18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

The Kalman Laplace filter: A new deterministic algorithm for nonlinear Bayesian filtering

Paul Bui Quang
CEA, DAM, DIF
F-91297 Arpajon, France
Email: paul.bui-quang@cea.fr

Christian Musso
Onera – The French Aerospace Lab
F-91123 Palaiseau, France
Email: christian.musso@onera.fr

François Le Gland
Inria Rennes Bretagne–Atlantique
F-35042 Rennes, France
Email: francois.le_gland@inria.fr

***Abstract*—We propose a new recursive algorithm for nonlinear Bayesian filtering, where the prediction step is performed like in the extended Kalman filter, and the update step is done thanks to the Laplace method for integral approximation. This algorithm is called the Kalman Laplace filter (KLF). The KLF provides a closed–form non–Gaussian approximation of the posterior density. The hidden state is estimated by the maximum a posteriori, using a dimension reduction method to alleviate the computation cost of the maximization. The KLF is tested on three simulated nonlinear filtering problems: target tracking with angle measurements, population dynamics monitoring, motion reconstruction by neural decoding. It exhibits a good performance, especially when the observation noise is small.**

I. Introduction

Bayesian filtering is the problem of sequentially estimating the hidden state in dynamical state–space models thanks to the available observations. This problem is solved by computing the conditional distribution of the hidden state with respect to the observations at each time step, i.e., the posterior distribution. Bayesian filtering arises in various contexts, in engineering, economics, or computational biology for example.

The Kalman filter [1] yields optimal estimation when the state–space model is linear and Gaussian. When the model is nonlinear, adaptations of the Kalman filter, namely the extended Kalman filter (EKF) [1] and the unscented Kalman filter (UKF) [2], can be used to perform the estimation. In the Kalman filter, the EKF, and the UKF, the posterior distribution is approximated by a Gaussian and the state is estimated by the posterior distribution mean. Another type of filtering algorithms are the particle filters, or sequential Monte Carlo methods. Particle filters approximate the posterior distribution by a weighted sum of Dirac measures. The first practical implementation of a particle filter is the bootstrap filter, proposed in [3] with a target tracking application.

In this paper, we propose a novel deterministic Bayesian filtering algorithm, the Kalman Laplace filter (KLF), adapted to nonlinear problems. This algorithm combines approximation techniques inspired by the Kalman filter and the Laplace method. The Laplace method is an integral approximation method, which has many applications in Bayesian estimation problems [4]. In the KLF, at each time iteration, the prediction step is performed like in the EKF and the update step is performed thanks to the Laplace method. The posterior distribution is approximated by a non–Gaussian density for which a closed–form expression is provided. This closed–form expression allows to compute the maximum of the posterior distribution (MAP) using standard optimization routines. The MAP is retrieved as a state estimate.

The outline of the paper is as follows. In Section II, we define state–space models and present the Bayesian filtering problem. The Laplace method and the Kalman Laplace filter are described in Section III. In Section IV, we derive a method to alleviate the cost of the MAP computation as in [5]. Simulation experiments are performed in Section V. Three nonlinear filtering problems are tackled: target tracking with angle measurements, population dynamics monitoring with survey counts, neural decoding for motion analysis.

II. State–space models and Bayesian filtering

A. State–space models

State–space models are time series models where the discrete–time observation process $\{Y_k\}_{k\geq 0}$ is driven by a hidden (i.e., unobserved) state Markov process $\{X_k\}_{k\geq 0}$. The distribution of the joint process $\{Y_k, X_k\}_{k\geq 0}$ is supposed to verify[1]

$$\begin{aligned}&\mathbb{P}[X_{0:n} \in dx_{0:n}, Y_{0:n} \in dy_{0:n}]\\ &\quad = \mathbb{P}[X_0 \in dx_0]\mathbb{P}[Y_0 \in dy_0|X_0 \in dx_0]\\ &\qquad \times \prod_{k=1}^{n} \mathbb{P}[Y_k \in dy_k|X_k \in dx_k]\mathbb{P}[X_k \in dx_k|X_{k-1} \in dx_{k-1}].\end{aligned}$$

In this paper, we consider that the hidden state X_k takes values in $\mathbb{R}^d$ for all $k \geq 0$. We denote[2] $g_k(x_k)$ the conditional density of Y_k w.r.t. X_k, for all $k \geq 0$. We denote $q_k(x_{k-1}, \cdot)$ the conditional density of X_k w.r.t. X_{k-1}, i.e., the Markov transition kernel density, for all $k \geq 1$. The density of the initial state X_0 is denoted q_0.

State–space models are sometimes referred to as hidden Markov models [6], yet mostly when the hidden state takes symbolic values, i.e., when the state space is countable.

[1]Throughout the paper, we use the notation $a_{1:n} = (a_1, \ldots, a_n)$.

[2]Throughout the paper, we ignore in our notations the dependency on the observations of any quantity. This is because all the operations (derivatives, integrals) are made w.r.t. the state variable, so that this mild abuse allows to alleviate notations.

State–space models are often in the following form. The state process is a nonlinear autoregressive time series in the form of

$$X_k = F_k(X_{k-1}, V_k) \tag{1}$$

for all $k \geq 1$, where $F_k : \mathbb{R}^d \times R^{d_X} \to \mathbb{R}^d$ is a nonlinear mapping and $\{V_k\}_{k\geq 0}$ is an i.i.d. Gaussian white noise. V_k takes values in $\mathbb{R}^{d_V}$ and $V_k \sim \mathcal{N}(0, Q_k)$ for all $k \geq 1$. The initial state distribution is Gaussian, $X_0 \sim \mathcal{N}(m_0, Q_0)$. The observation model is in the form of

$$Y_k = H_k(X_k, W_k) \tag{2}$$

for all $k \geq 0$, where $H_k : \mathbb{R}^d \times R^{d_W} \to \mathbb{R}^{d_Y}$ is a nonlinear mapping and $\{W_k\}_{k\geq 0}$ is an i.i.d. Gaussian white noise. W_k takes values in $\mathbb{R}^{d_W}$ and $W_k \sim \mathcal{N}(0, R_k)$ for all $k \geq 1$.

In particular, the linearized versions of the Kalman filter (EKF and UKF) can perform filtering in such models when the mappings F_k and H_k are differentiable w.r.t. their two arguments. The Kalman Laplace filter that we propose in Section III can handle more general state–space models, since it not required the observation model is in the form of (2) (for example when the observations are discrete).

B. Bayesian filtering

The problem of Bayesian filtering consists in estimating the conditional distribution of the state X_k w.r.t. all the available observations $Y_{0:k}$, at each time iteration $k \geq 0$. This distribution obeys to a recursion, which we derive here.

