Evaluating the Bayesian Cramér-Rao Bound for multiple model filtering

Lennart Svensson
Dept. of Signals and Systems
Chalmers Univ. of Technology
Göteborg, Sweden
lennart.svensson@chalmers.se

Abstract – *We propose a numerical algorithm to evaluate the Bayesian Cramér-Rao Bound (BCRB) for multiple model filtering problems. It is assumed that the individual models have additive Gaussian noise and that the measurement model is linear. The algorithm is also given in a recursive form, making it applicable for sequences of arbitrary length. Previous attempts to calculate the BCRB for multiple model filtering problems are based on rough approximations which usually make them simple to calculate. In this paper, we propose an algorithm which is based on Monte Carlo sampling, and which is hence more computationally demanding, but yields accurate approximations of the BCRB. An important observation from the simulations is that the BCRB is more overoptimistic than previously suggested bounds, which we motivate using theoretical results.*

Keywords: Performance bounds, multiple model filtering, non-linear filtering, IMM, PCRLB, BCRB, BFG

1 Introduction

The Bayesian Cramér-Rao Bound (BCRB)[1] on estimation performance, initially proposed by Van Trees [1, 2], has been used extensively in many areas. For non-linear filtering problems, a major breakthrough was presented in [3], in which a recursive formula to calculate the bound was developed. Since then, techniques to evaluate the bound also for prediction, smoothing [4, 5] and tracking in clutter [6, 7] have also been suggested. These results have been used, e.g., to design and manage different sensor systems [8, 9].

An important area for which the BCRB has not yet been calculated is multiple model filtering, i.e., for Markovian switching systems. In many applications, multiple models are used to describe motions that change character at different points in time. A well known example is the motion of airborne targets, which is often described as a jump Markov process that can switch between, e.g., a maneuvering and a non-maneuvering state. In such models, a discrete valued model index is introduced in addition to the original continuous valued state vector. Among the many techniques to filter such problems, the Interacting Multiple Models (IMM) algorithm [10] has been the most popular choice for many years.

There are essentially two, strongly related, reasons why it is more difficult to calculate the BCRB for multiple model filtering than for nonlinear single model filtering. First, as the BCRB involves derivatives with respect to the vectors of interest, it can not handle discrete valued parameters. As a consequence, the model index has to be marginalized from all densities. Second, unless the discrete valued model index is included in the state vector, the state process is not Markovian. Of fundamental importance for the BCRB is the Fisher information matrix, see [2]. In [3], it was shown that the Markov property yields a block diagonal structure of the information matrix that enables a recursive calculation of the BCRB. For multiple model filtering, the absence of this structure makes it more difficult to obtain a recursive algorithm.

Two articles that target the BCRB for multiple model filtering are [11] and [12]. Conditioned on a model sequence, the BCRB can be found using the results from [3]. To obtain a lower bound for the unconditional problem, the idea in [11] is to calculate the expected value of the conditional bound when the sequence of models is instead considered stochastic. The result is a lower bound, here referred to as the Enumer-BCRB, which is overoptimistic and sometimes predicts far from attainable performance. To overcome weaknesses in [11], Hernandez et al [12] performed a study limited to linear and Markovian switching systems. To produce a performance measure they approximated the original system with a linear and Gaussian model for which the BCRB has an analytical solution [3]. The derivations yield a performance approximation which is easy to compute and that often corresponds reasonably well to existing filters. Though the algorithm does not produce a lower bound, it has many useful properties. A potential weakness, however, is that the algorithm is not designed to handle systems that switch between nonlinear models, even though

[1]In some articles, the bound is also referred to as the Posterior Cramér-Rao Lower Bound (PCRLB).

possible remedies to this problem are mentioned in [12].

We propose a numerical evaluation of the Fisher information matrix for the complete trajectory of state vectors, using a Monte Carlo method. The algorithm is straightforward to implement and implicitly marginalizes the model index. The BCRB for the current state vector is obtained by inverting the Fisher information matrix. However, as the dimensions of the information matrix increases over time, the algorithm eventually becomes impractical. To maintain a constant complexity over time, we develop a recursive algorithm to compute the BCRB. The algorithm is based on an assumption that the dependence between state vectors only persists for a limited time. Following this assumption, the information matrix takes a form for which it is possible to derive a recursive method.

Initial studies reveals the surprising result that the BCRB is even more overoptimistic than the Enumer-BCRB [11]. This important result, which shows the inadequacy of the BCRB, is further explained in Section 6. We also notice that the BCRB is very computationally demanding to evaluate accurately. Still, even for fairly short sequences it is faster to calculate the BCRB than to compute the average performance of the optimal estimator, and this difference increases rapidly with the length of the sequence. Moreover, it is straightforward to combine the proposed technique to evaluate the BCRB with previous methods to handle uncertainties in measurement origin [6, 7]. For such problems, the difference in complexity between the optimal estimator and the BCRB would be even more significant. However, in the light of the simulation results, it is difficult to see why one should use BCRB instead of, e.g., the Enumer-BCRB.

2 Model assumptions

Let $\mathbf{x}_k$ denote a target state vector of length r and let $\mathbf{y}_k$ be the observation vector at time k. In this article, we consider multiple model filtering,

$$\mathbf{x}_k = f_k(\mathbf{x}_{k-1}, m_k) + \mathbf{v}_k \tag{1}$$

$$\mathbf{y}_k = \mathbf{G}_k \mathbf{x}_k + \mathbf{w}_k \tag{2}$$

where m_k is a finite state Markov chain taking values in $\{1, \dots, M\}$, with known transition probabilities $\Pr\{m_k | m_{k-1}\}$. The noise vectors are mutually independent white processes distributed as $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k,m_k})$ and $\mathbf{w}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{R}_k)$ where $\mathbf{Q}_{k,m_k}$ is a nonsingular covariance matrix. As indicated by the notation, $\mathbf{Q}_{k,m_k}$ depends on m_k and can also vary over time. At $k = 0$, we are given prior information in terms of the probability density function $p(\mathbf{x}_0)$ and the probability mass function $\Pr\{m_0\}$. For notation, let us introduce $\mathbf{X}_{t_1:t_2} = [\mathbf{x}_{t_1}^T, \mathbf{x}_{t_1+1}^T, \dots, \mathbf{x}_{t_2}^T]^T$ and $\mathbf{Y}_{t_1:t_2} = [\mathbf{y}_{t_1}^T, \mathbf{y}_{t_1+1}^T, \dots, \mathbf{y}_{t_2}^T]^T$. Note that $\mathbf{x}_k$, $\mathbf{y}_k$, $\mathbf{X}_{t_1:t_2}$ and $\mathbf{Y}_{t_1:t_2}$ are all column vectors.

The noise processes are hence assumed both additive and Gaussian distributed and the measurement equation is assumed to be linear. The algorithms presented in Section 4 can, in fact, be extended to a wider family of models (e.g., to also include nonlinear measurement models). However, by limiting our study to the above family of models we enable a more elegant presentation of the derivations. We also believe that the studied models are sufficiently general to be of great practical and theoretical interest.

