Time Series Filtering, Smoothing and Learning using the Kernel Kalman Filter

Liva Ralaivola
LIF UMR 6166 CNRS – Universié de Provence
CMI, 39, rue F. Joliot Curie
F-13453 Marseille Cedex 13, France
email: liva@lif.univ-mrs.fr

Florence d'Alché-Buc
LaMI UMR 8042 CNRS – Université Evry Val d'Essonne
523 Place des Terrasses
F-91000 Évry, France
email: dalche@lami.univ-evry.fr

***Abstract*— In this paper, we propose a new model, the Kernel Kalman Filter, to perform various nonlinear time series processing. This model is based on the use of Mercer kernel functions in the framework of the Kalman Filter or Linear Dynamical Systems. Thanks to the kernel trick, all the equations involved in our model to perform filtering, smoothing and learning tasks, only require matrix algebra calculus whilst providing the ability to model complex time series. In particular, it is possible to learn dynamics from some nonlinear noisy time series implementing an exact Expectation–Maximization procedure.**

I. Introduction

Time series modeling has become an appealing field for machine learning approaches over the last decade and in order to deal with data taking the form of non linear time series, there is a need of transversal and flexible tools that can be engineered easily. In this paper, we propose a kernel-based approach to time series modeling enabling the implementation of prediction, denoising and learning tasks. On the one hand, our method presents the advantage of keeping the framework of linear dynamical systems usable, and, on the other hand, the processing of non vectorial time series can be considered.

The paper is organized as follows. In section II, we describe our Kernel Kalman Filter (KKF) model. Section III focuses on a few works which are closely related to ours, while section IV reports empirical results achieved by our model on various times series processing tasks. Eventually, section V gives hints about future developments of KKF.

II. Kalman Filtering with Kernels

A. Kalman Filter Model

The Kalman filter is based on the idea that a dynamical process is *hidden* and can only be *observed* or *measured* through some time series. The dependency between two consecutive *states* of the process is assumed linear as well as the dependency between the *measurements* and the state process.

Let $\mathbf{x}_{1:T} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ be a sequence of T observation in $\mathbb{R}^d$. The Kalman filter model [1] assumes the following model for the series $\mathbf{x}_{1:T}$:

$$\mathbf{s}_{t+1} = A\mathbf{s}_t + \boldsymbol{\mu}_s + \boldsymbol{\nu}_s \quad (1a)$$

$$\mathbf{x}_t = C\mathbf{s}_t + \boldsymbol{\mu}_x + \boldsymbol{\nu}_x \quad (1b)$$

where $\boldsymbol{\nu}_s$ and $\boldsymbol{\nu}_x$ are zero-mean gaussian noise vectors of variances σ_s^2 and σ_x^2, respectively; if n is the dimension of $\mathbf{s}_t$, A is a $n \times n$ matrix, $\boldsymbol{\mu}_s$ an n-dimensional vector, C a $d \times n$ matrix and $\boldsymbol{\mu}_x$ a d-dimensional vector. The hidden state process is modeled by (1a) while the observation process corresponds to (1b). It can be noticed that the studied series must intrinsically have linear dynamics to be accurately modelled by (1).

Two kinds of tasks are associated with the model (1). The first one encompasses *filtering* and *smoothing* and it deals with the *estimation* of the state vector $\mathbf{s}_t$ when the parameters $\boldsymbol{\theta} = \{A, \boldsymbol{\mu}_s, \sigma_s^2, \boldsymbol{\mu}_1, \sigma_1^2, C, \boldsymbol{\mu}_x, \sigma_x^2\}$ are known and a series $\mathbf{x}_{1:T}$ is observed. The second issue deals with the situation where the parameters $\boldsymbol{\theta}$ of the model have to be *learned*: this task is classically addressed using an *Expectation-Maximization* (EM) [2] algorithm.

A set of *Kalman Filter* and *Kalman Smoother* equations is available to perform the estimation of the states. The filtering equations provide an estimate $\mathbf{s}^t(t)$ of $\mathbf{s}_t$ from the observations $\mathbf{x}_1, \ldots, \mathbf{x}_t$. Setting $\mathbf{s}^0(1) = \boldsymbol{\mu}_1$ and $\Sigma^0(1) = \sigma_1^2 I$, they can be written (see e.g. [3]):

$$\mathbf{s}^{t-1}(t) = A\mathbf{s}^{t-1}(t-1) + \boldsymbol{\mu}_s \quad (2)$$

$$\Sigma^{t-1}(t) = A\Sigma^{t-1}(t-1)A' + \sigma_s^2 I \quad (3)$$

$$\Sigma_e(t) = C\Sigma^{t-1}(t)C' + \sigma_x^2 I \quad (4)$$

$$G_t = \Sigma^{t-1}(t)C'\Sigma_e^{-1}(t) \quad (5)$$

$$\mathbf{e}_t = \mathbf{x}_t - C\mathbf{s}^{t-1}(t) - \boldsymbol{\mu}_x \quad (6)$$

$$\mathbf{s}^t(t) = \mathbf{s}^{t-1}(t) + G_t\mathbf{e}_t \quad (7)$$

$$\Sigma^t(t) = \Sigma^{t-1}(t) - G_tC\Sigma^{t-1}(t) \quad (8)$$

The Kalman Smoother computes an estimate $\hat{\mathbf{s}}(t)$ of $\mathbf{s}_t$ from the entire observation sequence $\mathbf{x}_1, \ldots, \mathbf{x}_T$. It requires the filtering procedure to be performed and it implements the following set of equations:

$$J_{t-1} = \Sigma^{t-1}(t-1)A'(\Sigma^{t-1}(t-1))^{-1} \quad (9)$$

$$\varsigma(t-1) = \mathbf{s}^{t-1}(t-1) + J_{t-1}(\varsigma(t) - \mathbf{s}^{t-1}(t)) \quad (10)$$

$$\Gamma(t-1) = \Sigma^{t-1}(t-1) + J_{t-1}(\Gamma(t) - \Sigma^{t-1}(t))J_{t-1}' \quad (11)$$

$$\Gamma^{t-1}(t) = \Gamma(t)J_{t-1}' \quad (12)$$

starting with $\varsigma(T) = \mathbf{s}^{(T)}(T)$ and $\Gamma(T) = \Sigma^{(T)}(T)$.

0-7803-9048-2/05/$20.00 ©2005 IEEE

B. Kalman Filtering with Kernels

1) Proposed Model: In order to tackle nonlinear time series processing, we propose a kernelized version of the Kalman filter. To do so, instead of directly working with the nonlinear $\mathbf{x}_{1:T} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$, we focus on the associated series $\mathbf{x}^\phi_{1:T} = \{\mathbf{x}^\phi_1, \ldots, \mathbf{x}^\phi_T\}$, where $\mathbf{x}^\phi_t$ stands for $\phi(\mathbf{x}_t)$. The nonlinear mapping $\phi : \mathbb{R}^d \to \mathcal{H}$ is assumed to map vectors from the *input space* $\mathbb{R}^d$ to a so-called *feature space* $\mathcal{H}$, where the inner product $\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$ can be evaluated thanks to a Mercel kernel function k such that: $\langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle = k(\mathbf{x}, \mathbf{y})$ [4]. The generative model we propose to study is the following:

$$\mathbf{s}^\phi_{t+1} = A^\phi \mathbf{s}^\phi_t + \boldsymbol{\mu}^\phi_s + \boldsymbol{\nu}^\phi_s \tag{13a}$$

$$\mathbf{x}^\phi_t = \mathbf{s}^\phi_t + \boldsymbol{\mu}^\phi_x + \boldsymbol{\nu}^\phi_x \tag{13b}$$

where all involved vectors are in $\mathcal{H}$ and where A^ϕ is a matrix of appropriate size to be multiplied by a vector of $\mathcal{H}$; $\boldsymbol{\nu}^\phi_s$ and $\boldsymbol{\nu}^\phi_x$ are zero-mean gaussian noise vectors of covariances $\sigma^2_s I$ and $\sigma^2_x I$, respectively. Actually, these noise vectors are assumed to live in the space $\mathcal{H}_x$ spanned by the vectors $\mathbf{x}^\phi_1, \ldots, \mathbf{x}^\phi_T$ and I is the identity matrix in this space. Although this might seem to be very strong an assumption, the efficiency of Kernel PCA [5] for denoising shows that such a noise model can cover a wide range of noise distributions in the input space. This model is a generalization of the model proposed by [6] (for which $\boldsymbol{\mu}^\phi_x = 0$ and $\sigma^2_x = 0$).

As for all kernel methods, the key idea behind the model (13) is that a linear model defined in the feature space corresponds to a nonlinear one in the input space. The ability to fully exploit the model (13) should thus be connected to the ability of addressing some nonlinear time series processing. Two different kernels could have been used in (13a) and (13b); resorting to one single kernel for the state and the observation processes nevertheless proves to be general enough (see results) while making it possible to generalize the classical Kalman Filter procedures to the kernel framework, and we only investigated this strategy.

2) Kernel Filtering and Smoothing: Assume (see next subsection) that the parameters A^ϕ, $\boldsymbol{\mu}^\phi_s$, $\boldsymbol{\mu}^\phi_1$ and $\boldsymbol{\mu}^\phi_x$ are of the form $B\tilde{A}B'$, $B\tilde{\boldsymbol{\mu}}_s$, $B\tilde{\boldsymbol{\mu}}_1$ and $B\tilde{\boldsymbol{\mu}}_x$, respectively, where $B = [\mathbf{b}_1 \cdots \mathbf{b}_\ell]$ is an orthonormal basis of the space $\mathcal{H}_x$ (thus of dimension ℓ). Such a basis can be obtained from the principal axes of a Kernel PCA [5] of $\mathbf{x}^\phi_1, \ldots, \mathbf{x}^\phi_T$ having strictly positive eigenvalues, $\mathbf{x}^\phi_{1:T} = \{\mathbf{x}^\phi_1, \ldots, \mathbf{x}^\phi_T\}$ being the considered training series.

