, Irrelevant, and Novel Attributes in Instance-Based Learning Algorithms*. International Journal of Man-Machine Studies, **36**, 267-287.

[2] Bishop, C.M. *Latent variable models* In Learning and Graphical Models, M.I. Jordan (Ed.). MIT Press, 1999, 371-403.

[3] Burden, R.L. & Faires, J.D. Numerical Analysis. PWF-Kent Publishing, 1989.

[4] Dasarathy, B.V. Nearest neighbor (NN) norms: NN pattern classification techniques. Washington: IEEE Computer Society Press, 1990.

[5] Hyvärinen, A., Karhunen J. & Oja, E. Independent component analysis. John Wiley, 2001.

[6] Ghahramani, Z. & Beal, M.J. *Variational inference for Bayesian mixture of factor analysers*. In Advances in Neural Information Processing Systems 12, S.A. Solla, T.K. Leen, & K. R. Müller (Eds). MIT Press, 2000.

[7] Joshi, K.P. (1997). Analysis of Data Mining Algorithms, `http://userpages.umbc.edu/~kjoshi1/data-mine/proj_rpt.htm`.

[8] Strang, G. Introduction to Linear Algebra. Wellesey-Cambridge Press, 1993.

[9] Wettschereck, D. *A Hybrid Nearest-Neighbor and Nearest-Hyperrectangle Algorithm*. In Lecture Notes in Artificial Intelligence 784. Springer-Verlag, 1994, 323-335.

Appendix A

List of Participants

Rita Aggarwala	University of Calgary	rita@math.ucalgary.ca
Katharina Baamann	Georgia Institute of Technology	kbaamann@math.gatech.edu
Charles Bergeron	Ecole polytechnique de Montreal	charles.bergeron@polymtl.ca
C. Sean Bohun	Pennsylvania State University	csb15@psu.edu
Christopher Bose	University of Victoria	cbose@math.uvic.ca
Mark Braverman	University of Toronto	mbraverm@cs.toronto.edu
Thalya N. Burden	University of Kentucky	tburden@ms.uky.edu
Shengyuan (Michael) Chen	University of British Columbia	mchen@coe.ubc.ca
Zhenlu Cui	Florida State University	zcui@math.fsu.edu
Xinghua Deng	University of Alberta	xdeng@math.ualberta.ca
Olivier Dubois	McGill University	dubois@math.mcgill.ca
Marcio F. Gameiro	Georgia Institute of Technology	gameiro@math.gatech.edu
Nadine Gärtner	Clemson University	ngaertn@clemson.edu
Hongbin Guo	University of Alberta	hguo@math.ualberta.ca
Ying Han	McGill University	ying@math.mcgill.ca
John R. Hoffman	Lockheed-Martin	
Huaxiong Huang	York University	hhuang@yorku.ca
Samet Y. Kadioglu	Florida State University	skadio@math.fsu.edu
Christian W. Ketelsen	Washington State University	cketelsen@wsu.edu
Edward Keyes	Orisar Inc.	
Ilona Kletskin	University of Toronto	kletskin@math.utoronto.ca
Mike Kouritzin	University of Alberta	mkouritz@math.ualberta.ca
Rachel Kuske	University of British Columbia	rachel@math.ubc.ca
Yasong Jin	University of Kansas	jinyasong@math.ukans.edu
Gerry LaBute	University of Calgary	labute@math.ucalgary.ca
Serguei Lapin	University of Houston	slapin@math.uh.edu
Seung Youn Lee	Ohio State University	seunglee@math.ohio-state.edu
Hua Li	University of Calgary	lih@math.ucalgary.ca
Mingfei Li	Michigan State University	limingfe@msu.edu
Qingguo Li	Simon Fraser University	qlib@sfu.ca
Alfonso L. Limon	Claremont Graduate University	alfonso.limon@cgu.edu
Rongsong Liu	York University	rsliu@mathstat.yorku.ca
Wei Lu	Manifold Data Mining	

Jack Macki	University of Alberta	jmacki@gpu.srv.ualberta.ca
Bruce McGee	McMillan-McGee Corp.	mcgee@mcmillan-mcgee.com
Veena B. Mendiratta	Lucent Technologies	veena@lucent.com
Yuriy Mileyko	New Jersey Institute of Technology	ym4@njit.edu
Anuj Mubayi	University of Texas, Arlington	anujmubayi@yahoo.com
Nilima Nigam	McGill University	nigam@math.mcgill.ca
Robert Piché	Tampere University of Technology, Finland	robert.piche@tut.fi
Juan Restrepo	University of Arizona	restrepo@math.arizona.edu
Fadil Santosa	IMA/University of Minnesota	santosa@ima.umn.edu
Kostyantyn Stepankevych	University of Calgary	
Andrew C. Taylor	University of Calgary	ataylor@math.ucalgary.ca
Carlos Tolmasky	Cargill Inc.	carlos_tomalsky@cargill.com
Tzvetalin S. Vassilev	University of Saskatchewan	tsv552@mail.usask.ca
Lalitha Venkataramanan	Schlumberger Doll Research	
Qian Wang	University of Alberta	qwang@math.ualberta.ca
Tony Ware	University of Calgary	ware@math.ucalgary.ca
Rex Westbrook	University of Calgary	westbroo@ucalgary.ca
Haris Widjaya	Simon Fraser University	htw@cs.sfu.ca
Yujun Wu	University of Kentucky	ywu@ms.uky.edu
Kerianne Yewchuk	University of Alberta	kyewchuk@ualberta.ca
Fabien M. Youbissi	Laval University	fyoubi@mat.ulaval.ca
Lin Zhou	New Jersey Institute of Technology	lz5@njit.edu

PIMS Contact Information

email: pims@pims.math.ca

http://www.pims.math.ca

Director’s Office
Pacific Institute for the Mathematical Sciences
Room 200, 1933 West Mall
University of British Columbia
Vancouver BC V6T 1Z2
Canada

Pacific Institute for the Mathematical Sciences