A Minimax Approach to Sensor Fusion for Intrusion Detection

Matthew Pugh
Sandia National Laboratories
Livermore, CA 94550, USA
Email: mopugh@sandia.gov

***Abstract*—The goal of sensor fusion is to combine the information obtained by various sensors to make better decisions. By better, it is meant that the sensor fusion algorithm provides, for example, better detectability or lower false alarm rates compared to decisions based upon a single sensor. This work is motivated by combining the data gathered by multiple passive infrared (PIR) sensors to detect intrusions into a room. Optimal decision theoretic approaches typically include statistical models for both the background (non-event) data, and intrusion (event) data. Concurrent work by the author has shown that by appropriately processing multiple PIR data streams, a statistic can be computed which has a known distribution on the background data. If the distribution of the statistic during an event is known, optimal decision procedures could be derived to perform sensor fusion. It is shown, however, that it is difficult to statistically model the event data. This paper thus focuses on using minimax theory to derive the worst-case event distribution for minimizing Bayes risk. Because of this, using the minimax distribution as a surrogate for the unknown true distribution of the event data provides a lower bound on risk performance. The minimax formulation is very general and will be used to consider loss functions, the probability of intrusions events and consider non-binary decisions.**

I. Introduction

The best way to combine information from multiple sensors and make optimal decisions is an active area of research. Much work in the area of sensor fusion has shown that by processing many sensors as opposed to a single sensor, better performance can be achieved such as improved detectability, resolution, false alarm rates, etc. See for example [1], [2]. This work is motivated by the problem of using multiple PIR sensors in a room to detect an intruder. The primary difficulty in approaching this problem is how to differentiate an intrusion event from the background, or non-event. In an accompanying work [3], it is shown that by appropriately processing the PIR signals, a statistic on the background data can be computed whose distribution is well known. If the distribution of the statistic computed on event data were known, using both the event and non-event distributions with classical detection theory (see [4]) allows one to construct a receiver operating characteristic (ROC) curve to optimally trade-off detectability and false alarm rate in the binary decision problem.

SAND Number 2014-18933C. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

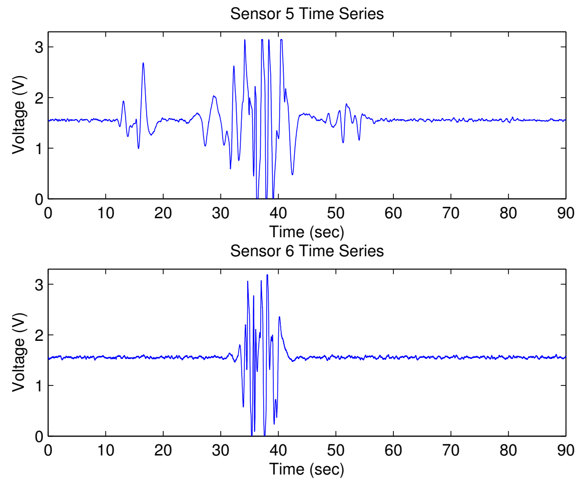

Fig. 1. Intruder Event as Observed By Two Different Sensors.

Unfortunately, the event distribution, as discussed in [3], is unknown, and the above ROC curve approach to binary detection cannot be applied. Figure 1 shows the voltage time series from two PIR sensors in our test bed during an intrusion event. It is observed that even though both sensors observe the same intrusion event, the signals look quite different. This is most likely due to the spatial distance separating the sensors and the distance and observation angle to the person entering the room. Because accurately modeling the event distribution is challenging, a different approach will be considered in this work.

This paper considers the problem of detecting a binary event, i.e. either an intrusion or non-intrusion (background) event, but an arbitrary number of decisions are possible. For example, in Table I, there are only two possible truth states,

Truth/Decision	Non-Event	Uncertain	Event
Non-Event	Correct Decision	?	Missed Detection
Event	False Positive	?	Correct Decision

TABLE I
Possible Outcomes of Binary Decision

event or non-event, but there are three possible decisions: declare an event, non-event or uncertainty. In the event that uncertainty is declared, a different action could be taken. For example, in the case of uncertainty, perhaps the system could begin using a more power intensive sensor to try to determine the true state of the room. In this toy example, power is saved by leaving the power intensive sensor in a default off state until its help is required. This example demonstrates a ternary decision problem, but in this work, an arbitrary number of decisions can be considered.

