Computational methods for hidden Markov tree models - An application to wavelet trees.

Jean-Baptiste Durand, Paulo Goncalvès, Yann Guédon

Computational Methods for Hidden Markov Tree Models – An Application to Wavelet Trees

Jean-Baptiste Durand [*] Paulo Gonçalvès [†] Yann Guédon [‡]

August 26, 2004

Abstract: *The hidden Markov tree models were introduced by Crouse, Nowak and Baraniuk in 1998 for modeling nonindependent, non-Gaussian wavelet transform coefficients. In their article, they developed the equivalent of the forward-backward algorithm for hidden Markov tree models and termed it the "upward-downward algorithm". This algorithm is subject to the same numerical limitations as the forward-backward algorithm for hidden Markov chains. In this paper, adapting the ideas of Devijver from 1985, we propose a new "upward-downward" algorithm, which is a true smoothing algorithm and is immune to numerical underflow. Furthermore, we propose a Viterbi-like algorithm for global restoration of the hidden state tree. The contribution of those algorithms as diagnosis tools is illustrated through the modeling of statistical dependencies between wavelet coefficients with a special emphasis on local regularity changes.*

Keywords: *hidden Markov tree model, EM algorithm, hidden state tree restoration, upward-downward algorithm, wavelet decomposition, scaling laws, change detection*

1 Introduction

The hidden Markov tree models (HMT) were introduced by Crouse, Nowak and Baraniuk (1998) [6]. The context of their work was the modeling of statistical dependencies between wavelet coefficients in signal processing, for which observations are organized in a tree structure. Applications of such models are: image segmentation, signal classification, denoising and image document categorization; see Choi and Baraniuk (1999) [4], and Diligenti, Frasconi and Gori (2001) [9]. Dasgputa *et al.* (2001) [7] used a mixture of

[*] INRIA Rhône-Alpes, Montbonnot, France - email : Jean-Baptiste.Durand@inrialpes.fr
[†] INRIA Rhône-Alpes, Montbonnot, France - email : Paulo.Goncalves@inrialpes.fr
[‡] CIRAD, Montpellier, France - email : guedon@cirad.fr

hidden Markov trees with a Markovian regime for target classification using measured acoustic scattering data.

These models share similarities with hidden Markov chains (HMCs): both are models with hidden states, parameterized by a transition probability matrix and emission (or observation) distributions. Both models can be identified through the EM algorithm, involving two recursions acting in opposite directions. In both cases, these recursions involve probabilities that tend toward zero exponentially fast, causing underflow problems on computers.

The use of hidden Markov models (HMMs) relies on two main algorithms, namely, the smoothing algorithm and the global restoration algorithm. The former computes the probabilities of being in state j at node u given all the observed data. These probabilities, as a function of the index parameter u, constitute a relevant diagnosis tool; see Churchill (1989) [5] in the context of DNA sequence analysis. The smoothing algorithm also enables an efficient implementation of the E step of the EM algorithm. In most applications, the knowledge of the hidden states provides an interpretation of the data, based on the model. This motivates the need for the latter algorithm. The aim of this paper is to provide a smoothing algorithm, which is immune to underflow, and a solution for the global hidden state tree restoration.

Thus, we derive a smoothing algorithm for the HMT model, adapted from the forward-backward algorithm of Devijver (1985) [8] for HMCs. This algorithm is based on a direct decomposition of the smoothed probabilities. However the adaptation to hidden Markov tree models is not straightforward and the resulting algorithm requires an additional recursion consisting in computing the hidden state marginal distributions. Then, we present the Viterbi algorithm for HMT models. We show that the well-known Viterbi algorithm for HMCs cannot be adaptated to the HMT model. Thus, we propose a Viterbi algorithm for HMCs based on a backward recursion, which appears as a building block of the *hybrid restoration algorithm* of Brushe *et al.* (1988) [3]. This the basis for our HMT Viterbi algorithm.

Thereafter, for illustrative purpose, we apply our proposed algorithms to a segmentation problem, a standard and yet difficult signal processing task. This canonical study echoes a host of real world situations (image processing, network traffic analysis, biomedi-

cal engineering,...) where the classifying parameter is the local regularity of the measure. Wavelets have proved particularly efficient at estimating this parameter, and we show how our upward-downward algorithm circumvents the underflow problem that generally precludes classical approaches from applying to large data sets. In a second step, we elaborate on a specific use of the smoothing algorithm we propose, and principally, we motivate the usefulness of probabilistic maps when the local regularity is no longer a deterministic parameter, but the realization of a random variable with unknown density (e.g., multifractals).

This paper is organized as follows. The hidden Markov tree models are introduced in Section 2. The *upward-downward* algorithm of Crouse *et al.* (1998) [6] is summarized in Section 3. A parallel is drawn between the *forward-backward* algorithm for hidden Markov chains and their algorithm, which is shown to be subject to underflow. Then we give an *upward-downward* algorithm using smoothed probabilities for hidden Markov tree models. A solution for the global restoration problem is proposed in Section 4. An application based on simulations is provided in Section 5. This illustrates the importance of the HMT model and that of our algorithms in signal processing. Section 6 consists of concluding remarks.

2 The Hidden Markov Tree model

We use the general notation $P()$ to denote either a probability mass function or a probability density function, the true nature of $P()$ being obvious from the context. This notation obviates any assumption on the discrete or continuous nature of the output process.

Let $\bar{\boldsymbol{X}}_1 = (X_1, \ldots, X_n)$ be the output process, which is assumed to be indexable as a tree rooted in X_1. An hidden Markov tree model is composed of the observed random tree $(X_1, \ldots, X_n)$ and a hidden random tree $(S_1, \ldots, S_n)$, which has the same indexing structure as the observed tree. The variables S_u are discrete with K states, denoted $\{1, \ldots, K\}$. These variables can be indexed as a tree rooted in S_1. This model can be considered to be an unobservable state process $\bar{\boldsymbol{S}}_1$, called a Markov tree and is related to the "output" tree $\bar{\boldsymbol{X}}_1$ by a probabilistic mapping parameterized by observation or emission distributions.

Let $\boldsymbol{c}(u)$ denote the set of children of node u and let $\rho(u)$ denote the parent of node u

for $u \neq 1$. We also introduce the following notations.

- $\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u$ is the observed subtree rooted at node u. Thus $\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1$ is the entire observed tree;
- $\bar{\boldsymbol{X}}_{\boldsymbol{c}(u)} = \bar{\boldsymbol{x}}_{\boldsymbol{c}(u)}$ denotes the collection of observed subtrees rooted at children of node u (that is the subtree $\bar{\boldsymbol{x}}_u$ except its root x_u);
- if $\bar{\boldsymbol{X}}_u$ is a proper subtree of $\bar{\boldsymbol{X}}_v$ then $\bar{\boldsymbol{X}}_{v\backslash u} = \bar{\boldsymbol{x}}_{v\backslash u}$ is the subtree rooted at node v except the subtree rooted at node u;
- $\bar{\boldsymbol{X}}_{1\backslash \boldsymbol{c}(u)} = \bar{\boldsymbol{x}}_{1\backslash \boldsymbol{c}(u)}$ denotes the entire tree except the subtrees rooted at children of node u.

These notations, xhich are illustrated in Figure 1, transpose to the hidden state tree, with, for instance, $\bar{\boldsymbol{S}}_u = \bar{\boldsymbol{s}}_u$, the state subtree rooted at node u.