Let p_{k-1} denote the conditional density of X_{k-1} w.r.t. $Y_{0:k-1}$ and $p_{k|k-1}$ denote the conditional density of X_k w.r.t. $Y_{0:k-1}$. We have that

$$p_{k|k-1}(x_k) = \int_{\mathbb{R}^d} q_k(x_{k-1}, x_k) p_{k-1}(x_{k-1}) dx_{k-1}, \tag{3}$$

which is called the prediction step. Then, the Bayes formula yields

$$p_k(x_k) = \frac{g_k(x_k) p_{k|k-1}(x_k)}{\int_{\mathbb{R}^d} g_k(x_k) p_{k|k-1}(x_k) dx_k}, \tag{4}$$

which is called the update step. Hence, in Bayesian terms, g_k is the likelihood, $p_{k|k-1}$ is the prior (also called predictor in the context of filtering), and p_k is the posterior.

The Bayesian filtering algorithms provide an approximation $\hat{p}_k$ of the posterior at each time iteration. This approximation is deterministic and Gaussian in the Kalman filter [1], the EKF [1], and the UKF [2]), in the form of

$$\hat{p}_k(x) = (2\pi)^{-d/2} \det \hat{P}_k^{-1/2} \times \exp\left(-\frac{1}{2}(x - \hat{m}_k)^T \hat{P}_k^{-1} (x - \hat{m}_k)\right)$$

for all $x \in \mathbb{R}^d$, where $\hat{m}_k$ and $\hat{P}_k$ are respectively estimates of the posterior mean and covariance matrix.

In the particle filters [3], [7], the approximation $\hat{p}_k$ is a Monte Carlo approximation, in the form of a weighted sum of Dirac measures,

$$\hat{p}_k = \frac{1}{N} \sum_{i=1}^{N} w_k^i \delta_{\xi_k^i}$$

where $\xi_k^1, \ldots, \xi_k^N$ is a random sample of so-called particles. In the simulations in Section V, we use the most common version of a particle filter, the sequential importance resampling (SIR) algorithm (see Algorithm 4 in [7]). The particles are propagated thanks to the Markov kernel of the state process. Resampling is performed when the effective sample size is too low.

III. The Kalman Laplace filter

In this Section we describe the Kalman Laplace filter (KLF). This algorithm includes features from the EKF and features from the Laplace method. The Laplace method is an integral approximation method, which has been widely used for Bayesian estimation, see [4] for example. We first describe it in a general framework. A comprehensive presentation can be found in [9, Chapter 1].

A. The Laplace method

The Laplace method is used to compute integral in the form of

$$\int_{\mathbb{R}^d} b(x) e^{-\lambda h(x)} dx \tag{5}$$

where λ is a large positive parameter. h is supposed to admit a global minimum at $\hat{x}$ and to be regular in a neighbourhood of $\hat{x}$, so that its Jacobian matrix (size $1 \times d$) verifies $h'(\hat{x}) = (0, \ldots, 0)$ and its Hessian matrix (size $d \times d$) verifies $\det h''(\hat{x}) > 0$.

The Laplace method consists in considering integral (5) as an integral w.r.t. a Gaussian density with small variance of order $1/\lambda$. In the integrand, h is replaced by its second–order Taylor expansion at $\hat{x}$. The idea is that the contribution of the term $e^{-\lambda h(x)}$ to the integral mostly occurs around its maximum value, which is the minimum of h. Hence, the integral (5) becomes

$$e^{-\lambda h(\hat{x})} \int_{\mathbb{R}^d} b(x) e^{-\frac{\lambda}{2}(x-\hat{x})^T h''(\hat{x})(x-\hat{x})} dx.$$

When λ is large, the integral of b w.r.t. the Gaussian density

$$x \longmapsto (2\pi)^{-d/2} \det[\lambda h''(\hat{x})]^{1/2} \times \exp\left(-\frac{\lambda}{2}(x - \hat{x})^T h''(\hat{x})(x - \hat{x})\right)$$

is close to $b(\hat{x})$, since the norm of the covariance matrix $\frac{1}{\lambda} h''(\hat{x})^{-1}$ is small so that the density is concentrated around its maximum. Thus, we have that

$$\int_{\mathbb{R}^d} b(x) e^{-\frac{\lambda}{2}(x-\hat{x})^T h''(\hat{x})(x-\hat{x})} dx \approx b(\hat{x})(2\pi)^{d/2} \det[\lambda h''(\hat{x})]^{-1/2},$$

so that we can approximate the integral (5) as

$$\int_{\mathbb{R}^d} b(x) e^{-\lambda h(x)} dx \approx (2\pi)^{d/2} b(\hat{x}) e^{-\lambda h(\hat{x})} \det[\lambda h''(\hat{x})]^{-1/2}, \tag{6}$$

which is the Laplace approximation.

B. Algorithm

We now present the KLF algorithm. The KLF, as the EKF and the UKF, is applicable to state–space models, where the state process is in the form of a nonlinear autoregressive time series disturbed by a Gaussian white noise, as in Equation (1). The KLF, however, can handle more general observation models than the EKF and the UKF, which are not necessarily in the form of Equation (2) (for example discrete observation models).

For all $k \geq 1$, let us denote $\partial_x F_k$ the derivative (i.e., Jacobian matrix) of F_k w.r.t. its first argument (the state) and $\partial_v F_k$ the derivative of F_k w.r.t. its second argument (the state noise).

In the KLF, the prediction step is identical as that of the EKF. Let $\hat{x}_{k-1}$ and $\hat{P}_{k-1}$ respectively be the estimates of the hidden state and the posterior covariance matrix at time $k-1$. The mean and covariance matrix of the predictor $p_{k|k-1}$ at time k are estimated as

$$\hat{x}_{k|k-1} = F_k(\hat{x}_{k-1}, 0) \tag{7}$$

and

$$\begin{aligned} \hat{P}_{k|k-1} &= \partial_x F_k(\hat{x}_{k-1},0)^T \hat{P}_{k-1} \partial_x F_k(\hat{x}_{k-1},0) \\ &\quad + \partial_v F_k(\hat{x}_{k-1},0)^T Q_k^{-1} \partial_v F_k(\hat{x}_{k-1},0), \end{aligned} \tag{8}$$

see [8]. The predictor is approximated by the Gaussian density with mean $\hat{x}_{k|k-1}$ and covariance matrix $\hat{P}_{k|k-1}$, i.e.,

$$\begin{aligned} \hat{p}_{k|k-1}(x_k) &= (2\pi)^{-d/2} \det \hat{P}_{k|k-1}^{-1/2} \\ &\quad \times \exp\left(-\frac{1}{2}(x_k - \hat{x}_{k|k-1})^T \hat{P}_{k|k-1}^{-1}(x_k - \hat{x}_{k|k-1})\right). \end{aligned} \tag{9}$$

