Wavelet-Based Combined Signal Filtering and Prediction

Olivier Renaud, Jean-Luc Starck, and Fionn Murtagh

***Abstract*— We survey a number of applications of the wavelet transform in time series prediction. We show how multiresolution prediction can capture short-range and long-term dependencies with only a few parameters to be estimated. We then develop a new multiresolution methodology for combined noise filtering and prediction, based on an approach which is similar to the Kalman filter. Based on considerable experimental assessment, we demonstrate the powerfulness of this methodology.**

***Index Terms*— Wavelet transform, filtering, forecasting, resolution, scale, autoregression, time series, model, Kalman filter.**

I. Introduction

There has been abundant interest in wavelet methods for noise removal in 1D signals. In many hundreds of papers published in journals throughout the scientific and engineering disciplines, a wide range of wavelet-based tools and ideas have been proposed and studied. Initial efforts included very simple ideas like thresholding of the orthogonal wavelet coefficients of the noisy data, followed by reconstruction. More recently, tree-based wavelet denoising methods were developed in the context of image denoising, which exploit tree structures of wavelet coefficients and parent-child correlations, which are present in wavelet coefficients. Also, many investigators have experimented with variations on the basic schemes – modifications of thresholding functions, level-dependent thresholding, block thresholding, adaptive choice of threshold, Bayesian conditional expectation nonlinearities, and so on. In parallel, several approaches have been proposed for time-series filtering and prediction by the wavelet transform, based on a neural network [24], [3], [14], Kalman filtering [6], [13], or an AR (autoregressive) model [19]. In Zheng et al. [24] and Soltani et al. [19], the undecimated Haar transform was used. This choice of the Haar transform was motivated by the fact that the wavelet coefficients are calculated only from data obtained previously in time, and the choice of an undecimated wavelet transform avoids aliasing problems. See also Daoudi et al. [7], which relates the wavelet transform to a multiscale autoregressive type of transform.

O. Renaud is with Methodology and Data Analysis, Section of Psychology, University of Geneva, 1211 Geneva 4, Switzerland. J.-L. Starck is with DAPNIA/SEDI-SAP, Service d'Astrophysique, CEA-Saclay, 91191 Gif sur Yvette, France. F. Murtagh is with the Department of Computer Science, Royal Holloway, University of London, Egham, Surrey TW20 0EX, England. Contact for correspondence, F. Murtagh, fmurtagh@acm.org

In this paper, we propose a new combined prediction and filtering method, which can be seen as a bridge between the wavelet denoising techniques and the wavelet predictive methods. Section II introduces the wavelet transform for time series, and section III describes the Multiscale Autoregressive Model (MAR), that is based on the above transform. It can be used either merely for prediction or as the first step of the new filtering approach presented in section IV.

II. Wavelets and Prediction

A. Introduction

The continuous wavelet transform of a continuous function produces a continuum of scales as output. However input data are usually discretely sampled, and furthermore a "dyadic" or two-fold relationship between resolution scales is both practical and adequate. The latter two issues lead to the discrete wavelet transform.

The output of a discrete wavelet transform can take various forms. Traditionally, a triangle (or pyramid in the case of 2-dimensional images) is often used to represent the information content in the sequence of resolution scales. Such a triangle comes about as a result of "decimation" or the retaining of one sample out of every two. The major advantage of decimation is that just enough information is kept to allow exact reconstruction of the input data. Therefore decimation is ideal for an application such as compression. It can be easily shown too that the storage required for the wavelet-transformed data is exactly the same as is required by the input data. The computation time for many wavelet transform methods is also linear in the size of the input data, i.e. $O(N)$ for an N-length input time series.

A major disadvantage of the decimated form of output is that we cannot simply – visually or graphically – relate information at a given time point at the different scales. With somewhat greater difficulty, however, this goal is possible. What is not possible is to have shift invariance. This means that if we had deleted the first few values of our input time series, then the output wavelet transformed, decimated, data would not be the same as heretofore. We can get around this problem at the expense of a greater storage requirement, by means of a redundant or non-decimated wavelet transform.

A redundant transform based on an N-length input time series, then, has an N-length resolution scale for each of the resolution levels that we consider. It is easy, under these circumstances, to relate information at each resolution scale for the same time point. We do have shift invariance. Finally, the extra storage requirement is by no means excessive. The redundant, discrete wavelet transform described in the next section is one used in Aussem et al. [2]. The successive resolution levels are formed by convolving with an increasingly dilated wavelet function.

B. The à trous Wavelet Transform

Our input data is decomposed into a set of band-pass filtered components, the wavelet coefficients, plus a low-pass filtered version of our data, the continuum (or background or residual).

We consider a signal or time series, $\{c_{0,t}\}$, defined as the scalar product at samples t of the function $f(x)$ with a scaling function $\phi(x)$ which corresponds to a low-pass filter:

$$c_{0,t} = \langle f(x), \phi(x-t) \rangle \tag{1}$$

The scaling function is chosen to satisfy the dilation equation:

$$\frac{1}{2}\phi\left(\frac{x}{2}\right) = \sum_k h(k)\phi(x-k) \tag{2}$$

where h is a discrete low-pass filter associated with the scaling function. This means that a low-pass filtering of the signal is, by definition, closely linked to another resolution level of the signal. The distance between levels increases by a factor 2 from one scale to the next.

The smoothed data $\{c_{j,t}\}$ at a given resolution j and at a position t is the scalar product

$$c_{j,t} = \frac{1}{2^j}\langle f(x), \phi\left(\frac{x-t}{2^j}\right)\rangle \tag{3}$$

This is consequently obtained by the convolution:

$$c_{j+1,t} = \sum_k h(k) \quad c_{j,t+2^j k} \tag{4}$$

The signal difference between two consecutive resolutions is:

$$w_{j+1,t} = c_{j,t} - c_{j+1,t} \tag{5}$$

which we can also, independently, express as:

$$w_{j,t} = \frac{1}{2^j}\langle f(x), \psi\left(\frac{x-t}{2^j}\right)\rangle \tag{6}$$

Here, the wavelet function is defined by:

$$\frac{1}{2}\psi\left(\frac{x}{2}\right) = \phi(x) - \frac{1}{2}\phi\left(\frac{x}{2}\right) \tag{7}$$

Equation 6 defines the discrete wavelet transform, for a resolution level j.

A series expansion of the original signal, c_0, in terms of the wavelet coefficients is now given as follows. The final smoothed signal is added to all the differences: for any time t,

$$c_{0,t} = c_{J,t} + \sum_{j=1}^{J} w_{j,t} \tag{8}$$

Note that in time series, the function f is known only through the series $\{X_t\}$, which consists of discrete measurements at fixed intervals. It is well known that $c_{0,t}$ can be satisfactorily approximated by X_t, see e.g. [22].

Equation 8 provides a reconstruction formula for the original signal. At each scale j, we obtain a set, which we call a wavelet scale. The wavelet scale has the same number of samples as the signal, i.e. it is redundant, and decimation is not used.

Equation 4 also is relevant for the name of this transform (“with holes”: [12]). Unlike widely used non-redundant wavelet transforms, it retains the same computational requirement (linear, as a function of the number of input values). Redundancy (i.e. each scale having the same number of samples as the original signal) is helpful for detecting fine features in the detail signals since no aliasing biases arise through decimation. However this algorithm is still simple to implement and the computational requirement is $O(N)$ per scale. (In practice the number of scales is set as a constant.)