3 Background on BCRB

Our objective is to calculate the BCRB for estimating $\mathbf{x}_k$ given $\mathbf{Y}_{1:k} = \left[\mathbf{y}_1^T, \dots, \mathbf{y}_k^T\right]^T$. Let us introduce the operators

$$\begin{aligned}\nabla_{\mathbf{u}} &= \left[\frac{\partial}{\partial u_1}, \dots, \frac{\partial}{\partial u_n}\right]^T \\ \triangle_{\mathbf{u}}^{\mathbf{t}} &= \nabla_{\mathbf{u}} \nabla_{\mathbf{t}}^T\end{aligned}$$

for any vectors $\mathbf{u}$ and $\mathbf{t}$. Using this notation, the Fisher information matrix for $\mathbf{X}_{1:k}$ is defined as

$$\mathbf{J}^k = E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k})\right]. \tag{3}$$

The information matrix, $\mathbf{J}_k$, for the parameter of interest, $\mathbf{x}_k$, is the inverse of the $(r \times r)$ lower-right submatrix of $\left[\mathbf{J}^k\right]^{-1}$. To obtain the desired BCRB for $\mathbf{x}_k$ we calculate the inverse of $\mathbf{J}_k$. The BCRB is a bound on the estimation performance, stating that

$$E\left\{(\mathbf{x}_k - \delta(\mathbf{Y}_{1:k}))(\mathbf{x}_k - \delta(\mathbf{Y}_{1:k}))^T\right\} \geq \mathbf{J}_k^{-1} \tag{4}$$

for all estimators $\delta(\mathbf{Y}_{1:k})$. For matrices $\mathbf{A}$ and $\mathbf{B}$, the relation $\mathbf{A} \geq \mathbf{B}$ holds if $\mathbf{A} - \mathbf{B}$ is a positive semidefinite matrix. The BCRB holds with equality, i.e., there exists an estimator $\delta(\mathbf{Y}_{1:k})$ such that $E\left\{(\mathbf{x}_k - \delta(\mathbf{Y}_{1:k}))(\mathbf{x}_k - \delta(\mathbf{Y}_{1:k}))^T\right\} = \mathbf{J}_k^{-1}$, only when the posterior density $p(\mathbf{x}_k | \mathbf{Y}_{1:k})$ is Gaussian, see [1]. For multiple model filtering, the bound is therefore not identical to the optimal performance, but may still be close depending on the model properties.

It is often useful to separate the information matrix into the contribution from a priori information and the contribution from data

$$\mathbf{J}^k = \mathbf{J}_{\mathbf{X}_{1:k}} + \mathbf{J}_{\mathbf{Y}_{1:k}} \tag{5}$$

where

$$\mathbf{J}_{\mathbf{X}_{1:k}} = E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k})\right] \tag{6}$$

$$\mathbf{J}_{\mathbf{Y}_{1:k}} = E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{Y}_{1:k} | \mathbf{X}_{1:k})\right]. \tag{7}$$

For the considered measurement models, the information from data can be neatly expressed using the relations

$$\begin{aligned}\mathbf{J}_{\mathbf{Y}_{1:k}} &= \sum_{n=1}^{k} E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{y}_n | \mathbf{x}_n)\right] && (8) \\ &= \sum_{n=1}^{k} E\left[-\mathbf{I}_k \otimes \triangle_{\mathbf{x}_n}^{\mathbf{x}_n} \log p(\mathbf{y}_n | \mathbf{x}_n)\right] && (9) \\ &= \sum_{n=1}^{k} \mathbf{I}_k \otimes \mathbf{G}_n^T \mathbf{R}_n^{-1} \mathbf{G}_n. && (10)\end{aligned}$$

Here, $\mathbf{I}_k$ denotes a $(k \times k)$ dimensional identity matrix whereas $\otimes$ is the Kronecker product.

The Fisher information is also commonly expressed in terms of gradients

$$\mathbf{J}_{\mathbf{X}_{1:k}} = E\left[\nabla_{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k})\left(\nabla_{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k})\right)^T\right], \tag{11}$$

and sometimes also written on the form

$$\mathbf{J}_{\mathbf{X}_{1:k}} = E\left[\frac{\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k})\left(\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k})\right)^T}{p(\mathbf{X}_{1:k})^2}\right], \tag{12}$$

which we will make use of below.

4 Calculating the BCRB

This section contains our main contributions. If we can calculate $\mathbf{J}_{\mathbf{X}_{1:k}}$, we can obtain the BCRB of $\mathbf{x}_k$ from

$$\left[\mathbf{J}^k\right]^{-1} = \left[\mathbf{J}_{\mathbf{X}_{1:k}} + \mathbf{J}_{\mathbf{Y}_{1:k}}\right]^{-1}. \tag{13}$$

In Section 4.1, we therefore present a numerical algorithm to approximate $\mathbf{J}_{\mathbf{X}_{1:k}}$. Still, the dimension of $\mathbf{J}^k$ grows with time and $\mathbf{J}^k$ eventually becomes impractical to handle. It is thus desirable with a recursive technique to compute $\mathbf{J}_k$. In Section 4.2, we propose one such recursive method, which uses a modified version of the algorithms from Section 4.1.

4.1 The full Fisher information matrix

We focus here on methods to calculate the Fisher information matrix $\mathbf{J}_{\mathbf{X}_{1:k}}$. For settings with multiple motion models it is difficult to find analytical expressions for $\mathbf{J}_{\mathbf{X}_{1:k}}$. In fact, even for basic scalar examples like

$$J_{x_1} = E\left\{\frac{d^2}{dx_1^2} \log p(x_1)\right\}, \tag{14}$$

with $p(x_1) = \alpha\mathcal{N}(x_1; \hat{x}^1, \sigma^2) + (1-\alpha)\mathcal{N}(x_1; \hat{x}^2, \sigma^2)$ and $0 \leq \alpha \leq 1$, the solution is difficult to express analytically. In what follows, we hence resort to numerical approximations.

Using a Monte Carlo technique, the expected value in (12) can be approximated as

$$\mathbf{J}_{\mathbf{X}_{1:k}} \approx \frac{1}{N}\sum_{n=1}^{N} \frac{\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)})(\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)}))^T}{p(\mathbf{X}_{1:k}^{(n)})^2}, \tag{15}$$

where $\mathbf{X}_{1:k}^{(n)}$, $n = 1, \ldots, N$, are independent and identically distributed vectors such that $\mathbf{X}_{1:k}^{(n)} \sim p(\mathbf{X}_{1:k})$. To proceed, we need to develop an algorithm to generate $\mathbf{X}_{1:k}^{(n)}$, $n = 1, \ldots, N$ and to evaluate

$$\frac{-\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)})(\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)}))^T}{p(\mathbf{X}_{1:k}^{(n)})^2}. \tag{16}$$

To compute $\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)})$ and $p(\mathbf{X}_{1:k}^{(n)})$ recursively, we calculate $\Pr\{m_k\}$, $p(\mathbf{X}_{1:k}^{(n)} | m_k)$ and $\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)} | m_k)$ at each time k, for all values of m_k and n.

What follows is an algorithm to do this. At time $k = 0$, we initialize the algorithm by generating $m_0^{(n)} \sim \Pr\{m_0\}$ and $\mathbf{x}_0^{(n)} \sim p(\mathbf{x}_0)$ and defining $\mathbf{X}_{1:0}^{(n)} = \emptyset$ for $n = 1, \ldots, N$.