If a noisy time series $\mathbf{o}_{1:\Delta} = \{\mathbf{o}_1, \ldots, \mathbf{o}_\Delta\}$ assumed to be generated from the same model as $\mathbf{x}_{1:T}$ is observed, it is possible to perform the filtering and the smoothing procedures, extending (2)–(12) to obtain the set of *kernel filtering* and *kernel smoothing* equations thanks to some matrix algebra (the details are omitted for sake of clarity). Introducing $(\beta_t)_{t\geq 1}$, $(\alpha_t)_{t\geq 1}$ and $(\lambda_t)_{t\geq 1}$

$$\beta_1 = \sigma^2_1, \quad \text{and} \quad \beta_t = \sigma^2_x, \quad t = 2, \ldots, \Delta$$

$$\alpha_t = \frac{\beta_t}{\sigma^2_x + \beta_t}, \quad \lambda_t = (1 - \alpha_t)\beta_t, \quad t = 1, \ldots, \Delta$$

the kernel filtering equations are, for $t = 1, \ldots, \Delta$:

$$\tilde{\mathbf{s}}^{t-1}(t) = \tilde{A}\left[\alpha_{t-1} B'\phi(\mathbf{o}_{t-1}) + \tilde{\mathbf{s}}^{t-1}(t-1)\right] + \tilde{\boldsymbol{\mu}}_s$$

$$\tilde{\Sigma}^{t-1}(t) = \tilde{A}\left[\lambda_{t-1} I + \tilde{\Sigma}^{t-1}(t-1)\right]\tilde{A}'$$

$$\tilde{G}_t = (1 - \alpha_t)\left[(\sigma^2_x + \beta_t)I + \tilde{\Sigma}^{t-1}(t)\right]^{-1}\tilde{\Sigma}^{t-1}(t)$$

$$\tilde{\mathbf{s}}^t(t) = \tilde{G}_t B'\phi(\mathbf{o}_t) + \left[(1 - \alpha_t)I - \tilde{G}_t\right](\tilde{s}^{t-1}(t) + \tilde{\boldsymbol{\mu}}_x) - \tilde{\boldsymbol{\mu}}_x$$

$$\tilde{\Sigma}^t(t) = -\beta_t \tilde{G}_t + \left[(1 - \alpha_t)I - \tilde{G}_t\right]\tilde{\Sigma}^{t-1}(t)$$

with $\tilde{\mathbf{s}}^0(1) = \tilde{\boldsymbol{\mu}}_1$ and $\tilde{\Sigma}^0(1) = 0$. $B'\phi(\mathbf{o}_t)$ is an ℓ-dimensional vector whose computation only depends on kernel evaluations. From these equations and starting with $\tilde{\boldsymbol{\varsigma}}(\Delta) = \tilde{\mathbf{s}}^\Delta(\Delta)$ and $\tilde{\Gamma}(\Delta) = \tilde{\Sigma}^\Delta(\Delta)$, the kernel smoothing equations follow readily for $t = \Delta, \ldots, 2$:

$$H_{t-1} = \lambda_{t-1} I + \tilde{\Sigma}^{t-1}(t-1)$$

$$\tilde{J}_{t-1} = \frac{1}{\lambda_{t-1}} H_{t-1}\tilde{A}'\left[I - H^{-1}_{t-1}\tilde{\Sigma}^{t-1}(t-1)\right]$$

$$\tilde{\boldsymbol{\varsigma}}(t-1) = \tilde{\mathbf{s}}^{t-1}(t-1) + \tilde{J}_{t-1}\left[\alpha_t B'\phi(\mathbf{o}_t) + \tilde{\boldsymbol{\varsigma}}(t) - \tilde{\mathbf{s}}^{t-1}(t)\right]$$

$$\tilde{\Gamma}(t-1) = \tilde{\Sigma}^{t-1}(t-1) - \tilde{J}_{t-1}\left[\alpha_t\beta_t I + \tilde{\Sigma}^{t-1}(t) - \tilde{\Gamma}(t)\right]\tilde{J}'_{t-1}$$

$$\tilde{\Gamma}^{t-1}(t) = \left[\lambda_t I + \tilde{\Gamma}(t)\right]\tilde{J}_{t-1}$$

These procedures make it possible to estimate the most probable process states and to make prediction or denoising tasks when nonlinear process are studied. The filtered value $\mathbf{o}^\phi_f(t)$ and the smoothed value $\mathbf{o}^\phi_s(t)$ of $\phi(\mathbf{o}_t)$ are respectively given by:

$$\mathbf{o}^\phi_f(t) = \alpha_t\phi(\mathbf{o}_t) + B\tilde{\mathbf{s}}^t(t) + B\tilde{\boldsymbol{\mu}}_x$$

$$\mathbf{o}^\phi_s(t) = \alpha_t\phi(\mathbf{o}_t) + B\tilde{\boldsymbol{\varsigma}}(t) + B\tilde{\boldsymbol{\mu}}_x.$$

3) Learning the Parameters: In this section, we show how the parameters of the Kernel Kalman Filter can be learned in combination with the filtering and smoothing pass by the use of an EM procedure in a way similar to that recalled in [3].

Recall that the EM algorithm for linear dynamical systems based on a training sequence $\mathbf{x}_{1:T}$ aims at maximizing the likelihood $\mathcal{L}_{lds}$

$$\begin{aligned}\mathcal{L}_{lds}(\mathbf{x}_{1:T}|\boldsymbol{\theta}) = &-\frac{d}{2}\log\sigma^2_1 - \frac{1}{2\sigma^2_1}(\mathbf{x}_1 - \boldsymbol{\mu}_1)'(\mathbf{x}_1 - \boldsymbol{\mu}_1)\\ &- \frac{d(T-1)}{2}\log\sigma^2_s - \frac{dT}{2}\log\sigma^2_x\\ &- \frac{1}{2\sigma^2_s}\sum_{t=2}^{T}\|\mathbf{s}_t - A\mathbf{s}_{t-1} - \boldsymbol{\mu}_s\|^2\\ &- \frac{1}{2\sigma^2_x}\sum_{t=2}^{T}\|(\mathbf{x}_t - C\mathbf{s}_t - \boldsymbol{\mu}_x)\|^2.\end{aligned}$$

