FIXED-LAG BLIND EQUALIZATION AND SEQUENCE ESTIMATION IN DIGITAL COMMUNICATIONS SYSTEMS USING SEQUENTIAL IMPORTANCE SAMPLING

Tim C. Clapp and Simon J. Godsill

Department of Engineering, Cambridge University, UK
tcc1000@cam.ac.uk, sjg@eng.cam.ac.uk

ABSTRACT

We present methods for fixed-lag smoothing using Sequential Importance sampling (SIS) on a discrete non-linear, non-Gaussian state space system with unknown parameters. Our particular application is in the field of digital communication systems. Each input data point is taken from a finite set of symbols. We represent the transmission media as a fixed filter with a finite impulse response (FIR), hence a discrete state-space system is formed. Conventional Markov chain Monte Carlo (MCMC) techniques such as the Gibbs sampler are unsuitable for this task because they can only perform processing on a batch of data. Data arrives sequentially, so it would seem sensible to process it in this way. In addition, many communication systems are interactive, so there is a maximum level of latency that can be tolerated before a symbol is decoded. We will demonstrate this method by simulation and compare its performance to existing techniques.

1. INTRODUCTION

We are interested in obtaining estimates of the states $\mathbf{x}_{t-p}$ at time t, given observations $\mathbf{y}_{0:t}$, that is there is a fixed lag of length p after which we will output our estimates of $\mathbf{x}$. For example, in many communications systems, even interactive ones, a small amount of delay or latency can be tolerated. By allowing a delay of around 4–5 times the channel length, significant gains in performance can be obtained [8].

1.1. Model Formulation and assumptions

We shall assume a state-space model, where the current state, $\mathbf{x}_t$, is dependent only upon the previous state and the current input, $\mathbf{b}_t$. The observation, $\mathbf{y}_t$ is dependent on the current state, some parameters, $\boldsymbol{\theta}$, and a noise term, $\mathbf{v}_t$:

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}) + \mathbf{b}_t$$
$$\mathbf{y}_t = g(\mathbf{x}_t, \boldsymbol{\theta}) + \mathbf{v}_t$$

We assume that:

- We can evaluate $p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})$.
- We can sample from $p(\boldsymbol{\theta} \mid \mathbf{x}, \mathbf{y})$ by some means.
- $\mathbf{x}$ is discrete, or if $\mathbf{x}$ is continuous we require that $f(.)$ is linear and that $p(\mathbf{y} \mid \mathbf{x}, \boldsymbol{\theta})$ is Gaussian.

This work is supported by the Engineering and Physical Sciences Research Council.

For our particular application, namely a digital communications system, $\mathbf{x}$ is discrete. For the rest of this paper we shall describe the methods assuming a discrete state and continuous unknown parameters, however they can be easily adapted to the linear Gaussian continuous state case by using techniques described in [2].

1.2. The Particle filter

Our method is based upon the particle filter ideas seen in [6], [11], [1] and [12]. We are interested in the distribution of all the states up to time $t-p$, given the received data up to the current time t, that is $p(\mathbf{x}_{0:t-p} \mid \mathbf{y}_{0:t})$, the fixed-lag smoothing density where a collection of states is denoted by $\mathbf{x}_{0:t} = \{\mathbf{x}_0, \ldots, \mathbf{x}_t\}$. This distribution is represented by a finite number of state trajectories, $\mathbf{x}^{(i)}_{0:t-p}$, or 'particles', each with an associated weight, $w^{(i)}$. With the arrival of a new data point, the estimate of the new probability distribution is made by propagating the current distribution — a draw of the new state and calculation of the new weight (for each trajectory) — using a Bayesian update rule.

1.3. Organisation

The remainder of this paper is organised as follows: The basic technique of Sequential Importance Sampling is described in section 2. Section 3 describes two methods for achieving fixed-lag filtering using the filtering density, $p(\mathbf{x}_t \mid \mathbf{y}_{0:t})$. start or the end of the time lag. Section 4 describes a method for fixed-lag filtering by updating the smoothing density, $p(\mathbf{x}_{t-p} \mid \mathbf{y}_{0:t})$. A comparison of the algorithms in the form of simulations on a digital communications system is presented in section 5.