Additionally, this work considers a cost associated with each truth/decision pair (e.g. each element in Table I) and a prior probability of occurrence for each truth state to make the problem as general as possible. For example the cost associated with a missed detection can differ from the cost associated with a false alarm. Also, the probability of an event is most likely much lower than the probability of no event. A Bayes risk formulation is used which accounts for the costs associated with each truth/decision pair and the prior probability of the truth states. With the multi-decision Bayes risk defined, and assuming the background distribution found in [3], an optimization problem can be formulated to find the worst-case event distribution – the minimax distribution (see [5], [6]). Here, worst-case event distribution means if there is an adversary whose goal is to maximize the Bayes risk, the adversary would choose the minimax distribution. Designing the detection system based on the worst-case event distribution thus provides a lower bound on the Bayes risk performance of the system.

II. Data Collection and Background Statistics

A. Data Collection

The data used to characterize the background statistics (described in more detail in Section II-B) is collected in a rectangular shielded room. The shielded room is instrumented with eight custom designed sensor modules which are mounted along the walls. Each sensor module contains a Kionix KXRB5-2050 tri-axis accelerometer, a Marktech 5052TD photo diode and a Panasonic AMN24112 PIR sensor. This paper will only consider the PIR channel on each module. Each PIR sensor has a Fresnel lens. The sensor modules connect to each other via a CAN bus, and one module designated as the "mother node" connects to a laptop via USB to record the data.

The PIR voltage signal for each module lies between 0 and 3.3 volts. The raw analog voltage signal is uniformly scalar quantized with a 12-bit quantizer at a rate of 100 Hz. As mentioned in [3], lower quantization rates were considered but yielded poor performance in terms of characterizing the background statistics.

Background data is collected over the course of several days in the closed and empty shielded room. Event data is collected for simple intrusions such as a person opening the shielded room door, walking into and about the room, exiting the room and closing the door. The time series in Figure 1 is captured from such an intrusion event. More effort is expended in gathering background data to accurately model the background statistics rather than model the event statistics, which as previously mentioned is difficult. In lieu of a model, the objective of this work is to compute the worst-case event distribution for the computed statistic, described in the next section, from a Bayes risk perspective.

B. Background Statistics

This subsection describes the results found in [3], which are used to compute a statistic on the background data with a known probability distribution. In [3], a frequency domain approach is taken. Each block of 128 quantized voltage samples is converted with the FFT into the frequency domain. The real and imaginary parts of each of the FFT coefficients appears to be Gaussian. Because the PIR signal is real, the 128 point FFT has 65 unique complex coefficients – the first and middle (65^{th}) coefficients are purely real. The real and imaginary parts of the unique coefficients are stacked to create a vector $\hat{\mathbf{x}} \in \mathbb{R}^{128}$. During a training phase (or in this case on the collected test data), sensor i computes the mean vector $\mu_i \in \mathbb{R}^{128}$ and covariance matrix $\Sigma_i \in \mathbb{R}^{128\times128}$ of $\hat{\mathbf{x}}$. Using Principal Component Analysis (PCA) (see [7]), the subspace for each sensor that contains 95% of the variance is found. Let β_i be the dimension of this PCA subspace for the i^{th} sensor. The mean and covariance in this subspace, denoted $\mu_{i_{PCA}}$ and $\Sigma_{i_{PCA}}$, is then computed for each sensor.

For this work it is assumed that the statistics are stationary over the duration of the experiment (in reality the background statistics appear to drift slowly over time, but this issue will not be addressed here). Thus, with the learned values of μ_i,Σ_i,$\mu_{i_{PCA}}$ and $\Sigma_{i_{PCA}}$, the background data is no longer needed. For each sensor, each new block of 128 time series samples is converted to the frequency domain and the 128 unique coefficients are extracted as $\hat{\mathbf{x}}_i$. The coefficients $\hat{\mathbf{x}}_i$ are then projected into the PCA subspace mentioned above, producing $\hat{\mathbf{x}}_{i_{PCA}} \in \mathbb{R}^{\beta_i}$. Each $\hat{\mathbf{x}}_{i_{PCA}}$ is used to compute D_{M_i}, the Mahalanobis distance (see [7]) at sensor i, defined as follows:

$$D_{M_i}\left(\hat{\mathbf{x}}_{i_{PCA}}\right) = \left(\hat{\mathbf{x}}_{i_{PCA}} - \mu_{i_{PCA}}\right)^T \Sigma_{i_{PCA}}^{-1} \left(\hat{\mathbf{x}}_{i_{PCA}} - \mu_{i_{PCA}}\right),$$

where the superscript T denotes transpose. With D_{M_i} computed for each sensor, at some centralized controller, the summary statistic $D_{M_{total}} = \sum_{i=1}^{N} D_{M_i}$ is calculated, where N is the number of sensors. The key result in [3] is that $D_{M_{total}}$ is a chi-squared random variable with $\beta_{total} = \sum_{i=1}^{N} \beta_i$ degrees of freedom. For the remainder of this paper, we will assume that $D_{M_{total}}$ is computed at a central controller for each 128 block of time series samples and that the background statistics are chi-squared distributed with β_{total} degrees of freedom. If the distribution of $D_{M_{total}}$ during an event were known, optimal decision procedures could be derived. Because the distribution of $D_{M_{total}}$ during an event is unknown, the goal of this work is to determine the worst-case distribution for β_{total} during an event to maximize the Bayes risk (defined in Section III), which can then be used to compute lower bounds on system performance. Note that the sensor fusion aspect of the problem is in the computation of the statistic $D_{M_{total}}$ which incorporates the individual statistics, D_{M_i}, computed at each individual sensor to make a decision.

Aside: The above procedure may look computationally complex. With the assumption of stationarity of the background statistics, all of the key components μ_i,Σ_i,$\mu_{i_{PCA}}$ and $\Sigma_{i_{PCA}}$ can be computed offline. The only computations required in real-time are an FFT, the computation of the quadratic form D_{M_i} and finally summing the values of D_{M_i}.

III. Bayes Risk

This section describes Bayes risk and how it relates to the sensor fusion problem under consideration. The discussion of Bayes risk is motivated by the discussion in [7]. Let $\omega_j, j \in \{1, 2\}$ be the true state of nature, where ω_1 denotes there is no event and ω_2 means there is an event. To ease notation, equivalently $\omega_1 = \omega_{\neg E}$ and $\omega_2 = \omega_E$ where E stands for "event." This work focuses on the binary case where there are only two states of nature, but this can be easily extended to an arbitrary number of states. Let $\{\alpha_1, \ldots, \alpha_a\}$ denote the a possible decisions or actions of the sensor fusion algorithm. For example, in the binary decision problem $a = 2$, and in the example mentioned in Section I where there are three possible decisions, the third being deciding "uncertainty", $a = 3$. Let $\lambda(\alpha_i|\omega_j)$ be the loss function, or cost, associated with taking action α_i when the true state of nature is ω_j. For example, in the binary decision problem let α_1 be the action of declaring ω_1 as the true state of nature and α_2 be the action of declaring ω_2 the true state of nature. Then $\lambda(\alpha_2|\omega_1)$ is the cost of a false alarm. Lastly, let $p(\omega_j|x)$ be the posterior distribution of the true state of nature given an observation x. To simplify notation, in this paper x is the observed statistic $D_{M_{total}}$ computed every 128 time samples.

Having defined the aforementioned quantities, the conditional risk is defined as

$$R(\alpha_i|x) = \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j)p(\omega_j|x). \tag{1}$$

The above equation quantifies the risk in making decision α_i given that x is observed. The overall expected loss is then given as

$$R = \int R(\alpha(x)|x)p(x)dx \tag{2}$$

where $\alpha(x)$ is the action taken given that x is observed and $p(x)$ is the probability of x being observed. The function $\alpha(x)$ is to be designed such that the risk defined in Equation 2 is minimized. Because $p(x)$ is always non-negative, Equation 2 is minimized by minimizing the conditional risk given in Equation 1 for each value of x. Thus, $\alpha(x)$ is given by solving the following minimization problem:

$$\begin{aligned}\alpha(x) &= \arg\min_{\alpha_i} R(\alpha_i|x) \\ &= \arg\min_{\alpha_i} \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j)p(\omega_j|x).\end{aligned} \tag{3}$$

This decision procedure is called Bayes' decision rule, and the overall minimum risk achieved in Equation 2 by using this decision rule is called the Bayes risk, denoted as R^*.

