DISSERTATION

IMPROVEMENTS TO THE TRACKING PROCESS

Submitted by

Codie T. Lewis

Department of Electrical and Computer Engineering

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Fall 2024

Doctoral Committee:

Advisor: Margaret Cheney
Co-Advisor: Venkatacha Chandrasekaran

David Crouse
Michael Kirby

ABSTRACT

IMPROVEMENTS TO THE TRACKING PROCESS

Accurate target tracking is a fundamental requirement of modern automated systems. An accurate tracker must correctly associate new observations to existing tracks and update those tracks to reflect the new information. An accurate tracker is one which predicts assignments and measurement distributions closely matching the ground truth. This work will show that aspects of the GNP algorithm and IMM filter require amendments and renewed investigation. To aide the framing of the solutions in the context of tracking, some general background will be presented first. More specific background will be given prior to the corresponding contributions.

Modern sensor networks require the alignment of track pictures from multiple sensors (sometimes called sensor registration). This issue was described in the 1990s and termed the global nearest pattern problem in the early 2000s. The following work presents a correction and extension of the solution to the global nearest pattern problem with a heuristic error estimation algorithm. Its use for sensor calibration is demonstrated.

Once measurements have been associated to tracks, there still remain several choices that define the tracking algorithm, one being the filtering algorithm which updates the track state. One common solution for filtering is the interacting multiple model filter which was originally developed in the 1980s. This is essentially a bank of Kalman filters which are weighted and mixed based on a predefined Markov chain. The validity of the assumptions on that Markov chain will be discussed and recommendation for replacing those assumptions with neural networks will be proposed and assessed.

Finally, following association of two tracks for a single target, it is necessary to combine their information while respecting the lack of knowledge about correlations between the

tracks. Covariance intersection was developed in the 1990s and 2000s for track-to-track fusion when tracks are assumed Gaussian. A generalization of covariance intersection, Chernoff fusion, was developed in the 2000s for handling general track states. A connection made in the literature which allows for direct analysis of the error of Chernoff fusion is used to evaluate the effectiveness of Fibonacci lattices for quasi-Monte Carlo integration solutions required by Chernoff fusion.

ACKNOWLEDGEMENTS

This work was sponsored in part by the Office of Naval Research under award numbers N0001422WX00049 and N000142412241.

DEDICATION

This work is dedicated to my parents, who always strove to give me the means to pursue my own dreams; to my wife, who supported and encouraged me through times of self-doubt; and to my advisors and mentors, who guided me in my professional pursuits.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction to Tracking

1.1 Motivation and Prelude

Today, sensors of all types are more ubiquitous than ever. As the number of sensors has grown, so has the compute power available for processing. Many old tracking algorithms were written under heavy computing constraints, but today, improved processing and memory capabilities allow use of more computationally intensive algorithms. Similarly, theoretical developments have enabled better computational efficiency than was possible before. Combined, these advancements have had an especially large impact on the target tracking field, which relies heavily on fast and efficient processing to cope with propagating uncertainties. In the following work, improvements to existing tracking related algorithms will be presented. Due to the invention and proliferation of connectivity technologies, modern systems frequently work as part of a network, sharing information to increase situational awareness. Therefore, many of the improvements will be either directly or indirectly concerned with networked sensors.

This work will consist of self-contained chapters, each addressing an improvement to a subproblem within the tracking field. These components could be combined into a single tracker, but the goal of this work is to present them in a broad setting so that parameters are not needlessly restricted. That choice will result in the need for narrower introductions and some open-ended suggestions within each chapter. This introduction will provide a general unifying context for the chapters to follow.

While presented in a language that is my own, the following introductory ideas came about after attempting to synthesize information from several general reference texts on tracking [1]–[6]. Additionally, other introductory sources in the form of articles or slides have influenced my approach to this field [7]–[10]. These texts generally follow the classical

treatment of tracking as it has been developed over the past half century or so, with set theory, combinatorics, probability theory, and control theory. In the past two decades, there have been generalized formulations of tracking theory proposed using the formalisms of random finite set statistics and analytic combinatorics [11]–[18]. The random finite set statistics approach resulted in the development of the probability hypothesis density (PHD) filter and its variants [19]–[22]. Analytic combinatorics has aimed to be a more approachable unifying framework than random finite set statistics, but is still very abstract and relatively new. Since the PHD filter will not be needed in what follows and the classical approach provides sufficient tools for the work presented, the classical approach will be the path followed here.

1.2 The Subproblems of Target Tracking

Target tracking can be thought of as requiring components for sequentially solving four subproblems: sensor pre-processing, association, estimation, and fusion. Each component must be developed under constraints from the environment while accounting for the input/output flow of the tracker as a whole. One of the most important constraints, as noted above, is computation time. Ideal processing often requires too much time to be feasible. However, a tracker that produces unusable results is not acceptable either. So a balance must be maintained between computational expense and quality of results. This balance will be mentioned at several points in the following work along with practical heuristics for real-time implementation.

Sensors provide imperfect observations (also called measurements), a collection $Z \coloneqq \{\mathbf{z}_i\}$ taken from a measurement space $\mathcal{Z}$, sampled from objects in the surveillance environment. The intention is to illuminate targets of interest, a collection $T \coloneqq \{\mathbf{x}_i\}$ taken from a state space $\mathcal{T}$, while minimizing the amount of uninteresting observations called clutter, which have a distribution to be discussed later. Measurements are generated via a function which

maps the target state space into a measurement space defined by the sensor:

$$g : \mathcal{T} \times \mathcal{C} \to \mathcal{Z} \tag{1.1}$$

where $\mathcal{C}$ is a space containing the various potential sources of clutter (also called "false alarms"). Measurements are also called observations or detections, although detections typically refer to the measurements a sensor provides on successfully detected targets as opposed to the total set of measurements available if a target is detected. The binary question of whether a detection is made can be represented by the inclusion of a Bernoulli random variable ι with the measurement so that $Z \coloneqq \{(\mathbf{z}_i, \iota_i)\}$ where $\iota_i \in \{0, 1\}$. The words "detection" and "measurement" will be used interchangeably unless a distinction is required.

In a system of sensors, multiple sensors will be used as inputs to a tracker with the idea that many sensors can provide better coverage of the surveillance area. In this case, using ℓ to index the sensors, there would be a $g^{(\ell)}$ and $\mathcal{Z}^{(\ell)}$ for every sensor, though the targets and clutter will remain the same. There is of course a cost associated with this seemingly simple addition of more measurements. A tracker must have logic in subsequent components for handling multiple collections of measurements and the shear increase in the number of measurements will increase processing requirements. The latter issue places pressure on the former, requiring fast logic to maintain a reasonable processing time. Part of the logic's job is to address the alignment problem resulting from discrepancies in sensor calibrations. Sensors report their measurements in a coordinate system based on that sensor's estimation of its own kinematic properties. If the sensor's estimations are incorrect (due to a faulty sensor or incorrect calibration), then the reported observations will also be systematically incorrect as well. Even without systematic errors, errors are inevitable since transformation of measurements to a common coordinate system are often nonlinear and require approximations which introduce errors. Therefore, measurements from multiple sensors must be aligned so that they share a common global coordinate system. The alignment problem is also relevant when sharing tracks (estimated information about true target states) instead

of measurements between trackers. The alignment problem will be the focus of Chapter 2 where a solution will be proposed.

Alignment of measurements on multiple targets requires an understanding of which measurements correspond to the same targets. To accomplish this, measurements must be passed to an association component. The goal for this component is to generate a mapping from measurements to the source of the measurement (i.e. which true target or clutter object was observed). For one sensor, this mapping would be:

$$h : \mathcal{Z} \to \mathcal{T} \times \mathcal{C} \tag{1.2}$$

where $\mathcal{Z}$ is the set of measurements. In the case of N_ℓ sensors indexed by ℓ, the domain of h is modified to account for the additional measurement collections:

$$h : \mathcal{Z}^{(1)} \times \cdots \times \mathcal{Z}^{(N_\ell)} \to \mathcal{T} \times \mathcal{C}. \tag{1.3}$$

A typical assumption is that one measurement per sensor can be assigned to each true target at a single time. This will be used in Chapter 2 for sensor alignment purposes.

Once measurements are assigned to a target, the tracker must use them to update tracks. Tracks are estimates of the target's characteristics at the current instant given a history of detections. For kinematic tracking, the kth instance of a track is represented by a timestamp t_k which implies when a track was valid, a state vector $\mathbf{x}_k$ containing position components and the desired derivatives thereof (e.g. velocity and acceleration), and other statistics related to the state such as covariance $\mathbf{P}_k$. Estimation for a track is typically solved using a version of the Kalman filter or particle filter, which takes a predicted track state computed from prior information and uses the associated measurements to update the prediction. The Kalman filter variants all assume that the track and measurement distributions are well-approximated by Gaussian distributions. The original Kalman filter also assumed linearity in target motion and measurement model, but later developments allowed for use of some

nonlinear functions. Particle filter techniques allow for a relaxation of those Gaussian and linearity assumptions at the cost of more computations. In general, the track prediction and update process is done following Bayesian probability principles to acquire posterior estimates. The prediction piece requires the tracker to encode possible changes to the track state between observation periods:

$$f : \mathcal{X} \to \mathcal{X} \tag{1.4}$$

where $\mathcal{X}$ is the track space which is estimable given measurements from Z, which may be different from the actual target space $\mathcal{T}$. Any elements of $\mathcal{C}$ which are indistinguishable from at least one element of $\mathcal{T}$ under the mapping $\mathcal{T} \times \mathcal{C} \to \mathcal{X}$ must be considered valid targets to the tracker or the indistinguishable element of $\mathcal{T}$ must be removed as a trackable target. In the problems considered here, the true motion model is unknown to the tracker and must be approximated. But there are algorithms for hypothesizing multiple motion models and combining the resulting predictions and updates to better match the true target motion. Chapter 3 will discuss multiple model tracking in more depth and propose an augmentation which will yield improved track estimates.

1.3 General Bayesian Target Tracking

Now that the subproblems of tracking are laid out, it will help to provide some context into usual assumptions made in target tracking. A typical problem statement for a tracker will state what types of targets should be tracked and what sensors are available to track them. For this reason, the state update and sensor measurement equations are often discussed together, being immediately derivable from the problem statement. Recall that mapping f encodes the expectations about how targets $\mathcal{T}$ are expected to behave given a model restricted to $\mathcal{X}$. Mapping g then encodes the quantities which are accessible to a sensor and which should make the track state in $\mathcal{X}$ observable. These also are the two primary sources of error in a tracker, as the functions f and g each introduce fundamentally irreducible sources of error that accumulate over time.

The process model describes the predicted changes of a target state over time (captured by f) along with any uncertainties that might affect those changes. The motion model of a target is typically not known, even if the target is cooperative. The impossibility of fully controlling an environment leads to discrepancies between the intended motion of the target and its actual response. For an aircraft, the pilot may steer the plane in a straight line, but occasional wind gusts will inevitably push the plane off course. These sources of aleatory uncertainty (fundamentally irreducible randomness) are characteristic of cooperative targets. With uncooperative aircraft, the dominant uncertainty is epistemic uncertainty (a subjective lack of information). The hypothesized motion model will almost always fail to capture some aspect of the true motion of the target. Therefore, an effort must be made to quantify the degree to which one trusts that the hypothesized motion accurately captures the target's motion. Regardless of which type of uncertainty is dominant, for convenience the total uncertainty (aleatory and epistemic) at timestamp t_k will be represented here by an additive noise process $q(\mathbf{w}(t_k))$. The input noise variable $\mathbf{w}$ is often assumed to be zero-mean mean Gaussian noise with covariance matrix $\mathbf{Q}$, $\mathbf{w} \sim \mathcal{N}(0, \mathbf{Q})$. These assumptions yield the process equation:

$$\mathbf{x}(t_k) \coloneqq f(\mathbf{x}(t_{k-1})) + q(\mathbf{w}(t_k)). \tag{1.5}$$

The use of discrete time steps k implies an assumed discretized model of the underlying continuous dynamics. Continuous dynamics of random variables are represented via stochastic differential equations (SDEs).

There are many motion models to choose from depending on the problem context. For an extensive survey, see [23]. The nearly constant velocity model (NCV) is by far the preferred motion model in the tracking literature as it provides linear transitions and requires only position and velocity estimates. When inclusion of other models is desired, it is often done using a multiple model algorithm which entertains a set of possible motion models, NCV included. Justification on why the NCV is preferable to higher order linear models, nearly

constant acceleration (NCA) in particular, is provided in [24]. The use of multiple model filtering will be further discussed in Chapter 3.

For acquiring measurements, each sensor has a measurement model g which describes the conversion from the target and clutter state domain $\mathcal{X} \times \mathcal{C}$ into the measurement domain $\mathcal{Z}$ as described above. Of course, sensors are designed for a given purpose, but there still may exist unforeseen or unaddressed sources of error in the operating environment. Any such errors would be epistemic in nature. More typical though is consideration of the aleatory uncertainty. Processing for radar sensors for example must contend with finite resolution capabilities and random perturbations to the signal in transit from transmitter to target to receiver. The detectors also can be affected by internal electrical and heat interference which will show up as noise in the measurement. While irreducible, these quantities are typically measured and specified for the sensor in question, and if not, they can be estimated and fitted with a distribution. Again, the typical assumption is that these errors in aggregate amount to additive zero-mean mean Gaussian noise with covariance matrix $\mathbf{R}$, $\mathbf{v} \sim \mathcal{N}(0, \mathbf{R})$. Of course, measurements can be biased, making the mean vector nonzero. Chapter 2 will discuss that case explicitly, but unless otherwise stated, the bias will be taken as a zero vector. In either case, the measurement process at timestamp t_k can be expressed as follows:

$$\mathbf{z}(t_k) \coloneqq g(\mathbf{x}(t_k)) + r(\mathbf{v}(t_k)). \tag{1.6}$$

for additive noise $r(\mathbf{v}(t_k))$ and assuming a single target produces a single measurement. The generalization which avoids this assumption considers g as a set-valued function. In what follows, it will be assumed that a single measurement is returned for a single target, that is $Z(t_k) = \{(\mathbf{z}(t_k), \iota)\}$ in the single-target case. For a survey of common measurement models, see [25].

Bayesian tracking provides a recursive procedure which moves from prior to posterior to subsequent prior. A review of the fundamental equations will be given below based on the treatment in Chapter 3 of [6]. First, to compress notation, time indices will be moved to the

subscript and the following alternative notations will be defined:

$$\hat{\mathbf{x}}_{k|k-1} \coloneqq \mathbb{E}[\mathbf{x}_k | Z_{k-1}, ..., Z_0] \tag{1.7}$$

$$f_k \coloneqq f(\mathbf{x}_{k-1}) \tag{1.8}$$

$$q_k \coloneqq q(\mathbf{w}_k) \tag{1.9}$$

$$g_k \coloneqq g(\mathbf{x}_k) \tag{1.10}$$

$$r_k \coloneqq r(\mathbf{v}_k). \tag{1.11}$$

The subscript notation in $\hat{\mathbf{x}}_{k|k-1}$ indicates the estimate of $\mathbf{x}$ at time index k given information up to time index $k-1$. Assuming an initial prior probability $p(x_k|Z_{k-1}, ..., Z_0)$ describing the target location at time index k given measurements up to time index $k-1$, Bayes' rule provides a method for incorporating the new information from a measurement taken at timestamp t_k. Bayes' rule transforms the prior into the posterior as follows:

$$\begin{aligned} p(\mathbf{x}_k | Z_k, ..., Z_0) &= \frac{p(Z_k | \mathbf{x}_k, Z_{k-1}, ..., Z_0) p(\mathbf{x}_k | Z_{k-1}, ..., Z_0)}{p(Z_k | Z_{k-1}, ..., Z_0)} \\ &= \frac{p(Z_k | \mathbf{x}_k) p(\mathbf{x}_k | Z_{k-1}, ..., Z_0)}{p(Z_k)} \end{aligned} \tag{1.12}$$

where the simplification follows from assuming time-independent measurements. This completes the prior to posterior step. The posterior distribution can then be used to compute an expected value for $\mathbf{x}_k$ and its covariance given all measurements, including for the current

time step:

$$\hat{\mathbf{x}}_{k|k} \coloneqq \mathbb{E}[\mathbf{x}_k|Z_k, ..., Z_0] \tag{1.13}$$

$$= \int_{\mathcal{X}} \mathbf{x}_k p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \tag{1.14}$$

$$= \int_{\mathcal{X}} \left[f_k + q_k\right] p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \tag{1.15}$$

$$= \int_{\mathcal{X}} f_k p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \tag{1.16}$$

$$= \int_{\mathcal{X}} f_k \frac{p(Z_k|\mathbf{x}_k) p(\mathbf{x}_k|Z_{k-1}, ..., Z_0)}{p(Z_k)} d\mathbf{x}_k \tag{1.17}$$

$$\mathbf{P}_{k|k} \coloneqq \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})'|Z_k, ..., Z_0] \tag{1.18}$$

$$= \int_{\mathcal{X}} (\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})' p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \tag{1.19}$$

$$= \int_{\mathcal{X}} \left[f_k + q_k - \hat{\mathbf{x}}_{k|k}\right] \left[f_k' + q_k' - \hat{\mathbf{x}}_{k|k}'\right] p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \tag{1.20}$$

$$\begin{aligned} = &\int_{\mathcal{X}} [f_k - \hat{\mathbf{x}}_{k|k}][f_k - \hat{\mathbf{x}}_{k|k}]' p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \\ &+ \int_{\mathcal{X}} [f_k - \hat{\mathbf{x}}_{k|k}] q_k' p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \\ &+ \int_{\mathcal{X}} q_k [f_k - \hat{\mathbf{x}}_{k|k}]' p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \\ &+ \int_{\mathcal{X}} q_k q_k' p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \end{aligned} \tag{1.21}$$

$$= \int_{\mathcal{X}} [f_k - \hat{\mathbf{x}}_{k|k}][f_k - \hat{\mathbf{x}}_{k|k}]' p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k + \mathbf{Q_k} \tag{1.22}$$

$$= \int_{\mathcal{X}} [f_k - \hat{\mathbf{x}}_{k|k}][f_k - \hat{\mathbf{x}}_{k|k}]' \frac{p(Z_k|\mathbf{x}_k) p(\mathbf{x}_k|Z_{k-1}, ..., Z_0)}{p(Z_k)} d\mathbf{x}_k + \mathbf{Q_k} \tag{1.23}$$

where Equation (1.16) assumes zero mean process noise $q(\mathbf{w}_k)$ and Equation (1.22) assumes the process noise is uncorrelated with $f(\mathbf{x}_{k-1})$. This completes one Bayesian iteration of prior to posterior.

Moving from the current posterior (on step k) to the subsequent prior for the next iteration (step $k+1$) requires the Chapman-Kolmogorov equation:

$$p(\mathbf{x}_{k+1}|Z_k, ..., Z_0) = \int_{\mathcal{X}} p(\mathbf{x}_{k+1}|\mathbf{x}_k) p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k. \tag{1.24}$$

This new prior distribution can then be used at the beginning of the Bayesian update for the $k+1$ time step. To compute an expected value and covariance for $\mathbf{x}_k$ given all prior measurements, apply Equation (1.24) to Equations (1.16) and (1.22), respectively:

$$\hat{\mathbf{x}}_{k+1|k} \coloneqq \mathbb{E}[\mathbf{x}_{k+1}|Z_k, ..., Z_0] \tag{1.25}$$

$$= \int_{\mathcal{X}} \int_{\mathcal{X}} \mathbf{x}_{k+1} p(\mathbf{x}_{k+1}|\mathbf{x}_k) p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k d\mathbf{x}_{k+1} \tag{1.26}$$

$$= \int_{\mathcal{X}} f_{k+1} p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \int_{\mathcal{X}} p(\mathbf{x}_{k+1}|\mathbf{x}_k) d\mathbf{x}_{k+1} \tag{1.27}$$

$$= \int_{\mathcal{X}} f_{k+1} p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k. \tag{1.28}$$

Using the expected value for the prediction in Equation 1.28, a covariance of that prediction can be computed:

$$\mathbf{P}_{k+1|k} \coloneqq \mathbb{E}[(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k})(\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k})'|Z_k, ..., Z_0] \tag{1.29}$$

$$= \int_{\mathcal{X}} \int_{\mathcal{X}} [\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k}][\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k}]' p(\mathbf{x}_{k+1}|\mathbf{x}_k) p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k d\mathbf{x}_{k+1} \tag{1.30}$$

$$= \int_{\mathcal{X}} [\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k}][\mathbf{x}_{k+1} - \hat{\mathbf{x}}_{k+1|k}]' p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k \int_{\mathcal{X}} p(\mathbf{x}_{k+1}|\mathbf{x}_k) d\mathbf{x}_{k+1} \tag{1.31}$$

$$= \int_{\mathcal{X}} [f_{k+1} - \hat{\mathbf{x}}_{k+1|k}][f_{k+1} - \hat{\mathbf{x}}_{k+1|k}]' p(\mathbf{x}_k|Z_k, ..., Z_0) d\mathbf{x}_k + \mathbf{Q}_{k+1}. \tag{1.32}$$

Finally, Equation (1.6) says that measurements are noisy and dependent on the state vector. The measurement mean, measurement covariance, and measurement-state cross-

covariance, respectively, associated with the probability $p(\mathbf{z}_k|\mathbf{x}_k)$ are computed as follows:

$$\begin{aligned}\hat{\mathbf{z}}_k &\coloneqq \mathbb{E}[\mathbf{z}_k|\mathbf{x}_k, Z_{k-1}, ..., Z_0] \\ &= \int_{\mathcal{X}} h(\mathbf{x}_k) p(\mathbf{x}_k|Z_{k-1}, ..., Z_0) d\mathbf{x}_k \qquad (1.33)\\ \mathbf{R}_k &\coloneqq \mathbb{E}[(\mathbf{z}_k - \hat{\mathbf{z}}_k)(\mathbf{z}_k - \hat{\mathbf{z}}_k)'|\mathbf{x}_k, Z_{k-1}, ..., Z_0] \\ &= \int_{\mathcal{X}} [h_k - \hat{\mathbf{z}}_k][h_k - \hat{\mathbf{z}}_k]' p(\mathbf{x}_k|Z_{k-1}, ..., Z_0) d\mathbf{x}_k + \mathbf{S}_k \\ &\text{where } \mathbf{S}_k \coloneqq \int_{\mathcal{X}} r_k r_k' p(\mathbf{x}_k|Z_{k-1}, ..., Z_0) d\mathbf{x}_k \qquad (1.34)\\ \mathbf{T}_k &\coloneqq \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})(\mathbf{z}_k - \hat{\mathbf{z}}_k)'|Z_{k-1}, ..., Z_0] \\ &= \int_{\mathcal{X}} [f_k - \hat{\mathbf{x}}_{k|k-1}][h_k - \hat{\mathbf{z}}_k]' p(\mathbf{x}_k|Z_{k-1}, ..., Z_0) d\mathbf{x}_k. \qquad (1.35)\end{aligned}$$

There are numerous methods for carrying out the Bayesian process described above. The most well-known is the linear Kalman filter which assumes $f(\mathbf{x}) = \mathbf{F}\mathbf{x}$ and $g(\mathbf{x}) = \mathbf{G}\mathbf{x}$, such that $\mathbf{F}$ and $\mathbf{G}$ are matrices independent of $\mathbf{x}$, and propagates the first and second moments of a distribution [3], [26], [27]. The Kalman filter was later extended to handle nonlinear f and g by creating linearized approximations: $f(\mathbf{x}) \approx \tilde{\mathbf{F}}\mathbf{x}$ and $g(\mathbf{x}) \approx \tilde{\mathbf{G}}\mathbf{x}$ where $\tilde{\mathbf{F}}$ and $\tilde{\mathbf{G}}$ are matrices independent of $\mathbf{x}$ [3], [27], [28]. This linearization approach to nonlinear filtering is called the extended Kalman filter. While useful in nearly-linear cases, it was noted as being unreliable and difficult to tune [29]. The unscented Kalman filter, and later the cubature and general sigma-point Kalman filters, were developed to provide better performance on nonlinear functions [28]–[32]. For highly nonlinear f and g, numerical approximations of the Bayesian equations above can be achieved with particle filtering methods [33]–[35].

The Kalman filter variants all serve as first and second moment prediction and update methods which is all the detail that will be required. For more details on the specifics of implementing Kalman filters, see the Kalman filter functions available in the Tracker Component Library [9]. For Chapter 3, the important aspect of filtering given the discussion above will be that Kalman filters are required to assume a single f and g in the process and measurement equations (Equations (1.5) and (1.6), respectively). This is true regardless

of which Kalman filter variant is chosen. Multiple model filters such as the interacting multiple model (IMM) filter, generalized pseudo-Bayesian (GPB) filters, and later variants were developed to make such choices less detrimental to the overall tracker performance [2]–[4], [36]–[38]. The details of the IMM and GPB filters will be covered in Chapter 3.

1.4 Association

In multitarget tracking there is often ambiguity in the assignment of measurements to tracks, corresponding to the selection of h in Equation (1.2) for the single-sensor case or Equation (1.3) for the multi-sensor case. Given a detection probability less than 1, the measurement set will often have a cardinality less than the total number of targets and clutter measurements in the surveilled volume provided that the assumption each target or clutter object generates at most one measurement holds. It is also assumed that targets can be identified with tracks, that is each target has a unique associated track, which is the quantity in state space $\mathcal{X}$ which will be associated to measurements in measurement space $\mathcal{Z}$. The association is done in the measurement space, using the projection of tracks into $\mathcal{Z}$ via Equation (1.6).

Define the track and measurement sets as follows:

$$X \coloneqq \{\mathbf{x}_i \in \mathcal{X} : i = 1, ..., N_X\} \tag{1.36}$$

$$Z \coloneqq \{\mathbf{z}_j \in \mathcal{Z} : j = 1, ..., N_Z\}. \tag{1.37}$$

The assignment problem considered here can be stated in terms of optimization over bipartite graphs such as the one illustrated in Figure 1.1. The definition of the optimal association mapping h from Equation (1.2) is the bipartite graph which minimizes some given cost function. Note that since not every measurement needs to have an associated track, there is also usually a nonassignment cost.

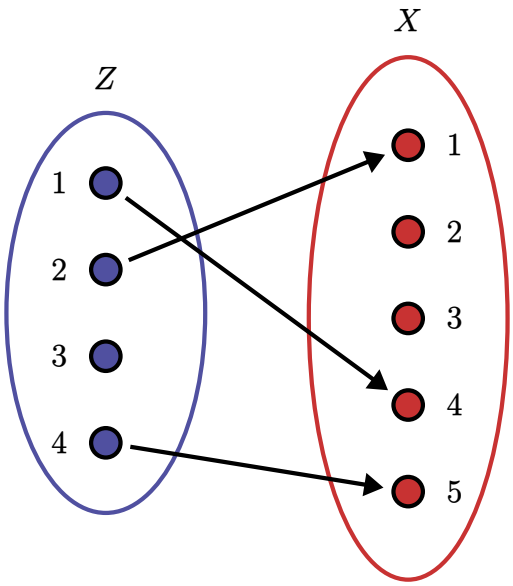


Figure 1.1: An example of a bipartite graph for measurement-to-track association. This graph can be interpreted as a representation of which targets generated which measurements. The target associated with track $\mathbf{x}_1$ generated measurement $\mathbf{z}_2$. Similarly, the targets associated with $\mathbf{x}_4$ and $\mathbf{x}_5$ generated measurements $\mathbf{z}_1$ and $\mathbf{z}_4$, respectively. The lack of edges for tracks $\mathbf{x}_2$ and $\mathbf{x}_3$ would be called "missed detections". The lack of an edge for $\mathbf{z}_3$ would indicate either a "false alarm" or a target without an existing track.

The simplest optimization approach are the local nearest neighbor algorithms. First consider nearest neighbor without replacement (NNwoR) which enforces the one-measurement-to-one-track hard assignment rule. Let $d(\mathbf{a}, \mathbf{b})$ be some metric defined on $\mathcal{Z}$ and let it be implied that an argument from $\mathcal{X}$ must first be projected into $\mathcal{Z}$ as $g(\mathbf{x}) \in \mathcal{Z}$ for $\mathbf{x} \in \mathcal{X}$. Next, randomly shuffle X with respect to the track indices. This is to prevent preferential assignment over many time steps, though other options such as prioritizing tracks which have existed for a longer time could be chosen instead. To avoid more notation, we will assume the tracks were sufficiently shuffled in their original ordering. Next, for each track in the shuffled X, compute $j^* = \arg\min_{j \in A_i} d(\mathbf{x}_i, \mathbf{z}_j)$ where the descending set sequence $\{A_i\}_{i=1}^{N_X+1}$ contains the sets of unassigned measurements when track $\mathbf{x}_i$ is next in the queue for assignment. If $A_i = \emptyset$, meaning no measurements are available for assignment, or if $d(\mathbf{x}_i, \mathbf{z}_{j^*}) = \infty$, meaning all available measurement assignments are infeasible, then track $\mathbf{x}_i$ is not in the image of h. If $A_{N_X+1} \neq \emptyset$, then define $h(\mathbf{z}_j) = 0$ for all $\mathbf{z}_j \in A_{N_X+1}$ where 0

here represents a null target. Using this procedure, one then has a NNwoR description of the association mapping:

$$h_{\text{NNwoR}}(\mathbf{z}_j) \coloneqq \begin{cases} 0, & \text{if } \mathbf{z}_j \in A_{N_X+1} \\ \mathbf{x}_i, & \text{if } j = \arg\min_{j \in A_i} d(\mathbf{x}_i, \mathbf{z}_j). \end{cases} \tag{1.38}$$

subject to the constraints that:

$$\sum_{i=0}^{N_X} a_{i,j} = 1 \quad \text{for all } j = 1, ..., N_Z \tag{1.39}$$

$$\sum_{j=0}^{N_Z} a_{i,j} = 1 \quad \text{for all } i = 1, ..., N_X \tag{1.40}$$

$$\text{where } a_{i,j} = \begin{cases} 1, & \text{if } h(\mathbf{z}_j) = \mathbf{x}_i \\ 0, & \text{otherwise.} \end{cases} \tag{1.41}$$

These constraints state that any non-null measurement (the $j = 0$ case being the null measurement) is assigned to exactly one track. Similarly, any non-null track (the $i = 0$ case being the null track) must be assigned to exactly one measurement. The NNwoR mapping minimizes the cost function described by metric d with ranks over $1, ..., N_X$. The ranks are determined by the ordering chosen for queuing up tracks from X. So define the rank of a measurement as $r_j = i$ such that $h(\mathbf{z}_j) = \mathbf{x}_i$. For any other h, at the lowest rank r_j^* for which the measurement assignment differs, the value $d(h(\mathbf{z}_j), \mathbf{x}_{r_j^*})$ will be greater than $d(h_{\text{NNwoR}}(\mathbf{z}_j), \mathbf{x}_{r_j^*})$.

Nearest neighbor with replacement (NNwR) works similarly to NNwoR, except now the set of available measurements for all i is Z. Therefore, define a decreasing set sequence $\{B_i\}_{i=1}^{N_X+1}$ such that B_i are the sets of unassigned measurements after the first $i-1$ iterations over the track queue. Then define $h(\mathbf{z_j}) = 0$ for all $\mathbf{z}_j \in B_{N_X+1}$. The NNwR association

mapping is then defined:

$$h_{\mathrm{NNwR}}(\mathbf{z}_j) \coloneqq \begin{cases} 0, & \text{if } \mathbf{z}_j \in B_{N_X+1} \\ \mathbf{x}_i, & \text{if } j = \arg\min_{j \in A_i} d(\mathbf{x}_i, \mathbf{z}_j). \end{cases} \tag{1.42}$$

The NNwR mapping minimizes the following cost function, notably simpler to state than NNwoR:

$$\mathcal{C}_{NNwR}(h) = \sum_{(i,j)} d(\mathbf{x}_i, \mathbf{z}_j) a_{i,j} \quad \text{where } a_{i,j} = \begin{cases} 1, & \text{if } h(\mathbf{z}_j) = \mathbf{x}_i \\ 0, & \text{otherwise.} \end{cases} \tag{1.43}$$

The local nearest neighbor algorithms both have flaws when compared to what should occur intuitively. In NNwoR, the one measurement to one track principle is respected, however not every track gets a chance to contend for every measurement. In NNwR, each track is assigned to the best measurement under metric d, but the assignments are unconstrained, possibly resulting in many tracks being assigned to the same measurement. The global nearest neighbor (GNN) algorithm solves these issues simultaneously. Under the constraint that every non-null track is assigned to one measurement and each non-null measurement is assigned to one track, the metric is minimized over the space of all feasible pairings. The problem can be stated as follows:

minimize: (1.44)

$$\mathcal{C}(h) = \sum_{(i,j)} d(\mathbf{x}_i, \mathbf{z}_j) a_{i,j} \quad \text{where } a_{i,j} = \begin{cases} 1, & \text{if } h(\mathbf{z}_j) = \mathbf{x}_i \\ 0, & \text{otherwise} \end{cases} \tag{1.45}$$

subject to: (1.46)

$$\sum_{i=0}^{N_X} a_{i,j} = 1 \quad \text{for all } j = 1, ..., N_Z \tag{1.47}$$

$$\sum_{i=0}^{N_Z} a_{i,j} = 1 \quad \text{for all } j = 1, ..., N_X. \tag{1.48}$$

This can be recognized as a linear assignment problem for which many (strong and weak) polynomial time algorithms exist. Commonly cited algorithms include the Hungarian algorithms [39], the Munkres algorithms [40], [41], auction algorithms [42]–[44], and the Jonker-Volgenant (JV) and Jonker-Volgenant-Castanon (JVC) algorithms [45], [46]. Many modifications to these have been made with varying results, and comparisons of optimality can be found in the literature [4], [47]–[50]. The JV/JVC algorithms tend to perform best in the widest range of assignment problems and therefore are preferred over the other options. In the event that the k-best solutions are needed, one can use Murty's algorithm [51]. Optimized versions of Murty's algorithm provide the best performance to date for this k-best problem [52].

This section has focused on measurement-to-track association. However, there was no distinction that prohibited the same techniques from being applied to general association. If the two sets X and Z above are changed to $X^{(1)}$ and $X^{(2)}$ for track-to-track association or $Z^{(1)}$ and $Z^{(2)}$ for measurement-to-measurement association, the same algorithms are applicable. The connection is illustrated in Figure 1.2. The problem presented above is also called the 2d assignment problem and can be solved in polynomial time as noted in the comparative studies mentioned above. There is a generalization to s-dimensional (sd) assignment ($s \geq 3$ and usually corresponding to the number of sensors or time frames) for the case of associating s-many distinct sets at one time, both in the case of needing a single best assignment and k-best assignments [53]–[55]. This is also called multidimensional assignment or MDA, and is no longer a bipartite graph problem. Unfortunately, the sd assignment problem is known to be NP-hard and no polynomial time algorithm is known to exist. However, there are approaches relying on Lagrangian relaxation which solve a sequence of 2d assignment problems and yield workable solutions without a guarantee of finding the optimal assignment [3].

A point that will become important in Chapter 2 is the acknowledgement that the GNN algorithms require independently calculable costs for each hypothesized association. That is, the cost of any edge in the bipartite graph such as that in Figure 1.1 needs to be computed

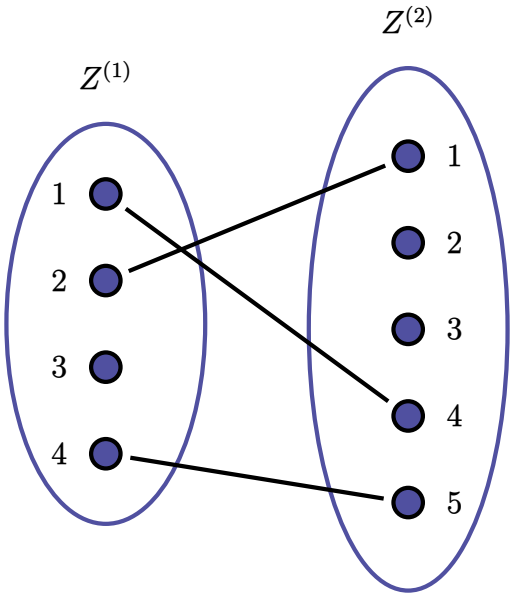


Figure 1.2: An example of a bipartite graph for measurement-to-measurement association. This graph can be interpreted as a representation of which measurements from sensor 1 were generated by corresponding targets measured by sensor 2. The target which generated measurement $\mathbf{z}_2$ in sensor 1 also generated measurement $\mathbf{z}_1$ in sensor 2. Similarly, the targets which generated measurements $\mathbf{z}_1$ and $\mathbf{z}_4$ in sensor 1 generated measurements $\mathbf{z}_4$ and $\mathbf{z}_5$ in sensor 2, respectively. The lack of edges for tracks $\mathbf{z}_3$ in sensor 1 and $\mathbf{z}_2$ and $\mathbf{z}_3$ in sensor 2 would be deemed unassociated measurements implying they were generated by different sources not captured in the opposing measurement set. Measurements can also be associated with clutter sources, though that case was excluded in this example.

prior to deciding on all of the other edges. If independence of $d(\mathbf{x}_i, \mathbf{z}_j)$ for any two pairs (i_1, j_1) and (i_2, j_2) such that $i_1 \neq i_2$ and $j_1 \neq j_2$ cannot be assumed, then solving GNN is no longer a viable approach to finding the optimal assignment. This problem is further exacerbated by gating. Gating is the restriction of acceptable pairings via constraints understood through the problem. One common method for gating with Gaussian tracks and measurements is ellipsoidal gating, which provides an ellipsoid gating region. In measurement space given two Gaussian states $(\mathbf{z}_1, \mathbf{P}_1)$ and $(\mathbf{z}_2, \mathbf{P}_2)$, the ellipsoid gating region is described as follows:

$$V \coloneqq \{\mathbf{z} \in \mathcal{Z} : (\mathbf{z} - (\mathbf{z}_1 - \mathbf{z}_2))' (\mathbf{P}_1 + \mathbf{P}_2)^{-1} (\mathbf{z} - (\mathbf{z}_1 - \mathbf{z}_2)) \leq \gamma\} \tag{1.49}$$

where γ is usually taken as the value returned by evaluating the corresponding inverse χ^2 distribution at a chosen probability α. Typically one of the measurements is taken to be the expected measurement given the predicted target state $\mathbb{E}[g(\mathbf{x})|\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1}]$. The use of a χ^2 distribution follows from the Gaussian measurement assumption, since Equation (1.49) is a quadratic form of Gaussian variables and positive definite scaling matrix. This gating approach works so long as the given means are accurate and there is no relative bias between them given the observed target. When one or both of the measurements is biased, the relative bias between them becomes nonzero and the region V ceases to be useful on its own. There are algorithms which attempt to inflate V in order to account for the bias, but they often require a priori knowledge of the bias and increase the opportunities for misassociation [56]. Chapter 2 will present a solution to the biased assignment problem without using covariance inflation.

1.5 Conclusion

This chapter has provided an overview of the problems that must be solved by a tracker. The various spaces $(\mathcal{X}, \mathcal{Z}, \mathcal{T}, \mathcal{C})$ must be considered when determining the target motion models and measurement models. Those choices in turn determine the filtering techniques

that would yield acceptable errors, which will be important for the work presented in Chapter 3. The requisite solution of assignment problems for measurement-to-track and track-to-track association was also discussed. It was noted that the usual metric (d) used in GNN must assume zero bias, which is not always a valid assumption. Solution of the assignment problem when bias is nonzero will be the topic of Chapter 2. Finally, a study of a track-to-track fusion algorithm called Chernoff fusion will be described in Chapter 4. It will be shown that Fibonacci points provide a demonstrable benefit for the QMC approximation that arises when using Chernoff fusion.