The EM algorithm to maximize this likelihood requires the implementation of the two following steps:

- E step: perform the filtering and smoothing procedures given the parameter $\boldsymbol{\theta}$ with the observed (training) sequence $\mathbf{x}_{1:T}$;
- M step: maximize the expectation of $\mathcal{L}_{lds}$ with respect to $\boldsymbol{\theta}$ using the filtered and smoothed values and the

values $R(t) = \Gamma(t) + \varsigma(t)\varsigma'(t)$ and $R^{t-1}(t) = \Gamma^{t-1}(t) + \varsigma(t)\varsigma'(t-1)$ (see [3] for more details).

This EM procedure remains valid when the learning of the parameters of a Kernel Kalman Filter is addressed. Given a basis B of $\mathcal{H}_x$, the filtering and the smoothing pass described above must first be implemented.

To perform the M step, it may be useful to compute the values $\tilde{R}(t)$, $\tilde{R}^{t-1}(t)$ and the auxiliary vectors $\tilde{\mathbf{c}}_t$:

$$\begin{aligned}\tilde{\mathbf{c}}_t &= \alpha_t B'\mathbf{x}_t^\phi + \tilde{\varsigma}(t)\\ \tilde{R}(t) &= \tilde{\Gamma}(t) + \tilde{\mathbf{c}}_t\tilde{\mathbf{c}}_t'\\ \tilde{R}^{t-1}(t) &= \tilde{\Gamma}^{t-1}(t) + \tilde{\mathbf{c}}_t\tilde{\mathbf{c}}_{t-1}'\end{aligned}$$

These values and subsequent values rely on the assumption that each vector $\mathbf{s}_t^\phi$ of (1a) is searched for as an element of $\mathcal{H}_x$. In addition, let the vectors $\mathbf{f} = \sum_{t=2}^T \tilde{\mathbf{c}}_t$ and $\mathbf{g} = \sum_{t=2}^T \tilde{\mathbf{c}}_{t-1}$ together with the matrices

$$\begin{aligned}G &= \sum_{t=2}^T \tilde{R}(t-1), \qquad R = G - \mathbf{g}\mathbf{g}'/(T-1)\\ L &= \sum_{t=2}^T \tilde{R}^{t-1}(t), \qquad S = L - \mathbf{f}\,\mathbf{g}'/(T-1).\end{aligned}$$

According to the values of A and $\boldsymbol{\mu}_s$ obtained when a linear dynamical system is considered, we get:

$$\tilde{A} = \frac{1}{\sum_{t=2}^T \lambda_{t-1}} S\left[I - \left(\left(\sum_{t=2}^T \lambda_{t-1}\right)I + R\right)^{-1} R\right] \tag{14}$$

$$\tilde{\boldsymbol{\mu}}_s = \frac{1}{T-1}(\mathbf{f} - \tilde{A}\mathbf{g}) \tag{15}$$

$$\tilde{\boldsymbol{\mu}}_1 = \tilde{\mathbf{c}}_1 \tag{16}$$

from which $A = B\tilde{A}B'$, $\boldsymbol{\mu}_s^\phi = B\tilde{\boldsymbol{\mu}}_s$ and $\boldsymbol{\mu}_1^\phi = B\tilde{\boldsymbol{\mu}}_1$. In order to cope with the possible high complexity induced by the use of kernels, it may be sensible to put a gaussian prior p_γ

$$p_\gamma(A) \propto \exp\left(-\frac{\gamma}{2\sigma_s^2}\,\mathrm{tr}\,(A'A)\right) \tag{17}$$

on A, favoring A with small entries. Resorting to such control complexity, each occurence of $\sum_{t=2}^T \lambda_{t-1}$ in (14) must be replaced by $\sum_{t=2}^T \lambda_{t-1} + \gamma$.

For σ_s^2 and σ_1^2 we have:

$$\begin{aligned}\sigma_s^2 &= \frac{1}{T-1}\sum_{t=2}^T \lambda_{t-1} + \frac{1}{\ell(T-1)}\,\mathrm{tr}\left(F - \tilde{A}L - \tilde{\boldsymbol{\mu}}_s\mathbf{f}'\right)\\ \sigma_1^2 &= \lambda_1 - \frac{1}{\ell}\tilde{\mathbf{c}}_1'\tilde{\mathbf{c}}_1.\end{aligned}$$

Finally, the values for $\boldsymbol{\mu}_x^\phi$ and σ_x^2 are given by:

$$\begin{aligned}\tilde{\boldsymbol{\mu}}_x &= \frac{1}{T}\sum_{t=1}^T((1-\alpha_t)B'\mathbf{x}^\phi - \tilde{\varsigma}(t))\\ \sigma_x^2 &= \frac{1}{\ell T}\sum_{t=1}^T\left((1-\alpha_t)\mathbf{x}_t^{\phi\prime}BB'\mathbf{x}_t^\phi - \mathbf{x}_t^{\phi\prime}B(\tilde{\varsigma}(t) + \tilde{\boldsymbol{\mu}}_x)\right).\end{aligned}$$

from which $\boldsymbol{\mu}_x^\phi = B\tilde{\boldsymbol{\mu}}_x$ (note that $\mathbf{x}_t^{\phi\prime}BB'\mathbf{x}_t^\phi = k(\mathbf{x}_t, \mathbf{x}_t)$). Hence, the assumption that each $\mathbf{s}_t^\phi$ is in $\mathcal{H}_x$ validates the form used by the kernel filtering and smoothing procedures (cf. (14)–(16)).