In order to compute the Bayes decision rule in Equation 3, the posterior distribution $p(\omega_j|x)$ must be computed. Using Bayes' rule, the posterior can be rewritten as

$$p(\omega_j|x) = \frac{p(x|\omega_j)p(\omega_j)}{p(x)} \tag{4}$$

where the normalization term $p(x)$ is given by

$$p(x) = \sum_{j=1}^{m} = p(x|\omega_j)p(\omega_j), \tag{5}$$

where m is the number of states of nature. Bayes' rule coverts the computation of the posterior $p(\omega_j|x)$ into a computation involving the likelihood $p(x|\omega_j)$ and the prior distribution of the true states of nature $p(\omega_j)$. It is assumed that the prior probabilities $p(\omega_j)$ are assigned by a subject matter expert. Because only binary states are being considered (event and non-event) $p(\omega_E) = 1 - p(\omega_{\neg E})$, so only a single probability needs to be defined – either the probability of an event or the probability of a non-event. Additionally, a subject matter expert is required to assign the values to the loss function $\lambda(\alpha_i|\omega_j)$, e.g. the cost of false alarms, missed detections, etc. With the assumption that the prior $p(\omega_j)$ and loss function $\lambda(\alpha_i|\omega_j)$ are known, the remaining unknowns are the likelihood functions $p(x|\omega_j)$. From our discussion in Section II-B, it is assumed that $p(x|\omega_{\neg E})$ has a chi-squared distribution with β_{total} degrees of freedom. Thus, the only remaining unknown to solve for the Bayes decision rule is the likelihood $p(x|\omega_E)$, the unknown distribution of $D_{M_{total}}$ during an event. The minimax distribution found as the solution to an optimization problem described in the next section will be used as a surrogate for the unknown true likelihood.

IV. Minimax Solution for the Likelihood

The likelihood function given an event, $p(x|\omega_E)$ is unknown, so it can be optimized. Here the likelihood will be the minimax distribution, that is the distribution chosen by an adversary to maximize the risk from Equation 2, i.e. to minimize the performance of the system. While it is incredibly unlikely that the true likelihood $p(x|\omega_E)$ is the minimax distribution, it allows lower bounds on the performance of the system to be derived since it is the most difficult distribution. From a game-theoretic perspective, the minimax distribution is a saddle point so that any other likelihood function yields better performance.

Much of the minimax problem formulation will follow that described in [5]. Because a convex optimization program will be used to solve for the minimax distribution, the probability distribution must be discretized rather than be defined on the real line. Let N be the length of the support and $\mathbf{x} \in \mathbb{R}^N$ be a vector whose components are the locations of probability mass. All of the probabilities, likelihoods and posteriors will be defined with respect to this common support $\mathbf{x}$. Because of this discretization, some modifications are required. In the continuous case, $p(x|\omega_{\neg E})$ is chi-squared, but the discretized version will be defined as:

$$p(x_i|\omega_{\neg E}) = F_{\chi^2}(x_i) - F_{\chi^2}(x_{i-1}) \tag{6}$$

where $F_{\chi^2}(\cdot)$ is the CDF of the chi-squared distribution with β_{total} degrees of freedom. Thus this discretization assigns to x_i probability mass equal to the the probability of the chi-squared random variable $X \in (x_{i-1}, x_i]$. In Equation 6, x_0 is defined as $-\infty$ so that $F_{\chi^2}(x_0) = 0$. This choice of discretization is chosen for ease and other methods may be used. The discretization also effects the evaluation of risk as defined in Equation 2. Now, rather than being an integral, the risk will be a summation as defined by

$$R = \sum_{i}^{N} R(\alpha(x_i)|x_i)p(x_i). \tag{7}$$

For notation, whenever a subscript such as x_i is used, the value is discrete and refers to the i^{th} element of the support $\mathbf{x}$.