For $k = 1, 2, \ldots$, and $n = 1, 2, \ldots, N$ do

i) Generate $m_k^{(n)} \sim \Pr\{m_k | m_{k-1}^{(n)}\}$ and $\mathbf{x}_k^{(n)} \sim p(\mathbf{x}_k | \mathbf{x}_{k-1}^{(n)}, m_k^{(n)})$. Also, set $\mathbf{X}_{1:k}^{(n)} = \left[\mathbf{X}_{1:k-1}^{(n)},\ \mathbf{x}_k^{(n)}\right]$.

ii) Update the stored quantities, $\Pr\{m_{k-1}\}$, $p(\mathbf{X}_{1:k-1}^{(n)} | m_{k-1})$ and $\nabla_{\mathbf{X}_{1:k-1}} p(\mathbf{X}_{1:k-1}^{(n)} | m_{k-1})$, using the relations

$$\Pr\{m_k\} = \sum_{m_{k-1}=1}^{M} \Pr\{m_k | m_{k-1}\} \Pr\{m_{k-1}\}, \tag{17}$$

$$p(\mathbf{X}_{1:k}^{(n)} | m_k) = \\ = \sum_{m_{k-1}=1}^{M} p(\mathbf{X}_{1:k}^{(n)} | m_k, m_{k-1}) \Pr\{m_{k-1} | m_k\} \tag{18}$$

where $p(\mathbf{X}_{1:k}^{(n)} | m_k, m_{k-1}) =$

$$= p(\mathbf{x}_k^{(n)} | \mathbf{x}_{k-1}^{(n)}, m_k) p(\mathbf{X}_{1:k-1}^{(n)} | m_{k-1}) \tag{19}$$

$$\Pr\{m_{k-1} | m_k\} = \frac{\Pr\{m_k | m_{k-1}\} \Pr\{m_{k-1}\}}{\Pr\{m_k\}}, \tag{20}$$

$$\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)} | m_k) = \sum_{m_{k-1}=1}^{M} \Pr\{m_{k-1} | m_k\} \\ \Big\{p(\mathbf{x}_k^{(n)} | \mathbf{x}_{k-1}^{(n)}, m_k) \nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k-1}^{(n)} | m_{k-1}) \\ + \left(\nabla_{\mathbf{X}_{1:k}} p(\mathbf{x}_k^{(n)} | \mathbf{x}_{k-1}^{(n)}, m_k)\right) p(\mathbf{X}_{1:k-1}^{(n)} | m_{k-1})\Big\}. \tag{21}$$

iii) Now, the required quantities in (15) are calculated as

$$\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)}) = \sum_{m_k=1}^{M} \nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)} | m_k) \Pr\{m_k\} \tag{22}$$

$$p(\mathbf{X}_{1:k}^{(n)}) = \sum_{m_k=1}^{M} p(\mathbf{X}_{1:k}^{(n)} | m_k) \Pr\{m_k\}. \tag{23}$$

Based on these three steps we can evaluate $\mathbf{J}_{\mathbf{X}_{1:k}}$ and hence $\left[\mathbf{J}^k\right]^{-1}$, which is essentially sufficient in order to calculate the desired bound.

The quantities in the above algorithm are defined by the models and most of them are straightforward to evaluate. The only exception is the gradient $\nabla_{\mathbf{X}_{1:k}} p(\mathbf{x}_k^{(n)} | \mathbf{x}_{k-1}^{(n)}, m_k)$ that has to be derived[2]. The following lemma presents the results that we need in order to express this gradient.

[2] As $\nabla_{\mathbf{x}_k} p(\mathbf{X}_{1:k-1}^{(n)} | m_{k-1}) = \mathbf{0}$, the gradient $\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k-1}^{(n)} | m_{k-1})$ is obtained from $\nabla_{\mathbf{X}_{1:k-1}} p(\mathbf{X}_{1:k-1}^{(n)} | m_{k-1})$, computed in the previous recursion.

Lemma 4.1 *Suppose $\mathbf{x}_k$ evolves according to* (1)*. For the gradients of interest it holds that*

$$\begin{aligned}\nabla_{\mathbf{x}_{k-1}} p(\mathbf{x}_k \big| \mathbf{x}_{k-1}, m_k) =& p(\mathbf{x}_k \big| \mathbf{x}_{k-1}, m_k) \\ \big(\nabla_{\mathbf{x}_{k-1}} f_k&(\mathbf{x}_{k-1}, m_k)^T\big) \mathbf{Q}_{k,m_k}^{-1} (\mathbf{x}_k - f_k(\mathbf{x}_{k-1}, m_k))\end{aligned} \tag{24}$$

$$\begin{aligned}\nabla_{\mathbf{x}_k} p(\mathbf{x}_k \big| \mathbf{x}_{k-1}, m_k) =& - p(\mathbf{x}_k \big| \mathbf{x}_{k-1}, m_k) \\ & \mathbf{Q}_{k,m_k}^{-1} (\mathbf{x}_k - f_k(\mathbf{x}_{k-1}, m_k)).\end{aligned} \tag{25}$$

For $j \notin \{k-1, k\}$ we have $\nabla_{\mathbf{x}_j} p(\mathbf{x}_k \big| \mathbf{x}_{k-1}, m_k) = \mathbf{0}$.

Proof of Lemma 4.1 *See Appendix A.*

Due to Lemma 4.1 we have all the components that make up the gradient $\nabla_{\mathbf{X}_{1:k}} p(\mathbf{x}_k^{(n)} \big| \mathbf{x}_{k-1}^{(n)}, m_k)$.

4.2 Recursive calculation of the BCRB

In scenarios when the models are not time varying, the BCRB will converge to a stationary value (a matrix). For such models, the above algorithm may be sufficient as we are often satisfied with the ability to calculate $\mathbf{J}_k$ for relatively small values of k. Consequently, it is not a problem to store, or invert, the matrix $\mathbf{J}^k$ even though its dimension grows with k. In non-stationary settings, on the other hand, it may be essential to evaluate $\mathbf{J}_k$ for large values of k. To manage this, we need a recursive algorithm that does not involve matrices of increasing dimension, like $\mathbf{J}^k$.

For single model filtering, $\mathbf{x}_k$ is a Markov process, i.e.,

$$p(\mathbf{x}_k \big| \mathbf{x}_{k-1}, \mathbf{x}_{k-2}, \mathbf{x}_{k-3}, \dots) = p(\mathbf{x}_k \big| \mathbf{x}_{k-1}). \tag{26}$$

As a consequence, the Fisher information matrix $\mathbf{J}^k$ inhabits a block diagonal structure with fortunate effects on the expressions for its inverse, $\left[\mathbf{J}^k\right]^{-1}$. In [3], these properties were used to find a recursive algorithm to calculate $\mathbf{J}_k$.

The same structure of $\mathbf{J}^k$ is not present for multiple model filtering. Still, the features of $\mathbf{J}^k$ has to be exploited in order to design the desired algorithm.