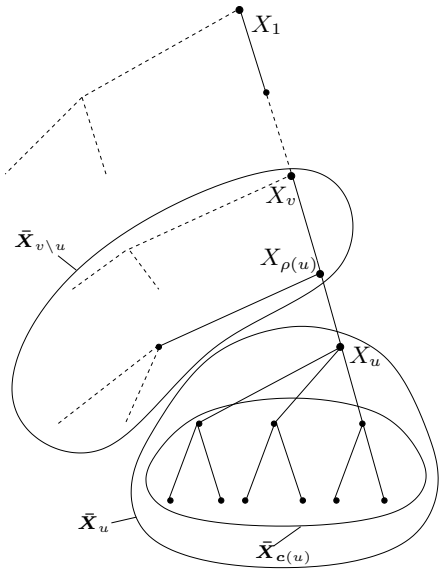

Figure 1: *Notations used for indexing trees.*

A distribution $P()$ is said to satisfy the hidden Markov tree property if and only if it fulfills the following two assumptions.

- $\forall u \in \{1, \ldots, n\}$, X_u arise from a mixture of distributions with probability

$$P(X_u = x) = \sum_{j=1}^{K} P(S_u = j) P(X_u = x | S_u = j).$$

- factorization property:

$$\forall\,(\bar{\boldsymbol{x}}_1, \bar{\boldsymbol{s}}_1), \quad P(\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1, \bar{\boldsymbol{S}}_1 = \bar{\boldsymbol{s}}_1)$$
$$= P(S_1 = s_1) \left\{ \prod_{u \neq 1} P(S_u = s_u | S_{\rho(u)} = s_{\rho(u)}) \right\} \prod_u P(X_u = x_u | S_u = s_u). \quad (1)$$

The influence diagram is a graphical way for describing conditional independence relations between random variables; see Smyth, Heckerman and Jordan (1997) [21]. The influence diagram corresponding to HMT models is shown in Figure 2. The conditional independence properties of the HMT models can be deduced from the factorization property (1).

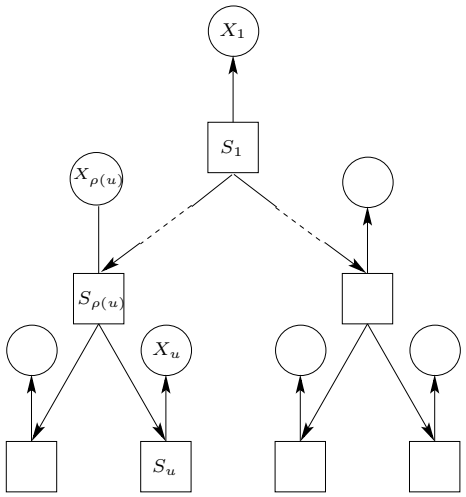

Figure 2: *Influence diagram for hidden Markov tree models.*

An hidden Markov tree model $(\bar{\boldsymbol{X}}_1, \bar{\boldsymbol{S}}_1)$ is defined by the following parameters:

- the initial distribution $\pi = (\pi_j)_j = (P(S_1 = j))_j$ for the root node S_1 and the transition probabilities $P = (p_{ij})_{i,j}$ defined by $p_{ij} = P(S_u = j | S_{\rho(u)} = i)$;

- the parameters of the emission distributions $(\theta_1, \ldots, \theta_K)$, such as

$$P(X_u = x | S_u = j) = P_{\theta_j}(x),$$

  where P_θ belongs to a parametric distribution family. We call them the *emission parameters*. For example, P_θ can be a Gaussian distribution. In this case, $\theta = (\mu, \Sigma)$ denotes its mean and its variance matrix.

Crouse *et al.* (1998) [6] considered the possibility that transition probability matrices and emission parameters depend on node u. These models do not enable a reliable estimation using only one observed tree $\bar{\boldsymbol{x}}_1$, as discussed by the authors. Thus, we directly consider *homogeneous models* (*i.e.* models having transition probabilities and emission parameters independent of u), which is usual in the literature of hidden Markov models. Our results can be easily extended to non-homogeneous models, at the cost of tedious notation.

3 Upward-downward algorithm

Since the state tree $\bar{\boldsymbol{S}}_1$ is not observable, the EM algorithm is a natural way to obtain maximum likelihood estimates of a HMT. The E step requires the computation of the conditional distributions $\xi_u(j) = P(S_u = j|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$ (smoothed probabilities) and $P(S_u = j, S_{\rho(u)} = i|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$. Crouse *et al.* (1998) [6] proposed the so-called *upward-downward* algorithm to calculate these quantities, which basically computes $P(S_u = j, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$ for each node u and each state j. This is a direct transposition to the HMT context of the *forward-backward* algorithm for hidden Markov chains proposed by Baum *et al.* (1970) [2]. Both the *upward-downward* and the *forward-backward* algorithms suffer from underflow problems; see Ephraim and Merhav (2002) [10] for the case of hidden Markov chains. This difficulty has been initially overcome by Levinson *et al.* (1983) [14], who proposed the use of scaling factors on rather heuristic grounds. On the basis of this work, Devijver (1985) [8] derived a true smoothing algorithm for hidden Markov chains, which can be interpreted in the setting of state space models. This motivates the need for a true smoothing algorithm for HMT models with the following properties:

- the smoothed probabilities $P(S_u = j|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$ are computed instead of $P(S_u = j, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$. These quantities are also useful diagnosis tools for HMT models, as will be shown in the application (section 5);
- the probabilities $P(S_u = j, S_{\rho(u)} = i|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$ can be directly extracted from this smoothing algorithm. Consequently, it implements the E step of the EM algorithm for parameter estimation;
- this algorithm is immune to underflow.

3.1 Upward-downward algorithm of Crouse, Nowak and Baraniuk (1998)

Its objective is to compute the probability $P(S_u = j, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$ for each node u and each state j. The authors define the following quantities:

$$\begin{array}{rcl} \tilde{\beta}_u(j) & = & P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | S_u = j); \\ \tilde{\beta}_{\rho(u),u}(j) & = & P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | S_{\rho(u)} = j); \\ \tilde{\alpha}_u(j) & = & P(S_u = j, \bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}). \end{array}$$

Their algorithm is based on the following decomposition of the joint probabilities:

$$\begin{array}{rcl} P(S_u = j, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) & = & P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | S_u = j) P(S_u = j, \bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}) \\ & = & \tilde{\beta}_u(j) \tilde{\alpha}_u(j). \end{array}$$

The *upward* and *downward* recursions, based on the algorithm of Ronen *et al.* (1995) [20] for Markov trees with missing data, are defined as follows:

Upward recursion

$$\begin{aligned} \tilde{\beta}_u(j) &= P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | S_u = j) \\ &= \left\{ \prod_{v \in \mathbf{c}(u)} P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v | S_u = j) \right\} P(X_u = x_u | S_u = j) \\ &= \left\{ \prod_{v \in \mathbf{c}(u)} \tilde{\beta}_{u,v}(j) \right\} P_{\theta_j}(x_u); \end{aligned} \tag{2}$$

$$\begin{aligned} \tilde{\beta}_{\rho(u),u}(j) &= P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | S_{\rho(u)} = j) \\ &= \sum_k P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | S_u = k) P(S_u = k | S_{\rho(u)} = j) \\ &= \sum_k \tilde{\beta}_u(k) p_{jk}. \end{aligned} \tag{3}$$

Since, from the equations above, the computation of $\tilde{\beta}_u(j)$ requires the quantities $(\tilde{\beta}_v(k))_k$ for each child v of u, this procedure can be implemented by an *upward* inductive tree traversal.

