Signal Processing 89 (2009) 989–999

Contents lists available at ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Relaxed maximum a posteriori fault identification

Argyrios Zymnis *, Stephen Boyd, Dimitry Gorinevsky

Department of Electrical Engineering, Stanford University, Stanford, CA 94305, USA

ARTICLE INFO

Article history:
Received 20 May 2008
Received in revised form
30 September 2008
Accepted 28 November 2008
Available online 7 December 2008

Keywords:
Fault detection
Statistical estimation
Convex relaxation
Interior-point methods

ABSTRACT

We consider the problem of estimating a pattern of faults, represented as a binary vector, from a set of measurements. The measurements can be noise corrupted real values, or quantized versions of noise corrupted signals, including even 1-bit (sign) measurements. Maximum a posteriori probability (MAP) estimation of the fault pattern leads to a difficult combinatorial optimization problem, so we propose a variation in which an approximate maximum a posteriori probability estimate is found instead, by solving a convex relaxation of the original problem, followed by rounding and simple local optimization. Our method is extremely efficient, and scales to very large problems, involving thousands (or more) of possible faults and measurements. Using synthetic examples, we show that the method performs extremely well, both in identifying the true fault pattern, and in identifying an ambiguity group, i.e., a set of alternate fault patterns that explain the observed measurements almost as well as our estimate.

1. Introduction

In this paper we present a method for identifying a pattern of faults, given a set of noisy, possibly quantized measurements of a system. The goal is to estimate the fault pattern most likely to have occurred. An important secondary goal is to identify a (possibly empty) set of other fault patterns, called the *ambiguity group*, that explain the measurements almost as well as the most likely one.

If there are only 10 or so possible faults, it is possible to search over all possible fault patterns, taking the fault pattern with largest posterior probability as our estimate. If only one or (for small enough problems) two faults are likely to have occurred, the fault identification problem can be solved by evaluating the posterior probability of all possible single or double fault patterns, given the measurements. We are interested in cases where these simple methods are not practical, for example, when there are hundreds (or many more) of possible faults, and the number that may have occurred is more than one or two.

We are also interested in the case when the measurements are, possibly, quantized (in addition to being noise corrupted). As an extreme case, we have 1-bit quantization, in which a measurement tells us only that a noisy signal lies above or below a given threshold. We will see in examples that even crudely quantized measurements (say, 3- or 4-bit) can be surprisingly effective in estimating the fault pattern, using our method.

The method we describe here is based on forming a convex relaxation of the (maximum a posteriori probability) MAP problem, which we show can be solved very efficiently, even when the measurements are quantized. We then round these relaxed estimates, and possibly, perform simple local optimization, to obtain our final estimate of the true fault, as well as our estimate of the ambiguity group.

1.1. Prior and related work

Fault detection problems arise in most computer based engineering systems. Examples include aerospace (e.g., jet

* Corresponding author. Tel.: +16503870059.
E-mail addresses: azymnis@stanford.edu (A. Zymnis), boyd@stanford.edu (S. Boyd), gorin@stanford.edu (D. Gorinevsky).

0165-1684/$ - see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.sigpro.2008.11.014

engine health monitoring [1,2]), industrial process control [3], automotive [4], telecommunications and networking [5,6], computer systems [7], circuit fault identification [8], and many others.

In most papers in the literature it is assumed that faults affect measurements in an additive way. In fact there is a whole body of research devoted to optimal signature matrix design [9,10]. A number of publications are concerned with the problem of fault identification in linear dynamical systems for both parametric [2] and binary faults [11,12]. A number of heuristics have been proposed to tackle this problem, including variations of least-squares [2] and methods based on Kalman filtering [1,11]. For some general references on this kind of work see Refs. [13–15].

The problem of fault identification with binary measurements has been extensively studied by the computer science community. The main references in this work are [16,17], in which this problem is posed as a logical constraint satisfaction problem and a number of heuristics are proposed for solving it. More recent references on this type of problem are [18,19], in which the authors formulate this problem as a minimum set cover problem on a graph, which they approximately solve using a method based on Lagrangian relaxation.

The idea of using convex relaxation as the basis for a heuristic for solving a combinatorial problem is quite old. Some recent examples include compressed sensing [20], sparse regressor selection [21], sparse signal detection [22], and sparse decoding [23]. Other applications that use convex relaxations include portfolio optimization with transaction costs [24], controller design [25], circuit design [26], sensor selection [27], and mixed state estimation in linear dynamical systems [28].

The fault estimation problem is (mathematically) closely related to several detection problems arising in communications. In multi-user detection in CDMA (code division multiple access) systems, the received signal plays the role of the measurements, and the transmitted bit pattern plays the role of the fault pattern; the goal is to estimate the transmitted bit pattern. (One important difference is that in the detection problem, each bit typically has an equal probability of being 0 or 1, whereas in fault detection, the prior probabilities that a bit is 1 is typically much lower.) As has been pointed out in the literature, a good approach here is to solve a relaxed version of the resulting combinatorial problem. For example in [29,30] the authors propose a semidefinite programming (SDP) relaxation of the resulting mixed integer quadratic program. The performance of this method is theoretically analyzed in Ref. [31], while the authors of Ref. [32] perform an extensive experimental comparison of this SDP relaxation with various other heuristics.

The SDP relaxation heuristic has also been applied to the problem of MIMO detection in communications. In Refs. [33,34] the authors establish theoretical conditions under which the SDP heuristic solution coincides with the maximum likelihood solution for this problem. In Ref. [35], the authors show that an SDP-based detector for this problem achieves maximum diversity, when the elements of the channel matrix are independently drawn from a zero mean Gaussian distribution. This shows that the performance of the SDP relaxation based detector is the same as the maximum likelihood detector when the SNR is high.

The method we propose for fault estimation, which is based on *linear* relaxation of Boolean constraints, is *less* sophisticated than the SDP relaxation. However, our method has the advantage of easily scaling to very large problems, and (in conjunction with a rounding and local optimization scheme we propose) seems to give excellent results in fault estimation. We will see that SDP relaxation of the fault detection problem does provide better bounds on the optimal posterior probability than the linear relaxation; but after rounding and a simple local search, the SDP relaxation performs no better, in terms of fault detection, than the simple linear relaxation, which requires far less computational effort.

Finally we mention another technique that we believe could work well for the fault idetification problem: belief propagation. This is an iterative message passing algorithm, that has been proven to work very well in the context of multi-user detection in CDMA systems [36–38], as well as the problem of distributed beamforming [39]. We have not found any literature on belief propagation applied to fault identification; but we can refer the reader to the tutorial papers [40,41] for a general discussion of belief propagation.

1.2. Outline

In Section 2 we describe the measurement setup and the basic MAP approach, mostly to set our notation for later use. In Section 3 we describe three possible convex relaxations of the combinatorial MAP problem and we briefly compare them in terms of effectiveness and computational complexity. In Sections 4 and 5 we describe our method for approximately solving the MAP problem, and we illustrate its effectiveness in numerical examples in Section 6. We conclude by listing some extensions in Section 7.

2. MAP estimation

In this section we describe the model in detail, and the basic MAP method for estimating the fault pattern.

2.1. Fault model and prior distribution

We consider a system in which any of the 2^n combinations of n potential faults can occur. (In Section 7 we show how to handle the case when there are constraints linking the possible faults.) We encode a fault pattern, i.e., a set of faults, as a vector $x \in \{0,1\}^n$, where $x_j = 1$ means that fault j has occurred. We assume that faults occur independently, and that fault j occurs with known probability p_j. Thus, the (prior) probability of fault

pattern x occurring is

$$p(x) = \prod_{j=1}^{n} p_j^{x_j}(1-p_j)^{1-x_j}.$$

The fault pattern $x = 0$ corresponds to the null hypothesis, the situation in which no faults have occurred. This occurs with probability $p(0) = \prod_{j=1}^{n}(1-p_j)$. The expected number of faults is $\sum_{j=1}^{n} p_j$.