Having explained the discretization procedure, define $P \in \mathbb{R}^{N\times m}$ as the matrix whose $(k,j)^{th}$ element is the probability that the observation random variable X takes the value x_k given the true state of nature is w_j, i.e.

$$p_{k,j} = \Pr\left[X = x_k|\omega_j\right]. \tag{8}$$

Recall that the observation x is the metric $D_{M_{total}}$ is a continuous random variable but will be discretized to x_i if $x \in (x_{i-1}, x_i]$. Since there are only two states of nature and the length of the support is N, $P \in \mathbb{R}^{N\times 2}$. Also recall from the beginning of Section III that there are a possible action $\{\alpha_1, \ldots, \alpha_a\}$. Define the decision matrix $T \in \mathbb{R}^{a\times N}$ as the matrix having elements

$$t_{i,k} = \Pr\left[\alpha_i|X = x_k\right],$$

i.e. the probability of taking action α_i given that x_k is observed. Note that the action that is taken can be randomized. If the entries of T are either 0 or 1, then the decision is deterministic. Finally, define the detection probability matrix $D = TP \in \mathbb{R}^{a\times 2}$ where each element can be interpreted as

$$D_{i,j} = \Pr\left[\alpha_i|\omega_j\right],$$

i.e. the probability of taking action α_i given the true state of nature ω_j.

Recall from Section III the loss $\lambda(\alpha_i|\omega_j)$ and define the loss matrix $\Lambda \in \mathbb{R}^{a\times 2}$ as the matrix with the following elements:

$$\Lambda_{i,j} = \lambda(\alpha_i|\omega_j).$$

With the preceding definition, the risk can be rewritten as

$$R = \mathbf{1}^T\left((\Lambda \cdot \text{diag}(p)) \circ (TP)\right)\mathbf{1}, \tag{9}$$

where $\text{diag}(p) \in \mathbb{R}^{2\times 2}$ is a diagonal matrix with diagonal elements from the prior distribution $p(\omega_i)$, $\mathbf{1} \in \mathbb{R}^a$ is a vector of ones and $\circ$ denotes Hadamard (element-wise) multiplication.

All of these definition allow for the minimax optimization problem to be easily stated in terms of matrix operations. In the minimax optimization problem, there are two optimization variables – the decision matrix $T \in \mathbb{R}^{a\times N}$ and the event probability distribution $\mathbf{p} \in \mathbb{R}^N$. Because the non-event state is indexed as ω_1, the first column of the probability matrix P given in Equation 8 is the discretized chi-squared distribution and the second column is the optimization variable $\mathbf{p}$. For consistency with our observed event data that was collected, a mean constraint μ_{event} will be placed on the distribution $\mathbf{p}$, that is the event distribution will be forced to have a mean equal to that observed on the collected event data. More will be said about possible constraints later. The minimax optimization problem can be written as

$$\begin{array}{ll} \min\limits_{T\in\mathbb{R}^{p\times N}} \max\limits_{\mathbf{p}\in\mathbb{R}^N} & \mathbf{1}^T\left((\Lambda \cdot \text{diag}(p)) \circ (TP)\right)\mathbf{1} \\ \text{subject to} & \mathbf{p}^T\mathbf{x} = \mu_{\text{event}} \\ & \mathbf{p}^T\mathbf{1} = 1 \\ & \mathbf{p} \geq 0 \\ & T \geq 0 \\ & \mathbf{1}^T T = \mathbf{1}^T \end{array}. \tag{10}$$

The objective function is the reformulated Bayes risk from Equation 9, the first constraint is the imposed mean constraint, the next two inequalities ensure that $\mathbf{p}$ is a valid probability distribution and the last two constraints guarantee that the columns of the decision matrix T are valid probabilities, e.g. have non-negative values and sum to one. The statement of the optimization problem gives a nice game theoretic interpretation to the minimax problem. An adversary is trying to choose an event distribution $\mathbf{p}$ to maximize the Bayes risk and the algorithm designer is trying to choose a decision matrix T to minimize the risk. The optimization problem as stated in Equation 10 cannot be directly plugged into a solver. Further manipulations are required to get it into the proper form. The goal is to be able to write the optimization problem as a single minimization problem, i.e. convert the maximization problem into a minimization problem. This can be achieved by utilizing the duality theory of linear programming ([5], [6]).