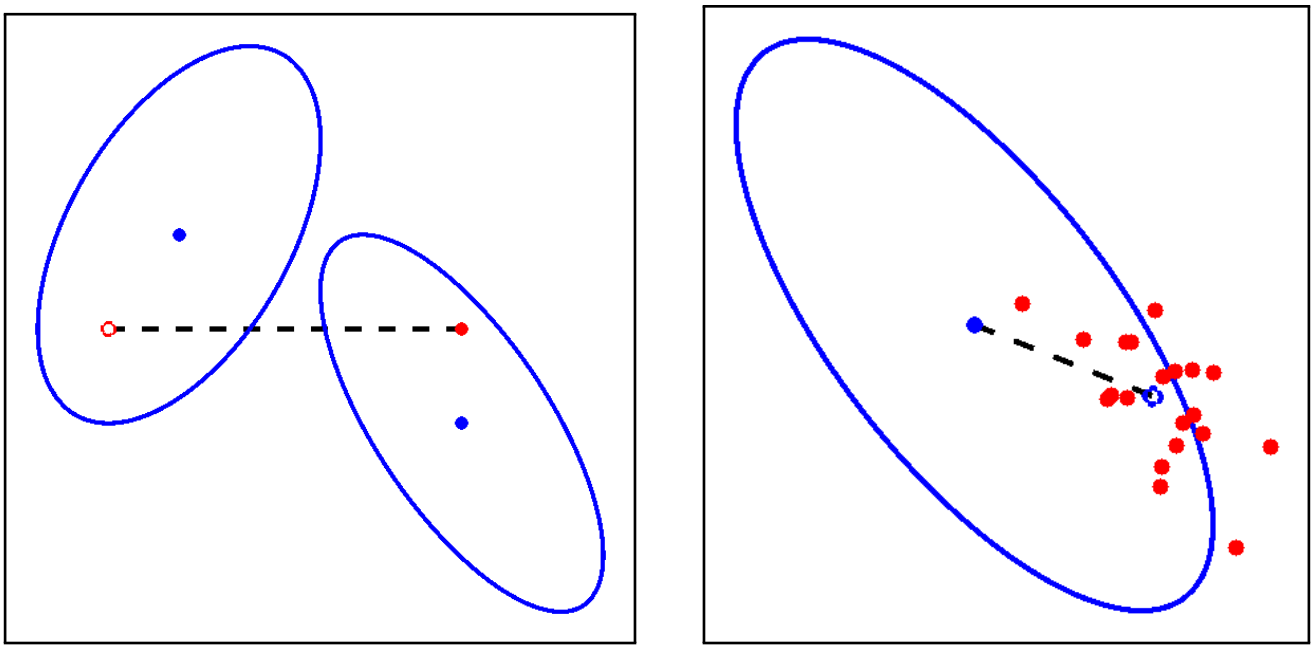
Chapter 2

A Scale-free Global Nearest Pattern Criterion

2.1 Introduction

In practical multi-sensor tracking scenarios, sensors may exhibit systemic errors which introduce a bias in detections and state estimates. Bias errors are often assumed to be small enough to be encapsulated by the process and measurement noises typically assumed for target tracking problems and therefore a non-issue from an algorithmic viewpoint. However, even with relatively small biases, association performance can be significantly impacted by such errors. For example, biased state estimates can end up gating and associating with the wrong target (see Figure 2.1a). Alternatively, biased detections can fall within the gate for a target less frequently than would be expected based on the modeled errors (see Figure 2.1b). The result of the data failing to align with the association assumptions will be track degradation (estimation accuracy below what was predicted) if not track loss (failure to maintain an accurate target estimate for the lifetime of the target).

Identification and alignment of patterns shared by sets of tracking data is known as the global nearest pattern (GNP) problem, a term coined by Levedahl [57]–[60]. The scope of the problem considered here are translation and scale biases. Solving the GNP problem involves performing association in a multi-sensor, multi-target scenario under the assumption of a persistent bias between state estimates of the same target made by different sensors. Kenefic addressed this point in [61], where he presented a method for determining whether a friendly sensor platform (e.g. the own-ship) was being misidentified by another platform's tracker as an enemy platform. Unfortunately, Kenefic's algorithm did not address the combinatorial explosion due to the required enumeration of all possible hypotheses which quickly becomes computationally infeasible [57]. In [57] and [62], a solution called global nearest pattern matching (GNPM) was proposed as a means of both associating state estimates from different



(a) Incorrect Association **(b)** Missed Gating

Figure 2.1: Two examples of potential association problems caused by bias error are associating a detection with the wrong target (incorrect association) and failing to correctly gate detections with their correct target (missed associations). In Figure 2.1a, the detection without bias (solid red circle) should gate with one target (bottom-right blue circle). However, the detection is biased to the hollow red circle, as shown by the dashed black line, and associates with another target (top-left blue circle). In Figure 2.1b, the target (solid blue circle) is observed with biased detections (solid red circles). The blue ellipse is drawn so that 99.97% of detections on that target should fall within the ellipse. The bias (dashed black line) shifts detections to a center at the hollow blue circle, which causes fewer detections to fall within the gating ellipse than expected.

sensors and aligning sensors given limited prior information about the sensors' orientations. Both Levedahl and Kenefic approached the problem by explicitly introducing a bias term in the cost function to be optimized. However, Levedahl's GNPM algorithm introduced a gating technique for eliminating infeasible associations prior to the optimization procedure [57], [62]. The sequence of papers [62]–[64] coauthored by Levedahl on GNPM following [57] seek to further optimize the GNPM algorithm so as to eliminate a large number of associations prior to cost optimization.

GNPM is not the only algorithm for solving GNP [65]. Early approaches attempted to iteratively provide a bias estimate over time [66]. However, this runs into the circular problem that one needs to designate assignments to produce bias estimates to improve assignments.

The marginal track-to-track association method (MTTA), an approach which marginalizes the bias estimate and ultimately provides a cost function similar to that of GNPM, is explored in [64] in parallel with GNPM after the original derivation in [60]. Some metrics for assessing association accuracy in GNP problems can be found in [67]. One alternative approach to GNP is provided in [68], which uses integer nonlinear programming to establish the associations. Another alternative solution is provided in [69], which performs track-to-track fusion via soft data association. In spite of attempts at more effective techniques, GNPM is very efficient and fast for problems with enumerable hypotheses and is still a state-of-the-art algorithm [65]. An extension to nonlinear measurement models has been derived in [70] under the name distributed linear bias (DLB) model, but the core of the algorithm remains the same as the simplified GNPM algorithm presented here.

GNPM will be considered exclusively for a solution to the GNP problem in what follows. In [64], an analysis of the GNPM score function is given, revealing the relationship between the GNN and GNPM costs. One problem with utilizing GNPM arises from the growth in computational complexity as the cardinalities of the detection sets grow. GNPM couples association decisions into a single hypothesis, which is used to estimate the bias needed in the GNPM formulas. This coupling makes establishing a gating criterion like that used in GNN fairly difficult. Gating criteria are given in [57], [59], [63], [64], but the authors note that the formulas require augmentation based on the units used for the state space. In one example problem given in [64], the provided gating criterion yields a negative result which is supposed to be used as a bound on a quadratic form involving covariance matrices. Since covariance matrices are positive-definite, the criterion in [64] would seem to suggest that no associations are possible in the example, an incorrect result. In order to correct for this deficiency in the GNPM algorithm, a scale-invariant gating criterion will be derived below. An estimate of how confident one should be in any particular GNPM solution will also be provided.

In Section 2.2, the GNP problem is defined and the approach of GNPM is laid out. Section 2.3 provides the derivation of the desired gating criterion and a discussion of the prediction error. Then Section 2.4 provides an example of the derived criterion being applied. Finally, Section 2.6 provides a brief conclusion.

2.2 Problem Description

Assume two sensors (S1 and S2) are attempting to detect N targets $\{\mathbf{x}_n\}_{n=1}^{N}$ in d-dimensional Cartesian state space on a single scan. For simplicity, the detection is assumed to be given in state space coordinates. The goal of global nearest pattern (GNP) association is to identify which state estimates from S1 correspond to state estimates from S2 and make the correct associations between those state estimates. The simplest detection model for sensor S1 assumes measurements are mapped directly to the state space, $\mathcal{Z} = \mathcal{X}$. Using the $\mathcal{Z} = \mathcal{X}$ assumption, Equation (1.6) becomes Equation (2.1). If $\mathcal{Z} \neq \mathcal{X}$ and the measurement mapping becomes nontrivial, the error and bias statistics will have to be adjusted either by a simple linear transformation as shown in [64] or by inclusion of Jacobian terms demonstrated in [70]. However, as long as the resulting measurements are well-approximated by Gaussians, the derivations below will remain similar. Therefore, the simplified $\mathcal{Z} = \mathcal{X}$ assumption will be used. Detections obtained by sensor S1 are assumed to be Gaussian perturbations of the true target location. Similarly, we define an analogous detection model for sensor S2 in Equation (2.2). Here, $\mathbf{b}$ is an unknown detection bias for state estimates from sensor S2 relative to S1 and is assumed independent of the Gaussian measurements. Assume this bias is common to all the measurements in the detection set of S2 and that $\mathbf{b} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\mathbf{b}})$.

$$\mathbf{x}_{\mathrm{S1},n} = \mathbf{x}_n + \boldsymbol{\xi}_{\mathrm{S1},n}, \quad \boldsymbol{\xi}_{\mathrm{S1},n} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\mathrm{S1},n}) \tag{2.1}$$

$$\mathbf{x}_{\mathrm{S2},m} = \mathbf{x}_m + \boldsymbol{\xi}_{\mathrm{S2},m} - \mathbf{b}, \quad \boldsymbol{\xi}_{\mathrm{S2},m} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\mathrm{S2},m}) \tag{2.2}$$

Let $\hat{N}_{\mathrm{S1}}$ and $\hat{N}_{\mathrm{S2}}$ be the number of state estimates registered by sensors S1 and S2, respectively. Assume missed detections are possible so that the number of true targets N may be above the number of registered state estimates by either sensor. Furthermore, assume that each sensor registers at most one detection per target per measurement interval. It is typical to assume that there are no false alarms, as these would require including the probability of false alarm in the following derivations, but otherwise would essentially act like "targets" that only one sensor detected [57], [62]–[64]. Therefore, assume neither sensor registers false alarms. Then without loss of generality, let $\hat{N}_{\mathrm{S1}} \geq \hat{N}_{\mathrm{S2}}$ and define an association tuple $h = (h_1, ..., h_{\hat{N}_{\mathrm{S1}}})$. The tuple h describes the assignment of the detection i from sensor S1 to detection h_i from sensor S2. If detection i from sensor S1 is not assigned to any detection from sensor S2, $h_i \coloneqq 0$.

Using the association function h described above, let $\mathcal{K}_{\mathrm{S1}} \coloneqq \{i : h_i = 0\}$ (the set of all detection indices from sensor S1 that do not associate to a detection from sensor S2), $n_a \coloneqq |\mathcal{K}_{\mathrm{S1}}^c| = |\{1, ..., \hat{N}_{\mathrm{S1}}\} \setminus \mathcal{K}_{\mathrm{S1}}|$, and define the following for all $i \in \mathcal{K}_{\mathrm{S1}}^c$:

$$\mathbf{x}_{\Delta,i} \coloneqq \mathbf{x}_{\mathrm{S1},i} - \mathbf{x}_{\mathrm{S2},h_i} \tag{2.3}$$

$$\mathbf{Q}_{\Delta,i} \coloneqq \mathbf{Q}_{\mathrm{S1},i} + \mathbf{Q}_{\mathrm{S2},h_i} \tag{2.4}$$

which are the associated difference and covariance, respectively. Finally, assumptions on the probability of detection for each sensor must be stated. Since there are two sensors, there are 4 possible outcomes for any individual target in the surveilled region which are given in Table 2.1. The probability of false alarm and probability of detection can be obtained for any sensor. Since the assumption was made that there are no false alarms, the joint probabilities of detection denoted in Table 2.1 can be computed solely by multiplying the probabilities of detection for each sensor under the further assumption that the sensor detection rates are independent.

Table 2.1: Symbols for possible outcomes after attempting to detect a single target at one time instance with two sensors.

Symbol	Outcome	Probability
$\circ\circ$	Both sensors detect the target.	$P_{\circ\circ}$
$\circ\bullet$	S1 detects the target, S2 does not.	$P_{\circ\bullet}$
$\bullet\circ$	S2 detects the target, S1 does not.	$P_{\bullet\circ}$
$\bullet\bullet$	Neither sensor detects the target.	$P_{\bullet\bullet}$

With the context described above, the assignment problem can now be stated. Given two collections of state estimates $\mathbf{X}_{\mathrm{S1}} \coloneqq \{\mathbf{x}_{\mathrm{S1},n,i}\}_{i=1}^{\hat{N}_{\mathrm{S1}}}$ and $\mathbf{X}_{\mathrm{S2}} \coloneqq \{\mathbf{x}_{\mathrm{S2},m,j}\}_{j=1}^{\hat{N}_{\mathrm{S2}}}$ from a common time instant, find the association function h and corresponding bias $\mathbf{b}$ which maximize the joint likelihood of the hypothesized associations and bias:

$$\mathcal{L}(h, \mathbf{b} \mid \mathbf{X}_{\mathrm{S1}}, \mathbf{X}_{\mathrm{S2}}) = \prod_{\ell \in \mathcal{K}_{\mathrm{S1}}^c} p(\mathbf{x}_{\Delta,\ell} - \mathbf{b}, \circ\circ \mid \mathbf{b}) p(\mathbf{b}) \prod_{\ell \in \mathcal{K}_{\mathrm{S1}}} p(\mathbf{x}_{\mathrm{S1},\ell}, \circ\bullet) \prod_{\ell \in \mathcal{K}_{\mathrm{S2}}} p(\mathbf{x}_{\mathrm{S2},\ell}, \bullet\circ) \tag{2.5}$$

$$\begin{aligned} &= p(\mathbf{b}) \prod_{\ell \in \mathcal{K}_{\mathrm{S1}}^c} p(\mathbf{x}_{\Delta,\ell} - \mathbf{b} \mid \mathbf{b}) p(\circ\circ \mid \mathbf{x}_{\mathrm{S1},\ell}, \mathbf{x}_{\mathrm{S2},h_\ell}) p(\mathbf{x}_{\mathrm{S1},\ell}, \mathbf{x}_{\mathrm{S2},h_\ell}) \\ &\quad \times \prod_{\ell \in \mathcal{K}_{\mathrm{S1}}} p(\mathbf{x}_{\mathrm{S1},\ell}) p(\circ\bullet \mid \mathbf{x}_{\mathrm{S1},\ell}) \qquad (2.6) \\ &\quad \times \prod_{\ell \in \mathcal{K}_{\mathrm{S2}}} p(\mathbf{x}_{\mathrm{S2},\ell}) p(\bullet\circ \mid \mathbf{x}_{\mathrm{S2},\ell}) \\ &= (V\beta)^{n_a} p(\mathbf{b}) \prod_{\ell \in \mathcal{K}_{\mathrm{S1}}^c} p(\mathbf{x}_{\Delta,\ell} - \mathbf{b} \mid \mathbf{b}) p(\circ\circ \mid \mathbf{x}_{\Delta,\ell} - \mathbf{b}) \\ &\quad \times (V\beta)^{\hat{N}_{\mathrm{S1}} - n_a} \prod_{\ell \in \mathcal{K}_{\mathrm{S1}}} p(\circ\bullet \mid \mathbf{x}_{\mathrm{S1},\ell}) \qquad (2.7) \\ &\quad \times (V\beta)^{\hat{N}_{\mathrm{S2}} - n_a} \prod_{\ell \in \mathcal{K}_{\mathrm{S2}}} p(\bullet\circ \mid \mathbf{x}_{\mathrm{S2},\ell}) \end{aligned}$$

where we have assumed independence of state estimates in Equation (2.5) and uniform prior probabilities for all targets in Equation (2.7). The uniform prior target density for bounded volume V is expressed as β. $\mathcal{K}_{\mathrm{S2}}$ is, analogous to $\mathcal{K}_{\mathrm{S1}}$, defined as the set of state estimates

from sensor S2 which have no associated detection from S1 via h:

$$\mathcal{K}_{\mathrm{S2}} = \{j \in \{1, ..., \hat{N}_{\mathrm{S2}}\} \mid j \notin \mathrm{Im}(h)\}. \tag{2.8}$$

Assume that the number of state estimates for each sensor is Poisson distributed in the surveilled volume V. Plugging in the above Gaussian assumptions on state estimates and the bias, Equation (2.7) becomes Equation (2.9).

$$\begin{aligned}\mathcal{L}(h, \mathbf{b} \mid \mathbf{X}_{\mathrm{S1}}, \mathbf{X}_{\mathrm{S2}}) = &\frac{1}{\sqrt{\det(2\pi\mathbf{Q}_{\mathbf{b}})}} \exp\left\{-\frac{1}{2}\mathbf{b}'\mathbf{Q}_{\mathbf{b}}^{-1}\mathbf{b}\right\} \\ &\times \prod_{\ell \in \mathcal{K}_{\mathrm{S1}}^c} \frac{1}{\sqrt{\det(2\pi\mathbf{Q}_{\Delta,i})}} \exp\left\{-\frac{1}{2}(\mathbf{x}_{\Delta,\ell} - \mathbf{b})'\mathbf{Q}_{\Delta,i}^{-1}(\mathbf{x}_{\Delta,\ell} - \mathbf{b})\right\} \\ &\times (P_{\circ\bullet}V\beta)^{\hat{N}_{\mathrm{S1}} - n_a}(P_{\bullet\circ}V\beta)^{\hat{N}_{\mathrm{S2}} - n_a}(P_{\circ\circ}V\beta)^{n_a}\end{aligned} \tag{2.9}$$

2.3 GNPM Gating Criterion Derivation

2.3.1 The Cost Function

Note that the quantities β, V, $\hat{N}_{\mathrm{S1}}$, $\hat{N}_{\mathrm{S2}}$, $\mathbf{Q}_{\mathbf{b}}$, $P_{\circ\circ}$, $P_{\circ\bullet}$, and $P_{\bullet\circ}$ do not vary from one hypothesized association h to another for a given set of state estimates (nor does the dimensionality d of those state estimates). It will be convenient to multiply the right-hand side of Equation (2.9) by the constant term given in Equation (2.10). The result is shown in Equation (2.11) (analogous to Equation (37) of [64]).

$$P_{\circ\circ}^{-\hat{N}_{\mathrm{S1}}} P_{\circ\bullet}^{(\hat{N}_{\mathrm{S1}} - \hat{N}_{\mathrm{S2}})} (\beta V)^{-\hat{N}_{\mathrm{S2}}} V^{n_a} (2\pi)^{d(\hat{N}_{\mathrm{S1}}+1)/2} \sqrt{\det(\mathbf{Q}_{\mathbf{b}})} \tag{2.10}$$

$$\begin{aligned}\mathcal{L}(h, \mathbf{b} \mid \mathbf{X}_{\mathrm{S1}}, \mathbf{X}_{\mathrm{S2}}) \propto &\left(\frac{P_{\circ\circ}}{(2\pi)^{d/2}\beta V P_{\bullet\circ}P_{\circ\bullet}}\right)^{-(\hat{N}_{\mathrm{S1}} - n_a)} \exp\left\{-\frac{1}{2}\mathbf{b}'\mathbf{Q}_{\mathbf{b}}^{-1}\mathbf{b}\right\} \\ &\times \prod_{\ell \in \mathcal{K}_{\mathrm{S1}}^c} \frac{V}{\sqrt{\det(\mathbf{Q}_{\Delta,\ell})}} \exp\left\{-\frac{1}{2}(\mathbf{x}_{\Delta,\ell} - \mathbf{b})'\mathbf{Q}_{\Delta,\ell}^{-1}(\mathbf{x}_{\Delta,\ell} - \mathbf{b})\right\}\end{aligned} \tag{2.11}$$

The term containing the detection probabilities is what [64] defines as G_0, defined here in Equation (2.12).

$$G_0 \coloneqq \frac{P_{\circ\circ}}{(2\pi)^{d/2}\beta V P_{\bullet\circ}P_{\circ\bullet}} \tag{2.12}$$

Taking the negative logarithm of Equation (2.11) and multiplying by 2 yields the cost function $\mathcal{C}$ shown in Equation (2.13):

$$\begin{aligned}\mathcal{C} \coloneqq{}& \mathbf{b}'\mathbf{Q}_{\mathbf{b}}^{-1}\mathbf{b} + 2(\hat{N}_{\mathrm{S1}} - n_a)\log(G_0) - 2n_a\log(V) \\ &+ \sum_{\ell\in\mathcal{K}_{\mathrm{S1}}^{\mathrm{c}}} \left[\log(\det(\mathbf{Q}_{\Delta,\ell})) + (\mathbf{x}_{\Delta,\ell} - \mathbf{b})'\mathbf{Q}_{\Delta,\ell}^{-1}(\mathbf{x}_{\Delta,\ell} - \mathbf{b})\right].\end{aligned} \tag{2.13}$$

Since the true bias is not known a priori, an initial bias estimate is necessary. Let $\hat{\mathbf{b}}$ be an a priori estimate of $\mathbf{b}$. In [64], this was chosen to be the definition given in Equation (2.14) which is the bias that maximizes the posterior likelihood in Equation (2.11) [70]. Note that the leading inverted term in Equation (2.14) is an expression for the posterior covariance of the bias [70]. While this is a valid bias for computing the cost of an entire hypothesis, it will not be useful when considering each individual pairing in Section 2.3.2 due to the need for all assignments in the summations. Therefore, a tentative prior bias can be computed via the average pairwise distance of state estimates from sensors S1 and S2 as given in Equation (2.15). The accompanying estimated bias covariance is given in Equation (2.16).

$$\hat{\mathbf{b}}_{\mathrm{ML}} \coloneqq \underbrace{\left(\hat{\mathbf{Q}}_{\mathbf{b}}^{-1} + \sum_{\ell\in\mathcal{K}_{\mathrm{S1}}^{\mathrm{c}}} \mathbf{Q}_{\Delta,\ell}^{-1}\right)^{-1}}_{\text{posterior } \hat{\mathbf{Q}}_{\mathbf{b}}} \sum_{\ell\in\mathcal{K}_{\mathrm{S1}}^{\mathrm{c}}} \mathbf{Q}_{\Delta,\ell}^{-1}\mathbf{x}_{\Delta,\ell} \tag{2.14}$$

$$\hat{\mathbf{b}}_{\text{avg.}} \coloneqq \frac{1}{\hat{N}_{\mathrm{S1}}\hat{N}_{\mathrm{S2}}} \sum_{(i,j)} (\mathbf{x}_{\mathrm{S1},i} - \mathbf{x}_{\mathrm{S2},j}) \tag{2.15}$$

$$\hat{\mathbf{Q}}_{\mathbf{b}} \coloneqq \frac{1}{V\beta P_{\circ\circ} - 1} \sum_{(i,j)} (\mathbf{Q}_{\mathrm{S1},i} + \mathbf{Q}_{\mathrm{S2},j}) \tag{2.16}$$

Note that the sum in Equation (2.16) is divided by the expected number of true samples of the bias $V\beta P_{\circ\circ}$ minus one, not the number of total differences included in the sum $\hat{N}_{\mathrm{S1}}\hat{N}_{\mathrm{S2}}$

minus one. This avoids overconfidence in the estimator $\hat{\mathbf{b}}_{\text{avg.}}$. Additionally, assuming $V\beta$ is approximately equal to the number of true targets, $V\beta P_{\circ\circ} \to \hat{N}_{\text{S1}}\hat{N}_{\text{S2}}$ as $P_{\circ\circ} \to 1$, retaining the typical covariance estimator as a limiting case. This covariance estimator is invalid when $V\beta P_{\circ\circ} < 1$; however, in that case it should not be expected that one could compute a bias estimate at all since one would expect that no sampling of the bias occurred. Regardless of the choice of $\hat{\mathbf{b}}$, the prior cost for a proposed association hypothesis given the bias estimate $\hat{\mathbf{b}}$ is shown in Equation (2.17).

$$\begin{aligned} \mathcal{C}_{\text{prior}} \coloneqq\ & \hat{\mathbf{b}}'\hat{\mathbf{Q}}_{\mathbf{b}}^{-1}\hat{\mathbf{b}} + 2(\hat{N}_{\text{S1}} - n_a)\log(G_0) - 2n_a\log(V) \\ & + \sum_{\ell \in \mathcal{K}_{\text{S1}}^{\text{c}}} \left[\log(\det(\mathbf{Q}_{\Delta,\ell})) + (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})'\mathbf{Q}_{\Delta,\ell}^{-1}(\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})\right] \end{aligned} \tag{2.17}$$

2.3.2 The Gating Criterion

Since evaluating every possible permutation of assignments for h is computationally prohibitive, a method for sifting out the least likely associations is necessary. Such a gating criterion requires a method of computing the marginal cost for each association without reference to other associations in the candidate h. It was noted above that $\mathcal{C}$ includes $\mathbf{b}$ which is estimated from all nonzero associations in h. The use of $\mathcal{C}_{\text{prior}}$ and $\hat{\mathbf{b}} = \hat{\mathbf{b}}_{\text{avg}}$ will therefore be necessary for deriving a gating criterion.

Consider the value of the null assignment case, $n_a = 0$. The null assignment cost is given in Equation (2.18). If any hypothesis h is to be chosen over the null hypothesis, then the cost will have to be less than $\mathcal{C}_{\text{prior}}^0$. The minimum cost $\mathcal{C}_{\text{prior}}^*$ therefore would satisfy $0 \leq \mathcal{C}_{\text{prior}}^0 - \mathcal{C}_{\text{prior}}^*$. Expanding and simplifying the difference expression on right-hand side of the inequality yields Equation (2.19). There is one more simplification that can be made concerning the $\hat{\mathbf{b}}'\hat{\mathbf{Q}}_{\mathbf{b}}\hat{\mathbf{b}}$ term. Since this is a quadratic form, it is necessarily greater than or equal to 0. The negative sign in front of the term means that any nonzero estimate $\hat{\mathbf{b}}$ will yield fewer feasible solutions. Therefore, more feasible solutions can be entertained by simply setting the $\hat{\mathbf{b}}'\hat{\mathbf{Q}}_{\mathbf{b}}\hat{\mathbf{b}}$ to zero, yielding Equation (2.20). In the event there are too many

feasible solutions from this simplification, reverting to Equation (2.19) should yield fewer. The benefit to using Equation (2.20) is primarily that it leaves room for error in estimations of $\hat{\mathbf{b}}$ and $\hat{\mathbf{Q}}_{\mathbf{b}}$.

$$\mathcal{C}^0_{\text{prior}} = 2\hat{N}_{\text{S1}} \log(G_0) \tag{2.18}$$

$$\begin{aligned} 0 \leq\ & 2n_a \log(G_0) + 2n_a \log(V) - \hat{\mathbf{b}}' \hat{\mathbf{Q}}_{\mathbf{b}} \hat{\mathbf{b}} \\ & - \sum_{\ell \in \mathcal{K}_{\text{S1}}} \left[\log(\det(\mathbf{Q}_{\Delta,\ell})) + (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})' \mathbf{Q}^{-1}_{\Delta,\ell} (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}}) \right] \end{aligned} \tag{2.19}$$

$$\begin{aligned} \leq\ & 2n_a \log(G_0) + 2n_a \log(V) \\ & - \sum_{\ell \in \mathcal{K}_{\text{S1}}} \left[\log(\det(\mathbf{Q}_{\Delta,\ell})) + (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})' \mathbf{Q}^{-1}_{\Delta,\ell} (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}}) \right] \end{aligned} \tag{2.20}$$

The marginal difference specific to a chosen associated pair of state estimates is contained in the summation. The quantities outside of the summation term are dependent on the chosen hypothesis h via the n_a coefficient. Since n_a cannot be known as a marginal quantity, it must be chosen so as to allow all feasible solutions to satisfy the inequality in Equation (2.20). The complication here is that the terms including n_a may be positive or negative depending on the values of β, V, $P_{\circ\circ}$, $P_{\circ\bullet}$, $P_{\bullet\circ}$, and d as shown in Figure 2.2. The curve in Figure 2.3 curve shows the threshold where $G_0 = 1$, assuming the sensors have identical probabilities of detection for various target densities.

The expression in Equation (2.21) must be determined to provide a valid substitution for n_a.

$$0 \gtrless 2n_a \log(G_0) + 2n_a \log(V) = 2n_a (\log(G_0 V)) \tag{2.21}$$

Note that $V > 0$, so the condition can be simplified to Equation (2.22). Rearranging terms yields the relation in Equation (2.23), where the terms on the right-hand side are mostly

sensor dependent, and the terms on the left-hand side are mostly scenario dependent.

$$1 \lessgtr G_0 V \tag{2.22}$$

$$\beta \overset{\hat{N}_{S1}}{\underset{0}{\lessgtr}} \frac{P_{\circ\circ}}{(2\pi)^{d/2} P_{\bullet\circ} P_{\circ\bullet}} \tag{2.23}$$

When the greater than inequality in Equation (2.21), less than inequality in Equation (2.22), or less than inequality in Equation (2.23) are satisfied, choose $n_a = \hat{N}_{S1}$. Otherwise, choose $n_a = 0$.

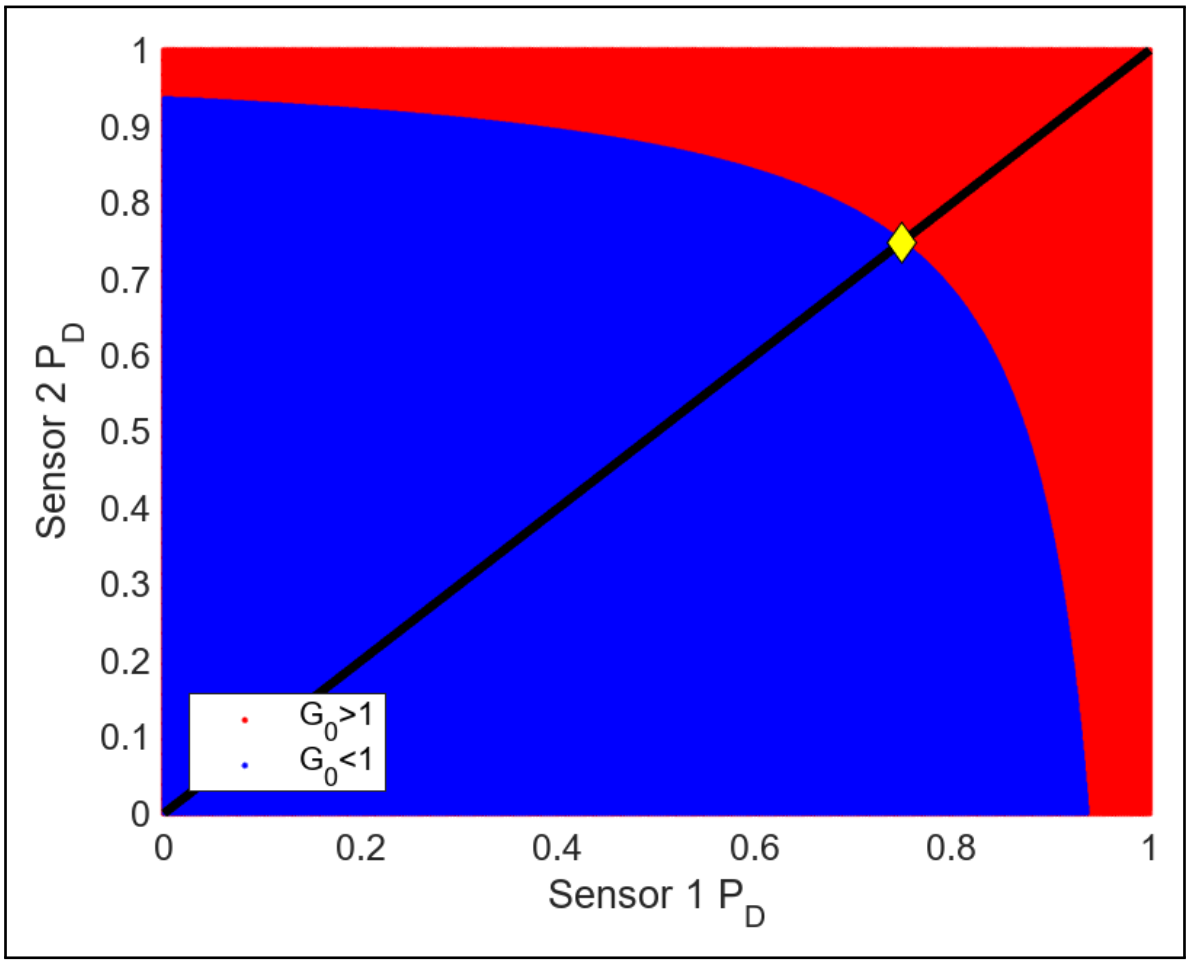


Figure 2.2: Comparing the value of G_0 to 1 for possible sensor detection probabilities P_D assuming no false alarms. The values for β and V are fixed at 1 and the dimensionality d is fixed at 3. The black line marks where the sensors have equivalent detection probabilities and the yellow diamond is the threshold on that line under these assumptions.

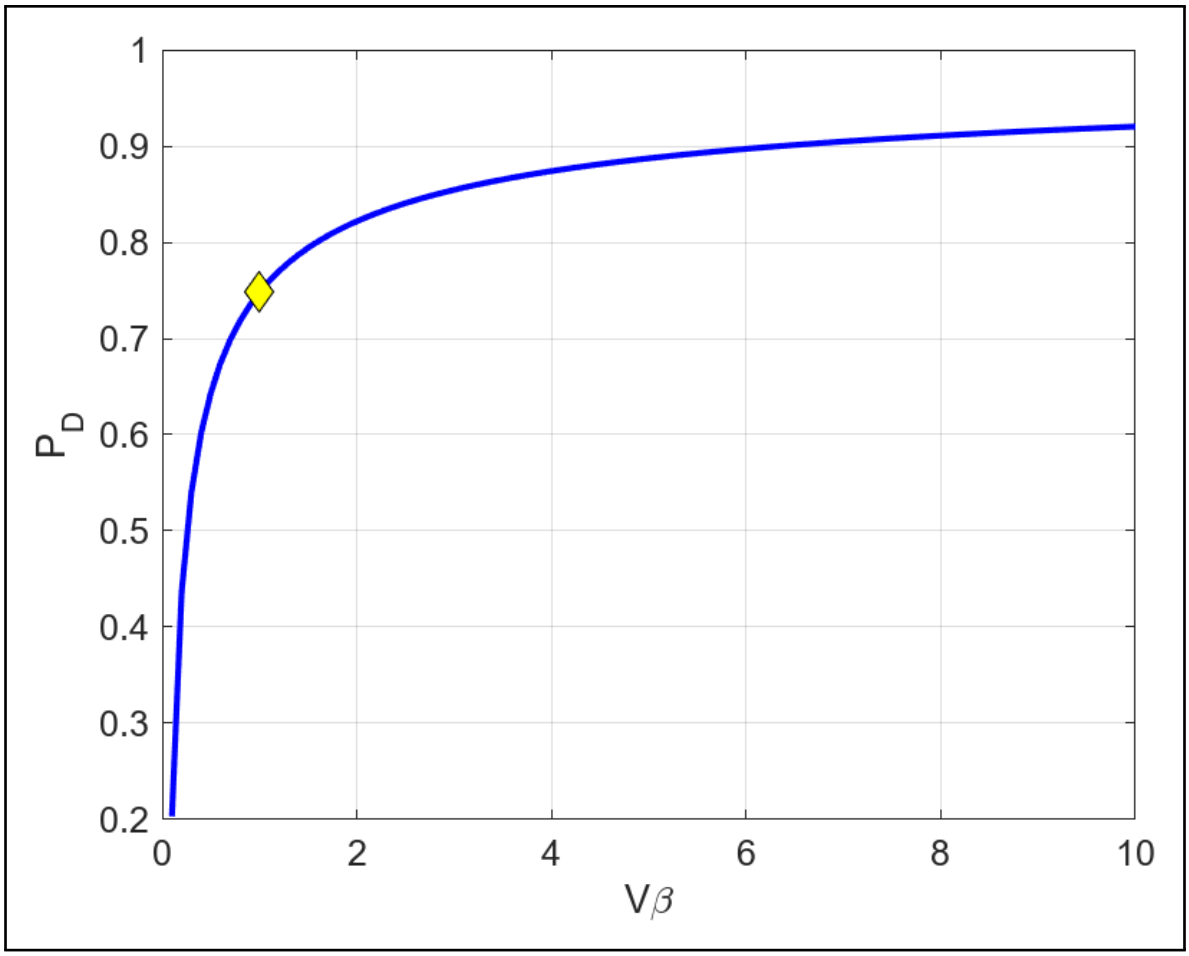


Figure 2.3: A curve showing the $G_0 = 1$ threshold given $V\beta$ expected targets assuming both sensors have the same probability of detection P_D. The dimension is set to 3. The area below the curve corresponds to $G_0 < 1$, and the area above the curve corresponds to $G_0 > 1$. The yellow diamond indicates the same point marked on the black line in Figure 2.2.

Once the choice of n_a has been made, some rearranging of terms in Equation (2.20) will yield Equation (2.24):

$$\sum_{\ell \in \mathcal{K}_{S1}} \left[\log(\det(\mathbf{Q}_{\Delta,\ell})) + (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})' \mathbf{Q}_{\Delta,\ell}^{-1} (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}}) \right] \leq 2n_a \log(G_0 V). \tag{2.24}$$

The left-hand side is entirely dependent on the given sets of state estimates, and the right-hand side is entirely dependent on the sensors and surveillance region assumptions. The determinant term is necessarily positive since covariance matrices are positive-definite. The quadratic form term is also positive. Consequently, every term in the summation is positive.