2.2. Measurement model

We assume that m scalar real measurements, denoted $y \in \mathbf{R}^m$, are available. These measurements depend on the fault pattern x in the following way:

$$y = \phi(Ax + v),$$

where $A \in \mathbf{R}^{m\times n}$ is the *fault signature matrix*, $v \in \mathbf{R}^m$ is a vector of measurement noises, and $\phi : \mathbf{R}^m \to \mathbf{R}^m$ is the measurement nonlinearity, which acts separately on the individual measurements:

$$\phi(z) = (\phi_1(z_1), \ldots, \phi_m(z_m)),$$

where $\phi_i : \mathbf{R} \to \mathbf{R}$ is the nonlinearity in the ith measurement. When $\phi(z) = z$, we say the measurement is linear.

We assume the fault signature matrix A is known. Its jth column $a_j \in \mathbf{R}^m$ gives the measurement, if the measurement were linear and there were no noise, when only fault j has occurred. For this reason a_j is called the jth *fault signature*. Since x is a Boolean vector, Ax is just the sum of the fault signatures corresponding to the faults that have occurred. So our real assumption here is only that, without measurement noise and nonlinearity, the measurements would be additive in the faults.

We assume the measurement noise $v \in \mathbf{R}^m$ is random, with v_i independent of each other and x, each with $\mathcal{N}(0, \sigma^2)$ distribution.

2.2.1. Quantized measurements

Now we turn to the measurement nonlinearity. In addition to the linear case ($\phi(z) = z$), we will consider the case in which the nonlinearities ϕ_i represent quantization, with *threshold levels* $t_1 < t_2 < \cdots < t_{K-1}$:

$$\phi_i(z_i) = \begin{cases} w_1, & z_i < t_1, \\ w_k, & t_{k-1} \leqslant z_i < t_k, \quad k = 2, \ldots, K-1, \\ w_K, & t_{K-1} \leqslant z_i. \end{cases}$$

The numbers $w_1, \ldots, w_K$ are the output values of the quantizer, typically taken to be

$$w_k = (t_{k-1} + t_k)/2, \quad k = 2, \ldots, K-1,$$

$w_1 = t_1 - 1$, and $w_K = t_K + 1$. The actual values of these numbers do not matter, as long as they are distinct; any specific measurement simply identifies which of the K intervals z_i lies in. For example, $\phi_i(z_i) = w_1$ means that $z_i \in (-\infty, t_1)$, i.e., $z_i < t_1$; $\phi_i(z_i) = w_3$ means that $z_i \in (t_2, t_3)$. If $K = 2^b$, we say that the quantizer has b bits. Our treatment of arguments that are exactly at a threshold level (i.e., assigning $\phi_i(t_k) = w_{k-1}$ instead of w_k) is arbitrary, and statistically irrelevant.

The simplest example of a quantizer function is a 1-bit quantizer (i.e., $K = 2$) with threshold level $t_1 = 0$. In this case we have

$$\phi_i(z_i) = \mathbf{sgn}(z_i) = \begin{cases} +1 & z_i \geqslant 0, \\ -1 & z_i < 0. \end{cases}$$

2.3. Posterior probability

Let $p(x|y)$ denote the (posterior) probability of fault pattern x, given the measurement y. We define the loss of x, given the measurement y, as the log of the ratio of the posterior probability of the null hypothesis to the posterior probability of x, i.e.,

$$\begin{aligned} l_y(x) &= \log \frac{p(0|y)}{p(x|y)} \\ &= \log\left(\frac{p(0)p(y|0)}{p(x)p(y|x)}\right) \\ &= \log p(0) - \log p(x) + \log p(y|0) - \log p(y|x) \\ &= \lambda^T x + \sum_{i=1}^{m} (\log p(y_i|0) - \log p(y_i|x)), \end{aligned}$$

where $\lambda_j = \log((1-p_j)/p_j)$. In these expressions we have to interpret $p(y_i|x)$ and $p(y_i|0)$ carefully. When y is a linear measurement, these are conditional *densities*; when y_i takes on only a finite number of values, as occurs with quantized measurements, these are actual *probabilities*.

The loss tells us how improbable it is that fault x has occurred, given the measurement y, compared to the null hypothesis $x = 0$. If $l_y(x) = 0$, the fault pattern x is just as probable as the null hypothesis $x = 0$. If $l_y(x) = -1$, the fault pattern x is e times more probable than the null hypothesis.

We now work out the loss function more explicitly, for the cases of linear and quantized measurements.

2.3.1. Linear measurements

In this case the conditional density of y_i given x is

$$p(y_i|x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-(y_i - \tilde{a}_i^T x)^2/2\sigma^2),$$

where $\tilde{a}_i$ is the ith row of A. Therefore we have

$$\begin{aligned} l_y(x) &= \lambda^T x + (1/2\sigma^2)\sum_{i=1}^{m}(-y_i^2 + (y_i - \tilde{a}_i^T x)^2) \\ &= \lambda^T x + (1/2\sigma^2)(-\|y\|^2 + \|Ax - y\|^2) \\ &= (1/2\sigma^2)x^T A^T A x + (\lambda - (1/\sigma^2)A^T y)^T x, \end{aligned}$$

which is a convex quadratic function of x.

2.3.2. Quantized measurements

Our measurement y_i tells us that $\tilde{a}_i^T x + v_i$ lies in an interval, say, $(t_i^{\text{low}}, t_i^{\text{high}})$. (The index i here refers to measurement i, not threshold level i.) In this case the conditional probability of y_i given x is

$$p(y_i|x) = \Phi\left(\frac{-\tilde{a}_i^T x + t_i^{\text{high}}}{\sigma}\right) - \Phi\left(\frac{-\tilde{a}_i^T x + t_i^{\text{low}}}{\sigma}\right),$$

where

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} \exp(-t^2/2)\,\mathrm{d}t$$

is the cumulative distribution function of the standard normal distribution. Thus, we have

$$l_y(x) = \lambda^T x + \sum_{i=1}^{m} \log\left(\Phi\left(\frac{t_i^{\mathrm{high}}}{\sigma}\right) - \Phi\left(\frac{t_i^{\mathrm{low}}}{\sigma}\right)\right) - \sum_{i=1}^{m} \log\left(\Phi\left(\frac{-\tilde{a}_i^T x + t_i^{\mathrm{high}}}{\sigma}\right) - \Phi\left(\frac{-\tilde{a}_i^T x + t_i^{\mathrm{low}}}{\sigma}\right)\right).$$

This is a complicated, but *convex* function of x. The first term is linear, and the second is constant; the tricky part is verifying convexity of each term in the third part of $l_y(x)$. This follows from concavity, with respect to the variable z, of

$$\varphi(z) = \log(\Phi(\alpha - z) - \Phi(\beta - z)) = \log \frac{1}{\sqrt{2\pi}} \int_{\beta - z}^{\alpha - z} \exp(-t^2/2)\,\mathrm{d}t,$$

where $\alpha > \beta$. (This is the log of the probability that a $\mathcal{N}(z,1)$ random variable lies in $[\beta, \alpha]$.) Concavity of φ follows from log-concavity of $\Phi(\alpha - z) - \Phi(\beta - z)$, which is the convolution of two log-concave functions (the Gaussian density and the function that is one between β and α and zero elsewhere). For more on log-concavity and convolution, see Ref. [42, Section 3.5.2].