Our application – prediction of the next value – points to the critical importance for us of the final values. Our time series is finite, of size say N, and values at times N, $N-1$, $N-2$, ..., are of greatest interest for us. Any symmetric wavelet function is problematic for the handling of such a boundary (or edge). We cannot use wavelet coefficients if these coefficients are calculated from unknown future data values. An asymmetric filter would be better for dealing with the edge of importance to us. Although we can hypothesize future data based on values in the immediate past, there is nevertheless discrepancy in fit in the succession of scales, which grows with scale as larger numbers of immediately past values are taken into account.

In addition, for both symmetric and asymmetric functions, we have to use some variant of the transform that handles the edge problem. It can be the mirror or periodic border handling or the transformation of the border wavelets and scaling functions as in [5]. Although all these methods work well in a lot of applications, they are very problematic in prediction applications as they add artifacts in the most important part of the signal: its right border values.

The only way to alleviate this boundary problem is to use a wavelet basis that does not suffer from it, namely the Haar system.

On the other side, the first values of our time series, which also constitute a boundary, may be arbitrarily treated as a consequence, but this is of little importance.

C. The Redundant Haar Wavelet Transform

The Haar wavelet transform was first described in the early years of the 20th century and is described in almost every text on the wavelet transform. The asymmetry of the wavelet function used makes it a good choice for edge detection, i.e. localized jumps. The usual Haar wavelet transform, however, is a decimated one. We now develop a non-decimated or redundant version of this transform.

The non-decimated Haar algorithm is the à trous algorithm described in the previous section, with a low-pass filter h equal to $(\frac{1}{2}, \frac{1}{2})$. See Figure 1. Here h is non-symmetric. Consider the creation of the first wavelet resolution level. We have created it by convolving the original signal with h. Then:

$$c_{j+1,t} = 0.5(c_{j,t-2^j} + c_{j,t}) \tag{9}$$

and

$$w_{j+1,t} = c_{j,t} - c_{j+1,t} \tag{10}$$

At any time point, t, we never use information after t in calculating the wavelet coefficient. Figure 1 shows which time steps of the input signal are used to calculate the last wavelet coefficient in the different scales. A wavelet coefficient at a position t is calculated from the signal samples at positions less than or equal to t, but never larger.

Because we do not shift the signal, the wavelet coefficients at any scale j of the signal $(X_1, \ldots, X_t)$ are strictly equal to the first t wavelet coefficients at scale j of the signal $(X_1, \ldots, X_N)$, $N > t$.

This is convenient in practice. For instance, if the data are regularly updated (i.e. we get new measurements), we do not have to recompute the wavelet transform of the full signal.

D. Signal Denoising

Many filtering methods have been proposed in the last ten years. *Hard thresholding* consists of setting to 0 all wavelet coefficients which have an absolute value lower than a threshold λ_j and *soft thresholding* consists of replacing each wavelet coefficient by the value $\tilde{w}$ where

$$\tilde{w}_{j,t} = \begin{cases} sgn(w_{j,t})(\mid w_{j,t} \mid - \lambda_j) & \text{if } \mid w_{j,t} \mid \geq \lambda_j \\ 0 & \text{otherwise} \end{cases}$$

Different threshold values have been proposed such as the universal threshold [8], [9] $\lambda_j = \sqrt{2\log(N)}\sigma_j$, where N is the number of time steps in the input data, the SURE (Stein unbiased risk estimator) threshold [10], or the MULTI-SURE threshold. (The SURE method is applied independently on each band of the wavelet transform). The multiscale entropy filtering method [20], [21] (MEF) consists of measuring the information h_W relative to wavelet coefficients, and of separating this into two parts h_s, and h_n. The expression h_s is called the signal information and represents the part of h_W which is definitely not contaminated by the noise. The expression h_n is called the noise information and

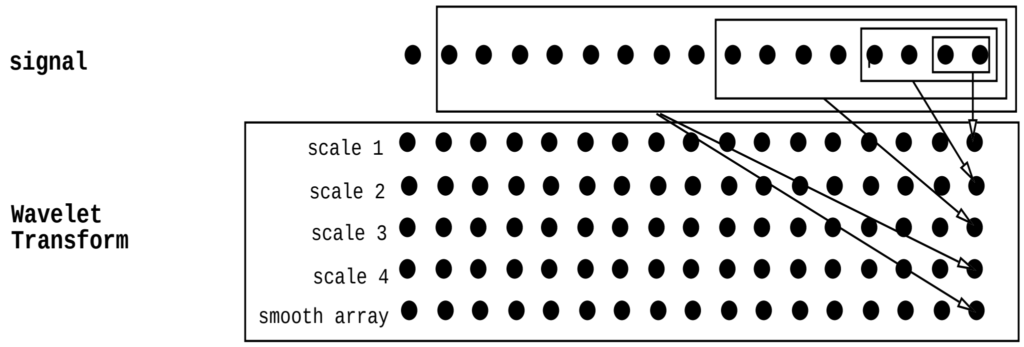


Fig. 1. Redundant Haar wavelet transform: This figure shows which time steps of the input signal are used to calculate the last wavelet coefficient in the different scales.

represents the part of h_W which may be contaminated by the noise. We have $h_W = h_s + h_n$. Following this notation, the corrected coefficient $\tilde{w}_j$ should minimize:

$$J(\tilde{w}_j) = h_s(w_j - \tilde{w}_j) + \alpha h_n(\tilde{w}_j - w_m) \qquad (11)$$

where w_m is a prior ($w_m = 0$ in the general case). I.e. there is a minimum of information in the residual ($w_j - \tilde{w}_j$) which can be due to the significant signal, and a minimum of information which could be due to the noise in the solution, $\tilde{w}_j$.

In order to verify a number of properties, the following functions have been proposed for h_s and h_n in the case of Gaussian noise [21]:

$$\begin{aligned} h_s(w_j) &= \frac{1}{\sigma_j^2}\int_0^{|w_j|} u \text{ erf}\left(\frac{\mid w_j \mid - u}{\sqrt{2}\sigma_j}\right) du \\ h_n(w_j) &= \frac{1}{\sigma_j^2}\int_0^{|w_j|} u \text{ erfc}\left(\frac{\mid w_j \mid - u}{\sqrt{2}\sigma_j}\right) du \end{aligned} \qquad (12)$$

Simulations have shown [20] that the MEF method produces a better result than the standard soft or hard thresholding, from both the visual aspect and PSNR (peak signal to noise ratio).

The next section will describe how prediction can be achieved using the wavelet coefficients, and section IV will show how the MEF filtering can use this prediction.

III. Prediction using Multiscale Models

A. Introduction

In this section, we use the above decomposition of the signal for prediction. Instead of using the vector of past observations $X = (X_1, \ldots, X_N)$ to predict X_{N+1}, we will use its wavelet transform.

The first task is to know how many and which wavelet coefficients will be used at each scale. A sparse representation of the information contained in the decomposition is the key to addressing this. After some simulations and for theoretical reasons that will become clear, the wavelet and scaling function coefficients that will be used for the prediction at time $N+1$ have the form $w_{j,N-2^j(k-1)}$ and $c_{J,N-2^J(k-1)}$ for positive values of k, as depicted in Figure 2. Note that for each N this subgroup of coefficients is part of an orthogonal transform.

B. Stationary Signal

Assume a stationary signal $X = (X_1, \ldots, X_N)$ and assume we want to predict X_{N+1}. The basic idea is to use the coefficients $w_{j,N-2^j(k-1)}$ for $k = 1, \ldots, A_j$ and $j = 1, \ldots, J$ and $c_{J,N-2^J(k-1)}$ for $k = 1, \ldots, A_{J+1}$ (see Figure 2) for this task.

We focus on an autoregressive type of prediction, but generalization to virtually any type of prediction is not difficult.

