

Placeholder  

---

Image

Placeholder  

---

Image

# SCATTERING HIDDEN MARKOV TREES

IMAGE REPRESENTATION AND SCATTERING TRANSFORM  
MODELING

Jean-Baptiste REGLI

2013-2014

RESEARCH REPORT

Academic supervisor : James Nelson  
Sponsor : Dstl/UCL Impact studentship

UCL  
Department of Statistical Science  
London



# Contents :

<b>1</b>	<b>Introduction :</b>	<b>5</b>
1.1	Need for a better signal representation : . . . . .	5
1.2	Image representation : . . . . .	6
1.2.1	Intuition of a “good” image representation : . . . . .	6
1.2.2	Formalization of a “good” image representation : . . . . .	7
1.2.3	State of the art in image representation : . . . . .	9
1.3	Probabilistic graphical model : . . . . .	9
1.4	Outline of the report : . . . . .	9
1.5	Contribution : . . . . .	9
<b>2</b>	<b>The Scattering transform :</b>	<b>10</b>
2.1	Scattering wavelets : . . . . .	10
2.2	Scattering Convolution Network : . . . . .	13
2.3	Properties of the scattering transform : . . . . .	14
2.3.1	Non-expansivity : . . . . .	14
2.3.2	Energy preservation : . . . . .	14
2.3.3	Translation invariance : . . . . .	16
2.3.4	Lipschitz continuity to the action of diffeomorphisms : . . . . .	16
2.4	Application to classification : . . . . .	17
<b>3</b>	<b>Probabilistic graphical models :</b>	<b>18</b>
3.1	Bayesian Network : . . . . .	18
3.1.1	Architecture : . . . . .	18
3.1.2	Learning : . . . . .	19
	Expectation-maximization : . . . . .	20
	Variational Bayes : . . . . .	20
3.1.3	Inference : . . . . .	20
3.2	Markov Models : . . . . .	20
3.2.1	Architecture : . . . . .	20
3.2.2	Learning : . . . . .	20
3.2.3	Inference : . . . . .	20
<b>4</b>	<b>Scattering hidden Markov tree :</b>	<b>21</b>
4.1	SCHMT model and related works : . . . . .	22
4.2	Hypothesis : . . . . .	24
4.3	Learning the tree structure : . . . . .	26
4.3.1	E-Step : . . . . .	27
	Upward recursion : . . . . .	27
	Downward recursion : . . . . .	30
	Conditional properties : . . . . .	30
4.3.2	M-Step : . . . . .	31

4.3.3	EM algorithm : . . . . .	31
4.4	Classification : . . . . .	31
<b>5</b>	<b>Experimental results :</b>	<b>34</b>
5.1	Shapes : . . . . .	34
5.2	Sonar Imagery : . . . . .	34
<b>6</b>	<b>Conclusion :</b>	<b>37</b>
<b>7</b>	<b>Acknowledgements :</b>	<b>38</b>

# List of figures :

1.1	High dimensional signals . . . . .	5
1.2	Translation invariance . . . . .	6
1.3	Stability to deformations . . . . .	7
1.4	Rotation invariance . . . . .	7
2.1	Complex Morlet wavelet. . . . .	11
2.2	The scattering convolutional network architecture . . . . .	14
2.3	Frequency decreasing scattering convolution network . . . . .	16
3.1	Example of Bayesian network. . . . .	19
4.1	Scattering transform tree. . . . .	21
4.2	Scattering Hidden Markov Tree. . . . .	22
4.3	Wavelet hidden Markov tree. . . . .	23
4.4	Two populations - experiment 1. . . . .	25
4.5	Two populations - experiment 2. . . . .	25
4.6	Persistence - experiment 1. . . . .	26
4.7	Persistence - experiment 2. . . . .	26
5.1	Shape classification : square . . . . .	34
5.2	Shape classification : triangle . . . . .	35
5.3	Seabed patches . . . . .	35
5.4	Ripple patches . . . . .	35

# 1 Introduction :

Nowadays statistical signal processing and machine learning methods are applied to a wide range of problems from prediction to optimal control. While some of those application involves low-dimensional data, full branches of those fields are dedicated to studying high-dimensional learning problems. High-dimensional problems have been made very popular lately for several reasons. First over the couple last decades data got cheaper to collect. This mainly because of the progress made on digital cameras, microphones or any sensors. Second the cost of data storage also massively decrease during that time. And finally the computational power available today allows to try more computationally intensive methods. And thus methods to leverage those data have been developed across the range of applications in machine learning and statistical signal processing.

This document is restricted to one of those applications, high-dimensional classification problems. And even more precisely it aims at introducing a probabilistic approach to high-dimensional classification problems.

## 1.1 Need for a better signal representation :

Consider the problem of learning the labeling function  $f$ , say, given  $N$  sampled training values  $\{bf x_i, y_i = f(\mathbf{x}_i)\}_{i \leq N}$  where  $x(i) = \{x_1 \dots x_s\}$  where for all  $i \in \llbracket 1, N \rrbracket$ ,  $d \sim 10^6$ ,  $bf x(.) \in \mathbb{R}$  and  $y_i \in \mathbb{N}$ . Examples of such signals are speech waveforms or digital photographs.

A naive solution to this problem would be to infer the class of a new realization  $\mathbf{x}$  by looking at its neighbors in a similar fashion to K-Nearest Neighbors (KNN), for example. This approach is sound for low-dimensional problems [Cover and Hart, 1967]. However it shows limitations in high dimensional cases [Beyer et al., 1999] because the number of samples required to find a neighbor of a new realization  $\mathbf{x}_{new}$  grows exponentially with the number of dimensions.



FIGURE 1.1 – (a) Sound waveform (b) Picture



FIGURE 1.2 – A human can easily tell that those two images are from the same class.

However one can make the assumption that the signal  $\mathbf{x}$  belongs to a manifold  $\Omega$  of  $\mathbb{R}^d$  and this leads to two cases. The subset  $\Omega$  can be low dimensional and there is no curse of dimensionality once the manifold has been isolated. The task at hand is thus manifold learning ([Lin and Zha, 2008], [Zhang et al., 2012]) or sparse dictionary representation ([Kreutz-Delgado et al., 2003]). However for complex signals the manifold  $\Omega$  is also high dimensional. In this case one has to reduce the volume of the space without losing the crucial information necessary to recognize the signal. Hence one has to reduce the volume of  $\Omega$  according to the property of the classification function  $f$ .

## 1.2 Image representation :

A mapping  $\Phi$  is sought which represents the signal in a new space such that the classification task is simpler. The space should not only capture the main information and discriminatory content in the data but it should also remain stable with respect to appropriate transformations and deformations. Before providing a formal mathematical description it is instructive to consider the following intuitive example.

### 1.2.1 Intuition of a “good” image representation :

Properties of a “good” representation for classification can be intuited by looking how humans handle the classification/labeling of visual stimulus and what are the elements ensuring good generalization capacities. Following this approach one can intuit the following properties :

- The projection has to be informative enough to permit classification. This is ensure that  $\Phi$  preserves separability between the different classes.
- The mapping also has to be invariant to translations. Indeed to a human eye there is no difference in the information carried by a signal if it is shifted. This means that the  $\Phi$  has to provide similar, if not equal, outputs for shifted versions of the same signal.
- To some extent the mapping has to be stable under deformations is also a required. Once again to a human eye, it is still possible to recognize a signal if it has undergone -small- deformations. Yet if the deformations are too important the informational content of the signal can be lost. This means that to a certain degree the answer of the projection to morphed realizations of the same signal should be similar. However one need to define a limit to this invariance to ensure that the representation created is still informative enough.



FIGURE 1.3 – A human can easily tell that (a) and (b) are from the same class. (c) can still be recognized even though it is slightly more challenging.



FIGURE 1.4 – A human can easily tell that (a) and (b) are from the same class. (c) could be a '6' slightly rotated or a '9' heavily rotated.

- Again to a certain degree the projection has to be invariant to rotations. Only local invariant to rotation is wanted because excessive rotation applied to the original signal can be destructive for the information carried (see Figure 1.4). Solutions based on the method described in this document exist [Sifre and Mallat, 2013] but they are still work in progress and thus they will not be addressed nor used in this review.

### 1.2.2 Formalization of a “good” image representation :

Throughout, attention will be restricted to square-integrable  $d$ -dimensional functions over the reals, namely  $\mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d)$ . For more accuracy, the classification function is split in two stages.  $f$  will now denote the soft classification function, **i.e.**  $f(\mathbf{x}) \in \mathbb{R}^K$ , and represents the distance to the separating surface.  $h$  is the labeling function and  $y = h \circ f(\mathbf{x})$  is now the label associated to an signal  $\mathbf{x}$ . To be informative enough, a representation must preserve separability between elements of different classes. This is encapsulated by the following definition.

**Definition 1.2.1. *Preservation of the separability between classes***

*A representation  $\Phi$  preserves separability if all elements of two different classes are distant of at least a margin  $C$  in the representation space, **i.e.** :*

$$\forall x, x' \in \mathbb{R}^d \exists C \in \mathbb{R} \text{ s.t. } h \circ f(x) \neq h \circ f(x') \Rightarrow \|\Phi(x) - \Phi(x')\| \geq C^{-1}$$

Translations in the input space should not affect the representation. In this document let  $L_{(.)}$  be the translation operator for the function in  $\mathcal{L}^2(\mathbb{R}^d)$ , **i.e.** for  $\mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d)$  and  $(u, c) \in$



$(\mathbb{R}^d)^2$   $L_c \mathbf{x}(u) = \mathbf{x}(u - c)$ . A mapping  $\Phi$  is translation invariant -respectively : canonical translation invariant if it maps a translated signal to the same point as its original version.

**Definition 1.2.2. Translation invariant**

An operator  $\Phi : \mathcal{L}^2(\mathbb{R}^d) \rightarrow \mathcal{H}$  where  $\mathcal{H}$  is an Hilbert space is translation invariant if :

$$\forall c \in \mathbb{R}^d \text{ and } \forall \mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d) \quad \Phi(L_c \mathbf{x}) = \Phi(\mathbf{x}).$$

**Definition 1.2.3. Canonical translation invariant**

An operator  $\Phi : \mathcal{L}^2(\mathbb{R}^d) \rightarrow \mathcal{H}$  where  $\mathcal{H}$  is an Hilbert space is canonical translation invariant if :

$$\forall \mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d) \quad \Phi(L_a \mathbf{x}) = \Phi(\mathbf{x}) \text{ where } a \in \mathbb{R}^d \text{ is function of } \mathbf{x}.$$

For the usual representation operators instabilities to deformations are known to appear -especially at high frequencies. To prevent this, one would like the representation to be non-expansive.