Downward recursion

$$\begin{aligned} \tilde{\alpha}_u(j) &= P(S_u = j, \bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}) \\ &= \sum_i P(S_u = j, S_{\rho(u)} = i, \bar{\boldsymbol{X}}_{1\setminus\rho(u)} = \bar{\boldsymbol{x}}_{1\setminus\rho(u)}, \bar{\boldsymbol{X}}_{\rho(u)\setminus u} = \bar{\boldsymbol{x}}_{\rho(u)\setminus u}) \\ &= \sum_i P(S_u = j | S_{\rho(u)} = i) \frac{P(\bar{\boldsymbol{X}}_{\rho(u)} = \bar{\boldsymbol{x}}_{\rho(u)} | S_{\rho(u)} = i)}{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | S_{\rho(u)} = i)} \\ &\quad \times P(S_{\rho(u)} = i, \bar{\boldsymbol{X}}_{1\setminus\rho(u)} = \bar{\boldsymbol{x}}_{1\setminus\rho(u)}) \\ &= \sum_i \frac{p_{ij} \tilde{\beta}_{\rho(u)}(i) \tilde{\alpha}_{\rho(u)}(i)}{\tilde{\beta}_{\rho(u),u}(i)}. \end{aligned}$$

Since from the equations above, the computation of $\tilde{\alpha}_u(j)$ requires the quantities $(\tilde{\alpha}_{\rho(u)}(i))_i$ for the parent of node u, this procedure can be implemented by a *downward* inductive tree traversal where each subtree $\bar{\boldsymbol{X}}_{1\backslash u} = \bar{\boldsymbol{x}}_{1\backslash u}$ is visited once.

The complexity of an *upward-downward* recursion is in $\mathcal{O}(nK^2)$. As for hidden Markov chains (see Levinson *et al.*, 1983 [14]), it can be seen from equations (2) and (3) that $\tilde{\beta}_u(i)$ consists of the sum of a large number of terms, each of the form

$$\left(\prod_v p_{s_{\rho(v)} s_v} \prod_v P_{\theta_{s_v}}(x_v) \right)$$

where v takes all the values in the set of descendants of u. Since each $p_{s_{\rho(v)} s_v}$ and $P_{\theta_{s_v}}(x_v)$ is generally significantly less than one, the successively computed upward probabilities tend to zero exponentially fast when progressing toward the root node, while the successively computed downward probabilities tend to zero exponentially fast when progressing toward the leaf nodes. In the next section we present an algorithm that overcomes this difficulty.

3.2 Upward-downward algorithm for smoothed probabilities

We present an alternative *upward-downward* algorithm, which is a true smoothing algorithm that is immune to underflow problems and whose complexity remains in $\mathcal{O}(nK^2)$.

In order to avoid underflow problems with hidden Markov chains, Devijver (1985) [8] suggests the replacement of the decomposition of the joint probabilities

$$P(S_t = j, \boldsymbol{X}_1^n = \boldsymbol{x}_1^n) = P(\boldsymbol{X}_{t+1}^n = \boldsymbol{x}_{t+1}^n | S_t = j) P(S_t = j, \boldsymbol{X}_1^t = \boldsymbol{x}_1^t)$$

with the decomposition

$$P(S_t = j | \boldsymbol{X}_1^n = \bar{\boldsymbol{x}}_1^n) = \frac{P(\boldsymbol{X}_{t+1}^n = \boldsymbol{x}_{t+1}^n | S_t = j)}{P(\boldsymbol{X}_{t+1}^n = \boldsymbol{x}_{t+1}^n | \boldsymbol{X}_1^t = \boldsymbol{x}_1^t)} P(S_t = j | \boldsymbol{X}_1^t = \boldsymbol{x}_1^t),$$

where for hidden Markov chains, we denote the observed sequence $X_t = x_t, \ldots, X_n = x_n$ by $\boldsymbol{X}_t^n = \boldsymbol{x}_t^n$. A natural adaptation of this method would be to use the following decomposition of the smoothed probabilities for hidden Markov tree models

$$P(S_u = j | \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) = \frac{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | S_u = j)}{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u | \bar{\boldsymbol{X}}_{1\backslash u} = \bar{\boldsymbol{x}}_{1\backslash u})} P(S_u = j | \bar{\boldsymbol{X}}_{1\backslash u} = \bar{\boldsymbol{x}}_{1\backslash u}).$$

This decomposition does not enable one to design a smoothing algorithm since the probabilities $P(S_u = j | \bar{\boldsymbol{X}}_{1\backslash u} = \bar{\boldsymbol{x}}_{1\backslash u})$ cannot be computed in an initial downward pass. Only

a quantity such as $P(S_u = j|\boldsymbol{X}_1^u = \boldsymbol{x}_1^u)$ where $\boldsymbol{X}_1^u = \boldsymbol{x}_1^u$ denotes the output path from the root to node u, can be computed in a initial downward pass. The quantities

$$P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|S_u = j)/P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}) = \tilde{\beta}_u(j)/P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u})$$

cannot be computed in an initial upward pass due to the normalizing quantity $P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u})$. By similar arguments, the scaling factor method proposed by Levinson *et al.* (1983) [14] for hidden Markov chains, which is equivalent to Devijver's algorithm, cannot be adapted to HMT models. Finally, we use the alternative decomposition of the smoothed probabilities $\xi_u(j)$

$$\xi_u(j) = \frac{P(\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}|S_u = j)}{P(\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}|\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)} P(S_u = j|\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u).$$

Consequently, we introduce the following quantities

$$\begin{aligned}
\beta_u(j) &= P(S_u = j|\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u); \\
\beta_{\rho(u),u}(j) &= \frac{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|S_{\rho(u)} = j)}{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)}; \\
\alpha_u(j) &= \frac{P(\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}|S_u = j)}{P(\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}|\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)}.
\end{aligned}$$

The corresponding new *upward-downward* algorithm includes the recursions described below. The proof of these equations is based on factorizations of conditional probabilities deduced from conditional independence properties following from equation (1), or, equivalently, from the influence diagram (see Figure 2).

As will become apparent in the following, the *upward* and *downward* recursions require the preliminary knowledge of the marginal state distributions $P(S_u = j)_j$ for each node u. This is achieved by a downward recursion initialized for the root node by $P(S_1 = j) = \pi_j$. Then, for each of the remaining nodes taken downwards, we have the following recursion:

$$P(S_u = j) = \sum_i p_{ij} P(S_{\rho(u)} = i).$$

Upward recursion

The upward recursion is initialized for each leaf by

$$\begin{aligned}\beta_u(j) &= P(S_u = j|X_u = x_u) \\ &= \frac{P(X_u = x_u|S_u = j)P(S_u = j)}{P(X_u = x_u)} \\ &= \frac{P_{\theta_j}(x_u)P(S_u = j)}{N_u}.\end{aligned}$$

Then, for each of the remaining nodes taken upwards, we have the following recursion

$$\begin{aligned}\beta_u(j) &= P(S_u = j|\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u) \\ &= \left\{\prod_{v\in\boldsymbol{c}(u)} P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v|S_u = j)\right\} P(X_u = x_u|S_u = j)\frac{P(S_u = j)}{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)} \\ &= \left\{\prod_{v\in\boldsymbol{c}(u)} \frac{P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v|S_u = j)}{P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v)}\right\} P(X_u = x_u|S_u = j)P(S_u = j) \\ &\quad \times\frac{\prod_{v\in\boldsymbol{c}(u)} P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v)}{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)} \\ &= \frac{\left\{\prod_{v\in\boldsymbol{c}(u)} \beta_{u,v}(j)\right\} P_{\theta_j}(x_u)P(S_u = j)}{N_u}. \qquad (4)\end{aligned}$$

Since $\sum_j \beta_u(j) = 1$, the normalizing factor N_u is given by

$$N_u = P(X_u = x_u) = \sum_j P_{\theta_j}(x_u)P(S_u = j)$$

for the leaf nodes, and

$$\begin{aligned}N_u &= \frac{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)}{\prod_{v\in\boldsymbol{c}(u)} P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v)} \\ &= \sum_j \left\{\prod_{v\in\boldsymbol{c}(u)} \beta_{u,v}(j)\right\} P_{\theta_j}(x_u)P(S_u = j) \qquad (5)\end{aligned}$$

for the nonleaf nodes.