4.2.1 Properties of $\mathbf{J}^k$

For multiple model filtering, $\mathbf{x}_k$ is not a Markov process since[3]

$$p(\mathbf{x}_k \big| \mathbf{x}_{k-1}, \mathbf{x}_{k-2}, \mathbf{x}_{k-3}, \dots) \neq p(\mathbf{x}_k \big| \mathbf{x}_{k-1}). \tag{27}$$

Nonetheless, the dependence between $\mathbf{x}_k$ and $\mathbf{x}_{k-l}$ generally decreases rather rapidly, especially conditioned on the state vectors of all intermediate times, $\mathbf{X}_{k-l+1:k-1}$. Therefore, a reasonable assumption is that there exists an integer d such that

$$p(\mathbf{x}_k \big| \mathbf{X}_{1:k-1}) \approx p(\mathbf{x}_k \big| \mathbf{X}_{k-d:k-1}), \tag{28}$$

i.e., $\mathbf{x}_k$ and $\mathbf{x}_{k-l}$ are approximately independent for all $l > d$, when we condition on the state vectors, $\mathbf{X}_{k-d:k-1}$.

[3]To obtain a state vector which is a Markov process, $\mathbf{x}_k$ should be augmented with m_k.

The above assumption has two important implications on the Fisher information matrix $\mathbf{J}^k$. To clarify these we first introduce the notation

$$\mathbf{J}^k_{t_1:t_2 \times t_3:t_4} = E\left[-\triangle_{\mathbf{X}_{t_1:t_2}}^{\mathbf{X}_{t_3:t_4}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k})\right], \tag{29}$$

i.e., let $\mathbf{J}^k_{t_1:t_2 \times t_3:t_4}$ denote the submatrix of $\mathbf{J}^k$ that contains the rows that correspond to time t_1 to t_2 and the columns that correspond to time t_3 to t_4. Note that the dimension of $\mathbf{J}^k_{t_1:t_2 \times t_3:t_4}$ is $r(t_2 - t_1 + 1) \times r(t_4 - t_3 + 1)$ where r is the length of the state vector, $\mathbf{x}_k$.

Lemma 4.2 *Suppose $\mathbf{x}_k$ and $\mathbf{X}_{1:k-d-1}$ are conditionally independent for all $k > d$, in the sense that $p(\mathbf{x}_k \big| \mathbf{X}_{1:k-1}) = p(\mathbf{x}_k \big| \mathbf{X}_{k-d:k-1})$. It then follows that*

$$\mathbf{J}^k_{k_1:k_1 \times 1:k_1-1-d} = \left(\mathbf{J}^k_{1:k_1-1-d \times k_1:k_1}\right)^T = \mathbf{0} \tag{30}$$

$$\mathbf{J}^k_{1:k-d \times 1:k-d} = \mathbf{J}^{k+1}_{1:k-d \times 1:k-d}, \tag{31}$$

for any k and k_1 such that $k \geq k_1 > d$.

Proof of Lemma 4.2 *See Appendix B.*

From Lemma 4.2, we learn that large parts of $\mathbf{J}^k$ are zero and that the Fisher information matrix $\mathbf{J}^{k+1}$ has many elements in common with the slightly smaller matrix $\mathbf{J}^k$.

4.2.2 A recursive algorithm

To present a recursive algorithm, based on the conditions in (28), let us introduce the following set of notations[4],

$$\mathbf{A}_k = \mathbf{J}^k_{1:k-d-1 \times 1:k-d-1} \tag{32}$$

$$\mathbf{B}_k = \mathbf{J}^k_{k-d:k-1 \times 1:k-d-1} \tag{33}$$

$$\mathbf{C}_k = \mathbf{J}^k_{k-d:k-1 \times k-d:k-1} \tag{34}$$

$$\tilde{\mathbf{C}}_{k+1} = \mathbf{J}^{k+1}_{k-d:k-1 \times k-d:k-1} \tag{35}$$

$$\mathbf{D}_k = \mathbf{J}^k_{k:k \times k-d:k-1} \tag{36}$$

$$\tilde{\mathbf{D}}_{k+1} = \mathbf{J}^{k+1}_{k:k \times k-d:k-1} \tag{37}$$

$$\mathbf{E}_k = \mathbf{J}^k_{k:k \times k:k} \tag{38}$$

$$\tilde{\mathbf{E}}_{k+1} = \mathbf{J}^{k+1}_{k:k \times k:k}. \tag{39}$$

Proposition 4.1 *Let $\mathbf{H}_k$ be a matrix of dimension $rd \times rd$ such that*

$$\mathbf{J}_k = \mathbf{E}_k - \mathbf{D}_k \mathbf{H}_k^{-1} \mathbf{D}_k^T. \tag{40}$$

In general, $\mathbf{H}_k$ can be expressed as

$$\mathbf{H}_k = \mathbf{C}_k - \mathbf{B}_k \mathbf{A}_k^{-1} \mathbf{B}_k^T. \tag{41}$$

However, $\mathbf{H}_k$ can also be updated recursively using the formula

$$\mathbf{H}_k = \begin{bmatrix} \tilde{\mathbf{H}}_k^{22} & \left(\tilde{\mathbf{D}}_k^2\right)^T \\ \tilde{\mathbf{D}}_k^2 & \tilde{\mathbf{E}}_k \end{bmatrix} - \begin{bmatrix} \tilde{\mathbf{H}}_k^{12} \\ \tilde{\mathbf{D}}_k^1 \end{bmatrix} \left(\tilde{\mathbf{H}}_k^{11}\right)^{-1} \begin{bmatrix} \tilde{\mathbf{H}}_k^{12} \\ \tilde{\mathbf{D}}_k^1 \end{bmatrix}^T, \tag{42}$$

[4]See (83) and (86), Appendix C, for illustrations of their relations to $\mathbf{J}^k$.

where $\tilde{\mathbf{H}}_k = \mathbf{H}_{k-1} + \tilde{\mathbf{C}}_k - \mathbf{C}_{k-1}$. The remaining matrices in (42) *are submatrices of $\tilde{\mathbf{H}}_k$ and $\tilde{\mathbf{D}}_k$ according to*

$$\tilde{\mathbf{H}}_k = \begin{bmatrix} \tilde{\mathbf{H}}_k^{11} & \left(\tilde{\mathbf{H}}_k^{21}\right)^T \\ \tilde{\mathbf{H}}_k^{21} & \tilde{\mathbf{H}}_k^{22} \end{bmatrix} \tag{43}$$

$$\tilde{\mathbf{D}}_k = \begin{bmatrix} \tilde{\mathbf{D}}_k^1 & \tilde{\mathbf{D}}_k^2 \end{bmatrix} \tag{44}$$

where $\tilde{\mathbf{H}}_k^{11}$ and $\tilde{\mathbf{D}}_k^1$ are of dimension $r \times r$, $\tilde{\mathbf{H}}_k^{21}$ and $\left(\tilde{\mathbf{D}}_k^2\right)^T$ are of dimension $(d-1)r \times r$, whereas $\tilde{\mathbf{H}}_k^{22}$ is a $(d-1)r \times (d-1)r$ dimensional matrix.