For the update step, the KLF uses the Laplace method. According to Equation (4), the approximation of the posterior at time k should verify

$$\hat{p}_k(x_k) \approx \frac{g_k(x_k)\hat{p}_{k|k-1}(x_k)}{\int_{\mathbb{R}^d} g_k(x_k)\hat{p}_{k|k-1}(x_k)dx_k} \tag{10}$$

for all $x_k \in \mathbb{R}^d$. We approximate the integral at the denominator thanks to the Laplace method. According to Equation (6), we have that

$$\begin{aligned} &\int_{\mathbb{R}^d} g_k(x_k)\hat{p}_{k|k-1}(x_k)dx_k \\ &\qquad \approx (2\pi)^{d/2} \det \hat{P}_k^{1/2} g_k(\hat{x}_k)\hat{p}_{k|k-1}(\hat{x}_k) \end{aligned}$$

where

$$\hat{x}_k = \operatorname*{argmax}_{x \in \mathbb{R}^d} \{g_k(x)\hat{p}_{k|k-1}(x)\}$$

and

$$\hat{P}_k = \hat{J}_k^{-1},$$

with $\hat{J}_k = -(\log g_k)''(\hat{x}_k) + \hat{P}_{k|k-1}^{-1}$. Replacing the integral at the denominator of (10) by its Laplace approximation (11), we obtain the closed–form approximation of the posterior density

$$\hat{p}_k(x_k) = (2\pi)^{-d/2} \det \hat{P}_k^{-1/2} \frac{g_k(x_k)\hat{p}_{k|k-1}(x_k)}{g_k(\hat{x}_k)\hat{p}_{k|k-1}(\hat{x}_k)} \tag{11}$$

Fig. 1: Kalman Laplace filter

{initialization} $k \leftarrow 0$
compute
  $\hat{x}_0 = \operatorname*{argmax}_{x \in \mathbb{R}^d} \{g_0(x)q_0(x)\}$
  $\hat{P}_0 = [-(\log g_0)''(\hat{x}_0) - (\log q_0)''(\hat{x}_0)]^{-1}$
loop
  {time iteration} $k \leftarrow k+1$
  {prediction} compute
    $\hat{x}_{k|k-1}$ as in Equation (7)
    $\hat{P}_{k|k-1}$ as in Equation (8)
  {update} compute
    $\hat{x}_k = \operatorname*{argmax}_{x \in \mathbb{R}^d} \{g_k(x)\hat{p}_{k|k-1}(x)\}$, where $\hat{p}_{k|k-1}$ is given in Equation (9)
    $\hat{P}_k = \left[-(\log g_k)''(\hat{x}_k) + \hat{P}_{k|k-1}^{-1}\right]^{-1}$
end loop

for all $x_k \in \mathbb{R}^d$.

We have that $\hat{p}_k \propto g_k \hat{p}_{k|k-1}$, hence $\hat{x}_k$ is an approximation of the maximum of the posterior density (MAP). Besides, $\hat{J}_k$ is an approximation of the observed information matrix evaluated at the MAP.

After the update step, $\hat{x}_k$ and $\hat{P}_k$ are propagated to time $k+1$ like in the EKF, to get a Gaussian approximation of the new predictor $\hat{p}_{k+1|k}$, which is updated thanks to the Laplace method, and so on. The KLF algorithm is described in Figure 1.

Note that the state estimate $\hat{x}_k$ retrieved by the KLF is the MAP, whereas it is the posterior mean in the Kalman filter (and EKF or UKF) and in the particle filters. The MAP computation is possible thanks to the closed–form approximation of the posterior density $\hat{p}_k$ (Equation (11)) provided by the KLF, which allows to use standard optimization routines to compute its maximum.

To get an approximation of the posterior mean $\mathbb{E}[X_k|Y_{0:k}]$, one needs to compute the integral

$$\int_{\mathbb{R}^d} x_k \hat{p}_k(x_k)dx_k = \frac{\int_{\mathbb{R}^d} x_k g_k(x_k)\hat{p}_{k|k-1}(x_k)dx_k}{\int_{\mathbb{R}^d} g_k(x_k)\hat{p}_{k|k-1}(x_k)dx_k}.$$

Since it is straightforward to sample from a Gaussian, this integral can be computed by importance sampling. Indeed, by sampling from the (Gaussian) approximated predictor and weighting according to the likelihood, one gets the approximation

$$\frac{\sum_{i=1}^N g_k(\zeta^i)\zeta^i}{\sum_{i=1}^N g_k(\zeta^i)}$$

where $\zeta^1, \ldots, \zeta^N$ is an i.i.d. sample from $\hat{p}_{k|k-1}$. However this Monte Carlo approximation is not consistent like in the SIR algorithm, i.e., it does not converge to the true posterior mean $\mathbb{E}[X_k|Y_{0:k}] = \int_{\mathbb{R}^d} x_k p_k(x_k)dx_k$, but to $\int_{\mathbb{R}^d} x_k \hat{p}_k(x_k)dx_k$, as $N \to \infty$.

Fig. 2: Laplace Gaussian filter

{initialization} $k \leftarrow 0$
compute $\hat{x}_0 = \underset{x \in \mathbb{R}^d}{\operatorname{argmax}} \{g_0(x) q_0(x)\}$
loop
{time iteration} $k \leftarrow k+1$
{state estimation} compute

$$\hat{x}_k = \underset{x \in \mathbb{R}^d}{\operatorname{argmax}} \{g_k(x) q_k(\hat{x}_{k-1}, x)\}$$

end loop

C. Comparison with the Laplace Gaussian filter

The KLF that we propose is similar to the Laplace Gaussian filter (LGF) proposed by Koyama et al. in [10]. The LGF is also a deterministic algorithm. It uses the Laplace method to compute the predictor density and the posterior moments at each time iteration. The LGF proceeds as follows. Given an approximation $\hat{x}_{k-1}$ of the hidden state at time $k-1$, the predictor is approximated as $q_k(\hat{x}_{k-1}, \cdot)$ (higher–order approximations of the form $q_k(\hat{x}_{k-1}, \cdot) + a_1 q_k'(\hat{x}_{k-1}, \cdot) + a_2 q_k''(\hat{x}_{k-1}, \cdot) + \dots$ are also possible, where $q_k(\hat{x}_{k-1}, \cdot)$ is the dominating term, see Equation (A.13) in [10]). This approximation of the predictor comes from the approximation of the integral (3) by the Laplace method. The hidden state is estimated by the MAP $\hat{x}_k = \underset{x \in \mathbb{R}^d}{\operatorname{argmax}} \{g_k(x) q_k(\hat{x}_{k-1}, x)\}$. The posterior covariance matrix can be estimated by $\hat{P}_k = [-(\log g_k)''(\hat{x}_k) - (\log q_k)''(\hat{x}_{k-1}, \hat{x}_k)]^{-1}$, where $(\log q_k)''(\hat{x}_{k-1}, \cdot)$ is the Hessian matrix of $\log q_k(\hat{x}_{k-1}, \cdot)$. As an approximation of the posterior density, the LGF retrieves the Gaussian with mean $\hat{x}_k$ and covariance matrix $\hat{P}_k$.