**Definition 1.2.4. Non-expensive representation** A representation  $\Phi$  is non-expensive if,

$$\forall (\mathbf{x}, \mathbf{x}') \in (\mathcal{L}^2(\mathbb{R}^d))^2 \quad \|\Phi(\mathbf{x}) - \Phi(\mathbf{x}')\| \leq \|\mathbf{x} - \mathbf{x}'\|. \quad (1.1)$$

The stability to deformations of a non-expansive operator can be expressed as its Lipschitz continuity to the action of deformations close to translations [Mallat, 2012]. Such a diffeomorphism transform can be expressed as,

$$\begin{aligned} L_\tau : \mathcal{L}^2(\mathbb{R}^d) &\rightarrow \mathcal{L}^2(\mathbb{R}^d) \\ \mathbf{x} &\rightarrow \mathbf{x}((.) - \tau(.)) \end{aligned}$$

where  $\tau(u) \in \mathbb{R}^d$  is a displacement field.

**Definition 1.2.5. Lipschitz continuous** A translation invariant operator  $\Phi$  is said to be Lipschitz continuous to the action of  $\mathcal{C}^2$  diffeomorphisms if for any compact  $\Omega \in \mathbb{R}^d$  there exists  $C$  such that for all  $f \in \mathcal{L}^2(\mathbb{R}^d)$  supported in  $\Omega$  and all  $\tau \in \mathcal{C}^2(\mathbb{R}^d)$ ,

$$\|\Phi(\mathbf{x}) - \Phi(L_\tau \mathbf{x})\|_H \leq C \|\mathbf{x}\| \left( \sup_{u \in \mathbb{R}^d} |\nabla \tau(u)| + \sup_{u \in \mathbb{R}^d} |H\tau(u)| \right) \quad (1.2)$$

where  $\nabla \tau(u)$  is a matrix whose norm  $|\nabla \tau(u)|$  measure the deformation amplitude at  $u$  and  $H\tau(u)$  is the Hessian matrix of the deformation whose norm  $|H\tau(u)|$  measure its amplitude.

Hence a Lipschitz continuous operator  $\Phi$  is almost invariant to deformations by  $\tau(.)$ , up to the first and second order deformations terms. The Equation 1.2 also implies that  $\Phi$  is invariant to global translations but this is already enforced by the translation invariance requirement.

### 1.2.3 State of the art in image representation :

A classic representational method is the modulus of the Fourier transform. This operator is informational enough to allow -to a certain extent- discrimination between different type of signal [Baker et al., 2014]. It is also translation invariant [Bracewell, 1965]. However it is well known that those operators present instabilities to deformation at high frequencies [Hörmander, 1971] and thus are not Lipschitz continuous to the action of diffeomorphisms.

Wavelet transform is another popular representation method. Again they provide a representation suitable for classification of different signals [De Chazal et al., 2000]. Plus by grouping high frequencies into dyadic packet in  $\mathbb{R}^d$ , wavelet operators are stable to -small- deformations [Bruna and Mallat, 2013].

$$W\mathbf{x} = \begin{pmatrix} \mathbf{x} * \phi \\ \mathbf{x} * \psi_\lambda \end{pmatrix} \begin{matrix} \rightarrow \text{averaging part} \\ \rightarrow \text{high frequency part} \end{matrix} \quad (1.3)$$

However only the averaging part of a wavelet is invariant to translation and thus wavelets themselves are known to be non-invariant to translations.

Another signal representation method popular are the convolutional neural networks [Le-Cun and Bengio, 1995]. As opposed to the two previously mentioned methods, those operators are not fixed but learned from the data [Simard et al., 2003]. Over the past few year they have provided state of the art results on many standard classification tasks, on images datasets such as MNIST, CIFAR [Hinton et al., 2012] and ImageNet [Krizhevsky et al., 2012] as well as on speech processing problem such as TIMIT [Abdel-Hamid et al., 2012]. Those good results are used to advocate that those networks are learning "good" representations. However it seems that in certain cases they learn representation of the data that are, for example, not invariant to deformations [Szegedy et al., 2013].

## 1.3 Probabilistic graphical model :

??? - not sure yet

## 1.4 Outline of the report :

The remainder of this document is organized as follow. Section 2 summarizes and explains recent work Stéphane Mallat and his group on the Scattering Transform (ST), a wavelet-based operator fulfilling all the properties of what we have defined as a "good" representation for signal classification. Section 3 introduces the concept of Probabilistic Graphical Models (PGMs) as generative models that can be used -among other tasks- for classification. Section 4 describes how the representation produced by the scattering transform can be modeled by an hidden Markov tree, using what we have named Scattering Hidden Markov Trees (SCHMTs). Finally Section 5 provides some examples of applications.

## 1.5 Contribution :

The first two sections of this documents are mostly dedicated at introducing preexisting notions and concepts that are useful to later construct the Scattering Hidden Markov Trees. Hence readers already familiar with with Scattering Transforms or Probabilistic Graphical Models can respectively skip Section 2 or 3 and focus on the novelties introduced in Section 4.

## 2 The Scattering transform :

This section describes the construction of a mathematical operator - the scattering transform (ST)- designed to generate what has been defined earlier as an interesting representation of signal (see 1.2). This operator delocalizes signal informational content into scattering decomposition paths, computed by cascading wavelet/modulus operators through an architecture similar to a Convolutional Neural Network (CNN) where the synaptic weights would be given by a wavelet operator instead of learned.

The remainder of the chapter is organized as follow. First, Section 2.2 defines the scattering operators. Second, Section 2.2 describes how those operators can be stacked to create a Scattering Convolutional Network (SCN), an architecture comparable to CNNs. Then Section 2.3 reviews some of the SCNs' important properties. Finally, Section 2.4 presents how the scattering transform is usually used in classification tasks.

### 2.1 Scattering wavelets :

A two-dimensional directional wavelet is obtained by scaling and rotating a single band-pass filter  $\psi$ . If one let  $G$  be a discrete, finite rotation group of  $\mathbb{R}^2$ , multi-scale directional wavelet filters are defined for any scale  $j \in \mathbb{Z}$  and rotation  $r \in G$  by,

$$\psi_{2^j r}(u) = 2^{2j} \psi(2^j r^{-1} u). \quad (2.1)$$

To simplify the notations, let now  $\lambda = \lambda(j, r) \stackrel{d}{=} 2^j r \in G \times \mathbb{Z}$ .

A wavelet transform filters the signal  $\mathbf{x}$  using a family of wavelets  $\{\mathbf{x} * \psi_\lambda(u)\}_\lambda$  computed from a filter bank of dilated and rotated wavelets having no orthogonality property. This generates a multi-scales and multi-orientations representation of the input signal.

If  $u.u'$  and  $\|u\|$  define respectively the inner product and the norm in  $\mathbb{R}^2$ , the Morlet wavelet  $\psi$  is an example of wavelet given by,

$$\psi(u) = C_1(e^{iu.\xi} - C_2)e^{\|u\|^2/(2\sigma^2)},$$

where  $C_1$ ,  $\xi$  and  $\sigma$  are meta-parameters of the wavelet and  $C_2$  is adjusted so that  $\int \psi(u) du = 0$ . Figure 2.1 shows a Morlet wavelet for  $\xi = 3\pi/4$  and  $\sigma = 0.85$ .

As opposed to the Fourier sinusoidal waves, wavelets are operators stable to local  $\mathcal{L}^2$  deformations as they can be expressed as localized waveforms [Mallat, 1999]. However, as wavelet transform computes a convolutions with a wavelet basis, the resulting transform is a translation covariant operator [Bruna and Mallat, 2013].

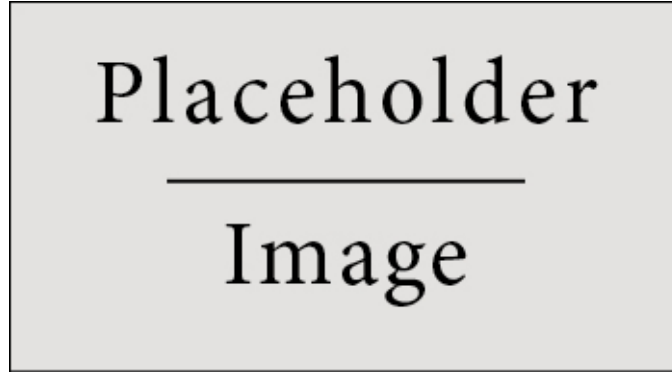


FIGURE 2.1 – Complex Morlet wavelet.

To ensure a translation invariant behavior for an operator initially commuting with translations, one has to introduce a non-linearity. Integration is an example of such a non-linearity. For example, let  $R$  be a linear or non-linear operator commuting with translations  $L_c$ , **i.e.**  $R(L_c \mathbf{x}) = L_c R(\mathbf{x})$ , then the integral  $\int R(\mathbf{x}(u)) du$  is translation invariant. Applying such a non-linearity to a wavelet transform is equivalent to setting  $R(\mathbf{x}) = \mathbf{x} * \psi_\lambda$  and one gets the trivial invariant,

$$\int \mathbf{x} * \psi_\lambda(u) du = 0,$$

for all signal  $\mathbf{x}$  as  $\int \psi_\lambda(u) du = 0$ .

However for the scattering operator to preserve its informative character, one has to ensure a non-vanishing integral. To do so a second operator  $M$  has to be introduced such that  $M \circ R(\mathbf{x}) = M(\mathbf{x} * \psi_\lambda)$ . If  $M$  was a linear transformation commuting with translation then the integral still vanishes. Hence one has to choose  $M$  among the non-linear operators.

Keeping in mind that the scattering transform has to be stable to deformations and taking advantages of the wavelet transform stability to small deformations in the input space, one also imposes that  $M$  commutes with deformations,

$$\forall \tau(u), \quad M L_\tau = L_\tau M.$$

If a weak differentiability condition is added, one can prove [Bruna, 2012] that  $M$  must necessarily be a point-wise operator, **i.e.**  $M \circ R(\mathbf{x}(u))$  only depends on the value of  $\mathbf{x}(u)$ .

Finally, by adding an  $\mathcal{L}^2(\mathbb{R}^2)$  stability constraint,

$$\forall (\mathbf{x}, \mathbf{x}') \in \mathcal{L}^2(\mathbb{R}^2)^2, \quad \|M \circ R(\mathbf{x})\| = \|\mathbf{x}\| \quad \text{and} \quad \|M \circ R(\mathbf{x}) - M \circ R(\mathbf{x}')\| \leq \|\mathbf{x} - \mathbf{x}'\|,$$

one can also show [Bruna, 2012] that necessarily  $M(R(\mathbf{x})) = e^{i\alpha} |R(\mathbf{x})|$ . For the scattering transform, the simplest solution  $\alpha = 0$  is chosen and therefore the resulting coefficients are the  $\mathcal{L}^1(\mathbb{R}^2)$  norms :

$$\|\mathbf{x} * \psi_\lambda\|_1 = \int |\mathbf{x} * \psi_\lambda| du$$

The family of the  $\mathcal{L}^1(\mathbb{R}^2)$  normed wavelet  $\{\|\mathbf{x} * \psi_\lambda\|_1\}_\lambda$  generates a crude signal representation which measures the sparsity of the wavelet coefficients. It can be proven that the signal

$\mathbf{x}$  can be reconstructed from  $\{|\mathbf{x} * \psi_\lambda(u)|\}_\lambda$  up to a multiplicative constant [Waldspurger et al., 2015]. This means that the information loss in  $\{|\mathbf{x} * \psi_\lambda|_1\}_\lambda$  occurs during the integration of the absolute value  $|\mathbf{x} * \psi_\lambda(u)|$  which removes all non-zero frequencies. However those components can be recovered by calculating the wavelet coefficients  $|\mathbf{x} * \psi_{\lambda_1}| * \psi_{\lambda_2}(u)$  of the new signal  $|\mathbf{x} * \psi_{\lambda_1}|$ . By doing so their  $\mathcal{L}^1(\mathbb{R}^2)$  norms define a much larger family of invariants :

$$\forall(\lambda_1, \lambda_2) \quad \|\mathbf{x} * \psi_{\lambda_1}| * \psi_{\lambda_2}\|_1 = \int \|\mathbf{x} * \psi_{\lambda_1}(u)| * \psi_{\lambda_2}\| du$$

By further iterating on the wavelet/modulus operators more translation invariant coefficients can be computed. The building bloc of such a model -the scattering propagator- is thus the absolute value of the convolution between a wavelet and the input signal.