The entire summation of positive terms must satisfy the upper bound inequality in Equation (2.24); therefore, every individual term in the summation must also obey the inequality. This leads to the final simplification in Equation (2.25).

$$\log(\det(\mathbf{Q}_{\Delta,\ell})) + (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})'\mathbf{Q}_{\Delta,\ell}^{-1}(\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}}) \leq 2n_a \log(G_0 V) \tag{2.25}$$

2.3.3 Errors

There are two primary sources of error in Equations (2.25) and (2.17), detection error and bias estimate error. These errors are both contained in the quadratic form in Equation (2.26), which is always positive given the positive-definite property of $\mathbf{Q}_{\Delta,i}$. Therefore, any errors in the estimate of $\mathbf{x}_{\Delta,i} - \hat{\mathbf{b}}$ will result in a positive contribution to the left-hand side of Equation (2.25). This can result in incorrect disqualification of some hypotheses. To prevent or at least account for such occurrences, the probability distribution for this quadratic form error will be considered below.

$$(\mathbf{x}_{\Delta,i} - \hat{\mathbf{b}})'\mathbf{Q}_{\Delta,i}^{-1}(\mathbf{x}_{\Delta,i} - \hat{\mathbf{b}}) \tag{2.26}$$

The detection error is a result of the Gaussian variables $\boldsymbol{\xi}_{\text{S1}}, \boldsymbol{\xi}_{\text{S2}}$ which have covariances $\mathbf{Q}_{\text{S1},i}$ and $\mathbf{Q}_{\text{S2},h_i}$ respectively. Following the definitions established above leads to the simplified result in Equation (2.27) where $\boldsymbol{\xi}_b$ is the random difference between the estimator $\hat{\mathbf{b}}$ and true bias $\mathbf{b}$.

$$\begin{aligned} \mathbf{x}_{\Delta,i} - \hat{\mathbf{b}} &= \mathbf{x}_{\text{S1},i} - \mathbf{x}_{\text{S2},h_i} - (\mathbf{b} + \boldsymbol{\xi}_{\mathbf{b}}) \\ &= \boldsymbol{\xi}_i - (\boldsymbol{\xi}_{h_i} - \mathbf{b}) - (\mathbf{b} + \boldsymbol{\xi}_{\mathbf{b}}) \\ &= \boldsymbol{\xi}_i - \boldsymbol{\xi}_{h_i} - \boldsymbol{\xi}_{\mathbf{b}} \end{aligned} \tag{2.27}$$

Define $\boldsymbol{\xi}_{i,h_i} \coloneqq \boldsymbol{\xi}_i - \boldsymbol{\xi}_{h_i}$ for ease of notation. The terms $\boldsymbol{\xi}_{i,h_i}$ and $\boldsymbol{\xi}_{\mathbf{b}}$ are the aforementioned detection and bias errors, respectively. Recall that $\boldsymbol{\xi}_{i,h_i}$ and $\boldsymbol{\xi}_{\mathbf{b}}$ were assumed independent in

the GNP problem statement. The covariance matrix of $\boldsymbol{\xi}_{i,h_i}$ was previously defined as $\mathbf{Q}_{\Delta,i}$, and the covariance matrix of $\boldsymbol{\xi}_{\mathbf{b}}$ is $\hat{\mathbf{Q}}_{\mathbf{b}}$ as given in Equation (2.16). The covariance of the combined error is then $\mathbf{Q}_{\Delta,i} + \hat{\mathbf{Q}}_{\mathbf{b}}$. This implies that the variance of the quadratic form term in Equation (2.26) is given by Equation (2.28). The expected value is also given in Equation (2.29).

$$\begin{aligned} \sigma^2_{\text{quad}} &\coloneqq \text{var}\left[(\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})' \mathbf{Q}^{-1}_{\Delta,i} (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})\right] \\ &= 2 \cdot \text{tr}\Big[\mathbf{Q}^{-1}_{\Delta,i} (\mathbf{Q}_{\Delta,i} + \hat{\mathbf{Q}}_{\mathbf{b}}) \mathbf{Q}^{-1}_{\Delta,i} (\mathbf{Q}_{\Delta,i} + \hat{\mathbf{Q}}_{\mathbf{b}})\Big] \end{aligned} \tag{2.28}$$

$$\begin{aligned} \mu_{\text{quad}} &\coloneqq \mathbb{E}\left[(\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})' \mathbf{Q}^{-1}_{\Delta,i} (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})\right] \\ &= \text{tr}\left(\mathbf{Q}^{-1}_{\Delta,i} (\mathbf{Q}_{\Delta,i} + \hat{\mathbf{Q}}_{\mathbf{b}})\right) \end{aligned} \tag{2.29}$$

The distribution of a quadratic form consisting of a Gaussian vector and positive-definite matrix is known to follow a generalized χ^2 distribution [71]–[73]. Figure 2.4 shows how some generalized χ^2 statistics scale relative to the ratio of the sensor bias' covariance to the sensor measurements' covariance. Plots of the distribution's PDF can be observed in Figures 2.9 in Section 2.4.

A final consideration is that the quadratic form error in Equation (2.17) is the result of a sum of such generalized χ^2 variables, where the number of terms in the sum is the number of associated pairs. Fortunately, the distribution of a sum of generalized χ^2 variables is also a generalized χ^2 variable. This can be seen by rewriting the sum as a concatenated vector and block diagonal matrix as shown in Equation (2.30). Therefore, the error on the quadratic form sum term in Equation (2.17) follows a generalized χ^2 distribution. The sum distribution allows one to compute a bound on how likely it is that the total cost error exceeds a given

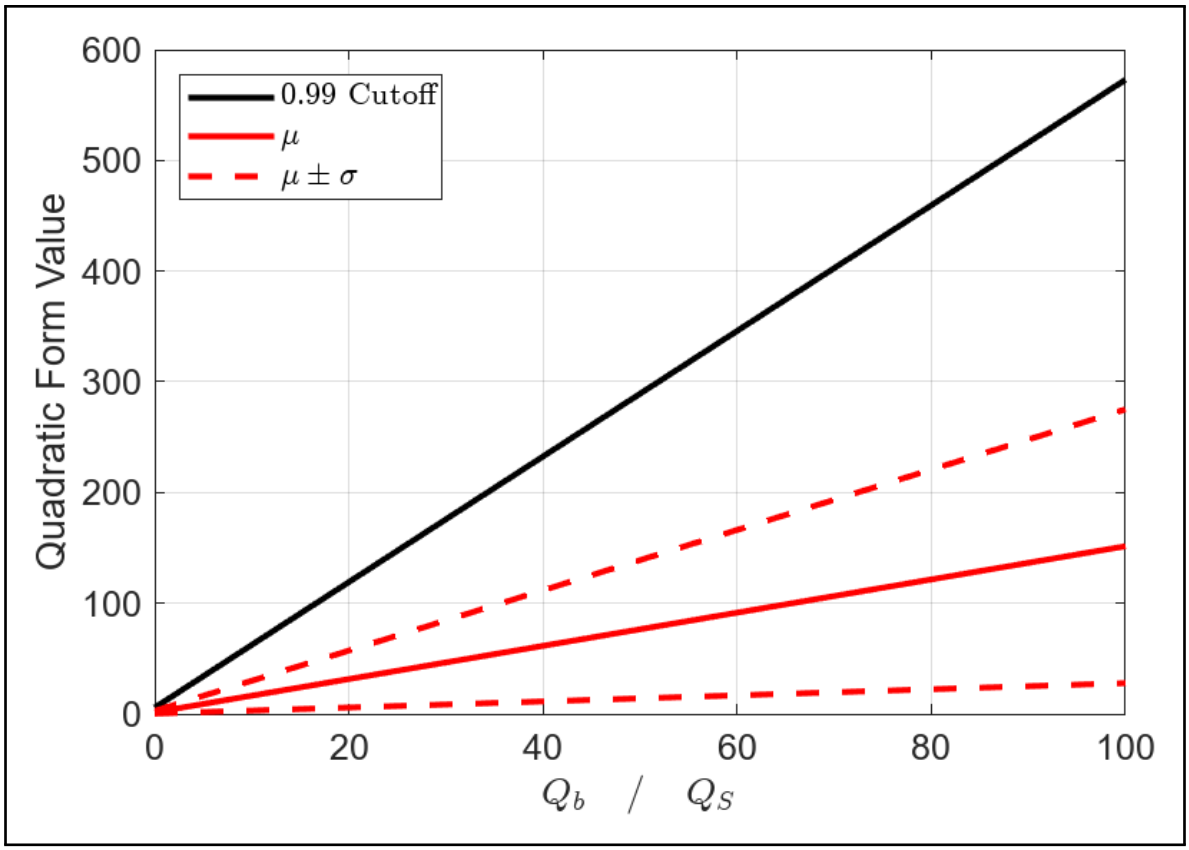


Figure 2.4: A comparison of the mean and 99% cumulative distribution function (CDF) cutoff are plotted against the bias covariance to sensor covariance ratio assuming both sensors have equal covariances. The sensor covariance was taken as the identity matrix and the x-axis denotes the ratio of the scalar multiples for $\mathbf{Q_b}$ and $\mathbf{Q_S}$.

value.

$$\sum_{i=1}^{n} \gamma_i' \mathbf{A}_i \gamma_i = \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_n \end{bmatrix}' \begin{bmatrix} \mathbf{A}_1 & & & \\ & \mathbf{A}_2 & & \\ & & \ddots & \\ & & & \mathbf{A}_3 \end{bmatrix} \begin{bmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_n \end{bmatrix} \tag{2.30}$$

Suppose that a best hypothesis h^* has been chosen, and we now wish to ask how likely it is that another hypothesis h should have been chosen instead. Define the concatenated

difference vectors and block diagonal matrices as follows:

$$\mathbf{X}^* \coloneqq [(\mathbf{x}_{\Delta,1} - \hat{\mathbf{b}})', (\mathbf{x}_{\Delta,2} - \hat{\mathbf{b}})', ..., (\mathbf{x}_{\Delta,n_{a,h^*}} - \hat{\mathbf{b}})']' \qquad \text{for } h^* \tag{2.31}$$

$$\mathbf{Q}_\Delta^* \coloneqq \text{diag}(\mathbf{Q}_{\Delta,1}, \mathbf{Q}_{\Delta,2}, ..., \mathbf{Q}_{\Delta,n_{a,h^*}}) \qquad \text{for } h^* \tag{2.32}$$

$$\mathbf{X} \coloneqq [(\mathbf{x}_{\Delta,1} - \hat{\mathbf{b}})', (\mathbf{x}_{\Delta,2} - \hat{\mathbf{b}})', ..., (\mathbf{x}_{\Delta,n_{a,h}} - \hat{\mathbf{b}})']' \qquad \text{for } h \tag{2.33}$$

$$\mathbf{Q}_\Delta \coloneqq \text{diag}(\mathbf{Q}_{\Delta,1}, \mathbf{Q}_{\Delta,2}, ..., \mathbf{Q}_{\Delta,n_{a,h}}) \qquad \text{for } h. \tag{2.34}$$

Then compute the value of the generalized χ^2 CDF ($G\chi^2$) at zero for the concatenated parameters as shown in Equation (2.35).

$$G\chi^2\left(0; [Z, Z^*]', \mathbf{Q}_\Delta, -\mathbf{Q}_\Delta^*\right) \tag{2.35}$$

This is the probability that, on a different set of realized measurements by the sensors on the same associated targets under the same measurement distributions, the hypothesis h would have a cost less than that of h^*. Of course, this distribution will be different for every pairing since each hypothesis has a distinct distribution.

Computing generalized χ^2 distributions is costly because there is no closed formula for the distribution $G\chi^2$, requiring numerical approximations [73]. The results demonstrated in the example below use code associated with [71], [72]. To compute the 0.99 cutoff depicted in Figure 2.8 required about 0.0356 seconds on the first run and 0.0025 seconds on subsequent runs in MATLAB 2023a using a 12th Gen Intel® Core™ i9-12950HX 2.30 GHz processor. Therefore, to avoid performing this computation thousands of times, a suboptimal but still useful heuristic would be to determine some probability level α and compute the inverse CDF $(G\chi^2)^{-1}$ for the parameters of h^* as shown in Equation (2.36).

$$\alpha \text{ cutoff} = (G\chi^2)^{-1}\left(\alpha; [Z^*, Z^*]', \text{diag}(\mathbf{Q}_\Delta^*, \mathbf{Q}_\Delta^*)\right) \tag{2.36}$$

The parameter is effectively twice the hypothesis error of h^*. The α cutoff is therefore the maximal difference in cost that would be expected in α percent of realizations. Therefore, a confidence interval of $[C_{\text{prior}}, C_{\text{prior}} + \alpha \text{ cutoff}]$ can be computed and used to determine which hypotheses can be soundly rejected and which warrant some further consideration. Using the confidence interval, one could then either consider all hypotheses within the confidence interval or simply quantify the confidence in the single best hypothesis.

2.4 Literature Example

The first example in [64] provides a simple scenario with 11 targets in order to highlight a fact about the stability of the gating criterion. The quantities for the example are defined in Equation (2.37) and assumed as given.

$$\begin{gathered} \mathbf{Q}_{\text{SA},i} = \mathbf{Q}_{\text{SB},j} = \text{diag}(0.3, 0.3, 0.3) \\ \mathbf{b} = [-0.8045, 2.6427, 0.8549]' \\ d = 3 \qquad V = \frac{4}{3}\pi(10)^3 \qquad \beta = 11/V \\ P_{\circ\circ} = 3/11 \qquad P_{\bullet\circ} = 8/11 \qquad P_{\circ\bullet} = 6/11 \end{gathered} \tag{2.37}$$

Let the 11 targets be uniformly distributed in a sphere of radius 3 (the actual radius was not given in [64]) and suppose Sensor SA detects 6 targets, Sensor SB detects 8 targets, and 3 of the targets are detected by both sensors. The corresponding G_0 for these parameters is approximately 0.0127 as given in [64]. However, the G_0 quantity defined in the paper seems to be incorrect or at least does not provide a useful result in the example as stated in [64]. Recall that G_0 in [64] was defined without accounting for distance units. In the original example, a value of $G_0 = 0.0127$ was computed. Using the criterion put forth in Equation (46) of [64], this G_0 value would mean that any feasible association would have to satisfy

Equation (2.38), which is the gating criterion given in [64].

$$(\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}})' \mathbf{Q}_{\Delta,i}^{-1} (\mathbf{x}_{\Delta,\ell} - \hat{\mathbf{b}}) < 2 \cdot \log(G_0) - \log(\det(\mathbf{Q}_{\Delta,i})) \tag{2.38}$$

$$= 2 \cdot \log(0.0127) - 3 \log(0.3) \tag{2.39}$$

$$< -5 \tag{2.40}$$

Note that the left-hand side is a quadratic form with a positive definite matrix, implying it is impossible for the result on the left-hand side to be less than 0. Moreover, only the second term on the right-hand side is dependent on the scale of the chosen units. If the $\mathbf{Q}_{\Delta,i}$ given above was in square-meters, then changing to square-kilometers ideally shouldn't change the gating decisions. However, $0.3\,\mathrm{m}^2$ converts to $3 \times 10^{-7}\,\mathrm{km}^2$ meaning the second term on the right hand side would be $3\log(3 \times 10^{-7})$. As a result, the right-hand side of Equation (2.38) would change from being less than -5 to more than 36. A sufficiently well-chosen bias estimate would make the left-hand side near 0, which shows there are now feasible associations manufactured by a simple change in units. This problem does not occur with the inequalities proposed in Equation (2.25) because changes in units will evenly scale the V and $\det(\mathbf{Q}_{\Delta,i})$ terms. The correctness of Equation (2.25) will be demonstrated below.

The values for the example above are used below with the goal of showing the criterion derived as part of this work correctly thresholds against infeasible associations while permitting correct associations. To simulate the assigned detection probabilities, three targets are chosen to be mutually detected by both sensors. The indices for these three chosen targets were 1, 2, 3 among Sensor SA state estimates and 6, 7, 8 among Sensor SB state estimates. The scenario is depicted in Figure 2.5. Histograms of the pairwise differences are shown in Figure 2.6, along with the estimated and true bias.

Applying the gating criterion derived above, the resulting table of booleans indicating feasible or infeasible association pairings is shown in Table 2.2. Here the row indices correspond to the detection index from Sensor SA and the column indices correspond to the

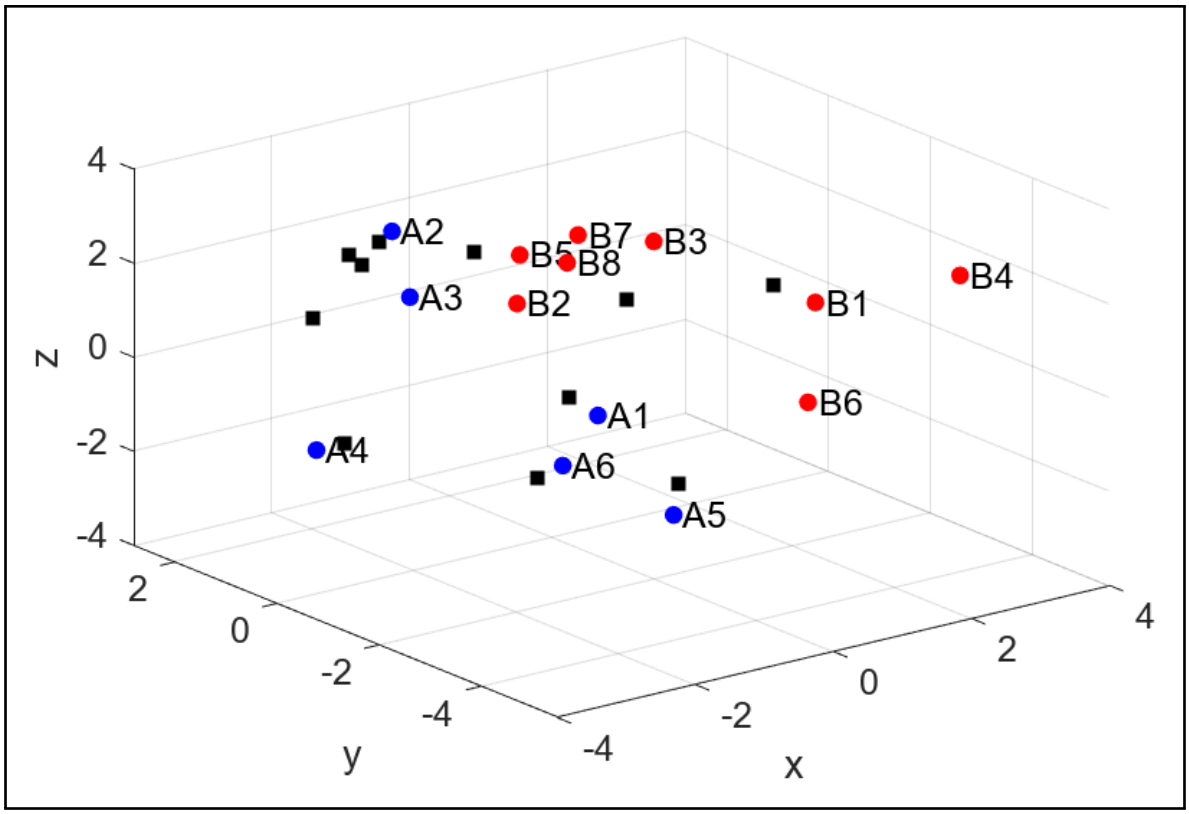


Figure 2.5: The generated scenario. Blue dots belong to Sensor A and red dots belong to Sensor B. The black squares are the true target locations.

detection index from Sensor SB. The diagonal of (A1,B6), (A2,B7), and (A3,B8) having ones shows that the correct hypothesis is available from the remaining hypotheses. After computing the feasible pairings shown in Table 2.2, Equation (2.17) must be optimized over the remaining hypotheses. This requires computation using the full hypotheses, so the prior bias $\hat{\mathbf{b}}_{\text{avg.}}$ should be replaced with the posterior bias $\hat{\mathbf{b}}_{\text{ML}}$. All feasible hypotheses (choices of feasible pairings that respect the assumption of one detection per target per sensor) are shown sorted in Figure 2.8. The optimal solution (minimal cost hypothesis) is shown in Figure 2.7 and the cost of the best hypothesis $[6, 7, 8, 0, 0, 0]$ is identified as -86.7406. Visual inspection of Figure 2.7 might suggest other hypotheses which should yield similar costs. For example, the cost of the alternative hypothesis $[6, 5, 2, 0, 0, 0]$ (assigning B5 to A2 and B2 to A3) is -80.3959. This and similar observations verify that similar hypotheses do correspond to similar costs. The posterior bias and bias covariance are given in Equation (2.41).

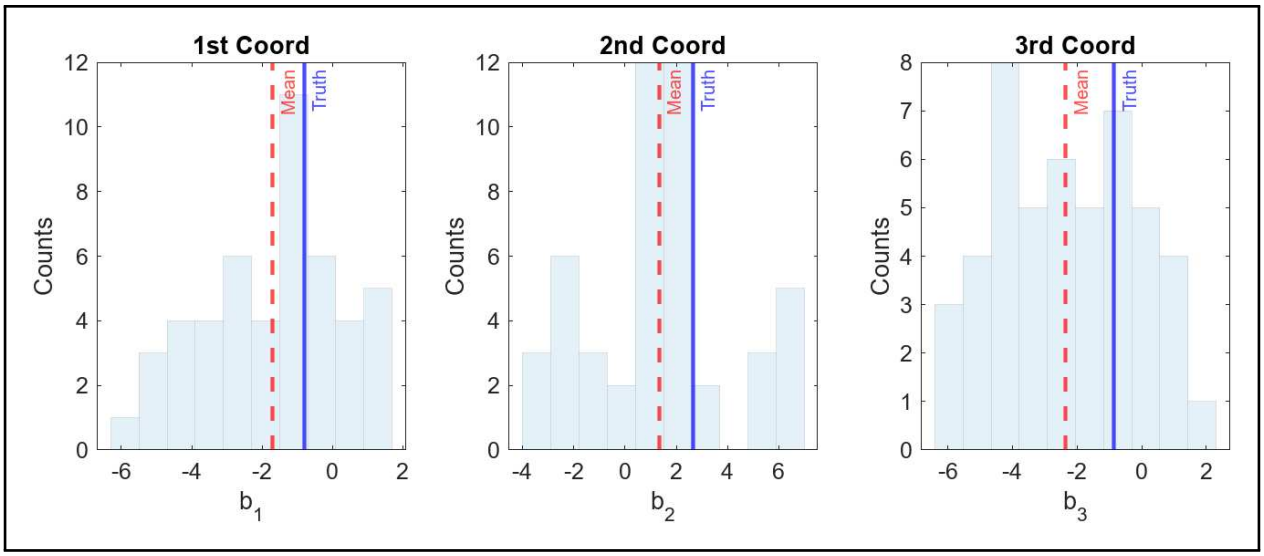


Figure 2.6: Estimates for each component of the bias using pairwise differences between all state estimates. The estimated bias is $\hat{\mathbf{b}}_{\text{avg.}} = [-1.7027, 1.3295, -2.3477]'$ and the true bias is $\mathbf{b} = [-0.8045, 2.627, -0.7549]'$.

Table 2.2: The GNP gating results using the example in Figure 2.5 and Equation (2.25).

	B1	B2	B3	B4	B5	B6	B7	B8
A1	0	0	0	1	0	1	0	1
A2	0	0	0	0	1	0	1	1
A3	0	1	1	0	1	0	1	1
A4	0	1	0	0	1	0	1	1
A5	0	0	0	1	0	1	0	0
A6	1	0	1	0	0	0	0	0

$$\begin{aligned} \hat{\mathbf{b}}_{\text{ML}} &= [-0.8886, 2.3656, -1.0361]' \\ \hat{\mathbf{Q}}_{\mathbf{b}} &= \text{diag}(0.0592, 0.0592, 0.0592) \end{aligned} \tag{2.41}$$

Finally, the quadratic form error can be computed using the new $\hat{\mathbf{Q}}_{\mathbf{b}}$ and compared to the corresponding generalized χ^2 distribution. Plugging in $\mathbf{Q}_{\Delta,i}$, $\hat{\mathbf{Q}}_{\mathbf{b}}$, and $\hat{\mathbf{b}}$ to Equations (2.28) and (2.29) yields prior values of $\sigma_{\text{quad}} \approx 61.2372$ and $\mu_{\text{quad}} = 75.0000$ and posterior values of $\sigma_{\text{quad}} \approx 3.2548$ and $\mu_{\text{quad}} = 3.9863$. Realized values of the quadratic form in both

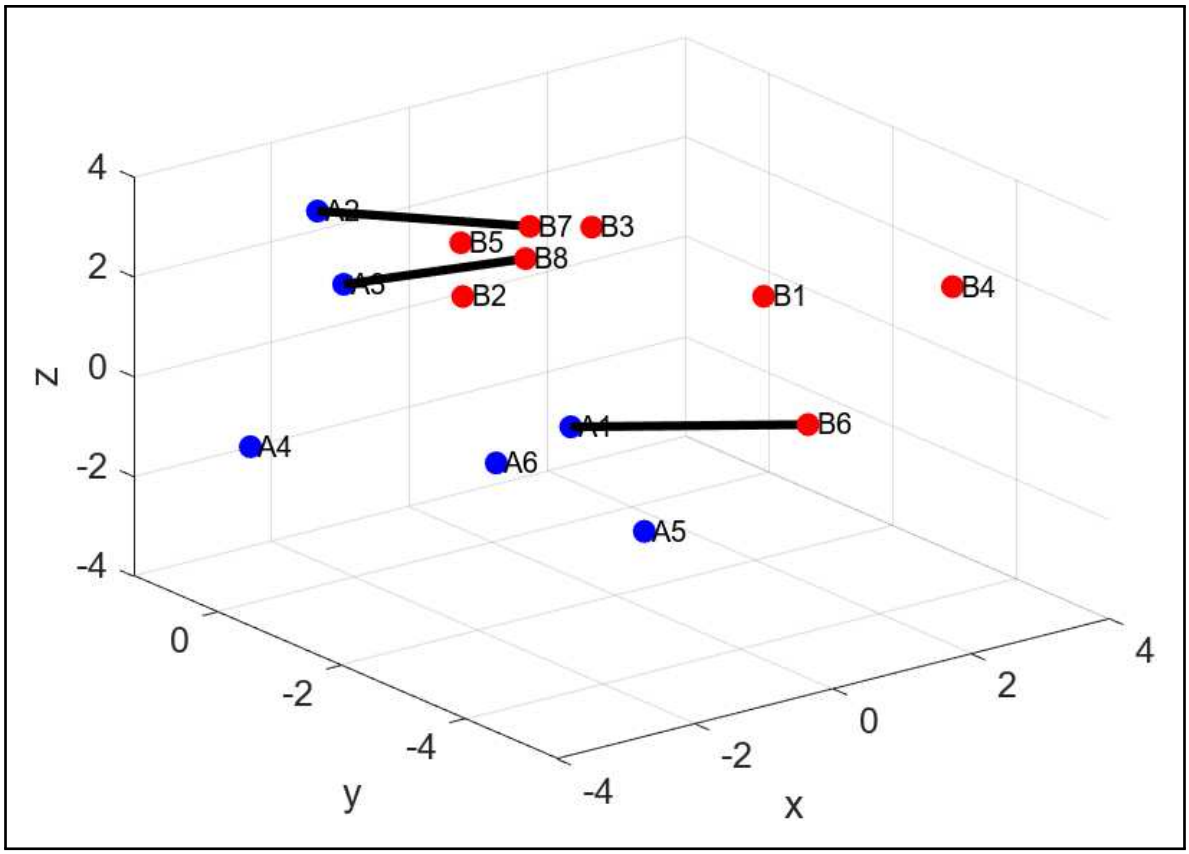


Figure 2.7: The lowest cost global association hypothesis with pairings corresponding to the same target denoted by a black line.

instances are given in Table 2.3. The results demonstrate a marked improvement in the likely posterior quadratic form error which follows the distribution in Figure 2.9.

A confidence interval on the optimal cost can be defined using the generalized χ^2 distribution for the posterior bias estimates. Following the heuristic given at the end of Section 2.3.3, we choose to use the 99% confidence value as the cutoff. Figure 2.8 shows the 0.99 cutoff line and how it compares to the actual hypothesis costs. These lines could be used to prune unlikely hypotheses and maintain others over extended intervals of time. Alternatively, if only a single hypothesis is maintained, then a degree of confidence in that association is afforded by computing how likely it is that the chosen hypothesis was not the correct one.

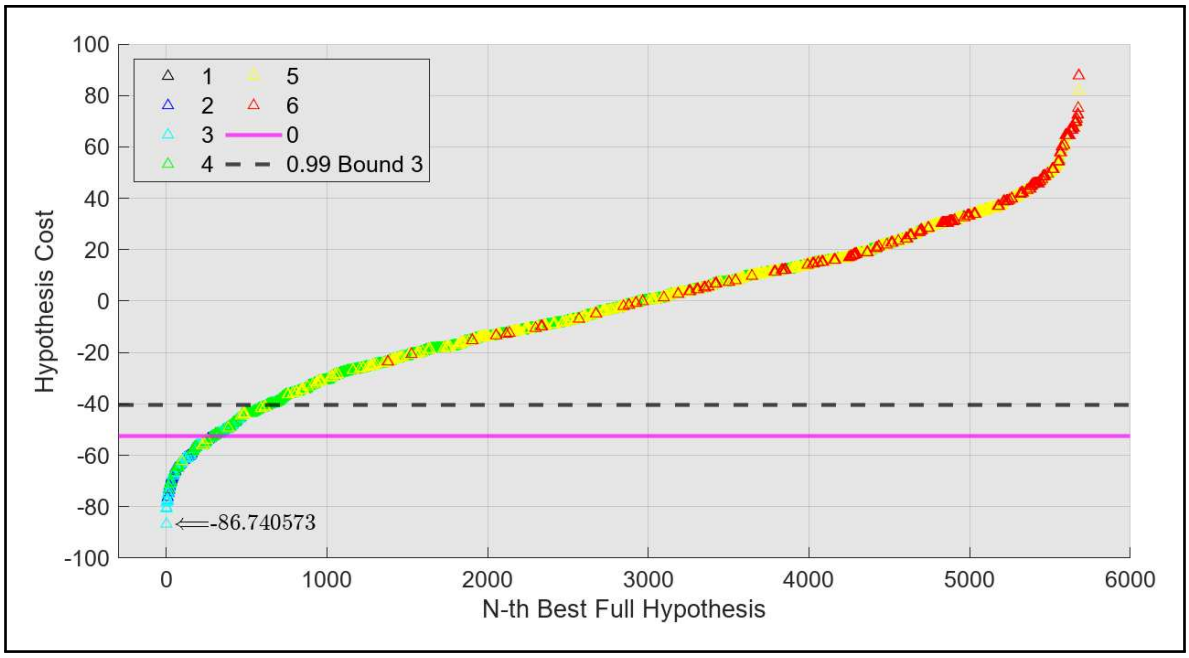


Figure 2.8: Sorted feasible hypotheses color coded by number of associated state estimates. The markers indicate the cost of each hypothesis and the dashed line indicates the 99% bound on twice the error of the best hypothesis.

2.5 Bias Correction

The example in Section 2.4 demonstrated the ability to make a correct association on a single time step. Now, an example will be given that demonstrates the potential of the GNP algorithm to iteratively correct for error and keep the sensor bias within measurement error.

The example trajectory will be based on parameters modified from [74]. The 2-dimensional scenario will have an initial position of $[60\,\mathrm{km}, 40\,\mathrm{km}]'$ and an initial velocity of $[-300\,\mathrm{m\,s^{-1}}, 300\,\mathrm{m\,s^{-1}}]'/\sqrt{2}$ and will proceed by a sequence of transition models as follows:

1. Constant velocity of $300\,\mathrm{m\,s^{-1}}$ for $50\,\mathrm{s}$,
2. Coordinated turn with a turn rate of $4.68\,\mathrm{deg}$ for $96\,\mathrm{s}$,
3. Constant velocity of $300\,\mathrm{m\,s^{-1}}$ for $30\,\mathrm{s}$,
4. Coordinated turn with a turn rate of $-2.8\,\mathrm{deg}$ for $64\,\mathrm{s}$,
5. Constant velocity of $300\,\mathrm{m\,s^{-1}}$ for $30\,\mathrm{s}$.

Table 2.3: Some quadratic form values computed using Equation 2.26. The first 3 rows demonstrate the correctly associated pairings. The subsequent rows demonstrate quadratic form values for some other hypothetical pairings.

SA Index	SB Index	Prior Value	Posterior Value
A1	B6	19.6897	0.6717
A2	B7	25.8521	0.7414
A3	B8	15.4428	1.7914
A4	B4	184.4526	217.0623
A2	B3	62.1329	49.6419
A2	B5	39.3955	12.8962
A1	B3	134.6161	202.2626

For simplicity, the motion model noise will be set to zero and the measurement interval will be a constant 10 s. To simulate 10 closely-spaced targets, the trajectory described above will be duplicated and shifted by a random Gaussian vector sampled from $\mathcal{N}(\mathbf{0}, \mathrm{diag}(1000\,\mathrm{m}, 1000\,\mathrm{m}))$.

The remaining parameters are defined in Equation (2.42). For each time step, the measurements will be randomly sampled from Gaussian distributions centered on each of the true target positions. Probabilities of detection will also be simulated by randomly selecting which targets each sensor detects. The described scenario is shown in Figure 2.10 and a realized example of the initial measurements can be seen in Figure 2.11a.

$$\begin{gathered}
\mathbf{Q}_{\mathrm{SA},i} = \mathbf{Q}_{\mathrm{SB},j} = \mathrm{diag}(100\,\mathrm{m}^2, 100\,\mathrm{m}^2) \\
\mathbf{b}_0 = [100\,\mathrm{m}, 230\,\mathrm{m}]' \\
d = 2 \qquad V = 4.189 \times 10^3\,\mathrm{m}^3 \qquad \beta = 10/V \\
P_{\circ\circ} = 0.6 \cdot 0.8 \qquad P_{\bullet\circ} = (1 - 0.6) \cdot 0.8 \qquad P_{\circ\bullet} = 0.6 \cdot (1 - 0.8)
\end{gathered} \tag{2.42}$$

The goal will be to use the scenario generated in Figure 2.10 to demonstrate bias reduction. On each time step, the generated measurements will be fed into the GNP algorithm described above. After a posterior bias estimate is computed, the posterior bias will be

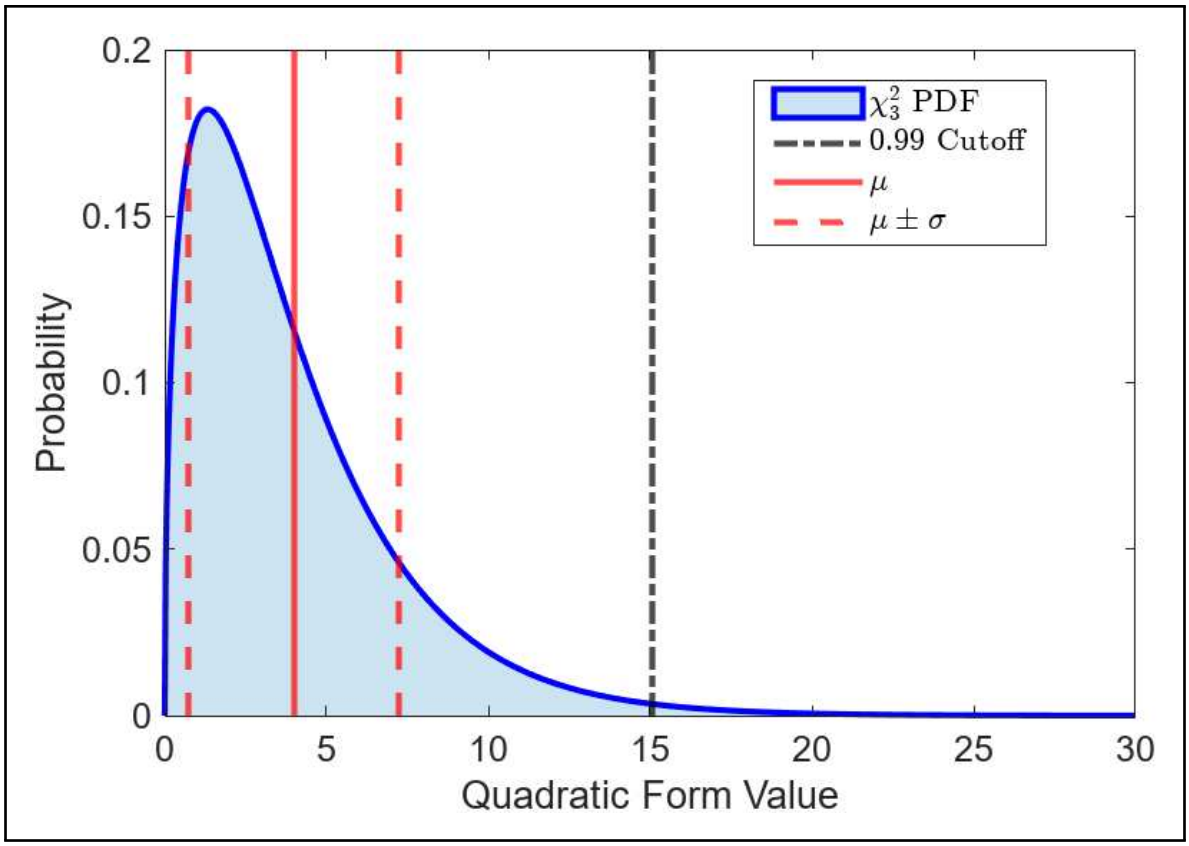


Figure 2.9: This shows the generalized χ^2 PDF for the given example assuming the posterior bias covariance $\hat{\mathbf{Q}}_\mathbf{b}$. The mean and the standard deviation interval, as computed by Equations (2.29) and (2.28), are also plotted as red lines for comparison. The black line marks the 99% cutoff as determined by the inverse CDF of the distribution generalized χ^2 distribution.

subtracted from $\mathbf{b}_0$ and the result will be used as the sensor bias in the next iteration. To determine whether an operator who only sees the estimated bias calculation could consider the bias to be within measurement error, an observed bias estimate will be provided via a linear Kalman filter. The observed bias estimate given the posterior bias data computed by the GNP algorithm will not be used to adjust the bias, only to monitor when it has likely been reduced to within measurement error (10 m). Finally, even with the gating criterion presented above, there are still GNP problems which yield too many hypotheses to check. To keep the check simple, if the number of hypotheses to consider exceeds 10,000, then the null hypothesis will be assumed and the posterior bias will be set to $\mathbf{0}$. The number of hypotheses

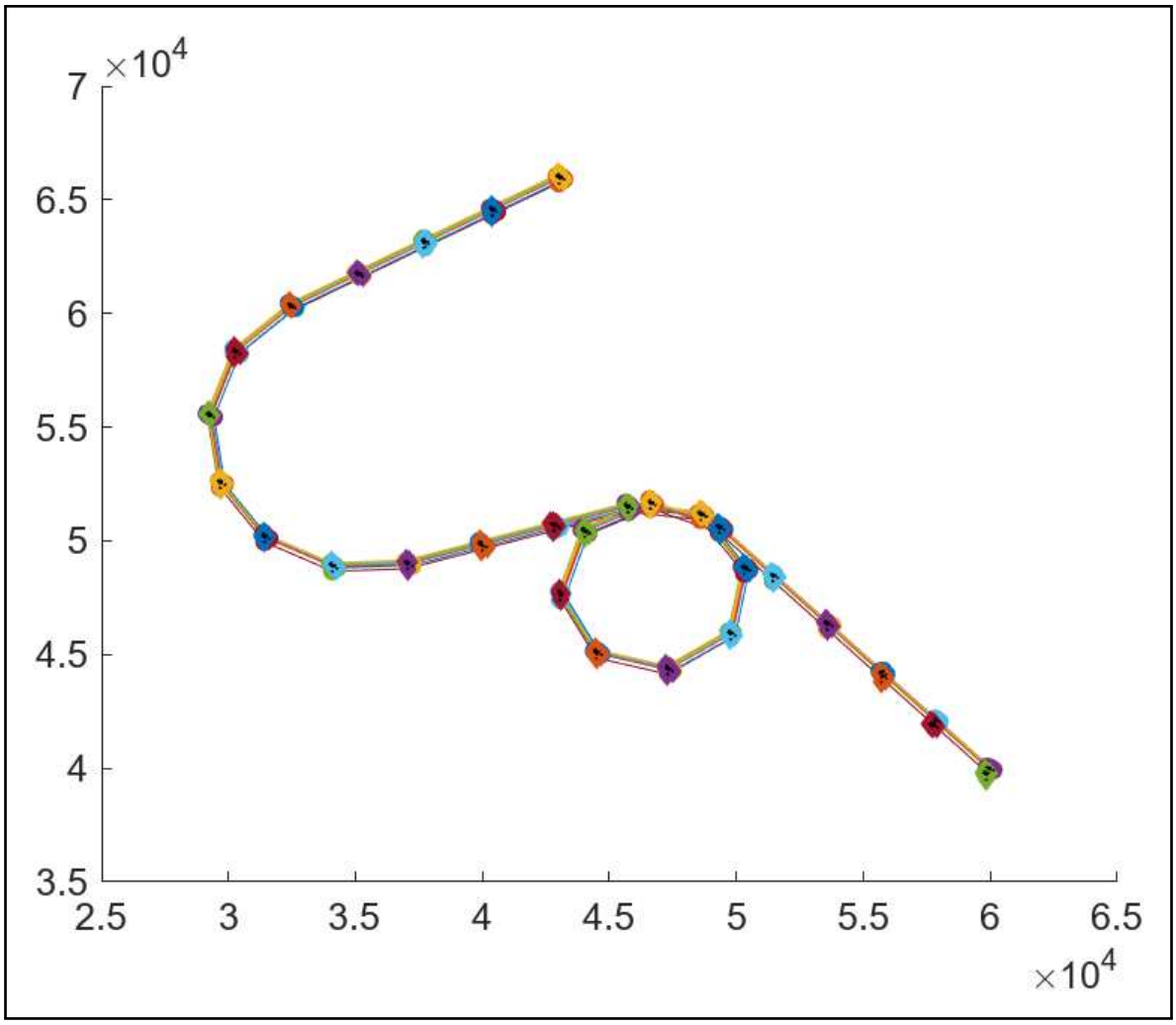
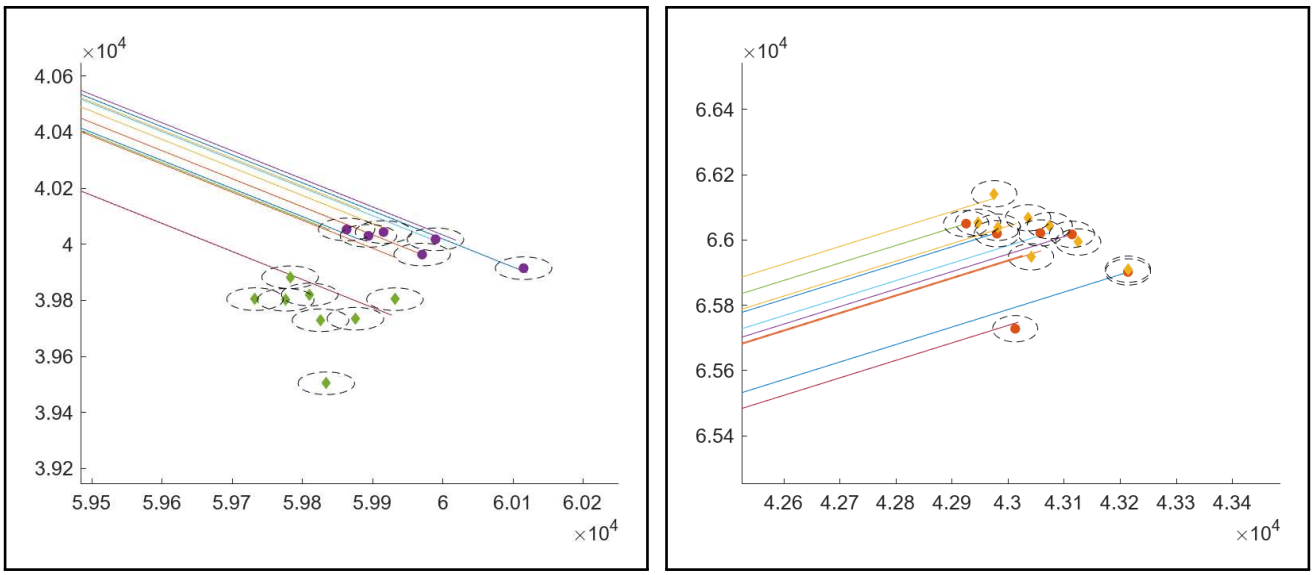


Figure 2.10: The generated scenario with lines drawn between each truth position for each of the 10 targets and markers placed to show where measurements are taken. The trajectory starts in the bottom right of the figure and ends at the top.

to consider will be computed as the product over the number of feasible associations from SB to each measurement in SA.