2.4. MAP estimation

To find the fault pattern x with MAP (or equivalently, minimum loss $l_y(x)$) we solve the problem

$$\begin{array}{ll} \text{minimize} & l_y(x) \\ \text{subject to} & x_j \in \{0,1\},\ j = 1,\ldots,n, \end{array} \tag{1}$$

with variable x. We have already seen that with linear or quantized measurements, the objective function is convex. The constraint is that x is a Boolean vector, so a problem like this is sometimes called a Boolean-convex program or a mixed-integer convex program. When the measurements are linear, the MAP problem (1) is a convex mixed-integer quadratic program (MIQP).

Any solution of the MAP problem (1) is a MAP estimate of the fault pattern, i.e., a fault pattern that is most probable, given the measurement. It is also very useful to obtain the ambiguity group, i.e., the set of fault patterns with loss that is near to the loss of a MAP estimate. If all other fault patterns have a loss much larger than the MAP estimate (i.e., the ambiguity group is empty), we can have high confidence in our estimate. On the other hand, if several other fault patterns have a loss similar to the MAP loss, they explain the measurement almost as well as the MAP estimate, and so must be considered possible values of the true fault. One way to determine the ambiguity group is to find the K fault patterns with least loss, i.e., highest posterior probability. From these ambiguity group candidates, we can form the ambiguity group by taking only the patterns with loss near to the MAP loss.

Unfortunately, it can be very difficult to solve the MAP problem (1) exactly. There are, of course, 2^n feasible points for the MAP problem (1), corresponding to all possible fault patterns. One method to solve it is to simply evaluate $l_y(x)$ for each fault pattern. But this is feasible only for n smaller than around 15 or so. Another approach to solving the MAP problem (1) exactly is to use a branch and bound algorithm [43,44]. While these methods do solve the problem exactly, their worst-case complexity is exponential, i.e., the same as direct enumeration. In practice as well, branch and bound algorithms can be very slow. But it is not clear that solving the MAP problem (1) globally is needed, since our real goal is to estimate the fault pattern.

Several simple heuristics have been used to find a pattern x with low, if not lowest, loss. One approach is to directly evaluate $l_y(x)$ for a small subset of fault patterns. For example, we can search over all n single fault patterns, i.e., $x = e_i$, where e_i is the ith unit vector. If n is small enough, we can search over all $n(n-1)/2$ fault patterns with two faults. If there is a high chance that only one or two faults have occurred, this method will work well; but obviously it cannot work well when the expected number of faults is more than two. A wide variety of heuristics and local optimization methods can be used to enhance the chance of finding a pattern with low loss, while searching over a reasonable number of patterns. For all of these enumeration methods, we can maintain a list of ambiguity group candidates, by simply keeping track of the K best fault patterns found.

The method we present in this paper is, like these methods, a heuristic for solving the MAP problem (1) approximately. Our approach is based on relaxing the original MAP problem to a convex problem and then using the solution of this relaxed problem to generate a set of good candidate fault patterns, whose loss we evaluate.

3. Convex relaxations

In this section we describe three different convex relaxations of the MAP problem (1). These relaxed problems are convex, and so can be effectively (globally) solved. They each give a *lower bound* on the minimum loss value of the MAP problem, as well as a *soft decision* on the fault pattern, i.e., a value between 0 and 1 for each fault.

3.1. Linear relaxation

The first relaxation of problem (1) that we look at is

$$\begin{array}{ll} \text{minimize} & l_y(x) \\ \text{subject to} & 0 \leqslant x_j \leqslant 1,\ j = 1,\ldots,n, \end{array} \tag{2}$$

with variable $x \in \mathbf{R}^n$. In (2) we allow the variables x_i to take values between 0 and 1, whereas in the MAP problem (1), these variables must be either 0 or 1. The relaxed MAP problem (2) is a *convex* problem and can be solved very efficiently and reliably in many ways, e.g., via interior-point methods [42,45]. In the case where all measurements are linear, the relaxed MAP problem reduces to a (convex) quadratic program and is easily solved [46,47]. The general relaxed MAP problem (2) is called the linear

relaxation, since we replace the Boolean constraints with linear ones. When the measurements are linear, the resulting problem is a QP, in which case we refer to the problem as the QP relaxation.

Since the feasible set for the relaxed MAP problem (2) contains the feasible set for the MAP problem (1), the optimal value of the relaxed MAP problem, which we denote l_{lb}, gives a lower bound on the optimal value of the MAP problem.

Let $z^\star$ be an optimal point for the relaxed MAP problem (2), so we have $l_{\mathrm{lb}} = l_y(z^\star) \leqslant l_y(x)$ for any Boolean x. If $z^\star$ is also Boolean, i.e., $z_j^\star \in \{0,1\}$ for all j, we conclude that $z^\star$ is in fact optimal for the MAP problem. In other words: When a solution to the relaxed MAP problem turns out to have Boolean entries, it is optimal for the MAP problem. In general, of course, this does not happen; at least some values of $z_j^\star$ will lie between 0 and 1.

As a special case, we conclude that when $z^\star = 0$, i.e., 0 is a solution of the relaxed MAP problem, then 0 is a solution of the MAP problem. In other words, the MAP estimate is the null hypothesis; we can guess that no faults have occurred. We can derive simple conditions under which $z = 0$ solves the relaxed MAP (and hence, MAP) problem. This occurs if and only if

$$\nabla l_y(0) \geqslant 0,$$

where the inequality is elementwise. (See, e.g., [42, Section 4.2.3].) For the case of linear measurements, this corresponds to

$$\lambda \geqslant \frac{1}{\sigma^2} A^T y.$$

If this inequality is satisfied, we can immediately conclude that the null hypothesis $x = 0$ maximizes the posterior probability.

3.2. SDP relaxations for linear measurements

We now derive two convex relaxations of the MAP problem that are restricted to the case when the measurements are linear. Both relaxations are (or can be expressed as) SDPs. For a general reference on SDP relaxations of quadratically constrained quadratic programs see Ref. [48]. We start by writing the MAP problem (1), for the case of linear measurements, as

$$\begin{array}{ll} \text{minimize} & (1/2\sigma^2)\mathbf{Tr}(A^TAX) + (\lambda - (1/\sigma^2)A^Ty)^Tx \\ \text{subject to} & X = xx^T \\ & X_{jj} = x_j, \quad x_j \in \{0,1\}, \ j = 1,\ldots,n, \end{array} \tag{3}$$

with variables $X \in \mathbf{S}^n$ and $x \in \mathbf{R}^n$; $\mathbf{S}^n$ denotes the set of symmetric $n \times n$ matrices (the constraint $x_j \in \{0,1\}$ is redundant; we include it to make the SDP relaxation of this problem more natural.) Problems (1) and (3) are equivalent, since $\mathbf{Tr}(A^TAxx^T) = x^TA^TAx$.