**Definition 2.1.1. Scattering Propagator**

The scattering operator  $U$  for a scale and an orientation  $\lambda \in G \times \mathbb{Z}$  is defined as the absolute value of the input convolved with the wavelet operator at this scale and orientation.

$$U[\lambda](\mathbf{x}) \stackrel{d}{=} |\mathbf{x} * \psi_\lambda| \quad (2.2)$$

**Definition 2.1.2. Path Ordered Scattering Propagators**

Any sequence  $p = (\lambda_1, \lambda_2, \dots, \lambda_m)$  where  $\forall i \in \llbracket 1, m \rrbracket \quad \lambda_i \in G \times \mathbb{Z}$  defines a path of length  $m$ , i.e. the ordered product of non-linear and non-commuting operators,

$$\begin{aligned} U[p]\mathbf{x} &\stackrel{d}{=} U[\lambda_m] \dots U[\lambda_2]U[\lambda_1](\mathbf{x}) \\ &= ||| |\mathbf{x} * \psi_{\lambda_1}| * \psi_{\lambda_2}| \dots | * \psi_{\lambda_m}|. \end{aligned} \quad (2.3)$$

With the convention :  $U[\emptyset]\mathbf{x} = \mathbf{x}$

From there one can provide a first formal definition of the scattering transform.

**Definition 2.1.3. Scattering Coefficient**

A scattering coefficient along the path  $p$  is defined as an integral of the  $p$  ordered scattering propagators, normalized by the response of a Dirac :

$$\bar{S}[p](\mathbf{x}) \stackrel{d}{=} \mu_p^{-1} \int U[p]\mathbf{x}(u) du \quad (2.4)$$

with,

$$\mu_p \stackrel{d}{=} \int U[p]\delta(u) du$$

We shall see later (see Section 2.3) that each scattering coefficient  $\bar{S}[p](\mathbf{x})$  is -as desired- invariant to translation of the input  $\mathbf{x}$  and Lipschitz continuous to deformations. But for classification tasks, one might only want to compute localized descriptors only invariant to translations smaller than a predefined scale  $2^J$ , while keeping the spatial variability at scales larger than  $2^J$ . This can be achieved by localizing the scattering integral with a scaled spatial window  $\phi_{2^J}(u) = 2^{-2J}\phi(2^{-2J}u)$ . And this yields to the definition of the windowed scattering transform.

**Definition 2.1.4. -Windowed- Scattering Coefficient Of Order  $m$** 

If  $p$  is a path of length  $m \in \mathbb{N}$ , the -windowed- scattering coefficient of order  $m$  at scale  $2^J$  ( $J \in \mathbb{N}$ ) is defined as :

$$\begin{aligned}
S_J[p](\mathbf{x}) &\stackrel{d}{=} U[p]\mathbf{x} * \phi_{2^J}(u) \\
&= \int U[p]\mathbf{x}(v) \phi_{2^J}(u - v) dv \\
&= ||| \mathbf{x} * \psi_{\lambda_1} | * \psi_{\lambda_2} | \dots | * \psi_{\lambda_m} | * \phi_{2^J}(u)
\end{aligned} \tag{2.5}$$

With the convention :  $S_J[\emptyset]\mathbf{x} = \mathbf{x} * \phi_{2^J}$

**2.2 Scattering Convolution Network :**

This section introduces the scattering transform as an iterative process over a one-step operator and creates a parallel with convolutional neural networks [LeCun et al., 2010]. For  $J \in \mathbb{N}$  let  $U_J[\Omega] \stackrel{d}{=} \{U_J[p]\}_{p \in \Omega}$  and  $S_J[\Omega] \stackrel{d}{=} \{S_J[p]\}_{p \in \Omega}$ . They defines families of operators indexed by a path set  $\Omega$ .

One can compute a windowed scattering transform by iterating over the one-step -windowed- propagator  $U_J$ .

**Definition 2.2.1. One-step propagator**

The one-step propagator  $U_J$  can be defined as,

$$\forall \mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d) \quad U_J\mathbf{x} = \{A_J\mathbf{x}, (U[\lambda]\mathbf{x})_{\lambda \in \Lambda_J}\}, \tag{2.6}$$

where  $A_J\mathbf{x} = \mathbf{x} * \phi_{2^J}$ ,  $U[\lambda]\mathbf{x} = |\mathbf{x} * \psi_\lambda|$  and  $\Lambda_J = G \times \llbracket 0, J \rrbracket$ .

Indeed after calculating  $U_J\mathbf{x}$ , applying  $U_J$  again to each  $U[\lambda]$  generates a larger infinite family of functions. And since  $U[\lambda]U[p] = U[p + \lambda]$  and  $A_JU[p] = S_J[p]$  it holds that,

$$\forall \mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d) \quad U_JU[p]\mathbf{x} = \{S_J[p]\mathbf{x}, (U[p + \lambda]\mathbf{x})_{\lambda \in \Lambda_J}\}, \tag{2.7}$$

Let now  $\Lambda_J^m$  be a set of path of length  $m$  with the convention  $\Lambda_J^0 = \{\emptyset\}$ , its propagation is,

$$\forall \mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d) \quad U_JU[\Lambda_J^m]\mathbf{x} = \{S_J[\Lambda_J^m]\mathbf{x}, (U[\Lambda_J^{m+1}]\mathbf{x})_{\lambda \in \Lambda_J}\}. \tag{2.8}$$

Hence  $S_J[\mathcal{P}_J]\mathbf{x}$  can be computed from  $\mathbf{x} = U[\emptyset]\mathbf{x}$  by iteratively computing  $U_JU[\Lambda_J^m]\mathbf{x}$  for  $m$  going from 0 to  $\infty$ . This iterative process is illustrated in Figure 2.2 and one can notice that the scattering calculation as the same general architecture as the convolutional neural networks introduced by [LeCun and Bengio, 1995]. Both CNN and scattering convolutional network (SCN) cascade convolutions and a “pooling” non linearity. However while convolutional neural networks use kernel filters learned from the data with back-propagation algorithm, SCNs use a fixed wavelet filter bank.



FIGURE 2.2 – see mallat long paper p1341

## 2.3 Properties of the scattering transform :

Now that the scattering coefficient have been defined one can be interested in the characteristics of such a data representation. This section provides an overview of some of the properties of the scattering transform. It also describes an approximation to the scattering convolution network defined in the previous section, leading to computationally tractable networks.

### 2.3.1 Non-expansivity :

The scattering propagator  $U_J \mathbf{x} = \{A_J \mathbf{x}, (|W_J \mathbf{x}|)_{\lambda \in \Lambda_J}\}$  results of the composition of a wavelet transform  $W_J$  that is unitary and of a modulus operator that is non-expansive - as  $\forall(a, b) \in \mathbb{C}^2 \ ||a| - |b|| \leq |a - b|$  - and is thus non-expansive. Since  $S_J[\mathcal{P}_J]$  iterates on  $U_J$ , which is non-expansive, the proposition (proved by [Lohmiller and Slotine, 1998]) proves that  $S_J[\mathcal{P}_J]$  is also non-expansive.

#### Proposition 2.3.1. *Non-expansive*

*The scattering transform is non expansive.*

$$\forall(\mathbf{x}, \mathbf{x}') \in \mathcal{L}^2(\mathbb{R}^d)^2 \quad \|S_J[\mathcal{P}_J]\mathbf{x} - S_J[\mathcal{P}_J]\mathbf{x}'\| \leq \|\mathbf{x} - \mathbf{x}'\| \quad (2.9)$$

### 2.3.2 Energy preservation :

Each propagator  $U[\lambda]\mathbf{x} = |\mathbf{x} * \psi_\lambda|$  captures the frequency energy contained in the signal  $\mathbf{x}$  over a frequency band covered by the Fourier transform  $\hat{\psi}_\lambda$  and propagates this energy towards lower frequencies. It can thus be proved that under some assumption on the wavelet -admissible wavelet, the whole scattering energy ultimately reaches the minimum frequency  $2^{-J}$  and is trapped by the low-pass filter  $\phi_{2^J}$ . Thus the energy propagated by a -windowed-scattering transform goes to 0 as the path length increases, implying that  $\|S_J[\mathcal{P}_J]\| = \|x\|$

But prior to showing this, one must states the necessary assumptions to be made on the wavelet used.

*Note.* Formal proofs of most of those properties can be found in [Mallat, 2012].

#### Definition 2.3.2. *Admissible scattering wavelet*

*A scattering wavelet  $\psi$  is admissible if there exist  $\eta \in \mathbb{R}^d$  and  $\rho \geq 0$ , with  $|\hat{\rho}(\omega)| \leq |\hat{\phi}(2\omega)|$  and  $\hat{\rho}(\omega) = 0$ , such that the function,*

$$\hat{\Psi}(\omega) = |\hat{\rho}(\omega - \eta)|^2 - \sum_{k=1}^{+\infty} k(1 - |\hat{\rho}(2^{-k}(\omega - \eta))|^2), \quad (2.10)$$

satisfies,

$$\alpha = \inf_{1 \leq |\omega| \leq 2} \sum_{j=-\infty}^{+\infty} \sum_{r \in G} \hat{\Psi}(2^{-j}r^{-1}\omega) \left| \hat{\psi}(2^{-j}r^{-1}\omega) \right|^2 > 0. \quad (2.11)$$

For an admissible wavelet one can prove that the scattering transform conserves the energy of the signal.

**Theorem 2.3.3. Energy conservation**

If the scattering wavelet  $\psi$  is admissible, then for all signal  $\mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d)$ ,

$$\lim_{m \rightarrow +\infty} \|U[\Lambda_J^m] \mathbf{x}\|^2 = \lim_{m \rightarrow +\infty} \sum_{n=m}^{+\infty} \|S_J[\Lambda_J^n] \mathbf{x}\|^2 = 0, \quad (2.12)$$

and

$$\|S_J[P_J] \mathbf{x}\|^2 = \|\mathbf{x}\|. \quad (2.13)$$

The proof of the theorem 2.3.3 also shows that the scattering energy propagates progressively towards lower frequencies and that the energy of  $U[p] \mathbf{x}$  is mainly concentrated along frequency decreasing paths  $p = (\lambda_k)_{k \leq m}$ , **i.e.** for which  $|\lambda_{k+1}| \leq |\lambda_k|$ . The energy contained in the other paths is negligible and thus for the rest of the document only frequency decreasing paths are be considered.

