2005 International Symposium on Nonlinear
Theory and its Applications (NOLTA2005)
Bruges, Belgium, October 18-21, 2005

On Comparing Nonlinear Filtering Algorithms

Jochen Bröcker†

†Centre for the Analysis of Time Series
London School of Economics
Houghton Street
London WC2A 2AE, United Kingdom
Email: j.broecker@lse.ac.uk

Abstract—In this paper we consider the performance of filtering algorithms, which means algorithms to retrieve the underlying state of a nonlinear system in a causal way. Since for nonlinear systems the optimal filter is computationally prohibitively expensive in general, one faces a trade–off between desired filtering accuracy and computational complexity. In order to ascertain whether a certain more powerful filtering algorithm is worth the computational effort, it is neccessary to carefully evaluate the potential benefits. We show that skill scores are a useful concept for this. We compare the performance of a number of filtering algorithms applied to a simple nonlinear system. We also discuss the perfomance in relation to the required computational resources. In general, higher accuracy requires more cpu-power.

1. Introduction

Nonlinear filtering is the problem of recovering the state of a (perhaps only partially known) nonlinear system from noisy observations in a causal manner, that is by not allowing future observations to enter the current state estimation. For linear systems with Gaussian uncertainties the problem is completely solved by the famous Kalman filter [8]. Although there exists a general optimal filter theory [7, 5], both theoretical analysis and practical application meet with the fundamental problem that optimal filters for nonlinear systems possess an infinite complexity, in a sense that can be made rigorous [9]. Consequently, applications rely on (suboptimal) approximations to the optimal filter. The theoretical understanding of such approximations is still unsatisfactory though, and comparable studies of different algorithms are lacking. A fair comparison of nonlinear filtering algorithms needs to take the probabilistic character of the problem into account. Furthermore, filter performance should be seen in relation to the computational burden of the implementation. Since the optimal filter has an infinite complexity, however well a given filter scheme might perform, an even better algorithm, although requiring more computer resources, still exists. This paper aims at providing a framework for comparable studies of filtering algorithms. As an example, we present a comparison of five filtering algorithms applied to a partially observed chaotic system.

The dynamical systems considered in this work are iterated nonlinear maps of the following form:

$$X_{n+1} = f(X_n), \tag{1}$$

where $f(\cdot)$ is a diffeomorphism of $\mathbb{R}^d$.

In applications the underlying dynamics of a measured time series are often are modeled by processes like (1). The problem is to estimate the unknown state X_n using noisy measurements. In this work we assume measurements of the form

$$Y_n = GX_n + \sigma \cdot W_n, \tag{2}$$

where W_n are independend normal random variables with zero mean and unit variance. G is a linear mapping. We assume Y_n to be one dimensional. Let $\mathcal{Y}^n := \{Y_1, \ldots, Y_n\}$. Then a rigorous approach to estimate X_n from $\mathcal{Y}^n$ is to consider the *conditional probability* $P(X_n|\mathcal{Y}^n)$ or the corresponding probability density function henceforth denoted by $\pi_n(x)$ (the dependency on $\mathcal{Y}^n$ is ommited in this notation). It turns out [5] that this pdf satisfies the iterative equation

$$\pi_{n+1}(x) = c \cdot q(Y_{n+1}, x) \cdot \mathcal{L}^* \pi_n(x). \tag{3}$$

Here $\mathcal{L}^*$ denotes the Frobenius–Peron operator which can be calculated from the equation (1) and turns out to be

$$\mathcal{L}^* p(x) = \det\left(\frac{\partial f^{-1}(x)}{\partial x}\right)^{-1} p(f^{-1}(x)). \tag{4}$$

The density $q(Y_{n+1}, x)$ is given by

$$\begin{aligned} q(y, x) &= p_{Y_n|X_n=x}(y) \\ &= \mathcal{N}(y, G(x), \sigma^2), \end{aligned}$$

where $\mathcal{N}$ is a gaussian pdf.

In order for Equation (3) to be useful in a practical application, it is of course necessary to represent $\pi_n(x)$ by a suitable finite dimensional parametrisation. If the dynamics are linear and the errors are gaussian, the Kalman filter provides such a parametrisation. As already mentioned though, if f is nonlinear, the filtering process $\pi_n(x)$ does not admit a finite dimensional parametrisation. Therefore, approximations are essential for applications. The approximative filter has to be, of course, finite dimensional and as optimal as possible. A large variety of approximation

methods have been conceived. All those methods face a tradeoff between computational complexity and accuracy. Although we will present some approximations in the following section, the purpose of this paper is not to provide a comprehensive study of filtering algorithms, but rather to present a framework to compare those methods in a fair and meaningful way, thus studying the tradeoff between computational complexity and accuracy.