Recall that to minimize the mean square error, the one-step forward prediction of an AR(p) process is written $\hat{X}_{N+1} = \sum_{k=1}^{p} \hat{\alpha}_k X_{N-(k-1)}$, where $\hat{\alpha}_k$ is an estimated weight.

In order to use the wavelet decomposition, we modify the prediction to give AR multiscale prediction (MAR):

$$\begin{aligned} \hat{X}_{N+1} &= \sum_{j=1}^{J}\sum_{k=1}^{A_j} \hat{a}_{j,k} w_{j,N-2^j(k-1)} \\ &+ \sum_{k=1}^{A_{J+1}} \hat{a}_{J+1,k} c_{J,N-2^J(k-1)} \end{aligned} \qquad (13)$$

where the set $W = \{w_1, \ldots, w_J, c_J\}$ represents the Haar à trous wavelet transform of X. I.e., $X = \sum_{j=1}^{J} w_j + c_J$.

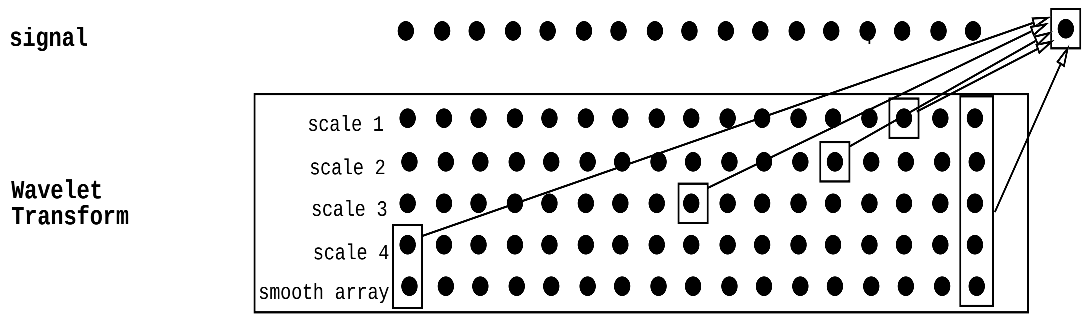


Fig. 2. Wavelet coefficients that are used for the prediction of the next value. Redundant Haar wavelet transform used.

Figure 2 shows which wavelet coefficients are used for the prediction using $A_j = 2$ for all resolution levels j, and a wavelet transform with five scales (four wavelet scales + the smoothed signal). In this case, we can see that only ten coefficients are used, including coefficients that take into account low-resolution information. This means that long-term prediction can easily be introduced, either by increasing the number of scales in the wavelet transform, or by increasing the AR order in the last scales, but with a very small additional number of parameters.

The choice of using lagged wavelet coefficients (every other coefficient on the first scale, each fourth coefficient on the second scale, and so on) is based both on simulation results that tend to show that this schema gives quite good results, and on a few theoretical results. We will now look at these reasons in more detail.

The first reason is completeness: if we let all the A_j grow, the selected coefficients form an orthogonal basis for all the past values, so for large A_j, these coefficients are just sufficient to convey all the information on the series. If the series has some Markov property (i.e. best prediction can be achieved knowing only a finite number of past values, like e.g. the case of AR processes), then for a finite value for the A_j, the selection coefficients can achieve the optimal prediction.

The second reason is parsimony. As stated, the selected coefficients are part of a basis, but the aim is to select through the choice of the A_j just enough so that we have the minimum number of coefficients to ensure a good prediction, with the proviso that we have as few coefficients as possible to estimate (variance-bias trade-off). These coefficients can be viewed as parsimonious summary statistics on the past values of the series.

This choice is an extremely important point in the proposed algorithm and we differ here from [19], where these authors use all the coefficients $w_{j,N-(k-1)}$ and $c_{J,N-(k-1)}$ (no lagging) for k running from 1 to a given value K. The idea of lagging on the other hand is the same as in [14], although we can note a few differences relative to the algorithm in [14]: there the values of the A_j are bound and fixed to be of the form 2^{J-j} but soft thresholding is included in the algorithm. Finally a referee proposed to view the selection of coefficients as a problem of model selection. This is an interesting idea that is worth pursuing, but our objective is to provide an easy and very fast algorithm for the prediction, since it will be included in a more complex structure for filtering.

To further link this method with prediction based on a regular AR, note that if on each scale the lagged coefficients follow an AR(A_j), the addition of the predictions on each level would lead to the same prediction expression 13.

If we use the linear form given in expression 13 for prediction, then the theory of autoregressive models can be applied here. This means that according to [4] (Theorems 8.1.1, 8.1.2, 8.10.1 and 10.8.2), the estimating of the parameters $a_{j,k}$ by MLE (maximum likelihood estimation), by Yule-Walker, or by least squares, has the same asymptotic efficiency. In our programs, to estimate the $Q = \{A_j, j = 1, \ldots J+1\}$ unknown parameters, we used the least squares approach: we minimize the sum of squares of the differences between the prediction as in expression 13 and the actual value X_{N+1} over all the values of N in the training sample time.

The AR order at the different scales must now be defined. A global optimization of all A_j parameters would be the ideal method, but is too computer intensive. However, by the relative non-overlapping frequencies used in each scale, we can consider selecting the parameters A_j independently on each scale. This can be done by standard methods, based on AIC, AICC or BIC methods [18].

C. Non-linear and Non-stationary Generalizations

This Multiresolution AR prediction model is actually linear. To go beyond this, we can imagine using virtually any type of prediction, linear or non-linear, that uses the previous data $X_N, \dots, X_{N-q}$ and generalize it through the use of the coefficients $w_{j,t}$ and $c_{J,t}$, $t \leq N$ instead.

One example is to feed a multilayer perceptron neural network with these coefficients w and c as inputs, use one or more hidden layer(s), and obtain X_{N+1} as the (unique) output, as has been done in [15]. In this case, a backpropagation algorithm can be used for the estimation of the parameters.

We can also think of using a model that allows for some form of non-stationarity. For example we could use an ARCH or GARCH model on each scale to model the conditional heteroscedasticity often present in financial data.

Finally, if we stay with the linear MAR, it is easy to change slightly the algorithm to handle signal with a piecewise smooth trend. We show in [16] how to extend this method, and to take advantage of the last smooth signal to estimate it.

D. Assessments on Real Data

Using financial futures, we used a set of 6160 highs. The first half of the time series was taken as training data, and the second half as test data. Using BIC, the best AR model was an AR(1) model, giving a standard deviation error on the test set of 13.9613. The best MAR model, again based on BIC, was MAR(1), in line with the illustration of the Haar à trous transform in Figure 2. In this case, the standard deviation error was 13.8951 for a 2-scale wavelet architecture.

In another setting, we took web site accesses, measured as numbers of bytes per hour, from an active web site (run by one of the authors in the 1990s). The first three values were: 2538, 55540, 14691. The last three values were 9152, 438370, 416421. Data was gathered for 34,726 successive hours (or about 4 years' data). Modeling and forecasting of web site accesses [2], just as for other Internet traffic [1], [23], is important for performance evaluation, and for preemptive performance-supporting options such as caching and prefetching.

data	best AR		best MAR	
	order	std.err	order	std.err
Fin. futures	1	13.9613	1,1,1	13.8951
Internet	10	1,019,880	1 (5x)	1,010,330

TABLE I

ORDER SELECTION AND STANDARD ERROR ON TEST SAMPLE OF DIFFERENT METHODS ON REAL DATASETS.

Evaluation on this data used the first half of the time series as training data, and the second half as test data. In all cases, the BIC criterion was used for model selection. A simple autoregressive model was used, with AR(10) selected as best. The prediction error, given by the standard deviation, on the evaluation part (second half) of the time series was 1,019,880.