We now present two SDP relaxations of problem (3). The first one is

$$\begin{array}{ll} \text{minimize} & (1/2\sigma^2)\mathbf{Tr}(A^TAX) + (\lambda - (1/\sigma^2)A^Ty)^Tx \\ \text{subject to} & X \succeq xx^T \\ & X_{jj} = x_j, \quad 0 \leqslant x_j \leqslant 1, \ j = 1,\ldots,n, \end{array} \tag{4}$$

with variables $X \in \mathbf{S}^n$ and $x \in \mathbf{R}^n$, while the second one is

$$\begin{array}{ll} \text{minimize} & (1/2\sigma^2)\mathbf{Tr}(A^TAX) + (\lambda - (1/\sigma^2)A^Ty)^Tx \\ \text{subject to} & X \succeq xx^T \\ & X_{jj} = x_j, \quad 0 \leqslant x_j \leqslant 1, \quad j = 1,\ldots,n, \\ & X_{kj} \geqslant 0, \quad k = 1,\ldots,n, \ j = 1,\ldots,n, \end{array} \tag{5}$$

again with variables $X \in \mathbf{S}^n$ and $x \in \mathbf{R}^n$. The relaxations (4) and (5) can be expressed as SDPs, using

$$X \succeq xx^T \iff \begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0.$$

They can both be solved using interior-point methods [42, Chapter 11]; software for solving SDPs, such as SDPT3 [47], is widely available.

3.3. Comparison of relaxations

We can compare the lower bounds obtained with each of the three relaxations, for the case of linear measurements. Let l_{qp}, l_{sdp}, and $l_{\mathrm{sdp+}}$ be the optimal values of problems (2), (4), and (5), respectively. We will now show that

$$l_{\mathrm{qp}} \leqslant l_{\mathrm{sdp}} \leqslant l_{\mathrm{sdp+}}. \tag{6}$$

In other words, the basic SDP relaxation always gives at least as good a lower bound on the optimal loss as the QP relaxation, and the augmented SDP relaxation, with the additional nonnegativity constraint, always gives the same, or better, bound as the basic SDP relaxation.

It is easy to see why the righthand inequality in (6) holds. Problem (5) is the same as problem (4), except for the addition of the inequality constraints of the form $X_{kj} \geqslant 0$. Thus the feasible set for problem (4) is smaller than the feasible set of problem (4), which means that $l_{\mathrm{sdp}} \leqslant l_{\mathrm{sdp+}}$.

To show that the lefthand inequality in (6) holds, we first reformulate the QP relaxation (2) as the following SDP

$$\begin{array}{ll} \text{minimize} & (1/2\sigma^2)\mathbf{Tr}(A^TAX) + (\lambda - (1/\sigma^2)A^Ty)^Tx \\ \text{subject to} & X \succeq xx^T, \ 0 \leqslant x_j \leqslant 1, \quad j = 1,\ldots,n, \end{array} \tag{7}$$

with variables $x \in \mathbf{R}^n$ and $X \in \mathbf{S}^n$. To see why these two problems are equivalent it is sufficient to note that the linear matrix inequality $X \succeq xx^T$ holds with equality at optimality. This is due to the fact that $\mathbf{Tr}(A^TAX)$ is a matrix increasing function of X [42, Section 3.6.1], so for fixed x, the optimal choice for X is $X = xx^T$.

The problems (4) and (2) differ only in the addition of the constraints $X_{jj} = x_j$; it follows that the feasible set of (4) is smaller than that of (2). This implies that $l_{\mathrm{qp}} \leqslant l_{\mathrm{sdp}}$. In fact this inequality holds with equality if the solution to either problem is binary and is thus equal to the solution of the MAP problem.

While the basic and augmented SDP relaxations give better lower bounds that the QP relaxation, they are considerably harder to solve. A generic interior-point method, that does not exploit any problem structure, requires $O(n^3)$ operations to solve the QP relaxation (2) [42,45], but $O(n^6)$ operations to solve the augmented SDP (5), since it involves both a matrix inequality and

inequalities on the individual components. While faster methods for solving the SDP relaxations can be developed, we will see that, in terms of the quality of fault estimation, the SDP relaxations provide no benefit over the QP relaxation.

4. Relaxed MAP estimation

4.1. Variable threshold rounding

Let $z^\star$ denote the optimal point in one of the convex relaxations. We refer to $z^\star$ as a *soft decision*, since its components can be strictly between 0 and 1. The next step is to round the soft decision $z^\star$ to obtain a valid Boolean fault pattern (or *hard decision*). Let $\theta \in (0,1)$ and set

$$\hat{x} = \mathbf{sgn}(z^\star - \theta).$$

To create $\hat{x}$, we simply round all entries of $z_i^\star$ smaller than the threshold θ to zero. Thus θ is a threshold for guessing that a fault has occurred, based on the relaxed MAP solution $z^\star$. As θ varies from 0 to 1, this method generates up to n different estimates $\hat{x}$, as each entry in z falls below the threshold. We can efficiently find them all by sorting the entries of $z^\star$, and setting the values of $\hat{x}_i$ to one in the order of increasing $z_i^\star$.

We evaluate the loss for each of these, and can take as our relaxed MAP fault estimate the one that has least loss, which we denote by x^{rmap}. We can also take the K fault patterns with least loss as candidates for the ambiguity group.

4.2. Local optimization

Further improvement in our estimate can sometimes be obtained by a local optimization method, starting from x^{rmap}. We describe here the simplest possible local optimization method. We initialize $\hat{x}$ as x^{rmap}. We then cycle through $j = 1, \ldots, n$, at step j replacing $\hat{x}_j$ with $1 - \hat{x}_j$. If this leads to a reduction in the loss function, we accept the change, and continue. If (as usually is the case) flipping the jth bit results in an increase in l_y, we go on to the next index. We continue until we have rejected changes in all entries in $\hat{x}$. (At this point we can be sure that $\hat{x}$ is 1-OPT, which means that no change in one entry will improve the loss function.) Numerical experiments show that this local optimization method often has no effect, which means that x^{rmap} is often 1-OPT. In some cases, however, it can lead to modest reduction of loss, compared to x^{rmap}.

This local optimization method can also be used to improve our candidate ambiguity group. When we evaluate the loss of a candidate, we insert it in our list of K least loss fault patterns, whenever it is better than the worst fault pattern in the list.

We refer to the procedure of convex relaxation, followed by variable threshold rounding, and, possibly, local optimization, as *relaxed MAP (RMAP) estimation*.

5. Approximate RMAP estimation

It is clearly not necessary to solve the RMAP (2) to high accuracy, since we round the entries to form our fault pattern estimate. In this section we describe a method for solving problem (2) approximately, while still retaining a guaranteed lower bound l_{lb}. The method is extremely efficient, can scale to large problems, and produces fault pattern estimation results of equal quality. We describe this method for the linear relaxation (2) only; similar methods can be derived for the SDP relaxations.

5.1. Barrier approximation

We form the optimization problem

$$\text{minimize } l_y(x) + \kappa\psi(x), \tag{8}$$

with variable x, where

$$\psi(x) = -\sum_{j=1}^{n}(\log x_j + \log(1 - x_j))$$

is the logarithmic barrier for the constraints $0 \leqslant x \leqslant \mathbf{1}$, and $\kappa > 0$ is a parameter. This is a smooth convex optimization problem, with implicit constraints $0 < x_i < 1$, and can be solved using Newton's method [42, Section 9.5].

The parameter κ controls the accuracy with which (8) approximates (2); for example, it can be shown that the solution of (8) is no more than $2n\kappa$ suboptimal for the problem (2) (see, *e.g.*, [42, Section 11.2.2]). It follows that the optimal value of (8), minus $2n\kappa$, is a bound on the optimal value of the original MAP problem (1). The suboptimality is only bounded by $2n\kappa$; the true suboptimality is often less. Moreover, since we will round the solution, and possibly perform local optimization, we will see that the solution of (8) yields very good fault estimates even when κ is relatively large, i.e., when $2n\kappa$ is not small. The idea of approximating convex constraints with a fixed barrier term has been applied succesfully to a number of problems such as fast computation of optimal contact forces [49] and fast model predictive control [50].