First, the risk can again be rewritten as

$$\begin{aligned} &\mathbf{1}^T\left((\Lambda \cdot \text{diag}(p)) \circ (TP)\right)\mathbf{1} \\ &= \text{trace}\left((\Lambda \cdot \text{diag}(p))^T TP\right) \\ &= \mathbf{1}^T\left(T^T\left(\Lambda \cdot \text{diag}(p)\right) \circ P\right)\mathbf{1} \\ &= \text{vec}\left(T^T\left(\Lambda \cdot \text{diag}(p)\right)\right)^T \text{vec}(P) \\ &= \text{vec}\left(T^T\left(\Lambda \cdot \text{diag}(p)\right)\right)^T \begin{bmatrix} \mathbf{p}_{\text{non-event}} \\ \mathbf{p} \end{bmatrix} \\ &= \text{vec}\left(T^T\left(\Lambda \cdot \text{diag}(p)\right)\right)^T \tilde{\mathbf{p}} \end{aligned} \tag{11}$$

where we have defined $\tilde{\mathbf{p}} = \begin{bmatrix} \mathbf{p}_{\text{non-event}} \\ \mathbf{p} \end{bmatrix}$ and $\mathbf{p}_{\text{non-event}}$ is the discretized chi-squared distribution. Introduce the optimization variable $\hat{\mathbf{p}} \in \mathbb{R}^{2N}$. This variable differs from the optimization variable $\mathbf{p}$ in Equation 10 because $\mathbf{p}$ represented the unknown event distribution but $\hat{\mathbf{p}}$ will have both the non-event and event distribution unknown. The knowledge of the non-event distribution will be incorporated as an equality constraint in the reformulated optimization problem. Next, let

$\mathbf{c} = \text{vec}\left(T^T \cdot (\Lambda \cdot \text{diag}(p))\right)$ and let $A \in \mathbb{R}^{(N+2)\times 2N}$ matrix defined as

$$A = \left[\begin{array}{c:c} \mathbf{I}_N & \mathbf{0}_N \\ \hdashline \mathbf{0}_{1\times N} & \mathbf{x} \\ \hdashline \mathbf{0}_{1\times N} & \mathbf{1}_{1\times N} \end{array}\right]$$

where $\mathbf{I}_N \in \mathbb{R}^{N\times N}$ is the identity matrix, $\mathbf{0}_N \in \mathbb{R}^{N\times N}$ is a matrix of zeros and $\mathbf{0}_{1\times N}$ and $\mathbf{1}_{1\times N}$ are row vectors of zeros and ones respectively. Lastly, the constraint values in Equation 10 are captured by $\mathbf{b}$, the constraint vector, defined as

$$\mathbf{b} = \begin{bmatrix} \mathbf{p}_{\text{non-event}} \\ \mu_{\text{event}} \\ 1 \end{bmatrix} \in \mathbb{R}^{N+2} \tag{12}$$

The inclusion of $\mathbf{p}_{\text{non-event}}$ incorporates our knowledge of the background distribution. With the previous definitions and using the equational form of a linear program (see [6]) or standard form linear program (see [5]), the maximization problem in 10 can be written as

$$\begin{array}{ll} \max\limits_{\hat{\mathbf{p}}\in\mathbb{R}^{2N}} & \mathbf{c}^T\hat{\mathbf{p}} \\ \text{subject to} & A\hat{\mathbf{p}} = \mathbf{b} \\ & \hat{\mathbf{p}} \geq 0 \end{array} . \tag{13}$$

Introducing the dual variable $\mathbf{y} \in \mathbb{R}^{N+2}$, the dual problem to the maximization problem in Equation 13 can be written as

$$\begin{array}{ll} \min\limits_{\hat{\mathbf{y}}\in\mathbb{R}^{N+2}} & \mathbf{b}^T\mathbf{y} \\ \text{subject to} & A^T\mathbf{y} - \mathbf{c} \geq 0 \\ & \mathbf{y} \geq 0 \end{array} . \tag{14}$$

Putting everything together, the optimization problem in Equation 10 can be rewritten as a single minimization problem which can be plugged into a solver:

$$\begin{array}{ll} \min\limits_{T\in\mathbb{R}^{p\times n},\hat{\mathbf{y}}\in\mathbb{R}^{N+2}} & \mathbf{b}^T\mathbf{y} \\ \text{subject to} & A^T\mathbf{y} - \mathbf{c} \geq 0 \\ & T \geq 0 \\ & \mathbf{1}_{2\times 1}^T T = \mathbf{1}_{N\times 1}^T. \end{array} \tag{15}$$

The optimization problems in Equations 10 and 15 provides a template for the types of constraints that can be placed on the minimax distribution. For example, moment constraints on the minimax distribution can be easily handled by adding the constraint $\mathbf{p}^T\mathbf{x}^n = M_n$, where $\mathbf{x}^n$ means raising each element of $\mathbf{x}$ to the n^{th} power and M_n is the n^{th} moment constraint. Support constraints can be added to restrict where the minimax distribution has probability mass by zeroing out columns in the A. Additionally, rather than have equality constraints on μ_{event} or the known background distribution, box constraints can be used to constrain the values to an interval. These are just a few examples of possible constraints. Indeed, any constraint that is linear in $\hat{\mathbf{p}}$ can easily be incorporated.