Moreover, the decay of  $\sum_{n=m}^{+\infty} \|S_J[\Lambda_J^n] x\|^2$  implies that there exist a path length  $m > 0$  after which all longer path can be neglected. For signal processing applications, this decay appears to be exponential. And for classification applications path of length  $m = 3$  provides the most interesting results [Andén and Mallat, 2011], [Bruna and Mallat, 2010].

The two restrictions stated above yield to an easier parametrization of a scattering network. Indeed when only the frequency decreasing paths up until a given order a scattering network is completely defined by :

- $\psi$  : The admissible wavelet used. Unless stated otherwise the Morlet wavelet is used.
- $M$  : The maximum path length considered.
- $J$  : The coarsest scale level considered.
- $L$  : The number of orientation considered, which can be define as the cardinality of the previously define  $G$ .

Hence for a given set of parameter  $(\psi, M, J, L)$ , one can generate one and only one scattering network -with frequency decreasing paths. Each node  $i$  of this network generates a -possibly empty- set of of nodes of size  $(j_i - 1) \times |G|$  where  $j_i$  is the scale of node  $i$  and  $|G|$  is the number of orientations considered. Finally the number of nodes  $O$  of this network can be expressed as,

$$O = \sum_{m=0}^{M-1} \binom{J}{m} \cdot |G|^m \quad (2.14)$$

and it has the architecture displayed by Figure 2.3.





FIGURE 2.3 – Frequency decreasing scattering convolution network

### 2.3.3 Translation invariance :

The translation invariance of the scattering transform  $S_J[\mathcal{P}_J]$  can be proved for a limit metric when  $J$  goes to infinity. To do so one can first prove that the scattering distance  $\|S_J[\mathcal{P}_J]\mathbf{x} - S_J[\mathcal{P}_J]\mathbf{x}'\|$  converges when  $J$  goes to infinity - as it is non-increasing when  $J$  increases (see section 2.3.1). From there one can bound the distance between the scattering transform of the signal and the one of its translated version  $\|S_J[S_J[\mathcal{P}_J]\mathcal{L}_c\mathbf{x} - \mathcal{P}_J]\mathbf{x}\|$  and prove that this bound tends to 0 when  $J$  goes to infinity. This proves the translation invariance.

#### Theorem 2.3.4. *Translation invariance*

For admissible scattering wavelets,

$$\forall \mathbf{x} \in \mathcal{L}^2(\mathbb{R}^d), \forall c \in \mathbb{R}^d \quad \lim_{J \rightarrow \infty} \|S_J[\mathcal{P}_J]\mathcal{L}_c\mathbf{x} - S_J[\mathcal{P}_J]\mathbf{x}\| = 0 \quad (2.15)$$

### 2.3.4 Lipschitz continuity to the action of diffeomorphisms :

The Lipschitz continuity to the action of diffeomorphisms of  $\mathbb{R}^d$  can be proved for deformations sufficiently close to a translation. Such diffeomorphisms map  $u$  to  $u - \tau(u)$  where  $\tau(u)$  is a displacement field such that  $\|\nabla\tau\|_\infty < 1$  - i.e. invertible transformations [Bruna and Mallat, 2013]. Let  $L_\tau\mathbf{x}(u) = \mathbf{x}(u - \tau(u))$  denotes the action of such diffeomorphisms on the signal  $\mathbf{x}$ . Once again one can find an upper bound to the distance between the scattering transform of the signal and the one of its deformed version  $\|S_J[\mathcal{P}_J]\mathcal{L}_\tau\mathbf{x} - S_J[\mathcal{P}_J]\mathbf{x}\|$ . With a bit of work on this bound one can then prove that the consequences of the action of  $L_\tau$  is bounded by a translation term proportional to  $2^{-J}\|\tau\|_\infty$  and a deformation error proportional to  $\|\nabla\tau\|_\infty$ . Finally some more work on the bounding term provides the Lipschitz continuity.

#### Theorem 2.3.5. *Lipschitz continuity to the action of diffeomorphisms*

There exists  $C$  such that all  $\mathbf{x} \in \mathcal{L}(\mathbb{R}^d)$  with  $\|U[\mathcal{P}_J]\mathbf{x}\|_1 < \infty$  and all  $\tau \in \mathcal{C}^2(\mathbb{R}^d)$  with  $\|\nabla\tau\|_\infty < \frac{1}{2}$  satisfy,

$$\|S_J[\mathcal{P}_J]\mathcal{L}_\tau\mathbf{x} - S_J[\mathcal{P}_J]\mathbf{x} + \tau \cdot \nabla S_J[\mathcal{P}_J]\mathbf{x}\| \leq C \|U[\mathcal{P}_J]\mathbf{x}\|_1 K(\tau), \quad (2.16)$$

with

$$K(\tau) = 2^{-2J} \|\tau\|_\infty^2 + \|\nabla\tau\|_\infty \left( \max \left( \log \frac{\|\Delta\tau\|_\infty}{\|\nabla\tau\|_\infty}, 1 \right) \right) + \|H\tau\|_\infty \quad (2.17)$$

*Remark.* If the case where  $2^J \gg \|\tau\|_\infty$  and  $\|\nabla\tau\|_\infty + \|H\tau\|_\infty \ll 1$ , then  $K(\tau)$  becomes negligible and the displacement field  $\tau(u)$  can be estimated at each  $u \in \mathbb{R}^d$ . This can be done by solving the linear equation resulting from 2.16 under the assumptions mentioned above,

$$\forall p \in \mathcal{P}_J \|S_J[p]\mathcal{L}_\tau \mathbf{x} - S_J[p]\mathbf{x} + \tau \cdot \nabla S_J[p]\mathbf{x}\| \approx 0. \quad (2.18)$$

This estimate of the displacement field can be used for many applications such as object tracking in video sequences or image sequence restoration [Brailean and Katsaggelos, 1996].

## 2.4 Application to classification :

The scattering transform has been designed in order to provide a “good” representation for signal classification. The classical way of using it for such a task is to use the features generated by the scattering transform of the dataset considered as inputs for a discriminative classifier (e.g. Support Vector Machine classifier). Using this model provides results able to compare with state of the art CNNs on some standard datasets *cite mallat*.

## 3 Probabilistic graphical models :

Those models use a *graph based representation of conditional dependence between a set of random variables* and thus encode a complete distribution over a multi-dimensional space in a compact -or factorized- graph. The field of PGMs can be split into two main families, the *Bayesian Networks* (BNs) and the *Markov models* (MMs). As they are graphical models, both families encompass the properties of factorization and independences defined by the graph, but differ when it comes to the specificities of the set of independences they can encode as well as the factorization of the distribution that they can induce ( *cite : Bishop, Christopher M. (2006). "Chapter 8. Graphical Models" (PDF). Pattern Recognition and Machine Learning. Springer. pp. 359–422. ISBN 0-387-31073-8. MR 2247587*).

Add par on how they will be used.

The aim of this section is not to provide a complete overview of the Probabilistic graphical models but rather to introduce some interesting concepts that have been used in our work. Someone with more interest in PGMs could refer to *cite A tutorial on learning BN* or *cite cours dafney koller*. This chapter introduces those two main classes of probabilistic graphical models. Section 3.1 focuses on the Bayesian networks, while section 3.2 provides more details about the Markov models.

### 3.1 Bayesian Network :

A BN is subclass of probabilistic graphical model where the set of random variables and their conditional dependencies are expressed via a *directed acyclic graph* (DAG). The architecture of the Bayesian Networks is further explained in section 3.1.1. Then section 3.1.2 presents a brief overview of the state of the art in terms of learning algorithm for BNs and section 3.1.3 describes the inference mechanism for those networks.

#### 3.1.1 Architecture :

Bayesian networks can be defined as,

**Definition 3.1.1. Bayesian Network**

For a set of random variables  $\mathbf{X} = \{X_i\}_{i \in \llbracket 1, N \rrbracket}$  a Bayesian network consists of a **direct acyclic graph**  $\mathcal{G}$  encoding a set of conditional independence assertions about the random variables in  $\mathbf{X}$  and a set  $P$  of **local probability distribution associated with each variable**. Each node of  $\mathcal{G}$  represent one of the random variable  $X_i$  and each edge  $E_{i \rightarrow j}$  represents the conditional dependence between the nodes  $X_i$  and  $X_j$ .

For such networks the following property holds,

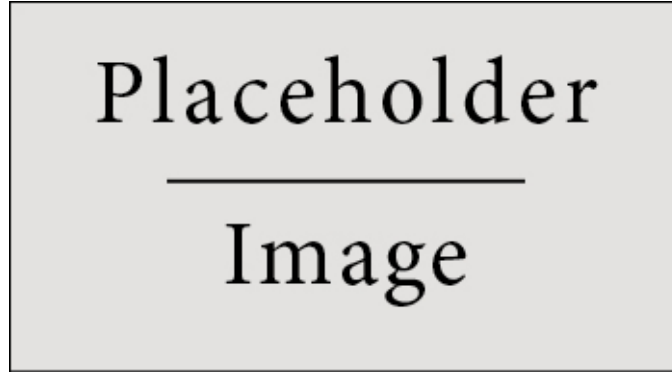


FIGURE 3.1 – Example of Bayesian network.

**Proposition 3.1.2. Conditional independence for Bayesian networks**

In a Bayesian network each node of the graph are "conditionally independent of any subset of the nodes that are not descendants of itself given its parent".

$$P(\{X_i\}_{i \in [1, N]}) = \prod_{i=1}^N P(X_i | \rho(X_i)) \quad (3.1)$$

where  $\rho(X_i)$  are the parents of the node  $X_i$ .

Thus in a Bayesian network a node with no parents is not conditioned on the other random variables. Such a node is called a **prior probability**.

By their architecture, Bayesian networks allow to simplify the computation of the joint probability distribution. For example for the network defined by figure 3.1, one can obtain  $P(X_1, X_2, X_3, X_4, X_5, X_6)$  by the use of chain rules and theory on conditional independent,

$$\begin{aligned} P(X_1, X_2, X_3, X_4, X_5, X_6) &= P(X_6 | X_3, X_4, X_5) P(X_1, X_2, X_3, X_4, X_5) \\ &= P(X_6 | X_3, X_4, X_5) P(X_3 | X_1, X_5) P(X_4 | X_2) P(X_5 | X_2) P(X_1, X_2) \\ &= P(X_6 | X_3, X_4, X_5) P(X_3 | X_1, X_5) P(X_4 | X_2) P(X_5 | X_2) P(X_2 | X_1) P(X_1). \end{aligned}$$

### 3.1.2 Learning :

In some application the structure  $\mathcal{G}$  and the set  $P$  of local probability distribution associated with each variable of the network are provided. In such a case the BNs are "simply" used for inference. The task is then to infer the most probable values for a subset  $\mathbf{Y} \subset \mathbf{X}$ , given a partially complete set of realizations  $\mathbf{X}_{obs} \subset \mathbf{X} \setminus \mathbf{Y}$ .