III. Related Work

This work is related to the extended versions of the classical Kalman filter, namely the Extended Kalman Filter or EKF [7], [8] and the Unscented Kalman Filter or UKF [9], [10] and their variants. These algorithms can indeed be used in tasks involving prediction and/or noise removal when nonlinear processes are studied. However, as evidenced previously, our model does not need to resort to any approximated estimation strategy while, in addition, it makes it possible to implement the learning of the model parameters.

Two other works are closely related to ours but are limited to the nonlinear filtering problem: the approach described in [11], which tackles the so-called N-stage optimal control problem using support vector regressors to model the transition process and the work developed in [12], which proposes to learn (online) the weight vector of a kernel regressor using a Kalman filter with an invariant state transition process.

We may also notice that the work presented in [6] is a special case of the Kernel Kalman Filter and is only suitable to address the problem of time series prediction (see section IV).

Eventually, regarding the issue of parameters learning, our algorithm must be related to the work of [13] and [14]. The algorithms presented in these papers however assume some semi-parametric models whose parameters have to be learned, while no such assumption on the model family is made in our approach, the types of kernels to be used being the only choice required by our approach.

IV. Numerical Simulations

A. Datasets and Protocol

The experiments presented in this section have been carried out on four datasets: the univariate time series *Laser* and *Mackey-Glass* MG_{30} and the multi-dimensional time series *Ikeda* and *Lorenz*. Two kinds of KKF performances are evaluated: the prediction capacity and the ability to process noisy nonlinear time series.

The *Ikeda map* is related to a laser dynamics and is defined from a starting point $(x_1(0), x_2(0))$ by

$$\begin{cases}\omega(t) = c_1 - \dfrac{c_3}{1 + x_1^2(t) + x_2^2(t)}\\ x_1(t+1) = r + c_2\left(x_1(t)\cos\omega(t) - x_2(t)\sin\omega(t)\right)\\ x_2(t+1) = c_2\left(x_1(t)\sin\omega(t) + x_2(t)\cos\omega(t)\right)\end{cases}$$

where c_1, c_2, c_3 and r are real valued parameters; we considered $c_1 = 0.4$, $c_2 = 0.84$, $c_3 = 6.0$, $r = 1.0$ and $\mathbf{x}(0) = [x_1(0)\ x_2(0)]' = [1.0\ 0.001]'$.

A *Lorenz* attractor is the solution of the system of differential equations

$$\begin{cases}\dfrac{dx(t)}{dt} = -ax + ay\\ \dfrac{dy(t)}{dt} = -xz + rx - y\\ \dfrac{dz(t)}{dt} = xy - bz\end{cases}$$

We set $a = 10$, $r = 28$ and $b = 8/3$.

The *Laser* and *Mackey-Glass* MG_{30} time series are univariate and, as proposed in [15], [16], we introduce the embedding vectors $\mathbf{x}_t = [x_{t-(d-1)\kappa} \cdots x_{t-\kappa}\ x_t]'$ where κ is a fixed step size; the series $\mathbf{x}_{1:T} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ is then used as a training series. For the *Laser* time series, we set $\kappa = 1$ and $d = 8$ whereas $\kappa = 6$ and $d = 6$ are used for MG_{30}. Conversely, *Ikeda* and *Lorenz* are multi-dimensional first-order serie and the resort to embedding vectors (though possible) is not considered.

For all datasets, the data are split up in $n_{train} = 300$ training data, $n_{test} = 200$ test data to choose the hyperparameters of the models (number of neurons on the hidden layer of the MLPs, widths of the gaussian kernels) and $n_{valid} = 300$ validation data. Three training methods for the prediction of time series are tested: a multilayer perceptron (MLP), a support vector regressor (SVR) and a Kernel Kalman Filter (KKF). For those last two kernel machines, gaussian kernels $k(\mathbf{x}, \mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\sigma_k^2})$ have been used. The nonlinearity used in the MLPs is implemented by a tanh function and the code used to implement the MLP and other filtering techniques is that of R. van der Werve's ReBEL toolbox[1]. The training of multi-dimensional time series with support vector regressors is done by decomposing the learning problem in as many number of training problem as the dimension of the time series.

For all the experiments, the determination of the preimages is done with the strategy described in [17].

The measurement error used to evaluate the performance of the different algorithms is the mean squared error *err* defined for a machine f and a series $\mathbf{x}_{1:T}$ as:

$$err(f, \mathbf{x}_{1:T}) = \frac{1}{(T-1)} \sum_{t=2}^{T} \|\mathbf{x}_t - f(\mathbf{x}_{t-1})\|^2.$$

B. Prediction Accuracy of KKF

Two kinds of prediction capacities are evaluated for KKF. The first one is the *one-step* ahead prediction ability, which merely consists in realizing the prediction at time $t+1$ from the actual observation at time t. The second one is the *multi-step* ahead or *trajectory* prediction prediction where the model relies on its own output estimates to compute future observations.