5.2. Newton's method

We use Newton's method to solve (8), starting from the point $x = (1/2)\mathbf{1}$. At each iteration we compute the Newton step

$$\Delta x_{\mathrm{nt}} = -(\nabla^2 l_y(x) + \kappa\nabla^2\psi(x))^{-1}(\nabla l_y(x) + \kappa\nabla\psi(x)).$$

Then we use a backtracking line search to compute a step size $s \in (0,1)$, and update the current point to $x + s\Delta x_{\mathrm{nt}}$. (See [42, Chapter 10] for all details.)

For completeness we give explicit formulas for the gradient and Hessian of ψ and l_y. The gradient of the barrier function is

$$(\nabla\psi(x))_j = \frac{2x_j - 1}{x_j(1 - x_j)}, \quad j = 1, \ldots, n.$$

Its Hessian is diagonal, with diagonal entries

$$(\nabla^2\psi(x))_{jj} = \frac{x_j^2 + (1-x_j)^2}{x_j^2(1-x_j)^2}, \quad j = 1,\ldots,n.$$

Now consider l_y for the case of linear measurements. Its gradient and Hessian are

$$\nabla l_y(x) = \lambda + (1/\sigma^2)A^T(Ax-y), \quad \nabla^2 l_y(x) = (1/\sigma^2)A^TA,$$

respectively.

Now we consider the case of quantized measurements. The gradient and Hessian of $l_y(x)$ are then given by

$$\nabla l_y(x) = \lambda - (1/\sigma)A^Tc, \quad \nabla^2 l_y(x) = (1/\sigma^2)A^TDA,$$

where

$$c_i = \frac{\exp(-l_i^2/2) - \exp(-u_i^2/2)}{\int_{l_i}^{u_i}\exp(-t^2/2)\,dt},$$
$$u_i = (-a_i^Tx + t_i^{\text{high}})/\sigma,$$
$$l_i = (-a_i^Tx + t_i^{\text{low}})/\sigma,$$

for $i = 1,\ldots,m$. The matrix D is diagonal with

$$D_{ii} = \frac{u_i\exp(-u_i^2/2) - l_i\exp(-l_i^2/2)}{\int_{l_i}^{u_i}\exp(-t^2/2)\,dt} + c_i^2, \quad i = 1,\ldots,m.$$

5.3. Complexity of newton's method

A formal upper bound on the number of steps required by Newton's method can be found from self-concordance theory (see [51] or [42, Section 9.6]), but this bound does not accurately reflect the very good practical performance of Newton's method for solving the approximate RMAP problem (8). Newton's method typically requires 10 or fewer iterations to converge to high accuracy, and rarely more than 25 or so.

We can analyze the computational cost per iteration. Each step requires forming $\nabla^2\phi(x)$ and computing the Newton step, which costs on the order of mn^2 operations (if $m \geqslant n$) or m^2n operations (if $m \leqslant n$). In particular, the complexity grows linearly in the larger of m and n. This complexity analysis assumes that all calculations are done with full (dense) matrices; if sparsity can be exploited, the Newton step can be computed much more efficiently. For m and n on the order of 1000, each Newton step requires only a fraction of a second on a typical 2 GHz personal computer; the entire approximate relaxed optimization problem (8) can be solved in a few seconds. With m and n on the order of 100, the problem can be solved in a few milliseconds.

6. Numerical examples

6.1. Small example with linear measurements

We consider an example with $m = 50$ sensors, $n = 100$ possible faults, and linear measurements. The elements of A are chosen randomly and independently with $A_{ij} \sim \mathcal{N}(0,1)$. We set the noise standard deviation to $\sigma = 1$. The fault probability is $p_j = 0.05$ for all j, which means that the expected number of faults is 5. The signal to noise ratio (SNR) of each measurement is

$$\sqrt{\frac{\mathbf{E}(a_i^Tx)^2}{\mathbf{E}v_i^2}} = \sqrt{\frac{5}{1}} \approx 2.2.$$

The problem would seem to be quite challenging, since we have only 50 measurements, each with a signal to noise ratio around 2, to estimate a fault pattern with 100 possible faults.

We first generate one instance of the problem, which happens to have 4 faults. We solve the three relaxations, to obtain the lower bounds l_{qp}, l_{sdp}, and $l_{\text{sdp+}}$, which are shown in Fig. 1 as vertical lines. These were computed using the convex optimization modeling language CVX [46], which relies the SDP solver SDPT3 [47]. The dashed vertical line at right shows zero loss, which corresponds to fault patterns that are just as (un)likely as the null hypothesis.

The figure shows the fault estimates (and ambiguity groups) produced by searching over all single and double fault patterns, and also for RMAP (the basic QP relaxation, using the barrier approximation with $\kappa = 0.01$), SDP, and augmented SDP methods, all with local optimization. Each row shows one method, with the horizontal axis giving loss. The row labeled SDP uses the basic SDP relaxation (4); the row labeled SDP+ uses the augmented SDP relaxation (5). In each row, the 10 fault patterns with least loss are shown; the filled circles show the fault patterns with least loss.

For this problem instance, the basic RMAP and SDP methods found the same pattern with least loss, which happened to be the true fault pattern. (This was suggested, but not guaranteed, by the gap in loss between x^{rmap} and the next best fault pattern found.) We suspect, but do not know, that this fault pattern is the MAP estimate, i.e., a global solution of the MAP problem (1). The ambiguity groups produced by the three methods are also similar.

The time required to carry out the RMAP fault estimation, including local optimization, is a few milliseconds. On the other hand the time required to solve problem (4) using SDPT3 is about half a minute, while problem (5) requires about half an hour. (Substantial improvement in these running times could be obtained by developing a custom solver for these specific SDPs, but the basic approximate RMAP method will still be far faster.)

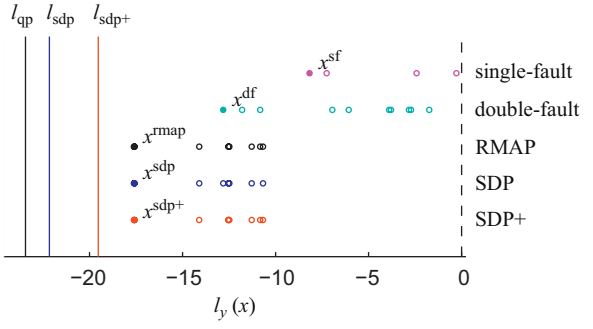

Fig. 1. Loss of generated fault patterns for a small problem instance.

When the same method is carried out on a large number of instances, with the same matrix A but different fault patterns and noise, we find that x^{rmap} (without local optimization) is equal to the true fault pattern 90% of the time, and that the true fault pattern is included in the ambiguity group 93% of the time. When we use local optimization, we find that our final RMAP estimate is equal to the true fault pattern 95% of the time, and the true fault pattern lies within the ambiguity group 98% of the time. This shows that the local optimization method does indeed improve our fault estimates. The performance of RMAP seems quite surprising, since we have only half as many measurements as faults to detect, and the SNR is quite low.

To be sure that our soft decisions are contributing substantially to the quality of the final estimate, after local optimization, we run the local optimization method, starting from the null hypothesis $x = 0$, instead of our RMAP estimate. When this is done, the fraction of time the true fault is identified is only 69%. This experiment shows that local optimization alone performs much more poorly than local optimization, starting from a rounded soft decision.