Next, a multiresolution autoregressive model was used. An MAR(1) model gave the best fit. The prediction error, given by the standard deviation, on the evaluation part (second half) of the time series was 1,010,330, therefore improving on the best AR fit, AR(10).

Other examples and an extended simulation study can be found in [16].

E. Conclusion on Prediction

This prediction method, that is a very flexible procedure, permits capturing of short-range as well as long-range dependencies with only a few parameters. In its simplest form this method is a generalization of the standard AR method: instead of a linear prediction based on past values, we use a linear prediction based on some coefficients of the decomposition of the past values. Our proposed method can easily be extended to generalize more sophisticated methods such as GARCH. The concept is very simple and easy to implement, while the potential is clear. We will see in the following how the prediction approach can be extended to the case of noisy data.

IV. Filtering using the Multiscale Prediction

A. Introduction

In this section, we show how to execute the prediction and filtering in the wavelet domain to take advantage of the decomposition. We will consider stochastic signals measured with an error noise. Therefore the type of equations that are believed to drive the signal can be expressed as the following measurement and transition equations:

$$\mathbf{y}_{N+1} = Z_{N+1}\mathbf{x}_{N+1} + \mathbf{v}_{N+1} \tag{14}$$

$$\mathbf{x}_{N+1} = \text{function}(\mathbf{x}_N) + \epsilon_{\mathbf{N+1}}, \tag{15}$$

where the $\mathbf{v}_t$ are IID $\mathcal{N}(0, \Sigma_v)$, the $\epsilon_{\mathbf{t}}$ are zero-mean noise, and the $\mathbf{v}_t$ are independent of the $\epsilon_{\mathbf{t}}$. The process $\{\mathbf{x}_t\}$ has a stochastic behavior, but we measure it only through $\{\mathbf{y}_t\}$, which is essentially a noisy version of it. Data that seem to follow this kind of equation can be found in many different fields of science, since error in measurement is more the rule than the exception. The aim is clearly to best predict the future value of the underlying process $\mathbf{x}_{N+1}$, based on the observed values $\mathbf{y}$.

Solving these equations and finding the best prediction and the best filtered value at each time point seem difficult and computer intensive. However in a special case of equations 14 and 15, the Kalman filter gives rise to an algorithm that is easy to implement and that is very efficient, since the future predicted and filtered values are only based on the present values, and depend on the past values only through a pair of matrices that are updated at each time point. This method supposes in addition to the previous equation that $\mathbf{x}_{N+1} = T_{N+1}\mathbf{x}_N + \epsilon_{\mathbf{N+1}}$, that both errors are Gaussian, $\mathbf{v}_t$ are $\mathcal{N}(0, \Sigma_v)$ and $\epsilon_{\mathbf{t}}$ are $\mathcal{N}(0, \Sigma_e)$, and that Z_t, Σ_v, T_t and Σ_e are known. The Kalman filter gives a recursive construction of the predicted and filtered values which has optimal properties. The predicted $(\mathbf{p})$ and filtered $(\mathbf{f})$ values of $\mathbf{x}_{N+1}$ are given by the coupled recurrence:

$$\mathbf{p}_{N+1} = T_{N+1}\mathbf{f}_N \tag{16}$$

$$\mathbf{f}_{N+1} = \mathbf{p}_{N+1} + K_{N+1}(\mathbf{y}_{N+1} - Z_{N+1}\mathbf{p}_{N+1}) \tag{17}$$

where K_{N+1} is the gain matrix that is also given by a recursive scheme and depends on Z_{N+1}, Σ_v, T_{N+1} and Σ_e (see [11], [17]). If $\mathbf{v}_t$ and $\epsilon_{\mathbf{t}}$, $t \leq N+1$, are Gaussian, $\mathbf{p}_{N+1}$ is the best prediction in the sense that it minimizes the mean square error given the past values $(= E(\mathbf{x}_{N+1}|\mathbf{y}_1, \dots, \mathbf{y}_N))$ and $\mathbf{f}_{N+1}$ is the best filter in the sense that it minimizes the mean square error given all values up to $N+1$ $(= E(\mathbf{x}_{N+1}|\mathbf{y}_1, \dots, \mathbf{y}_{N+1}))$. If the errors are not Gaussian, the estimators are optimal within the class of linear estimators, see [11].

Under Gaussianity of the errors, if Σ_v, T_t and Σ_e are known only up to a given number of parameters, the maximum likelihood of the innovations can be stated and either a Newton-Raphson or an EM (expectation-maximization) algorithm can be used to minimize the likelihood which is severely non-linear in the unknown parameters, and gives estimates for these parameters.

B. Filtering

In this work we propose a method similar to the Kalman filter but that allows two generalizations on the type of equation that can be treated. The first one is to allow for different functions in the transition equation 15. The versatility of the wavelet transform and the freedom to choose the prediction form allow for very general function approximation. The second generalization is to allow the noise of the transition equation $\epsilon_{\mathbf{t}}$ to be non-Gaussian and even to have some very large values. Using multiscale entropy filtering, these cases will be detected and the filtering will not be misled.

Like in the Kalman filter, we define a recurrence scheme to obtain the predicted values and the filtered values. This allows us to have a fast algorithm that does not have to recompute all estimations at each new value. However, the recurrence equations will not be linear as in the Kalman filter and they will not be based on the values themselves, but will be carried out in the wavelet domain.

For notational convenience, we present our method for a simplified equation where $\mathbf{x}_t$ is unidimensional and Z_t is equal to 1 for all $t \leq N$. For the prediction part, instead of using the vector of past filtered values $\mathbf{f} = (f_1, \dots, f_N)$ to predict p_{N+1}, as in 16, we will use its wavelet transform.

It has been shown previously that the wavelet and scaling function coefficients that must be used for the prediction at time $N+1$ have the form $w^f_{j,N-2^j(k-1)}$ and $c^f_{J,N-2^J(k-1)}$ for positive values of k. Given the wavelet decomposition of the filtered values f_t, w^f and c^f, we predict the next value based on the Multiresolution Autoregressive (MAR) model

$$p_{N+1} = \sum_{j=1}^{J} \sum_{k=1}^{A_j} \hat{a}_{j,k} w^f_{j,N-2^j(k-1)} + \sum_{k=1}^{A_{J+1}} \hat{a}_{J+1,k} c^f_{J,N-2^J(k-1)} \quad (18)$$

where the set $W = \{w^f_1, \dots, w^f_J, c^f_J\}$ represents the Haar à trous wavelet transform of f, and we have: $f = \sum_{j=1}^{J} w^f_j + c^f_J$. Again, depending on the supposed process that generated the data, we can imagine other prediction equations based on these coefficients, like Multiresolution Neural Network (MNN), Multiresolution GARCH, and so on.

This first part of the algorithm has the same role as equation 16 for the Kalman filter. Given the filtered values up to time N , f_t, $t = 1, \dots, N$, and its Haar à trous wavelet transform $w^f_{j,t}$ for $t = 1, \dots, N$ and $j = 1, \dots, J$, and $c^f_{J,t}$ for $t = 1, \dots, N$, we use the wavelet decomposition of f_t to predict the next value p_{N+1}. The difference with Kalman is the form of the prediction: it is based on the multiresolution decomposition and might be non-linear if so desired.

