Analysis of CBRN Sensor Fusion Methods

Scott Lundberg, Randy Paffenroth, and Jason Yosinski
Numerica Corporation
4850 Hahns Peak Drive, Suite 200
Loveland, CO, USA
{scott.lundberg,randy.paffenroth,jason.yosinski}@numerica.us

Abstract – *Many well-defined metrics have been developed for data fusion problems in the target tracking domain. However, much less is known about the proper evaluation of Chemical, Biological, Radiological, and Nuclear (CBRN) data fusion methods. In this paper we begin to address the issue of evaluating the performance of CBRN fusion algorithms. As a demonstration of the applicability of modified target tracking metrics to CBRN problems we apply such techniques to an illustrative data fusion method [1]. Of the metrics we have studied one stands out among the rest, namely covariance consistency, and we pay special attention to its proper application to CBRN scenarios. Covariance consistency is of particular importance since it allows us to evaluate how accurately an algorithm is reporting the uncertainty of its estimates, and we would contend that uncertainty measures are of paramount importance in CBRN scenarios.*

Keywords: Tracking, CBRN, estimation, ambiguity, sensor data fusion, covariance consistency, Cramér-Rao bound.

1 Introduction

Managing and combining many different sensors across a network has received substantial attention over the course of many years [2]. This is in part because the assessment of the type and severity of potential hazards provided to decision makers can be improved by merging measurements from multiple sensors into a common operating picture. For Chemical, Biological, Radiological, and Nuclear (CBRN) detection scenarios there are several sensors that one might consider for such fusion problems including long range instruments such as radar, infrared (IR), electro-optical, and long wave hyper spectral; short range instruments such as Raman spectrometers; and a wide array of point sensors such as ion mobility spectrometers (IMS) and chemical-resistor arrays, to name but a few. IR spectrometers [3, 4, 5], IMS [6], and Raman Spectrometers [7] have in the past been particularly useful for chemical detection.

An area of key importance in data fusion is that of *ambiguity*. There is an extensive body of literature on the proper handling of ambiguity in target tracking problems [8, 2, 9, 10, 11, 12, 13] (and references therein) as well as a similar, but substantially smaller, body of work that addresses ambiguity in chemical detection problems [14, 15, 16, 17, 18]. We note that an introduction to this material can also be found in [1], parts of which are summarized here for the reader's convenience. While an ambiguity analysis has been performed for various *components* of a system for CBRN consequence management, the authors are not aware of any studies that address the interactions between differing sources of ambiguity. Accordingly, herein we describe approaches that illustrate how data fusion algorithms that explicitly represent these interactions can be evaluated.

As the key motivator for our work we observe that the information provided to the user *must be correct*. The worst possible sin in such life and death circumstances would be to provide the user with information that is not factual. Note, there is an important and subtle distinction here. We consider providing the user with information whose veracity is unknown, or ambiguous, and *not informing them of this fact* to be as bad as providing incorrect information. The act of informing the user that a suspect cloud is composed of water vapor means very different things depending on whether one is 99.999% sure it is water vapor or one is 51% sure it is water vapor and 49% sure it is nerve gas, even though both situations lead to the same "most probable estimate".

As a framework for our discussion we consider a cloud representation scheme from [1] in Section 2. In Section 3 we systematically consider many of the performance metrics that are common in the target tracking domain and evaluate their applicability to the CBRN domain. Some of these methods will need to be significantly modified, while others are almost directly applicable with only minor modifications. Of the metrics considered in Section 3, *covariance consistency* is a metric of particular interest. Section 4 describes in more detail how covariance consistency can be applied to CBRN fusion problems. As an illustrative example we apply these methods to evaluate the performance of a fusion method described in [1]. We center our discussion on the covariance consistency metric because of the wide applicability it has been found to have in the target tracking domain. Section 5 further explores the types of estimation errors that arise in CBRN scenarios and provides directions for future work.

2 Gaussian Sum Representations

When deriving metrics for data fusion methods an important factor is the representational scheme or "tracking space" in which we fuse the measurements. For the purposes of our derivations here we will use the *Gaussian sum approach* described in [1]. While space does not allow for a full derivation of this approach, we note that the core ideas of this approach are based on the desire to use a relatively small number of parameters to characterize the distribution of a cloud of interest. A small number of parameters is advantageous in problems where the sensor data is limited and one desires to both compute the parameters of the cloud distribution as well as their uncertainties. Since the solution to the diffusion equation of a point source emitter is a Gaussian, we assume that the densities of CBRN clouds will likely exhibit an exponential decay profile much like a Gaussian distribution [1]. By using sums of Gaussian functions to represent a cloud we attempt to reduce the number of parameters we wish to estimate as much as possible, and hence to also reduce the number of measurements required to estimate the cloud state.