However, most of the time the full characterization of the BN is not provided. Ignoring the case of missing data, one can split the learning problem into two main categories :

- **Local probability distributions** : In this case the structure of the graph  $\mathcal{G}$  is known and fixed before hand. It can be provided by an expert (e.g. Microsoft trouble shooting system *cite*) or be imposed by some construction rules (e.g. Boltzmann Machine *cite*, Restricted Boltzmann Machine *cite* ...). The task at end is then to learn the parameters governing the local probability distributions of the Bayesian network.
- **Architecture and local probability distributions** : In this case the architecture of the network has to be learned along side with the local probability distributions. This problem is not developed in the rest of this paper, but one could refer to *cite A tutorial on learning BN* for an introduction to this problem.

**Expectation-maximization :**

**Variational Bayes :**

**3.1.3 Inference :**

[http://www.cse.unsw.edu.au/~cs9417ml/Bayes/Pages/Bayesian\\_Networks\\_Inference.html](http://www.cse.unsw.edu.au/~cs9417ml/Bayes/Pages/Bayesian_Networks_Inference.html)

## **3.2 Markov Models :**

Quick overview over the main methods for HMMs.

**3.2.1 Architecture :**

TBD

**3.2.2 Learning :**

TBD

**3.2.3 Inference :**

TBD

## 4 Scattering hidden Markov tree :

Section 2.4 introduced the usage of scattering networks combined with an support vector machine classifier to achieve state of the art classification performances on some problems. However this method provides a boolean answer for each class. Some methods to express the output of an SVM as a probability exists [Platt et al., 1999] but they are just a rescaling of the output and not a true probabilistic approach. If one is interested in a true probabilistic model to describe the scattering coefficient it is quite natural to try to express them as a probabilistic graphical model. Indeed if one hide the propagation step from the scattering transform (see Section 2.2) the scattering network defines a tree structure.

To simplify the notation in the remainder of this section, let  $\mathcal{T}$  denotes the tree structure defines as state above and depicted in figure 4.1. Let  $I$  denotes the total number of nodes -**i.e.** scattering coefficient and let  $S_i$  for  $i \in \llbracket 0, I - 1 \rrbracket$  denotes one of the node  $S[p_i]x \in \mathcal{T}$  for a given path  $p_i = [\lambda_0 \dots \lambda_u]$  ( $u \in \mathbb{N}$ ). Note that  $S_i$  represents a node and does not depends on the signal  $x$ . For a given signal  $x$ , a realization of the node  $i$  is denoted by  $S_i = s_i = S[\lambda_i]x$  where  $\lambda_i$  denotes the path in the scattering tree to the node  $i$ . Note also that in the remainder of the paper the short notation  $i \in \mathcal{T}$  will be use to denote  $\lambda_i$ . Let also use the convention  $S_0 = S[\emptyset]x$ . Finally let  $\rho_i$  and  $\mathcal{C}(i)$  denote respectively the parent of a node  $i$  and the set of children of the node  $i$ . Note also that a node  $S_i$  can have no children, in such a case this node is a leaf of the tree.

The remainder of this section is organized as follows : Section 4.1 introduces related work and provide a description of the SCHMT model. Section 4.2 details the hypothesis needed to develop this model as well as provides some intuition on their validity. Finally Section 4.3 and 4.4 respectively describe the learning algorithm for the parameters of the model and the classification method.



FIGURE 4.1 – Scattering transform tree.



FIGURE 4.2 – Scattering Hidden Markov Tree.

## 4.1 SCHMT model and related works :

The idea behind the SCHMT model is to say that the more detailed representation of the signal is somehow correlated to the less detailed one from which it is generated. More formally this means that for a signal  $x$ ,  $S_i$  is somehow correlated to  $S_{\rho(i)}$ . To do so one could model the scattering network by a Markov tree and assumes the following :

$$P(S_i|\mathcal{T}) = P(S_i|S_{\rho(i)}). \quad (4.1)$$

Those independence properties yield to the graph displays in Figure 4.1. However models trying to describe directly the correlation across coefficients at different scales have been studied for traditional wavelet transforms [Crouse et al., 1998] and it seems that a simple one-step Markovian assumption across scale is not sufficient to describe the complex relationship between wavelet coefficients.

A common approach when a direct Markovian model does not hold is to introduce hidden states and to assume the Markovian property across those states. The observed nodes are then only dependent on their state. This is the architecture adopted for the SCHMT and its graph is represented in Figure 4.2. This model has the following independence properties,

$$P(H_i|\mathcal{T}) = P(H_i|H_{\rho(i)}), \quad (4.2)$$

$$P(S_i|\mathcal{T}) = P(S_i|H_i). \quad (4.3)$$

As the scattering transform is closely related to wavelet transforms it is not surprising to find similar ideas exploited for wavelet trees. MS. Crouse proposed where an Hidden Markov Tree Model is used to model the wavelet coefficients [Crouse et al., 1998] of a classic wavelet trees. Later N. Kingsbury [Kingsbury, 2001] adapted MS. Crouse’s model to his Dual wavelet complex trees. The resulting hidden Markov tree models provides better classification performances than MS. Course’s WHMT as the wavelet used generate a “better” representation of the signal -in the sense defined in Section 1.2.1. Indeed this version can leverage the quasi-translation invariance property of the complex wavelets. The improvement in performances due to the quasi-invariance property provides a good motivation to try the hidden Markov tree modeling on the scattering transform as they have even “better” representational properties (see 2.3. The parameters of the original WHMT were trained using a



FIGURE 4.3 – Wavelet hidden Markov tree.

version of the **Expectation-Maximization** adapted to binary hidden Markov trees. However this learning method suffered from overflowing issues *cite : ? - Durand ?*, JB. Durand [Durand et al., 2004] proposed a smoothed version of the training algorithm preventing overflowing.

A scattering hidden Markov tree is composed by a set of visible nodes  $\{\mathbf{S}_i\}_{i \in \mathcal{T}}$  and a set of hidden node  $\{\mathbf{H}_i\}_{i \in \mathcal{T}}$ . Then the tree is parametrised as follow,

- For any index  $i$  of the tree,  $S_i \in \mathbb{R}^d$  and  $H_i \in \llbracket 1, K \rrbracket$  where  $K$  is the number of possible hidden states.
- The initial hidden state is drawn from a discrete non uniform **initial distribution**  $\pi_0$  such that :

$$\forall k \in \llbracket 1, K \rrbracket \quad \pi_0(k) = P(H_0 = k). \quad (4.4)$$

- For any index  $i$  of the tree, the **emission distribution** describe the probability of the visible node  $S_i$  conditional to the hidden state  $H_i$ ,

$$\forall i \in \mathcal{T} \quad \forall k \in \llbracket 1, K \rrbracket \quad \text{and} \quad \forall s \in \mathbb{R} \quad P(S_i = s | H_i = k) = P_{\theta_{k,i}}(s) \quad (4.5)$$

Where  $P_\theta$  belongs to a parametric distribution family and where  $\theta$  is the vector of **emission parameters**. In the remainder of the paper the emission distribution is a Gaussian so that,

$$\forall i \in \mathcal{T} \quad \forall k \in \llbracket 1, K \rrbracket \quad \text{and} \quad \forall s \in \mathbb{R} \quad P(S_i = s | H_i = k) = \mathcal{N}(\mu_{k,i}, \sigma_{k,i}). \quad (4.6)$$

where  $\theta_{k,i} = (\mu_{k,i}, \sigma_{k,i})$  where  $\mu_{k,i}$  and  $\sigma_{k,i}$  are respectively the mean and the variance of the Gaussian for the  $k$ -th value of the mixture and the node  $i$ .

- For any index  $i$  of the tree, the probability to move from one state to the other in characterized by a **transition probability**,

$$\forall i \in \mathcal{T} \setminus \{0\} \quad \forall g, k \in \llbracket 1, K \rrbracket^2 \quad \epsilon_i^{(gk)} = P(H_i = k | H_{\rho(i)} = g), \quad (4.7)$$

where  $\epsilon_i$  defines a transition probability matrix such that,

$$\forall i \in \mathcal{T} \setminus \{0\} \quad \forall k \in \llbracket 1, K \rrbracket \quad P(H_i = k) = \sum_{g=1}^K \epsilon_i^{(gk)} P(H_{\rho(i)} = g). \quad (4.8)$$

And using the chain rule of probability one can express  $P(H_i)$  from the root node's initial distribution.



The SCHMT model is thus fully described by,

$$\Theta = (\pi_0, \{\epsilon_i, \{\theta_{k,i}\}^{k \in \llbracket 1, K \rrbracket}\}_{i \in \mathcal{T}}). \quad (4.9)$$

The SCHMT model differs from the previous work by the shape of its tree structure. Previous work are based on regular binary tree structures where all the leafs have the same depth the scattering tree is a irregular tree. As seen in section 2.2 each node has *TODO* children. This yields to an architecture where each node have a different number of children and where leafs can be found at any depth of the tree. However as detailed in section 4.3, the learning algorithm can be adapted to this case. Another difference between SCHMT and the previous works is the non-homogeneity of the transition matrix. Indeed by the nature of the scattering transform one can expect a non homogeneous transmission of the information across the orders and thus to accomodate this the learning algorithm has also be adapted to this case.

Even though the theoretical framework of SCHMT holds for any  $K \in \mathbb{N}^*$ , in the remainder of the work  $K$  is set to 2. This means that the scattering coefficient are described by a mixture of two Gaussians and can be in *two states*, *(H) High or (L) Low*. This model yields to sparser representations as the number of hidden states is highly constrained.

## 4.2 Hypothesis :

Expressing the dependencies between the scattering coefficients as an Hidden Markov Tree implies that two modeling assumptions are done. The first assumption reflects the fact that the scattering coefficients can effectively be expressed by two hidden states,

### *Assumption 1. Two populations :*

A signal's scattering transform can be described as two clusters of coefficients' value. The smooth regions are represented by small scattering coefficients, while edges, ridges, and other singularities are represented by large coefficients.

As this assumption is common for standard or complex wavelet (*cite : Kingsbury* and because a scattering coefficient of order  $m$  can simply be seen as the modulus of a “new” signal **-i.e.** the scattering coefficient of order  $m - 1$ , the two-populations assumption for scattering network is reasonable. This intuition can be confirmed by plotting the scattering coefficients obtained for several signals.

Figure 4.4 displays the scattering coefficients of a noisy binary square. Note that for sake of clarity a “small” network has been used. This does not affect the observations that can be made and one can notice that the highest values of the scattering coefficient are obtained on highly informational pixels (edges in this case) while the low informational pixels are represented by scattering coefficients near 0. Similar observations can be made for more complex signal -such as the one displays in 4.5. A statistical interpretation of the two-populations implies that scattering coefficients have a non-Gaussian marginal statistics, that is, their marginal probability density functions have a large peak at zero (many small coefficients) and heavy tails (a few large coefficients). Finally since many real-world signals (photograph-like images, for example) consist mostly of smooth regions separated by a few singularities, the two populations assumption tells us that the scattering coefficients are a sparse representation for these signals (this notion of sparsity can be made mathematically precise; see *cite : Kingsbury - 4, 5* for example). Most of the scattering coefficient magnitudes (representing the smooth regions) are small, while a few of them (representing the singularities and encoding the informational content) are large.