Hence, the differences between the LGF and the KLF are:

- the prediction step, which is performed like in the EKF in the KLF, and which is obtained thanks to the Laplace method in the LGF,
- the closed–form approximation of the posterior density, which is non-Gaussian in the KLF (see Equation (11)), and which is the Gaussian density whose moments are the approximated MAP and posterior covariance matrix in the LGF.

The common features of the LGF and the KLF are that both algorithms are deterministic, use the Laplace method, and retrieve an approximation of the MAP as a state estimate.

The LGF is described in Algorithm 2.

IV. MAP COMPUTATION

At each time iteration of the KLF, we have to maximize the (unnormalized) density $g_k \hat{p}_{k|k-1}$, where $\hat{p}_{k|k-1}$ is a Gaussian density given by Equation (9) and g_k is the likelihood. This maximization may be computationally costly, especially when the state–space dimension d is large. For example, using Newton's method requires to invert a $d \times d$ Hessian matrix at each step of the algorithm, whose cost is typically of order $O(d^3)$.

We now describe a way to facilitate the maximization, using the fact that the likelihood can often be considered as a function of an "observable" vector with smaller dimension than the whole state vector (see also [5]). To alleviate the notations, let us denote $g = g_k$ and $q = \hat{p}_{k|k-1}$ in this section.

Suppose that there exists g^{o} such that $g^{\mathrm{o}}(Ax) = g(x)$ for all $x \in \mathbb{R}^d$, where A is a $d^{\mathrm{o}} \times d$ known matrix with $d^{\mathrm{o}} < d$ (hence, Ax has a smaller dimension than x). The QR decomposition of A^T yields $A^T = QR$, where Q is a $d \times d$ unitary matrix and R is a $d \times d^{\mathrm{o}}$ matrix such that

$$R = \begin{pmatrix} R^{\mathrm{o}} \\ 0_{(d-d^{\mathrm{o}}) \times d^{\mathrm{o}}} \end{pmatrix}.$$

R^{o} is a $d^{\mathrm{o}} \times d^{\mathrm{o}}$ upper triangular matrix and $0_{(d-d^{\mathrm{o}}) \times d^{\mathrm{o}}}$ is a $(d-d^{\mathrm{o}}) \times d^{\mathrm{o}}$ matrix whose all coefficients are zero.

In the sequel, for all vector $v = (v_1, \dots, v_d)^T \in \mathbb{R}^d$, we denote $v^{\mathrm{o}} = (v_1, \dots, v_{d^{\mathrm{o}}})^T \in \mathbb{R}^{d^{\mathrm{o}}}$ the vector composed by its first d^{o} components and $v^{\mathrm{n}} = (v_{d^{\mathrm{o}}+1}, \dots, v_d)^T \in \mathbb{R}^{d-d^{\mathrm{o}}}$ the vector composed by its last $d - d^{\mathrm{o}}$ components.

For all $x \in \mathbb{R}^d$, let $q_1(x) = q(Qx)$, $g_1(x^{\mathrm{o}}) = g^{\mathrm{o}}([R^{\mathrm{o}}]^T x^{\mathrm{o}})$, and $p_1(x) = g_1(x^{\mathrm{o}}) q_1(x)$. We have that

$$\begin{aligned} p(x) &= g^{\mathrm{o}}(Ax) q(x) \\ &= g^{\mathrm{o}}(R^T Q^{-1} x) q(x) \\ &= g^{\mathrm{o}}([R^{\mathrm{o}}]^T [Q^{-1}x]^{\mathrm{o}}) q(QQ^{-1}x) \\ &= g_1([Q^{-1}x]^{\mathrm{o}}) q_1(Q^{-1}x) \\ &= p_1(Q^{-1}x) \end{aligned}$$

for all $x \in \mathbb{R}^d$.

Let now $\hat{x}_1 = \underset{x \in \mathbb{R}^d}{\operatorname{argmax}} \{p_1(x)\}$ and let $\hat{x} = Q\hat{x}_1$. Then, for all $x \in \mathbb{R}^d$,

$$p(\hat{x}) = p_1(Q^{-1}\hat{x}) = p_1(\hat{x}_1) \geq p_1(Q^{-1}x) = p(x),$$

which implies that $\hat{x} = \underset{x \in \mathbb{R}^d}{\operatorname{argmax}} \{p(x)\}$. Hence, maximizing p is equivalent to maximizing p_1.

Besides, we have that

$$\begin{aligned} \max_{x \in \mathbb{R}^d} \{p(x)\} &= \max_{x \in \mathbb{R}^d} \{p_1(x)\} \\ &= \max_{x \in \mathbb{R}^d} \{g_1(x^{\mathrm{o}}) q_1 \begin{pmatrix} x^{\mathrm{o}} \\ x^{\mathrm{n}} \end{pmatrix}\} \\ &= \max_{x^{\mathrm{o}} \in \mathbb{R}^{d^{\mathrm{o}}}} \left\{ g_1(x^{\mathrm{o}}) \max_{x^{\mathrm{n}} \in \mathbb{R}^{d-d^{\mathrm{o}}}} \{q_1 \begin{pmatrix} x^{\mathrm{o}} \\ x^{\mathrm{n}} \end{pmatrix}\} \right\} \\ &= \max_{x^{\mathrm{o}} \in \mathbb{R}^{d^{\mathrm{o}}}} \left\{ g_1(x^{\mathrm{o}}) q_1 \begin{pmatrix} x^{\mathrm{o}} \\ \hat{x}^{\mathrm{n}}(x^{\mathrm{o}}) \end{pmatrix} \right\} \end{aligned}$$

where $\hat{x}^{\mathrm{n}}(x^{\mathrm{o}}) = \underset{x^{\mathrm{n}} \in \mathbb{R}^{d-d^{\mathrm{o}}}}{\operatorname{argmax}} \{q_1 \begin{pmatrix} x^{\mathrm{o}} \\ x^{\mathrm{n}} \end{pmatrix}\}$ for all $x^{\mathrm{o}} \in \mathbb{R}^{d^{\mathrm{o}}}$. Therefore, the maximum of p_1 can be expressed as $\hat{x}_1 = \begin{pmatrix} \hat{x}^{\mathrm{o}} \\ \hat{x}^{\mathrm{n}}(\hat{x}^{\mathrm{o}}) \end{pmatrix}$ where $\hat{x}^{\mathrm{o}} = \underset{x^{\mathrm{o}} \in \mathbb{R}^{d^{\mathrm{o}}}}{\operatorname{argmax}} \left\{ g_1(x^{\mathrm{o}}) q_1 \begin{pmatrix} x^{\mathrm{o}} \\ \hat{x}^{\mathrm{n}}(x^{\mathrm{o}}) \end{pmatrix} \right\}$.