To avoid confusion, let us emphasize that the calculations we discuss here use Gaussian distributions in two ways. First, we use them as a *Gaussian mass density function* to represent how gas is distributed in a cloud. Of course, this Gaussian mass density must be appropriately parametrized and we happen to choose the parametrization

$$\alpha_g = [x, y, e, f, g, c], \tag{1}$$

where x and y are the center of mass of the distribution, e, f, and g give the size and shape of the distribution's elliptical level sets, and c is a scaling factor for the total mass of chemical in the cloud. The parameter c differentiates the above Gaussian mass density function from a Gaussian probability density function (PDF) by allowing the total mass of the cloud to vary.

Second, we use a *Gaussian probability density function* to represent the statistics of the parameters of our chosen model. The reader may decide whether it is a beautiful or confusing mathematical happenstance that we assume the parameters of our Gaussian mass density are described by a Gaussian probability density function. We will endeavor to carefully distinguish between the two uses in what follows.

Given the model stated above, one can define a measurement function h_g as

$$h_g(\alpha_g, x_i, y_i) = c\frac{1}{2\pi\sqrt{\|E\|}}\exp(-\frac{1}{2}\left(\bar{x}-\bar{x_i}\right)^T E^{-1}(\bar{x}-\bar{x_i})) \tag{2}$$

where $\bar{x} = \begin{pmatrix} x \\ y \end{pmatrix}$, $\bar{x_i} = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$, $E = \begin{pmatrix} e & f \\ f & g \end{pmatrix}$. [1]

If we are provided with a set of measurements z_i from a collection of N sensors at positions (x_i, y_i), each equipped with a measurement (co)variance R_i, then we can define the weighted non-linear least squares problem by

$$Ls(\alpha_g, x, y, R, z) = \min_{\alpha_g} \sum_{i=1}^{N} (h_g(\alpha_g, x_i, y_i) - z_i)^T R_i^{-1} (h_g(\alpha_g, x_i, y_i) - z_i), \tag{3}$$

a solution of which by the Levenberg-Marquardt algorithm leads to an estimate of α_g as well as its associated 6 covariance matrix P_{α_g}. While Equation 3 is formulated above for the multi-variate case, that the metrics analysis which follows is valid for basis functions other than Gaussians but that Gaussian sums will be our focus herein.

The two ways in which Gaussians are being used can now be seen somewhat more clearly. First, we assume that the concentration of chemicals in the cloud follows a Gaussian mass distribution, just as predicted by the diffusion equation. Second, we assume that there are errors in the parameters that we estimate by way of taking measurements and using a weighted non-linear least squares formulation. For example, perhaps there are errors associated with the placement of its center of mass x, y. We assume that these errors are well represented by a Gaussian probability distribution, so that they are completely characterized by a mean and covariance matrix. Of course, this assumption needs to be carefully checked and that will be precisely the focus of Section 4. For a more detailed discussion of covariance matrices and their definition we refer the reader to [19, 2, 13].

[1] We admit that E looks much like a covariance matrix, but thinking in these terms can lead one astray. e, f, and g parametrize the shape of the cloud and have nothing to do with probability densities!

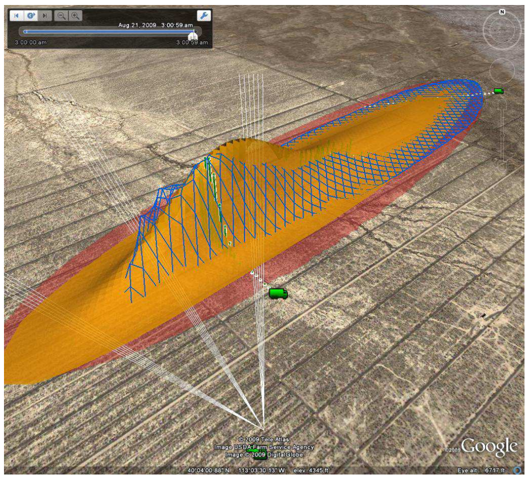

Figure 1: This is 30 minutes after a single chemical warhead filled with thickened VX nerve gas has exploded at an altitude of 1000 meters. The blue grid is the simulated true ground level concentration, while the orange region is our estimate of the current chemical cloud. Note that using two Gaussians we are able to fit the shape of the dispersion cloud quite well given the readings of both the point and stand-off sensors.