Proof of proposition 4.1 *See Appendix C.*

The proposition provides us with a recursive algorithm to calculate the BCRB. It relies, however, on the assumption that we can calculate $\mathbf{J}^k_{k-d:k\times k-d:k}$ at each recursion. As this is a submatrix of $\mathbf{J}^k$, it can be obtained using the techniques in Section 4.1. However, we are now only interested in $\mathbf{J}^k_{k-d:k\times k-d:k}$, and we can approximate the corresponding parts of $\mathbf{J}_{\mathbf{X}_{1:k}}$ as

$$\frac{1}{N}\sum_{n=1}^{N} \frac{\nabla_{\mathbf{X}_{k-d:k}} p(\mathbf{X}_{1:k}^{(n)})(\nabla_{\mathbf{X}_{k-d:k}} p(\mathbf{X}_{1:k}^{(n)}))^T}{p(\mathbf{X}_{1:k}^{(n)})^2}. \tag{45}$$

A simpler version of the algorithm in Section 4.1 hence emerges; instead of handling the vector $\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)})$, it is sufficient to store and update the much shorter vector $\nabla_{\mathbf{X}_{k-d:k}} p(\mathbf{X}_{1:k}^{(n)})$. In order to handle $\nabla_{\mathbf{X}_{k-d:k}} p(\mathbf{X}_{1:k}^{(n)})$ instead of $\nabla_{\mathbf{X}_{1:k}} p(\mathbf{X}_{1:k}^{(n)})$, we replace $\nabla_{\mathbf{X}_{1:k}}$ with $\nabla_{\mathbf{X}_{k-d:k}}$ in equations (15), (21) and (22).

5 Performance evaluation

We compare the BCRB to the performance of the optimal filter and to Enumer-BCRB [11]. In order to enable an exact implementation of an optimal filter, we limit our study to systems that switch between linear and Gaussian models.

5.1 Reference algorithm

We here provide some brief technical background for the two algorithms that we use to benchmark performance.

5.1.1 Optimal filter

We use a Monte Carlo simulation to evaluate the average performance of the optimal filter in the Mean Square Error (MSE) sense. The result of the simulation converges to the desired optimal performance for an increasing number of Monte Carlo samples, but is very demanding to compute.

Conditioned on the model sequence, $\mathbf{m}_k = [m_1, \ldots, m_k]^T$, the posterior density, $p(\mathbf{x}_k | \mathbf{Y}_{1:k}, \mathbf{m}_k)$, is Gaussian, $\mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k,\mathbf{m}_k}, \hat{\mathbf{P}}_{k,\mathbf{m}_k})$, where the posterior moments, $\hat{\mathbf{x}}_{k,\mathbf{m}_k}$ and $\hat{\mathbf{P}}_{k,\mathbf{m}_k}$, are easily calculated using a Kalman filter. The optimal filter output, which minimizes the MSE, is the posterior mean

$$\hat{\mathbf{x}}_k = \sum_{\mathbf{m}_k} \hat{\mathbf{x}}_{k,\mathbf{m}_k} \Pr\{\mathbf{m}_k | \mathbf{Y}_{1:k}\}, \tag{46}$$

where

$$\Pr\{\mathbf{m}_k | \mathbf{Y}_{1:k}\} \propto \Pr\{\mathbf{m}_k\} p(\mathbf{Y}_{1:k} | \mathbf{m}_k). \tag{47}$$

Here, the model likelihood, $p(\mathbf{Y}_{1:k} | \mathbf{m}_k)$, is calculated using the same Kalman filter that we used to obtain $\hat{\mathbf{x}}_{k,\mathbf{m}_k}$ and $\hat{\mathbf{P}}_{k,\mathbf{m}_k}$. Each term in (46) is therefore easy to evaluate, but since the number of possible model sequences, M^k, increases rapidly, the filter is intractable even for moderate values of k. Of course, to obtain the average performance we also need to run it many times.

5.1.2 Enumer-BCRB

The bound here referred to as the Enumer-BCRB, was proposed in [11]. The underlying idea is to use results for the BCRB when the model sequence is given. Let $\mathbf{J}^k(\mathbf{m}_k)$ denote the information matrix for $\mathbf{X}_{1:k}$ when $\mathbf{m}_k$ is known,

$$\mathbf{J}^k(\mathbf{m}_k) = E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k} | \mathbf{m}_k) | \mathbf{m}_k\right]. \tag{48}$$

Hence, $\mathbf{J}^k(\mathbf{m}_k)^{-1}$ is the BCRB for $\mathbf{X}_{1:k}$ given $\mathbf{m}_k$ and we denote the corresponding BCRB for $\mathbf{x}_k$ as $\mathbf{J}_k(\mathbf{m}_k)^{-1}$, where $\mathbf{J}_k(\mathbf{m}_k)^{-1}$ is the bottom right submatrix of $\mathbf{J}^k(\mathbf{m}_k)^{-1}$. Due to results in [3] this bound can be computed recursively, and for the models in these simulations the bound even has an analytic expression. To obtain a lower bound when $\mathbf{m}_k$ is unknown, we notice that $E\left\{(\mathbf{x}_k - \delta(\mathbf{Y}_{1:l}))(\mathbf{x}_k - \delta(\mathbf{Y}_{1:l}))^T\right\} =$

$$
\begin{aligned}
&= E\left[E\left[(\mathbf{x}_k - \delta(\mathbf{Y}_{1:k}))(\mathbf{x}_k - \delta(\mathbf{Y}_{1:k}))^T | \mathbf{m}_k\right]\right] && (49)\\
&\geq E\left[\mathbf{J}_k(\mathbf{m}_k)^{-1}\right] && (50)\\
&= \sum_{\mathbf{m}_k} \mathbf{J}_k(\mathbf{m}_k)^{-1} \Pr\{\mathbf{m}_k\}. && (51)
\end{aligned}
$$

Here $\sum_{\mathbf{m}_k} \mathbf{J}_k(\mathbf{m}_k)^{-1} \Pr\{\mathbf{m}_k\}$ is the Enumer-BCRB.

5.2 Simulation scenario

Consider a two model system that can switch between a Nearly Constant Acceleration (NCA) model and a Nearly Constant Velocity (NCV) model. The three elements of the state vector, $\mathbf{x}_k = [x_k, \dot{x}_k, \ddot{x}_k]^T$, represent position, velocity and acceleration of an object that moves in one dimension. In the simulations, we use a measurement model defined by

$$\mathbf{G}_k = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T \tag{52}$$

$$\mathbf{w}_k = w_k \sim \mathcal{N}(0, 50), \tag{53}$$

i.e., we make a noisy observation of the position state, x_k, see (2). The sample interval is $T = 5$ s and the prior density for $\mathbf{x}_k$ is $\mathbf{x}_0 \sim \mathcal{N}(\begin{bmatrix} 2 & 2 & 2 \end{bmatrix}^T, \mathbf{I})$. We define our model transition probabilities using the relations

$$\Pr\{m_k = 1 | m_{k-1} = 1\} = 0.9 \tag{54}$$

$$\Pr\{m_k = 2 | m_{k-1} = 2\} = 0.9, \tag{55}$$

and set the initial model probability to $\Pr\{m_0 = 1\} = 0.5$ (and therefore naturally, $\Pr\{m_0 = 2\} = 0.5$).

5.2.1 Model 1

For the Nearly Constant Acceleration (NCA) model, for which $m_k = 1$, we have,

$$f_k(\mathbf{x}_{k-1}, 1) = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}_{k-1} \tag{56}$$

$$\mathbf{Q}_{k,1} = \mathbf{Q}_1 = s_1 \begin{bmatrix} T^5/20 & T^4/8 & T^3/6 \\ T^4/8 & T^3/3 & T^2/2 \\ T^3/6 & T^2/2 & T \end{bmatrix}. \tag{57}$$

Here, $s_1 = 0.4$ is the power spectral density.