The *upward* recursion also involves the computation of the quantities $\beta_{\rho(u),u}(j)$, which are extracted from the $(\beta_u(j))_j$ quantities, since

$$\begin{aligned}\beta_{\rho(u),u}(j) &= \frac{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|S_{\rho(u)} = j)}{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)} \\ &= \frac{\sum_k P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|S_u = k)P(S_u = k|S_{\rho(u)} = j)}{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)} \qquad (6)\end{aligned}$$

$$
\begin{aligned}
&= \sum_k \frac{P(S_u = k|\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)}{P(S_u = k)} P(S_u = k|S_{\rho(u)} = j) \\
&= \sum_k \frac{\beta_u(k) p_{jk}}{P(S_u = k)}.
\end{aligned}
$$

In a first step, the quantities

$$
\gamma_u(j) = \frac{P(S_u = j, \bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)}{\prod_{v \in \mathbf{c}(u)} P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v)} = \beta_u(j) N_u
$$

are computed with $N_u = \sum_j \gamma_u(j)$. By convention, $\gamma_u(j) = P(S_u = j, X_u = x_u)$ for the leaf nodes. In a second step, the quantities $\beta_u(j)$ are extracted as $\gamma_u(j)/N_u$. Finally, the quantities $\beta_{\rho(u),u}(j)$ are extracted from $(\beta_u(j))_j$, and the algorithm processes the nodes at lower depth.

It can be seen that

$$
\begin{aligned}
P(\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) &= \prod_u \frac{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)}{\prod_{v \in \mathbf{c}(u)} P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v)} \\
&= \prod_u N_u
\end{aligned}
$$

(recall that for each leaf u, $N_u = P(X_u = x_u)$). Hence the log-likelihood is

$$
\log P(\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) = \sum_u \log N_u.
$$

It follows from equation (5) that the log-likelihood can be computed as a byproduct of the *upward* recursion. The log-likelihood computation allows, among other potential applications, the monitoring of the EM algorithm convergence; see McLachlan and Krishnan (1997) [16].

It is possible to build a downward recursion on the basis of the quantities $\alpha_u(j)$ or on the basis of the smoothed probabilities $\xi_u(j) = P(S_u = j|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$. This is a direct transposition of the argument of Devijver (1985) to the case of hidden Markov tree models.

Downward recursion based on $\xi_u(j)$

The downward recursion is initialized for the root node by

$$\xi_1(j) = P(S_1 = j|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) = \beta_1(j).$$

Then, for each of the remaining nodes taken downwards, we have the following recursion.

$$\begin{aligned}
\xi_u(j) &= P(S_u = j|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) \\
&= \sum_i \frac{P(S_u = j, S_{\rho(u)} = i, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)}{P(S_{\rho(u)} = i, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)} P(S_{\rho(u)} = i|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) \\
&= P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|S_u = j) \sum_i \frac{P(S_u = j|S_{\rho(u)} = i)}{P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u|S_{\rho(u)} = i)} \\
&\quad \times \frac{P(S_{\rho(u)} = i, \bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u})}{P(S_{\rho(u)} = i, \bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u})} P(S_{\rho(u)} = i|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) \\
&= \frac{\beta_u(j)}{P(S_u = j)} \sum_i \frac{p_{ij}\xi_{\rho(u)}(i)}{\beta_{\rho(u),u}(i)}. \qquad (7)
\end{aligned}$$

Since for each u, $\xi_u(j) = \beta_u(j)\alpha_u(j)$, the *downward* recursion based on $\alpha_u(j)$ is directly deduced from (7). This is initialized by $\alpha_1(j) = 1$, and for each of the remaining nodes taken downwards, we have the following recursion

$$\alpha_u(j) = \frac{1}{P(S_u = j)} \sum_i \frac{p_{ij}\beta_{\rho(u)}(i)\alpha_{\rho(u)}(i)}{\beta_{\rho(u),u}(i)}.$$

The conditional probabilities $P(S_u = j, S_{\rho(u)} = i|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$ required for the reestimation of the parameters by the EM algorithm are directly extracted during the downward recursion

$$P(S_u = j, S_{\rho(u)} = i|\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) = \frac{\beta_u(j)p_{ij}\xi_{\rho(u)}(i)}{P(S_u = j)\beta_{\rho(u),u}(i)}.$$

Table 1 points out the differences between the *upward-downward* of Crouse *et al.* (1998) [6], using the decomposition of joint probabilities, and our algorithm, using the decomposition of the smoothed probabilities.

As for Devijver's algorithm, the execution of the above procedure does not cause underflow problems. The term that dominates the recursion complexity for the computation of the hidden state distributions is $2nK^2$. The complexities of our *upward* and *downward* recursions also have dominant term $2nK^2$ for binary trees (or for any tree such as the degree of each node remains bounded). Thus, the complexity of the *upward-downward* algorithm using smoothed probabilities remains in $\mathcal{O}(nK^2)$, as the algorithm of Crouse *et al.* (1998), but with the complexity increasing by 50%.

Table 1: *Differences between the upward-downward algorithm of Crouse et al. (1998) and our smoothing upward-downward algorithm*

Algorithm of Crouse *et al.*	Our smoothing algorithm
upward probabilities $\tilde{\beta}_u(j) = P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u \mid S_u = j)$	upward probabilities $\beta_u(j) = P(S_u = j \mid \bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)$
downward probabilities $\tilde{\alpha}_u(j) = P(\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}, S_u = j)$	downward probabilities $\alpha_u(j) = \dfrac{P(\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u} \mid S_u = j)}{P(\bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u} \mid \bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u)}$
smoothed probabilities $P(S_u = j \mid \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) = \dfrac{\tilde{\beta}_u(j)\tilde{\alpha}_u(j)}{P(\bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{X}}_1)}$	smoothed probabilities $P(S_u = j \mid \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) = \alpha_u(j)\beta_u(j)$

4 Viterbi algorithm

Given an observed tree $\bar{\boldsymbol{x}}_1$, our aim is to find the hidden state tree $\bar{\boldsymbol{s}}_1^* = (s_1^*, \ldots, s_n^*)$ maximizing $P(\bar{\boldsymbol{S}}_1 = \bar{\boldsymbol{s}}_1 | \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$ – or, equivalently, $P(\bar{\boldsymbol{S}}_1 = \bar{\boldsymbol{s}}_1, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1)$, see Rabiner (1989) [18] – and the value P^* of the maximum. We call any algorithm solving this problem a Viterbi algorithm in reference to the hidden Markov chain terminology.