Figure 1 shows an example of using Gaussian sums combined with point and stand-off sensors to estimate the 2D concentration profile of a simulated chemical threat. The blue grid represents the true chemical concentration (as simulated), while the orange surface represents our estimate of the concentration. Note that the height of the surfaces correspond to concentration level not actual cloud height since this is a 2D model. The red region surrounding the orange surface is the region outside of which we are 95% confident that the chemical concentration is below a certain threshold. The fact that in Figure 1 the red region does not extend much beyond the edges of our orange estimate means that we are very confident in the accuracy of our estimate.

3 Metrics

Here we endeavor to find appropriate metrics for the performance evaluation of estimation algorithms in distributed CBRN sensing scenarios. In particular, we seek motivation from the target tracking domain and attempt to develop appropriate modifications of metrics from that domain to distributed CBRN sensing. One particular highlight of the current derivation is a generalization of the important idea of "covariance consistency" to the domain of CBRN detection problems. This section covers the boarder picture of metrics, while the next focuses on the details and application of covariance consistency.

As a motivation for considering algorithm performance metrics consider the benchmarking software that has been used extensively in the target tracking community to facilitate the development and assessment of algorithms. One is rarely, if ever, in possession of sufficient "real world data" to appropriately test estimation algorithms during their development, or to validate them before they are fielded. Especially in the case of algorithm development, synthetic benchmarks are extremely important both for the voluminous data they provide as well as for the ability to appropriately choose scenario complexity based upon the maturity of the algorithm.

An important attribute of most benchmarking software is that, besides providing data for testing candidate algorithms, they also often provide *metrics* for assessing the performance of algorithms. These metrics provide rigorous methods for determining the quality of the algorithms and are important both for algorithm developers and users. For algorithm developers, so they know under what rules they will be judged, as well as algorithm assessors, so they can fairly and accurately determine the quality of candidate algorithms.

Accordingly, in the current text we derive appropriate metrics for work on CBRN detection and fusion.

3.1 Current Metrics

In the target tracking literature there is a vast array of metrics that are commonly used to assess algorithms. In this section we describe a selection of such metrics and demonstrate how they map to chemical detection problems. We list metrics well known in the target tracking community and discuss how they can be applied to CBRN detection and tracking:

- *Track Completeness.* Completeness is a measure that describes how complete the common picture is, i.e., do different users at different sites have the same picture? It is computed as

  $$C_m(t) = \frac{JT_m(t)}{J_m(t)} \times 100\% \tag{4}$$

  where $JT_m(t)$ is the number of objects with at least one track assigned by participant m, and $J_m(t)$ is the total number of objects at time t.

  This metric has the same importance for CBRN detection problems as it has for target tracking problems. Just as it is desirable to have estimates for all of the targets in a scenario it is also clearly desirable to have estimates on all chemical plumes/sources.

- *Track Ambiguity.* This metric measures the degree of redundant tracks/clouds in any one track picture. It is computed as

  $$A_m(t) = \frac{NA_m(t)}{JT_m(t)} \tag{5}$$

where $NA_m(t)$ is the number of assigned, mutually uncorrelated tracks/clouds.

This metric is somewhat different in the case of CBRN fusion. As we have described in [1], a single cloud is unlikely to be well described by a single Gaussian density function, so we will likely *want* each cloud to consist of the sum of multiple densities. In this case it becomes important to define what we consider a unique "track". Our later discussion of modeling errors in Section 4.1 is related to this question.

- *ID Correctness.* This metric measures the degree to which the target IDs in the track/cloud picture are correct; it applies more to an identification function as opposed to a tracking function. It is computed as

$$IDCX_m(t) = \frac{JCX_m(t)}{JTX_m(t)} \times 100\% \tag{6}$$

where $JCX_m(t)$ is the number of tracked objects of type X with correct ID label and $JTX_m(t)$ is the true number of tracks/clouds with that label.

Just as in the target tracking domain, this metric is of key importance to the CBRN detection domain, with one small change. In the CBRN detection problem, the ID will likely be a vector of IDs. This is because a cloud may contain multiple chemicals.