Looking at the beginning to end results (first time step to last time step) in Figure 2.11, it is clear that the GNP algorithm has enabled a reduction in the sensor bias. The diamonds and circles are well within the confidence ellipses indicating alignment within the bounds of measurement error in Figure 2.11b. Also, it can be observed in the truth trajectories that the targets do change their relative spacing over the course of the scenario, which demonstrates some robustness of the algorithm. A breakdown of the association performance over time can

be found in Figure 2.12. Given the fact that each set of measurements perfectly reflects the sensors' respective probabilities of detection (0.6 and 0.8), there is always guaranteed to be at minimum four pairs of measurements that should be assigned to the same target. So when the first column of association counts in Figure 2.12 shows all unassigned measurements, that is an example of the hypotheses bound check being exceeded and the null hypothesis being assumed. The mean percent correct assignment decisions (whether associated or not) is 64%.



(a) **(b)**

Figure 2.11: Initial measurements and true target positions are shown in (a) and terminal measurements and true target positions are shown in (b). The unbiased sensor measurements in (a) and (b) are circles and biased sensor measurements are diamonds.

Despite the frequency of incorrect decisions, Figure 2.13 demonstrate that there is an immediate payoff when the GNP algorithm does make the correct assignments. Moreover, even with only using the instantaneous posterior bias estimate (as opposed to an estimator which accounts for a window of time), one can get reasonable bias reduction and expect the reduced bias to remain close to measurement accuracy. Using the estimated bias' covariance

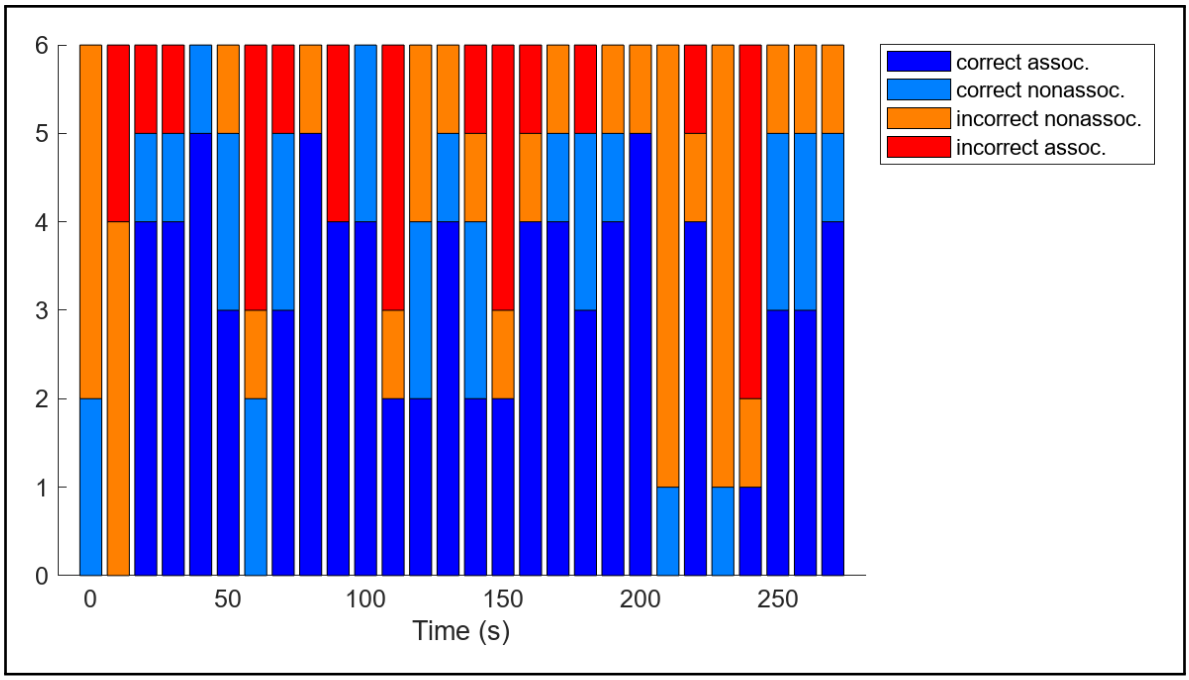


Figure 2.12: A comparison of the association performance over time. Generally correct association decisions are in blue or azure and generally incorrect decisions are in red or orange. The lighter colors (orange and light blue) denote decisions to not associate and darker colors (red and blue) denote decisions to associate.

output by the Kalman filter, one could establish when the bias has been sufficiently reduced depending on the desired accuracy of the measurement.

2.6 Conclusion

An improved gating criterion for GNPM was derived and applied to two examples. It was demonstrated that the proposed criterion correctly accounts for volume units and sifts out infeasible associations while maintaining correct associations. The resulting reduction in feasible hypotheses yields a more tractable start for the full cost comparison that is required for GNPM or any other association algorithm which accounts for bias. The criterion was given in Equation (2.25) and estimates for the bias term are provided in Equations (2.14), (2.15), and (2.16). A new error analysis was also provided, enabling the quantification of how likely the selected GNPM hypothesis is the correct one. Combining the GNPM cost function, the gating criterion, and the error estimation procedure presented here will allow

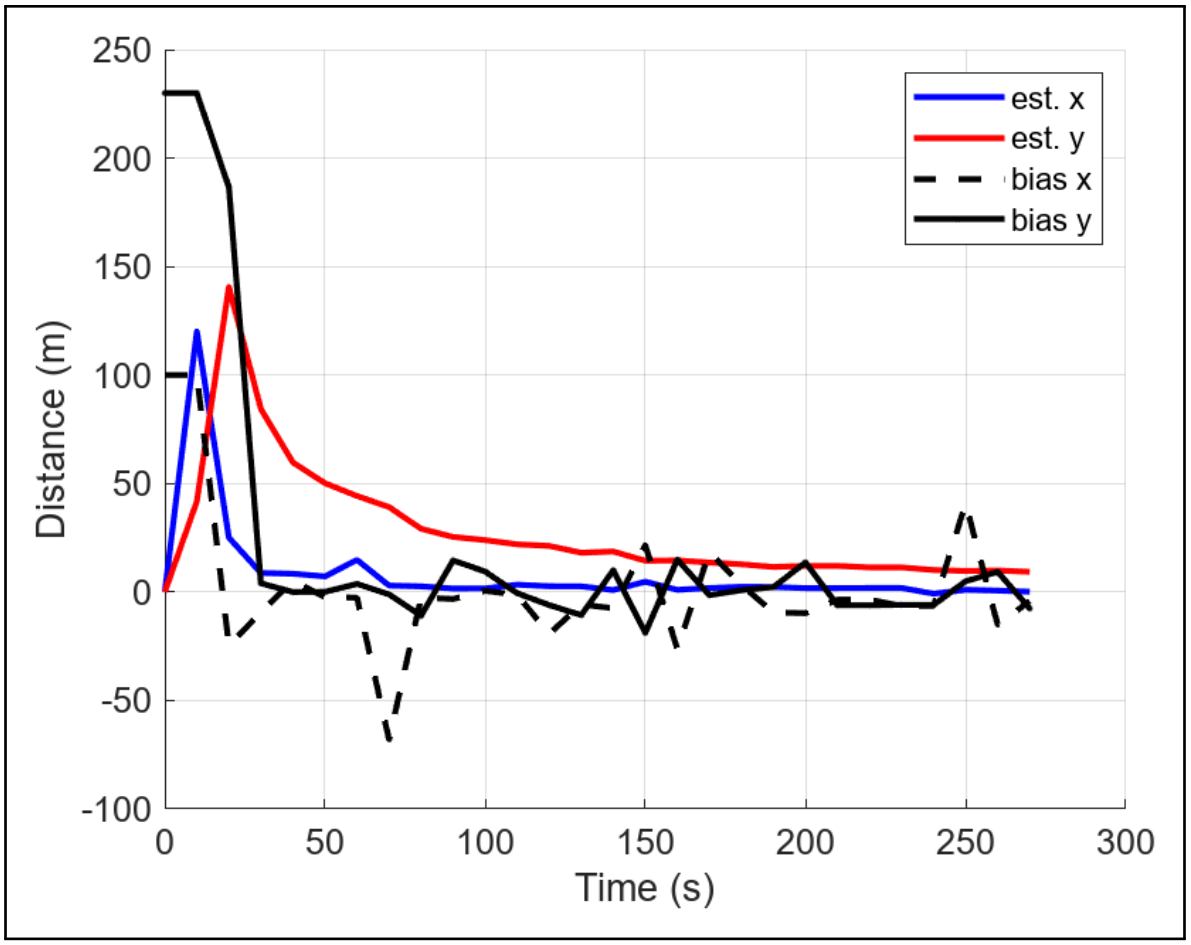


Figure 2.13: A comparison of the true bias and estimable bias given the observed measurements.

for rapid estimation of sensor bias, which can then be used to improve tracking association via bias reduction.

Chapter 3

A Neural Network Multiple Model Filter

3.1 Introduction

In target tracking, the interacting multiple model (IMM) filter has been widely used as an efficient algorithm for robust estimation [1], [75]–[77]. For predicting and updating a track state, IMM filters were designed to allow for consideration of multiple hypothesized target motion models [37]. In contrast, a single Kalman filter, or other such single-hypothesis estimators, requires a premature assumption that the tracked target follows one chosen motion model. Later work on efficient estimation algorithms have compared the IMM favorably to competing methods such as the generalized pseudo-Bayesian (GPB) algorithms in terms of computational cost, robustness, and accuracy [78]. The IMM algorithm has better accuracy than the GPB1 (first-order GPB) algorithm with both having $\mathcal{O}(N_M)$ complexity, with N_M being the number of motion models under consideration. Compared to the GPB2 (second-order GPB) algorithm, IMM is worse but has faster runtimes due to the $\mathcal{O}(N_M^2)$ complexity of GPB2 [3].

However, there are some known issues with the traditional IMM approach which leads to desires for improvements. First, the truncation of the mixing process at a first-order Markov chain leads to estimation errors which can accumulate over time. Efforts to reduce this error were made by extending the truncation to higher-ordered Markov chain processes; however, the implementation of these methods requires increasing amounts of on-line processing power [79]. Another problem is that placing the Markov assumption on the motion model switching dynamics leads to a requirement to anticipate switching probabilities. This imposes target maneuver assumptions on the tracker before any positive identification of the target has been made. One approach to fix this problem is to introduce a neural network which can make predictions without necessitating those design considerations. Of course, this opens a

Pandora's box of other concerns about what assumptions get baked into the neural network. The following sections will present considerations for designing a neural network for use with an IMM.

Using neural networks to aid in the target tracking problem is not new [80], [81]. Even in the narrow problem of IMM estimation, past work has run the gamut from merely preceding the IMM algorithm with a classifier to correcting the output of an IMM with a neural network to completely replacing the IMM algorithm with a neural network which takes in a track history and outputs future track predictions [78], [82]–[86]. One recent example of fully replacing the IMM with a neural network model is given in [86], which uses and encoder-decoder architecture to get promising results. However, in proposals which replace the IMM algorithm entirely, there is little intuition behind what the neural network is actually doing. For that reason, restricting the role of the neural network can be appealing and is the approach chosen below. Many of the attempts at improving IMM via pre or post-processing with a neural network use some form of recurrent neural network (RNN) such as a long short-term memory (LSTM) network. This approach is reasonable and may prove to be the best approach if efficiency is the only goal, but RNNs complicate the analysis of how inputs get processed into outputs. Taking into account the desire for interpretable models (see [87], [88]), the simpler feed-forward architecture is used in the work presented below.

The key difference between the past literature and what is presented here is that the approach below no longer requires a transition matrix designed using model switching assumptions. The approach below requires only that one know which motion models are allowable (a physics-based problem) and what measurement parameters will be used (a sensor design problem). These are problems which are possible to solve prior to deployment of a tracker. Any algorithm which requires a transition matrix based on switching assumptions must additionally issue claims about how a future target is likely to behave (a tactics problem). In many cases, the assumption that tactics information can be known before deploying the tracker is dubious. Therefore, while the use of neural networks with an IMM may be similar

to prior work, it is believed that the work below provides a more sound implementation by eliminating a subjective design element and providing a more trustworthy algorithm.

The sections that follow will describe and demonstrate the new neural network multiple model (NNMM) estimator as an improvement over the standard IMM algorithm. Section 3.2 provides descriptions of motion models, the IMM filter, and some justification of the choice of inputs for the neural network classifier (NNC). Section 3.3 continues a description of the choices made for creating a NNC which can be used to implement the proposed NNMM estimator. Section 3.4 demonstrates the NNMM estimator on an example scenario.

3.2 Background

3.2.1 Target Motion Models

The focus in this chapter will be on the estimation part of the tracking process, after detections have been identified and associated with an existing track. There are several commonly used target motion models in the literature [23]. The discussion that follows will be limited to 2-dimensional cases for ease of discussion, but extension to 3 dimensions only requires a few changes. The tracking literature exhibits a preference for linear motion models due to simplicity and computational speed so that other parts of the tracker can run before the next batch of measurements arrives. We have similar preferences here and will choose to also rely on linear models. This choice allows the process function from Equation (1.5) to be rewritten as:

$$\mathbf{x}_{k|k} = \mathbf{F}\mathbf{x}_{k-1|k-1} + \boldsymbol{\omega}_{k-1} \tag{3.1}$$

where $\mathbf{F}$ is a $\mathbb{R}^{d_x \times d_x}$ matrix. The default model in most cases is a constant velocity (CV) model represented by the matrix:

$$\mathbf{F}_{\text{CV}} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

for $\mathbf{x} = [\text{position, velocity}]'$, also sometimes referred to as the nonmaneuvering model. In trackers which try to use process noise to account for deviations from a single motion model, the CV model is typically used as the baseline motion model [3], [24]. A larger class of models which includes the CV model are the polynomial motion models, which are polynomial combinations of the state vector (the kinematic terms specifically). The class of polynomial models includes the constant acceleration (CA) model, represented by the matrix:

$$\mathbf{F}_{\text{CA}} = \begin{bmatrix} 1 & 0 & \Delta t & 0 & \Delta t^2 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & \Delta t^2 \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

for $\mathbf{x} = [\text{position, velocity, acceleration}]'$. The CA model can be used to represent a generic maneuver, deviating from the assumed CV model [3], [24]. The final class of models that will be considered here are the coordinated turn (CT) models. These include turns assuming nearly constant radius with respect the axis of rotation and of nearly constant speed and angular rate, which will be used below. A 2-dimensional coordinated turn model with known

turn rate τ is represented by the matrix:

$$\mathbf{F}_{\mathrm{CT}} = \begin{bmatrix} 1 & 0 & \sin(\tau\Delta t)/\tau & (\cos(\tau\Delta t)-1)/\tau \\ 0 & 1 & (1-\cos(\tau\Delta t))/\tau & \sin(\tau\Delta t)/\tau \\ 0 & 0 & \cos(\tau\Delta t) & -\sin(\tau\Delta t) \\ 0 & 0 & \sin(\tau\Delta t) & \cos(\tau\Delta T) \end{bmatrix} \tag{3.4}$$

for $\mathbf{x} = [\text{position, velocity}]'$.

These models are good approximations to most commonly observed targets, and they provide good coverage of possible locations where a feasible target would be found after a short time interval. In what follows, we consider the set of assumed models to be $M \coloneqq \{\text{CV, CA, CT}\}$. It is apparent that the various models defined above require states of different dimensionalities: 4 for CV, 6 for CA, and 5 for CT. In order to store all the information necessary for all models, a 7 dimensional state will be used: $\mathbf{x} = [\text{position, velocity, acceleration, turnrate}]'$. When applying each model, it will be assumed that any kinematic quantity which is not applicable to the model is zero. Therefore, the state will be reduced to the appropriate dimensionality for actual computations. The covariance matrix will likewise be selectively reduced to the applicable dimensions, ensuring positive definiteness.

The measurement process will also require definition before proceeding. We will assume that only position is directly observed: $g(\bullet, \boldsymbol{\nu}_k) : \mathbb{R}^7 \to \mathbb{R}^2$ where the distribution of $\boldsymbol{\nu}_k$ is known based on the sensor. The primary reason for this restriction is academic. In allowing the network to only observe the position, we provide a minimal amount of information for use in the classifier while still enabling straightforward application of the IMM algorithm. If the following shows that position is sufficient for the proposed technique, then systems which do provide measurements of more kinematic quantities will be able to benefit so long as they have position estimates.

3.2.2 Multiple Model Filters

In the event that the target motion is unplanned or stochastic, the true motion is unknown. It is therefore necessary to propose an approximate motion model $\mathbf{F}$ in order to use filtering methods such as Kalman filters for updating of track states. A maximum likelihood approach in choosing $\mathbf{F}$ would be to select whichever model gives the highest probability of the measurement originating from the predicted track (note that $\mathbf{F}$ is used to compute $\mathbf{x}_{k|k}$ from $\mathbf{x}_{k-1|k-1}$):

$$\mathbf{F}_{\mathrm{ML}} \coloneqq \arg\max_{\mathbf{F}} \mathcal{N}(\mathbf{z}_k|\mathbf{x}_k, \mathbf{P}_k). \tag{3.5}$$

However, when tracks are relatively new, the choice of only one model can result in overly confident tracks after the filtering stage, which may lead to missed gating in later association stages. For this reason, better performance can often be achieved by a probabilistic mixing of the results of all models. Estimation with uncertainty about the motion model $\mathbf{F}$ proceeds by computing $\mathbf{x}_k, \mathbf{P}_k$ with respect to several motion models, which produces one track estimate per model. Based on the distance between the observed measurement and each of the predicted track estimates, a set of weights are computed and the multiple track estimates are fused into a single estimate to be propagated for the next update. As noted in section 3.1, a common method for implementing this mixed model update scheme is IMM filtering.

In IMM filtering, a set of motion models M are proposed as approximations to a target's true motion along with prior probabilities $\boldsymbol{\mu} \coloneqq [\mu_1, \mu_2, ..., \mu_{N_M}]'$ for how likely each model is to be the closest match to the true target motion. The set of models $\{\mathbf{F}_m\}_{m \in M}$ is assumed to be exhaustive and finite, so the probabilities of all considered models sum to 1. In addition to the prior probabilities, the IMM must be given a transition matrix $\boldsymbol{\Lambda}$ which describes the switching probabilities for the continuous time Markov chain being described on M as shown in Figure 3.1. Those probabilities are derived from the sojourn times for a target given motion models in M [4], [77], [89], [90]. The relationship between the transition matrix $\boldsymbol{\Lambda}$

and transition rate matrix $\mathbf{J}$ is

$$\mathbf{\Lambda} \coloneqq \exp\{\Delta t \cdot \mathbf{J}\} \quad \text{where } \sum_{\substack{j=1 \\ j \neq i}}^{N_M} \mathbf{J}_{ij} = -\mathbf{J}_{ii} \text{ for all } i = 1, 2, ..., N_M. \tag{3.6}$$

The matrix $\mathbf{J}$ denotes the instantaneous transition rate and is being scaled by the time step Δt which is the time between the last measurement update and the current one. Since that interval may change, it is proper to treat $\mathbf{J}$ as the fundamental quantity and $\mathbf{\Lambda}$ as the derived quantity. Of course, as stated above, the assumption that $\mathbf{J}$ or $\mathbf{\Lambda}$ represent the true target dynamics is dubious except in highly controlled scenarios. Fortunately, the IMM is fairly robust to poor choices of Λ [3].

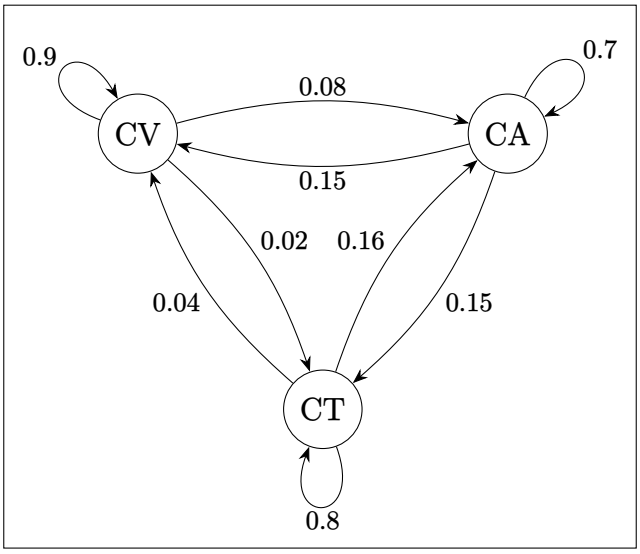


Figure 3.1: An example of a Markov chain on the set $M \coloneqq \{\text{CV, CA, CT}\}$. The elements of $\mathbf{\Lambda}$ are shown on their respective edges. This Markov chain's transition probabilities are later used in Equation (3.16).

Assuming a suitable $\mathbf{\Lambda}$ and establishing an initial prior $\boldsymbol{\mu}$ (e.g. a uniform prior), we define $\tilde{\mathbf{x}}_k^{(m)}$ as the prior state for model m at time step k. The prior states for each motion model (usually given as output from the previous iteration of the IMM filter) are first mixed

in what is called the interaction stage (labeled "State Mixing" in Figure 3.2):

$$\tilde{\mathbf{x}}_{k-1|k-1}^{(m)} \coloneqq \sum_{j=1}^{N_M} \tilde{\mathbf{\Lambda}}_{m,j} \mathbf{x}_{k-1|k-1}^{(m)} \qquad \text{where } \tilde{\mathbf{\Lambda}}_{m,j} \coloneqq \boldsymbol{\mu}_{k,j} \mathbf{\Lambda}_{m,j} \left(\sum_{m=1}^{N_M} \boldsymbol{\mu}_{k,j} \mathbf{\Lambda}_{m,j} \right)^{-1} \tag{3.7}$$

$$\tilde{\mathbf{P}}_{k-1|k-1}^{(m)} \coloneqq \sum_{j=1}^{N_M} \tilde{\mathbf{\Lambda}}_{m,j} \left(\mathbf{P}_{k-1|k-1}^{(m)} + \left(\mathbf{x}_{k-1|k-1}^{(m)} - \tilde{\mathbf{x}}_{k-1|k-1}^{(m)} \right) \left(\mathbf{x}_{k-1|k-1}^{(m)} - \tilde{\mathbf{x}}_{k-1|k-1}^{(m)} \right)' \right). \tag{3.8}$$

These estimates are then given as inputs (along with a single associated measurement) to the respective filters for each of the $m \in M$ motion models, and their respective outputs $\mathbf{x}_{k|k}^{(m)}$ and $\mathbf{P}_{k|k}^{(m)}$ are the posterior states which will be used as prior inputs in the next IMM iteration. After filtering, a quantity called the "innovation" ($\tilde{\mathbf{z}}$), which is computed as a part of most filtering algorithms, will be used to compute the likelihood of each model being the "correct" one:

$$\tilde{\mathbf{z}}_k^{(m)} \coloneqq \mathbf{z}_k - \mathbf{G}\mathbf{x}_{k|k-1}^{(m)} \tag{3.9}$$

$$\tilde{\mathbf{R}}_k^{(m)} \coloneqq \mathbf{G}\tilde{\mathbf{P}}_{k|k-1}^{(m)}\mathbf{G}' + \mathbf{S}_k \tag{3.10}$$

where $\mathbf{G}$ is the matrix representing the linearization of measurement process g. The likelihood of model m having yielded the "correct" prediction for a Gaussian measurement and track is given by:

$$\ell^{(m)} \coloneqq \mathcal{N}(\tilde{\mathbf{z}}_k^{(m)}; \mathbf{0}, \tilde{\mathbf{R}}_k^{(m)}). \tag{3.11}$$

Model probabilities are then updated as follows:

$$\boldsymbol{\mu}_k^{(m)} = \ell^{(m)} \boldsymbol{\mu}_{k-1} \left(\sum_{j=1}^{N_M} \ell^{(j)} \right)^{-1}. \tag{3.12}$$

The filtered states and updated probabilities can now be used to produce a weighted average state which is output as the estimated target state:

$$\hat{\mathbf{x}}_{k|k} = \sum_{m=1}^{N_M} \boldsymbol{\mu}_k^{(m)} \mathbf{x}_{k|k}^{(m)} \tag{3.13}$$

$$\hat{\mathbf{P}}_{k|k} = \sum_{m=1}^{N_M} \boldsymbol{\mu}_k^{(m)} \left(\mathbf{P}_{k|k}^{(m)} + (\mathbf{x}_{k|k}^{(m)} - \hat{\mathbf{x}}_{\mathbf{k}|\mathbf{k}})(\mathbf{x}_{k|k}^{(m)} - \hat{\mathbf{x}}_{\mathbf{k}|\mathbf{k}})' \right) \tag{3.14}$$

The individual filtered states and updated probabilities are stored for use in the next iteration. A summary of the IMM algorithm is shown in Figure 3.2. For more explanation, see [3], [4], [76].

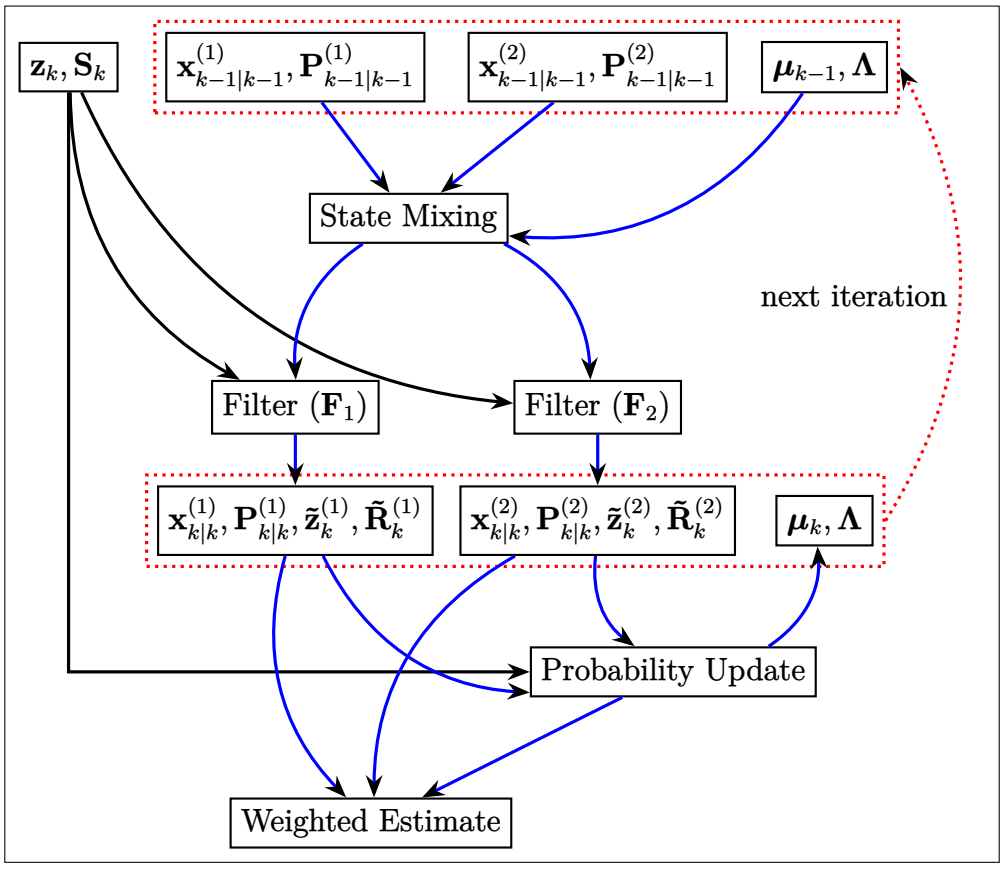


Figure 3.2: A flowchart illustrating one stage of the IMM algorithm. The quantities outlined in red can be maintained internally (if using an object-oriented scheme). The blue lines show the flow of those red items from top to bottom and the black arrows show where the measurement is injected into the process. The singular output by the algorithm is usually the weighted estimate at the bottom. This is a state space estimate of the target.

3.2.3 Neural Network Classifiers

Among the many applications of neural networks, one of the most ubiquitous is classification. Classification is the act of mapping from a state space $\mathcal{X}$ to a label space $\mathcal{Y}$. Neural networks achieve this task by first performing several intermediate linear algebra mappings and nonlinear activation functions before using the result of these transformations to compute the costs of assigning the transformed state to any one of a finite number of mutually exclusive categories contained in $\mathcal{Y}$.

The first consideration must be the definition of $\mathcal{X}$. While the kinematic state is already in vector form, this will not be used as the state space. First, use of the derivative quantities (i.e. velocity, acceleration, and turnrate) may be useful for improving the results depending on the scenario. However, the goal here will be to demonstrate a minimum level of improvement given only position measurements. Next, one shortcoming of the IMM is that it makes no direct use of the history of the track. There is technically a memory of the track history contained in the the probabilities vector $\boldsymbol{\mu}$, however, this information is fixed at each step. There is no ability for the IMM to somehow change the weight of information later. With a neural network, it is possible to process a chain of measurements on each iteration and allow that history to be more effectively represented in $\boldsymbol{\mu}$. The trajectory fed into the feed-forward neural network will be a column vector of concatenated positions so that $\mathcal{X} \coloneqq \mathbb{R}^{\alpha d_x}$ where α is the number of positions included in a trajectory. The neural network structure will have to be fixed upon deployment of a hypothetical sensor, so α must be fixed. If arbitrary length trajectories are needed (up to a finite number), then a neural network will have to be trained for every α, or recurrent neural networks (RNNs) such as long-short term memory (LSTM) networks will be required.

Establishing a fixed α for simpler feed-forward networks is relatively straightforward. It is possible to discern a minimal α such that the neural network generates useful output. If a target follows the same motion model for an extended period of time, then one only needs as many observations as would be necessary to fully estimate the model's parameters.

In a constant velocity model, that is 2 position measurements. In a constant acceleration model, 3 position measurements are necessary. For the 2-dimensional coordinated turn model, 3 position points are necessary (two for the turn rate computation itself and a third to establish whether the turn is clockwise or counterclockwise). So a minimal trajectory for the chosen M has $\alpha^\flat = 3$. Of course, the target's motion will be noisy due to random fluctuations in the true motion and the measurement process, so there will be a benefit to using trajectories longer than the minimum.

In addition to a minimal alpha, it will also be observed that a maximal α can be computed as well, though with less accuracy than the minimal α. The simplicity of the models in M gives some sense of certainty that there is a ceiling on the benefit of considering longer trajectories. Additionally, a trajectory must contain samples spread over some time interval which is governed by quantities such as the minimum sensor measurement interval and the scheduled revisit times. This time interval determines the minimum time needed to observe α positions to create a full trajectory to feed to the neural network. Relative to that time, there is a physical limit to how quickly a maneuvering target can change its position sufficiently to no longer appear to be following the same motion model as it was previously. If the time it takes to transition between motion models is less than the time it takes to acquire α position measurements, then the trajectory will contain measurements of multiple motion models. This is only a serious problem for state estimation if the time it takes to acquire a full trajectory is so long that none of the models in M can accurately predict the target's location. However, from the point of view of correctly identifying the target's motion, this is a considerable problem. A rule of thumb for choosing α is as follows:

$$\alpha^\sharp = \max_{\alpha \in \mathbb{Z}^+} \{\alpha : \alpha < \Delta t_{\text{switch}} / \Delta t_{\text{meas}}\} \tag{3.15}$$

where Δt_{meas} is the average time between measurements of the selected target and Δt_{switch} is the average time between discernable changes in motion models. It should be stressed that Δt_{switch} is not a known quantity. It is essentially a guess at how maneuverable the

intended targets are likely to be. Ideally $\Delta t_{\text{switch}} \gg \Delta t_{\text{meas}}$, in which case the target can be considered slowly maneuvering. If $\Delta t_{\text{switch}} < \Delta t_{\text{meas}}$, then it is impractical to suppose that the sensor is capable of discerning target motion quickly enough for the technique that follows. If $\alpha^\sharp < \alpha^\flat$, then the sensor likely does not provide enough measurements to form a trajectory which can resolve the models in M. For what follows, a trajectory of length 10 is used, though no claims of optimality are made regarding this choice.

3.3 Creation and Use of the Classifier

3.3.1 Data Generation and Representation

Training data was generated via functions in the Tracker Component Library [9]. To begin, idealized trajectories with ten 2-dimensional position measurements having no process noise were generated. These were subsequently perturbed with varying amounts of Cartesian measurement noise (independently sampled from a Gaussian distribution) to generate multiple noisy trajectories. Table 3.1 of intervals from which the initial state vector quantities were sampled. Note that three different time steps were sampled in order to allow some generalization over time as well.

Table 3.1: Data generation values for training the neural network for the toy example

Quantity	Value Set
Δt (s)	$\{0.5, 1, 2\}$
Pos. (m)	$[-1e2, 1e2]$
Vel. (m/s)	$[-1e3, 1e3]$
Acc. (m/s^2)	$[-30, 30]$
Turn Rate (rad/s)	$[-\pi, \pi]$
Meas. Cov. (m^2)	$[0, 10]$

3.3.2 Neural Network Structure and Training

The primary concern with constructing the neural network was to keep it small and fast. MATLAB has a function called `fitcnet` which will automatically train a neural network classifier and optimize a variety of parameters. The configuration which was settled on was a five layer model consisting of an input layer, a sequence of three hidden layers interleaved with rectified linear unit (ReLU) activation functions, and a final classification layer. The input layer standardized the input sequence by shifting to the mean and rescaling by the standard deviation. The layer sizes were 100, 10, and 30 respectively. No particular reason existed for these choices, and they don't correlate with any of the "optimized" layer sizes output by the `fitcnet` function's attempt to optimize the layer sizes. These choices produced reasonably generalizable results on the example. Training of the neural network was done using a 75/25 train/test split of the trajectory data (225,000 trajectories for training, 75,000 for validation). Randomization of the trajectories was done prior to each training epoch.

3.3.3 Interfacing with Multiple Model Algorithms

The neural network classifier is used to generate prior model probabilities $\boldsymbol{\mu}_{k-1}$ (see Figure 3.2 for context). On their own, the probability outputs of the neural network tend to be sporadic, roughly with the frequencies predicted by the confusion matrix. To patch this problem, a weighting was implemented as a sort of consistency restraint. The output probabilities from the last iteration of the IMM are multiplied by a weight $0 \leq w \leq 1$ and then added to the new prediction of the neural network weighted by $1 - w$. It was found that choosing w roughly equal to its accuracy was a good rule-of-thumb. Even for a highly accurate network, however, there should be a fairly strong demand for consistency as real targets will not follow any particular motion model exactly. For this reason, an upper bound on w should be observed regardless of the neural network's accuracy. The simulated examples will use $w = 0.9$.