When $\hat{x}^{\mathrm{n}}(\hat{x}^{\mathrm{o}})$ can be computed exactly, the problem of maximizing p_1 over $\mathbb{R}^d$ reduces to the lower dimensional problem of maximizing $g_1(x^{\mathrm{o}}) q_1 \begin{pmatrix} x^{\mathrm{o}} \\ \hat{x}^{\mathrm{n}}(x^{\mathrm{o}}) \end{pmatrix}$ w.r.t. $x^{\mathrm{o}} \in \mathbb{R}^{d^{\mathrm{o}}}$.

This is the case when q is a Gaussian density (recall that q here plays the same role as the predictor approximation $\hat{p}_{k|k-1}$ in the KLF). Indeed, suppose that q is a Gaussian density with mean m and covariance matrix Σ. Then, q_1 is a Gaussian density with mean $Q^T m = m_1 = \begin{pmatrix} m_1^{\mathrm{o}} \\ m_1^{\mathrm{n}} \end{pmatrix}$ (with $m_1^{\mathrm{o}} \in \mathbb{R}^{d^{\mathrm{o}}}$ and $m_1^{\mathrm{n}} \in \mathbb{R}^{d-d^{\mathrm{o}}}$) and covariance matrix $Q^T \Sigma Q = \Sigma_1 = \begin{pmatrix} \Sigma_1^{\mathrm{o}} & \Sigma_1^{\mathrm{on}} \\ \Sigma_1^{\mathrm{no}} & \Sigma_1^{\mathrm{n}} \end{pmatrix}$ (with Σ_1^{o} a $d^{\mathrm{o}} \times d^{\mathrm{o}}$ matrix, Σ_1^{n} a $(d-d^{\mathrm{o}}) \times (d-d^{\mathrm{o}})$ matrix, and $\Sigma_1^{\mathrm{on}} = [\Sigma_1^{\mathrm{no}}]^T$ a $d \times (d-d^{\mathrm{o}})$ matrix). Hence, $\hat{x}^{\mathrm{n}}(x^{\mathrm{o}}) = \underset{x^{\mathrm{n}} \in \mathbb{R}^{d-d^{\mathrm{o}}}}{\operatorname{argmax}} \{q_1 \begin{pmatrix} x^{\mathrm{o}} \\ x^{\mathrm{n}} \end{pmatrix}\}$ is the conditional mean $\mathbb{E}[X^{\mathrm{n}}|X^{\mathrm{o}}]$, where $X^{\mathrm{o}} \sim \mathcal{N}(m_1^{\mathrm{o}}, \Sigma_1^{\mathrm{o}})$, $X^{\mathrm{n}} \sim \mathcal{N}(m_1^{\mathrm{n}}, \Sigma_1^{\mathrm{n}})$, and $\begin{pmatrix} X^{\mathrm{o}} \\ X^{\mathrm{n}} \end{pmatrix} \sim \mathcal{N}(m_1, \Sigma_1)$. This conditional mean can be computed exactly and equals

$$\hat{x}^{\mathrm{n}}(x^{\mathrm{o}}) = m_1^{\mathrm{n}} + \Sigma_1^{\mathrm{no}} [\Sigma_1^{\mathrm{o}}]^{-1} (x^{\mathrm{o}} - m_1^{\mathrm{o}}). \tag{12}$$

Thus, when the likelihood is a function of a d^{o}–dimensional vector, which depends linearly on the d–dimensional state vector ($d^{\mathrm{o}} < d$), the method we describe in this section can be used in the KLF to compute the MAP. The maximization over $\mathbb{R}^d$ is replaced by a maximization over $\mathbb{R}^{d^{\mathrm{o}}}$, so that the MAP computation cost is significantly alleviated. It is the case in the first simulation in Section V below.

Note that this method can also be used in the LGF if the approximated predictor density $q_k(\hat{x}_{k-1}, \cdot)$ is Gaussian (see Figure 2). This is the case when the state dynamics is in the form of $X_k = F_k(X_{k-1}) + V_k$, where $\{V_k\}_{k \geq 1}$ is a Gaussian white noise.

This idea to simplify the MAP computation has been previously proposed in [5]. Here, we generalize the method to the case when the likelihood is a function of a linear mapping of the state vector that reduces its dimension. Besides, our derivation fully uses the fact the predictor is Gaussian and yields to the formula in Equation (12), which is different than the formula in Equation (20) in [5, Section V.B]. Note that our method is computationally cheaper: it uses the readily available block decomposition of the covariance matrix Σ_1 itself, and only requires the inversion of a block with dimension d^{o} (the observed component of the state vector has often a smaller dimension than its non-observed component, i.e., $d^{\mathrm{o}} < d - d^{\mathrm{o}}$).

To solve the d^{o}–dimensional maximization problem, one usually has to rely on numerical methods. In the special case where the observation model (2) takes the form

$$Y_k = H_k(X_k^{\mathrm{o}}) + W_k,$$

i.e., in the case of an additive observation noise, then the maximization problem is equivalent to the minimization problem

$$\min_{x^{\mathrm{o}} \in \mathbb{R}^{d^{\mathrm{o}}}} \{(x^{\mathrm{o}} - \hat{x}_{k|k-1}^{\mathrm{o}})^T [\hat{P}_{k|k-1}^{\mathrm{o}}]^{-1} (x^{\mathrm{o}} - \hat{x}_{k|k-1}^{\mathrm{o}}) + (Y_k - H_k(x^{\mathrm{o}}))^T R_k^{-1} (Y_k - H_k(x^{\mathrm{o}}))\}.$$

This minimization problem can be solved numerically using a Gauss–Newton method, which in practice [11] reduces to the iterations

$$x_{i+1} = \hat{x}_{k|k-1}^{\mathrm{o}} + K_i (Y_k - H_k(x_i) - H_k'(x_i)(\hat{x}_{k|k-1}^{\mathrm{o}} - x_i)),$$

of an iterated Kalman filter, with the Kalman gain matrix defined as

$$\begin{aligned} K_i &= [H_k'(x_i)^T R_k^{-1} H_k'(x_i) + (\hat{P}_{k|k-1}^{\mathrm{o}})^{-1}]^{-1} H_k'(x_i)^T R_k^{-1} \\ &= \hat{P}_{k|k-1}^{\mathrm{o}} H_k'(x_i)^T [H_k'(x_i) \hat{P}_{k|k-1}^{\mathrm{o}} H_k'(x_i)^T + R_k]^{-1}, \end{aligned}$$

and with the initial condition defined as $x_0 = \hat{P}_{k|k-1}^{\mathrm{o}}$ for $i = 0$.