- *Track Pair-wise Commonality.* Commonality is another measure that describes how good the common picture is, i.e., do the users have the same picture. It is computed as

$$PCM_q(t) = \frac{NC_{q,r}(t)}{NS_{q,r}(t)} \times 100\% \tag{7}$$

where $NC_{q,r}(t)$ is the number of assigned tracks/clouds held by platforms q and r such that: (i) each track/cloud has common, unique track/cloud number; (ii) position and time data are the same within some confidence; and (iii) the composition IDs are the same. Also, $NS_{q,r}(t)$ is the number of assigned tracks/clouds held by either q or r.

Again, the distributed chemical detection problem has need of precisely the same type of metric, with the caveat that the generalization to the chemical domain will require the ability to match two different cloud estimates to one another, and determine if they are both tracking the same material (and hence should be the same track/cloud).

- *Spurious Track Ratio.* This metric evaluates the ratio of spurious track/cloud counts to the target track/cloud count,

$$S_m(t) = \frac{N_m(t) - NA_m(t)}{N_m(t)} \times 100\% \tag{8}$$

where $NA_m(t)$ is the number of assigned, mutually uncorrelated tracks/clouds, and $N_m(t)$ is the number of tracks/clouds.

Just as for the track/cloud pair-wise commonality metric, the spurious track/cloud ratio is important in the CBRN fusion domain. However, also like track/cloud pair-wise commonality, this metric relies on a way to correlate cloud estimates and decide if they are attempting to model the same underlying object. Without this type of comparison it is not possible to determine which cloud estimates are spurious and which are valid.

- *Accuracy.* This metric computes the accuracy of the track/cloud estimates,

$$PA_{j,m} = \frac{\sqrt{\sum_{n \in D_m} \left(\alpha_e^{j,n,m} - \alpha_t^{j,n,m}\right)}}{NA_{j,m}} \tag{9}$$

where $\alpha_e^{j,n,m}$ is the estimated parametrization of track/cloud n held by participant m on object j and $\alpha_t^{j,n,m}$ is the optimal parametrization of track/cloud n held by participant m on object j (these two concepts are described more fully in Section 4.1). Also, $NA_{j,m}$ is the number of tracks/clouds on object j held by sensor m, and D_m is the set of assigned tracks/clouds for sensor m.

- *Covariance Consistency.* This metric evaluates how accurate the covariance estimate produce by the tracker in any one track picture is,

$$CTrCov_m = \frac{1}{N_e NA_m} \sum_{n \in D_m} (\alpha_e^n - \alpha_t^n)^T (P_e^n)^{-1} (\alpha_e^n - \alpha_t^n) \tag{10}$$

where α_e^n are the estimated parameters and α_t^n are the optimal parameters. N_e is the dimension of α_e^n, NA_m is the number of tracks held by sensor m, and P_e^n is the covariance matrix for track/cloud n.

The treatment and interpretation of this metric is delicate and we dedicate Section 4 to its consideration.

Many of the metrics above are directly applicable to the chemical detection problem. For example *track pair-wise commonality*, the measure of similarity of the scenario estimates at each of the nodes on the network, and *accuracy*, the measure of the accuracy of the mean of the estimate as compared to truth, require only minor modifications once a comparison method for estimates has been implemented. Of course, as we are estimating extended objects, such as chemical vapor plumes, many of the metrics require reinterpretation, such as redundant track ratio, which scores badly when a single object gives rise to multiple tracks. One metric of key importance above that requires substantial analysis in the chemical detection case is that of covariance consistency. It is this metric on which we will focus primarily, and the next section is dedicated to its analysis.

4 Covariance Consistency in Chemical Detection Problems

Covariance consistency provides a measure of the uncertainty estimates provided by the candidate algorithm. It has been our experience, as well as the experience of many others [19], that good performance in many of the other metrics are *implied* by good performance in the covariance consistency metric. How can that be? Well, in some sense, one can consider covariance consistency as a measure of *an algorithm's knowledge of its own inadequacies*. In other words, providing a covariance along with a mean allows an algorithm to advertise how much uncertainty it thinks it has in its estimate. Covariance consistency measures how often the algorithm is correct about its own uncertainty.