For the second part of the algorithm, like the Kalman filtering equation 17, we will compare the observed value y_{N+1} and the predicted value p_{N+1}, but in the wavelet domain. From the predicted value p_{N+1} we decompose in the wavelet domain (with – some of – the previous filtered values up to f_N) to obtain the $J+1$ coefficients $w^p_{j,N+1}$ for $j = 1, \dots, J$ and $c^p_{J,N+1}$. For the filtering part, we first decompose the new observation y_{N+1} in the wavelet domain (using also the previous observations y_t) to obtain the $J+1$ coefficients $w^y_{j,N+1}$ for $j = 1, \dots, J$ and $c^y_{J,N+1}$.

Informally, the Kalman filter computes a filtered value f_{N+1} on the basis of the predicted value p_{N+1} and corrects it only if the new observation is far from its predicted value. Here we work on the wavelet coefficients and we will also set the filtered value as close to the predicted value, unless the prediction is far from the actual coefficient. To carry out the compromise, we use the multiscale entropy given in expression 11, and tailor it to the situation. The wavelet coefficient of the filtered value will be the one that satisfies

$$\min_{w^f_{j,N+1}} h_s\left(\frac{w^y_{j,N+1} - w^f_{j,N+1}}{\sigma_v}\right) + \lambda h_n\left(\frac{w^f_{j,N+1} - w^p_{j,N+1}}{\sigma_e}\right) \quad (19)$$

and similarly for the smooth array $c^f_{J,N+1}$. The coefficient $w^f_{j,N+1}$ must be close to the same coefficient for the measured signal y and at the same time close to the predicted value $w^p_{j,N+1}$. Since the standard errors of the noise v_t and ϵ_t in equations 14 and 15 can be very different, we have to standardize both differences in 19. Note that λ plays the role of the trade-off parameter between the prediction and the new value. This can be viewed as the thresholding constant in the regression context. Having obtained the filtered coefficients for all scales, we can simply recover f_{N+1} with equation 8.

This algorithm with the Multiresolution AR is of the same complexity as the Kalman filter, since the à trous transform is of linear order. If one considers the number of scales as a (necessarily logarithmic) function of the number of observations, an additional factor of $\log(N)$ is found in the complexity. The key to this lies in the fact that we do not recompute the entire à trous transform of f, p or y each time a new value is added, but we compute only the $J+1$ coefficients $w_{j,t}$ for $j = 1, \dots, J$ and $c_{J,t}$ at a given time t. If one uses a more complex prediction scheme, like a neural network, then the complexity of our algorithm is driven by this prediction algorithm. The proposed algorithm is also simple, thanks to the Haar transform (equations 8, 9 and 10) which is especially easy to implement.

The estimation of the unknown parameters follows the same lines as for prediction, although the estimation of the noise levels is known to be a more difficult task. The key to the performance of the method in the pure prediction case was to allow it to adapt the number A_j of coefficients kept at each scale for the prediction. This was done using a penalization criterion such as BIC. We believe that it is important to do exactly the same in the filtering framework. We refer to Section III for these parameters. The estimation of σ_v, σ_e and λ is more delicate, and in our programs, we leave the choice to the user either to provide these values or to let them be estimated. Of course, when the true values for the standard deviations are known independently, the method is more effective, but we note that even if the algorithm over- or underestimates these values up to 50%, performance is still good.

First, the optimal λ is strongly related to the two standard deviations, and more precisely to their ratio. In our simulations, we set λ to be $0.1\sigma_v/\sigma_e$.

The leading parameter here is σ_v, since it drives all the smoothing, and once this parameter has been set or estimated, the procedure proceeds as in the prediction case. If the program has to estimate this parameter, a grid of 30 different values for σ_v is tried, from zero to the standard deviation of the observed data y. The resulting models are compared on the innovation error and we select the one that minimizes the error. We could use a more complex algorithm that refines the grid close to the minimal values, but we do not believe that this will improve the model in a significant way. On the contrary, we have found that this method is relatively robust to the parameter estimation and that there is an interval of values that leads to the same standard deviation of error.

V. Simulations

A. Experiment 1

In this section, we use a Monte-Carlo study to compare the proposed filtered method with the regular Kalman filter. The first two simulation studies have models where Kalman is known to be optimal. The aim is to see whether the multiresolution approach is able to get close to this "gold standard" in this special case. The second part uses a much more difficult model (ARFIMA plus noise) where Kalman is not optimal any more. We will be interested in the deterioration of performance of both methods.

In the first case, we generated processes $\{X_t\}$ that follow a causal AR process of order p, which means that they satisfy $X_t = \phi_1 X_{t-1} + \cdots + \phi_p X_{t-p} + \epsilon_t$. We measure only Y_t which is a noisy version of this process: $Y_t = X_t + v_t$. Both noises are taken to be Gaussian, i.e. $\{\epsilon_t\}$ are $\mathcal{N}(0, \sigma_e^2)$ and $\{v_t\}$ are $\mathcal{N}(0, \sigma_v^2)$. For any value of p this process can be written in the Kalman form of equations 14 and 15, with a p-dimensional $\mathbf{x}_t$ [11].

We compare 5 different filtering algorithms. First, True Kal is the oracular Kalman filter that has been fed with the true values of all parameters: the AR order p and the values of ϕ_i, σ_e and σ_v. This is the unattainable best filter. Second is Estim.Kal which is a Kalman filter that has been fed with the true AR order p but we have to estimate the values of ϕ_i, σ_e and σ_v. This is done through the likelihood of the innovations [17]. It is well known that the most difficult parameters to estimate are the estimates of both variabilities σ_e and σ_v. So the third method Est.AR.Kal is again a Kalman filter which has been fed with p, σ_e and σ_v, and has only to estimate the ϕ's. Note that these three Kalman methods are fed with the true order p. If they had to select it via a criterion, we could expect significant deterioration in their performance. The fourth method Obser is the simplistic choice not to do any filtering and to provide y_t as being the best estimate for x_t. If a method gives results that are worse than Obser, it has clearly missed the target. It is however not an easy task when σ_v is small. The last approach, MultiRes, is our proposed method, with 5 scales as in prediction, and with the level of the measurement noise that is given. So, contrary to the three Kalman filters, the method is not fed with the true order, but has to select it levelwise with the BIC criterion. However, like Est.AR.Kal, the method is fed with σ_v.

The 50 series are of length 1000 and the first 500 points are used for training and the last 500 are used to compare the standard deviation of the error between the filtered and the true values. A boxplot gives in the center the median value for the response and the box contains 50% of the responses, giving an idea of the variability between the samples. Hence the best method is the one for which the boxplot is the most stumpy and compact.

For Figure 3 the AR order is $p = 2$ with parameter $\phi' = (0.5, -0.7)$. The noise levels are as follows: for subplot (a) $\sigma_e = 1$ and $\sigma_v = 1$, for subplot (b) $\sigma_e = 1$ and $\sigma_v = 0.4$, for subplot (c) $\sigma_e = 1$ and $\sigma_v = 0.1$ and finally for subplot (d) $\sigma_e = 0.4$ and $\sigma_v = 1$. In all cases, the True.Kal method should be the best one. In addition, in case (c) where the measurement error is very small, the simple Obser should be competitive or even difficult to beat. This is in fact the case. In addition, the proposed method MultiRes has standard deviations of the errors almost as good as the oracular True.Kal, and is very competitive with Estim.Kal and Est.AR.Kal, which are both based on the correct Kalman filter but have to estimate some parameters.