The training sequences provided to the algorithms are clean (non noisy) sequences. Thus, for the prediction task, KKF is learned with $\mu_x^\phi = 0$ and $\sigma_x^2 = 0$.

Table I reports the validation mean squared error obtained by the different models tested for the one-step prediction and the trajectory prediction. For KKF, the complexity control discussed above (see (17)) has been tested: the results are shown for different values of γ in order to assess its influence on the prediction quality.

The analysis of the results leads to the first conclusion that KKF learning with no measurement noise is able to provide a predictor of quality comparable with that of MLP and SVR. This prediction capacity is witnessed for all the datasets even when the difficult task of the trajectory prediction is considered. This observation is all the more striking that the determination of the preimages is an approximate process and the accumulation of such approximation errors over iterated predictions may have been expected to deteriorate the overall performance.

A second observation relates to the dependence of our algorithm on the regularization γ: the unregularized method ($\gamma = 0$) never obtains the best error, neither for the one-step ahead prediction nor for the trajectory prediction. This evidences the excessive complexity induced by the use of kernel functions and, as a possible consequence, the presence of an over-fitting phenomenon. Conversely, when the regularization is too important (too large coefficient γ), the model cannot reach interesting performances.

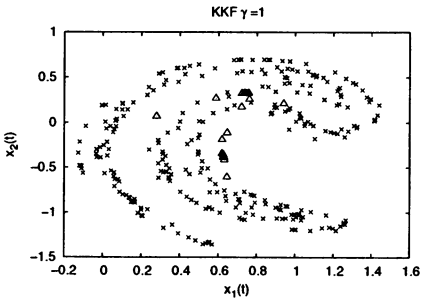

Fig. 1. Trajectory prediction made by KKF ($\gamma = 1$). The triangles correspond to the KKF prediction and the crosses to the actual time series (see text for comments).

Incidentally, a strange result about the prediction of the series *Ikeda* by our algorithm may be noticed: the best regularization parameters (with respect to the validation error) for the trajectory prediction is the one which has the worst results for the one-step prediction. Actually, if the forecasted trajectory provided by the corresponding model ($\gamma = 1$) is compared to the true *Ikeda* trajectory, it appears that this low quadratic error does not square with an actual prediction ability; it is instead a sort of degenerated quasi-periodical trajectory passing through particular position (see Figure 1). This phenomenon is also observed for the values $\gamma = 0.1$ and $\gamma = 0.001$ whereas for the other values of γ the trajectory envisaged is very similar with the real trajectory (see Figure 2). The trajectory predictions for the other time series do not undergo such an atypical behaviour.

These prediction experiments validate both the efficiency of the Kernel Kalman Filter and the learning strategy to compute the preimages.

C. Noise Extraction

The task tackled here consists in removing the noise of one series to which an observation noise has been added. For our experiments, we added a zero-mean gaussian noise of variance σ_{noise}^2 to the validation series of length n_{valid}; such a noise variance leads to a signal to noise ratio of 3dB.

The experiments described in this section have been carried out on the series MG_{30}, *Ikeda* and *Lorenz*. The parameters of

[1] http://choosh.ece.ogi.edu/rebel/index.html.

TABLE I

MEAN SQUARED ERROR FOR THE ONE-STEP AHEAD PREDICTION. γ IS THE REGULARIZING PARAMETER OF EQUATION (17).

		One-step prediction				Multi-step prediction			
Algo.	**Param.**	***Laser***	***MG*$_{30}$**	***Ikeda*** ($\times 10^{-4}$)	***Lorenz***	***Laser***	***MG*$_{30}$**	***Ikeda***	***Lorenz***
MLP	-	1.4326	0.0461	**7.093**	0.2837	4450	**1.046**	0.9153	486.4
SVR	-	0.2595	0.0313	8.103	**0.1811**	**2263**	1.173	0.9231	**411.8**
KKF	$\gamma = 0.00$	0.2812	0.0512	7.922	0.3134	2304	1.242	0.9705	422.2
	$\gamma = 10^{-9}$	0.2641	0.0453	7.914	0.3133	2297	1.121	0.9415	425.5
	$\gamma = 10^{-7}$	**0.2325**	0.0364	7.773	0.3133	2270	1.095	0.9914	428.1
	$\gamma = 10^{-5}$	0.2325	**0.0307**	8.421	0.3186	**2263**	1.090	1.0593	430.9
	$\gamma = 10^{-3}$	0.2732	0.0315	15.71	0.3230	2281	1.085	0.9528	441.7
	$\gamma = 10^{-1}$	0.2812	0.0374	30.81	0.7922	2298	1.111	0.8550	459.4
	$\gamma = 1.00$	0.3101	0.0521	35.65	5.1135	2315	1.241	**0.7263**	508.1

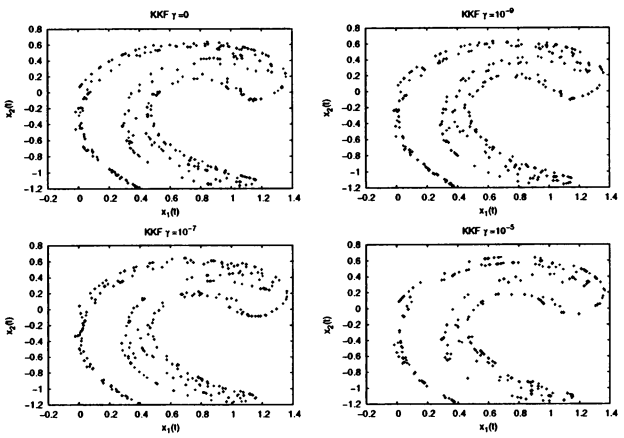

Fig. 2. Trajectory prediction made by KKF on the Ikeda series (above each plot, the value of γ).