Algorithms that consistently provide a covariance that is too large are being pessimistic about their own uncertainty. Such algorithms cause the user to have to contend with greater uncertainty and ambiguity than is actually necessary, and clearly are not desirable. Contrariwise, algorithms that consistently provide a covariance that is too small are being optimistic about their own uncertainty. We contend that such algorithms are even more dangerous than pessimistic ones. Such algorithms provide the user with *unwarranted confidence* in their knowledge of the scenario. Such unwarranted confidence can lead to dangerously incorrect decision making.

So, it is only through good covariance consistency that an algorithm demonstrates that it understands its own uncertainty and is providing appropriate information for downstream processing. In order to demonstrate the application of covariance consistency to a CBRN fusion algorithm we first must describe the representation and estimation scheme of a particular algorithm. Accordingly, we use the representation scheme derived in Section 2.

4.1 Computing Covariance Consistency

Assume that we have estimated some set of parameters α_e (such as those just described above) for a chemical cloud, we are tracking, as well as a covariance matrix P_e for those parameters. Suppose our simulation was sufficiently simple so that there happens to exist a set of "truth" parameters α_t. Following [19] we can write down the "normalized Mahalanobis distance of the state estimation error" as

$$Mdn(\alpha_e) = (\alpha_e - \alpha_t)^T P_e^{-1} (\alpha_e - \alpha_t)/N_e \qquad (11)$$

where N_e is the dimension of α_e.

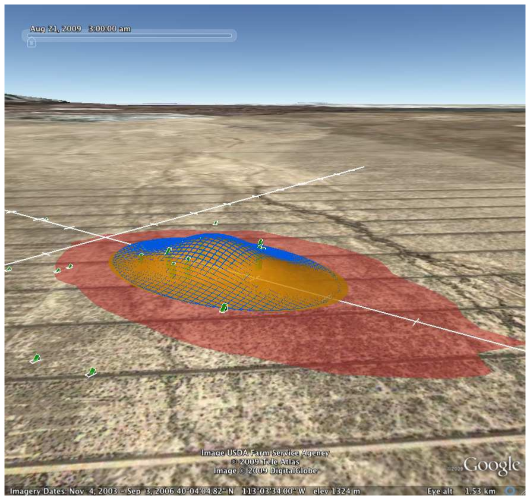
Aug 21, 2009 3:00:00 am
Image USDA Farm Service Agency
© 2009 Tele Atlas
Image © 2009 DigitalGlobe
Google
Imagery Dates: Nov 4, 2003 - Sep 3, 2006 40°04'04.82" N 113°03'34.00" W elev 1324 m
Eye alt 1.53 km

Figure 2: A sample setup used for the covariance consistency calculations. We took this scenario and performed 1000 Monte Carlo runs to generate data by which to evaluate the covariance consistency of our estimator. Note that the orange estimate and blue truth are sitting on top of multiple point chemical sensors shown by the green bars.

In order to compute covariance consistency we need to apply Equation 11 to repeated simulation results α_e and P_e from a candidate estimator. To do this we use the non-linear estimation method from [1] and apply it to a scenario with a Gaussian shaped chemical cloud sitting on top of multiple point sensors (as shown in Figure 2). The measurement function for the point sensors $h_g(\alpha_t, x_i, y_i)$ is defined as in Equation 2, where (x_i, y_i) is the location of a sensor and α_t is the parametrization of the chemical cloud in the simulator.

Given a collection of N sensor reports defined by

$$z_i^j \in N(h(\alpha_t, \beta_i), R_i) \qquad (12)$$

we can estimate α_e by solving Equation 3. The covariance matrix P_e can be computed as

$$P_e = (\nabla^2 Ls(\alpha_e, x, y, R, z))^{-1} \qquad (13)$$

from the inverse Hessian over the α parameters at the point α_e.

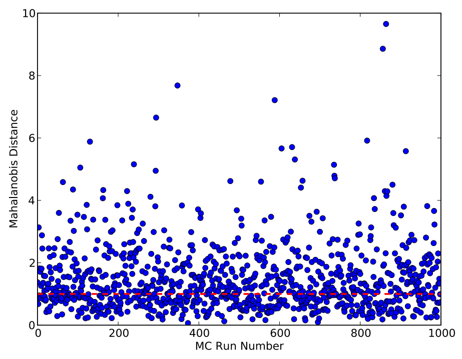

Figure 3: Normalized Mahalanobis distance values for 1000 Monte Carlo runs. The distances were computed between the parametrization of the truth and the parametrization obtained by the non-linear optimizer, scaled using the returned covariance. The red dashed line at 1.0 indicates where the center of the chi-squared distribution should fall if the estimator had perfect covariance consistency.