The second simulation, the results of which are

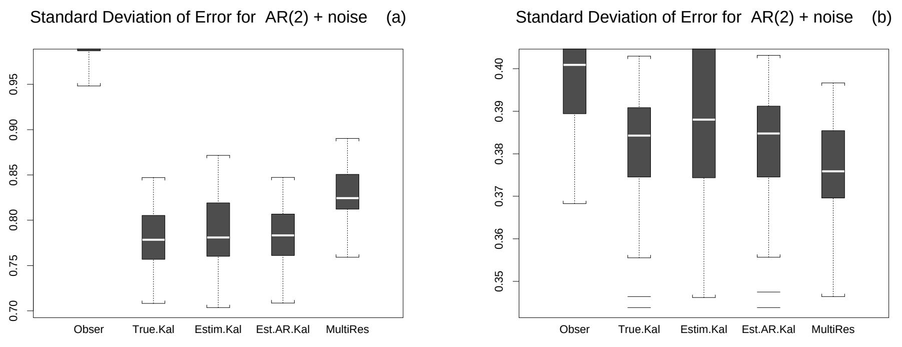
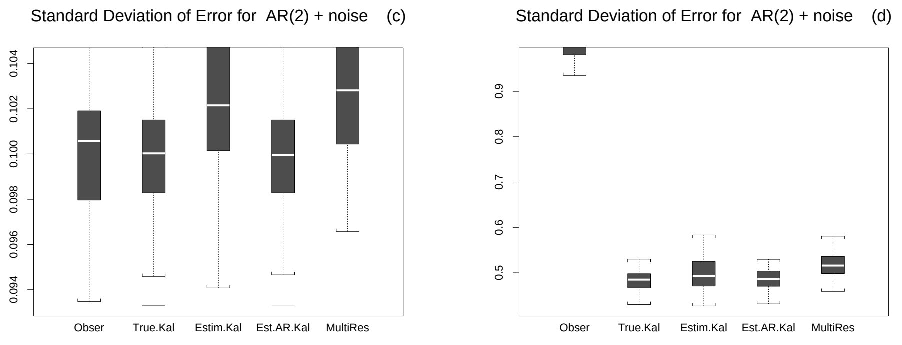



Fig. 3. Boxplots of the standard deviation of the error for an AR(2) plus noise model. The noise levels are $\sigma_e = 1$ and $\sigma_v = 1$ for (a), $\sigma_e = 1$ and $\sigma_v = 0.4$ for (b), $\sigma_e = 1$ and $\sigma_v = 0.1$ for (c) and $\sigma_e = 0.4$ and $\sigma_v = 1$ for (d). The proposed method MultiRes is compared with three versions of Kalman filter True.Kal, Estim.Kal and Est.AR.Kal, and to the naive filtering Obser. Although this framework is optimal for the Kalman filter, the proposed method is very competitive.

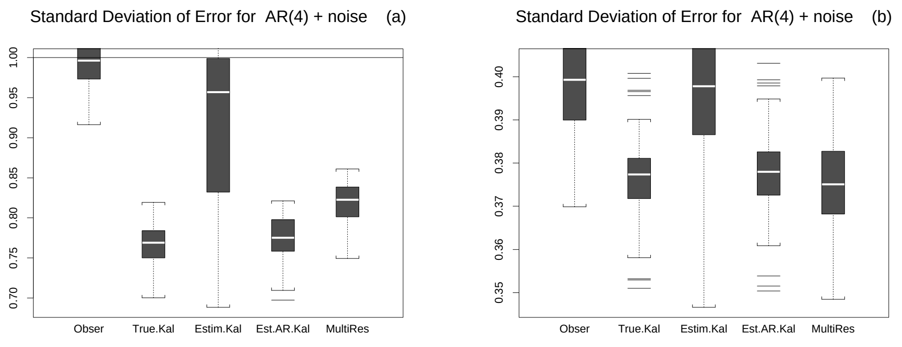
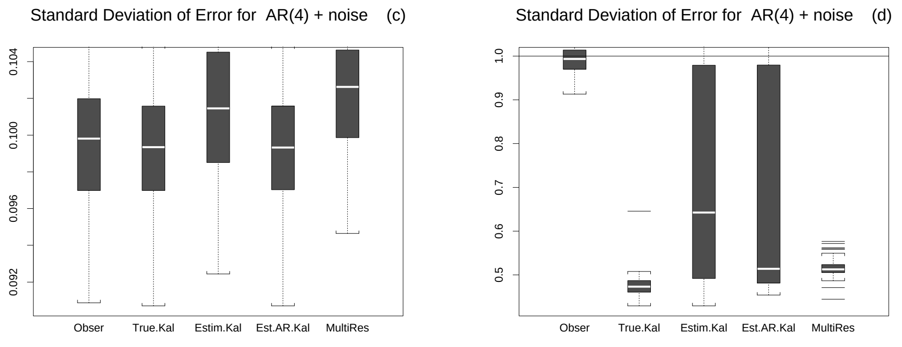



Fig. 4. Same setting as that of Figure 3, except that the model is AR(4) plus noise. Depending on the noise levels, the proposed method is either as good as or even better than the Kalman methods that have to estimate a few parameters.

shown in Figure 4, is an AR(4) plus noise, with parameter set $\phi' = (0.5, -0.5, -0.1, 0.3)$. The standard deviations for both noises are the same four couples as for Figure 3. As expected again, True.Kal is the best, but unattainable, method. However, since the number of parameters to estimate here has increased, we see a major deterioration in the performance of Estim.Kal in three out of four cases, (a), (b) and (d). This deterioration takes the form of a much wider range of errors (large boxplot) and of an average performance that is poorer (higher center of the boxplot). Est.AR.Kal performs well except in case (d) where the process noise is small $\sigma_e = 0.4$. It is remarkable that the proposed method MultiRes works well in all cases and still has compact boxplots and therefore consistently good results. It shows that this method adapts easily to different orders of the underlying process and to different values for both noises.

The last type of signal (Figure 5) is a fractional ARIMA$(1, 0.49, 0)$, with parameter $\phi_1 = -0.5$ and with a unit Gaussian innovation. The measurement error standard deviation is $\sigma_v = 1$ for subplot (a), $\sigma_v = 0.4$ for subplot (b) and $\sigma_v = 0.1$ for subplot (c). Our software does not allow us to vary the process noise level in this case, so we were not able to simulate the equivalent of subplot (d) of the two previous graphs. This type of process is more challenging to filter, since it is an example of so-called long memory or $1/f$ process. It cannot be put in a Kalman filter form, so there is no oracular True.Kal. However, we still can estimate the parameters of a Kalman filter that best approaches this model. This has been done for Estim.Kal and Est.AR.Kal, with an AR(4) plus noise structure. These are not optimal in the statistical sense. As in the two previous cases, the proposed method MultiRes can select the order on each scale with a BIC criterion. The boxplots in Figure 5 depict the results, and we see that MultiRes convincingly outperforms all other methods, which is again a demonstration that it can automatically adapt easily to a wide range of processes.

In conclusion, the proposed method seems to be very adaptive to the underlying process of the type of model considered. Equation 15 stated loosely that $\mathbf{x}_{N+1}$ must be some function of the past. Contrary to the Kalman filter that is designed (and optimal) for a very special form for this function, the multiresolution approach, empowered with the multiscale entropy filtering, can accommodate a much wider variety of processes.

B. *Experiment 2*

Previous experiments focused on the deterministic part of the underlying process and showed that the proposed method is quite flexible. We now turn to the stochastic part, the process error ϵ_t and show that here again the proposed method is more flexible.

Recall that the Kalman filter is designed for Gaussian error for both the process and the measurement noise. This explains that the best filtered value, as shown in equation 17, is a linear combination of the prediction $\mathbf{p}_{N+1}$ and the observed value $\mathbf{y}_{N+1}$. If a huge value in the process noise arrives, and hence an observed value very different from the prediction, the Kalman filter will still achieve a trade-off, and therefore be off target in the predicted and the filtered values for a few events after this burst (see Figure 6).