V. Simulation experiments

A. Performance assessment and simulation scheme

In our experiments, we do not use real observations and we do not have a ground truth giving the true value of the hidden state to estimate. The observations are instead generated from the observation process, and the true state is by definition the state value used for this generation, i.e., $Y_k \sim g_k(x_k^*)$ for all $k \in \{0, \ldots, n\}$, where $x_{0:n}^*$ is the true state sequence. n denotes the number of time iterations in the simulation scenario.

The performance of the filtering algorithms that we compare is assessed in terms of the evolution in time of the root mean squared error (RMSE) between the state estimate and the true state. The RMSE is defined by[3] $\mathbb{E}[|x_k^* - \hat{x}_k|^2]^{1/2}$ for all $k \geq 0$. To empirically estimate this quantity, we run independent simulations and we compute $\sqrt{\frac{1}{R} \sum_{r=1}^{R} |x_k^{*r} - \hat{x}_k^r|^2}$, where R is the number of runs. Each run corresponds to an observation sequence $Y_{0:n}^r$ and to a true state sequence $x_{0:n}^{*r}$. The observation sequences are independently generated as $Y_{1:n}^r \sim \prod_{k=0}^{n} g_k(x_k^{*r})$ for all $r \in \{1, \ldots, R\}$.

Besides, the true state sequences are independently generated using the state process, i.e., for all $r \in \{1, \ldots, R\}$, $x_0^{*r} \sim q_0$ and $x_k^{*r} \sim q_k(x_{k-1}^{*r}, \cdot)$ for all $k \geq 1$.

In the three nonlinear Bayesian filtering problems we address below, the number of simulation runs is set to $R = 100$.

The simulations have been performed in MATLAB. The code is available upon request to the first author.

B. Target tracking

The first filtering problem we consider is a simple target tracking problem. The azimuth of a moving target is measured by two sensors. The goal is to estimate the position and the velocity of the target thanks to these angle observations.

The hidden state X_k at each time $k \geq 0$ is defined as

$$X_k = \begin{pmatrix} X_{k,1} \\ \dot{X}_{k,1} \\ X_{k,2} \\ \dot{X}_{k,2} \end{pmatrix},$$

where $(X_{k,1}, X_{k,2})$ is the target position and $(\dot{X}_{k,1}, \dot{X}_{k,2})$ is the target velocity in an adapted Cartesian coordinates system. The state process describing the target motion is linear with additive Gaussian white noise,

$$X_k = F X_{k-1} + V_k \tag{13}$$

[3] $|\cdot|$ denotes the Euclidean norm in $\mathbb{R}^d$.

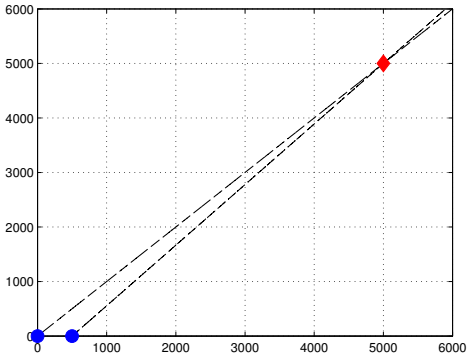


Fig. 3: Sensor position (blue dots) and initial mean of the target position (red diamond).

for all $k \geq 1$, where

$$F = \begin{pmatrix} 1 & \Delta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{14}$$

Δ is the sampling period. For all $k \geq 1$, $V_k \sim \mathcal{N}(0, Q)$, where

$$Q = \sigma^2 \begin{pmatrix} \frac{\Delta^3}{3} & \frac{\Delta^2}{2} & 0 & 0 \\ \frac{\Delta^2}{2} & \Delta & 0 & 0 \\ 0 & 0 & \frac{\Delta^3}{3} & \frac{\Delta^2}{2} \\ 0 & 0 & \frac{\Delta^2}{2} & \Delta \end{pmatrix}, \tag{15}$$

see [12]. The initial state distribution is $X_0 \sim \mathcal{N}(m_0, Q_0)$.

The observations are a vectors composed of two angles measured by two sensors. The positions of the sensors are $s_1 = (s_{1,1}, s_{1,2})$ and $s_2 = (s_{2,1}, s_{2,2})$. The observation at time $k \geq 0$ is given by

$$Y_k = \begin{pmatrix} \arctan \dfrac{X_{k,2} - s_{1,2}}{X_{k,1} - s_{1,1}} \\ \arctan \dfrac{X_{k,2} - s_{2,2}}{X_{k,1} - s_{2,1}} \end{pmatrix} + W_k$$

where $\{W_k\}_{k \geq 0}$ is a Gaussian white noise, with $W_k \sim \mathcal{N}(0, \rho^2 I_2)$ for all $k \geq 0$.

The sampling period is set to $\Delta = 0.5$s and the number of time iterations to $n = 120$, so that the duration of the tracking scenario is $n\Delta = 60$s. The initial state mean is set to $m_0 = (5000, 15/\sqrt{2}, 5000, 15/\sqrt{2})^T$ and the initial covariance matrix to $Q_0 = \mathrm{diag}(1000^2, 10^2, 1000^2, 10^2)$ (the distance unit is meter and the speed unit is meter per second). The parameter of the covariance matrix of the state dynamics noise is set to $\sigma = 1$. The sensor positions are $s_1 = (0, 0)$ and $s_1 = (0, 500)$. The parameter of the covariance matrix of the observation noise is set to $\rho = \pi/180$ (1 degree) or to $\rho = 0.1\pi/180$ (0.1 degree).

The configuration of the tracking scenario at initial time $k = 0$ is displayed in Figure 3.

We compare four algorithms: the EKF, the SIR algorithm, the LGF (Figure 2), and the KLF (Figure 1). For the SIR algorithm, the number of particles is $N = 1000$ or $N = 10000$.

Note that the likelihood g_k does not depend on the whole 4–dimensional state vector x_k (which includes the position and the velocity), but only on the 2–dimensional position. Hence, we can use the technique described in Section IV to alleviate the cost of the MAP computation in the LGF and the KLF at each time iteration.

The evolutions of the RMSE are displayed in Figure 4a when the observation noise standard deviation ρ equals 1 degree. The SIR algorithm yields the best performance with $N = 10000$, although the EKF becomes better at the end of the scenario. The LGF is inefficient here, as its error increases over time. The error of the KLF decreases over time. The KLF yields a lower error than the SIR algorithm with $N = 1000$, but it is outperformed by the SIR algorithm with $N = 10000$ and by the EKF.

Figure 4b displays the estimation performance when the observation noise standard deviation is reduced to 0.1 degree. In this case, the KLF achieves the best performance throughout the scenario. The LGF remains inefficient.

C. Population dynamics monitoring

The second filtering problem is the monitoring of a population dynamics thanks to periodical surveys. For example, we aim at estimating the population size of a wild animal living in a large area given aerial counts, which are performed every month or year. This problem is described in [13].