KKF model are, for each time series, those which correspond to the best one-step ahead prediction results . The other filtering algorithms make use of the best MLP to model the transition between the states $\mathbf{s}_t$ (see Table I). Each experiment of noise removal is repeated 20 times and Table II reports the mean-squared errors and standard deviations between the denoised signal and the original one.

The results of this table show that our method of filtering is as effective as the usually implemented methods when the problem of nonlinear series denoising is tackled. In particular, it may be noticed that in the case of *MG*$_{30}$, the measured mean squared error is lower than that provided by the others algorithms. Besides, in all cases, KKF produces results on average better than those of the usually used procedure of the Extended Kalman Filter. Regarding the *Lorenz* series, a relatively important difference between the results of our method and the best results can be noticed; the reported standard deviation however prevents from concluding about any kind of limitation regarding KKF for the *Lorenz* problem.

Finally, the main idea of these experiments of noise extraction is that our model makes it possible to proceed effectively to this task by using the matrix equations of recursive estimation given in section II-B.2.

TABLE II

MEAN SQUARED ERRORS AND STANDARD DEVIATION FOR THE DENOISING TASK. CDKF DIFFERS FROM UKF BY THE STRATEGY USED TO DETERMINE THE SO-CALLED *Sigma* POINTS.

Algo.	***MG*$_{30}$**	***Ikeda*** ($\times 0.1$)	***Lorenz*** ($\times 10$)
EKF	631.7(1230)	6.842(1.520)	111.1(213.1)
UKF	9.288(0.560)	1.624(0.086)	2.876(5.954)
CDKF	10.12(1.510)	1.387(0.095)	1.945(1.894)
KKF	8.741(0.435)	2.145(0.012)	20.22(31.41)

D. Dynamics Extraction

The last capacity of KKF we evaluate relates to the effectiveness of the training procedure that we described in section II-B.3. We focus on the *Ikeda* problem and proceed as follows. First, we produce a series $\mathbf{x}_{1:T} = \{\mathbf{x}_1, \ldots, \mathbf{x}_t\}$ of 300 training noisy points from the *Ikeda* map, using the transition process

$$\begin{cases} s_1(0), s_2(0) \\ \omega(t) = c_1 - \dfrac{c_3}{1 + s_1^2(t) + s_2^2(t)} \\ s_1(t+1) = r + c_2\left(s_1(t)\cos\omega(t) - s_2(t)\sin\omega(t)\right) + \nu_1^s \\ s_2(t+1) = c_2\left(s_1(t)\sin\omega(t) + s_2(t)\cos\omega(t)\right) + \nu_2^s \end{cases}$$

and the observation process

$$\left\{\mathbf{x}_t = \begin{bmatrix} x_1(t) \\ x_2(t) \end{bmatrix} = \begin{bmatrix} s_1(t) + \nu_1^x \\ s_2(t) + \nu_2^x \end{bmatrix}\right.$$

where the vectors $\boldsymbol{\nu}_s = [\nu_1^s \ \nu_2^s]'$ and $\boldsymbol{\nu}_x = [\nu_1^x \ \nu_2^x]'$ are noise vectors. Then, we provide KKF with the series $\mathbf{x}_{1:T}$ from which the learning procedure determines the optimal parameters. Lastly, we measure the one-step ahead prediction capacity on a clean validation series of length 300.

In order to show the genericity of our model, the kernel used here is no longer gaussian but polynomial of degree 5

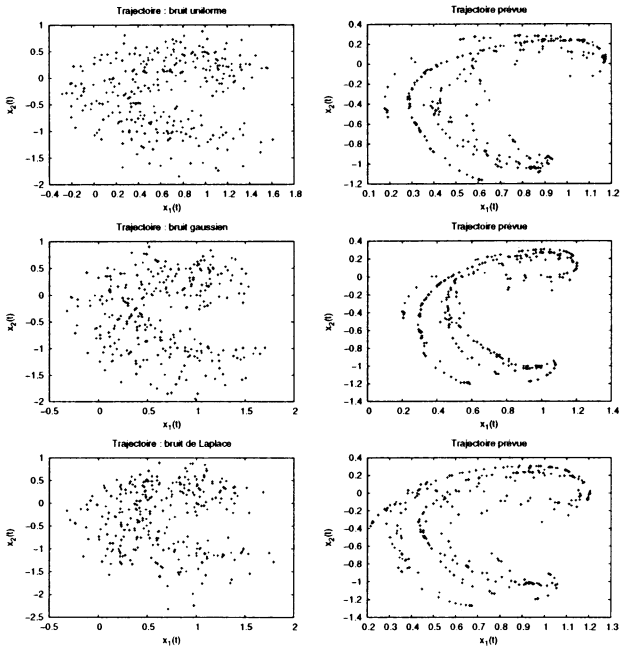

Fig. 3. Left column: noisy training series. Right column: extracted dynamics. The first row corresponds to a uniform noise (validation error: 0.061), the second one to a gaussian noise (error: 0.061) and the last one to a Laplace noise (error: 0.064).

such that $k(\mathbf{x},\mathbf{y}) = (\mathbf{x}\cdot\mathbf{y}+1)^5$. A value of regularization $\gamma = 10^{-5}$ is used.