The proposed method does not use a linear combination between the observed and predicted value but uses a more sensible non-linear and adaptive scheme through the multiscale entropy filtering, seen in expression 19. Informally, it acts as a threshold towards the predicted value (whereas the threshold is towards zero in the regression context). Hence if the difference between the observed and the predicted values is larger than expected the filtered wavelet coefficients will be quite close to the observed value. If the difference is smaller, the wavelet coefficient will be a larger compromise between them. (See equations 11 and 12.)

An illustration of this difference between Kalman filter and the proposed method is given in Figure 6. The process and measurement is exactly the same as for Figure 3(a), that is to say an AR(2) plus noise model. The only difference is that at the 700th time point, the process noise was increased by an amount of 10 units. We can see the underlying (not observed) process State and the measured values Obser around time 700. The Kalman filter that has been fed with all the true values for the parameters (except this burst) is also depicted (True.Kal) and we see that it does not follow the very large observed value at time 700 but goes only half way. At time 701 it has still not joined the large value, and only when the series changes sign it comes back closer to process and observed values. The estimated

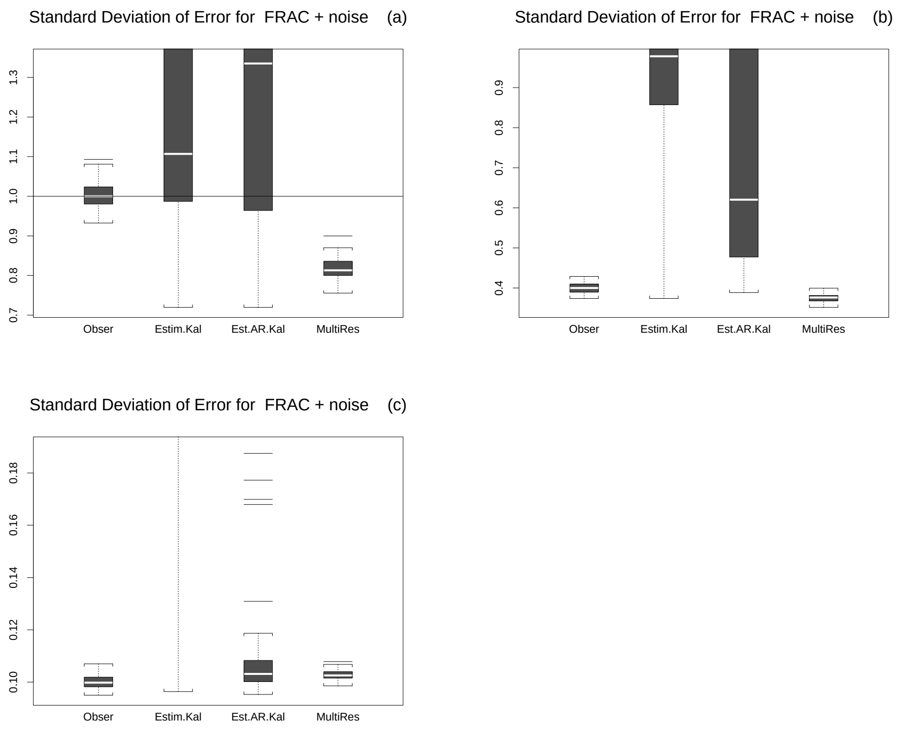


Fig. 5. Same setting as that of Figure 3 (a), (b) and (c), except that the model is a fractional AR plus noise. It is known that here the Kalman filter is not optimal any more, and its performance deteriorates a lot, whereas the proposed method is adapted to the nature of the process and gives very good results.

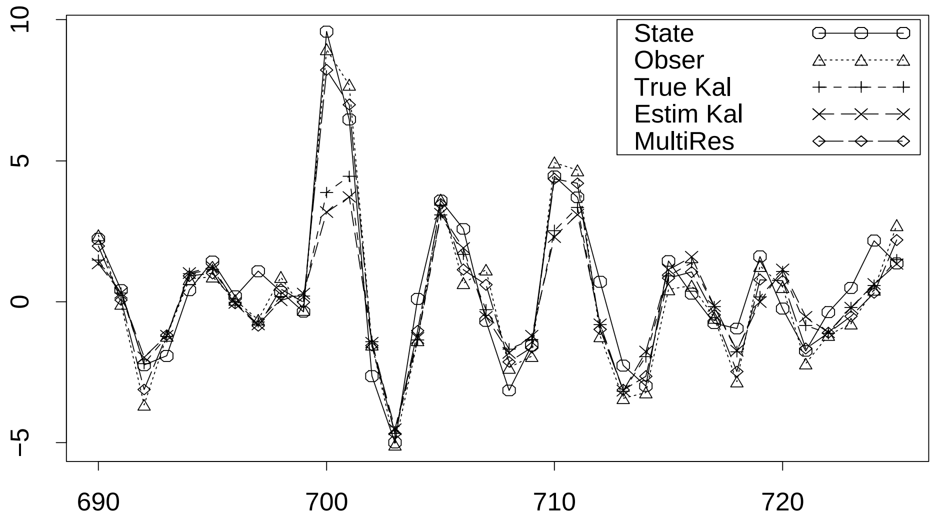


Fig. 6. Behavior of the same methods as in Figure 3 in the case where a burst happens at time 700. State is the unobserved process that has to be estimated. The proposed method MultiRes reacted correctly to the burst and stays close to the state values, whereas the Kalman methods are off target at time 700 and 701.

Kalman filter Estim.Kal has very similar behavior. On the other hand, the proposed method MultiRes detects immediately that something strange has happened at time 700, and decides to follow the observation and not to take into account the prediction. As a result, the proposed method, even if it does not know the equation of the process and even if it has to estimate some parameters, did a much better job than the Kalman filter that does not even have to estimate any parameters.

C. Experiments with Real Data

Assessing the quality of different methods on real data is more complicated than in the pure prediction case of section III. The problem resides in the fact that we want to compare the filtered values with the underlying state time series, which is unknown for real data. To circumvent this difficulty, we provide to the algorithms the real data time series perturbed with Gaussian white noise. Of course, the criteria for each method will be the standard deviation of the difference between the filtered values and the original, unperturbed real values.

data	Obser	best Est.AR.Kal		MAR	
	std.err	order	std.err	order	std.err
Internet	100,140	7	99,379	1 (5x)	89,217
log(Internet)	1.00017	4	0.954	1 (5x)	0.873

TABLE II
STANDARD ERROR OF FILTERING ON TEST SAMPLE OF DIFFERENT METHODS ON REAL DATASETS.

We first compare the methods on the web site access data, as described in subsection III-D. We added to the data a white Gaussian noise with standard deviation of 100,000. The first half of the data is for training and the second half for testing. If we do no filtering at all and use the observed data as our best guess (called Obser in the simulations), the standard error on the second half of the data is 100,140 (close to 100,000, as expected). Any method that does better than that improves the filtering. The best Kalman filter that was found was fed with the true standard deviation of both noises (called Est.AR.Kal in the simulations). It is an AR(7) and gives a standard error on the second part of the data of 99,379, improving slightly the non-filtered data. Fixing the number of wavelet scales to five, and providing the value of the standard deviation of the measurement noise (100,000), the proposed multiresolution method selected only one coefficient per scale and gives a standard error on the second part of the data of 89,217, which is a 10% improvement. Moreover, even if we provide for the measurement noise a value from 50,000 to 300,000, the standard error does not increase by more than 1%, which shows that the method is quite robust.

Note that the above values cannot be compared with the value in the prediction section. Indeed, in the filtering case the value to be predicted is not a future value, but the actual value of the unobserved state time series.