V. Results

Having formulated the optimization problem (which is a linear program) in Equation 15, the minimax distribution can be solved by using a convex solver. The results in this section were generated in MATLAB using YALMIP [8] as the parser and Mosek [9] as the solver. Other solvers were attempted, such as SDPT3 [10] and SeDuMi [11], encountered numerical issues.

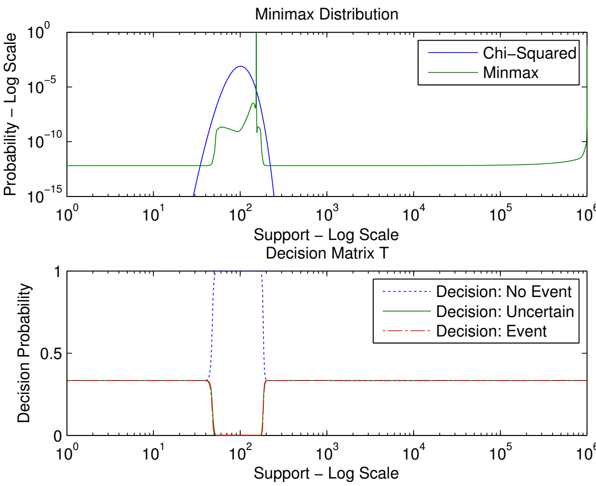

Fig. 2. Top: Known χ^2 background distribution and minimax distribution. Bottom: Decision probabilities versus observations.

The formulation in the previous section is very general and to solve for the minimax distribution, values have to be assigned to the problem parameters. The values used for the results to shown will now be discussed. There will be two states of nature ($m = 2$), either an event or no event. There will be three actions ($a = 3$), either declaring an event, declaring a non-event, or declaring uncertainty. The support $\mathbf{x}$ needs to be defined and actually plays an important role in the solution to the optimization problem. Ideally, the more support points the better, as then the discrete distributions better approximate the continuous distributions. However, the more points that are used, the more optimization variables and thus the more computational power is needed. The support $\mathbf{x}$ that is considered will have 5×10^4 points logarithmically spaced on $[1, 10^6]$. The empirical event mean was set to $\mu_{\text{event}} = 6.67 \times 10^4$, which is the mean measured on the test data. The prior probability of an event is set to $p(\text{event}) = 10^{-7} = 1 - p(\text{no event})$, i.e. events are very rare. Lastly, the loss matrix used is

$$\Lambda = \begin{bmatrix} -100 & 1000 \\ 50 & -500 \\ 100 & -1000 \end{bmatrix},$$

where the first column corresponds to "no event", the second column to an "event", first row for declaring no "event", second row for declaring "undecided" and the third row for declaring an "event." The risk is to be minimized, so negative numbers can be viewed as a reward.

The top plot in Figure 2 shows the known χ^2 background distribution for the test data which has 101 degrees of freedom and the minimax distribution which is the solution to the optimization problem. Because a χ^2 distribution has a well known density, the probability mass becomes astronomically small over most of the support as observed in the figure.

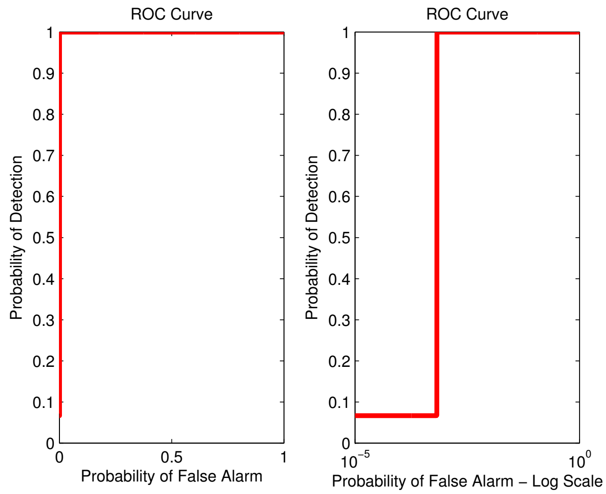

Fig. 3. Left: ROC Curve assuming the minimax distribution as the event distribution on a linear scale. Right: Same but with a logarithmic scale.