We also examined the effect of the parameter κ on our RMAP fault estimation method, with and without local optimization. To do this we generate 1000 random instances of this problem (all with the same signature matrix A) for 20 values of κ, logarithmically spaced between 10^{-3} and 10^{1}. Fig. 2 shows the percentage of times that the true fault pattern was identified, as a function of κ. The first thing to note is that RMAP works well, even without local optimization, for κ up to 0.05 or so. This corresponds to a suboptimality bound $2n\kappa = 10$, which corresponds to a probability ratio e^{10}, which is definitely not small. So we see that our approximate RMAP method yields good fault estimates, even when the solution of the approximate RMAP problem (2) is not a particularly good estimate of the solution of the RMAP problem (2). The plots show that the performance of both methods deteriorates for κ greater than10^{-1}, but much less rapidly when local optimization is used.

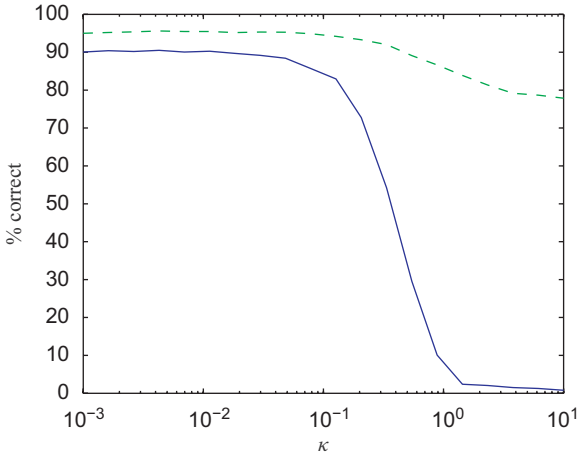

Fig. 2. Performance of RMAP method without (solid) and with (dashed) local optimization, versus κ.

We now explore how well our RMAP method with relaxation (2) works as the SNR varies. We let σ take on 30 values over a range from 0.1 to 4, and for each value, we generate 10 000 random instances of the problem. For each instance, we compute a set of fault patterns using RMAP (with $\kappa = 0.01$), with and without local optimization. We then check if our fault estimate is the true one, and whether the true one is included in our ambiguity group of the $K = 10$ fault patterns with least loss. The results are shown in Fig. 3. Fig. 4 Both methods work extremely well for SNR higher than 3, and fail to work at all for SNR less than 1. We can see that the use of local search improves the performance of the method, especially for the low SNR region. For example, for an SNR of about 1.4, the top 10 hit percentage without local optimization is 54%, while with local optimization it is 71%.

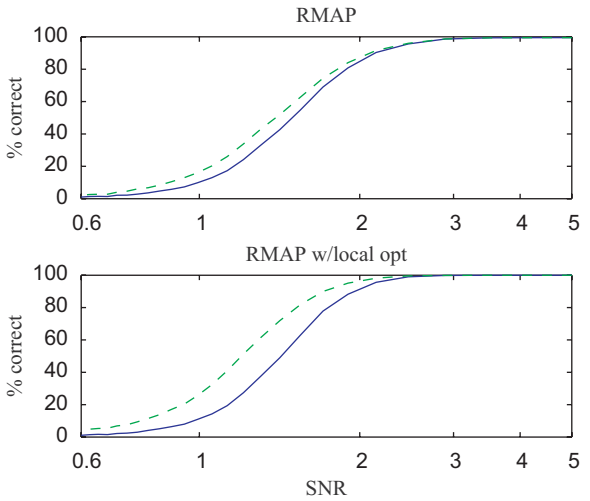

Fig. 3. Performance of RMAP method without (top) and with (bottom) local optimization, versus SNR. Solid line shows top hit percentage; dashed line shows top 10 hit percentage.

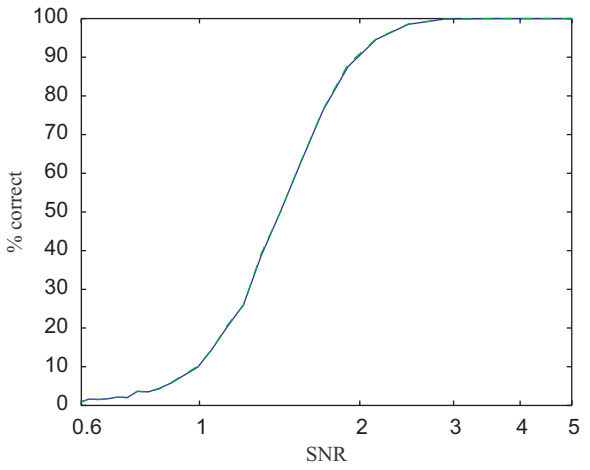

Fig. 4. Performance of QP (solid) and SDP (dashed) relaxations.

Finally, we compare the results obtained with the basic QP relaxation and the basic SDP relaxation, using the same set of experiments described above. The results, shown in Fig. 3, and almost the same for the two relaxations. The computational costs of the two methods, however, are considerably different.

6.2. Small example, quantized measurements

We now consider the same example, but with quantized measurements, with 1–4 bits per measurement, using the thresholds listed in the table below.

Quantization	Threshold levels
1-bit	0
2-bit	$0, \pm 2$
3-bit	$0, \pm 2, \pm 4, \pm 6$
4-bit	$0, \pm 1, \pm 2, \ldots, \pm 7$

We generated 10 000 instances of the given problem, for 30 values of σ linearly spaced between 0.1 and 4. For each instance we solved the RMAP problem (with local search) for the different quantization levels, including full precision (i.e., linear measurements). Fig. 5 shows the performance of the RMAP method versus SNR for all quantization levels. As expected, the performance improves with increasing numbers of bits in the quantizer. With 1-bit quantization (i.e., sign measurements), the performance is poor. The performance for 2-bit quantization is surprisingly good. The RMAP method performs very well for 3-bit measurements, while its performance with 4 bits is almost identical to the one with full precision.

For comparison, we also used an ad hoc method in which we ignored the quantization, treating each measurement as linear, using the midpoint of the quanization interval for its value. This method performed very poorly. Our use of the sophisticated loss function for quantized measurements is critical to achieving such high estimation performance from heavily quantized measurements.

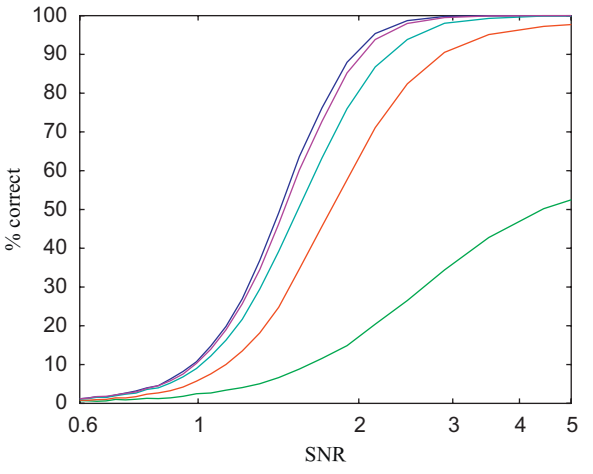

Fig. 5. Performance of RMAP method versus SNR for 1–4-bit quantization, and for linear measurements.

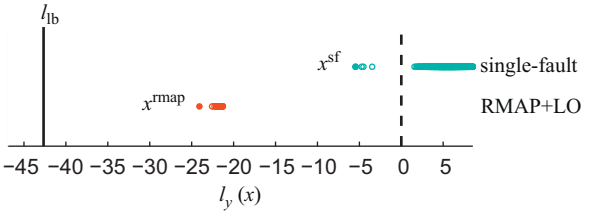

Fig. 6. Loss of generated fault patterns for a larger problem instance.