The initial global restoration algorithm for nonindependent mixture models is due to Viterbi. The Viterbi algorithm is originally intended for the analysis of Markov processes observed in memoryless noise. In the case of a hidden Markov chain $\{S_t, X_t; t = 0, 1, \ldots\}$, let $\boldsymbol{S}_1^n = \boldsymbol{s}_1^n$ denote the state sequence of length n and $\boldsymbol{X}_1^n = \boldsymbol{x}_1^n$ denote the output sequence of length n. The Viterbi algorithm for hidden Markov chains is basically a forward recursion computing the quantities

$$\tilde{\delta}_t(j) = \max_{s_1,\ldots,s_{t-1}} P(S_t = j, \boldsymbol{S}_1^{t-1} = \boldsymbol{s}_1^{t-1}, \boldsymbol{X}_1^t = \boldsymbol{x}_1^t), \tag{8}$$

starting at the initial state S_1.

A natural adaptation of the Viterbi algorithm to HMT models would involve a downward recursion starting at the root state S_1. We claim that this is not possible, for the same reason as for our smoothing algorithm, namely, that the *downward* recursion would require the results of the *upward* recursion (see section 3). Thus, we need to design a new Viterbi algorithm for hidden Markov chains based on a backward recursion, which will be the basis of our adaptation to HMT models.

Because the state process is a Markov chain, we have for all t the following decompo-

sition, adaptated from Jelinek (1997) [13]

$$\begin{aligned}
&\max_{s_1,\ldots,s_n} P(\boldsymbol{S}_1^n = \boldsymbol{s}_1^n, \boldsymbol{X}_1^n = \boldsymbol{x}_1^n) \\
&\quad = \max_{s_t}\{\max_{s_{t+1},\ldots,s_n} P(\boldsymbol{X}_t^n = \boldsymbol{x}_t^n, \boldsymbol{S}_{t+1}^n = \boldsymbol{s}_{t+1}^n | S_t = s_t) \\
&\qquad \times \max_{s_1,\ldots,s_{t-1}} P(\boldsymbol{S}_1^t = \boldsymbol{s}_1^t, \boldsymbol{X}_1^{t-1} = \boldsymbol{x}_1^{t-1})\}. \qquad (9)
\end{aligned}$$

Let us define

$$\delta_t(j) = \max_{s_{t+1},\ldots,s_n} P(\boldsymbol{X}_t^n = \boldsymbol{x}_t^n, \boldsymbol{S}_{t+1}^n = \boldsymbol{s}_{t+1}^n | S_t = j).$$

Decomposition (9) can then be rewritten as

$$\begin{aligned}
&\max_{s_1,\ldots,s_n} P(\boldsymbol{S}_1^n = \boldsymbol{s}_1^n, \boldsymbol{X}_1^n = \boldsymbol{x}_1^n) \\
&\quad = \max_j\{\delta_t(j) \max_{s_1,\ldots,s_{t-1}} P(S_t = j, \boldsymbol{S}_1^{t-1} = \boldsymbol{s}_1^{t-1}, \boldsymbol{X}_1^{t-1} = \boldsymbol{x}_1^{t-1})\}.
\end{aligned}$$

Using the quantities $\delta_t(j)$, we can build a Viterbi algorithm for hidden Markov chains based on a backward recursion, which is equivalent to that of Brushe *et al.* (1998) [3]. This is initialized for $t = n$ by

$$\begin{aligned}
\delta_n(j) &= P(X_n = x_n | S_n = j) \\
&= P_{\theta_j}(x_n).
\end{aligned}$$

The backward recursion is given, for $t = n-1, \ldots, 1$, by

$$\begin{aligned}
\delta_t(j) &= \max_{s_{t+1},\ldots,s_n} P(\boldsymbol{X}_t^n = \boldsymbol{x}_t^n, \boldsymbol{S}_{t+1}^n = \boldsymbol{s}_{t+1}^n | S_t = j) \\
&= \max_k\{\max_{s_{t+2},\ldots,s_n} P(\boldsymbol{X}_{t+1}^n = \boldsymbol{x}_{t+1}^n, \boldsymbol{S}_{t+2}^n = \boldsymbol{s}_{t+2}^n | S_{t+1} = k) \\
&\quad \times P(S_{t+1} = k | S_t = j)\} P(X_t = x_t | S_t = j) \\
&= \max_k\{\delta_{t+1}(k) p_{jk}\} P_{\theta_j}(x_t).
\end{aligned}$$

We obtain, for $t = 1$, $\delta_1(j) = \max_{s_2,\ldots,s_n} P(\boldsymbol{X}_1^n = \boldsymbol{x}_1^n, \boldsymbol{S}_2^n = \boldsymbol{s}_2^n | S_1 = j)$. Hence, the probability of the optimal state sequence associated with the observed sequence $\boldsymbol{x}_1^n$ is

$$\begin{aligned}
P^* &= \max_j\{\max_{s_2,\ldots,s_n} P(\boldsymbol{X}_1^n = \boldsymbol{x}_1^n, \boldsymbol{S}_2^n = \boldsymbol{s}_2^n | S_1 = j) P(S_1 = j)\} \\
&= \max_j\{\delta_1(j)\pi_j\}.
\end{aligned}$$

Transposing decomposition (9) to hidden Markov tree models yields for all u

$$\begin{aligned}
&\max_{\bar{\boldsymbol{s}}_1} P(\bar{\boldsymbol{S}}_1 = \bar{\boldsymbol{s}}_1, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) \\
&\quad = \max_{s_u}\{\max_{\bar{\boldsymbol{s}}_{\boldsymbol{c}(u)}} P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u, \bar{\boldsymbol{S}}_{\boldsymbol{c}(u)} = \bar{\boldsymbol{s}}_{\boldsymbol{c}(u)} | S_u = s_u) \\
&\qquad \times \max_{\bar{\boldsymbol{s}}_{1\setminus u}} P(\bar{\boldsymbol{S}}_{1\setminus \boldsymbol{c}(u)} = \bar{\boldsymbol{s}}_{1\setminus \boldsymbol{c}(u)}, \bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u})\}. \qquad (10)
\end{aligned}$$

Let us define

$$\delta_u(j) = \max_{\bar{\boldsymbol{s}}_{\boldsymbol{c}(u)}} P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u, \bar{\boldsymbol{S}}_{\boldsymbol{c}(u)} = \bar{\boldsymbol{s}}_{\boldsymbol{c}(u)} | S_u = j) \quad (11)$$

$$\delta_{\rho(u),u}(j) = \max_{\bar{\boldsymbol{s}}_u} P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u, \bar{\boldsymbol{S}}_u = \bar{\boldsymbol{s}}_u | S_{\rho(u)} = j). \quad (12)$$

Hence (10) can be rewritten as

$$\begin{aligned} & \max_{\bar{\boldsymbol{s}}_1} P(\bar{\boldsymbol{S}}_1 = \bar{\boldsymbol{s}}_1, \bar{\boldsymbol{X}}_1 = \bar{\boldsymbol{x}}_1) \\ = & \max_j \left\{ \delta_u(j) \max_{\bar{\boldsymbol{s}}_{1\setminus u}} P(S_u = j, \bar{\boldsymbol{S}}_{1\setminus u} = \bar{\boldsymbol{s}}_{1\setminus u}, \bar{\boldsymbol{X}}_{1\setminus u} = \bar{\boldsymbol{x}}_{1\setminus u}) \right\}. \end{aligned}$$

The main change with respect to hidden Markov chains is that it is not possible to design a downward recursion on the basis of this type of decomposition but solely an upward recursion.