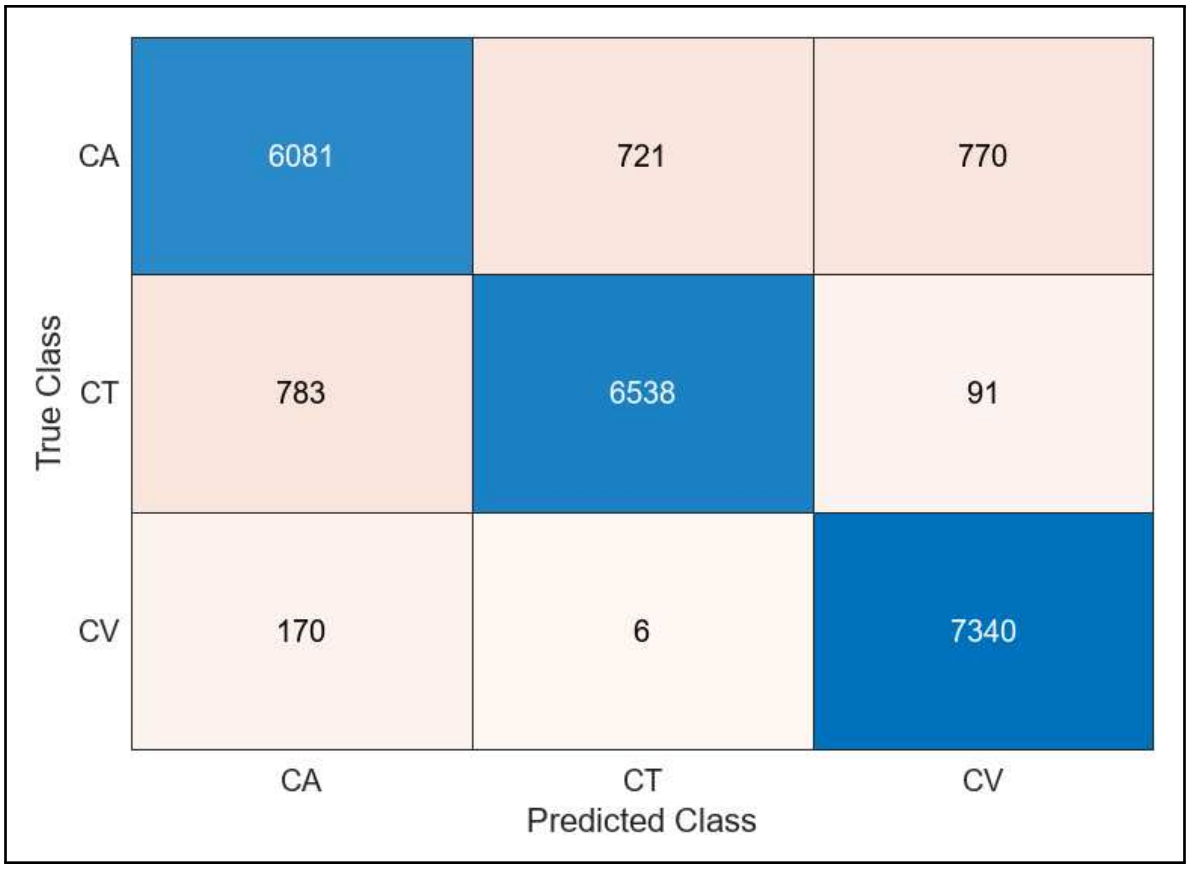


Figure 3.3: Confusion matrix for the neural network classifier

As noted above, the utility of neural networks for increasing accuracy of IMM models has been noted in the literature before. The novel contribution of this chapter is the additional use of the neural network's confusion matrix as a replacement for the designed transition matrix of the traditional multiple model algorithms. The confusion matrix is known once the neural network has been trained, so it is far more desirable for use in a practical algorithm when the target maneuver sequences are not known in advance. The confusion matrix must be made into a transition matrix by normalizing the rows. This normalization only has to be done once to get $\mathbf{\Lambda}$.

3.3.4 Extension to Distributed Systems

There are cases where it is either impractical or impossible to share full target states between trackers to perform fusion. This does not have to mean there is no possibility for

information fusion, however. In the case of a NNMM estimator, the weighted averaging step allows for incorporation of other trackers' motion model probabilities without need for full target state information. Assuming one has a scheme for assigning weights $\{w_i\}_{i=0}^{N}$ for N external trackers, $i = 0$ corresponding to the tracker on the ownship, and $\sum w_i = 1$, a weighted average of the model probabilities can be used as input to the multiple model estimator. This implementation would likely also necessitate a change to $\mathbf{\Lambda}$. The simplest modification would involve weighted addition of the $\mathbf{\Lambda}$ matrices from each tracker and then normalizing the rows so that the new matrix is a probability matrix.

3.4 Examples

Consider now a model set of $M \coloneqq \{\text{CV, CA, CT}\}$. The transition matrix for the traditional IMM algorithm will be

$$\mathbf{\Lambda} = \begin{bmatrix} 0.9 & 0.08 & 0.02 \\ 0.15 & 0.70 & 0.15 \\ 0.04 & 0.16 & 0.80 \end{bmatrix} \tag{3.16}$$

copying the values used in [76] and depicted in Figure 3.1. The models are as described above and the full target state is $\mathbf{x} = [x_p, y_p, x_v, y_v, x_a, y_a, \tau]'$. The simulated trajectory begins with a constant velocity model of $[1, -2]'$ meters per second followed for 30 s, switches to a coordinated turn with turn rate of -10 degrees per second (with the same speed), and then concludes beginning at 60 s with a constant acceleration model using an acceleration of $[-0.01, 0.03]'$ meters per second squared until the stop time at 150 s. The Cartesian measurement covariance is $\text{diag}(5, 5)$. This trajectory can be seen in Figure 3.4 and the measurements are shown in Figure 3.5 along with the track estimates.

The corresponding model probability vector $\boldsymbol{\mu}_k$ is plotted over time in Figure 3.6. As the neural network requires 10 prior estimates, the traditional IMM was used for the first

10 steps and the first prior probabilities of the classifier at step 11 were mixed with the posterior output by the traditional IMM on step 10. It is readily apparent that the NNMM track is slightly smoother than the IMM track. This is to be expected since the NNMM filter is performing some smoothing by incorporating a sliding window approach to the estimation. Computing the jitter metric (first version presented in [91]) on the two tracks (position coordinates only) yields 35.4462 meters for the IMM track and 18.7436 meters for the NNMM track. So in this sense, the NNMM track has half as much jitter. Smoothing aside, some modest improvements in position accuracy can also be seen in Figure 3.7. As a check, Figure 3.9 shows that this improved accuracy leaves the consistency of the estimated Gaussian distribution relatively intact over the course of one simulated trajectory, as both IMM and NNMM have comparable ranges of NEES scores and hover around 1. A collection of 100 Monte Carlo runs were used to verify the apparent accuracy gain and consistency. The Monte Carlo runs used the same trajectory as shown in Figure 3.4, but resampled the measurements. As can be observed in Figure 3.8, the accuracy gain persists over all 100 runs. The ANEES metric, shown in Figure 3.10, seems to exhibit more variability with the NNMM filter, but also has a mean closer to the desired value of 1. Taken together, these metrics seem to suggest that the NNMM represents a sound augmentation of IMM that is no worse than IMM provided the neural network is trained on a representative training set.

3.5 Conclusion

The well-known IMM algorithm was shown to yield improved accuracy when its inputs were first augmented by a neural network. The emphasis was not on a particular structure of the network. Different structures will be suited to different sensor types and maneuver models. Instead, emphasis was placed on the ability of the neural network to enable utilization of a longer track history and replacement of the assumptions required for constructing a transition probability matrix describing the model set's Markov chain.

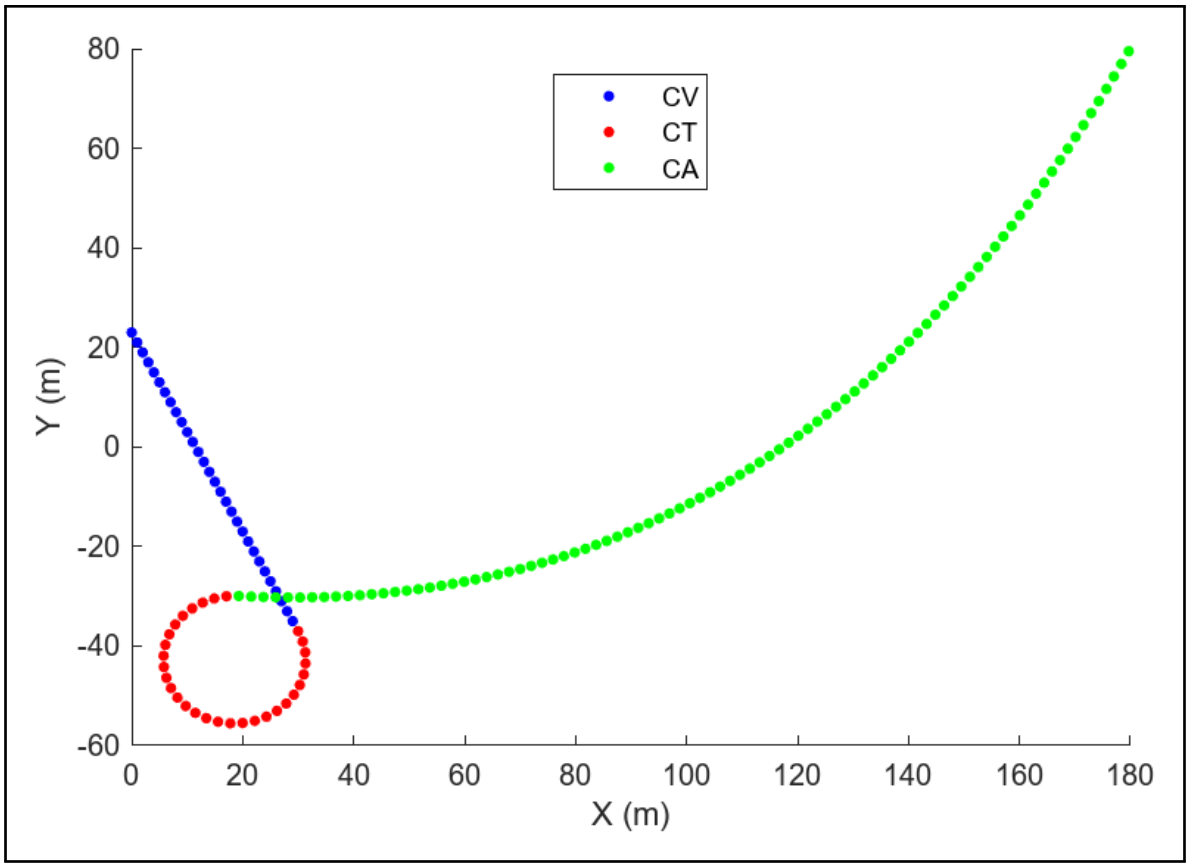
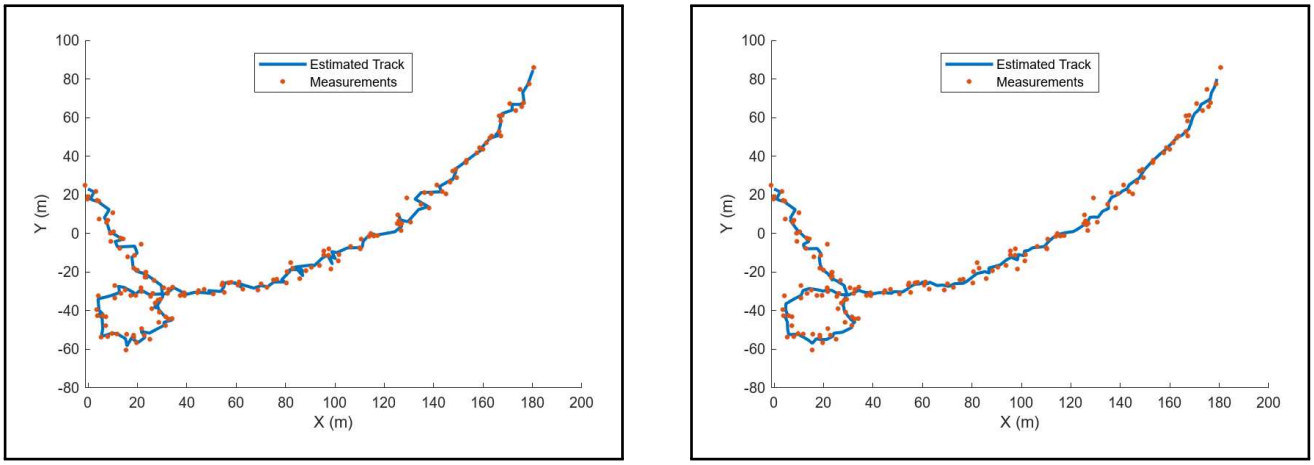
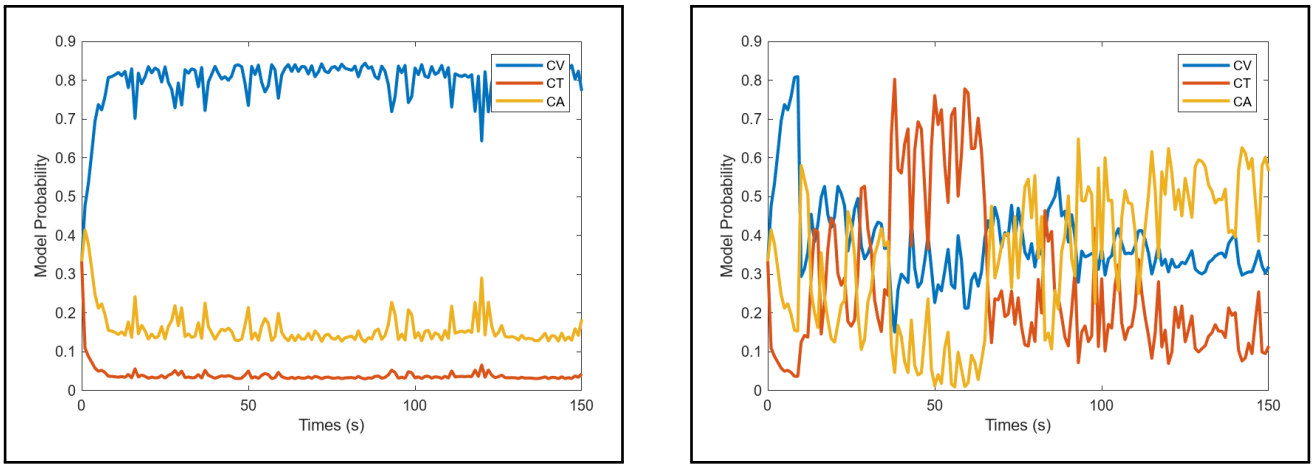


Figure 3.4: The simulated true trajectory



(a) **(b)**

Figure 3.5: The tracks estimated by the IMM filter (a) and NNMM filter (b).



(a) **(b)**

Figure 3.6: The motion model probability vector plotted over time for the IMM filter (a) and the NNMM filter (b).

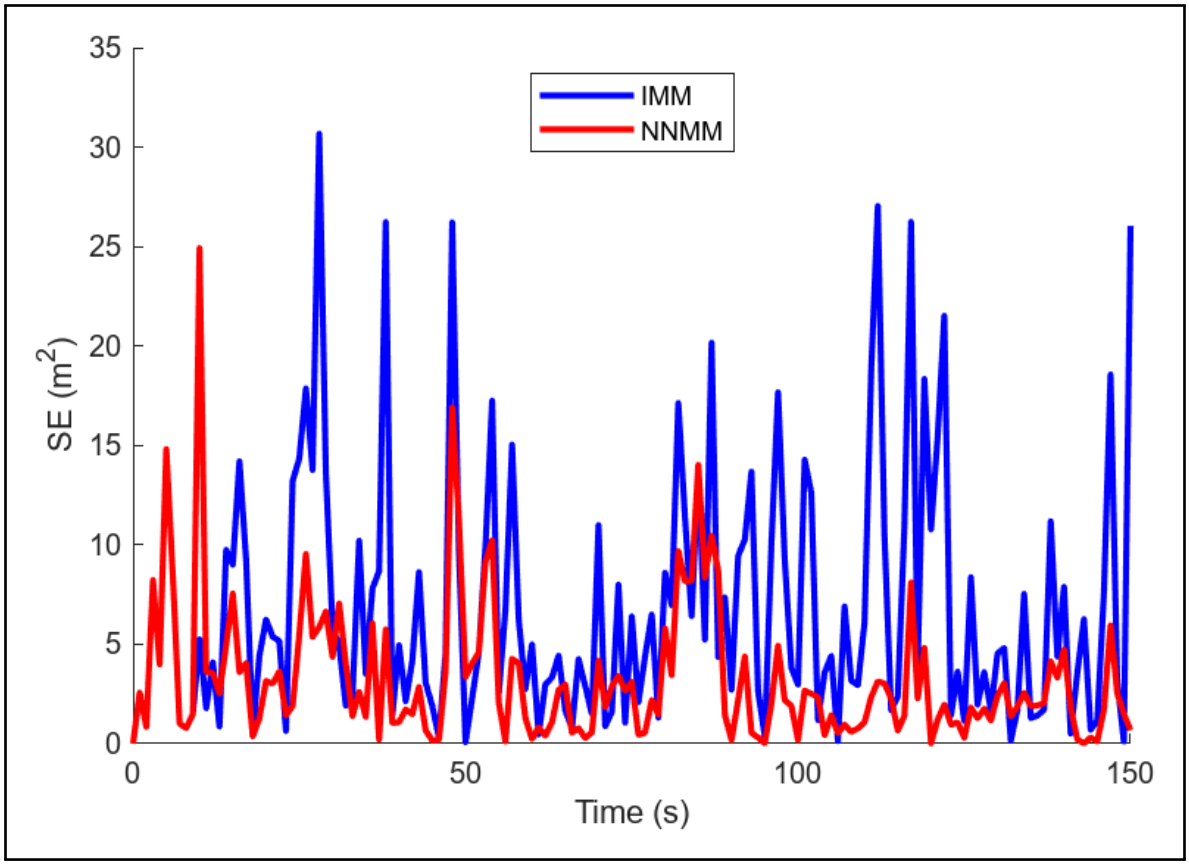


Figure 3.7: A comparison of the squared-error (SE) metric for the IMM track (blue) and NNMM track (red).

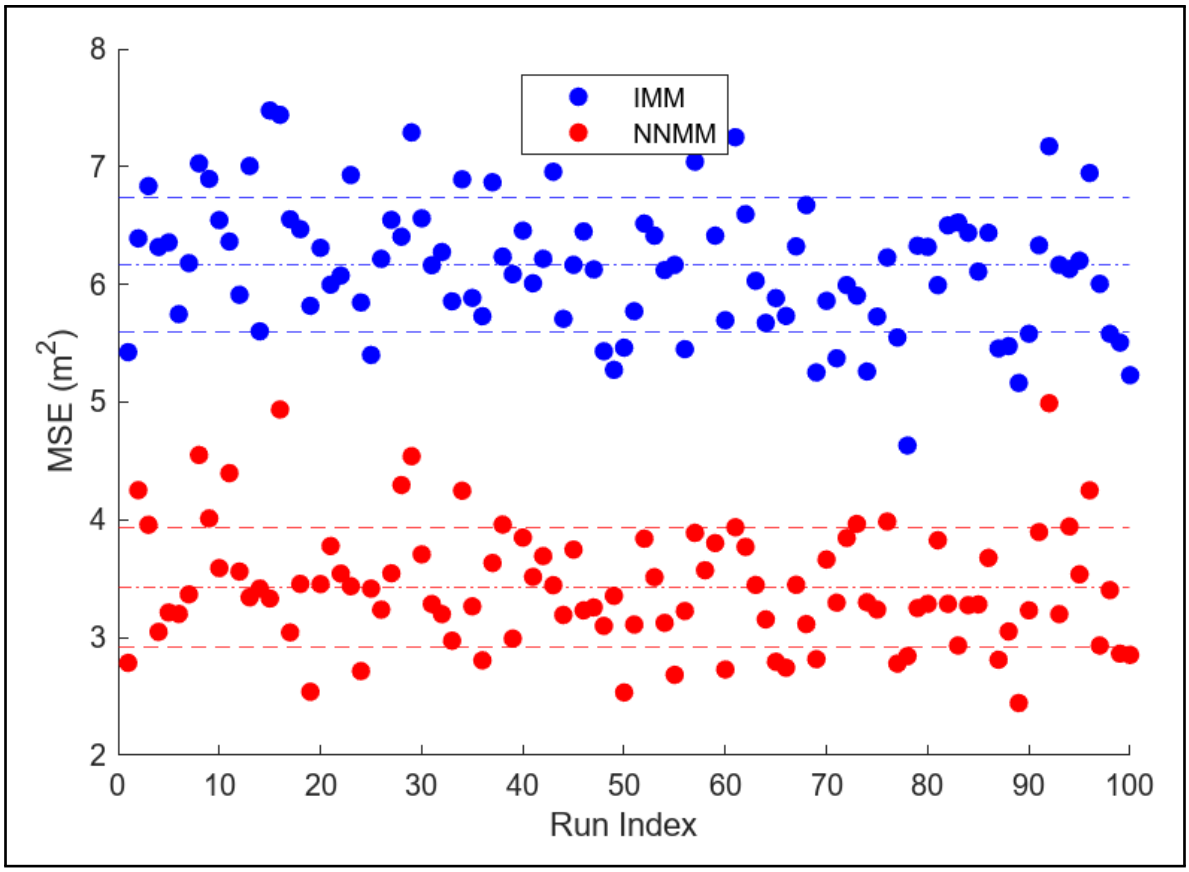


Figure 3.8: A comparison of the mean-squared-error (MSE) metric for the IMM track (blue) and NNMM track (red). Mean and standard deviation statistics are shown via the dot-dash and dashed lines, respectively.

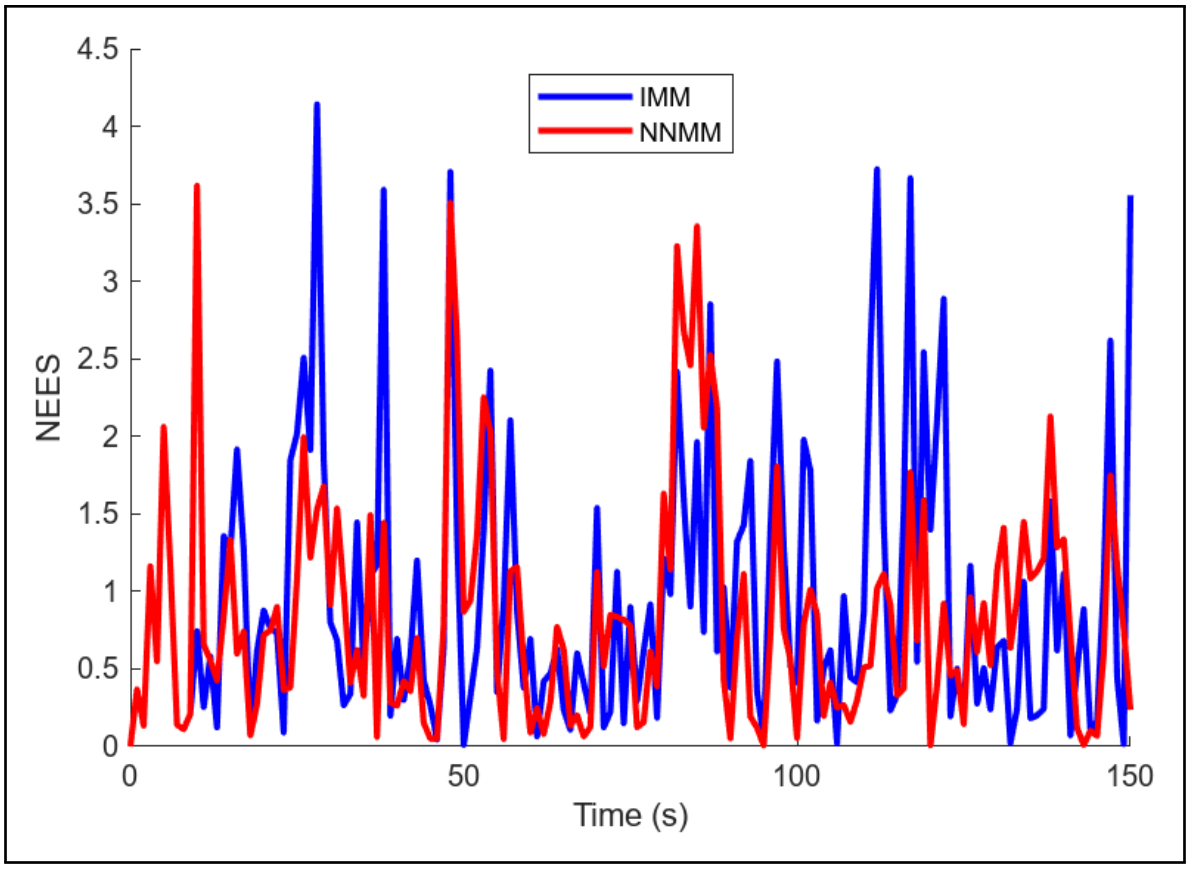
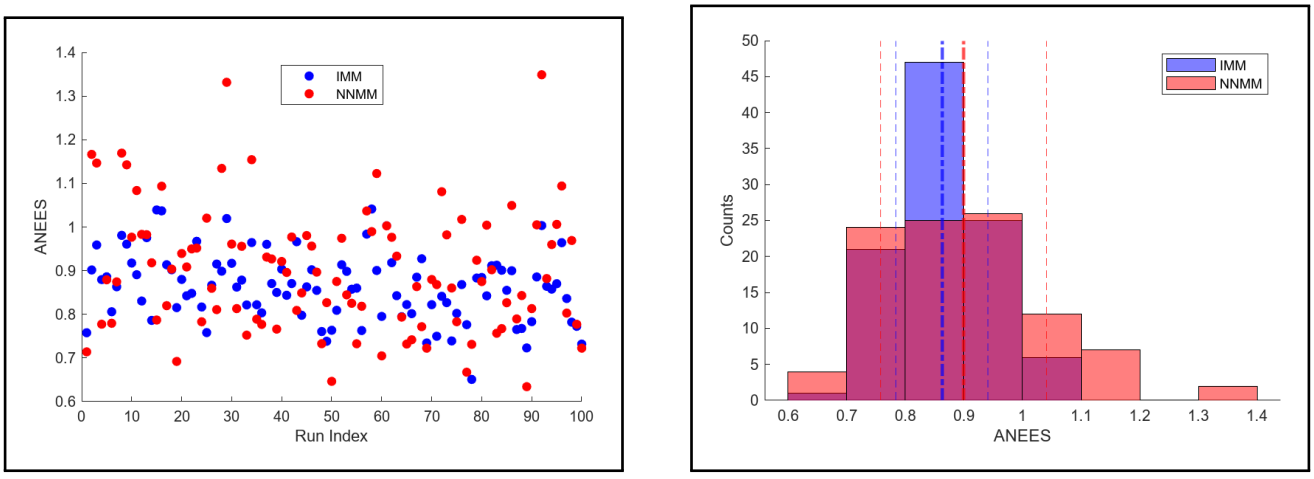


Figure 3.9: A comparison of the normalized-estimation-error-squared (NEES) metric for the IMM track (blue) and NNMM track (red).



(a) **(b)**

Figure 3.10: Comparisons of the average-normalized-estimation-error-squared (ANEES) metric for the IMM (blue) and NNMM (red) Monte Carlo runs. The scatter plots are shown in (a) and histograms of the same data are shown in (b). Mean (dot-dash) and standard deviation (dashed) statistics are plotted on the histograms with the means bolded for clarity.

Chapter 4

Fibonacci Sampling for Fusion

4.1 Introduction

When tracks are determined to represent the same target (potentially through the method described in Chapter 2), a tracker must combine their information with a track-to-track fusion algorithm. Often, the histories of tracks and the common assumptions on target motion will lead to correlations in track errors. The track-to-track fusion algorithm discussed below, Chernoff fusion, is capable of performing fusion of correlated inputs. There are two primary benefits to Chernoff fusion.

First, Chernoff fusion enables fusion of a general class of estimate distributions without foreknowledge of the correlation. The cost for this generality is the need to compute a product of two weighted densities and then normalize the result. For general distributions, this can only be done approximately with weighted samples from the support of each distribution. Even when general distributions are fused, it is useful to simplify their representation if possible by computing moment-based quantities such as mean and variance. Recall that these are integral calculations, so accurate sampling and integral approximation becomes essential for Chernoff fusion to remain accurate.

Second, Chernoff fusion is a generalization of a common fusion technique called covariance intersection. In the case where both distributions being fused are Gaussian, Chernoff fusion yields the same output as covariance intersection. This equivalence requires specific choices which will be discussed below. Still, this equivalence can provide a closed-form, true distribution for comparison against the Chernoff fusion results. This will allow for testing and comparison of several sampling techniques.

In what follows, an exposition of Chernoff fusion and covariance intersection will be given. Then, the case for using Fibonacci points in sampling algorithms will be made and the

different algorithms for generating Fibonacci points will be discussed. Finally, a comparison of performance for Chernoff fusion and covariance intersection under different Fibonacci point sampling methods will be evaluated. This evaluation has not been done in the literature and will help to inform best practices when using Chernoff fusion.

4.2 Covariance Intersection and Chernoff Fusion

The fusion of distributions requires considering the joint probability of both associated tracks. Let $(\mathbf{x}_1, \mathbf{P}_1)$ and $(\mathbf{x}_2, \mathbf{P}_2)$ be the means and covariances of the two associated tracks in $\mathbb{R}^{\eth}$ where $\eth$ is the dimension of the subspace of the track's state space which is being fused (below $\eth = 2$ will be assumed which could be viewed as only fusing the position coordinates of a planar motion target model). If it is known that the track estimates are uncorrelated, then the mean and covariance are fused as follows [3]:

$$\mathbf{P}_f = (\mathbf{P}_1^{-1} + \mathbf{P}_2^{-1})^{-1} \tag{4.1}$$

$$\mathbf{x}_f = \mathbf{P}_f(\mathbf{P}_1^{-1}\mathbf{x}_1 + \mathbf{P}_2^{-1}\mathbf{x}_2). \tag{4.2}$$

If, however, it cannot be assumed that the tracks are uncorrelated, then using this approach will lead to "overconfident" fusion results. Here, "overconfident" tracks have smaller covariances than justified by all the measurements, had they been filtered as a single track.

To fuse tracks which are possibly correlated, there exist several alternatives [3], [92]–[98]. Here, we will consider covariance intersection as it is the Gaussian special case of Chernoff fusion and the algorithm from which Chernoff fusion was derived [97]. The rule for covariance intersection uses a convex combination of the two input track probabilities:

$$\mathbf{P}_f = (w_1\mathbf{P}_1^{-1} + (1 - w_1)\mathbf{P}_2^{-1})^{-1} \tag{4.3}$$

$$\mathbf{x}_f = \mathbf{P}_f(w_1\mathbf{P}_1^{-1}\mathbf{x}_1 + (1 - w_2)\mathbf{P}_2^{-1}\mathbf{x}_2). \tag{4.4}$$

It is nontrivial to choose $w_1 \in [0,1]$. One approach, offered in [93], proposes that the weight should be chosen as a minimizer for a function of the fused covariance:

$$w_1^* \coloneqq \arg\min_{w_1 \in [0,1]} \{J(\mathbf{P}_f)\} \tag{4.5}$$

where J is an arbitrary cost function. While [93] offers both the determinant and trace as options for J, [97] notes that only the determinant is linked to Shannon entropy via the equality:

$$\underbrace{-\int_{\mathcal{X}} p_f(\mathbf{x}) \ln(p_f(\mathbf{x})) d\mathbf{x}}_{\text{Shannon Entropy}} = \frac{1}{2} \ln\left((2\pi)^{\eth} \det(\mathbf{P}_f)\right) + \frac{\eth}{2}. \tag{4.6}$$

Since the left-hand side is a monotonic function of $\det(\mathbf{P}_f)$, both the Shannon entropy formulation of Chernoff fusion (to be shown below) and the choice of $J = \det$ for covariance intersection will yield the same minimizer w_1^*. Therefore, let J be the determinant function det.

The derivation given in [93] uses a joint diagonalization to construct a polynomial with one of the roots being the optimal choice for w_1. Here, a truncated exposition of the derivation from [93] for dimension $\eth = 2$ is given. Define the following variables related to given covariances $\mathbf{P}_1$ and $\mathbf{P}_2$:

$$\mathbf{P}_1 = \mathbf{V}_1 \mathbf{\Lambda}_1 \mathbf{V}_1' \tag{4.7}$$

$$\mathbf{T}_1 \coloneqq \left(\mathbf{V}\sqrt{\mathbf{\Lambda}_1}\right)^{-1} \tag{4.8}$$

$$\tilde{\mathbf{P}}_2 \coloneqq \mathbf{T}_1 \mathbf{P}_2 \mathbf{T}_1' = \tilde{\mathbf{V}}_2 \tilde{\mathbf{\Lambda}}_2 \tilde{\mathbf{V}}_2' \tag{4.9}$$

$$\mathbf{T} \coloneqq \tilde{\mathbf{V}}_2 \left(\sqrt{\mathbf{\Lambda}_1}\right)^{-1} \tilde{\mathbf{V}}_2' \tag{4.10}$$

$$\mathbf{D} = \mathbf{T}\mathbf{P}_2\mathbf{T}' \tag{4.11}$$

where the equivalences in Equations (4.7) and (4.9) are given by eigenvalue decompositions. A consequence of these definitions which will be useful later is:

$$\mathbf{T}\mathbf{P}_f\mathbf{T}' = \left(w_1\mathbf{I} + (1-w_1)\mathbf{D}^{-1}\right)^{-1}. \tag{4.12}$$

Furthermore, let $\{t_1, t_2\}$ and $\{d_1, d_2\}$ denote the diagonal elements for $\mathbf{T}$ and $\mathbf{D}$ respectively. Finally, define the following derived quantity: $\bar{d}_i \coloneqq 1/d_i$ for $i = 1, 2$. The following computation then yields a polynomial with one root in [0,1]:

$$\begin{aligned}
w_1^* &\coloneqq \arg\min_{w_1 \in [0,1]} \{J(\mathbf{P}_f)\} && (4.13)\\
&= \arg\min_{w_1 \in [0,1]} \left\{\det\left(\mathbf{T}^{-1}\right)\det(\mathbf{T}\mathbf{P}_f\mathbf{T}')\det\left((\mathbf{T}')^{-1}\right)\right\} && (4.14)\\
&= \arg\min_{w_1 \in [0,1]} \{\det(\mathbf{T}\mathbf{P}_f\mathbf{T}')\} && (4.15)\\
&= \arg\min_{w_1 \in [0,1]} \left\{\frac{1}{(w_1 + (1-w_1)\bar{d}_1)(w_1 + (1-w_1)\bar{d}_2)}\right\} && (4.16)\\
&= \arg\max_{w_1 \in [0,1]} \left\{(w_1 + (1-w_1)\bar{d}_1)(w_1 + (1-w_1)\bar{d}_2)\right\}. && (4.17)
\end{aligned}$$

The move from Equation 4.14 to Equation 4.15 is accomplished by observing that $\det\left(\mathbf{T}^{-1}\right)\det\left((\mathbf{T}')^{-1}\right) > 0$. Solving for this root yields the desired weight:

$$w_1^* = -\frac{1}{2}\left(\frac{1}{1-d_1} + \frac{1}{1-d_2}\right). \tag{4.18}$$

This is the weighting choice which will be used in the covariance intersection algorithm in following sections.

The Chernoff fusion rule is as follows [97], [99]:

$$p_f(\mathbf{x}) = \frac{p_1^{w_1}(\mathbf{x})p_2^{1-w_1}(\mathbf{x})}{\int_{\mathcal{X}} p_1^{w_1}(\mathbf{x})p_2^{1-w_1}(\mathbf{x})d\mathbf{x}}. \tag{4.19}$$

Note that in the special case of p_1 and p_2 as Gaussian distributions, we have:

$$p_f(\mathbf{x}) \propto \exp\left\{-\frac{w_1}{2}(\mathbf{x}-\mathbf{x}_1)'\mathbf{P}_1^{-1}(\mathbf{x}-\mathbf{x}_1)\right\}\exp\left\{-\frac{(1-w_1)}{2}(\mathbf{x}-\mathbf{x}_2)'\mathbf{P}_2^{-1}(\mathbf{x}-\mathbf{x}_2)\right\} \tag{4.20}$$

$$= \exp\left\{-\frac{w_1}{2}(\mathbf{x}-\mathbf{x}_1)'\mathbf{P}_1^{-1}(\mathbf{x}-\mathbf{x}_1) - \frac{(1-w_1)}{2}(\mathbf{x}-\mathbf{x}_2)'\mathbf{P}_2^{-1}(\mathbf{x}-\mathbf{x}_2)\right\} \tag{4.21}$$

$$= \exp\left\{\frac{1}{2}(\mathbf{x}-\mathbf{x}_f)'\mathbf{P}_f(\mathbf{x}-\mathbf{x}_f)\right\}\exp\left\{w_1\mathbf{x}_1'\mathbf{P}_1\mathbf{x}_1 + (1-w_1)\mathbf{x}_2'\mathbf{P}_2\mathbf{x}_2 - \mathbf{x}_f'\mathbf{P}_f\mathbf{x}_f\right\} \tag{4.22}$$

$$\propto \exp\left\{\frac{1}{2}(\mathbf{x}-\mathbf{x}_f)'\mathbf{P}_f(\mathbf{x}-\mathbf{x}_f)\right\}. \tag{4.23}$$

From this it can be seen that p_f is a Gaussian [97]. However, similar to the problem for covariance intersection, it is not obvious what rule to use in optimizing w_1. If it can be shown that the choice of w_1 for Chernoff fusion agrees with that computed as w_1^* above, then it will be known that the output distribution of Chernoff fusion agrees with covariance intersection in the Gaussian case. Therefore, referring back to the relationship noted in Equation (4.6), it has been established that the determinant-based covariance intersection algorithm and Shannon-entropy-based Chernoff fusion algorithm yield equivalent output distributions given the same Gaussian input distributions [97].

Finally, the Chernoff fusion weight calculation requires optimization. To pare down the complexity, Farrell in [99] used some Gaussian entropy substitutions to arrive at a closed expression for each weight:

$$w_1 = C(1 - \exp(H_{1,2} - H_2) + \exp(H_{1,2} - H_1)) \tag{4.24}$$

$$w_2 = C(1 - \exp(H_{1,2} - H_1) + \exp(H_{1,2} - H_2)) \tag{4.25}$$

where C is a normalizing constant, $H_{1,2}$ is the Shannon entropy of the Bayesian fusion of the two distributions, and H_1 and H_2 are the entropies of the first and second input distributions respectively. A further simplification made in [99] is the use of a lower limit approximation for $H_{1,2}$:

$$H_{1,2} \approx \frac{\min\{H_1, H_2\}}{2}. \tag{4.26}$$

These simplifications will be used in the Chernoff fusion computations below.

4.3 Quasi-Monte Carlo and Low-Discrepancy Lattices

Computing the moments of a distribution or updating the distribution via Equations (1.16), (1.22), (1.28), and (1.32) requires integration. Without assuming a Gaussian form, these integrals generally cannot be written as closed-form expressions. It is therefore necessary to approximate them via a summation over a discrete set of sample points:

$$\int_{\mathcal{S}} f(\mathbf{x})d\mathbf{x} \approx \sum_{i=1}^{N} w_i f(\mathbf{x}_i) \quad \text{for } \{\mathbf{x}_i\}_{i=1}^{N} \subset \mathcal{S} \text{ and } \sum_{i=1}^{N} w_i = 1. \tag{4.27}$$

There are many terms used across the literature for the concept in Equation (4.27). In one dimension, this is called quadrature, though that term is sometimes used generally for all dimensionalities. For dimensionality greater than one, math literature generally refers to Equation (4.27) as cubature integration whereas tracking literature occasionally calls the same concept sigma-point integration when referring to integration as a part of filtering algorithms [100], [101]. In any dimensionality, the most generic term is numerical integration. Additionally, while $\mathcal{S}$ can in principle be any measurable space, most applications relevant to target tracking will permit a mapping from the unit volume in $\mathbb{R}^{\eth}$ to the volume of interest. For this reason, assume $\mathcal{S} = [\mathbf{0}, \mathbf{1}) \subset \mathbb{R}^{\eth}$ where $\mathbf{0}$ and $\mathbf{1}$ are vectors of length $\eth$ containing zeros and ones respectively. In general, the notation $[\mathbf{a}, \mathbf{b}) \subset \mathbb{R}^{\eth}$ will denote a volume bounded by vectors $\mathbf{a}$ and $\mathbf{b}$ such that $[\mathbf{a}, \mathbf{b}) = [a_1, b_1) \times [a_2, b_2) \times \cdots \times [a_{\eth}, b_{\eth})$.