5.2.2 Model 2

To maintain the dimensionality of the state vector, we define the Nearly Constant Velocity (NCV) model, $m_k = 2$, as

$$f_k(\mathbf{x}_{k-1}, 2) = \begin{bmatrix} 1 & T & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{x}_{k-1} \tag{58}$$

$$\mathbf{Q}_{k,2} = \mathbf{Q}_2 = s_2 \begin{bmatrix} T^3/3 & T^2/2 & 0 \\ T^2/2 & T & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{59}$$

We use the power spectral density $s_2 = 2$.

5.3 Simulation Results

We will now compare the Root Mean Square (RMS) error, i.e., the square root of the MSE, of the optimal estimator, with the square root of the diagonal of the BCRB and the Enumer-BCRB. In the top part of Figure 1, the BCRB is calculated using the non-recursive algorithm in Section 4.1, and for both the BCRB and the optimal filter we have used 15000 Monte Carlo samples. As we can see, the BCRB is slightly overoptimistic, which is expected as the posterior density is not Gaussian [1]. The Enumer-BCRB, however, is less optimistic and predicts the performance very well.

In the bottom part of Figure 1, we compare the output from the algorithm in Section 4.1 with the recursive version in Section 4.2. The recursive algorithm is evaluated for both $d = 5$ and $d = 15$. For the studied scenario, it appears that $d = 15$ is a sufficient depth.

6 Discussion

In [12], the Enumer-BCRB was criticized for being overoptimistic as it implicitly provides a bound for the optimal performance when the estimator *knows* the model sequence $\mathbf{m}_k$. Clearly, if the model uncertainties are significant, the best achievable performance may be far from this bound. Still, simulations show that the BCRB is often even more optimistic than the Enumer-BCRB, and sometimes the difference between the two is very large.

Generally speaking, the reason why the BCRB is sometimes very overoptimistic is that $\mathbf{J}^k$ is typically dominated by the most informative model sequences, i.e., the ones for which the estimation performance is the best. Therefore, if

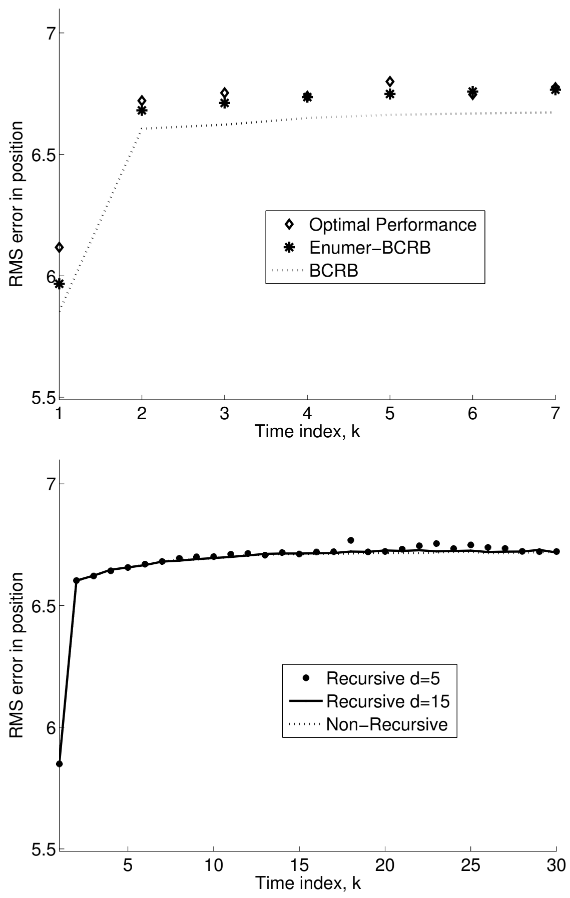

Figure 1: The RMS error of the position, x_k, compared to the square root of the BCRB and the Enumer-BCRB (top), and the square root of the BCRB for different implementations (bottom).

there are other model sequences, $\mathbf{m}_k$, for which the optimal average performance is much worse, this will have very limited influence on the BCRB, even though it may greatly deteriorate the best achievable performance.

To clarify this further, let us consider a situation where the relation between the BCRB and the Enumer-BCRB has an approximative analytic expression. Suppose the models are well separated, in the sense that

$$p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k}) = \sum_{\mathbf{m}'_k} \Pr\{\mathbf{m}'_k\} p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k} \big| \mathbf{m}'_k) \tag{60}$$

$$\approx \Pr\{\mathbf{m}_k\} p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k} \big| \mathbf{m}_k) \tag{61}$$

for almost all realizations $(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k}) \sim p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k} \big| \mathbf{m}_k)$. It follows that

$$\mathbf{J}^k(\mathbf{m}_k) = E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k} \big| \mathbf{m}_k) \big| \mathbf{m}_k \right] \tag{62}$$

$$\approx E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k}) \big| \mathbf{m}_k \right] \tag{63}$$

and we therefore get

$$\mathbf{J}^k = E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k})\right] \tag{64}$$

$$= E\left[E\left[-\triangle_{\mathbf{X}_{1:k}}^{\mathbf{X}_{1:k}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k}) \middle| \mathbf{m}_k\right]\right] \tag{65}$$

$$\approx E\left[\mathbf{J}^k(\mathbf{m}_k)\right]. \tag{66}$$

Thus, the relation between the BCRB and the Enumer-BCRB for $\mathbf{X}_{1:k}$ is

$$\left(\mathbf{J}^k\right)^{-1} \approx E\left\{\mathbf{J}^k(\mathbf{m}_k)\right\}^{-1} \leq E\left\{\mathbf{J}^k(\mathbf{m}_k)^{-1}\right\}, \tag{67}$$

where the inequality is due to Jensen's inequality. Note that the corresponding bounds for $\mathbf{x}_k$ is the bottom right submatrices of these matrices.

From (66), we understand that if there is one model for which $\mathbf{J}^k(\mathbf{m}_k)$ is much larger, that term will dominate $\mathbf{J}^k$ and hence $(\mathbf{J}^k)^{-1}$. The Enumer-BCRB, on the other hand, better resembles the optimal performance as it depends more on the least informative model. Of course, scenarios where the model uncertainties are negligible are not representative, and favor the Enumer-BCRB, but the derivations still illuminate an important problem with the BCRB.

7 Conclusions

In this paper, we have studied the problem of calculating the Bayesian Cramér-Rao Bound (BCRB) for Markovian switching systems. The calculation of the bound is complicated by the fact that the discrete valued model index has to be marginalized from the expressions. A numerical Monte Carlo method to evaluate the Fisher information matrix of the complete trajectory of state vectors has been proposed. Based on this information matrix we obtain the desired bound, which has not been calculated previously.

To obtain an algorithm which is applicable for sequences of arbitrary length, we have also developed a recursive technique to calculate the BCRB. The technique is based on the assumption that state vectors are conditionally independent if the intermediate time interval is sufficiently large. From the simulations, we observe that the BCRB is more overoptimistic than previously suggested bounds, and we provide a possible explanation of this important result.

A Proof of Lemma 4.1

We wish to calculate

$$\nabla_{\mathbf{x}_j} p(\mathbf{x}_k | \mathbf{x}_{k-1}, m_k) \tag{68}$$

and we are given the relation

$$\mathbf{x}_k = f_{k-1}(\mathbf{x}_{k-1}, m_k) + \mathbf{v}_k, \tag{69}$$

where $\mathbf{v}_k \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{k,m_k})$. For convenience, we temporarily replace $f_{k-1}(\mathbf{x}_{k-1}, m_k)$ and $\mathbf{Q}_{k,m_k}$ with the notations $f(\mathbf{x}_{k-1})$ and $\mathbf{Q}$, respectively.