The Viterbi algorithm for a hidden Markov tree is initialized for each leaf by

$$\begin{aligned} \delta_u(j) &= P(X_u = x_u | S_u = j) \\ &= P_{\theta_j}(x_u). \end{aligned}$$

Then, for each of the remaining nodes taken upwards, we have the following recursion

$$\begin{aligned} \delta_u(j) &= \max_{\bar{\boldsymbol{s}}_{\boldsymbol{c}(u)}} P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u, \bar{\boldsymbol{S}}_{\boldsymbol{c}(u)} = \bar{\boldsymbol{s}}_{\boldsymbol{c}(u)} | S_u = j) \\ &= \left\{ \prod_{v \in \boldsymbol{c}(u)} \max_{\bar{\boldsymbol{s}}_v} P(\bar{\boldsymbol{X}}_v = \bar{\boldsymbol{x}}_v, \bar{\boldsymbol{S}}_v = \bar{\boldsymbol{s}}_v | S_u = j) \right\} P(X_u = x_u | S_u = j) \\ &= \left\{ \prod_{v \in \boldsymbol{c}(u)} \delta_{u,v}(j) \right\} P_{\theta_j}(x_u); \end{aligned}$$

$$\begin{aligned} & \delta_{\rho(u),u}(j) = \max_{\bar{\boldsymbol{s}}_u} P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u, \bar{\boldsymbol{S}}_u = \bar{\boldsymbol{s}}_u | S_{\rho(u)} = j) \\ &= \max_k \left\{ \max_{\bar{\boldsymbol{s}}_{\boldsymbol{c}(u)}} P(\bar{\boldsymbol{X}}_u = \bar{\boldsymbol{x}}_u, \bar{\boldsymbol{S}}_{\boldsymbol{c}(u)} = \bar{\boldsymbol{s}}_{\boldsymbol{c}(u)} | S_u = k) P(S_u = k | S_{\rho(u)} = j) \right\} \\ &= \max_k \{ \delta_u(k) p_{jk} \}. \end{aligned}$$

The probability of the optimal state tree associated with the observed tree $\bar{\boldsymbol{x}}_1$ is

$$P^* = \max_j \{ \delta_1(j) \pi_j \}.$$

The Viterbi algorithm is similar to the *upward* recursion of Crouse *et al.* (1998) [6] where the summations on the states are replaced by maximizations. Its complexity is in $\mathcal{O}(nK^2)$, and no normalization quantities are required. To retrieve the optimal state tree, it is necessary to store for each node u and each state j the optimal states corresponding to each of the children. The backtracking procedure consists in tracing downward along the backpointers from the optimal root state to the optimal leaf states.

5 Application to signal processing

In this section, we develop one example of application, illustrating the importance of the hidden Markov tree model. Let $\boldsymbol{x}_1^T = (x_1, \ldots, x_T)$ be a realization of a sampled piecewise constant (Hölder) regularity process, for example a piecewise homogeneous fractional Brownian motion (H-FMB). The local regularity of a function (or of the trajectory of a stochastic process) is defined as by Mallat (1998) [15] as follows: the function f has local regularity $k < h < k+1$, at time t, if there exists two constants $0 < C < \infty$ and $0 < t_0$ as well as a polynomial P_k of order k, such that for all $t - t_0 < l < t + t_0$ and for all $h' \leq h$,

$$|f(l) - P_k(l)| < C|l - t|^{h'}. \tag{13}$$

In our simulation, we consider the slightly modified model of a compound-FBM. This model assumes that $T = 2^M$ and that from $t = 1$ to $t = T_0$ with $1 \leq T_0 < T$, the local regularity of the process is $H = H_0$, and from $t = T_0 + 1$ to $t = T$, its local regularity is $H = H_1$. Our aim is not to estimate H_0 or H_1 but rather to determine the transition time T_0. To motivate our work, we recall for instance the article of Abry and Veitch (1998) [1] where they show the major importance of detecting local regularity changes, in the context of network traffic analysis.

Our method is based on a multiresolution analysis of $\boldsymbol{x}_1^T$. As a first step, we compute an orthonormal discrete wavelet transform of $\boldsymbol{x}_1^T$ through the following inner product: $(w_n^m)_{1 \leq m \leq J_0,\ 0 \leq n \leq 2^m - 1}$, with $w_n^m = \sum_{k=1}^{2^M} x_k 2^{m/2} \psi(2^m k - n)$ and J_0 corresponding to the finest scale.

As in Crouse, Nowak and Baraniuk (1998), we combine a statistical approach with a wavelet-based signal processing. This means that we process the signal $\boldsymbol{x}_1^T$ by operating on its wavelet coefficients $(w_n^m)_{m,n}$ and that we consider these coefficients to be realizations of random variables $(W_n^m)_{m,n}$. The authors justify a hidden Markov binary tree model for the wavelet coefficients by two observations:

- residual dependencies remain between wavelet coefficients;
- wavelet coefficients are generally non-Gaussian.

We recall that the path of an H-FBM has local Hölder regularity H almost surely almost everywhere. Hence from Jaffard (1991) [12], Flandrin (1992) [11] and Wornell

et al. (1992) [22], the random variables W_n^m of its wavelet decomposition are normally, identically distributed within scale and centered with variance

$$\mathrm{var}(W_n^m) = \sigma^2 2^{m(2H+1)}.$$

In our simple test signal, where the local regularity is H_0 for $1 \leq t \leq T_0$ and H_1 for $T_0+1 \leq t \leq T$, we consider a two-state model with the following conditional distributions:

$$(W_n^m | S_n^m = j) \sim \mathcal{N}(0, \sigma_j^2 2^{m(2H_j+1)}).$$

Thus, we model the distribution of $(w_n^m)_{m,n}$ by the following hidden Markov tree model:

- W_n^m arises from a mixture of distributions with density

$$f(W_n^m = w_n^m) = \sum_{j=0}^{1} P(S_n^m = j) f_{\theta_j}(w_n^m)$$

  where S_n^m is a discrete variable with two states, denoted $\{0,1\}$, and $f_{\theta_j}(w_n^m)$ is the Gaussian distribution density with mean 0 and variance $\sigma_j^2 2^{m(2H_j+1)}$;

- $(S_n^m)_{m,n}$ is a Markov binary tree (*i.e.* each nonterminal node has exactly two children) with parameters $(\pi_j)_j$ and $(p_{ij})_{i,j}$;

- the wavelet coefficients are independent, conditionally to the hidden states.

As in Section 2, we denote the observed tree $(W_n^m)_{m,n}$ by $\bar{\boldsymbol{W}}_1 = \bar{\boldsymbol{w}}_1$ and the hidden tree $(S_n^m)_{m,n}$ by $\bar{\boldsymbol{S}}_1 = \bar{\boldsymbol{s}}_1$.

In the case of an abrupt regularity jump at time T_0, the hidden tree model $(\bar{\boldsymbol{W}}_1, \bar{\boldsymbol{S}}_1)$ satisfies the following two properties:

- for each subtree $\bar{\boldsymbol{S}}_u$ of $\bar{\boldsymbol{S}}_1$, there exists j in $\{0,1\}$ such as the left subtree of $\bar{\boldsymbol{S}}_u$ is entirely in state j, or its right subtree is entirely in state j,

- if $S_{t_1}^{J_0}$ and $S_{t_2}^{J_0}$ are two leaves with $t_1 < t_2$ such as $S_{t_1}^{J_0} = S_{t_2}^{J_0} = j$, then for all t between t_1 and t_2, $S_t^{J_0} = j$.