Two approaches are possible for generating a sample set from $\mathcal{S}$: random or deterministic sampling. Random sampling leads to various well-studied techniques under the umbrella of Monte Carlo integration. Sampling uniformly random elements of $\mathcal{S}$ would exhibit an error reduction rate of $\mathcal{O}(1/\sqrt{N})$, which can be seen by applying the central limit theorem [102]. Here, Monte Carlo is also being used to encompass pseudo-random sampling via pseudo-random number generators. While technically deterministic, pseudo-random numbers are

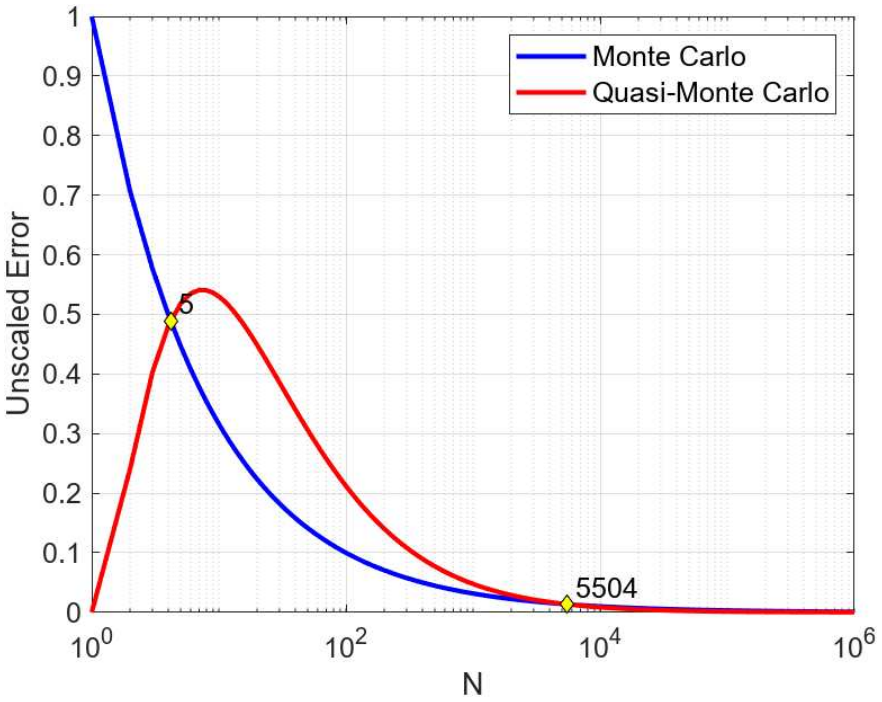


Figure 4.1: A comparison of the rates of convergence for Monte Carlo and QMC integration is shown. The yellow diamonds indicate the ceiling of the number of points at the intersections of the two curves.

designed to pass statistical tests for uniformity. Integration with deterministic points is called quasi-Monte Carlo (QMC) integration, although there is no random aspect to these algorithms [102]–[104]. QMC integration allows for offers an opportunity for faster convergence to the solution under the right circumstances [105], [106]. For dimension $\eth$ and number of points N, the rate of convergence of QMC is usually taken to be at worst $\mathcal{O}((\log N)^{\eth}/N)$ (the rate of convergence for QMC with Halton points) [107]. In the cases to be considered below for $\eth = 2$, this is $\mathcal{O}((\log N)^2/N)$. The unscaled errors are compared in Figure 4.1, where unscaled means the plots only account for the discrepancy of the sample points, not the variation of the integrand. The indication of the intersections in Figure 4.1 is meant to show that for some interval of small sample points, Monte Carlo samples may do better than QMC as the rate of convergence is an asymptotic result. If the number of dimensions being integrated grows considerably, use of Monte Carlo should be considered as that rate of convergence does not grow with $\eth$.

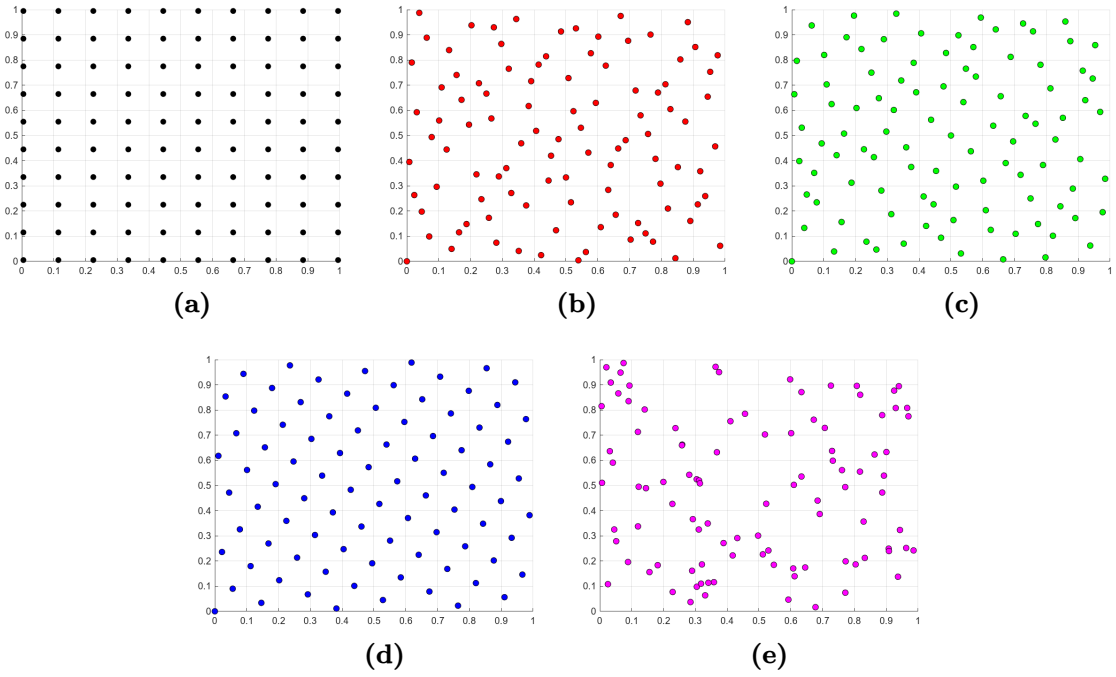


Figure 4.2: Examples of various samples of the unit square using 100 points according to the regular grid (a), Halton (b), Sobol (c), Fibonacci (d), and uniform random (e) point set generation algorithms.

There are a number of well-known deterministic point generation techniques. These include regular and rectilinear grids, as well as Halton [108]–[111], Sobolev [111]–[113], and Fibonacci point sets [114]–[117]. The important consideration for these point sets is geometric uniformity, not necessarily statistical uniformity. Examples of these point sets in two dimensions are shown in Figure 4.2 and compared with a random sample of uniformly distributed points. It will be immediately obvious that if sampling efficiency is important, deterministic points will be more desirable than random points. In the context of approximating Chernoff fusion, it is also beneficial to have nearly uniform (in a geometric sense) points so that any bias introduced into the state estimate can be known before the points themselves are calculated.

The default method of selecting geometrically uniform points is to take a rectilinear grid, which divides $\mathcal{S}$ into a collection of parallelotope. One problem faced by this choice is the potential for inefficient sampling of the transformed space $f(\mathcal{S})$. This is called sample degeneracy or sample collapse. A degenerate sample occurs when many points which are

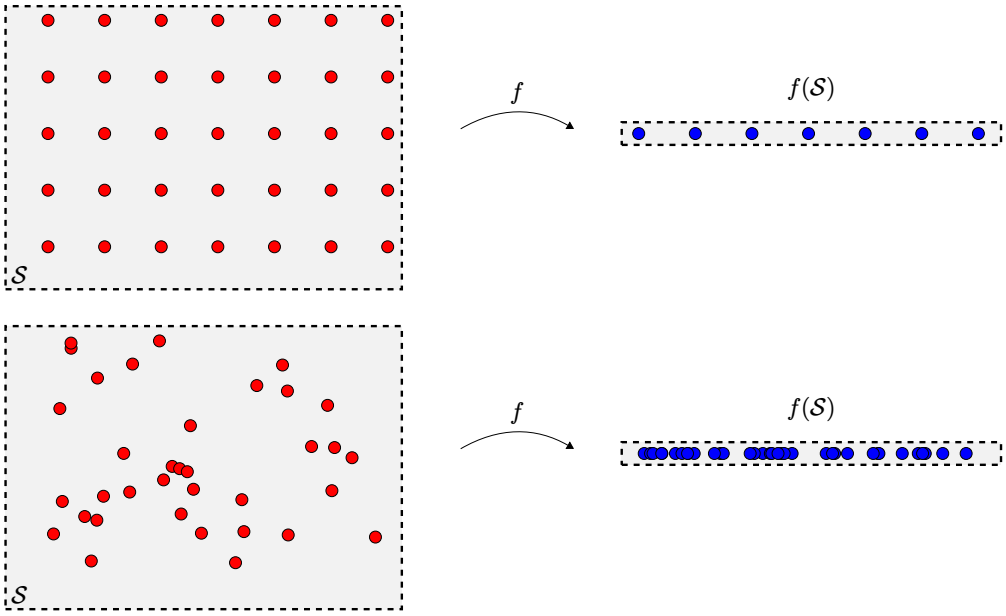


Figure 4.3: This demonstrates sample degeneracy, where the sample set collapses under transformation by chosen function f. The top image demonstrates a worst case collapse on a regular grid, where there are several identical samples of $f(S)$, making the sampling approach inefficient. The bottom figure demonstrates an improvement in sampling efficiency using randomly sampled points from a uniform distribution on $\mathcal{S}$.

expected to be distinct are mapped to identical or nearly identical points. An example of sample collapse is depicted in Figure 4.3. While the expected sample accuracy would be a function of the number of red points (35), the true accuracy would be governed by the number of blue points (7). This is especially important when integrating separable functions, or functions which are expected to be aligned with the coordinate axes. In track-to-track fusion, it is common to align eigenvectors of the covariance from one track to the coordinate axes. Therefore, while it will be shown below that a regular grid does perform well for the QMC integration, the point sets which are not aligned with the coordinate axes will be preferred. One option for reducing the likelihood of degenerate samples is Monte Carlo sets. Randomly sampled points are unlikely to possess simple intersample structure, the cause of sample degeneracy. However, as illustrated in Figure 4.3, random or pseudo-random samples tend to be inefficient at uniformly sampling due to random clustering behavior, which is undesirable if the goal is to sample $\mathcal{S}$ using as few points as possible [103], [104].

Now we briefly consider in what sense a set of points can be considered "good" for QMC. To compare point sets as better or worse for QMC integration, consider the following restatement of Equation (4.27):

$$\varepsilon_N = \left| \int_{\mathcal{S}} f(\mathbf{x}) d\mathbf{x} - \sum_{i=1}^{N} w_i f(\mathbf{x}_i) \right| \quad \text{for } \{\mathbf{x}_i\}_{i=1}^{N} \subset \mathcal{S} \text{ and } \sum_{i=1}^{N} w_i = 1. \tag{4.28}$$

where ε_N denotes the error of the approximation. Clearly, the goal is to minimize ε_N. So for the sample set $\{\mathbf{x}_i\}_{i=1}^{N} \subset \mathcal{S}$ and bounded variation function f define the discrepancy as:

$$D_N \coloneqq \sup_{\mathcal{A}} \left\{ \left| \frac{|\{\mathbf{x}_i\} \cap \mathcal{A}|}{N} - |\mathcal{A}| \right| : \mathcal{A} \subseteq \mathcal{S} \text{ is Lebesgue-measurable} \right\} \tag{4.29}$$

where $|\mathcal{A}|$ is taken as the Lebesgue measure for sets of infinite cardinality and the counting measure for finite sets. The discrepancy is a quantity for comparing the suitability of points for QMC integration. The product of the discrepancy and Hardy-Krause variation was proven to provide a bound on the error of QMC integrals for functions of bounded variation (for the definition of Hardy-Krause variation, see [118], [119]) via the Koksma-Hlawaka Theorem [102], [119]–[121]. This is of course difficult to compute in practice and $\mathcal{S}$ is usually assumed to be a unit volume $[\mathbf{0}, \mathbf{1}) \subset \mathbb{R}^{\eth}$, so a closely related quantity called *-discrepancy was defined by restricting $\mathcal{A}$ to the half-open intervals $[\mathbf{0}, \mathbf{s}) \subset \mathbb{R}^{\eth}$:

$$D_N^* \coloneqq \sup_{[\mathbf{0},\mathbf{s}) \subset \mathcal{S}} \left\{ \left| \frac{|\{\mathbf{x}_i\} \cap [\mathbf{0}, \mathbf{s})|}{N} - \prod_{k=1}^{\eth} s_k \right| \right\}. \tag{4.30}$$

With respect to these and other discrepancy definitions (varying by the restrictions on the underlying space or restrictions to the integrand), the traditional Fibonacci points perform either optimally or near-optimally under myriad transformations (from the unit volume to other manifolds) with minimal augmentation of the points [114], [120], [122]–[128]. For this reason, although the regular grid, Halton, and Sobolev points will be considered below, the focus will be on Fibonacci points. The Halton and Sobolev points will be generated

using MATLAB's Halton and Sobolev generation functions with all default settings in two dimensions.

As a brief aside, discrepancy is not to be confused with dispersion. The dispersion of a sequence is defined as:

$$\delta_N \coloneqq \sup_{\mathbf{x}\in\mathcal{S}} \left\{ \min_{i=1,\ldots,N} \{d(\mathbf{x},\mathbf{x}_i)\} \right\}. \tag{4.31}$$

This can be seen as a quantification of how poorly represented the least represented member of $\mathcal{S}$ will be if similarity is encoded by a metric d [129]. The traditional Fibonacci points are noted as a good choice for minimizing both discrepancy and dispersion [122], [130]–[132].

4.4 Fibonacci Lattices

The previous section referenced proven performance for the traditional Fibonacci points. The reason for the modifier "traditional" is a necessary distinction between the points traditionally studied in mathematical literature and modified versions which are also called Fibonacci points. Three variants of the Fibonacci points will be considered below and will be named as follows: Niederreiter-Sloan, Kronecker-Jacob, and Swinbank-Purser. Note these names are not commonly used (instead they are called Fibonacci points, Kronecker-Fibonacci points, and generalized Fibonacci points respectively), but the distinctive names with easily distinguished acronyms will be helpful. The central theme for all of these points is the use of Fibonacci numbers or related quantities, hence the attribution to Fibonacci. The Fibonacci numbers are defined as follows:

$$\mathfrak{F} \coloneqq \{\mathfrak{F}_n \coloneqq \mathfrak{F}_{n-1} + \mathfrak{F}_{n-2} : \mathfrak{F}_0 \coloneqq 0, \mathfrak{F}_1 \coloneqq 1\} = \{0,1,1,2,3,5,8,13,21,...\} \tag{4.32}$$

where sometimes the zeroth term is ignored and $\mathfrak{F}_2$ is just defined as 1 [133]. There are often-cited connections between the Fibonacci numbers and the golden ratio $\varphi \coloneqq (1+\sqrt{5})/2$, one

of which is the limit of ratios of consecutive Fibonacci numbers:

$$\lim_{n\to\infty} \frac{\mathfrak{F}_n}{\mathfrak{F}_{n-1}} = \varphi. \tag{4.33}$$

Attention will be restricted to the two-dimensional case. As another note, the point sets will be referred to as examples of lattices. Lattices are sets of points that form a group under the usual addition operation [134]. This means that for lattice $\mathcal{L} \subset \mathbb{R}^\eth$ the following properties are satisfied:

1. $\mathbf{0} \in \mathcal{L}$,
2. For all $\mathbf{x} \in \mathcal{L}$, $-\mathbf{x} \in \mathcal{L}$,
3. For all $\mathbf{x}, \mathbf{y} \in \mathcal{L}$, $\mathbf{x} + \mathbf{y} \in \mathcal{L}$.

The advantage of a lattice as opposed to other assorted point sets is the ease of search in optimization. The lattice property will not be used below, but since Chernoff fusion can be used with general distributions, using a lattice can lead to easier maximum likelihood searches. This is another reason for preferring the Fibonacci points over point sets which are not lattices.

The Niederreiter-Sloan lattice (NSL) is the traditional Fibonacci lattice defined by the following construction:

$$\mathcal{F}_{\text{NS}}^{(\mathfrak{F}_n)} \coloneqq \left\{ k \cdot \begin{bmatrix} 1/\mathfrak{F}_n \\ \mathfrak{F}_{n-1}/\mathfrak{F}_n \end{bmatrix} \mod 1 : k = 0, 1, 2, ..., \mathfrak{F}_n - 1 \right\} \quad \text{for } \mathfrak{F}_n \in \mathfrak{F}. \tag{4.34}$$

It notably only provides lattices with Fibonacci number cardinalities. However, the NSL is the definition assumed when properties of Fibonacci lattices are proven regarding dispersion and other sampling metrics. Therefore, when claims about optimality are made, they often are in reference to the NSL.

Aiming to extend the NSL to any number of points, one can consider the Kronecker lattice, which is defined for the unit volume of $\mathbb{R}^{\eth}$ as:

$$\mathcal{K}_{\boldsymbol{\alpha}}^{(L)} \coloneqq \left\{ k \cdot \begin{bmatrix} 1/L \\ \\ \boldsymbol{\alpha} \end{bmatrix} \mod 1 : k = 0, 1, 2, ..., L-1 \right\} \quad \text{where } \alpha \in \mathbb{R}^{\eth - 1}. \tag{4.35}$$

Observe that the second element of a NSL is the inverse of the ratio considered in Equation (4.33) and the first element is simply the inverse of the number of points in the lattice. Therefore, using the limit equivalence in Equation (4.33), we can construct a Kronecker lattice with $\boldsymbol{\alpha} = 1/\varphi$ which should asymptotically approach $\mathcal{F}_{\text{NS}}^{(\mathfrak{F}_n)}$ as n approaches infinity. This lattice, here referred to as a Kronecker-Jacob lattice (KJL) (in reference to Simon Jacob who noted that the Fibonacci numbers converged to the golden ratio [135]), is constructed as follows:

$$\mathcal{F}_{\text{KJ}}^{(L)} \coloneqq \mathcal{K}_{1/\varphi}^{(L)} = \left\{ k \cdot \begin{bmatrix} 1/L \\ \\ 1/\varphi \end{bmatrix} \mod 1 : k = 0, 1, 2, ..., L-1 \right\}. \tag{4.36}$$

Recall that the number of points in Figure 4.2 was 100, which is not a Fibonacci number. Those points shown in Figure 4.2d were in fact points from $\mathcal{F}_{\text{KJ}}^{(100)}$. The Binet formula provides an exact representation of the Fibonacci numbers in terms of the golden ratio [125], [133], [136]. It can be stated as:

$$\mathfrak{F}_n = \frac{\varphi^n - (-\varphi)^{-n}}{\varphi + \varphi^{-1}} = \frac{\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n}{\frac{1+\sqrt{5}}{2} - \frac{1-\sqrt{5}}{2}} = \frac{\left(1+\sqrt{5}\right)^n - \left(1-\sqrt{5}\right)^n}{2^n\sqrt{5}} \tag{4.37}$$

Using Equation 4.37, we can consider how appropriate a KJL is by comparing the second elements of $\mathcal{F}_{\text{KJ}}^{(\mathfrak{F}_n)}$ and $\mathcal{F}_{\text{NS}}^{(\mathfrak{F}_n)}$. We compute the following well-known expression for a ratio of

consecutive Fibonacci numbers:

$$\frac{\mathfrak{F}_{n-1}}{\mathfrak{F}_n} = \frac{\frac{\varphi^{n-1}-(-1)^{n-1}\varphi^{-n+1}}{\varphi-(-\varphi)^{-1}}}{\frac{\varphi^{n}-(-1)^{n}\varphi^{-n}}{\varphi-(-\varphi)^{-1}}} \tag{4.38}$$

$$= \frac{\varphi^{n-1}-(-1)^{n-1}\varphi^{-n+1}}{\varphi^{n}-(-1)^{n}\varphi^{-n}} \tag{4.39}$$

$$= \frac{\varphi^{n-1}-(-1)^{n-1}\varphi^{-n+1}}{\varphi^{-n}(\varphi^{2n}-(-1)^{n})} \tag{4.40}$$

$$= \frac{\varphi^{2n-1}-(-1)^{n-1}\varphi}{\varphi^{2n}-(-1)^{n}}. \tag{4.41}$$

The asymptotic limit as n approaches infinity is clearly $1/\varphi$ as was asserted in Equation 4.41. Also, it is now possible to express the absolute error of substituting $1/\varphi$ for the ratio $\mathfrak{F}_{n-1}/\mathfrak{F}_n$ in terms of only the golden ratio φ and the Fibonacci number index n:

$$\varepsilon_2 = \left| \frac{\mathfrak{F}_{n-1}}{\mathfrak{F}_n} - \frac{1}{\varphi} \right| \tag{4.42}$$

$$= \left| \frac{\varphi^{2n-1}-(-1)^{n-1}\varphi}{\varphi^{2n}-(-1)^{n}} - \frac{1}{\varphi} \right|. \tag{4.43}$$

Equation 4.43 is plotted in Figure 4.4. Reducing the ratio error below 10^{-8} requires $\mathfrak{F}_{20} = 6,765$ points, and one requires $\mathfrak{F}_{40} = 102,334,155$ points to push ε_2 below 10^{-16}. However, these errors are only valid for $k = 1$. The error is multiplied by k, which is maximized by $L - 1$ as shown in Figure 4.5a. Since the distance between NSL and KJL points for $L = \mathfrak{F}_n \in \mathfrak{F}$ is maximized by taking $k = \mathfrak{F}_n - 1$, the worst error for the approximation is:

$$(\mathfrak{F}_n - 1)\varepsilon_2 = \left| \left(\frac{\varphi^{n}-(-\varphi)^{-n}}{\sqrt{5}} - 1 \right) \left(\frac{\varphi^{2n-1}-(-1)^{n-1}\varphi}{\varphi^{2n}-(-1)^{n}} - \frac{1}{\varphi} \right) \right| \tag{4.44}$$

which is plotted against the case for $k = 1$ in Figure 4.5b. It is important to note that this does not necessarily invalidate using the KJL points. The "error" presented here is the distance between the NSL points and their corresponding KJL points, which does not directly translate into an error for the QMC integration. It does however show that far

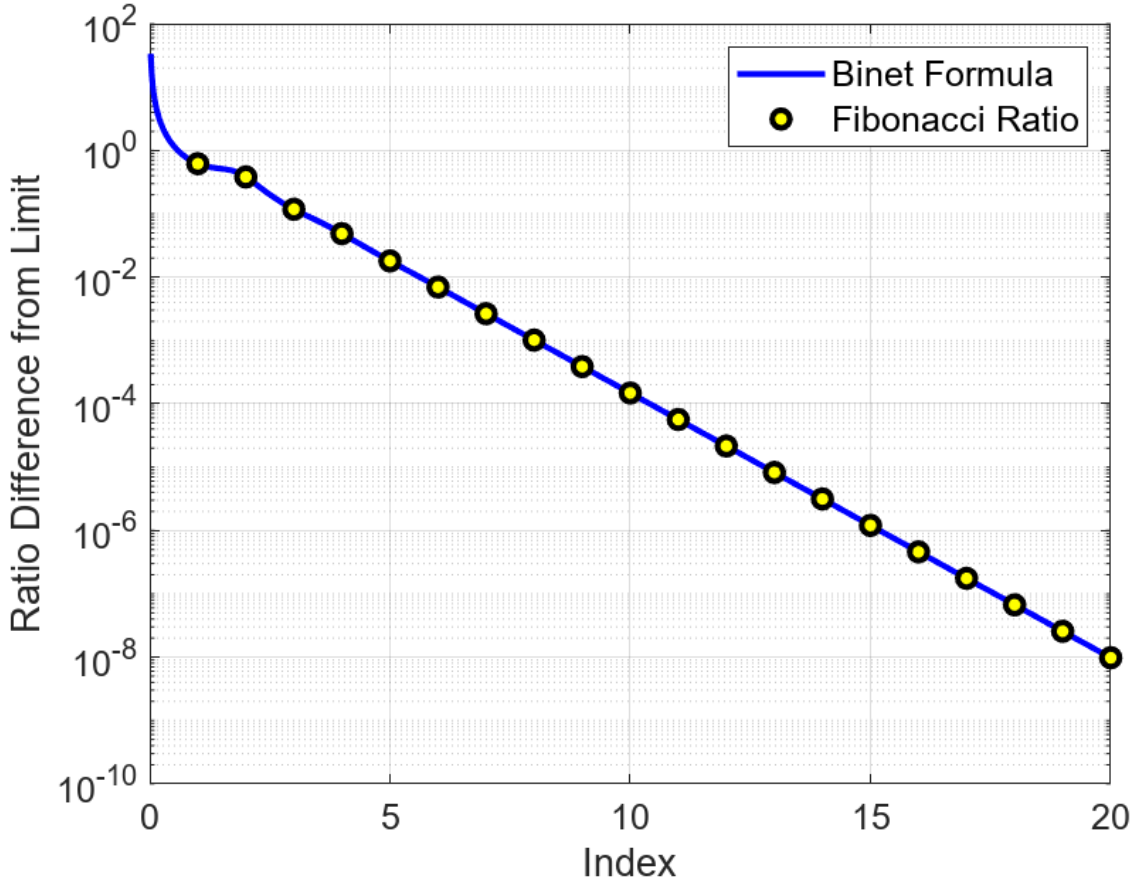
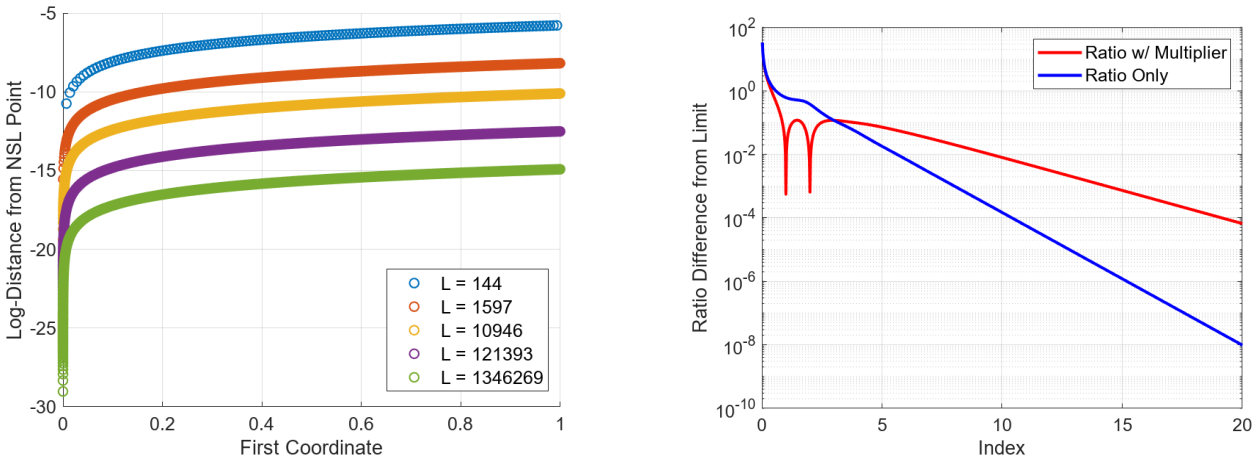


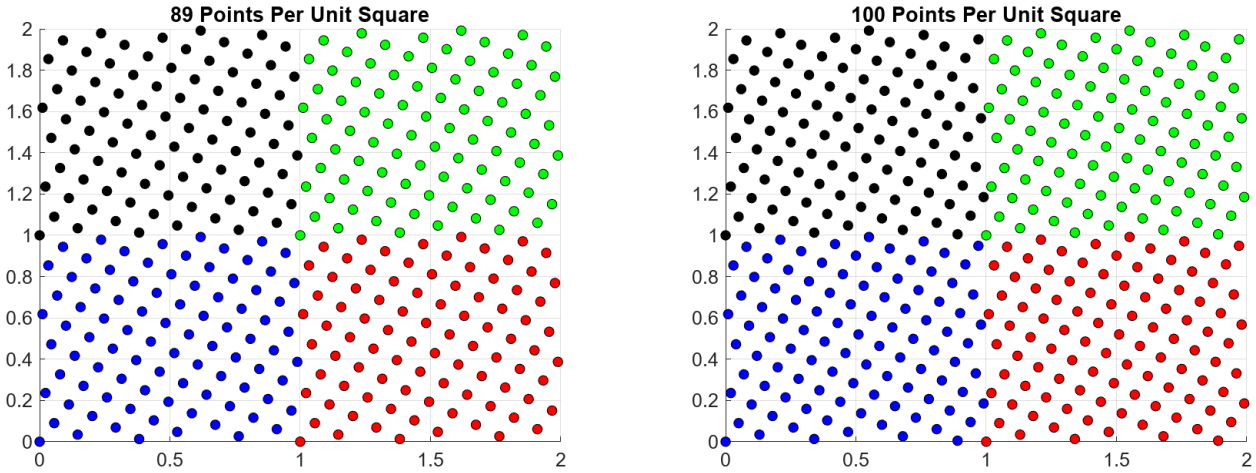
Figure 4.4: The absolute difference between the ratio of consecutive Fibonacci numbers and the limit of $1/\varphi$ is shown. The blue curve uses the expression in Equation 4.43, and the yellow dots show the results of computing the differences using the Fibonacci numbers $\mathfrak{F}_{n-1}/\mathfrak{F}_n$.

too many points are needed before the NSL and KJL are considered indistinguishable to a computer. This means that some properties of the NSL will not be guaranteed for the KJL. One example of such a property is periodicity. The NSL is a good lattice for the torus in part due to periodicity in both dimensions. This periodicity is not retained in the first coordinate of the KJL for $L \notin \mathfrak{F}$ as shown in Figure 4.6 [137]. Therefore, it is reasonable to test both lattices given their limitations. With this in mind, we proceed to consider a final alternative formulation of the Fibonacci matrix.



(a) **(b)**

Figure 4.5: In (a), the log-distance between each NSL and KJL point is shown for the first Fibonacci numbers greater than 1e2, 1e3, 1e4, 1e5, and 1e6 respectively. In (b), the worst-case separation of NSL and KJL points is plotted (red) against the case for $k = 1$ (blue).



(a) **(b)**

Figure 4.6: KJL points are plotted in four adjacent unit squares to demonstrate the loss of periodicity in the first coordinate when the number of points is not a Fibonacci number.

The Swinbank-Purser points (SPP) set is constructed from an alternative presentation of the Fibonacci numbers using matrices. The Fibonacci matrix is defined as

$$\mathfrak{F} \coloneqq \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix} \tag{4.45}$$

which has the eigenvalues of φ and $-1/\varphi$ [138]. Using the eigenvectors corresponding to these eigenvalues, one can scale and rotate a regular grid and prune to $[\mathbf{0}, \mathbf{1}]$ in such a way that the number of points is guaranteed. An algorithm for generating the SPP is given in [116] and is used to generate the SPP points considered here.

To illustrate the difference between SPP and the more traditional Fibonacci lattices, one can look at a property regarding the unit cells of each lattice. First, define a generating set of a lattice as a set of vectors which can be used to reproduce every element of the lattice using addition and negation. Here we will go further and consider only the minimal such set, which will contain 2 elements for the examples given. Unit cells of a lattice are defined as the parallelograms formed using the generators of a lattice. The NSL contains square unit cells if and only if the Fibonacci number's subscript is odd. This matters for approximating bilinear functions, as such functions are integrated exactly if the lattice is invariant under a 90 degree rotation [114], [134]. The KJL does not retain the square unit cell property of the NSL, but somewhat emulates the alternating pattern as expected. The SPP does not retain the pattern at all. For 8 and 144 points (the 6th and 12th Fibonacci numbers, respectively), the SPP was observed to be square. Some example lattices are shown for the NSL, KJL, and SPP using 89 and 144 points in Figures 4.7, 4.8, and 4.9. The angles for representative generating sets are plotted in Figure 4.10.

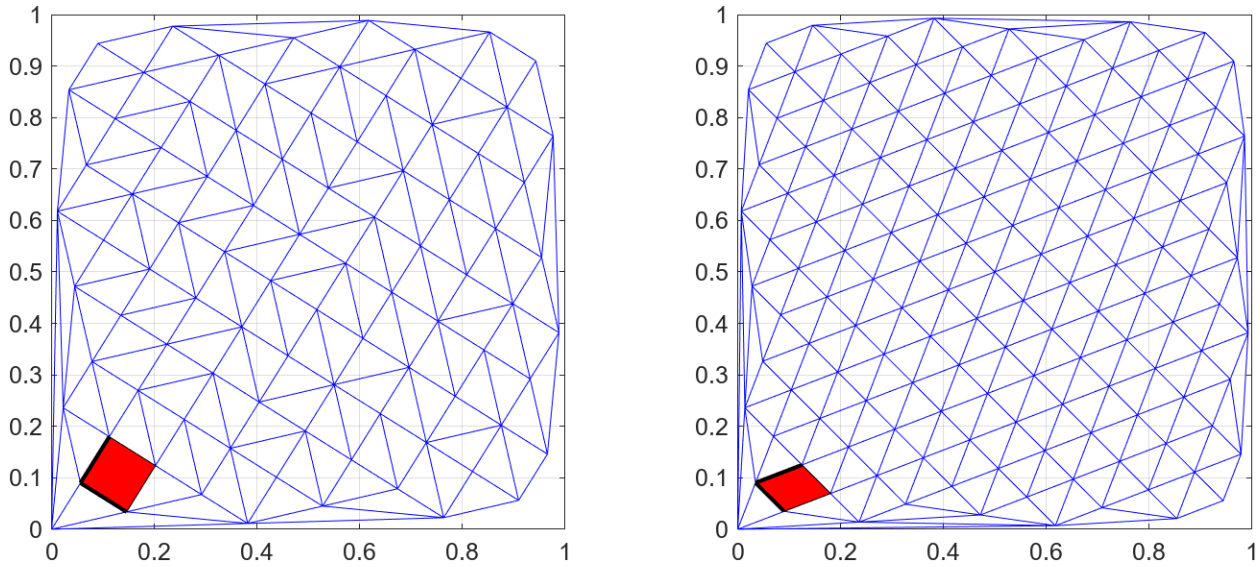


Figure 4.7: A Delaunay triangulation of the NSL for 89 (a) and 144 (b) points. The black lines highlight the generator set for the lattices and the red patches identify examples of unit cells.

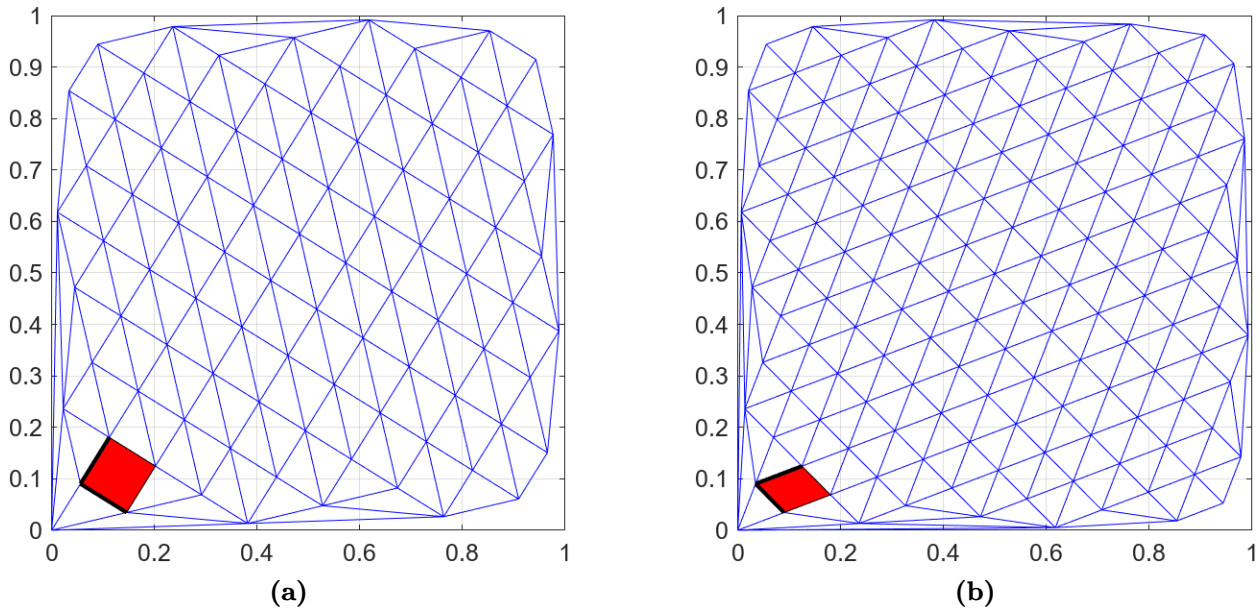


Figure 4.8: A Delaunay triangulation of the KJL for 89 (a) and 144 (b) points. The black lines highlight the generator set for the lattices and the red patches identify examples of unit cells.

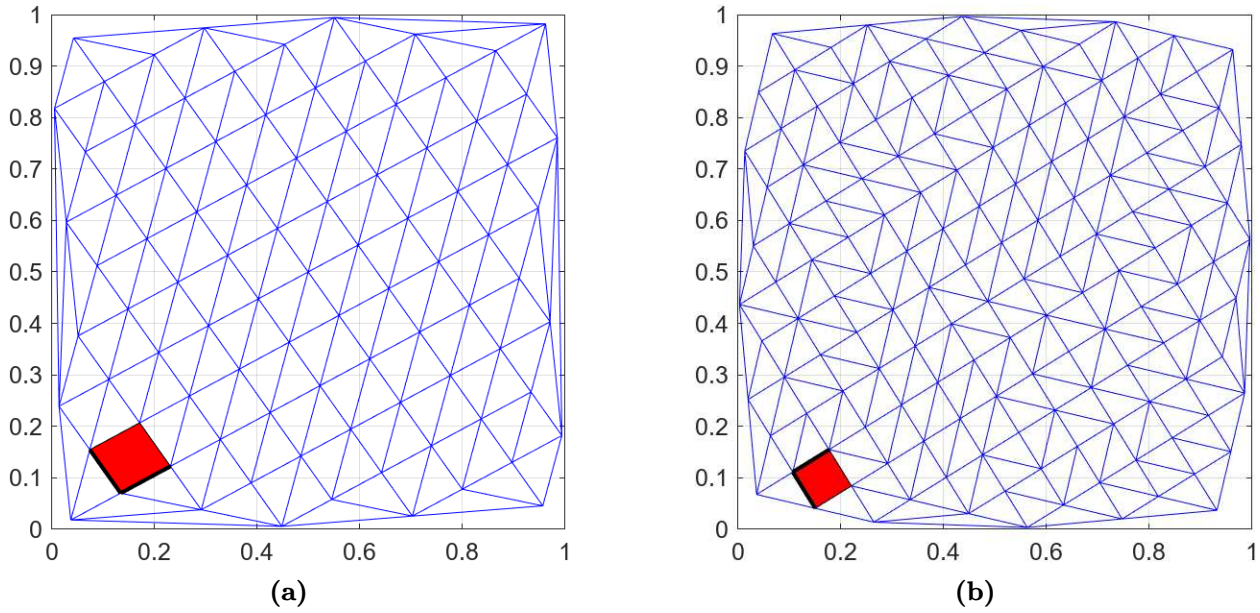


Figure 4.9: A Delaunay triangulation of the SPP for 89 (a) and 144 (b) points. The black lines highlight the generator set for the lattices and the red patches identify examples of unit cells.