The hidden state X_k at time $k \geq 0$ is the population size. The initial state distribution is log–normal, i.e., $e^{X_0} \sim \mathcal{N}(m_0, \sigma_0^2)$. The state process modelling the population dynamics is

$$X_k = X_{k-1}^{\alpha} e^{\beta + \sigma V_k}$$

for $k \geq 1$, where $\{V_k\}_{k \geq 0}$ is a Gaussian standardized white noise.

The observation at time $k \geq 0$ is a survey count of the population. The observation model is Poisson, where the intensity parameter is the hidden state, i.e.,

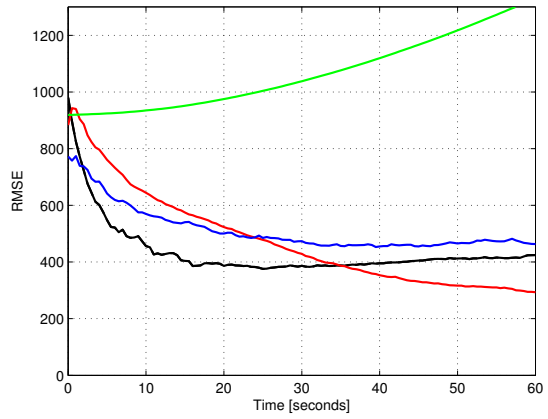
$$Y_k \sim \mathcal{P}(X_k),$$

so that the likelihood verifies $g_k(x_k) = e^{-x_k} \frac{x_k^{Y_k}}{Y_k!}$ for all $x_k \in \mathbb{R}$.

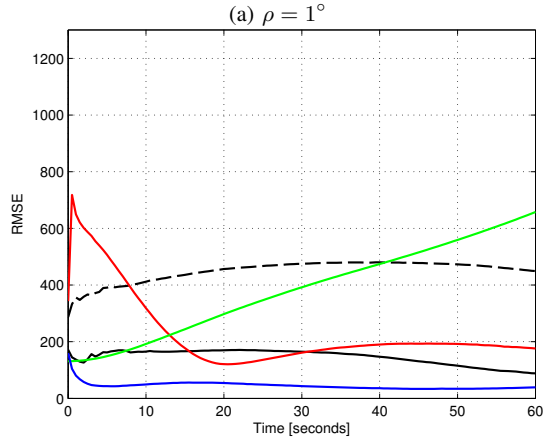
The model parameters are set like in [13]: $m_0 = 5$, $\sigma_0 = \sqrt{10}$, $\alpha = 1.6$, $\beta = 0.75$, $\sigma = 0.28$, $\rho = 0.25$; the number of time iterations is set to $n = 40$.

We compare the performance of three filtering algorithms: the SIR algorithm, the LGF, and the KLF. For the SIR algorithm, we use $N = 10000$ particles. The EKF or the UKF cannot be used here because the observations are discrete. The LGF and the KLF however can handle this type of model.

The evolution of the RMSE is displayed in Figure 5. Here, the three algorithm exhibit the same behaviour. Their errors are very close and decrease over time.



(a) $\rho = 1^\circ$



(b) $\rho = 0.1^\circ$

Fig. 4: Evolution of the estimation error. EKF: red line, SIR with $N = 1000$: black dashed line, SIR with $N = 10000$: black line, LGF: green line, KLF: blue line.

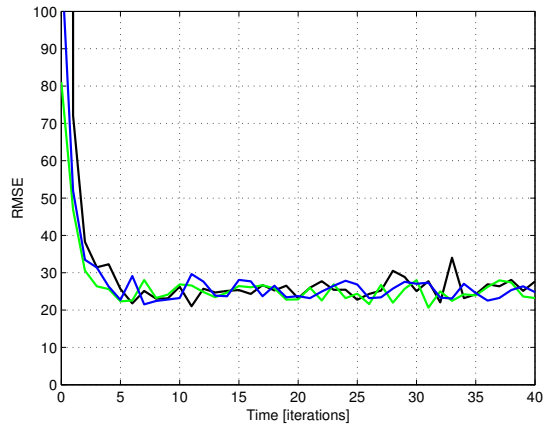


Fig. 5: Evolution of the estimation error. SIR with $N = 10000$: black line, LGF: green line, KLF: blue line.

D. Neural decoding

The third filtering problem we consider is called neural decoding and arises in neurosciences. It consists in reconstructing sensory or motion information from neural spike observations recorded in the brain. These observations come from the monitoring of the electrical activity of a set of firing neurons. In particular, the problem of estimating the motor activity of an animal's body with spike observations recorded in the motor cortex can be addressed as a Bayesian filtering problem [14], [10].

Consider that an animal (for example a monkey) can move its hand on a plane. We define the hidden state X_k at each time $k \geq 0$ as $X_k = (X_{k,1}, \dot{X}_{k,1}, X_{k,2}, \dot{X}_{k,2})^T$, where $(X_{k,1}, X_{k,2})$ is the hand position and $(\dot{X}_{k,1}, \dot{X}_{k,2})$ is the hand velocity in an adapted Cartesian coordinates system. The state process describing the hand motion is exactly the same as in the target tracking problem in Section V-B above. The state dynamics is given by (13) and (14), the state noise covariance matrix is given by (15), and the initial state distribution is $X_0 \sim \mathcal{N}(m_0, Q_0)$.

The observations are recorded by monitoring the electrical activity record of a set of M neurons. Each neuron $j \in \{1, \ldots, M\}$ fires $Y_{k,j}$ times during the time interval $[k\Delta, (k+1)\Delta]$ for all $k \geq 0$. The observation at time k is the collection of all the neural spike counts $Y_k = \{Y_{k,j}\}_{j \in \{1,\ldots,M\}}$. Hence, Y_k takes values in $\mathbb{N}^M$. For all $j \in \{1, \ldots, M\}$ and $k \geq 0$, the distribution of $Y_{k,j}$ is Poisson with a intensity parameter depending on the hidden state X_k, i.e.,

$$Y_{k,j} \sim \mathcal{P}(\lambda_j(X_k)\Delta).$$

The activity of each neuron is supposed to be independent. The likelihood associated with observation Y_k is then defined by $g_k(x_k) = \prod_{j=1}^{M} g_{k,j}(x_k)$ for all $x_k \in \mathbb{R}^4$, where $g_{k,j}(x_k) = e^{-\lambda_j(x)\Delta} \frac{(\lambda_j(x_k)\Delta)^{Y_{k,j}}}{Y_{k,j}!}$ is the likelihood associated with the observation $Y_{k,j}$ from neuron j. The relation between the Poisson intensity parameter and the hidden state is

$$\lambda_j(x_k) = \exp\left(\alpha + \beta_j^T x_k\right)$$

for all $x_k \in \mathbb{R}^4$.

This state–space model, adapted to neural decoding, is the similar to the one used in [10].