Finally, since the data are very asymmetric, we also wanted to test the different methods on the logarithm of the data. The log values look quite symmetric and Gaussian, and maybe more suited for the Kalman filter. In this case, we added a Gaussian white noise of unit standard deviation. Taken on the second half of the data, the standard deviation of the difference between the observed data and the real (log) data is 1.00017. The best Kalman (with the standard error of both noises provided) is an AR(4) and has an error of 0.954. With a fixed 5-scale architecture, our proposed method has an error of 0.873, again improving on the foregoing results. Additionally, the results are robust to the provided standard error of measurement: the error varies less than 2% for the range of values between 0.5 and 3.

VI. CONCLUSION

We have presented in this article a new method for the prediction of time series and for filtering processes that are measured with noise. It is based on a special (overcomplete) wavelet decomposition of the signal called the à trous wavelet transform. In the case of prediction, virtually any prediction scheme can be adapted to the multiresolution representation induced by the transformation, but even with the simplest scheme of all, the autoregressive model, this method is able to capture short and long memory components in an adaptive and very efficient way. In the filtering case, the same prediction scheme can be used and the use of multiscale entropy filtering instead of the usual trade-off inherent in the Kalman filter is powerful and has several advantages. This method is competitive in the cases where the Kalman filter is known to be optimal, but it is much more useful when the transition equation is not linear any more. Moreover, the multiscale entropy filtering is robust relative to Gaussianity of the transition noise.

References

[1] P. Abry, D. Veitch, and P. Flandrin. Long-range dependence: revisting aggregation with wavelets. *Journal of Time Series Analysis*, 19:253–266, 1998.

[2] A. Aussem and F. Murtagh. A neuro-wavelet strategy for web traffic forecasting. *Journal of Official Statistics*, 1:65–87, 1998.

[3] Z. Bashir and M.E. El-Hawary. Short term load forecasting by using wavelet neural networks. In *Canadian Conference on Electrical and Computer Engineering*, pages 163–166, 2000.

[4] P.J. Brockwell and R.A. Davis. *Time Series: Theory and Methods*. Springer-Verlag, 1991.

[5] A. Cohen, I. Daubechies, and J. Feauveau. Bi-orthogonal bases of compactly supported wavelets. *Comm. Pure and Appl. Math.*, 45:485–560, 1992.

[6] R. Cristi and M. Tummula. Multirate, multiresolution, recursive Kalman filter. *Signal Processing*, 80:1945–1958, 2000.

[7] K. Daoudi, A.B. Frakt, and A.S. Willsky. Multiscale autoregressive models and wavelets. *IEEE Transactions on Information Theory*, 45(3):828–845, 1999.

[8] D.L. Donoho. Nonlinear wavelet methods for recovery of signals, densities, and spectra from indirect and noisy data. In *Proceedings of Symposia in Applied Mathematics*, volume 47, pages 173–205. American Mathematical Society, 1993.

[9] D.L. Donoho and I.M. Johnstone. Ideal spatial adaptation via wavelet shrinkage. *Biometrika*, 81:425–455, 1994.

[10] D.L. Donoho and I.M. Johnstone. Adapting to unknown smoothness via wavelet shrinkage. *Journal of the American Statistical Association*, 90:1200–1224, 1995.

[11] A. C. Harvey. *Forecasting, Structural Time Series Models, and the Kalman Filter*. Cambridge University Press, 1990.

[12] M. Holschneider, R. Kronland-Martinet, J. Morlet, and Ph. Tchamitchian. A real-time algorithm for signal analysis with the help of the wavelet transform. In J.M. Combes, A. Grossmann, and Ph. Tchamitchian, editors, *Wavelets, Time-Frequency Methods and Phase Space*, pages 286–297. Springer-Verlag, 1989.

[13] L. Hong, G. Chen, and C.K. Chui. A filter-bank based Kalman filter technique for wavelet estimation and decomposition of random signals. *IEEE Transactions on Circuits and Systems – II Analog and Digital Signal Processing*, 45(2):237–241, 1998.

[14] U. Lotrič. Wavelet based denoising integrated into multilayered perceptron. *Neurocomputing*, 62:179–196, 2004.

[15] F. Murtagh, J.-L. Starck, and O. Renaud. On neuro-wavelet modeling. *Decision Support Systems*, 37:475–484, 2004.

[16] O. Renaud, J.-L. Starck, and F. Murtagh. Prediction based on a multiscale decomposition. *International Journal of Wavelets, Multiresolution and Information Processing*, 1(2):217–232, 2003.

[17] R.H. Shumway. *Applied Statistical Time Series Analysis*. Prentice-Hall Inc, 1988.

[18] R.H. Shumway and D.S. Stoffer. *Time Series Analysis and Its Applications*. Springer-Verlag, 1999.

[19] S. Soltani, D. Boichu, P. Simard, and S. Canu. The long-term memory prediction by multiscale decomposition. *Signal Processing*, 80:2195–2205, 2000.

[20] J.-L. Starck and F. Murtagh. Multiscale entropy filtering. *Signal Processing*, 76:147–165, 1999.

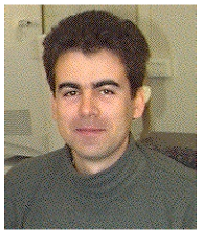
[21] J.-L. Starck, F. Murtagh, and R. Gastaud. A new entropy measure based on the wavelet transform and noise modeling. *IEEE Transactions on Circuits and Systems II*, 45:1118–1124, 1998.

[22] M. Vetterli and J. Kovačević. *Wavelets and Subband Coding*. Prentice Hall PTR, New Jersey, 1995.

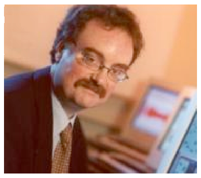
[23] W. Willinger, M. Taqqu, W.E. Leland, and D. Wilson. Self-similarity in high-speed packed traffic: analysis and modeling of ethernet traffic measurements. *Statistical Science*, 10:67–85, 1995.

[24] G. Zheng, J.-L. Starck, J. Campbell, and F. Murtagh. The wavelet transform for filtering financial data streams. *Journal of Computational Intelligence in Finance*, 7(3):18–35, 1999.

Olivier Renaud received the MSc degree in Applied Mathematics and the PhD degree in Statistics from Ecole Polytechnique Fédérale (Swiss Institute of Technology), Lausanne, Switzerland. He is currently Maître d'Enseignement et de Recherche in Data Analysis, University of Geneva. He earned a one-year fellowship at Carnegie-Mellon University, Pittsburgh, PA, and was also visiting scholar at Stanford University, Stanford, CA for a year. His research interests include non-parametric statistics, wavelet-like methods and machine learning.



Jean-Luc Starck has a Ph.D from University Nice-Sophia Antipolis and an Habilitation from University Paris XI. He was a visitor at the European Southern Observatory (ESO) in 1993 and at Stanford's Statistics Department in 2000 and 2005. He is a Senior Researcher at CEA. His research interests include image processing, multiscale methods and statistical methods in astrophysics. He is author of the books *Image Processing and Data Analysis: the Multiscale Approach* (Cambridge University Press, 1998), and *Astronomical Image and Data Analysis* (Springer, 2002).



Fionn Murtagh holds BA and BAI degrees in mathematics and engineering science, and an MSc in computer science, all from Trinity College Dublin, Ireland, a PhD in mathematical statistics from Université P. & M. Curie, Paris VI, France, and an Habilitation from Université L. Pasteur, Strasbourg, France. He is Professor of Computer Science in the University of London at Royal Holloway. He is Editor-in-Chief of The Computer Journal, a Member of the Royal Irish Academy, and a Fellow of the British Computer Society.