To detect the local regularity jump, we compute the discrete wavelet transform w_n^m of the signal using a compact support Daubechies wavelet. An important proviso is that the chosen wavelet has regularity larger than the regularity of the process itself. In our case,

we are dealing with $H \in (0,1)$; therefore we choose the simplest possible wavelet: the Haar with regularity one. Since our model assumes two states per tree, here, we need a single tree decomposition. This imposes only one wavelet coefficient at the coarsest scale (root node), and thus, $T = 2^{J_0}$ for full J_0-level tree [1]. Then, we estimate the model parameters by the EM algorithm, using our *upward-downward* algorithm with smoothed probabilities to implement the E step. We could not use the *upward-downward* algorithm of Crouse *et al.* (1998) [6] directly (*i.e.* without an *ad hoc* scaling procedure) since underflow errors occur for values of T typically greater than 128.

The H_j and σ_j parameters are estimated at the M step with a procedure adaptated from the maximum likelihood estimation derived by Wornell and Oppenheim (1992) [22]. Thus, we obtain $\hat{P}$, $\hat{\pi}$, $\hat{\sigma}_0$, $\hat{\sigma}_1$, $\hat{H}_0$, and $\hat{H}_1$. The jump detection is performed by a hidden state restoration under the two constraints above, using the Viterbi algorithm. We obtain a value for the hidden tree $\bar{\boldsymbol{S}}_1$ such as exactly one subtree $\bar{\boldsymbol{S}}_u$ of $\bar{\boldsymbol{S}}_1$ is in state j, and $\bar{\boldsymbol{S}}_{1\setminus t}$ is in state $1-j$. Thus, there is only one leave $S_{t^*}^{J_0}$ such as $S_{t^*}^{J_0} \neq S_{t^*+1}^{J_0}$. The jump time T_0 is estimated by:

$$\hat{T_0} = 2.t^*.$$

In practice, to avoid a too-severe discontinuity in the path at the transition time T_0 and to ensure that at any point t, the local regularity $H(t)$ is correctly defined, we synthesize a multifractional Brownian motion as proposed and defined in Lévy Véhel and Peltier (1995) [17], with a continuous transitional Hölder regularity (Figure 3):

$$\forall t \in \{1, \ldots, 1024\} \qquad H(t) = 0.1 \tanh(-20 + \frac{40(t-1)}{1023}) + 0.5. \tag{14}$$

We set the asymptotics $H_0 = 0.4$ and $H_1 = 0.6$. We then construct the process $\boldsymbol{x}_1^{1024} = (x(t))_{t=1,\ldots,1024}$ with local regularity given by (14). One realization path of such a process is shown in Figure 4 a).

Figure 4 b) shows the map of the smoothed probabilities $P(S_n^m = 1 | \bar{\boldsymbol{W}}_1 = \bar{\boldsymbol{w}}_1)$. We define the depth $J(u)$ (or scale) of a node u as the number of nodes on the path between the root and node u. Our convention is that the depth of the tree root is equal to one.

[1] If T is not a power of 2, we can zero-pad the series and consider a third state in the model with arbitrary small variance.

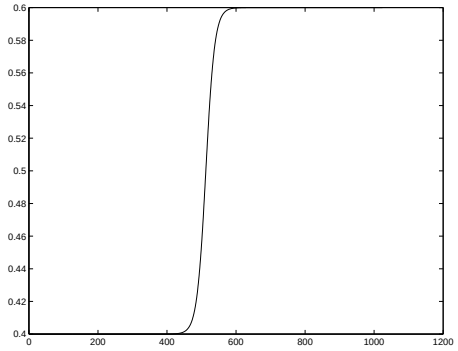

Figure 3: *Hölder trajectory: Time varying local regularity.*

The depth of the observed or hidden tree is defined by $\max_u J(u)$. Thus, in our case, the depth of the tree is equal to J_0. The Y-axis of the plot represents the tree depth, with root at the bottom line. Figure 4 c) shows the result of the hidden state restoration. The border between both states is used to locate the transition time T_0 in $H(t)$. The estimated parameters are $\hat{H}_0 = 0.3009$, $\hat{H}_1 = 0.6649$ and $\hat{T}_0 = 520$.

These results deserve several remarks. First, the estimates of H_0 and H_1 are imprecise, due to the small number of time-samples for each state. Nonetheless, they are coherent with the performances discussed in Wornell and Oppenheim (1992) [22]. In particular, the method used for the estimation of H_j and σ_j suffers from the same limitations as the algorithm described in Wornell and Oppenheim. On the other hand, and as far as the discrimination is concerned, the separation of the mixture components achieved by our method is very accurate. Most importantly, thanks to the restoration procedure, loose estimates for H do not affect the transition time determination $\hat{T}_0$.

In a perspective viewpoint, to improve the estimates, we could substitute likelihood maximization with alternative methods. For instance, to derive estimates of the parameters H_0 and H_1, it is possible to use a (weighted) linear regression of the within-scale empirical variance, restricted to a more relevant scale sub-range.

In our elementary example, the smoothed probability map (Figure 4-b) is merely a complementary stage to restoration. Yet, let us comment on the apparent uncertainty of the states observed after transition time T_0. Recalling the definition of the local Hölder regularity of a given path, h is the supremum over all h' satisfying inequality in (13). This means that pointwise, smaller estimated regularities are likely to occur. Now, when

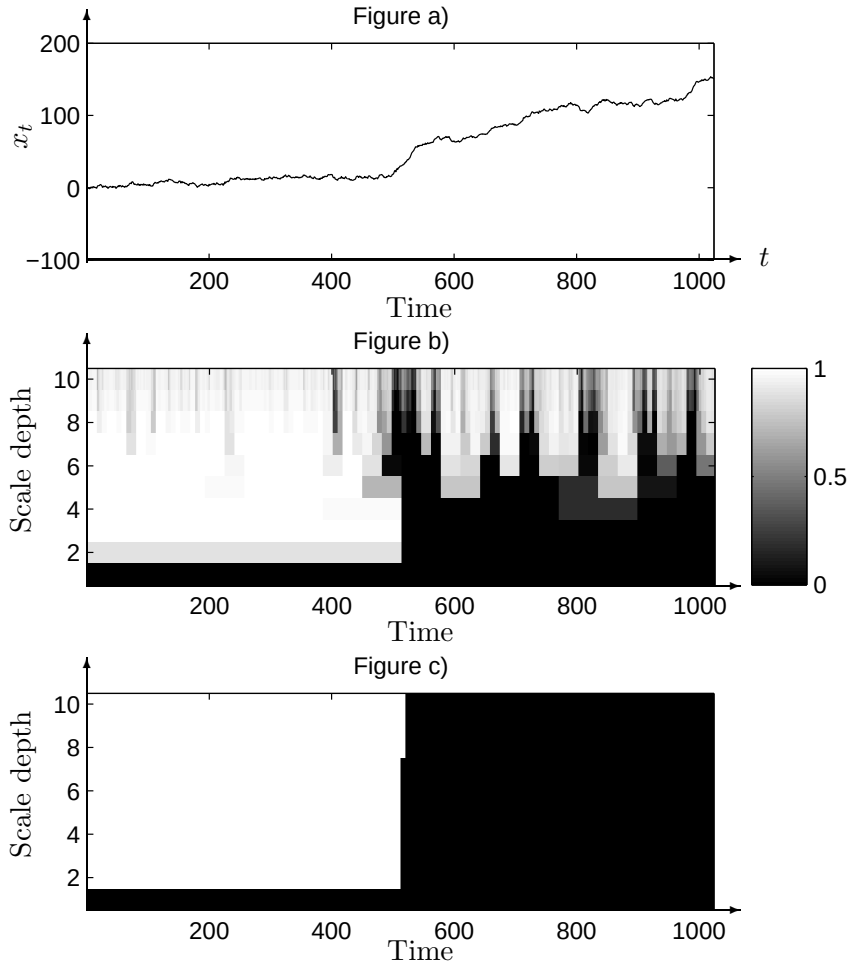

Figure 4: *Hidden tree associated with a wavelet decomposition of a signal – a) A path of a piecewise-constant FBM with regularity* $H_0 = 0.4$ *(corresponding to state 0) for* $t = 1, \ldots, 512$ *and regularity* $H_1 = 0.6$ *(corresponding to state 1) for* $t = 513, \ldots, 1024$*. b) Map of the smoothed probabilities. The grey level indicates the value of the conditional probability of state 1 occurrence at a given node. c) Restored hidden tree.*

analyzing the more regular part of the trace, the retained two-state model actually allows for estimating local regularities smaller than the effective one, hence, these changeovers. Again, in accordance with definition (13), this clearly does not happen with the left-hand side of the path (less regular part).