6.3. Larger example

We now look at a larger example, with $m = 10\,000$ measurements and $n = 2000$ potential faults. We choose A to be sparse, with about 200 000 (1%) nonzero elements. The indices of the nonzero elements are chosen at random. Thus, each measurement is affected by around 20 faults, and each fault will affect around 100 measurements. Each nonzero A_{ij} is chosen randomly and independently from a $\mathcal{N}(0,1)$ distribution. The noise standard deviation is $\sigma = 1.5$ and each fault has probability of occurence $p_f = 0.002$, so the expected number of faults is 4. The SNR for each measurement is

$$\sqrt{\frac{\mathbf{E}(a_i^T x)^2}{\mathbf{E} v_i^2}} = \sqrt{\frac{0.4}{1.5^2}} \approx 0.42.$$

In this case the measurement SNR is poor, but there are five times as many measurements as there are potential faults.

Fig. 6 shows the performance of RMAP, with local optimization, and κ set to 0.01, compared to direct enumeration of all single fault patterns, for a given instance of the problem. For this instance there are five faults in the true fault pattern. Our simple Matlab implementation, which exploits sparsity in A, requires about 20 s to run; a C implementation would have been far faster. In this instance, x^{rmap} corresponds to the true fault pattern.

We generated 100 random instances of the same problem (with the same signature matrix A but different true faults and noise). We found that RMAP found the true fault in 92 cases. In 98 cases, the true fault pattern was contained in the ambiguity group.

7. Extensions

7.1. Variable misclassification cost

We have dealt with the case where the cost of misclassification is the same for each fault, and the same for a false positive and false negative. We can easily extend our problem to minimize an average loss, with different loss values associated with false positive and false negative estimation errors, which can differ for each fault, since the objective function will remain convex.

7.2. Mixed measurements

We have described the problem for the cases of linear and quantized measurements separately. But in fact, we

can easily handle the case when some measurements are linear, and others are quantized. (The notation, however, grows cumbersome.)

7.3. Correlated noise

Our method can be easily extended to the case where the measurement noise is correlated, i.e., $v \sim (0, \Sigma)$, and the measurements are linear. In this case the conditional density of y given x is

$$p(y|x) = \frac{1}{(2\pi)^{m/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(y - Ax)^T \Sigma^{-1}(y - Ax)\right),$$

so the loss function becomes

$$l_y(x) = \frac{1}{2}((y - Ax)^T \Sigma^{-1}(y - Ax) - y^T \Sigma^{-1} y) + \lambda^T x,$$

which is again a convex quadratic function in x.

7.4. Noise distribution

We can consider the case when the noise distribution is zero-mean Laplacian, with variance $\sigma^2 = 2b^2$, i.e., with density

$$f(v_i) = \frac{1}{\sqrt{2}b} \exp(-|v_i|/b).$$

In this case the loss function for linear measurements is

$$l_y(x) = \frac{1}{b}(\|Ax - y\|_1 - \|y\|_1) + \lambda^T x,$$

where $\|z\|_1$ denotes the ℓ_1 norm. This loss function is convex, so the same approach can be used.

Since the Laplacian density is log-concave, the loss function for quantized measurements, with Laplacian noise, will also be convex. The formula is quite cumbersome, though, so we omit it.

7.5. Problem constraints

We take into account constraints on the occurence of faults, by imposing linear equality and inequality constraints on x, which are convex. For example the constraint that fault x_j occurs only if fault x_i occurs can be represented as $x_j \leqslant x_i$. To say that only one of faults 1–3 have occurred, we impose the constraint $x_1 + x_2 + x_3 = 1$. When we add these constraints, the rounding mechanism and local optimization steps must be modified to conform to them. With simple constraints like the ones described above, this is straightforward.

We consider an example as a special case. We consider a dynamical system, where $x(t) \in \{0, 1\}^n$ denotes the (vector of) faults that occur at time t, for $t = 1, \ldots, T$. We assume that each fault cannot fix itself; once a fault occurs, it stays. In this problem, then, we are to decide when (or if) each fault has occurred. We form the following relaxation:

$$\begin{array}{ll} \text{minimize} & l_y(x) \\ \text{subject to} & 0 \leqslant x(1) \leqslant \cdots \leqslant x(T) \leqslant \mathbf{1}, \end{array}$$

where the inequalities are componentwise, and $\mathbf{1}$ denotes the vector with all entries one. Our rounding method works as is; we can modify the local search to search over the fault occurence time, for each fault.

8. Conclusions

In this paper we propose a new method estimating which faults have occurred, given noise corrupted linear, or quantized, measurements. The method is based on approximately solving the MAP problem, using a convex relaxation followed by rounding, and, possibly, local optimization. The same method can also generate an approximation of the ambiguity group, i.e., a list of other fault patterns with nearly maximum posterior probability. Since the method is based on convex optimization, it is extremely efficient. Examples show that the method is very effective in cases where simpler methods, such as searching over all single or double faults, cannot be. More sophisticated methods, such as those based on SDP relaxations, produce better lower bounds on the optimal loss, but perform no better in terms of fault estimation.

Acknowledgments

This material is based upon work supported by the Focus Center Research Program Center for Circuit and System Solutions award 2003-CT-888, by the Precourt Institute on Energy Efficiency, by NSF award ECS-0423905, by NSF award 0529426, by DARPA award N66001-06-C-2021, by NASA award NNX07AEIIA, and by AFOSR award FA9550-06-1-0312.

We thank Tom Luo and Dani Palomar for pointing us to relevant work in the context of CDMA multiuser detection. We also want to thank Björn Ottersten for reading an earlier draft of this paper, as well as pointing us to useful references in the context of MIMO detection.

References

[1] D. Viassolo, S. Adibhatla, B. Brunell, J. Down, N. Gibson, A. Kumar, H. Mathews, L. Holcomb, Advanced estimation for aircraft engines, in: Proceedings of the American Control Conference, 2007, pp. 2807–2821.
[2] S. Ganguli, S. Deo, D. Gorinevsky, Parametric fault modeling and diagnostics of a turbofan engine, in: Proceedings of the IEEE International Conference on Control Applications, 2004, pp. 223–228.
[3] J. Hoskins, K. Kaliyur, D. Himmelblau, Fault diagnosis in complex chemical plants using artificial neural networks, A.I.Ch.E. Journal 37 (1) (1991) 137–141.
[4] J. Gertler, M. Costin, X. Fang, R. Hira, Z. Kowalczuk, Q. Luo, Model-based on-board fault detection and diagnosis for automotive engines, Control Engineering Practice 1 (1) (1993) 3–17.
[5] C. Hood, C. Ji, Proactive network-fault detection, IEEE Transactions on Reliability 46 (3) (1997) 333–341.
[6] F. Feather, D. Siewiorek, R. Maxion, Fault detection in an ethernet network using anomaly signature matching, Applications, Technologies, Architectures, and Protocols for Computer Communication (1993) 279–288.
[7] P. Stelling, C. DeMatteis, I. Foster, C. Kesselman, C. Lee, G. von Laszewski, A fault detection service for wide area distributed computations, Cluster Computing 2 (2) (1999) 117–128.
[8] J. Bandler, A. Salama, Fault diagnosis of analog circuits, Proceedings of the IEEE 73 (8) (1985) 1279–1325.