The number of monitored neurons is set to $M = 100$. The length of the time bins is $\Delta = 30 \cdot 10^{-3}$ seconds and the number of time iterations is $n = 100$, so that the duration of the scenario is $n\Delta = 3$ seconds. The state process parameters are set to $m_0 = (0, 0, 0, 0)^T$, $Q_0 = \mathrm{diag}(0.1^2, 0.2^2, 0.1^1, 0.2^2)$, $\sigma = 0.1$. The observation model parameters are set to $\alpha = 3$ and $\beta_j = (0, j/M, 0, 1 - j/M)^T$ for all $j \in \{1, \ldots, M\}$. Thus, the β_j's are regularly positioned on a unit circle and the firing intensity uniquely depends on the velocity of the animal's hand (not on its position on the plane).

We compare three algorithm: the SIR algorithm, the LGF, and the KLF. We use $N = 10000$ for the SIR algorithm. Like in the population dynamics monitoring problem in Section

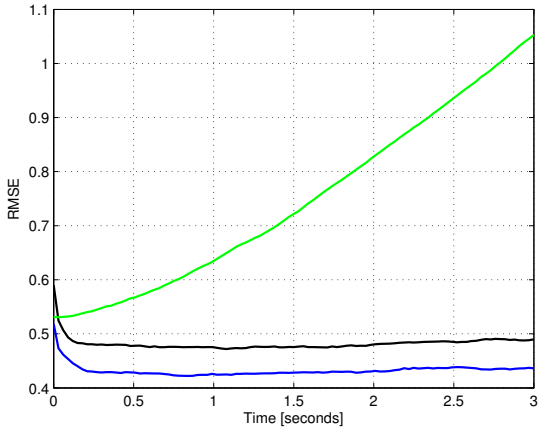


Fig. 6: Evolution of the estimation error. SIR with $N = 10000$: black line, LGF: green line, KLF: blue line.

V-C, the observations are discrete here, hence the EKF and the UKF cannot be used.

The likelihood g_k does not depend on the whole 4–dimensional state vector x_k but only on the 2–dimensional velocity. Thus, as in the target tracking problem in Section V-B, we can use the technique described in Section IV for the MAP computation in the KLF and the LGF.

Figure 5 shows the evolution of the RMSE for the three algorithms. The LGF is inefficient here since its error increases over time. The errors of the SIR algorithm and the KLF decrease. The KLF is more efficient than the SIR algorithm, as its error is the lowest throughout the scenario.

VI. Conclusion

We propose in this paper a new algorithm, the Kalman Laplace filter (KLF), for nonlinear Bayesian filtering. This algorithm is deterministic (no Monte Carlo sampling). In the KLF, the prediction is performed like in the extended Kalman filter, i.e., by linearizing the state dynamics. This yields a Gaussian approximation of the predictor. Then, the update is done thanks to the Laplace method. We obtain a non–Gaussian closed–form approximation of the posterior density. The state is estimated by the MAP and its covariance matrix is estimated by the inverse of the observation information matrix (i.e., the Hessian of the posterior log–density) evaluated at the MAP.

The KLF can handle more general state–space models than the EKF or the UKF, for example models where the observations are discrete. The KLF share similarities with the previously proposed Laplace Gaussian filter (LGF) [10]. The main differences are the prediction step and the closed–form approximation of the posterior density.

The KLF (as the LGF) involves the maximization of the posterior density at each time iteration. We describe a way to alleviate the computation cost of this maximization in the KLF. We use the fact that the likelihood often depends on a "observable" vector of smaller dimension than the state vector (in target tracking models for example), and that the posterior is approximated as a Gaussian. This allows to perform optimization over a subspace of smaller dimension than the entire state space. Since the computation cost of optimization routines typically depends on the space dimension, this method reduces the time devoted to the MAP computation in the KLF.

We have performed three simulation experiments involving nonlinear state–space models, which illustrate the good performance of the KLF. In particular, the KLF is very efficient in the target tracking simulation with a small observation noise. This may be related to the fact that the Laplace method yields consistent approximations as the precision of the observations increases (or equivalently, when the observation sample size increases in static models) [9, Chapter 4].

A theoretical study of the KLF in an adapted asymptotic regime remains to be done, like the asymptotic study of the LGF in [10]. Such a regime could be the state dynamics noise and observation noise jointly tending to zero, or the state dynamics noise tending to zero and the number of observations (i.e., the number of time iterations) tending to infinity.

References

[1] R.E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *J. of Basic Engineering*, vol. 82, 35–45, 1960.

[2] S.J. Julier and J.K. Uhlmann. New extension of the Kalman filter to nonlinear systems. In *Proc. of SPIE Conf. on Signal Processing, Sensor Fusion, and Target Recognition*, 1997.

[3] N.J. Gordon, D.J. Salmond, and A.F.M. Smith. Novel approach to nonlinear/non-Gaussian Bayesian state estimation. *IEE Proc. F (Radar and Signal Processing)*, vol. 140, 107–113, 1993.

[4] R. Kass, L. Tierney, and J. Kadane. Laplace's method in Bayesian analysis. *Contemporary Mathematics*, vol. 115, 89–99, 1991.

[5] M. Fatemi, L. Svensson, L. Hammarstrand, and M. Morelande. A Study of MAP Estimation Techniques for Nonlinear Filtering. In *Proc. of the 15th Intl Conf. on Information Fusion*, 2012.

[6] O. Cappé, E. Moulines, and T. Ryden. *Inference in hidden Markov models*, Springer, 2005.

[7] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. on Signal Processing*, vol. 50, 174–188, 2002.

[8] D. Simon. *Optimal State Estimation: Kalman, H_∞, and Nonlinear Approaches*. Wiley, 2006

[9] P. Bui Quang. Particle approximation and the Laplace method for Bayesian filtering. PhD thesis, Université de Rennes 1. Available online: http://www.theses.fr/2013REN1S038

[10] S. Koyama, L. Castellanos Pérez–Bolde, C.R. Shalizi, and R. Kass. Approximate Methods for State-Space Models. *J. of the American Statistical Association*, vol. 105, 170–180, 2010.

[11] B.M. Bell and F.W. Cathey. The iterated Kalman filter update as a Gauss–Newton method. *IEEE Trans. on Automatic Control*, vol. 38, 294–297, 1993.

[12] X. Rong Li and V.P. Jilkov. A Survey of Maneuvering Target Tracking: Dynamic Models. In *Proc. of SPIE Conf. on Signal and Data Processing of Small Targets*, 2000.

[13] J. Knape, N. Jonzén, and M. Sköld. On observation distributions for state space models of population survey data. *J. of Animal Ecology*, vol. 80, 1269–1277, 2011.

[14] A.E. Brockwell, R. Kass, and A.B. Schwartz. Statistical Signal Processing and the Motor Cortex. *Proc. of the IEEE*, vol. 95, 881–898, 2007.