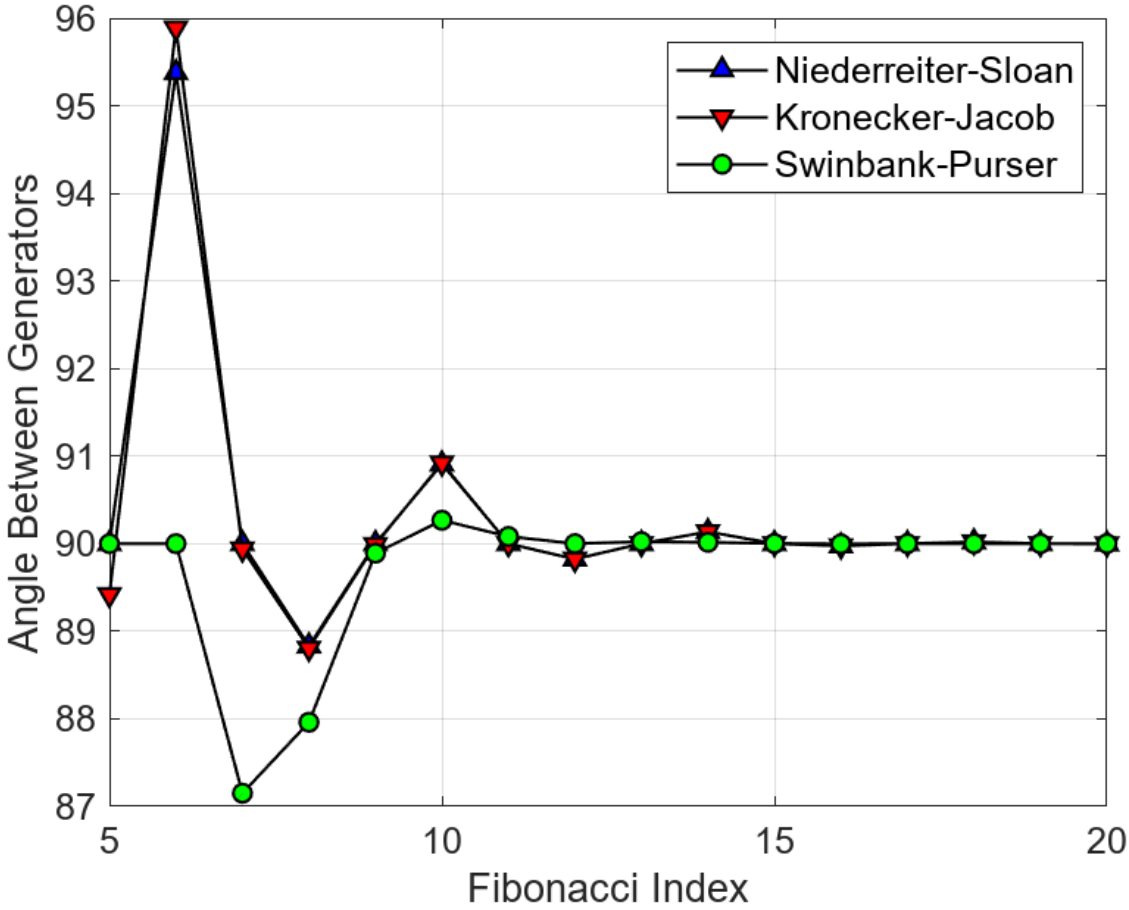


Figure 4.10: Angle between chosen generators for different Fibonacci number indices.

4.5 Evaluation of Chernoff Fusion

With the point sets discussed above, we will now observe how closely the Chernoff fusion QMC approximation matches the true distribution given by covariance intersection. Slightly offset means were chosen for two reasons. First, different estimation procedures rarely result in exactly the same expected value when the range is continuous. Therefore, the more realistic case is slightly offset means. Second, due to symmetries (especially for the regular grid), it can happen that perfect alignment of the fused estimates' means occurs, resulting in zero error. This does not show up on a loglog plot and is generally unhelpful, as we wish to test the approximation suitability without benefiting from coincidences. As a note, in these symmetric cases, the algorithms all demonstrate similar performance to what is highlighted below.

4.5.1 Perpendicular Covariances

First, we will consider perpendicular covariances. The goal is to evaluate the ability of the point sets to approximate the fused distribution when the bounding ellipses for 99% of each input distribution's mass is contained within the unit square. An example of the scenario considered and fusion comparison is shown in Figure 4.11. The input Gaussian parameters are

$$\mathbf{x}_1 = \begin{bmatrix} 0.55 \\ 0.55 \end{bmatrix} \qquad \mathbf{P}_1 = \begin{bmatrix} 0.002 & 0 \\ 0 & 0.02 \end{bmatrix} \tag{4.46}$$

$$\mathbf{x}_2 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \qquad \mathbf{P}_2 = \begin{bmatrix} 0.02 & 0 \\ 0 & 0.002 \end{bmatrix} \tag{4.47}$$

and the output of covariance intersection (rounded to four digits) is

$$\mathbf{x}_f = \begin{bmatrix} 0.5454 \\ \\ 0.5045 \end{bmatrix} \qquad \mathbf{P}_f = \begin{bmatrix} 0.0036 & 0 \\ \\ 0 & 0.0036 \end{bmatrix}. \tag{4.48}$$

Metrics for assessing the Chernoff fusion results are depicted in Figures 4.12 and 4.13. First, the Kullback-Leibler divergence is shown in Figure 4.12 to compare the Chernoff fusion and covariance intersection outputs as distributions. It can be seen that the Fibonacci point sets and regular grid outperform the Halton, Sobolev, and Monte Carlo points for large numbers of points (greater than 100). For an intermediate number of points (10 to 100), the Fibonacci points maintain better distribution estimation performance than the regular grid and other point sets. For more intuitive metrics, we turn to comparisons of the means and covariances in Figure 4.13. The 2-norm of the difference between the means most starkly exhibits a plateau after around $\mathfrak{F}_{13} = 233$ points. This is likely the result of the entropy approximations made in determining the weights for Chernoff fusion. It is notable that the regular grid seems to yield greater accuracy, but then returns to the same level as the Fibonacci points. This likely indicates that one should judge the Fibonacci points as more correctly representing the true plateau in performance sooner. Additionally, the Fibonacci points demonstrate improved performance over the regular grid through intermediate numbers of points, albeit a marginal improvement. Moving to the Frobenius norm of the difference between covariances, the Fibonacci points again all exhibit improvements over the other point sets through the intermediate numbers of points. The regular grid only yields equivalent performance after the aforementioned $\mathfrak{F}_{13} = 233$ points.

4.5.2 Oblique Covariances

Now the case of oblique covariances will be tested. The means from the perpendicular example will remain the same, but the covariance ellipse for the first state will be rotated 60 degrees counterclockwise. This yields the following state parameters (rounded to four

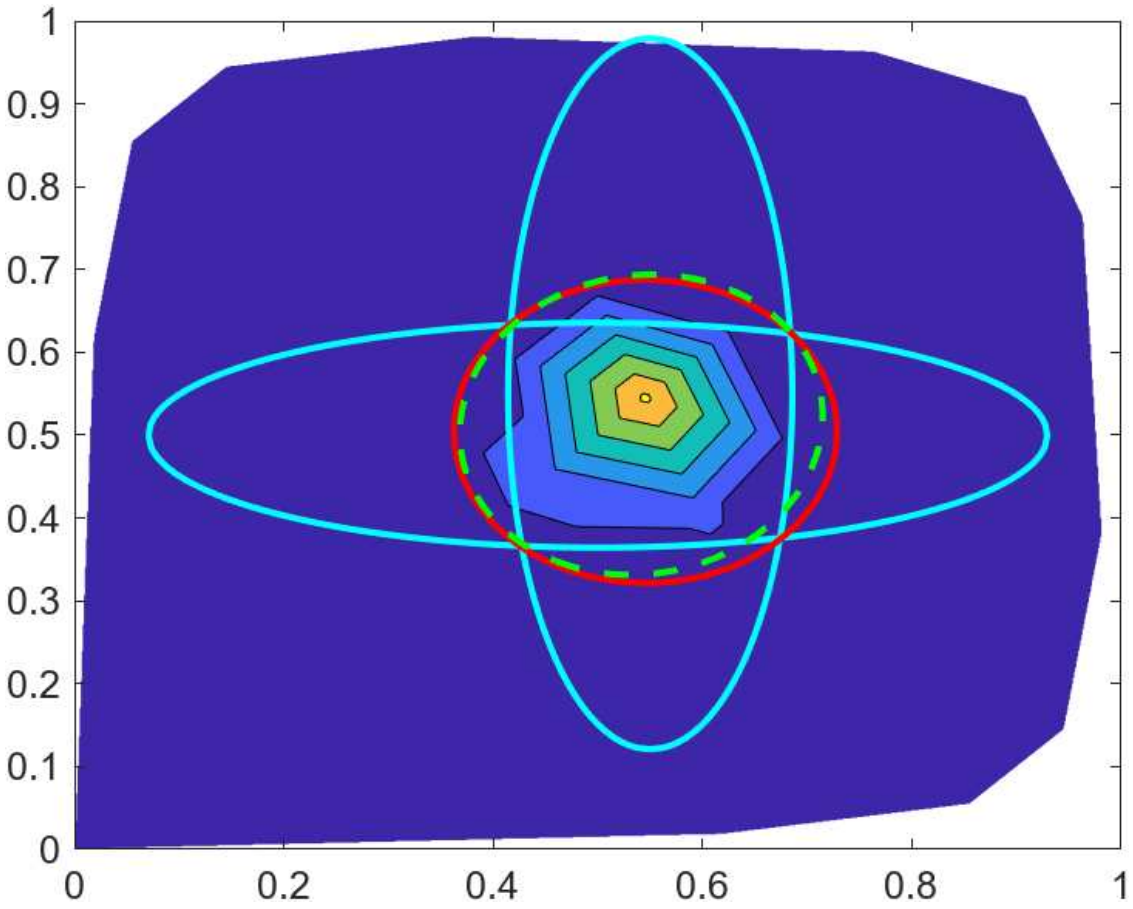


Figure 4.11: The considered scenario with an example of what an approximate contour looks like for a 50 point NSL. Each ellipse contains 99% of their respective Gaussian distribution. The input distributions are shown in cyan, the covariance intersection fusion result is shown in red, and the Chernoff fusion result is shown in dashed green.

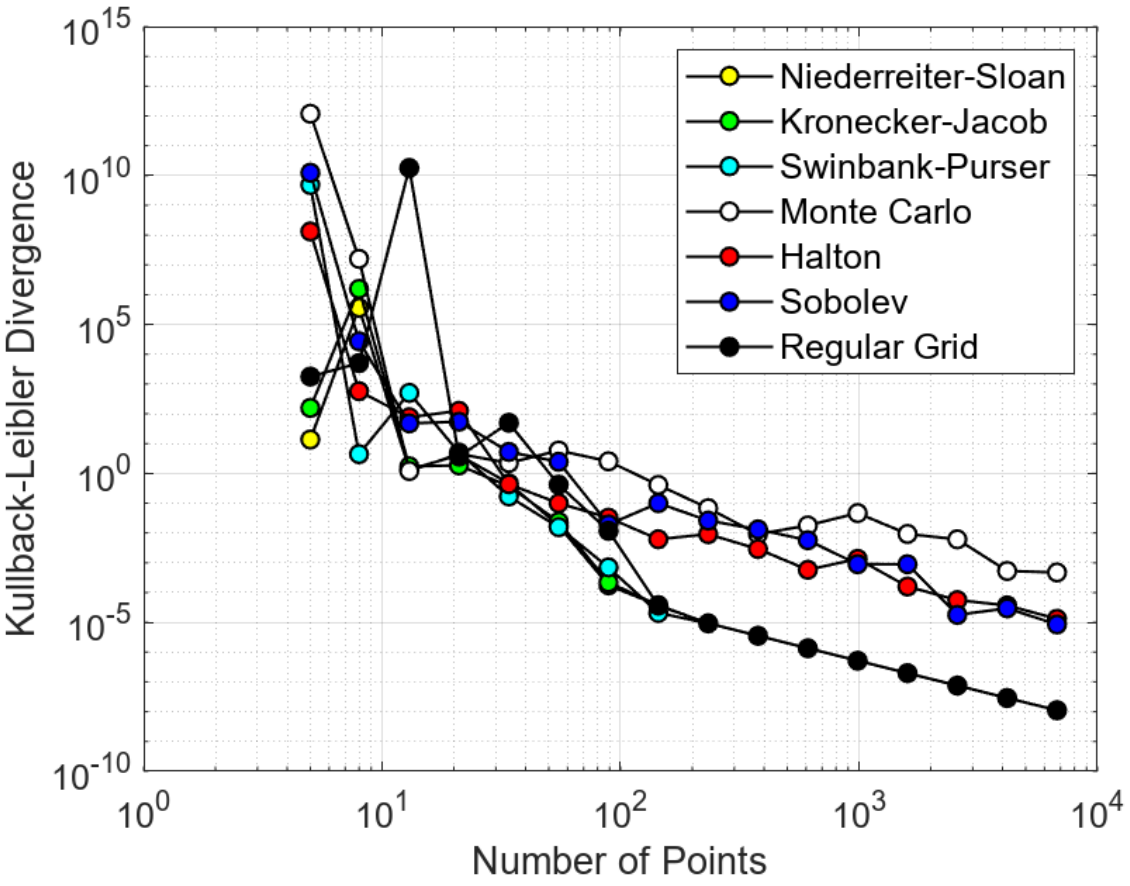
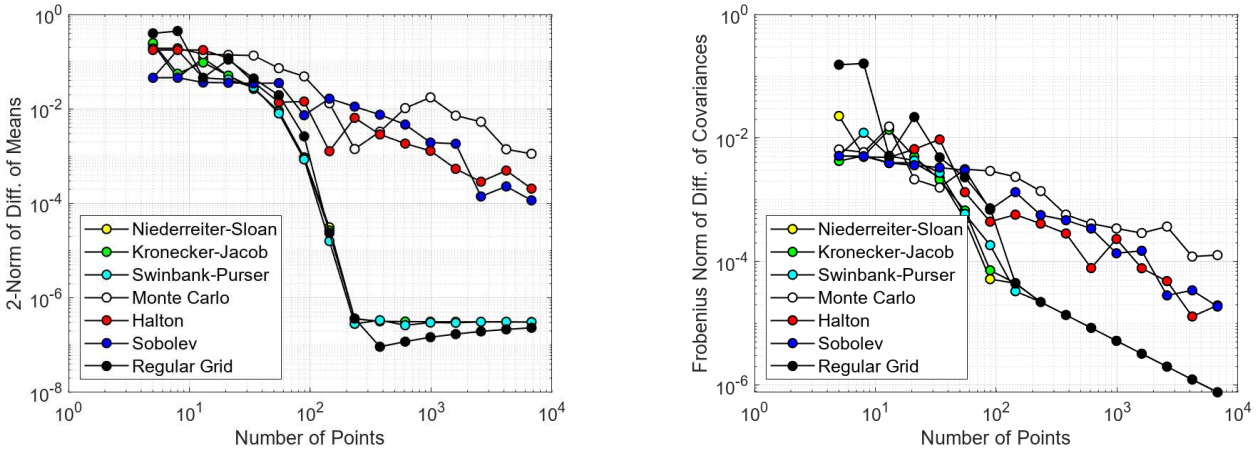


Figure 4.12: The KL divergence for the fused Chernoff fusion (approximating model) and covariance intersection (truth) distributions.



(a) **(b)**

Figure 4.13: Norm differences between the means and covariances for the fused Chernoff fusion and covariance intersection distributions.

digits):

$$\mathbf{x}_1 = \begin{bmatrix} 0.55 \\ 0.55 \end{bmatrix} \qquad \mathbf{P}_1 = \begin{bmatrix} 0.0155 & -0.0078 \\ -0.0078 & 0.0065 \end{bmatrix} \tag{4.49}$$

$$\mathbf{x}_2 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \qquad \mathbf{P}_2 = \begin{bmatrix} 0.02 & 0 \\ 0 & 0.002 \end{bmatrix} \tag{4.50}$$

and the output of covariance intersection (rounded to four digits) is

$$\mathbf{x}_f = \begin{bmatrix} 0.5676 \\ 0.5180 \end{bmatrix} \qquad \mathbf{P}_f = \begin{bmatrix} 0.0118 & -0.0026 \\ -0.0026 & 0.0028 \end{bmatrix}. \tag{4.51}$$

The scenario is shown in Figure 4.15 and metrics are provided in Figures 4.15 and 4.16. The metrics largely show the same behavior as noted in the perpendicular case.

4.5.3 Parallel Covariances

Finally, the case of parallel covariances will be tested. As with the oblique case, the means from the above examples will remain the same, but the covariance ellipse for the first state will be rotated 90 degrees counterclockwise, making it identical to P_2. This yields the following state parameters (rounded to four digits):

$$\mathbf{x}_1 = \begin{bmatrix} 0.55 \\ 0.55 \end{bmatrix} \qquad \mathbf{P}_1 = \begin{bmatrix} 0.02 & 0 \\ 0 & 0.002 \end{bmatrix} \tag{4.52}$$

$$\mathbf{x}_2 = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \qquad \mathbf{P}_2 = \begin{bmatrix} 0.02 & 0 \\ 0 & 0.002 \end{bmatrix} \tag{4.53}$$

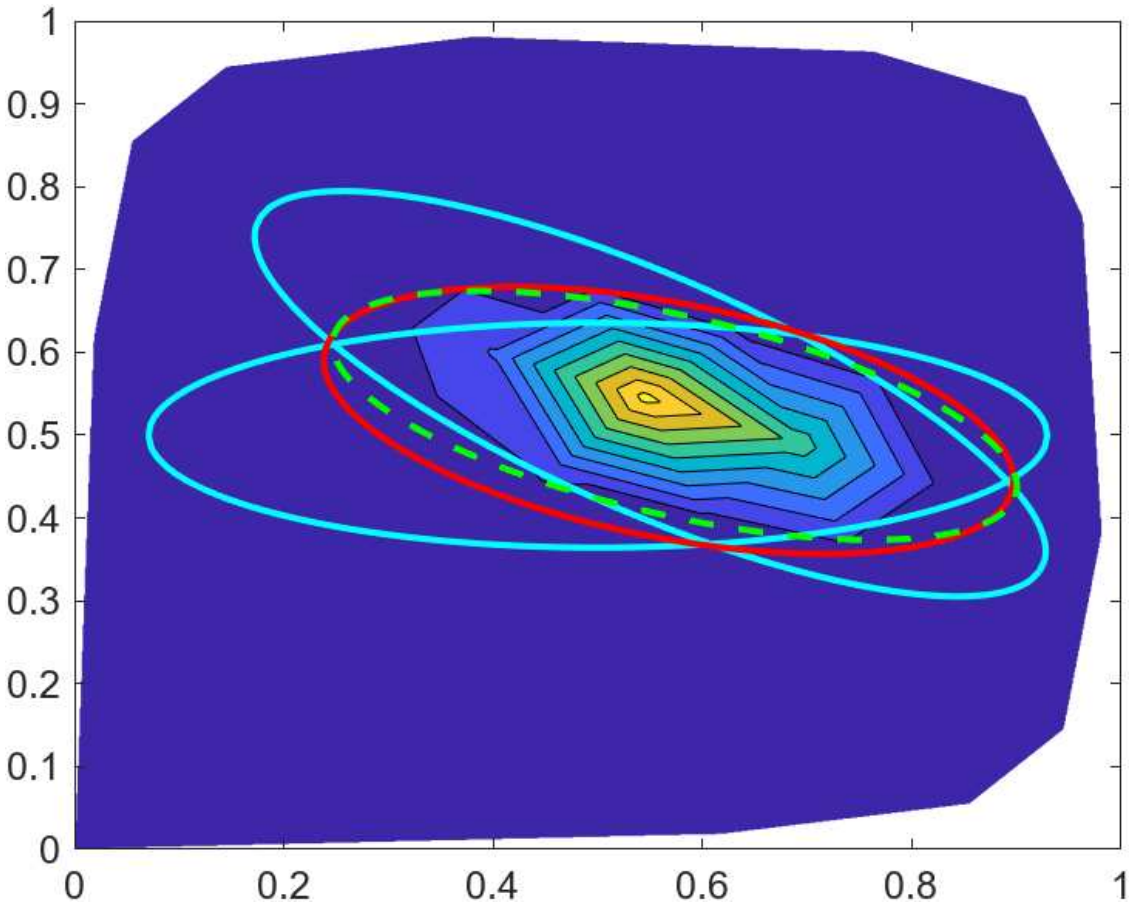


Figure 4.14: The considered scenario with an example of what an approximate contour looks like for a 50 point NSL. Each ellipse contains 99% of their respective Gaussian distribution. The input distributions are shown in cyan, the covariance intersection fusion result is shown in red, and the Chernoff fusion result is shown in dashed green.

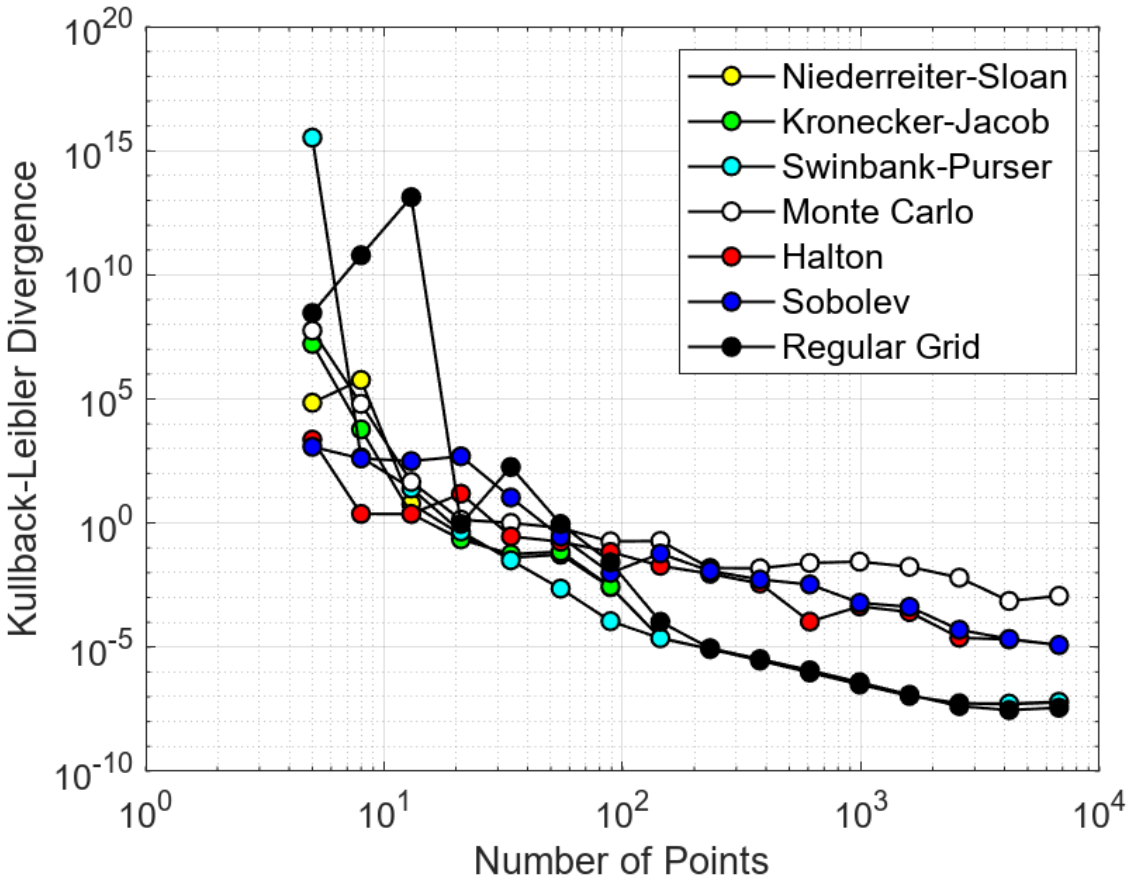
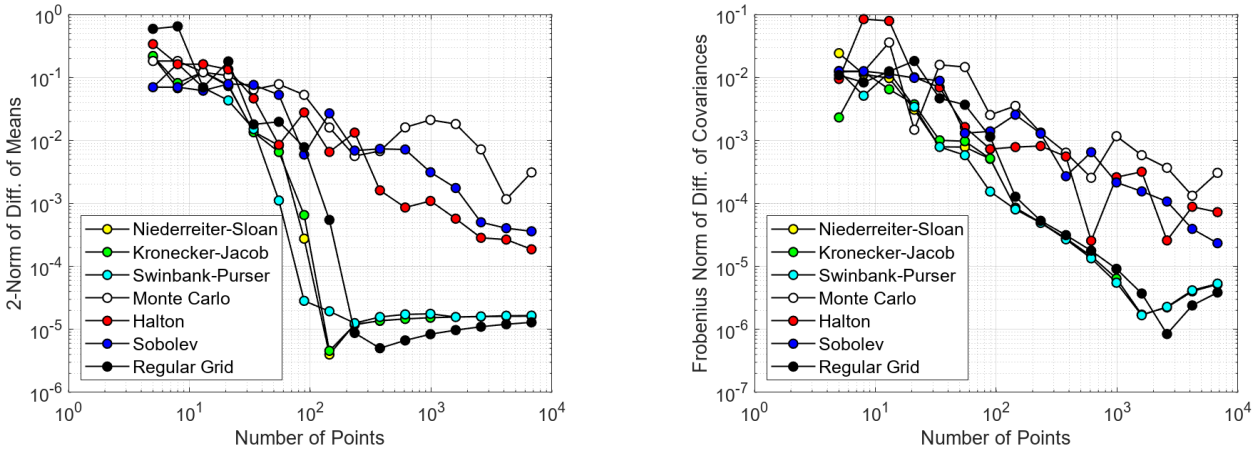


Figure 4.15: The KL divergence for the fused Chernoff fusion (approximating model) and covariance intersection (truth) distributions.



(a) **(b)**

Figure 4.16: Norm differences between the means and covariances for the fused Chernoff fusion and covariance intersection distributions.

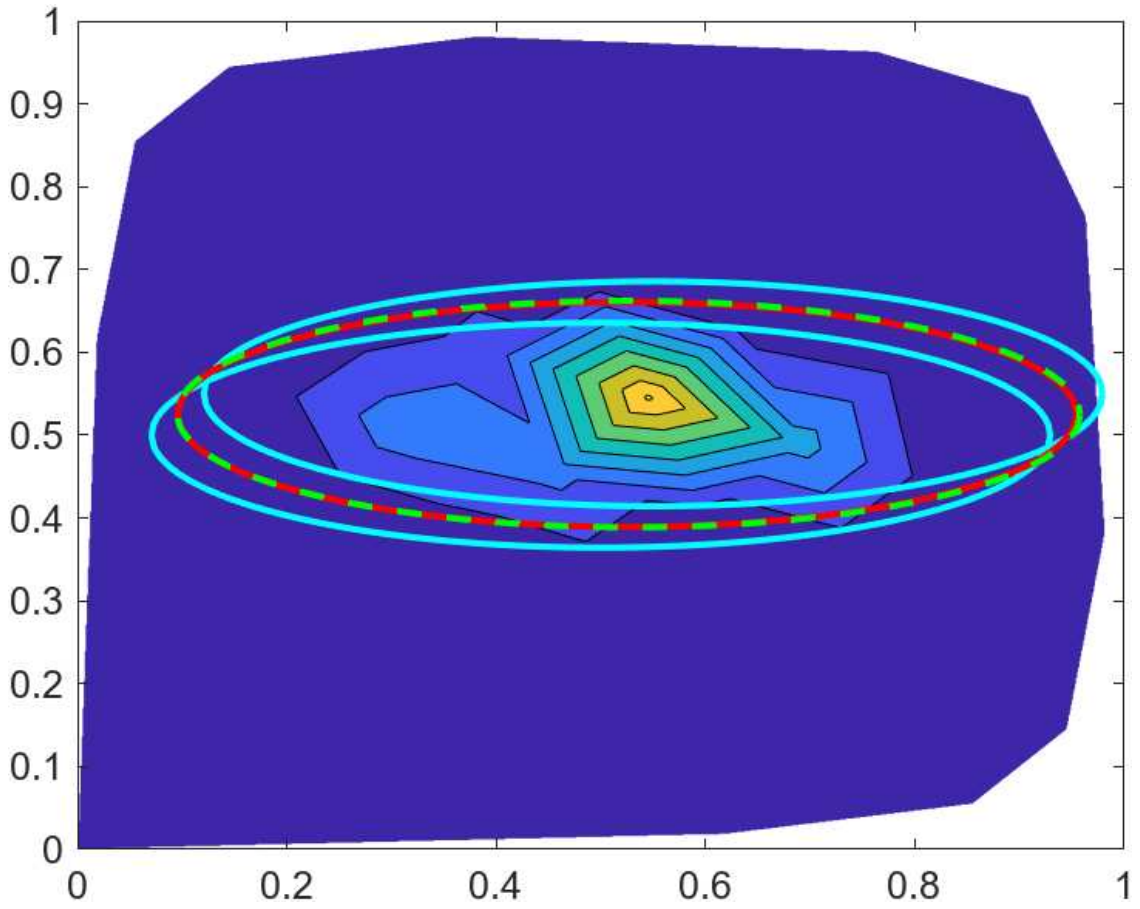


Figure 4.17: The considered scenario with an example of what an approximate contour looks like for a 50 point NSL. Each ellipse contains 99% of their respective Gaussian distribution. The input distributions are shown in cyan, the covariance intersection fusion result is shown in red, and the Chernoff fusion result is shown in dashed green.

and the output of covariance intersection (rounded to four digits) is

$$\mathbf{x}_f = \begin{bmatrix} 0.525 \\ 0.525 \end{bmatrix} \qquad \mathbf{P}_f = \begin{bmatrix} 0.02 & 0 \\ 0 & 0.002 \end{bmatrix}. \tag{4.54}$$

The scenario is shown in Figure 4.18 and metrics are provided in Figures 4.18 and 4.19. The metrics largely show the same behavior as noted in the perpendicular case.

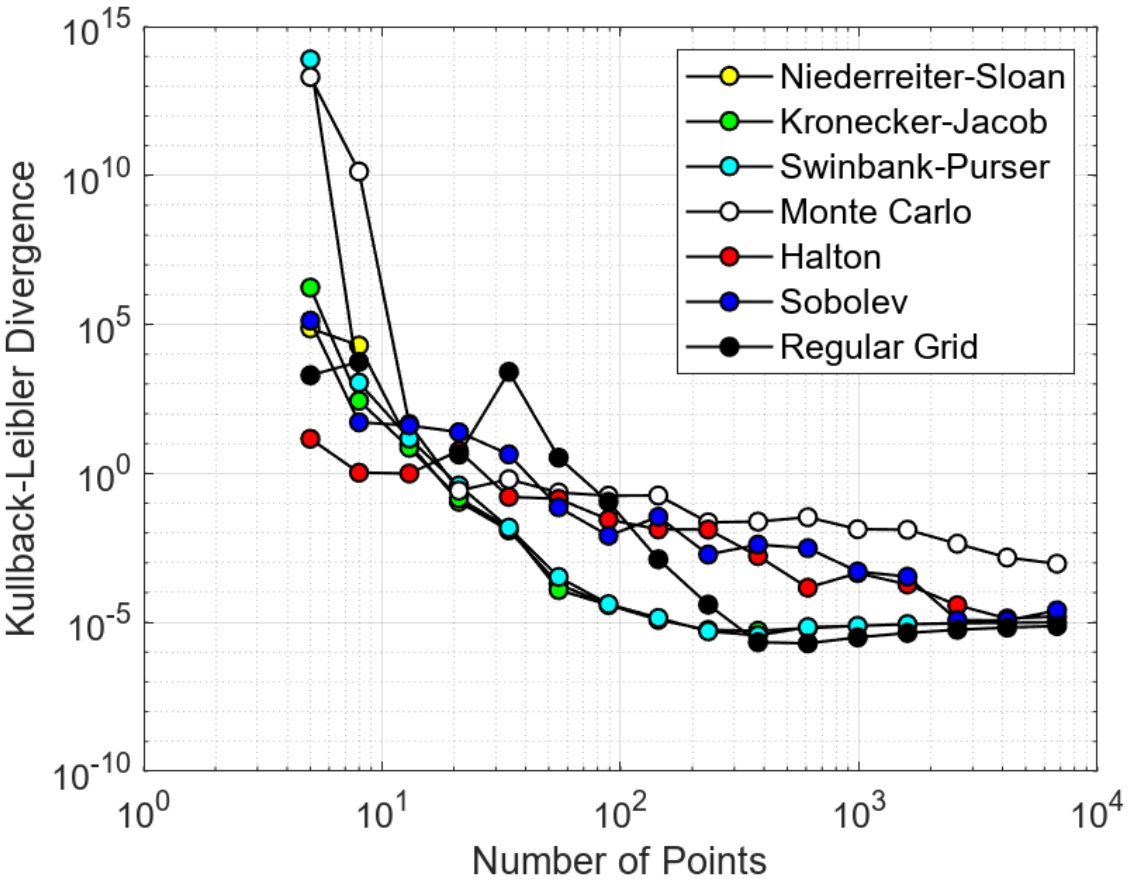
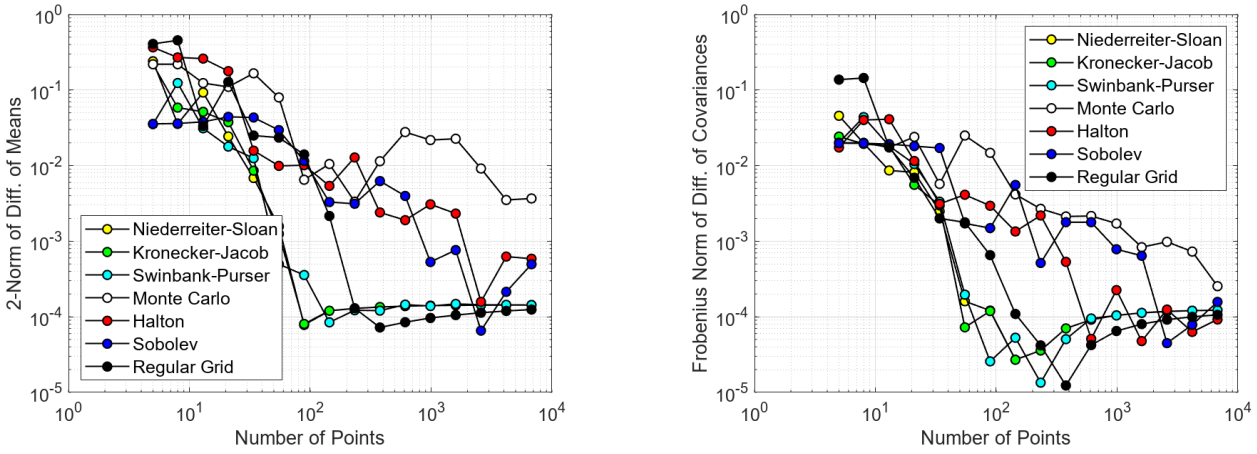
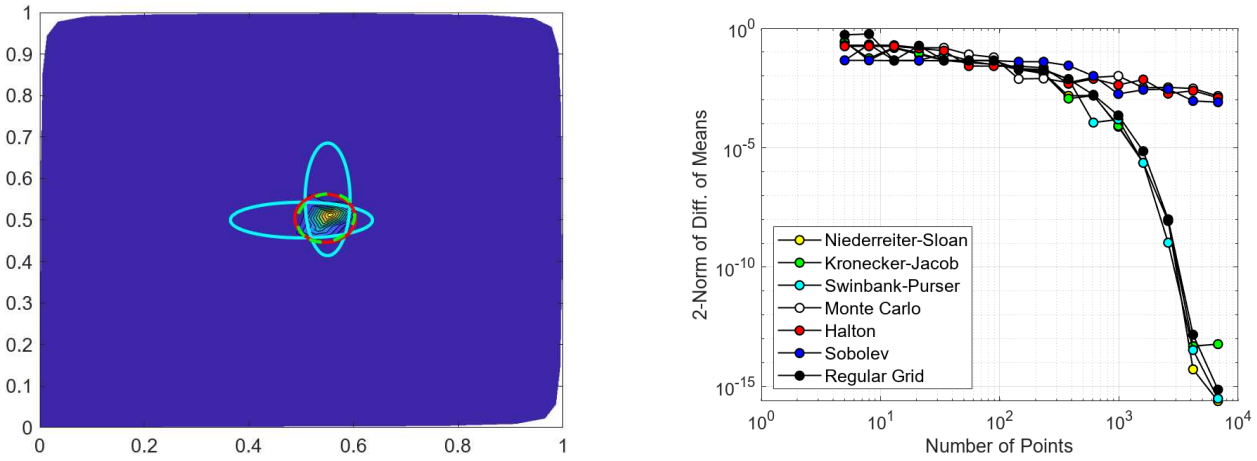


Figure 4.18: The KL divergence for the fused Chernoff fusion (approximating model) and covariance intersection (truth) distributions.



(a) **(b)**

Figure 4.19: Norm differences between the means and covariances for the fused Chernoff fusion and covariance intersection distributions.



(a) (b)

Figure 4.20: A scenario where the distributions to be fused are well contained by the unit square is depicted (a) with 500 NSL points and the resulting norm difference of means plot is shown (b). There is a noticeable lack of a plateau which was exhibited in the scenarios with larger covariances.

4.5.4 Discussion on Plateau of Means Error

The reason for a plateau in agreement between the means was not proven, but there is a likely culprit. Restriction of the distributions to the unit square truncates the Gaussian states being fused. This truncation biases the means if that truncation is not rotationally symmetric about the mean. To demonstrate this, the perpendicular example was reproduced using covariance matrices that scaled down by a factor of 10, shrinking the probability mass outside of the unit square as shown in Figure 4.20a. The resulting normed difference of means is shown in Figure 4.20b. If there is a bias, it is clearly smaller than that exhibited in the examples above. Regardless of the cause of the bias though, the results suggest there may be a persistent bias induced by QMC approximations of Chernoff fusion which should be factored into downstream calculations for a tracker.

4.6 Conclusion

In this chapter, I have compared the efficiency of three types of Fibonacci points to commonly used alternatives for the purpose of Chernoff fusion. An improvement in three metrics was observed for the fused distribution. In pursuit of these results, the utility of a

particular instance of the covariance intersection algorithm for ground truth comparison was noted and distinctions between the three types of Fibonacci points were analyzed.