2. Filtering Algorithms

In this section we briefly consider schemes to approximate the filtering process π_n. The approximative filtering process will be denoted by $\tilde{\pi}_n$. Probably the first widely applied nonlinear filtering algorithm was the Extended Kalman Filter. It consists essentially of the usual Kalman Filter Equations applied to the linearized nonlinear dynamics. We refer the reader to [5] for a thorough explanation of the technique and proceed to explain further techniques used in this paper.

2.1. Gaussian density filter

A simple method to obtain approximate solutions of Equation (3) is simply to *assume* that $\pi_n(x)$ is gaussian, even though it is not. After applying the operator $\mathcal{L}^*$, the density is generally nongaussian, but we can turn it into a gaussian again by retaining only the *predicted* mean and variance. Having done this, we have to multiply by $q(Y_{n+1}, x)$ in Equation (3), but this is just multiplication of two gaussians, resulting in a gaussian $\tilde{\pi}_{n+1}(x)$, which we take as an approximation to $\pi_{n+1}(x)$. If f is polynomial, this filter can be formulated explicity in terms of dynamical equations for the mean and the variance, which gives a very fast algorithm. If the underlying system has N dimensions, the filter has $N + \frac{N(N+1)}{2}$ dimensions and is implemented straightforwardly. This method can be extended using more complicated densities. For a general overview see [2].

2.2. Monte Carlo Method

The Monte Carlo Method discussed here is also known as *particle filter* and has been investigated in [3]. The idea is to generate M independent copies $\{X_n^{(k)}\}_{n\leq 0, k=1\ldots M}$ of (1), where n is, as before, the time and k denotes the k'th member of the ensemble. The method provides an approximation to $\pi_n(x)$ by a weighted average over δ–functions centered at the ensemble points as

$$\pi_n(x) \cong \sum_{k=1}^{M} w_n^{(k)} \cdot \delta(x - X_n^{(k)}).$$

To give an expression for $w_n^{(k)}$, define the quantities

$$g_j^{(k)} := q(Y_j, X_j^{(k)}) \qquad \text{for all } j \leq 0, k = 1, \ldots, M.$$

Theoretically one might guess that

$$w_n^{(k)} = c \cdot \prod_{j=1}^{n} g_j^{(k)},$$

where c is a constant chosen to yield

$$\sum_k w_n^{(k)} = 1$$

is a reasonable choice. A profound analysis of the problem however shows that this method tends to diverge, and one should rather implement a *limited memory version* of the filter, where the memory depends on the ensemble size M. This is done as follows: Let q_M be a certain positive integer depending on the ensemble size M. Then define the weights to be

$$w_n^{(k)} = c \cdot \prod_{j=n-q_M}^{n} g_j^{(k)},$$

where we define $g_j^{(k)} := 1$ if j is negative or zero. If q_M is set to q_M = integer closest to $2\sqrt{\log(M)}$, it can be shown that the Monte Carlo Filter converges to the optimal filter (in some sense) for $M \to \infty$. The general drawback of Monte Carlo Methods is the required computer power. It is necessary to store the ensemble points and the associated weight vectors. Furthermore, the dynamical equations (e.g. iterated maps or stochastic differential equations) have to be solved for all ensemble points in parallel. This obviously requires more power than the previously discussed low dimensional filters.

3. Proper Scoring Rules

The main focus of this paper is on the fair evaluation of filtering algorithms. The optimal filtering proces in our setup is $\pi_n(x)$, so a measure of the quality of a filtering process $\tilde{\pi}_n$ could be some sort of distance between $\pi_n(x)$ and $\tilde{\pi}_n(x)$. Although in general $\pi_n(x)$ is not available, the concept of *Skill Scores* provides a measure by which two approximative filtering processes can be compared in terms of their relative distance to $\pi_n(x)$. A skill score is a function $S(p, x)$, where p is a probability density and x is a real number. By means of a skill score we can compute the *skill*

$$\mathcal{S}(p, q) = \int S(p, x) q(x) \mathrm{d}x$$

of p with respect to q. We now consider some examples. The common *Mean Square Error* is the skill defined via the score

$$S(p, x) = (x - \int z p(z) \mathrm{d}z)^2.$$

This score measures the quality of the mean of $p(x)$ as an estimator for x. As is obvious from the definition, the Mean Square Error depends on $p(x)$ only through its first moment

and hence cannot be expected to value all aspects of p properly.