[9] P. Frank, Fault diagnosis in dynamic systems using analytical and knowledge-based redundancy—A survey and some new results, Automatica 26 (3) (1990) 459–474.
[10] A. Saberi, A. Stoorvogel, P. Sannuti, Fundamental problems in fault detection and identification, International Journal of Robust Nonlinear Control 10 (2000) 1209–1236.
[11] U. Lerner, R. Parr, D. Koller, G. Biswas, Bayesian fault detection and diagnosis in dynamic systems, in: Proceedings of the American Association of Artificial Intelligence, 2000, pp. 531–537.
[12] M. Morari, A. Bemporad, D. Mignone, A framework for control, state estimation, fault detection, and verification of hybrid systems, Scientific Computing in Chemical Engineering II 2 (1999) 46–61.
[13] R. Isermann, Process fault detection based on modeling and estimation methods, Automatica 20 (4) (1984) 387–404.
[14] J. Gertler, Fault Detection and Diagnosis in Engineering Systems, Marcel Dekker, New York, 1998.
[15] G. Goodwin, M. Seron, J.D. Doná, Constrained Control and Estimation: An Optimisation Approach, Springer, Berlin, 2004.
[16] R. Reiter, A theory of diagnosis from first principles, Artificial Intelligence 32 (1) (1987) 57–95.
[17] J. de Kleer, B. Williams, Diagnosing multiple faults, Artificial Intelligence 32 (1) (1987) 97–130.
[18] F. Tu, K. Pattipati, S. Deb, V. Malepati, Multiple fault diagnosis in graph-based systems, in: Proceedings of SPIE, Component and Systems Diagnostics, Prognostics, and Health Management II, 2002, pp. 168–179.
[19] F. Tu, K. Pattipati, S. Deb, V. Malepati, Computationally efficient algorithms for multiple fault diagnosis in large graph-based systems, IEEE Transactions on Systems, Man and Cybernetics, Part A 33 (1) (2003) 73–85.
[20] D. Donoho, Compressed sensing, IEEE Transactions on Information Theory 52 (4) (2006) 1289–1306.
[21] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society, Series B (Methodological) 58 (1) (1996) 267–288.
[22] J. Tropp, Just relax: convex programming methods for identifying sparse signals in noise, IEEE Transactions on Information Theory 52 (3) (2006) 1030–1051.
[23] J. Feldman, D. Karger, M. Wainwright, LP decoding, in: Proceedings of the 41st Allerton Conference on Communications, Control, and Computing, Monticello, Illinois, USA, 2003, pp. 1–3.
[24] M. Lobo, M. Fazel, S. Boyd, Portfolio optimization with linear and fixed transaction costs, Annals of Operations Research 152 (1) (2007) 341–365.
[25] A. Hassibi, J. How, S. Boyd, Low-authority controller design via convex optimization, in: Proceedings of the IEEE Conference on Decision and Control, vol. 1, 1998.
[26] L. Vandenberghe, S. Boyd, A. E. Gamal, Optimal wire and transistor sizing for circuits with non-tree topology, in: Proceedings of the 1997 IEEE/ACM International Conference on Computer Aided Design, 1997, pp. 252–259.
[27] S. Joshi, S. Boyd, Sensor selection via convex optimization, in: IEEE Transactions on Signal Processing, available from ⟨www.stanford.edu/∼boyd/sensor_selection.html⟩ 2008, to appear.
[28] A. Zymnis, S. Boyd, D. Gorinevsky, Mixed state estimation for a linear gaussian markov model, in: Proceedings of the IEEE Conference on Decision and Control, 2008, to appear.
[29] P. Tan, L. Rasmussen, The application of semidefinite programming for detection in CDMA, IEEE Journal on Selected Areas in Communications 19 (8) (2001) 1442–1449.
[30] M. Abdi, H. Nahas, A. Jard, E. Moulines, Semidefinite positive relaxation of the maximum-likelihood criterion applied to multiuser detection in a CDMA context, IEEE Signal Processing Letters 9 (6) (2002) 165–167.
[31] M. Kisialiou, Z. Q. Luo, Performance analysis of quasi-maximum-likelihood detector based on semi-definite programming, in: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 3, Philadelphia, PA, 2005.
[32] P.H. Tan, L.K. Rasmussen, Multiuser detection in CDMA—a comparison of relaxations, exact, and heuristic search methods, IEEE Transactions on Wireless Communications 3 (5) (2004) 1802–1809.
[33] J. Jaldén, C. Martin, B. Ottersten, Semidefinite programming for detection in linear systems—optimality conditions and space-time decoding, in: IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003.
[34] J. Jaldén, B. Ottersten, W.-K. Ma, Reducing the average complexity of ML detection using semidefinite relaxation, in: Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.
[35] J. Jaldén, B. Ottersten, The diversity order of the semidefinite relaxation detector, IEEE Transactions on Information Theory 54 (2008) 1406–1422.
[36] A. Montanari, D. Tse, Analysis of belief propagation for nonlinear problems: The example of CDMA, in: Proceedings of the IEEE Information Theory Workshop, Punta del Este, Uruguay, 2006.
[37] A. Montanari, B. Prabhakar, D. Tse, Belief propagation based multi–user detection, in: Proceedings 43th Allerton Conference on Communications, Control and Computing, Monticello, IL, USA, 2005.
[38] D. Bickson, O. Shental, P. H. Siegel, J. K. Wolf, D. Dolev, Gaussian belief propagation based multiuser detection, in: IEEE International Symposium on Information Theory, Toronto, Canada, July 2008.
[39] B. L. Ng, J. Evans, S. Hanly, Distributed downlink beamforming in cellular networks, in: IEEE International Symposium on Information Theory (ISIT), Nice, France, June 2007.
[40] F. Kschischang, B. Frey, H.-A. Loeliger, Factor graphs and the sum-product algorithm, IEEE Transactions on Information Theory 47 (2) (2001) 498–519.
[41] J. Yedidia, W. Freeman, Y. Weiss, Constructing free-energy approximations and generalized belief propagation algorithms, IEEE Transactions on Information Theory 51 (7) (2005) 2282–2312.
[42] S. Boyd, L. Vandenberghe, Convex Optimization, Cambridge University Press, 2004.
[43] E. Lawler, D. Wood, Branch-and-bound methods: a survey, Operations Research 14 (1966) 699–719.
[44] R. Moore, Global optimization to prescribed accuracy, Computers and Mathematics with Applications 21 (6–7) (1991) 25–39.
[45] J. Nocedal, S. Wright, Numerical Optimization, Springer, Berlin, 1999.
[46] M. Grant, S. Boyd, Y. Ye, CVX Version 1.1. Matlab Software for Disciplined Convex Programming, available from ⟨www.stanford.edu/∼boyd/cvx/⟩, 2007.
[47] K. Toh, R. Tütüncü, M. Todd, SDPT3 Version 4.0. A Matlab software for semidefinite-quadratic-linear programming, available from ⟨www.math.nus.edu.sg/∼mattohkc/sdpt3.html⟩, 2006.
[48] Y. Nesterov, H. Wolkowicz, Y. Ye, Semidefinite programming relaxations of nonconvex quadratic optimization, in: Handbook on Semidefinite Programming, Kluwer Academic Publishers, 2000, pp. 363–419 (Chapter 13).
[49] S. Boyd, B. Wegbreit, Fast computation of optimal contact forces, IEEE Transactions on Robotics 23 (6) (2007) 1117–1132.
[50] Y. Wang, S. Boyd, Fast model predictive control using online optimization, in: Proceedings of IFAC World Congress, 2008, pp. 6974–6997.
[51] Y. Nesterov, A. Nemirovskii, Interior-Point Polynomial Algorithms in Convex Programming, in: SIAM Studies in Applied Mathematics, SIAM, Philadelphia, PA, 1994.