To evaluate (68), we use (69) to get

$$\nabla_{\mathbf{x}_j} p(\mathbf{x}_k | \mathbf{x}_{k-1}, m_k) = \nabla_{\mathbf{x}_j} \left|2\pi\mathbf{Q}\right|^{-1/2} \exp\Big\{-\frac{1}{2}$$
$$\left(\mathbf{x}_k - f(\mathbf{x}_{k-1})\right)^T \mathbf{Q}^{-1} \left(\mathbf{x}_k - f(\mathbf{x}_{k-1})\right)\Big\} \tag{70}$$
$$= -\frac{1}{2} p(\mathbf{x}_k | \mathbf{x}_{k-1}, m_k)$$
$$\nabla_{\mathbf{x}_j} \left(\mathbf{x}_k - f(\mathbf{x}_{k-1})\right)^T \mathbf{Q}^{-1} \left(\mathbf{x}_k - f(\mathbf{x}_{k-1})\right). \tag{71}$$

Straightforward calculations yield that

$$-\frac{1}{2}\nabla_{\mathbf{x}_j} \left(\mathbf{x}_k - f(\mathbf{x}_{k-1})\right)^T \mathbf{Q}^{-1} \left(\mathbf{x}_k - f(\mathbf{x}_{k-1})\right)$$
$$= \begin{cases} \left(\nabla_{\mathbf{x}_{k-1}} f(\mathbf{x}_{k-1})^T\right) \mathbf{Q}^{-1}(\mathbf{x}_k - f(\mathbf{x}_{k-1})) & \text{if } j = k-1 \\ -\mathbf{Q}^{-1}(\mathbf{x}_k - f(\mathbf{x}_{k-1})) & \text{if } j = k \\ \mathbf{0} & \text{if } j \notin \{k-1, k\}. \end{cases} \tag{72}$$

Lemma 4.1 follows from (71) and (72).

B Proof of Lemma 4.2

We know that $\mathbf{J}^k = \mathbf{J}_{\mathbf{X}_{1:k}} + \mathbf{J}_{\mathbf{Y}_{1:k}}$. To show (30) we focus on $\mathbf{J}_{\mathbf{X}_{1:k}}$, since $\mathbf{J}_{\mathbf{Y}_{1:k}}$ is already known to have the block diagonal structure in (10). The assumption of conditional independence in the lemma yields that

$$p(\mathbf{X}_{1:k}) = p(\mathbf{X}_{1:d}) \prod_{t=d+1}^{k} p(\mathbf{x}_t | \mathbf{X}_{t-d:t-1}). \tag{73}$$

For $k_1 > d$ we get

$$\nabla_{\mathbf{x}_{k_1}} \log p(\mathbf{X}_{1:k}) = \sum_{t=k_1}^{k} \nabla_{\mathbf{x}_{k_1}} \log p(\mathbf{x}_t | \mathbf{X}_{t-d:t-1}). \tag{74}$$

Clearly, $\nabla_{\mathbf{x}_{k_1}} \log p(\mathbf{X}_{1:k})$ does not depend on $\mathbf{X}_{1:k_1-d-1}$ and we therefore have

$$\triangle_{\mathbf{x}_{k_1}}^{\mathbf{X}_{1:k_1-d-1}} \log p(\mathbf{X}_{1:k}) = \left(\triangle_{\mathbf{X}_{1:k_1-d-1}}^{\mathbf{x}_{k_1}} \log p(\mathbf{X}_{1:k})\right)^T = \mathbf{0}. \tag{75}$$

The information matrix $\mathbf{J}_{\mathbf{X}_{1:k}}$ thus has zeros in the corresponding elements, and this shows that (30) holds.

To prove that

$$\mathbf{J}^k_{1:k-d\times 1:k-d} = \mathbf{J}^{k+1}_{1:k-d\times 1:k-d}, \tag{76}$$

we return to the definition of the matrices. The right-hand side of (76) is defined as $\mathbf{J}^{k+1}_{1:k-d\times 1:k-d} =$

$$=E\left[-\triangle_{\mathbf{X}_{1:k-d}}^{\mathbf{X}_{1:k-d}} \log p(\mathbf{X}_{1:k+1}, \mathbf{Y}_{1:k+1})\right] \tag{77}$$

$$= -E\Big[\triangle_{\mathbf{X}_{1:k-d}}^{\mathbf{X}_{1:k-d}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k})$$
$$+ \triangle_{\mathbf{X}_{1:k-d}}^{\mathbf{X}_{1:k-d}} \log p(\mathbf{x}_{k+1}, \mathbf{y}_{k+1} | \mathbf{X}_{1:k})\Big] \tag{78}$$

$$= -E\left[\triangle_{\mathbf{X}_{1:k-d}}^{\mathbf{X}_{1:k-d}} \log p(\mathbf{X}_{1:k}, \mathbf{Y}_{1:k})\right] \tag{79}$$

$$=\mathbf{J}^k_{1:k-d\times 1:k-d}. \tag{80}$$

To obtain (79) from (78) we recall that $\log p(\mathbf{x}_{k+1}, \mathbf{y}_{k+1} | \mathbf{X}_{1:k}) = \log p(\mathbf{x}_{k+1}, \mathbf{y}_{k+1} | \mathbf{X}_{k-d+1:k})$. This concludes the proof.

C Proof of Proposition 4.1

In this derivation, we will repeatedly use that if

$$\mathbf{H} = \begin{bmatrix} \mathbf{H}_{11} & \mathbf{H}_{21}^T \\ \mathbf{H}_{21} & \mathbf{H}_{22} \end{bmatrix}, \tag{81}$$

then the submatrix in the lower right corner of $\mathbf{H}^{-1}$ (with the same dimensions as $\mathbf{H}_{22}$) is

$$\left(\mathbf{H}_{22} - \mathbf{H}_{21}\mathbf{H}_{11}^{-1}\mathbf{H}_{21}^T\right)^{-1}. \tag{82}$$

From the definitions in (32)–(39), we learn that

$$\mathbf{J}^k = \begin{bmatrix} \mathbf{A}_k & \mathbf{B}_k^T & \mathbf{0}_{(k-d-1)r\times r} \\ \mathbf{B}_k & \mathbf{C}_k & \mathbf{D}_k^T \\ \mathbf{0}_{r\times(k-d-1)r} & \mathbf{D}_k & \mathbf{E}_k \end{bmatrix} \tag{83}$$

and it therefore follows that

$$\mathbf{J}_k = \mathbf{E}_k - \mathbf{D}_k\left(\mathbf{C}_k - \mathbf{B}_k\mathbf{A}_k^{-1}\mathbf{B}_k^T\right)^{-1}\mathbf{D}_k^T \tag{84}$$

which verifies (41).