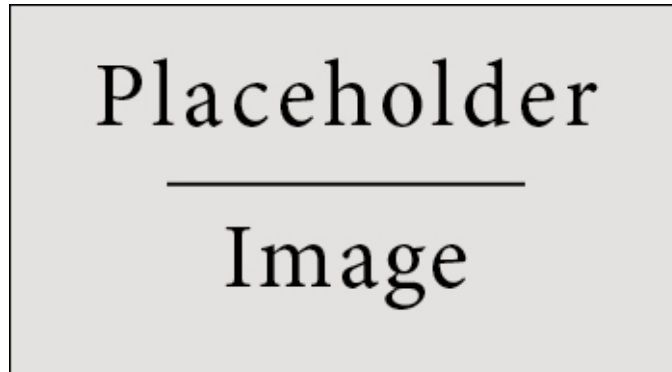


FIGURE 4.4 – Two populations - experiment 1 : The signal is a binary square (0 : background, 1 : square) with noise. The scattering network has  $m = 3$  layers,  $J = 3$  scales and  $L = 2$  orientations.



FIGURE 4.5 – Two populations - experiment 1 : The signal is the picture of Lena. The scattering network has  $m = 3$  layers,  $J = 3$  scales and  $L = 2$  orientations.



FIGURE 4.6 – Persistence - experiment 1 : Plot the magnitude of the scattering coefficients at a given index  $i$  of the tree against those of its father  $\rho(i)$ .



FIGURE 4.7 – Persistence - experiment 2 :.

The second assumptions transcripts the smoothness of the states across the scattering transform and can be stated as,

**Assumption 2. Persistence :**

Along a scattering path high and low scattering coefficient values cascade across the scattering orders.

This assumption codifies how the hidden states are structured. Smooth regions/singularities are represented by low/high values at every order. Persistence leads to scattering coefficient values that are statistically dependent along the branches of the scattering tree. Figure 4.6 displays the magnitude of the scattering coefficient for a given node  $i$  of the tree against those of its father  $\rho(i)$ . A clear positive correlation can be observed between the magnitude of the father and the son. Furthermore the transitions can be regrouped in two main clusters : Low-Low and High-High. This tends to confirm the validity of the persistence assumption. A second experiment computes the average correlation between each pair father/child of the tree for signal belonging to different classes. Again this experiments advocates in favor of a correlation in between the magnitude of the connected nodes.

### 4.3 Learning the tree structure :

As seen in 3.2.2 training Markov models can be done using *Expectation-Maximization* methods. Hidden Markov chains use a version of the EM algorithm called *Forward-Backward*

algorithm allowing the propagation of the hidden states along the chain. MS. Crouse *cite : Crouse* proposed the **Upward-Downward algorithm**, an adaptation to the hidden Markov trees of the Forward-Backward algorithm. Both algorithm were suffering from overflowing problems *cite : Durand 9* and JB. Durand *cite : Durand* adapted *name + cite : Durand 7* work to create a smoothed version of the learning algorithm for trees. This section proposes a rewriting of JB. Durand's version of the Upward-Downward algorithm adapted to non-regular non-binary HMTs.

To do so one needs to introduce the following notations,

- $\forall i \in \mathcal{T}$  let  $n_i$  be the number of children of the node  $i$ .
- $\forall i \in \mathcal{T}$  let  $\bar{\mathcal{S}}_i = \bar{s}_i$  be the observed subtree rooted at node  $i$ . By convention  $\bar{\mathcal{S}}_0$  denotes the entire observed tree.
- $\forall i \in \mathcal{T}$  let  $\bar{\mathcal{S}}_{c(i)} = \bar{s}_{c(i)}$  be the entire -possibly empty collection of observed subtrees rooted at children of node  $i$  (**i.e.** the subtree  $\bar{s}_i$  except its root  $s_i$ ).
- If  $\bar{\mathcal{S}}_i$  is a proper subtree of  $\bar{\mathcal{S}}_j$ , then  $\bar{\mathcal{S}}_{j \setminus i} = \bar{s}_{j \setminus i}$  is the subtree rooted at node  $j$  except the subtree rooted at node  $i$ .
- $\forall i \in \mathcal{T}$  let  $\bar{\mathcal{S}}_{0 \setminus c(i)} = \bar{s}_{0 \setminus c(i)}$  be the entire tree except for the subtrees rooted at children of node  $i$ .

Note that those notations transpose to the hidden state and for instance  $\bar{\mathcal{H}}_i = \bar{h}_i$  is the state subtree rooted at node  $i$ .

### 4.3.1 E-Step :

The smoothed version of the E-step requires the computation of the conditional probability distribution  $\xi_i(k) = P(H_i = k | \bar{\mathcal{S}}_i = \bar{s}_i)$  (smoothed probability) and  $P(H_i = k, H_{\rho(i)} = g | \bar{\mathcal{S}}_i = \bar{s}_i)$  for each node  $i \in mcalT$  and state  $k$  and  $g$ . A decomposition of the smoothed probability adapted to the HMT structure is,

$$\xi_i(k) = \frac{P(\bar{\mathcal{S}}_{0 \setminus i} = \bar{s}_{0 \setminus i} | H_i = k)}{P(\bar{\mathcal{S}}_{0 \setminus i} = \bar{s}_{0 \setminus i} | \bar{\mathcal{S}}_1 = \bar{s}_1)} P(H_i = k | \bar{\mathcal{S}}_i = \bar{s}_i) \quad (4.10)$$

The smoothed upward-downward algorithm requires the introduction the following quantities,

$$\beta_i(k) = P(H_i = k | \bar{\mathcal{S}}_i = \bar{s}_i) \quad (4.11)$$

$$\beta_{\rho(i)i}(k) = \frac{P(\bar{\mathcal{S}}_i = \bar{s}_i | H_{\rho(i)} = k)}{P(\bar{\mathcal{S}}_i = \bar{s}_i)} \quad (4.12)$$

$$\alpha_i(k) = \frac{P(\bar{\mathcal{S}}_{0 \setminus i} = \bar{s}_{0 \setminus i} | H_{\rho(i)} = k)}{P(\bar{\mathcal{S}}_{0 \setminus i} = \bar{s}_{0 \setminus i} | \bar{\mathcal{S}}_i = \bar{s}_i)} \quad (4.13)$$

The smoothed upward-downward algorithm also requires the preliminary knowledge of the marginal state distributions  $P(H_i = k)$  for each node  $i$ . However this can simply be achieved by a downward recursion initialized for the root node with  $P(H_0 = k) = \pi_0(k)$  and then using the recursive formula 4.8.

### Upward recursion :

The upward algorithm is initialized at all the leaf of the tree, by computing  $\beta_i(k)$  using,

$$\begin{aligned}
\beta_i(k) &= P(H_i = k | \bar{\mathcal{S}}_i = \bar{s}_i) \\
&= \frac{P(S_i = s_i | H_i = k) P(H_i = k)}{P(S_i = s_i)} \\
&= \frac{P_{\theta_{k,i}}(s_i) P(H_i = k)}{N_i},
\end{aligned} \tag{4.14}$$

where the normalization factor for the leaf nodes  $N_i$  is given by,

$$N_i = P(S_i = s_i) = \sum_{k=1}^K P_{\theta_{k,i}}(s_i) P(H_i = k). \tag{4.15}$$

Then one can recursively (upward recursion) compute  $\beta_i(k)$  for the remaining nodes of the tree using,

$$\begin{aligned}
\beta_i(k) &= P(H_i = k | \bar{\mathcal{S}}_i = \bar{s}_i) \\
&= \left[ \prod_{j \in c(i)} P(\bar{\mathcal{S}}_j = \bar{s}_j | H_i = k) \right] P(S_i = s_i | H_i = k) \frac{P(H_i = k)}{P(\bar{\mathcal{S}}_i = \bar{s}_i)} \\
&= \left[ \frac{\prod_{j \in c(i)} P(\bar{\mathcal{S}}_j = \bar{s}_j | H_i = k)}{P(\bar{\mathcal{S}}_j = \bar{s}_j)} \right] P(S_i = s_i | H_i = k) P(H_i = k) \frac{\prod_{j \in c(i)} P(\bar{\mathcal{S}}_j = \bar{s}_j)}{P(\bar{\mathcal{S}}_i = \bar{s}_i)} \\
&= \frac{\left[ \prod_{j \in c(i)} \beta_{ij}(k) \right] P_{\theta_{k,i}}(s_i) P(H_i = k)}{N_i},
\end{aligned} \tag{4.16}$$

where the normalization factor for the non-leaf nodes  $N_i$  is given by,

$$\begin{aligned}
N_i &= \frac{P(\bar{\mathcal{S}}_i)}{\prod_{j \in c(i)} P(\bar{\mathcal{S}}_j = \bar{s}_j)} \\
&= \sum_{k=1}^K \left[ \prod_{j \in c(i)} \beta_{ij}(k) \right] P_{\theta_{k,i}}(s_i) P(H_i = k).
\end{aligned} \tag{4.17}$$

For all node  $i$ , the quantities  $\beta_{\rho(i)i}(k)$  can be extracted from  $\beta_i$  using,

$$\begin{aligned}
\beta_{\rho(i)i}(k) &= \frac{P(\bar{\mathcal{S}}_i = \bar{s}_i | H_{\rho(i)} = k)}{P(\bar{\mathcal{S}}_i = \bar{s}_i)} \\
&= \frac{\sum_{g=1}^K P(\bar{\mathcal{S}}_i = \bar{s}_i | H_i = g) P(H_i = k | H_{\rho(i)} = k)}{P(\bar{\mathcal{S}}_i = \bar{s}_i)} \\
&= \sum_{g=1}^K \frac{P(H_i = g | \bar{\mathcal{S}}_i = \bar{s}_i)}{P(H_i = g)} P(H_i = g | H_{\rho(i)} = k) \\
&= \sum_{g=1}^K \frac{\beta_i(g) \epsilon_i^{(kl)}}{P(H_i = g)}.
\end{aligned} \tag{4.18}$$