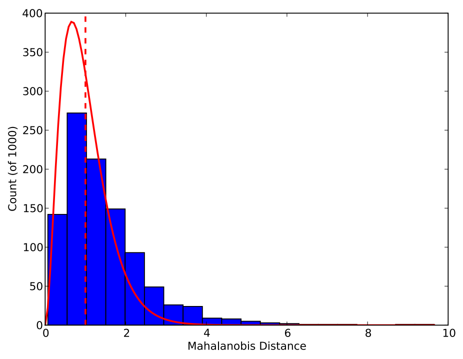

Figure 4: A histogram of the normalized Mahalanobis distance values in Figure 3, overlaid with the normalized chi-square distribution that the points would follow if their covariance consistency was perfect. The heights of the bars are normalized based on their width. The values are close but not exact, showing that the covariances returned by the estimator are nearly — but not perfectly — consistent.

Given the above approach we can generate many Monte Carlo runs and so evaluate the covariance consistency of our estimator. Figure 2 shows an example setting for our calculations. The true cloud in this scenario is a single Gaussian shaped cloud (shown as the blue mesh) centered over multiple chemical point sensors. Each of these sensors is shown by a green bar whose height corresponds to the concentration reading reported by the sensor z_i. The sensor readings are drawn from a Gaussian distribution in order to introduce uncertainty into our estimate.

Our estimate of the cloud profile is shown by the orange surface in Figure 2. This surface represents the MLE estimate of the cloud, while the red region surrounding the surface represents the area outside of which we are 95% confident that the cloud falls below a given density threshold. Because we are displaying not only our estimate, but also the uncertainty associated with that estimate it is reasonable to expect the orange estimate and the blue truth to differ. However, what is not acceptable is for the blue truth to fall outside the red uncertainty region more than 5% of the time. If that were to happen we would be lying about our uncertainty. Covariance consistency is a way to test our reported uncertainty. If we are correctly reporting our uncertainty the values coming out of Equation 11 will follow a chi-squared distribution centered at 1.0. If the distribution is shifted towards smaller values then we are being overly pessimistic. If they are shifted towards higher values then we are being overly optimistic, which as we noted earlier is a dangerous mistake.

Figure 3 shows the results of computing Equation 11 for 1000 different runs. The dashed red line at 1.0 shows where we expect the center of the chi-squared distribution to be if we satisfy covariance consistency. It is difficult to tell directly from Figure 3 how well the data fits this distribution so we have plotted in Figure 4 a histogram of the results and overlaid the expected chi-squared distribution. By looking at Figure 4 we can tell that to a large extent the results do fit the chi-squared distribution, though not perfectly. It can be seen that the area of the histogram is just to the right of the chi-squared area, meaning that in this case we are slightly over optimistic about our reported covariances. That is to be expected in this case since we are assuming linearity at the solution in order to generate our covariance matrix. These non-linear effects are important to account for, and the next section on future work presents several possible approaches to account for these and other issues that arise in more realistic settings.

5 Error Analysis and Future Work

While covariance consistency is a key metric and our results suggest that at least in certain situations it is possible to maintain covariance consistency, it is unfortunately the case that Equation 11 does not tell the whole story about errors in the chemical detection domain. In particular, one must carefully consider the "truth" parameters α_t and what they mean once we step outside the controlled simulation setting of models that have the same parametrization as the true cloud.

First, one must consider how α_t is to be computed. It is quite unlikely that α_t will be available in an actual fielded system. Accordingly, its main role is to be used within the context of a simulation environment for judging algorithms before they are fielded. Unfortunately, even in a simulation context if the true cloud and the cloud estimate each have different representations (e.g. the true cloud is not actually a Gaussian sum) then α_t is not entirely trivial to compute.

The parameter α_t is intended to represent the best possible representation of the true cloud. In other words, given a measurement function h_t, similar to H_g defined in Section 2, and a true cloud density profile denoted by $g(x, y)$, we wish to find a α_t such that

$$\alpha_t = \min_{\alpha} \|h_t(x, y, \alpha) - g(x, y)\| \tag{14}$$

is minimized. Equation 14 should be interpreted as representing the modeling error that remains even when an optimal α_t is chosen. In a similar vein, one can interpret α_e, from Equation 11 as the parametrization one achieves for a fixed set of noisy sensors, while α_t is what one can achieve with as many noiseless sensors as one desires.