We want to identify an expression for $\mathbf{H}_{k+1}$ from the relation

$$\mathbf{J}_{k+1} = \mathbf{E}_{k+1} - \mathbf{D}_{k+1}\mathbf{H}_{k+1}^{-1}\mathbf{D}_{k+1}^T. \tag{85}$$

However, in order to make the equations more compact we will express $\mathbf{E}_{k+1} - \mathbf{J}_{k+1}$ instead of $\mathbf{J}_{k+1}$. As a consequence of Lemma 4.2, we get $\mathbf{J}^{k+1} =$

$$\begin{bmatrix} \mathbf{A}_k & \begin{bmatrix}\mathbf{B}_k^T & \mathbf{0}_{(k-d-1)r\times r}\end{bmatrix} & \mathbf{0}_{(k-d-1)r\times r} \\ \begin{bmatrix}\mathbf{B}_k \\ \mathbf{0}_{r\times(k-d-1)r}\end{bmatrix} & \begin{bmatrix}\tilde{\mathbf{C}}_{k+1} & \tilde{\mathbf{D}}_{k+1}^T \\ \tilde{\mathbf{D}}_{k+1} & \tilde{\mathbf{E}}_{k+1}\end{bmatrix} & \begin{bmatrix}\mathbf{0}_{r\times r} \\ \mathbf{D}_{k+1}^T\end{bmatrix} \\ \mathbf{0}_{(k-d-1)r\times r} & \begin{bmatrix}\mathbf{0}_{r\times r} & \mathbf{D}_{k+1}\end{bmatrix} & \mathbf{E}_{k+1} \end{bmatrix} \tag{86}$$

and hence $\mathbf{E}_{k+1} - \mathbf{J}_{k+1} =$

$$= \check{\mathbf{D}}_{k+1}\left(\begin{bmatrix}\tilde{\mathbf{C}}_{k+1} & \tilde{\mathbf{D}}_{k+1}^T \\ \tilde{\mathbf{D}}_{k+1} & \tilde{\mathbf{E}}_{k+1}\end{bmatrix} - \begin{bmatrix}\mathbf{B}_k\mathbf{A}_k^{-1}\mathbf{B}_k^T & \mathbf{0}_{dr\times r} \\ \mathbf{0}_{r\times dr} & \mathbf{0}_{r\times r}\end{bmatrix}\right)^{-1}\check{\mathbf{D}}_{k+1}^T$$

$$= \check{\mathbf{D}}_{k+1}\begin{bmatrix}\tilde{\mathbf{C}}_{k+1} - \mathbf{B}_k\mathbf{A}_k^{-1}\mathbf{B}_k^T & \tilde{\mathbf{D}}_{k+1}^T \\ \tilde{\mathbf{D}}_{k+1} & \tilde{\mathbf{E}}_{k+1}\end{bmatrix}^{-1}\check{\mathbf{D}}_{k+1}^T, \tag{87}$$

where we have introduced $\check{\mathbf{D}}_{k+1} = \begin{bmatrix}\mathbf{0}_{r\times r} & \mathbf{D}_{k+1}\end{bmatrix}$.

We define $\tilde{\mathbf{H}}_{k+1} \triangleq \mathbf{H}_k + \tilde{\mathbf{C}}_{k+1} - \mathbf{C}_k$ and notice that $\tilde{\mathbf{H}}_{k+1} = \tilde{\mathbf{C}}_{k+1} - \mathbf{B}_k\mathbf{A}_k^{-1}\mathbf{B}_k^T$, see (41). With this notation, we get $\mathbf{E}_{k+1} - \mathbf{J}_{k+1} =$

$$\check{\mathbf{D}}_{k+1}\begin{bmatrix}\tilde{\mathbf{H}}_{k+1}^{11} & \left(\tilde{\mathbf{H}}_{k+1}^{21}\right)^T & \left(\tilde{\mathbf{D}}_{k+1}^{1}\right)^T \\ \tilde{\mathbf{H}}_{k+1}^{21} & \tilde{\mathbf{H}}_{k+1}^{22} & \left(\tilde{\mathbf{D}}_{k+1}^{2}\right)^T \\ \tilde{\mathbf{D}}_{k+1}^{1} & \tilde{\mathbf{D}}_{k+1}^{2} & \tilde{\mathbf{E}}_{k+1}\end{bmatrix}^{-1}\check{\mathbf{D}}_{k+1}^T \tag{88}$$

and from (85) and (88) we can identify the relation $\mathbf{H}_{k+1} =$

$$\begin{bmatrix}\tilde{\mathbf{H}}_{k+1}^{22} & \left(\tilde{\mathbf{D}}_{k+1}^{2}\right)^T \\ \tilde{\mathbf{D}}_{k+1}^{2} & \tilde{\mathbf{E}}_{k+1}\end{bmatrix} - \begin{bmatrix}\tilde{\mathbf{H}}_{k+1}^{21} \\ \tilde{\mathbf{D}}_{k+1}^{1}\end{bmatrix}\left(\tilde{\mathbf{H}}_{k+1}^{11}\right)^{-1}\begin{bmatrix}\tilde{\mathbf{H}}_{k+1}^{21} \\ \tilde{\mathbf{D}}_{k+1}^{1}\end{bmatrix}^T. \tag{89}$$

Since (89) holds for any k, this concludes our proof.

References

[1] H. V. Trees, *Detection, Estimation and Modulation Theory*. New York: Wiley, 1968.

[2] H. V. Trees and K. Bell, *Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking*. Wiley-IEEE Press, 2007.

[3] P. Tichavský, C. Muravchik, and A. Nehorai, "Posterior Cramér-Rao bounds for discrete-time nonlinear filtering," *IEEE Trans. Signal Process.*, vol. 46, no. 5, pp. 1386–1396, May 1998.

[4] N. Bergman, "Recursive Bayesian estimation: Navigation and tracking applications," Ph.D. dissertation, Department of Electrical Engineering, Linköping University, 1999.

[5] M. Šimandl, J. Královec, and P. Tichavský, "Filtering, predictive and smoothing Cramér-Rao bounds for discrete-time nonlinear dynamic systems," *Automatica*, vol. 37, pp. 1703–1716, 2001.

[6] X. Zhang, P. Willett, and Y. Bar-Shalom, "Dynamic Cramér-Rao bound for target tracking in clutter," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, no. 4, October 2005.

[7] M. L. Hernandez, A. Farina, and B. Ristic, "PCRLB for tracking in cluttered environments: Measurement sequence conditioning approach," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 42, no. 2, pp. 680–704, April 2006.

[8] R. Tharmarasa, T. Kirubarajan, M. Hernandez, and A. Sinha, "PCRLB-based multisensor array management for multitarget tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 43, no. 2, pp. 539–555, April 2007.

[9] M. L. Hernandez, T. Kirubarajan, and Y. Bar-Shalom, "Multisensor resource deployment using posterior Cramér-Rao bounds," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 40, no. 2, pp. 399–416, 2004.

[10] H. Bloom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with Markovian switching coefficients," *IEEE Trans. on Automatic Control*, vol. 33, pp. 780–783.

[11] A. Bessel, B. Ristic, A. Farina, W. Wang, and M. Arulampalam, "Error performance bounds for tracking a manoeuvering target," in *Proc. 6th Int. Conference on Inform. Fusion*, Cairns, Queensland, Australia, 2003.

[12] M. L. Hernandez, B. Ristic, A. Farina, T. Sathyan, and T. Kirubarajan, "Performance measure for Markovian switching systems using best-fitting Gaussian distributions," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 44, no. 2, pp. 724–747, April 2008.