Those relationships one can derive the the upward algorithm 1.

```

Meta-parameters :
 $K$ 
Initialization :
//  $P_{\theta_{k,i}}(s_i)$  :
for All the node  $i$  of the tree  $\mathcal{T}$  do
    |  $P_{\theta_{k,i}}(s_i) = \mathcal{N}(s_i | \mu_{k,i}, \sigma_{k,i})$ 
end
// Loop over the leafs  $i$  of the tree :
for All the leaf  $i$  of the tree  $\mathcal{T}$  do
    |  $\beta_i(k) = \frac{P_{\theta_{k,i}}(s_i)P(H_i=k)}{\sum_{g=1}^K P_{\theta_{g,i}}(s_i)P(H_i=g)}$ 
    |  $\beta_{i,\rho(i)}(k) = \sum_{g=1}^K \frac{\beta_i(g)\epsilon_i^{(kg)}}{P(H_i=g)} \cdot P(H_{\rho(i)} = k)$ 
    |  $l_i = 0$ 
end
Induction :
// Bottom-Up loop over the nodes of the tree :
for All non-leaf node  $i$  of the tree  $\mathcal{T}$  do
    |  $M_i = \sum_{k=1}^K P_{\theta_{k,i}}(s_i) \prod_{j \in c(i)} \frac{\beta_{j,i}(k)}{P(H_i=k)^{n_i-1}}$ 
    |  $l_i = \log(M_i) + \sum_{j \in c(i)} l_j$ 
    |  $\beta_i(k) = \frac{P_{\theta_{k,i}}(s_i) \prod_{j \in c(i)} (\beta_{j,i}(k))}{P(H_i=k)^{n_i-1} M_i}$ 
    | for All the children node  $j$  of node  $i$  do
        |  $\beta_{i \setminus c(i)}(k) = \frac{\beta_i(k)}{\beta_{i,j}(k)}$ 
    | end
    |  $\beta_{i,\rho(i)}(k) = \sum_{g=1}^K \frac{\beta_i(g)\epsilon_i^{(gk)}}{P(H_i=g)} \cdot P(H_{\rho(i)} = k)$ 
end

```

**Algorithm 1:** Smoothed upward algorithm.

### Downward recursion :

The downward recursion can either be built on the basis of the quantities  $\alpha_i(k)$  or on the basis of the smoothed probabilities  $\xi_i(k) = P(H_i = k | \bar{\mathcal{S}}_i = \bar{s}_i)$ . The downward recursion on  $\xi_i$  is initialized at the root node with,

$$\xi_0(k) = P(H_i = k | \bar{\mathcal{S}}_0 = \bar{s}_0) = \beta_0(k). \quad (4.19)$$

The quantity can then be computed recursively for each node of the tree using,

$$\begin{aligned} \xi_i(k) &= P(H_i = k | \bar{\mathcal{S}}_0 = \bar{s}_0) \\ &= \sum_{g=1}^K \frac{P(H_i = k, H_{\rho(i)} = g, \bar{\mathcal{S}}_0 = \bar{s}_0)}{P(H_{\rho(i)} = g, \bar{\mathcal{S}}_0 = \bar{s}_0)} P(H_{\rho(i)} = g | \bar{\mathcal{S}}_0 = \bar{s}_0) \\ &= P(\bar{\mathcal{S}}_i = \bar{s}_i | H_i = k) \sum_{g=1}^K \frac{P(H_i = k | H_{\rho(i)} = g)}{P(\bar{\mathcal{S}}_i = \bar{s}_i | H_{\rho(i)} = g)} P(H_{\rho(i)} = g | \bar{\mathcal{S}}_0 = \bar{s}_0) \\ &= \frac{\beta_i(k)}{P(H_i = k)} \sum_{g=1}^K \frac{\epsilon_i^{(gk)} \xi_{\rho(i)}(g)}{\beta_{\rho(i),i}(g)}. \end{aligned} \quad (4.20)$$

Using the fact that for all  $i \in mcalT$   $\xi_i(k) = \beta_i(k) \alpha_i(k)$  and the relationship from 4.20, one can express the downward pass as presented in 2.

#### Meta-parameters :

$K$

#### Initialization :

$$\alpha_0(k) = 1$$

#### Induction :

// Top-Down loop over the nodes of the tree :

for All node  $i$  of the tree  $\mathcal{T} \setminus \{0\}$  do

$$\quad \alpha_i(k) = \frac{1}{P(H_i = k)} \sum_{g=1}^K \alpha_{\rho(i)}(g) \epsilon_i^{(gk)} \beta_{\rho(i),i}(g) P(H_{\rho(i)} = g)$$

end

**Algorithm 2:** Smoothed downward algorithm.

### Conditional properties :

To complete the E-step one needs to compute the conditional probabilities for each node. This is done simply by using,

$$\forall i \in \mathcal{T} \quad P(H_i = k | \bar{\mathcal{S}}_0 = \bar{s}_0) = \alpha_i(k) \beta_i(k), \quad (4.21)$$

and

$$\forall i \in \mathcal{T} \setminus \{0\} \quad P(H_{\rho(i)} = g, H_i = k | \bar{\mathcal{S}}_0 = \bar{s}_0) = \frac{\beta_i(k) \epsilon_i^{(gk)} \alpha_{\rho(i)}(g) \beta_{\rho(i)}(g)}{P(H_i = k)}. \quad (4.22)$$

### 4.3.2 M-Step :

The *Maximization* step of the EM algorithm aims to maximize the log-likelihood of the observation with regards to the parameters and then use those pseudo-optimal parameters for the next *Expectation* step. Meaning that at the iteration  $m$  of the EM process, the M-step does,

$$\Theta^{m+1} = \underset{\Theta}{\operatorname{argmax}} \left( E[\ln f(x, H|\Theta)|x, \Theta^l] \right). \quad (4.23)$$

The  $\Theta$  maximizing the log-likelihood in 4.23 can be express analytically and this yields to the algorithm 3

**Meta-parameters :**

$K$ ,

Distribution family for  $P_\theta$  ;

// Here Gaussian

$N$  ;

// Number of observed realizations of the signal

**Initialization :**

$$\pi_0(k) = \frac{1}{N} \sum_{n=1}^N P(H_0^n = k | s_0^n, \Theta^l)$$

**Induction :**

// Loop over the nodes of the tree :

**for** All node  $i$  of the tree  $\mathcal{T} \setminus \{0\}$  **do**

$$P(H_i = k) = \frac{1}{N} \sum_{n=1}^N P(H_i^n = k | s_i^n, \Theta^l),$$

$$\epsilon_i^{gk} = \frac{\sum_{n=1}^N P(H_i^n = k, H_{\rho(i)}^n = g | w^n, \Theta^l)}{NP(H_{\rho(i)} = k)},$$

$$\mu_{k,i} = \frac{\sum_{n=1}^N s_i^n P(H_i^n = k | w^n, \Theta^l)}{NP(H_i = k)},$$

$$\sigma_{k,i}^2 = \frac{\sum_{n=1}^N (s_i^n - \mu_{k,i})^2 P(H_i^n = k | w^n, \Theta^l)}{NP(H_i = k)}.$$

**end**

**Algorithm 3:** M-step of the EM algorithm.

### 4.3.3 EM algorithm :

Finally the EM algorithm iterates over the *E-step* and the *M-step* as describe by the algorithm 4.

## 4.4 Classification :

Let  $\Theta_c$  now be a set of parameters for an SCHMT  $\mathcal{T}$  learned using the algorithm described in section 4.3 and a training set  $X_c$  composed of  $N$  realization of a signal of class  $c$ . Let also  $x_{new}$  be another realization of this signal, not used for training and  $\mathcal{T}^{(new)}$  be the instance of the SCHMT generated by this realization.

In this context the **MAP algorithm** aims at finding the optimal hidden tree  $\hat{h}_0^{new} = (\hat{h}_0^{new} \dots \hat{h}_{I-1}^{new}, \Theta_c)$  maximizing the probability of this sequence given the model's parameters  $P(\mathcal{H}_0 = \hat{h}_0^{new} | \mathcal{T}^{(new)})$ . The MAP framework also provides  $\hat{P}$  the value of this maximum.

For SCHMT the MAP algorithm has the form describe by the algorithm 5.



```

Meta-parameters :
 $K$  ;
Distribution family for  $P_\theta$  ; Convergence criteria ; // Iteration limit or information based
Initialization method for  $\Theta$  ; // Random or prior knowledge
Initialization :
 $l = 0$  ; // Iteration counter
Initialize( $\Theta$ )
Iteration :
while Not convergence do
    E-step : Calculate  $P(\bar{\mathcal{H}}|\bar{\mathcal{H}}, \Theta^l)$ .
    M-step : Set  $\Theta^{m+1} = \operatorname{argmax}_\Theta (E[\ln f(x, H|\Theta)|x, \Theta^l])$ .
     $l = l+1$ 
end

```

**Algorithm 4:** EM algorithm.

```

Meta-parameters :
 $K$  ;
Initialization :
for all leaf  $i$  of  $\mathcal{T}$  do
     $\gamma_i(k) = \beta_i(k)$  ; // The gamma for all  $k$  must be computed before the next step
     $\gamma_{i,\rho i}(k) = \max_{1 \leq g \leq K} \gamma_i(g) \epsilon_i^{kg}$ 
     $\xi_i(k) = \operatorname{argmax}_{1 \leq g \leq K} \gamma_i(g) \epsilon_i^{kg}$ 
end
Induction :
// Top-Down loop over the nodes of the tree :
for All node  $i$  of the tree  $\mathcal{T} \setminus \{0\}$  do
     $\gamma_i(k) = P_{\theta_{k,i}}(s_i) \prod_{j \in c(i)} \gamma_{j,i}(k)$ 
     $\gamma_{i,\rho i}(k) = \max_{1 \leq g \leq K} \gamma_i(g) \epsilon_i^{kg}$  ; // except at root node
     $\xi_i(k) = \operatorname{argmax}_{1 \leq g \leq K} \gamma_i(g) \epsilon_i^{kg}$ 
end
Termination :
 $\hat{P} = \max_{1 \leq g \leq K} \gamma_0(g)$ 
 $\hat{h}_0 = \operatorname{argmax}_{1 \leq g \leq K} \gamma_0(g)$ 
Downward tracking :
// Creation of the hidden tree from the root node
for All node  $i$  of the tree  $\mathcal{T} \setminus \{0\}$  do
     $\hat{h}_i = \xi_i(\hat{h}_{\rho(i)})$ 
end

```

**Algorithm 5:** MAP algorithm.

The MAP algorithm 5 can be used in a multi-class classification problem simply by training a SCHMT model per class and then for a new realization  $x_{new}$  simply compare the MAP provided by each model as describe by the algorithm 6.

**Meta-parameters :**

$K ; C ;$

// Number of class

**for** *All the class c* **do**

$\hat{P}_c = \text{MAP}(x_{new}, \Theta_c, K)$

**end**

$\hat{P} = \max_{0 \leq c < C} \hat{P}_c$

$l = \text{argmax}_{0 \leq c < C} \hat{P}_c$

**Algorithm 6:** MAP algorithm applied to multiclass classification problem.

## 5 Experimental results :

This section presents some experimental results obtained using scattering hidden Markov trees for classification tasks. First a binary classification task is performed on simple simulated shapes. Then SCHMT are applied to classify seabed and ripples in sonar imagery.

### 5.1 Shapes :

The SCHMT performances on classification tasks are at first assessed on telling appart different geometrical shapes. The data are noisy and translated realizations of two signals. The first signal is an 100 by 100 binary image picturing a centered black square on a white background. Examples of this class can be seen in figure 5.1. The second class is a 100 by 100 binary image picturing a centered black triangle on a white background. Examples of this class can be seen in figure 5.2.

### 5.2 Sonar Imagery :

The SCHMT is now tested on a more complex set of images. The data used are extracted from the *UDRC MCM sonar dataset* [https://www.gov.uk/government/uploads/system/uploads/attachment\\_data/file/326061/DefenceReporter\\_Winter2009.pdf](https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/326061/DefenceReporter_Winter2009.pdf). This dataset, designed initially for underwater mines detection, contains targets comparable to those faced by coalition forces in current operational engagements and comprises of Synthetic Aperture RADAR (SAS imagery) and Hyperspectral data. From those very high dimensional images, 100 by 100 patches have been extracted and labeled as either *seabed* or *ripple* (see respectively figure 5.3 and figure 5.4).