Notice that the minimax distribution tries to place most of it's mass in the same area as the background distribution so as to try to produce maximum confusion. There is also some probability mass at the right end of the support so that the mean constraint is met. The bottom plot in Figure 2 shows the rows of the decision matrix T which represent that probability of taking an action for a given observed value of the support x. Notice that for "typical" values of the background distribution, the algorithm declares a non-event with probability one. For most of the support where both the background and minimax probabilities have low probability, the algorithm essentially picks one of the three decisions (event, non-event, uncertainty) with equal probability.

As mentioned in the Section I, traditional decision theoretic analysis is based on knowledge of the distributions of all the events (in this case event and non-event) of interest. In an accompanying work [3], a threshold was chosen based solely on the non-event distribution since the event distribution is unknown. Revisiting the binary hypothesis problem, one can use the minimax distribution, which is the worst case distribution, as a proxy event distribution and compute classical metrics such as a ROC curve. Figure 3 shows the ROC curve using the minimax distribution computed in the considered problem. This is an example of what can be done, but in this case the results should be taken with a grain of salt. The problem considered is fundamentally a ternary decision problem so the minimax distribution found would not be the same minimax distribution found by solving a binary decision problem.Regardless, it may be useful to use the minimax approach to use the minimax distribution as a surrogate for the unknown event distribution to perform traditional hypothesis testing analyses such as a ROC curve.

VI. Conclusion

This work considers how to generate a surrogate distribution for the unknown event distribution so that a risk based analysis can be undertaken and decision algorithm can be designed. Specifically, the samples from PIR sensors are processed in a way developed in [3] to produce a metric with known distribution when there is no event. A binary state of nature is analyzed where each observed value of the metric was either generated by an event or the background. Although the state of nature was fixed to be binary, the framework is sufficiently general to allow for any number of states. A framework is developed such that an arbitrary number of decisions or actions a can be taken depending on the observed value of the metric. An expert is assumed to assign a loss to each combination of actions and states of nature and the prior probabilities of each state of nature are assumed known or given by an expert. With this knowledge, a minimax optimization problem is constructed to find the minimax distribution for the computed metric during an event. The minimax distribution is the most difficult distribution in the sense that it maximizes the risk. Thus, using the minimax distribution for analysis provides a lower bound for the risk performance of the system.

References

[1] L. A. Klein, *Sensor and data fusion: a tool for information assessment and decision making*, vol. 324. SPIE press Bellingham, WA, 2004.

[2] J. R. Raol, *Multi-sensor data fusion with MATLAB*. CRC Press, 2010.

[3] M. Pugh, J. Kvam, and J. Brewer, "Sensor fusion for intrusion detection under false alarm constraints." Submitted to 2015 IEEE Senors Applications Symposium.

[4] S. M. Kay, *Fundamentals of Statistical signal processing, Volume 2: Detection theory*. Prentice Hall PTR, 1998.

[5] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[6] J. Matoušek and B. Gärtner, *Understanding and using linear programming*, vol. 168. Springer, 2007.

[7] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern classification*. John Wiley & Sons, 2012.

[8] J. Lofberg, "Yalmip: A toolbox for modeling and optimization in matlab," in *Computer Aided Control Systems Design, 2004 IEEE International Symposium on*, pp. 284–289, IEEE, 2004.

[9] A. Mosek, "The mosek optimization software," *Online at http://www. mosek. com*, vol. 54, 2010.

[10] K.-C. Toh, M. J. Todd, and R. H. Tütüncü, "Sdpt3a matlab software package for semidefinite programming, version 1.3," *Optimization methods and software*, vol. 11, no. 1-4, pp. 545–581, 1999.

[11] J. F. Sturm, "Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones," *Optimization methods and software*, vol. 11, no. 1-4, pp. 625–653, 1999.