The *Ignorance Score* is defined by

$$S(p, x) = -\log(p(x))$$

The Ignorance Score is related to the log–likelihood [10] and plays an importand role in gambling theory. Another interesting score (although not used in this paper) is the *Proper Linear Score*. It is defined as

$$S(p, x) = \int p^2(z)\mathrm{d}z - p(x).$$

It should be noted that the Proper Linear Score depends on the functional form of p while the Ignorance depends on p only via the single number $p(x)$. This property is called *locality*. We note that in this paper skill scores are defined like cost functions: small numerical values indicate better skill. Especially we want the score to be minimal if p and q coincide. This property is called *propriety*. Mathematically, a skill score is proper if for any two densities $p(x)$ and $q(x)$

$$\mathcal{S}(q, p) \geq \mathcal{S}(p, p).$$

In other words, the minimum of the left hand side over $q(x)$ is obtained for $q = p$. A skill score is *strictly proper* if that happens *only* if $q = p$.

Propriety is a property only of the score itself. The Ignorance and the Proper Linear Score are proper. A general result due to Bernardo (see [1]) states that all smooth, proper and local scores are affine functions of the Ignorance. Proper scores in general have been characterized by Raftery & Gneiting [4].

The Mean Square Error is proper, but not strictly proper. In fact, it is easy to see that $\int (x - m)^2 q(x)\mathrm{d}x$ is minimal if $m = \int x\, q(x)\mathrm{d}x$ (proving that the Mean Square Error is proper), yet any other density $p(x)$ having the correct first moment m will achieve the same skill.

In filtering we are concerned not only with a single pdf p but with a sequence $\tilde{\pi}_n(x)$ of pdfs. If we have corresponding true states X_n available, we can estimate the *mean skill* of the filter (with respect to a proper skill score S). To this end, define the *empirical skill*

$$S(\tilde{\pi})_N := \frac{1}{N} \sum_{n=1}^{N} S(\tilde{\pi}_n, X_n)$$

We assume this to converge to the mean skill

$$S(\tilde{\pi})_N \to E\left[S(\tilde{\pi}_n, X_n)\right].$$

The mean skill can be written as

$$E\left[S(\tilde{\pi}_n, X_n)\right] = E\left[\int S(\tilde{\pi}_n, x)\pi_n(x)\mathrm{d}x\right].$$

To see this, condition on $\mathcal{Y}^n$ under the expectation. If the skill score is strictly proper, this expression is minimal if and *only* if $\tilde{\pi}_n = \pi_n$ for every n. In other words, the conditional probability π_n minimizes the mean skill. For improper skill scores though, a filtering process different from the optimal one could achieve a *better* skill, which essentially means that the filter we think is right would *not* achieve the best score.

4. Numerical Examples

To demonstrate our methodology, we applied the aforementioned approximative filtering algorithms to the well known Hénon system definded as

$$f(x) = [1 - 1.4x_1^2 + 0.3x_2, x_1].$$

As measurement variable we used $y = x_1 + x_2 + \sigma \cdot W_n$. We compared the Extended Kalman Filter (EKF), the Gaussian Density Filter (GDF), a Monte Carlo Filter using an ensemble of 1024 points (MCF1024), Monte Carlo Filter using an ensemble of 2048 points (MCF2048) and an Indistinguishable States Filter (ISF) (see [6]) that will not be described here. Actually, the invariant measure was used as a further "filter" for reference.

Each filter was allowed an initial training sequence of 256 points, during which certan parameter tuning was allowed. Then the empirical skill was measured as a mean over 1792 points. The skill score used was the Ignorance. We recorded the cpu–times needed to carry out the sweep through the data.

In Figure 1 we plotted the performance of the filtering algorithms versus the required cpu–time. The panels show the performance for a noise level (from top to bottom) of 15dB, 18dB and 21dB. The performance is actually measured relative to the zero–skill filter given by the invariant measure. This filter does not need any cpu–time (it does not take any measurements into account), so it appears furthest on the left with zero skill. The other filters generally show increase in perfomance with increasing cpu–demand, with the ISF being the most time consuming and best performing filter (except for the 15dB case). The gain in performance, however, appears to settle with increasing complexity. Bootstrap–bars indicate 90% isopleths. It should be noted that except for the ISF in the 15dB case, none of the filters could be termed better or worse that any other in general. A better performance in general has to be paid for by higher computational burden. Given that in higher dimensions ensemble filters need huge Monte Carlo ensembles to sample the state space, further studies should probably also take the size of the required storage into account. Furthermore, the requirements to store the actual pdf $\tilde{\pi}_n$ can be an issue, especcially if the results is to be disseminated, for example in weather forecasting.

5. Conclusion

In this paper we presented a framework to compare the performance of filtering algorithms. We suggested

that rather just the accuracy alone, the trade–off between desired filtering accuracy and computational complexity should be investigated. In order to ascertain whether a certain more powerful filtering algorithm is worth the computational effort, it is neccessary to carefully evaluate the potential benefits. Often filtering algorithms (as well as prediction and other estimation schemes) have been valued only in terms of the Mean Square Error which does not comprehensively assess all features of a pdf. We suggested that conceptually better measures are provided by skill scores. To illustrate our point, we compared the performance of a few algorithms applied to the henon system with the average cpu–time they required to carry out a single filtering step. In general, higher accuracy requires longer time. For an overall quality assessment, subsequent studies should take other aspects of the filtering algorithm into account, for example the size of the required storage or the size of the actual product.