The fact that Equation 14 might be large, and that α_t might not well represent the true state of the chemical cloud, is a difference between distributed chemical sensing and what one might commonly consider in target tracking. For example, consider the problem of tracking an aircraft at long range. In such a circumstance much of the pertinent information about the aircraft is well encoded by way of a position x, y, z and velocity vector $\dot{x}, \dot{y}, \dot{z}$. On the other hand it is not clear if *any low dimensional parametrization* including the Gaussian sums discussed in Section 2, can accurately represent a complicated chemical plume. So, Equation 11 only measures the estimator's ability to assess its mean and covariance accuracy against α_t and does not include an assessment of how well the optimal α_t actually models the cloud of interest.

Accordingly, for an effective set of metrics, one must also consider the errors arising from Equation 14 for a particular measurement function h.

5.0.1 Errors in $h(x, y, \alpha_t)$

In Section 2 we considered the use of sums of Gaussian densities to represent the chemical plumes of interest. Here we examine that representation in the context of its errors and convergence properties.

In particular, it is quite possible that the ℓ_2 norm of the error between our model $h(x, y, \alpha_t)$ and the true chemical distribution $g(x, y)$ over some region Ω, as defined by

$$ModelErr(\alpha_t) = \sqrt{\iint_{\Omega} (h(x, y, \alpha_t) - g(x, y))^2 \; dydx}, \tag{15}$$

is quite large.

If Equation 15 is large then even if Equation 11 is perfectly satisfied, the covariance P_h is clearly too small (or optimistic) since it does not include any information about the error in the model itself, here defined to be the ℓ_2 norm of the difference between $h(x, y, \alpha_t)$ and $g(x, y)$. Two challenges immediately present themselves.

First, how do we compute Equation 15? While $g(x, y)$ will likely be known when an algorithm runs within a simulator, it will definitely not be known in a fielded system. One possible solution comes from numerical analysis where it is a classic problem to examine the convergence rate of numerical schemes as one increases the number of parameters one admits for a particular representation [20]. Our case is no different and it is important to consider how the representation scheme in Section 2 converges to a given true chemical density. For example, consider two different measurement functions $h(x, y, \alpha_t)$ and $\bar{h}(x, y, \bar{\alpha_t})$ where $\bar{h}$ uses twice as many Gaussian densities to represent the given cloud as does h. One can then approximate Equation 15 by way of

$$ModelErr(\alpha_t) \cong \sqrt{\iint_{\Omega} (h(x, y, \alpha_t) - \bar{h}(x, y, \bar{\alpha_t}))^2 \; dydx}. \tag{16}$$

Several questions still remain to explore in later work, such as determining if we have enough measurements to compute $\bar{h}$.

Second, how do we include the uncertainty in the model estimate $h(x, y, \alpha_t)$ into the covariance matrix P_e. It is the authors' contention that this issue is even more delicate than the calculation of $g(x, y)$ and we will be examining this issue further in future work. We do note that the Schmidt-Kalman filter derived in [21] provides a promising path forward. In effect, the Schmidt-Kalman filter provides appropriately shaped covariances for unmodeled effects which are *known only by way of a statistical model.* So, we do not need to know the error term $\| h(x, y, \alpha_t) - g(x, y) \|_2$ (or equivalently $\| h(x, y, \alpha_t) - \bar{h}(x, y, \bar{\alpha_t}) \|_2$) exactly, but only need to know something about its statistics. An examination of this approach will appear in future work.

6 Conclusion

In order to effectively compare CBRN data fusion algorithms it is helpful to define metrics by which the algorithms can be evaluated. Here we have considered a sampling of the common metrics from the target tracking domain and briefly considered how they could be applied to CBRN data fusion algorithms. In particular, we chose to study how the covariance consistency metric can be applied to a sample estimation algorithm. We have also noted several areas where a more detailed analysis of the errors will be necessary before realistic scenarios can be run. While we are well aware that a significant amount of further work will be required before metrics from the target tracking community can be applied to real-world chemical fusion problems, we believe this work to be a helpful illustration of how powerful metrics such as covariance consistency can be applied to the domain of CBRN data fusion.