More interestingly, now, a probabilistic map becomes fully interesting on its own, when exploring more complex situations. To support our claim, let us elaborate on two examples.

- We return to our previous two-state example and assume a smooth transition from H_0 to H_1. This means that the local hölder regularity takes on infinitely many

values within interval $[H_0, H_1]$, turning the frontier between the two stable states very fuzzy. A binary segmentation obtained with the restoration algorithm may not be so sensible and necessarily implies some arbitrariness in selecting the transition time T_0. Instead, the probabilistic map, which is output of the smoothing algorithm, provides us with a *fuzzy segmentation* that conveys more valuable information concerning the dynamics of the transition.

- The second example concerns situations referred to as multifractals; see, e.g., Riedi (2000) [19]. In short, for such processes, local Hölder regularity H is itself a random variable leading to utterly erratic Hölder paths $(t, H(t))$ and whose pointwise estimation becomes totally unrealistic. Instead, we resort to the notion of singularity spectrum that allows for quantifying how frequently a given singularity strength $H(t) = h$ is assumed. Then, probabilistic maps, like the one displayed in figure 4-b, can easily be thought as a measure of occurrence of the quantized regularity h_k, $k = 1, \ldots, K$ associated with the given state k. Conceptually, it would suffice to marginalize these distributions and to represent the obtained *a priori* probabilities P_k versus h_k, to get a *discrete singularity density* (h_k, P_k).

Another very interesting extension of hidden Markov tree models is to consider, as for hidden Markov chains, continuous-valued hidden states. It is known that the estimation problem in such models is difficult. However, from an application viewpoint and when the model is entirely specified, it would allow for modeling signals with continuously time varying local regularity. In this context, it would be possible to compute the local regularity distribution conditionally to the observed wavelet coefficients.

6 Concluding remarks

In this paper, we developed a smoothing algorithm, which implements the E step of the EM algorithm for parameter estimation. The important improvement carried out to the existing algorithm of Crouse *et al.* (1998) [6] is that ours is not subject to underflow. This allows us to apply hidden Markov trees to large data sets.

Another important innovation of our methodology is the use of the smoothed probability map. In particular, we showed it to be relevant in a wavelet-tree application and,

in possible extensions, to models with continuous-valued hidden states.

As this application demonstrates the need for a global restoration algorithm, a solution based on the adaptation of a backward Viterbi algorithm for hidden Markov chains has been proposed.

Acknowledgments

The authors acknowledge helpful advice and discussion about inference algorithms in mixture models with Gilles Celeux. They are thankful to the anonymous reviewers for their useful comments.

References

[1] P. Abry and D. Veitch. Wavelet Analysis of Long Range Dependant Traffic. *IEEE Transactions on Information Theory*, 44(1):2–15, January 1998.

[2] L.E. Baum, T. Petrie, G. Soules, and N. Weiss. A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics*, 41(1):164–171, 1970.

[3] G.D. Brushe, R.E. Mahony, and J.B. Moore. A Soft Output Hybrid Algorithm for ML/MAP Sequence Estimation. *IEEE Transactions on Information Theory*, 44(7):3129–3134, November 1998.

[4] H. Choi and R.G. Baraniuk. Multiscale Image Segmentation using Wavelet-Domain Hidden Markov Models. *IEEE Transactions on Image Processing* , to be published in 2001.

[5] G.A. Churchill. Stochastic Models for Heterogeneous DNA Sequences. *Bulletin of Mathematical Biology*, 51:79–94, 1989.

[6] M.S. Crouse, R.D. Nowak, and R.G. Baraniuk. Wavelet-Based Statistical Signal Processing Using Hidden Markov Models. *IEEE Transactions on Signal Processing*, 46(4):886–902, April 1998.

[7] N. Dasgputa, P. Runkle, L. Couchman, and L. Carin. Dual hidden Markov model for characterizing wavelet coefficients from multi-aspect scattering data. *Signal Processing*, 81(6):1303–1316, 2001.

[8] P. A. Devijver. Baum's forward-backward Algorithm Revisited. *Pattern Recognition Letters*, 3:369–373, 1985.

[9] M. Diligenti, P. Frasconi, and M. Gori. Image Document Categorization using Hidden Tree Markov Models and Structured Representations. In S. Singh, N. Murshed, and W. Kropatsch, editors, *Second International Conference on Advances in Pattern Recognition. Lecture Notes in Computer Science*, 2001.

[10] Y. Ephraim and N. Merhav. Hidden Markov processes. *IEEE Transactions on Information Theory*, 48:1518–1569, June 2002.

[11] P. Flandrin. Wavelet Analysis and Synthesis of Fractional Brownian Motion. *IEEE Transactions on Information Theory*, 38:910–917, 1992.

[12] S. Jaffard. Pointwise Smoothness, two-microlocalization and Wavelet Coefficients. *Publications Mathématiques*, 35:155–168, 1991.

[13] F. Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, 1997.

[14] S.E. Levinson, L.R. Rabiner, and M.M. Sondhi. An Introduction to the Application of the Theory of Probabilistic Functions of a Markov Process in Automatic Speech Recognition. *Bell System Technical Journal*, 62:1035–1074, 1983.

[15] S. Mallat. *A Wavelet Tour of Signal Processing*. San Diego, California: Academic Press. xxiv, 1998.

[16] G. McLachlan and T. Krishnan. *The EM Algorithm and Extensions*. J. Wiley and sons, 1997.

[17] R. Peltier and J. Levy-Vehel. Multifractional Brownian Motion : Definition and Preliminary Results. Technical Report RR-2645, INRIA, 1995. Submitted to Stochastic Processes and their Applications.

[18] L.R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In *Proceedings of the IEEE*, volume 77, pages 257–286, February 1989.

[19] R. Riedi. *Long range dependence : theory and applications*, chapter Multifractal processes. Boston, MA: Birkhauser, 2000. Also Technical Report, ECE Dept. Rice Univ., TR 99-06.

[20] O. Ronen, J.R. Rohlicek, and M. Ostendorf. Parameter Estimation of Dependance Tree Models Using the EM Algorithm. *IEEE Signal Processing Letters*, 2(8):157–159, August 1995.

[21] P. Smyth, D. Heckerman, and M.I. Jordan. Probabilistic Independence Networks for Hidden Markov Probability Models. *Neural Computation*, 9(2):227–270, 1997.

[22] G.W. Wornell and A.V. Oppenheim. Estimation of Fractal Signals from Noisy Measurements Using Wavelets. *IEEE Transactions on Signal Processing*, 40(3):611–623, March 1992.