The task at hand is the classification of those realizations of those two signals. The scattering transform used has  $m = 3$  orders,  $J = 5$  scales,  $M = 3$  orientations and uses a Morlet



FIGURE 5.1 – (a) Original signal (b) realization

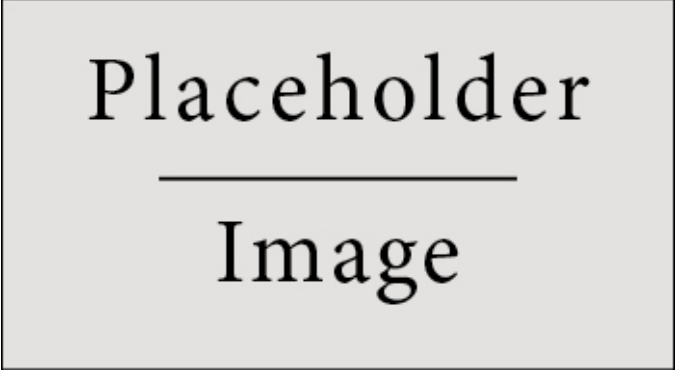


FIGURE 5.2 – (a) Original signal (b) realization

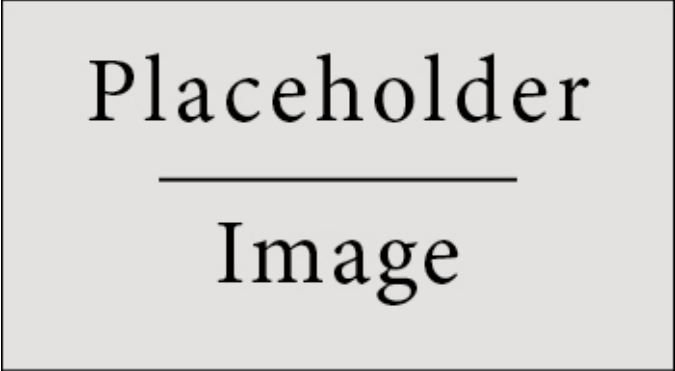


FIGURE 5.3 – Seabed patchs

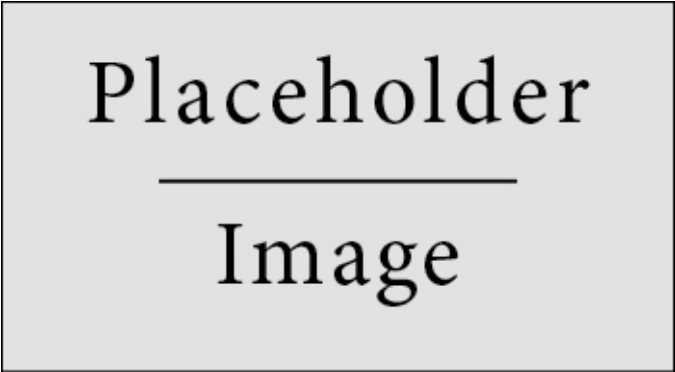


FIGURE 5.4 – Ripple patchs

Classification results				
Classification score :	Mean	Variance	Maximum	Minimum
With outliers :	0	0	0	0
Without outliers :	0	0	0	0

TABLE 5.1 – Classification performances over 100 experiments of Ripple/Seabed classification.

wavelet. The hidden Markov tree has  $K = 2$  states and is using a mixture of Gaussian to describe the relationship between the scattering coefficients and the hidden states. Two models -one for each class-  $\Theta_{ripple}$  and  $\Theta_{seabed}$  are trained on 200 realizations of their class signal. The testing is then realized on 80 images -40 of each classes. The performances of the SCHMT are assessed on 100 instances of this experiment and the results are displayed in table ??.

The results with best scores at 90% accuracy shows encouraging results as per the classification performances of the SCHMTs. However one can notice that some experiment provides scores as low as 50% accuracy. Those scores are obtained when one model has shown a better convergence than the other, resulting to classifying all the instances of the test set to one class. This problem could be addressed by using a validation set to discard those models before testing. Another leads would be to work on the EM algorithm to make it more robust.

## 6 Conclusion :

**Scattering hidden Markov tree :** The SCHMT appears to be a powerful model combining the interesting properties of the scattering transform for signal representation and the representational power of the hidden Markov models. It provides with analytically tractable interesting representation of high-dimensional signal as well as a probabilistic classification method. Experimental results confirm this intuition but also reveal some weakness of our model.

**Next steps :** Despite showing good performances when the learning phase has converged properly, SCHMT's performances are undermined by a sometime poor learning quality. This observation is one of the main drivers for the upcoming work. The other main driver for the next step is to keep a well define probabilist framework.

At this point the version of the EM algorithm considered in this report is using full Bayesian inference. However this methods is computationally expensive and sometimes yields to poor learning. Furthermore this version of the EM algorithm provides only a point-wise estimate of the model's parameters. It would be interesting to apply *variational methods* the this problem *cite : Graphical Models, Exponential Families, and Variational Inference*. Beside a potential improvement of the performance variational inference would also provide a estimated distribution for the model's parameter, thus allowing us to have access to a *measure of uncertainty* for our learning.

Another interesting direction to follow is the works on *hierarchical graphical models cite : The Hierarchical Hidden Markov Model : Analysis and Applications*. Those models would allow us to use the SCHMT model as a node of a wider probabilistic graphical model. Using such architecture, one could model a network of sensors each providing information on a targeted scene.

Finally another interesting lead would be to consider other architecture for the graphical model and make it includes the representation learning step. This would be possible using *Bayesian neural networks cite : Bayes backprop and PBP*.

## 7 Acknowledgements :

Jean-Baptiste Regli is funded by a Dstl/UCL Impact studentship and James Nelson is partially supported by grants from the Dstl and Innovate UK/EPSRC.Fpap

# Bibliographie

- [Abdel-Hamid et al., 2012] Abdel-Hamid, O., Mohamed, A.-r., Jiang, H., and Penn, G. (2012). Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*, pages 4277–4280. IEEE.
- [Andén and Mallat, 2011] Andén, J. and Mallat, S. (2011). Multiscale scattering for audio classification. In *ISMIR*, pages 657–662.
- [Baker et al., 2014] Baker, M. J., Trevisan, J., Bassan, P., Bhargava, R., Butler, H. J., Dorling, K. M., Fielden, P. R., Fogarty, S. W., Fullwood, N. J., Heys, K. A., et al. (2014). Using fourier transform ir spectroscopy to analyze biological materials. *Nature protocols*, 9(8) :1771–1791.
- [Beyer et al., 1999] Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U. (1999). When is “nearest neighbor” meaningful? In *Database Theory—ICDT’99*, pages 217–235. Springer.
- [Bracewell, 1965] Bracewell, R. (1965). The fourier transform and its applications. *New York*.
- [Brailean and Katsaggelos, 1996] Brailean, J. C. and Katsaggelos, A. K. (1996). Recursive map displacement field estimation and its applications. In *Image Processing, 1996. Proceedings., International Conference on*, volume 1, pages 917–920. IEEE.
- [Bruna, 2012] Bruna, J. (2012). Operators commuting with diffeomorphisms.
- [Bruna and Mallat, 2010] Bruna, J. and Mallat, S. (2010). Classification with scattering operators. *arXiv preprint arXiv :1011.3023*.
- [Bruna and Mallat, 2013] Bruna, J. and Mallat, S. (2013). Invariant scattering convolution networks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8) :1872–1886.
- [Cover and Hart, 1967] Cover, T. M. and Hart, P. E. (1967). Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1) :21–27.
- [Crouse et al., 1998] Crouse, M. S., Nowak, R. D., and Baraniuk, R. G. (1998). Wavelet-based statistical signal processing using hidden markov models. *Signal Processing, IEEE Transactions on*, 46(4) :886–902.
- [De Chazal et al., 2000] De Chazal, P., Celler, B., and Reilly, R. (2000). Using wavelet coefficients for the classification of the electrocardiogram. In *Engineering in Medicine and Biology Society, 2000. Proceedings of the 22nd Annual International Conference of the IEEE*, volume 1, pages 64–67. IEEE.
- [Durand et al., 2004] Durand, J.-B., Goncalves, P., and Guédon, Y. (2004). Computational methods for hidden markov tree models-an application to wavelet trees. *Signal Processing, IEEE Transactions on*, 52(9) :2551–2560.
- [Hinton et al., 2012] Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. (2012). Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv :1207.0580*.
- [Hörmander, 1971] Hörmander, L. (1971). Fourier integral operators. i. *Acta mathematica*, 127(1) :79–183.



- [Kingsbury, 2001] Kingsbury, N. (2001). Complex wavelets for shift invariant analysis and filtering of signals. *Applied and computational harmonic analysis*, 10(3) :234–253.
- [Kreutz-Delgado et al., 2003] Kreutz-Delgado, K., Murray, J. F., Rao, B. D., Engan, K., Lee, T.-W., and Sejnowski, T. J. (2003). Dictionary learning algorithms for sparse representation. *Neural computation*, 15(2) :349–396.
- [Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- [LeCun and Bengio, 1995] LeCun, Y. and Bengio, Y. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10).
- [LeCun et al., 2010] LeCun, Y., Kavukcuoglu, K., and Farabet, C. (2010). Convolutional networks and applications in vision. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 253–256. IEEE.
- [Lin and Zha, 2008] Lin, T. and Zha, H. (2008). Riemannian manifold learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(5) :796–809.
- [Lohmiller and Slotine, 1998] Lohmiller, W. and Slotine, J.-J. E. (1998). On contraction analysis for non-linear systems. *Automatica*, 34(6) :683–696.
- [Mallat, 1999] Mallat, S. (1999). *A wavelet tour of signal processing*. Academic press.
- [Mallat, 2012] Mallat, S. (Oct. 2012). Group invariant scattering. *Communications in Pure and Applied Mathematics*, 65(10) :1331–1398.
- [Platt et al., 1999] Platt, J. et al. (1999). Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in large margin classifiers*, 10(3) :61–74.
- [Sifre and Mallat, 2013] Sifre, L. and Mallat, S. (2013). Rotation, scaling and deformation invariant scattering for texture discrimination.
- [Simard et al., 2003] Simard, P. Y., Steinkraus, D., and Platt, J. C. (2003). Best practices for convolutional neural networks applied to visual document analysis. In *null*, page 958. IEEE.
- [Szegedy et al., 2013] Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *arXiv preprint arXiv :1312.6199*.
- [Waldspurger et al., 2015] Waldspurger, I., d’Aspremont, A., and Mallat, S. (2015). Phase recovery, maxcut and complex semidefinite programming. *Mathematical Programming*, 149(1-2) :47–81.
- [Zhang et al., 2012] Zhang, Z., Wang, J., and Zha, H. (2012). Adaptive manifold learning. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(2) :253–265.