Acknowledgments

We want to extend a special thanks to Dr. Oliver Drummond, and Dr. Louis Scharf for their very helpful contributions to this work. We are also very grateful to the Defense Threat Reduction Agency (DTRA) for providing us with the opportunity to study this problem under the SBIR Phase I "CBRN Sensor and Sensor Netting Algorithms." Additionally we would like to thank BAE Systems and the Army SMDC for making PEGEM 6.1 available to us for CBRN event simulations.

References

[1] S. Lundberg, R. Paffenroth, and J. Yosinski, "Algorithms for distributed chemical sensor fusion," in *Proceedings of SPIE*, Signal and Data Processing of Small Targets, 2010. To Appear.

[2] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems.* Norwood, MA: Artech House, 1999.

[3] R. G. Ewing, D. A. Atkinsona, G. A. Eicemanb, and G. J. Ewing, "A critical review of ion mobility spectrometry for the detection of explosives and explosive related compounds," *Talanta*, vol. 54.3, pp. 515–529, 2001.

[4] D. Long, *Raman Spectroscopy.* Texas: McGraw-Hill, 1977.

[5] S. E. Bisson, "Long-wave ir chemical sensing based on difference frequency generation in orientation-patterned gaas," *Journal Applied Physics B: Lasers and Optics*, vol. Volume 85, Numbers 2-3 / November, 2006, pp. 199–206, 2006.

[6] D. C. Collins and M. L. Lee, "Developments in ion mobility spectrometry-mass spectrometry," *Anal Bioanal Chem*, vol. 372, pp. 66–73, 2002.

[7] S. Kay, C. Xu, and D. Emge, "Chemical detection and classification in raman spectra," in *Proceedings of SPIE* (O. E. Drummond, ed.), vol. 6969, Signal and Data Processing of Small Targets, March 2008.

[8] O. Drummond, "Multiple target tracking with multiple frame, probabilistic data association," in *Proceedings of SPIE*, Signal and Data Processing of Small Targets, 1993.

[9] Y. Bar-Shalom, X.-R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation: Theory, Algorithms, and Software.* J. Wiley, 2001.

[10] Y. Bar-Shalom, ed., *Multitarget-Multisensor Tracking: Applications and Advances.* Dedham, MA: Artech House, 1992.

[11] Y. Bar-Shalom and X.-R. Li, *Estimation and Tracking: Principles, Techniques, and Software.* YBS, 1998.

[12] O. Drummond, "Multiple-frame best-hypotheses target tracking with multiple sensors," in *Proceedings of SPIE*, Signal and Data Processing of Small Targets, 2003.

[13] O. Drummond and D. Dana-Bashian, "Track covariance consistency compensation performance," in *Signal and Data Processing of Small Targets, Proceedings of the SPIE*, vol. 7445, 2009.

[14] S. J. Vanslyke and P. D. Wentzell, "Real-time principal componenet analysis using parallel kalman filter networks for peak purity analysis," *Analytical Chemistry*, vol. 63, pp. 2512–2519, 1991.

[15] P. D. Wentzell and S. J. Vanslyke, "Parallel kalman filter networks for kinetic methods of analysis," *Analytical Chimica Acta*, vol. 257, pp. 172–181, February 1992.

[16] P. D. Wentzell and S. J. Vanslyke, "Parallel kalman filters for peak purity analysis - extensions to non-ideal detector response," *Analytical Chimica Acta*, vol. 307, pp. 459–470, May 1995.

[17] J. C. Chen and S. C. Rutan, "Identification and quantification of overlapped peaks in liquid chromatography with uv diode array detection using an adaptive kalman filter," *Analytical Chimica Acta*, vol. 335, pp. 1–10, December 1996.

[18] S. M. Scott, D. James, and Z. Ali, "Data analysis for electronic nose systems," *Microchimica Acta*, no. 156, pp. 183–207, 2007.

[19] O. Drummond, A. Perrella, and S. Waugh, "On target track covariance consistency," in *Proceedings of SPIE*, Signal and Data Processing of Small Targets, 2006.

[20] J. Stoer, R. Bulirsch, W. Gautschi, and C. Witzgall, *Introduction to numerical analysis.* Springer Verlag, 2002.

[21] R. Paffenroth, R. Novoselov, S. Danford, M. Teixeira, S. Chan, and A. Poore, "Mitigation of biases using the Schmidt-Kalman filter," in *Proceedings of SPIE*, vol. 6699, p. 66990Q, 2007.