Chapter 5

Conclusion

This work has presented improvements to two important tracking algorithms: GNP and IMM. Chapter 2 showed why taking volume units into account is essential for a correct result in association between biased sensor measurements and provided a gating criterion based on the included volume term. A method of computing the probability of error for a chosen hypothesis was demonstrated, and an application of the GNP to a bias reduction scenario was provided.

Chapter 3 presented an improvement to the traditional implementation of the IMM filter by using a pre-trained neural network and its confusion matrix, replacing dubious assumptions about target maneuver tactics. The training considerations of an example neural network were discussed, and a comparison against a traditional IMM algorithm was done showing favorable performance.

Chapter 4 presented an analysis of the potential for improved Chernoff fusion results with Fibonacci points. These results suggest that more efficient use of samples for QMC estimation is yielded through Fibonacci points than other common choices. The distinctiveness of each of the types of Fibonacci point sets was also explored to demonstrate that the three types are in fact distinct for any reasonable number of sample points.

Bibliography

[1] Y. Bar-Shalom, Ed., *Multitarget-Multisensor Tracking: Applications and Advances*, 1st ed. Artech House, 1992, vol. 2, ISBN: 0-89006-517-9.

[2] Y. Bar-Shalom and W. D. Blair, Eds., *Multitarget-multisensor tracking: applications and advances*, 1st ed. Artech House, 2000, vol. 3, ISBN: 1-58053-091-5.

[3] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and data fusion*, 1st ed. YBS Publishing, 2011, ISBN: 978-0964-8312-78.

[4] S. S. Blackman and R. Popoli, *Design and analysis of modern tracking systems*, 1st ed. Artech House, 1999, ISBN: 978-1580530064.

[5] C.-B. Chang and K.-P. Dunn, *Applied state estimation and association*, 1st ed. MIT Press, 2016, ISBN: 978-80262034005.

[6] A. J. Haug, *Bayesian Estimation and Tracking*. Wiley, Jun. 2012, ISBN: 9780470621707. DOI: 10.1002/9781118287798. [Online]. Available: https://onlinelibrary.wiley.com/doi/book/10.1002/9781118287798.

[7] D. Crouse, "Basic tracking using nonlinear 3d monostatic and bistatic measurements," *IEEE Aerospace and Electronic Systems Magazine*, vol. 29, pp. 4–53, 8 Aug. 2014, ISSN: 08858985. DOI: 10.1109/MAES.2014.120229.

[8] D. Crouse, "Basic tracking using nonlinear 3d monostatic and bistatic measurements in refractive environments," *IEEE Aerospace and Electronic Systems Magazine*, vol. 29, pp. 54–75, 8 Aug. 2014, ISSN: 08858985. DOI: 10.1109/MAES.2014.120230.

[9] D. F. Crouse, "The tracker component library: Free routines for rapid prototyping," *IEEE Aerospace and Electronic Systems Magazine*, vol. 32, pp. 18–27, 5 May 2017, ISSN: 0885-8985. DOI: 10.1109/MAES.2017.160215. [Online]. Available: http://ieeexplore.ieee.org/document/7954147/.

[10] W. Koch, "On bayesian tracking and data fusion: A tutorial introduction with examples," *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, pp. 29–52, 7 Jul. 2010, ISSN: 0885-8985. DOI: 10.1109/MAES.2010.5546307. [Online]. Available: https://ieeexplore.ieee.org/document/5546307/.

[11] R. Mahler, "Nonadditive probability, finite-set statistics, and information fusion," vol. 2, IEEE, 1995, pp. 1947–1952, ISBN: 0-7803-2685-7. DOI: 10.1109/CDC.1995.480631. [Online]. Available: http://ieeexplore.ieee.org/document/480631/.

[12] R. Mahler, ""statistics 101" for multisensor, multitarget data fusion," *IEEE Aerospace and Electronic Systems Magazine*, vol. 19, pp. 53–64, 1 Jan. 2004, ISSN: 0885-8985. DOI: 10.1109/MAES.2004.1263231. [Online]. Available: http://ieeexplore.ieee.org/document/1263231/.

[13] R. Mahler, *Statistical multisource-multitarget information fusion*. Artech House, 2007, ISBN: 9781596930933.

[14] R. Mahler, ""statistics 102" for multisource-multitarget detection and tracking," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, pp. 376–389, 3 Jun. 2013, ISSN: 1932-4553. DOI: 10.1109/JSTSP.2013.2253084. [Online]. Available: http://ieeexplore.ieee.org/document/6490330/.

[15] R. Mahler, ""statistics 103" for multitarget tracking," *Sensors*, vol. 19, p. 202, 1 Jan. 2019, ISSN: 1424-8220. DOI: 10.3390/s19010202. [Online]. Available: https://www.mdpi.com/1424-8220/19/1/202.

[16] R. Streit, "Analytic combinatorics in multiple object tracking," IEEE, Oct. 2017, pp. 1–6, ISBN: 978-1-5386-3103-4. DOI: 10.1109/SDF.2017.8126348. [Online]. Available: http://ieeexplore.ieee.org/document/8126348/.

[17] R. Streit, "How i learned to stop worrying about a thousand and one filters and love analytic combinatorics," IEEE, Mar. 2017, pp. 1–21, ISBN: 978-1-5090-1613-6. DOI:

10.1109/AERO.2017.7943644. [Online]. Available: http://ieeexplore.ieee.org/document/7943644/.

[18] R. Streit, R. B. Angle, and M. Efe, *Analytic Combinatorics for Multiple Object Tracking*, 1st ed. Springer International Publishing, 2021, ISBN: 978-3-030-61190-3. DOI: 10.1007/978-3-030-61191-0. [Online]. Available: http://link.springer.com/10.1007/978-3-030-61191-0.

[19] R. Mahler, "Multitarget bayes filtering via first-order multitarget moments," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, pp. 1152–1178, 4 Oct. 2003, ISSN: 0018-9251. DOI: 10.1109/TAES.2003.1261119.

[20] B.-N. Vo and W.-K. Ma, "The gaussian mixture probability hypothesis density filter," *IEEE Transactions on Signal Processing*, vol. 54, pp. 4091–4104, 11 Nov. 2006, ISSN: 1053-587X. DOI: 10.1109/TSP.2006.881190.

[21] B.-T. Vo, B.-N. Vo, and A. Cantoni, "Analytic implementations of the cardinalized probability hypothesis density filter," *IEEE Transactions on Signal Processing*, vol. 55, pp. 3553–3567, 7 Jul. 2007, ISSN: 1053-587X. DOI: 10.1109/TSP.2007.894241.

[22] S. Nannuru, S. Blouin, M. Coates, and M. Rabbat, "Multisensor cphd filter," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, pp. 1834–1854, 4 Aug. 2016, ISSN: 0018-9251. DOI: 10.1109/TAES.2016.150265. [Online]. Available: http://ieeexplore.ieee.org/document/7738358/.

[23] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking: Dynamic models," O. E. Drummond, Ed., SPIE, Jul. 2000, pp. 212–235. DOI: 10.1117/12.391979. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=905730https://doi.org/10.1117/12.391979.

[24] W. D. Blair and Y. Bar-Shalom, "On the nca versus ncv models in tracking maneuvering targets," vol. 2023-May, Institute of Electrical and Electronics Engineers Inc., 2023, ISBN: 9781665436694. DOI: 10.1109/RadarConf2351548.2023.10149710.

[25] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking: Iii. measurement models," vol. 4473, SPIE, Nov. 2001, p. 423. DOI: 10.1117/12.492752.

[26] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, pp. 35–45, 1 Mar. 1960, ISSN: 0021-9223. DOI: 10.1115/1.3662552. [Online]. Available: https://asmedigitalcollection.asme.org/fluidsengineering/article/82/1/35/397706/A-New-Approach-to-Linear-Filtering-and-Prediction.

[27] J. M. Mendel, *Lessons in estimation theory for signal processing, communications, and control*. Pearson Education, 1995, ISBN: 0-13-120981-7.

[28] S. Julier and J. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, pp. 401–422, 3 Mar. 2004, ISSN: 0018-9219. DOI: 10.1109/JPROC.2003.823141. [Online]. Available: http://ieeexplore.ieee.org/document/1271397/.

[29] S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems," I. Kadar, Ed., Jul. 1997, p. 182. DOI: 10.1117/12.280797. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.280797.

[30] I. Arasaratnam and S. Haykin, "Cubature Kalman filters," *IEEE Transactions on Automatic Control*, vol. 54, pp. 1254–1269, 6 Jun. 2009, ISSN: 0018-9286. DOI: 10.1109/TAC.2009.2019800. [Online]. Available: http://ieeexplore.ieee.org/document/4982682/.

[31] B. Davis, "Efficient high degree cubature Kalman filters with reduced dimension update," vol. 2023-May, IEEE, May 2023, pp. 1–6, ISBN: 978-1-6654-3669-4. DOI: 10.1109/RadarConf2351548.2023.10149793. [Online]. Available: https://ieeexplore.ieee.org/document/10149793/.

[32] D. F. Crouse, "A crash course in basic single-scan target tracking (abridged)," SPIE-Intl Soc Optical Eng, May 2018, p. 25, ISBN: 9781510617773. DOI: 10.1117/12.2307567.

[33] S. Maskell, "A tutorial on particle filters for on-line nonlinear/non-gaussian bayesian tracking," vol. 2001, IEE, 2001, pp. v2–2–v2–2. DOI: 10.1049/ic:20010246. [Online]. Available: https://digital-library.theiet.org/content/conferences/10.1049/ic_20010246.

[34] F. Daum and J. Huang, "Nonlinear filters with particle flow," O. E. Drummond and R. D. Teichgraeber, Eds., vol. 7445, SPIE, Aug. 2009, 74450R, ISBN: 9780819477354. DOI: 10.1117/12.823458. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.823458.

[35] D. F. Crouse and C. Lewis, *Consideration of particle flow filter implementations and biases*, Feb. 2019. [Online]. Available: https://apps.dtic.mil/sti/citations/trecms/AD1091426.

[36] H. P. Blom, "An efficient filter for abruptly changing systems," IEEE, Dec. 1984, pp. 656–658. DOI: 10.1109/CDC.1984.272089. [Online]. Available: http://ieeexplore.ieee.org/document/4047965/.

[37] H. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with markovian switching coefficients," *IEEE Transactions on Automatic Control*, vol. 33, pp. 780–783, 8 1988, ISSN: 00189286. DOI: 10.1109/9.1299. [Online]. Available: http://ieeexplore.ieee.org/document/1299/.

[38] Y. Bar-Shalom, K. C. Chang, and H. A. P. Blom, "Automatic track formation in clutter with a recursive algorithm," in Y. Bar-Shalom, Ed., 1st ed. Artech House, 1990, pp. 25–42.

[39] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, pp. 83–97, 1-2 Mar. 1955, ISSN: 0028-1441. DOI: 10.1002/nav.3800020109. [Online]. Available: https://onlinelibrary.wiley.com/doi/10.1002/nav.3800020109.

[40] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, pp. 32–38, 1 Mar. 1957, ISSN: 0368-4245. DOI: 10.1137/0105003. [Online]. Available: http://epubs.siam.org/doi/10.1137/0105003.

[41] F. Bourgeois and J.-C. Lassalle, "An extension of the munkres algorithm for the assignment problem to rectangular matrices," *Communications of the ACM*, vol. 14, pp. 802–804, 12 Dec. 1971, ISSN: 0001-0782. DOI: 10.1145/362919.362945. [Online]. Available: https://dl.acm.org/doi/10.1145/362919.362945.

[42] D. P. Bertsekas, "A new algorithm for the assignment problem," *Mathematical Programming*, vol. 21, pp. 152–171, 1 Dec. 1981, ISSN: 0025-5610. DOI: 10.1007/BF01584237. [Online]. Available: http://link.springer.com/10.1007/BF01584237.

[43] D. P. Bertsekas, "Auction algorithms for network flow problems: A tutorial introduction," *Computational Optimization and Applications*, vol. 1, pp. 7–66, 1 Oct. 1992, ISSN: 0926-6003. DOI: 10.1007/BF00247653. [Online]. Available: http://link.springer.com/10.1007/BF00247653.

[44] D. Bertsekas, "New auction algorithms for the assignment problem and extensions," *Results in Control and Optimization*, vol. 14, p. 100 383, Mar. 2024, ISSN: 26667207. DOI: 10.1016/j.rico.2024.100383. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2666720724000134.

[45] R. Jonker and T. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," in 1988, vol. 38, pp. 622–622, ISBN: 978-3-540-19365-4. DOI: 10.1007/978-3-642-73778-7_164. [Online]. Available: http://link.springer.com/10.1007/978-3-642-73778-7_164.

[46] D. A. Castanon, "New assignment algorithms for data association," O. E. Drummond, Ed., Aug. 1992, pp. 313–323. DOI: 10.1117/12.139398. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=991288.

[47] D. B. Malkoff, "Evaluation of the jonker-volgenant-castanon (jvc) assignment algorithm for track association," I. Kadar, Ed., Jul. 1997, p. 228. DOI: 10.1117/12.280801. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.280801.

[48] I. Kadar, E. R. Eadan, and R. R. Gassner, "Comparison of robustized assignment algorithms," I. Kadar, Ed., Jul. 1997, p. 240. DOI: 10.1117/12.280802. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.280802.

[49] M. Levedahl, "Performance comparison of 2d assignment algorithms for assigning truth objects to measured tracks," O. E. Drummond, Ed., Jul. 2000, pp. 380–389. DOI: 10.1117/12.392018. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=905777.

[50] D. F. Crouse, "On implementing 2d rectangular assignment algorithms," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, pp. 1679–1696, 4 Aug. 2016, ISSN: 0018-9251. DOI: 10.1109/TAES.2016.140952. [Online]. Available: http://ieeexplore.ieee.org/document/7738348/.

[51] K. G. Murty, "Letter to the editor—an algorithm for ranking all the assignments in order of increasing cost," *Operations Research*, vol. 16, pp. 682–687, 3 Jun. 1968, ISSN: 0030-364X. DOI: 10.1287/opre.16.3.682. [Online]. Available: https://pubsonline.informs.org/doi/10.1287/opre.16.3.682.

[52] Q. Lu, W. Dou, R. Visina, K. Pattipati, Y. Bar-Shalom, and P. Willett, "Evaluation of optimizations of murty's m-best assignment," I. Kadar, Ed., SPIE, Apr. 2018, p. 10, ISBN: 9781510618039. DOI: 10.1117/12.2306486. [Online]. Available: https://www.spiedigitallibrary.org/conference-proceedings-of-spie/10646/2306486/Evaluation-of-optimizations-of-Murtys-M-best-assignment/10.1117/12.2306486.full.

[53] S. Deb, M. Yeddanapudi, K. Pattipati, and Y. Bar-Shalom, "A generalized s-d assignment algorithm for multisensor-multitarget state estimation," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 33, pp. 523–538, 2 Apr. 1997, ISSN: 0018-9251. DOI: 10.1109/7.575891. [Online]. Available: http://ieeexplore.ieee.org/document/575891/.

[54] R. Popp, K. Pattipati, and Y. Bar-Shalom, "M-best s-d assignment algorithm with application to multitarget tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, pp. 22–39, 1 2001, ISSN: 00189251. DOI: 10.1109/7.913665. [Online]. Available: http://ieeexplore.ieee.org/document/913665/.

[55] A. B. Poore, "Multidimensional assignment formulation of data association problems arising from multitarget and multisensor tracking," *Computational Optimization and Applications*, vol. 3, pp. 27–57, 1 Mar. 1994, ISSN: 0926-6003. DOI: 10.1007/BF01299390. [Online]. Available: http://link.springer.com/10.1007/BF01299390.

[56] P. A. Miceli, W. D. Blair, and M. M. Brown, "Isolating random and bias covariances in tracks," IEEE, Jul. 2018, pp. 2437–2444, ISBN: 978-0-9964527-6-2. DOI: 10.23919/ICIF.2018.8455530. [Online]. Available: https://ieeexplore.ieee.org/document/8455530/.

[57] M. Levedahl, "Explicit pattern matching assignment algorithm," O. E. Drummond, Ed., vol. 4728, SPIE, Aug. 2002, pp. 461–469. DOI: 10.1117/12.478526. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=886996.

[58] D. J. Papageorgiou and J.-D. Sergi, "Simultaneous track-to-track association and bias removal using multistart local search," IEEE, Mar. 2008, pp. 1–14, ISBN: 978-1-4244-1487-1. DOI: 10.1109/AERO.2008.4526430. [Online]. Available: http://ieeexplore.ieee.org/document/4526430/.

[59] D. J. Papageorgiou and M. Holender, "Track-to-track association and ambiguity management in the presence of sensor bias," IEEE, Jun. 2009, pp. 2012–2019, ISBN: 978-0-9824-4380-4. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5203719.

[60] D. J. Papageorgiou and M. Holender, "Track-to-track association and ambiguity management in the presence of sensor bias," *Journal of Advances in Information Fusion*, vol. 6, pp. 77–100, 2 Dec. 2011.

[61] R. Kenefic, "Local and remote track file registration using minimum description length," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 29, pp. 651–655, 3 Jul. 1993, ISSN: 0018-9251. DOI: 10.1109/7.220917. [Online]. Available: https://ieeexplore.ieee.org/document/220917/.

[62] M. D. Levedahl, *Method and system for assigning observations*, Aug. 2006.

[63] M. Levedahl and J. D. Glass, "Optimal non-assignment costs for the GNP problem," IEEE, Mar. 2020, pp. 1–7, ISBN: 978-1-7281-2734-7. DOI: 10.1109/AERO47225.2020.9172284. [Online]. Available: https://ieeexplore.ieee.org/document/9172284/.

[64] M. Levedahl and J. D. Glass, "Analysis of costs for the GNP problem," *Journal of Advances in Information Fusion*, vol. 16, pp. 17–30, 1 Jun. 2021. [Online]. Available: https://isif.org/files/isif/2024-01/17-30.pdf.

[65] S. Mori and C. Chong, "Comparison of bias removal algorithms in track-to-track association," O. E. Drummond and R. D. Teichgraeber, Eds., vol. 6699, SPIE, Sep. 2007, 66990T. DOI: 10.1117/12.735383. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=818568.

[66] S. S. Blackman and N. D. Banh, "Track association using correction for bias and missing data," O. E. Drummond, Ed., Jul. 1994, pp. 529–539. DOI: 10.1117/12.179077. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=966497.

[67] D. Dunham, G. Norgard, J. D. Glass, E. Everett, and D. Blair, "Pattern metrics for groups of target tracks," vol. 2018-March, IEEE, Mar. 2018, pp. 1–11, ISBN: 978-1-5386-2014-4. DOI: 10.1109/AERO.2018.8396700. [Online]. Available: https://ieeexplore.ieee.org/document/8396700/.

[68] H. Zhu, W. Wang, and C. Wang, "Robust track-to-track association in the presence of sensor biases and missed detections," *Information Fusion*, vol. 27, pp. 33–40, Jan. 2016, ISSN: 15662535. DOI: 10.1016/j.inffus.2015.05.002. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1566253515000469.

[69] X. Lin, T. Kirubarajan, and Y. Bar-Shalom, "Multisensor bias estimation using local tracks without a priori association," O. E. Drummond, Ed., vol. 5204, SPIE, Dec. 2003, pp. 334–345. DOI: 10.1117/12.503715. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=826722.

[70] M. Levedahl, J. D. Glass, and P. Powers, "A recursive distributed linear bias model," vol. 2023-March, IEEE Computer Society, 2023, ISBN: 9781665490320. DOI: 10.1109/AERO55745.2023.10115869.

[71] A. Das and W. S. Geisler, "A method to integrate and classify normal distributions," *Journal of Vision*, vol. 21, p. 1, 10 Sep. 2021, ISSN: 1534-7362. DOI: 10.1167/jov.21.10.1. [Online]. Available: https://jov.arvojournals.org/article.aspx?articleid=2776752.

[72] A. Das and W. S. Geisler, "Methods to integrate multinormals and compute classification measures," Dec. 2020. [Online]. Available: http://arxiv.org/abs/2012.14331.

[73] D. A. Bodenham and N. M. Adams, "A comparison of efficient approximations for a weighted sum of chi-squared random variables," *Statistics and Computing*, vol. 26, pp. 917–928, 4 Jul. 2016, ISSN: 0960-3174. DOI: 10.1007/s11222-015-9583-4. [Online]. Available: http://link.springer.com/10.1007/s11222-015-9583-4.

[74] X. R. Li, "Engineer's guide to variable-structure multiple-model estimation for tracking," in YBS Publishing, 2000, vol. 3, pp. 499–567. [Online]. Available: https://cites

eerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=ee1365d32d153bf1f60d99d27859180c4f30066e.

[75] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan, "Interacting multiple model methods in target tracking: A survey," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, pp. 103–123, 1 1998, ISSN: 00189251. DOI: 10.1109/7.640267. [Online]. Available: http://ieeexplore.ieee.org/document/640267/.

[76] A. F. Genovese and A. F. Genovese, "The interacting multiple model algorithm for accurate state estimation of maneuvering targets," *JOHNS HOPKINS APL TECHNICAL DIGEST*, vol. 22, pp. 614–623, 4 2001.

[77] M. T. Busch and S. S. Blackman, "Evaluation of IMM filtering for an air defense system application," O. E. Drummond, Ed., SPIE, Sep. 1995, pp. 435–447, ISBN: 0819419206. DOI: 10.1117/12.217717. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=1007070.

[78] R. R. Pitre, V. P. Jilkov, and X. R. Li, "A comparative study of multiple-model algorithms for maneuvering target tracking," I. Kadar, Ed., vol. 5809, SPIE, May 2005, p. 549. DOI: 10.1117/12.609681. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?doi=10.1117/12.609681.

[79] P. Suchomski, "High-order interacting multiple-model estimation for hybrid systems with Markovian switching parameters," *International Journal of Systems Science*, vol. 32, pp. 669–679, 5 Jan. 2001, ISSN: 0020-7721. DOI: 10.1080/00207720118857. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/00207720118857.

[80] C.-Y. Chong, "An overview of machine learning methods for multiple target tracking," IEEE, Nov. 2021, pp. 1–9, ISBN: 978-1-7377497-1-4. DOI: 10.23919/FUSION49465.2021.9627045. [Online]. Available: https://ieeexplore.ieee.org/document/9627045/.

[81] A. Wrabel, R. Graef, and T. Brosch, "A survey of artificial intelligence approaches for target surveillance with radar sensors," *IEEE Aerospace and Electronic Systems*

Magazine, vol. 36, pp. 26–43, 7 Jul. 2021, ISSN: 0885-8985. DOI: 10.1109/MAES.2021.3065069. [Online]. Available: https://ieeexplore.ieee.org/document/9475914/.

[82] L. Deng, D. Li, and R. Li, "Improved IMM algorithm based on RNNs," *Journal of Physics: Conference Series*, vol. 1518, p. 012 055, 1 Apr. 2020, ISSN: 1742-6588. DOI: 10.1088/1742-6596/1518/1/012055. [Online]. Available: https://iopscience.iop.org/article/10.1088/1742-6596/1518/1/012055.

[83] J. Rong, X. Wang, and H. Zhang, "A hybrid neural network-based IE and IMM architecture for target tracking," IEEE, Aug. 2008, pp. 214–217, ISBN: 978-0-7695-3342-1. DOI: 10.1109/PEITS.2008.28. [Online]. Available: https://ieeexplore.ieee.org/document/4634846/.

[84] M. Sun, Z. Ma, and Y. Li, "Maneuvering target tracking using IMM Kalman filter aided by elman neural network," vol. 1, IEEE, Aug. 2015, pp. 144–148, ISBN: 978-1-4799-8645-3. DOI: 10.1109/IHMSC.2015.241. [Online]. Available: http://ieeexplore.ieee.org/document/7334671/.

[85] M. Owen and A. Stubberud, "A neural extended Kalman filter multiple model tracker," IEEE, 2003, 2111–2119 Vol.4, ISBN: 0-933957-30-0. DOI: 10.1109/OCEANS.2003.178229. [Online]. Available: http://ieeexplore.ieee.org/document/1282797/.

[86] N. Forti, L. M. Millefiori, P. Braca, and P. Willett, "Model-based deep learning for maneuvering target tracking," IEEE, Jun. 2023, pp. 1–6, ISBN: 979-8-89034-485-4. DOI: 10.23919/FUSION52260.2023.10224081. [Online]. Available: https://ieeexplore.ieee.org/document/10224081/.

[87] M. Gariel, B. Shimanuki, R. Timpe, and E. Wilson, "Framework for certification of ai-based systems," Feb. 2023. [Online]. Available: http://arxiv.org/abs/2302.11049.

[88] E. Tabassi, *Artificial intelligence risk management framework (ai rmf 1.0)*, Jan. 2023. DOI: 10.6028/NIST.AI.100-1. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/ai/NIST.AI.100-1.pdf.

[89] J. R. Norris, *Markov chains*. Cambridge university press, 1998.

[90] L. Bloomer and J. Gray, "Are more models better?: The effect of the model transition matrix on the IMM filter," IEEE, 2002, pp. 20–25, ISBN: 0-7803-7339-1. DOI: 10.1109/SSST.2002.1026997. [Online]. Available: http://ieeexplore.ieee.org/document/1026997/.

[91] O. E. Drummond, "Methodologies for performance evaluation of multitarget multisensor tracking," O. E. Drummond, Ed., vol. 3809, SPIE, Oct. 1999, pp. 355–369. DOI: 10.1117/12.364034. [Online]. Available: http://proceedings.spiedigitallibrary.org/proceeding.aspx?articleid=905488.

[92] S. J. Julier and J. K. Uhlmann, "A non-divergent estimation algorithm in the presence of unknown correlations," in *Proceedings of the 1997 American Control Conference (Cat. No. 97CH36041)*, IEEE, vol. 4, 1997, pp. 2369–2373.

[93] M. Reinhardt, B. Noack, and U. D. Hanebeck, "Closed-form optimization of covariance intersection for low-dimensional matrices," in *2012 15th International Conference on Information Fusion*, IEEE, 2012, pp. 1891–1896.

[94] O. Bochardt, R. Calhoun, J. K. Uhlmann, and S. J. Julier, "Generalized information representation and compression using covariance union," in *2006 9th International Conference on Information Fusion*, IEEE, 2006, pp. 1–7.

[95] J. Nygårds, V. Deleskog, and G. Hendeby, "Safe fusion compared to established distributed fusion methods," in *2016 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, 2016, pp. 265–271. DOI: 10.1109/MFI.2016.7849499.

[96] A. Benaskeur, "Consistent fusion of correlated data sources," in *IEEE 2002 28th Annual Conference of the Industrial Electronics Society. IECON 02*, vol. 4, 2002, 2652–2656 vol.4. DOI: 10.1109/IECON.2002.1182812.

[97] M. Hurley, "An information theoretic justification for covariance intersection and its generalization," in *Proceedings of the Fifth International Conference on Information Fusion. FUSION 2002. (IEEE Cat.No.02EX5997)*, vol. 1, Int. Soc. Inf. Fusion, 2002, pp. 505–511, ISBN: 0-9721844-1-4. DOI: 10.1109/ICIF.2002.1021196. [Online]. Available: http://ieeexplore.ieee.org/document/1021196/.

[98] J Sijs, M Lazar, and P. P. J. v.d. Bosch, "State fusion with unknown correlation: Ellipsoidal intersection," in *Proceedings of the 2010 American Control Conference*, IEEE, Jun. 2010, pp. 3992–3997, ISBN: 978-1-4244-7427-1. DOI: 10.1109/ACC.2010.5531237.

[99] W. J. Farrell and C. Ganesh, "Generalized chernoff fusion approximation for practical distributed data fusion," in *2009 12th International Conference on Information Fusion*, 2009, pp. 555–562, ISBN: 978-0-9824-4380-4.

[100] D. F. Crouse, "Cubature/unscented/sigma point kalman filtering with angular measurement models," in *2015 18th International Conference on Information Fusion*, Institute of Electrical and Electronics Engineers, Jul. 2015, pp. 1550–1557, ISBN: 978-0-9824-4386-6.

[101] J. Pruher, T. Karvonen, C. J. Oates, O. Straka, and S. Sarkka, "Improved calibration of numerical integration error in sigma-point filters," *IEEE Transactions on Automatic Control*, vol. 66, pp. 1286–1292, 3 Mar. 2021, ISSN: 0018-9286. DOI: 10.1109/TAC.2020.2991698. [Online]. Available: https://ieeexplore.ieee.org/document/9084238/.

[102] J. Spanier and E. H. Maize, "Quasi-random methods for estimating integrals using relatively small samples," *SIAM Review*, vol. 36, pp. 18–44, 1 Mar. 1994, ISSN: 0036-1445. DOI: 10.1137/1036002. [Online]. Available: http://epubs.siam.org/doi/10.1137/1036002.

[103] H. Niederreiter, "Quasi-monte carlo methods and pseudo-random numbers," *Bulletin of the American mathematical society*, vol. 84, pp. 957–1041, 6 1978.

[104] H. Niederreiter, "Low-discrepancy and low-dispersion sequences," *Journal of Number Theory*, vol. 30, pp. 51–70, 1 Sep. 1988, ISSN: 0022314X. DOI: 10.1016/0022-314X(88)90025-X.

[105] S. K. Zaremba, "Good lattice points, discrepancy, and numerical integration," *Annali di Matematica Pura ed Applicata*, vol. 73, pp. 293–317, 1 Dec. 1966, ISSN: 0373-3114. DOI: 10.1007/BF02415091. [Online]. Available: http://link.springer.com/10.1007/BF02415091.

[106] I. H. Sloan and P. J. Kachoyan, "Lattice methods for multiple integration: Theory, error analysis and examples," *SIAM Journal on Numerical Analysis*, vol. 24, pp. 116–128, 1 Feb. 1987, ISSN: 0036-1429. DOI: 10.1137/0724010. [Online]. Available: http://epubs.siam.org/doi/10.1137/0724010.

[107] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*. SIAM, 1992.

[108] J. H. Halton, "On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals," *Numerische Mathematik*, vol. 2, pp. 84–90, 1 Dec. 1960, ISSN: 0029-599X. DOI: 10.1007/BF01386213. [Online]. Available: http://link.springer.com/10.1007/BF01386213.

[109] J. H. Halton, "Algorithm 247: Radical-inverse quasi-random point sequence," *Communications of the ACM*, vol. 7, pp. 701–702, 12 Dec. 1964, ISSN: 0001-0782. DOI: 10.1145/355588.365104. [Online]. Available: https://dl.acm.org/doi/10.1145/355588.365104.

[110] H. Faure and C. Lemieux, "Generalized halton sequences in 2008," *ACM Transactions on Modeling and Computer Simulation*, vol. 19, pp. 1–31, 4 Oct. 2009, ISSN: 1049-

3301. DOI: 10.1145/1596519.1596520. [Online]. Available: https://dl.acm.org/doi/10.1145/1596519.1596520.

[111] L. Kocis and W. J. Whiten, "Computational investigations of low-discrepancy sequences," *ACM Transactions on Mathematical Software*, vol. 23, pp. 266–294, 2 Jun. 1997, ISSN: 0098-3500. DOI: 10.1145/264029.264064. [Online]. Available: https://dl.acm.org/doi/10.1145/264029.264064.

[112] P. Bratley and B. L. Fox, "Algorithm 659," *ACM Transactions on Mathematical Software*, vol. 14, pp. 88–100, 1 Mar. 1988, ISSN: 0098-3500. DOI: 10.1145/42288.214372. [Online]. Available: https://dl.acm.org/doi/10.1145/42288.214372.

[113] S. Joe and F. Y. Kuo, "Remark on algorithm 659," *ACM Transactions on Mathematical Software*, vol. 29, pp. 49–57, 1 Mar. 2003, ISSN: 0098-3500. DOI: 10.1145/641876.641879. [Online]. Available: https://dl.acm.org/doi/10.1145/641876.641879.

[114] H. Niederreiter and S. I. H., "Integration of nonperiodic functions of two variables by fibonacci lattice rules," *Journal of Computational and Applied Mathematics*, vol. 51, pp. 57–70, 1 May 1994, ISSN: 03770427. DOI: 10.1016/0377-0427(92)00004-S. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/037704279200004S.

[115] R. Swinbank and R. J. Purser, "Fibonacci grids: A novel approach to global modelling," *Quarterly Journal of the Royal Meteorological Society*, vol. 132, pp. 1769–1793, 619 Jul. 2006, ISSN: 00359009. DOI: 10.1256/qj.05.227. [Online]. Available: http://doi.wiley.com/10.1256/qj.05.227.

[116] D. Frisch and U. D. Hanebeck, "Deterministic gaussian sampling with generalized fibonacci grids," in *2021 IEEE 24th International Conference on Information Fusion (FUSION)*, IEEE, Nov. 2021, pp. 1–8, ISBN: 978-1-7377497-1-4. DOI: 10.23919/FUSION49465.2021.9626975. [Online]. Available: https://ieeexplore.ieee.org/document/9626975/.

[117] D. Frisch and U. D. Hanebeck, "The generalized fibonacci grid as low-discrepancy point set for optimal deterministic gaussian sampling," *Journal of Advances in Information Fusion*, vol. 18, pp. 16–34, 1 2023, ISSN: 1557-6418.

[118] J. A. Clarkson and C. R. Adams, "On definitions of bounded variation for functions of two variables," *Transactions of the American Mathematical Society*, vol. 35, p. 824, 4 Oct. 1933, ISSN: 00029947. DOI: 10.2307/1989593. [Online]. Available: https://www.jstor.org/stable/1989593?origin=crossref.

[119] K. Basu and A. B. Owen, "Transformations and hardy–krause variation," *SIAM Journal on Numerical Analysis*, vol. 54, pp. 1946–1966, 3 Jan. 2016, ISSN: 0036-1429. DOI: 10.1137/15M1052184. [Online]. Available: http://epubs.siam.org/doi/10.1137/15M1052184.

[120] L Kuipers and H Niederreiter, *Uniform Distribution of Sequences*. Dover Publications, 2012, ISBN: 9780486149998. [Online]. Available: https://books.google.com/books?id=mnY8LpyXHM0C.

[121] H. Niederreiter, "Error bounds for quasi-monte carlo integration with uniform point sets," *Journal of Computational and Applied Mathematics*, vol. 150, pp. 283–292, 2 Jan. 2003, ISSN: 03770427. DOI: 10.1016/S0377-0427(02)00665-9.

[122] V. Temlyakov and M. Ullrich, "On the fixed volume discrepancy of the fibonacci sets in the integral norms," *Journal of Complexity*, vol. 61, p. 101 472, Dec. 2020, ISSN: 0885064X. DOI: 10.1016/j.jco.2020.101472. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0885064X20300157.

[123] T. K. Rusch, N. Kirk, M. M. Bronstein, C. Lemieux, and D. Rus, "Message-passing monte carlo: Generating low-discrepancy point sets via graph neural networks," May 2024.

[124] M. Patel, "Optimizing kronecker sequences for multidimensional sampling," *Journal of Computer Graphics Techniques (JCGT)*, vol. 11, pp. 55–81, 1 Mar. 2022, ISSN: 2331-7418. [Online]. Available: http://jcgt.org/published/0011/01/04/.

[125] F. B. da Silva, R. von Borries, and C. J. Miosso, "Golden number sampling applied to compressive sensing," in *2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI)*, IEEE, Apr. 2018, pp. 1–4, ISBN: 978-1-5386-6568-8. DOI: 10.1109/SSIAI.2018.8470345. [Online]. Available: https://ieeexplore.ieee.org/document/8470345/.

[126] N. Moshchevitin, "Distribution of kronecker sequences," in F Halter-Koch and R. F. Tichy, Eds. DE GRUYTER, 2000, pp. 311–330, ISBN: 9783110801958. DOI: 10.1515/9783110801958.311. [Online]. Available: https://www.degruyter.com/document/doi/10.1515/9783110801958.311/html.

[127] R. Marques, C. Bouville, K. Bouatouch, and J. Blat, "Extensible spherical fibonacci grids," *IEEE Transactions on Visualization and Computer Graphics*, vol. 27, pp. 2341–2354, 4 Apr. 2021, ISSN: 1077-2626. DOI: 10.1109/TVCG.2019.2952131. [Online]. Available: https://ieeexplore.ieee.org/document/8894500/.

[128] C. Schretter, L. Kobbelt, and P.-O. Dehaye, "Golden ratio sequences for low-discrepancy sampling," *Journal of Graphics Tools*, vol. 16, pp. 95–104, 2 Jun. 2012, ISSN: 2165-347X. DOI: 10.1080/2165347X.2012.679555.

[129] B. Gandar, G. Loosli, and G. Deffuant, *Sample dispersion is better than sample discrepancy for classification*, Oct. 2010. [Online]. Available: https://hal.science/hal-00679061.

[130] S. Breneis and A. Hinrichs, "Fibonacci lattices have minimal dispersion on the two-dimensional torus," *Discrepancy Theory*, pp. 117–132, May 2019. [Online]. Available: http://arxiv.org/abs/1905.03856.

[131] V. Temlyakov, "Smooth fixed volume discrepancy, dispersion, and related problems," *Journal of Approximation Theory*, vol. 237, pp. 113–134, Jan. 2019, ISSN: 00219045. DOI: 10.1016/j.jat.2018.09.002. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0021904518301035.

[132] C Aistleitner, J. S. Brauchart, and J Dick, "Point sets on the sphere $\{S\}^{\{2\}}$with small spherical cap discrepancy," *Discrete & Computational Geometry*, vol. 48, pp. 990–1024, 4 2012, ISSN: 1432-0444. DOI: 10.1007/s00454-012-9451-3. [Online]. Available: https://doi.org/10.1007/s00454-012-9451-3.

[133] D. Kalman and R. Mena, "The fibonacci numbers—exposed," *Mathematics Magazine*, vol. 76, pp. 167–181, 3 Jun. 2003, ISSN: 0025-570X. DOI: 10.1080/0025570X.2003.11953176. [Online]. Available: https://www.tandfonline.com/doi/full/10.1080/0025570X.2003.11953176.

[134] I. H. Sloan and S Joe, *Lattice Methods for Multiple Integration*. Clarendon Press, 1994, ISBN: 9780198534723. [Online]. Available: https://books.google.com/books?id=9twIWULueasC.

[135] P. Schreiber, "A supplement to j. shallit's paper "origins of the analysis of the euclidean algorithm"," *Historia Mathematica*, vol. 22, pp. 422–424, 4 Nov. 1995, ISSN: 03150860. DOI: 10.1006/hmat.1995.1033. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0315086085710336.

[136] E. Kilic, "The binet formula, sums and representations of generalized fibonacci," *European Journal of Combinatorics*, vol. 29, pp. 701–711, 3 Apr. 2008, ISSN: 01956698. DOI: 10.1016/j.ejc.2007.03.004. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0195669807000595.

[137] D. Frisch and U. D. Hanebeck, "Deterministic von mises–fisher sampling on the sphere using fibonacci lattices," in *2023 IEEE Symposium Sensor Data Fusion and International Conference on Multisensor Fusion and Integration (SDF-MFI)*, IEEE, Nov.

2023, pp. 1–8, ISBN: 979-8-3503-8258-7. DOI: 10.1109/SDF-MFI59545.2023.10361396. [Online]. Available: https://ieeexplore.ieee.org/document/10361396/.

[138] R. J. Purser, *Generalized fibonacci grids; a new class of structured, smoothly adaptive multi-dimensional computational lattices*, May 2008. [Online]. Available: https://repository.library.noaa.gov/view/noaa/6956.