Acknowledgments

The author would like to thank the members of the CATS group at the London School of Economics as well as Kevin Judd for fruitful discussions. Part of this work originated in research during JB's PhD, which was supervised by Ulrich Parlitz. Furthermore, this work was financially supported by CATS.

References

[1] Bernardo, J. M. Expected information as expected utility. *Annals of Statistics*, 7 (1979), 686–690.

[2] Bröcker, J. *Approximations and Applications of Nonlinear Filters*. PhD thesis, Universität Göttingen, 2003.

[3] Del Moral, P. Nonlinear filtering: monte-carlo particle resolution. Tech. Rep. 02, Laboratoire de Statistique et Probabilités, Université Paul Sabatier, 31062 Toulouse, 1996.

[4] Gneiting, T., and Raftery, A. Strictly proper scoring rules, prediction, and estimation. Tech. Rep. 436, Department of Statistics, University of Washington, 2004.

[5] Jazwinsky. *Stochastic Processes and Filtering Theory*, vol. 64 of *Mathematics in Science and Engineering*. Academic Press, 1970.

[6] Judd, K., and Smith, L. A. Indistinguishable states i: the perfect model scenario. *Physica D 151* (2001), 125–141.

[7] Kallianpur, G. *Stochastic Filtering Theory*. No. 13 in Applications of Mathematics. Springer Verlag, 1980.

[8] Kalman, R. E. A new approach to linear filtering and prediction problems. *Trans. ASME, Ser. D: J. Basic Eng. 82* (1960), 35–45.

[9] Levine, J., and Pignie, G. Exact finite–dimensional filters for a class of nonlinear discrete–time systems. *Stochastics 18* (1986), 97–132.

[10] Mood, A. M., Graybill, F. A., and Boes, D. C. *Introduction to the Theory of Statistics*. McGraw-Hill Series in Probability and Statistics. McGraw-Hill, 1974.

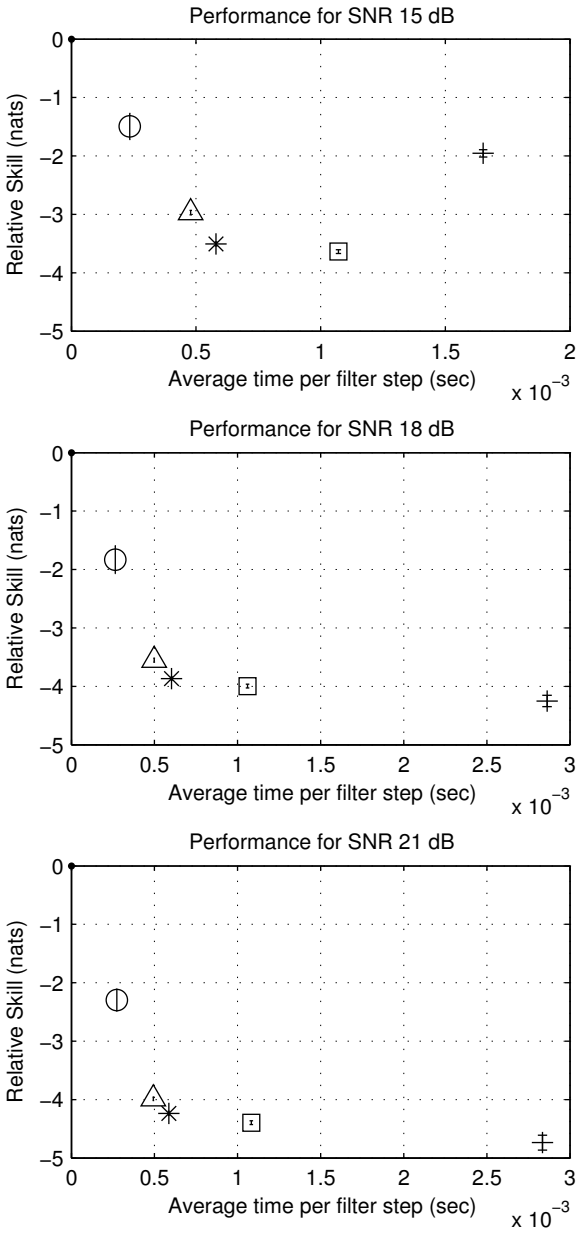

Figure 1: Performance of the filtering algorithms vs. time. The symbols indicate EKF (circle), GDF (triangle), MKF1024 (star), MKF2048 (box) and (ISF) (plus sign).