For the experiments, $\boldsymbol{\nu}_x$ is selected as a zero-mean gaussian vector with variance $\sigma_x^2 = 0.13$, which corresponds to the use of a signal to noise ratio of 3dB with respect to the validation time series. Three types of noise are used for $\boldsymbol{\nu}_s$: a zero-mean gaussian noise with variance $\sigma_x^2 = 0.01$ (Figure 3, top), a uniform noise on $[-0.2; 0.2]$ (Figure 3, middle) and a zero-mean Laplace noise distributed as $p(x) = \exp(-|x|/b)$ with $b = 0.044721$ (Figure 3, bottom); choosing b this way leads to a noise of variance 0.01.

Figure 3 illustrates the efficiciency of our method for the extraction of complex dynamics from a noisy series. In particular, it must be noticed that our algorithm is able to extract dynamics very similar to the actual dynamics whichever noise model is considered. These results are all the more remarkable as the studied dynamics are very complex, which again validates our kernel extension of linear dynamical systems.

V. Conclusion and Future Work

We have presented a new kernel method, the *Kernel Kalman Filter*, which is an extension of the Kalman filter and linear dynamical systems. We have set up the filtering, smoothing and learning procedures for this model showing that the involved equations are closely related to their classical linear counterparts. Throughout various experiments, we have shown the efficiency of KKF in tasks such as nonlinear time series denoising and prediction; the problem of extracting complex dynamics from noisy nonlinear dynamics has also successfully been tackled by our approach.

Several extensions to this work can be envisioned. First, the order of the matrices involved by the model linearly scales with the length of the studied time series, leading to $O(\ell^3)$ and $O(\ell^2)$ computation and memory requirements, respectively. A sparse incremental greedy method could be a workaround to this problem and is currently being investigated; this extension should make possible the use of KKF for problems involving a few thousand data. Second, whereas working with a linear process in the state space allows for some interpretability, this advantage is lost when working in the feature space. Therefore, an interesting extension would be to use kernels well-suited for interpretability and to learn their parameters in order to extract some valuable knowledge. Finally, the implementation of an automatic learning scheme to regularize the model, such as, for instance, a full Bayesian learning process, would be an extension of great interest to the presented work.

References

[1] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[2] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. of the Royal Statistics Society*, vol. 39, no. 1, 1977.

[3] A.-V. Rosti and M. Gales, "Generalised linear Gaussian models," Cambridge University Engineering Department, Tech. Rep. CUED/F-INFENG/TR.420, 2001.

[4] B. Boser, I. Guyon, and V. Vapnik, "A Training Algorithm for Optimal Margin Classifiers," in *Proc. of the 5th Workshop on Comp. Learning Theory*, vol. 5, 1992.

[5] S. Mika, B. Schölkopf, A. J. Smola, K.-R. Müller, M. Scholz, and G. Rätsch, "Kernel PCA and De-Noising in Feature Spaces," in *Adv. in Neural Information Processing Systems*, 1999.

[6] L. Ralaivola and F. d'Alché-Buc, "Dynamical Modeling with Kernels for Nonlinear Time Series Prediction," in *Adv. in Neural Information Processing Systems*, vol. 16, 2004.

[7] B. D. Anderson and J. B. Moore, *Optimal Filtering*. Englewood Cliffs, NJ: Prentice Hall, 1979.

[8] G. Welch and G. Bishop, "An introduction to the Kalman filter," University of North Carolina, Tech. Rep. TR 95-041, 1995.

[9] S. Julier and J. Uhlmann, "A New Extension of the Kalman Filter to Nonlinear Systems," in *Int. Symp. Aerospace/Defense Sensing, Simul. and Controls*, 1997.

[10] R. van der Merwe, N. de Freitas, A. Doucet, and E. Wan, "The Unscented Particle Filter," in *Adv. in Neural Information Processing Systems*, vol. 13, Nov 2001.

[11] J. A. K. Suykens, J. Vandewalle, and B. D. Moor, "Optimal control by least squares support vector machines," *Neural Networks*, vol. 14, no. 1, pp. 23–35, 2001.

[12] Y. Engel, S. Mannor, and R. Meir, "Sparse online greedy support vector regression," in *Proc. of the 13th European Conference on Machine Learning*, 2002.

[13] Z. Ghahramani and S. Roweis, "Learning nonlinear dynamical systems using an em algorithm," in *Adv. in Neural Information Processing Systems*, vol. 11, 1999.

[14] T. Briegel and V. Tresp, "Fisher Scoring and a Mixture of Modes Approach for Approximate Inference and Learning in Nonlinear State Space Models," in *Adv. in Neural Information Processing Systems*, vol. 13, 1999.

[15] S. Mukherjee, E. Osuna, and F. Girosi, "Nonlinear prediction of chaotic time series using support vector machines," in *Proc. of IEEE NNSP'97*, 1997.

[16] K. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik, "Predicting Time Series with Support Vector Machines," in *Proc. of Int. Conf. on Artificial Neural Networks*, 1997.

[17] G. H. Bakir, J. Weston, and B. Schölkopf, "Learning to Find Pre-Images," in *Adv. in Neural Information Processing Systems*, vol. 16, 2004.