2. SEQUENTIAL IMPORTANCE SAMPLING (SIS)

Suppose we have N samples $\mathbf{x}^{(i)}_{0:t}$ from the probability distribution of interest at time t, $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{0:t})$. With the arrival of a new data point $\mathbf{y}_{t+1}$, we would like to update this distribution to $p(\mathbf{x}_{0:t+1} \mid \mathbf{y}_{0:t+1})$ without modifying the simulated past trajectories, $\mathbf{x}^{(i)}_{0:t}$. Normally we can calculate the likelihood $p(\mathbf{y}_t \mid \mathbf{x}_t)$ and we know the state transition density $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t)$. We may do this by using importance sampling exploiting Bayesian updating as follows (without explicitly stating dependence on parameters, $\boldsymbol{\theta}$):

$$p(\mathbf{x}_{0:t+1} \mid \mathbf{y}_{0:t+1}) = \frac{p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1} \mid \mathbf{x}_t)}{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t})} \times p(\mathbf{x}_{0:t} \mid \mathbf{y}_{0:t}) \quad (1)$$

Typically one cannot calculate the normalising term $p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t})$, however it is not required since it is independent of the state trajectory. The new state is chosen by sampling

from an importance function. If we choose it to be of the form:

$$\pi(\mathbf{x}_{0:t+1} \mid \mathbf{y}_{0:t+1}) = \pi(\mathbf{x}_{t+1} \mid \mathbf{x}_{0:t}, \mathbf{y}_{0:t+1})\, \pi(\mathbf{x}_{0:t} \mid \mathbf{y}_{0:t})$$
$$= \pi(\mathbf{x}_0 \mid \mathbf{y}_0) \prod_{k=1}^{t+1} \pi(\mathbf{x}_{k+1} \mid \mathbf{x}_{0:k}, \mathbf{y}_{0:k+1})$$

we may evaluate the importance weights recursively. The unnormalised incremental importance weights are given by:

$$w_{t+1}^{(i)} = \frac{p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1} \mid \mathbf{x}_t)}{\pi(\mathbf{x}_{0:t+1} \mid \mathbf{y}_{0:t+1})}$$

We may choose a suitable importance function, exploiting any factorisation of the joint prediction density $p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1} \mid \mathbf{x}_t)$, so as to simplify this expression.

Statistics of interest may then be obtained as follows:

$$E\{f(\mathbf{x})\} = \sum_{i=1}^{N} f(\mathbf{x}^{(i)}) \frac{w^{(i)}}{\sum_{j=1}^{N} w^{(j)}} \tag{2}$$

If the variance of the weights becomes too large, we may resample the trajectories with the probability of each trajectory being selected equal to the corresponding weight. A useful measure of whether to resample is the effective sample size [11], which tells us how many particles are actually contributing significantly to the distribution.

3. FIXED-LAG SIMULATION USING THE FILTERING DENSITY

We may adapt the standard SIS technique to achieve fixed-lag smoothing, whilst updating the filtering density. There are two ways to achieve this:

1. The simplest way to achieve this is to update the filtering density as each data point arrives, i.e. track $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{0:t})$, and then output the decision after a lag of p symbols. For this updating we may use any of the standard importance distributions ranging from the prior distribution, $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t)$, introduced by [7, 6] to the optimal importance function, $p(\mathbf{x}_{t+1} \mid \mathbf{y}_{t+1}, \mathbf{x}_t, \boldsymbol{\theta})$, introduced by [14, 9, 11]. For a review of these methods see [4].
2. We may use the filtering density $p(\mathbf{x}_{0:t-p} \mid \mathbf{y}_{0:t-p})$ and attempt to make more accurate draws of the new state by drawing the new state from the importance distribution $\pi(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{t-p+1}, \mathbf{x}_{t-p}, \boldsymbol{\theta}) = p(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{0:t+1})$.

3.1. Current time filtering density

The optimal method for filtering the data without lookahead is the method of *sequential imputations* as described in [9, 11]. In this case the parameters are treated as 'nuisance parameters' and analytically marginalised. At each step only a draw of the state from $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{y}_{t+1})$ is required. The incremental importance weight is given by $p(\mathbf{y}_{t+1} \mid \mathbf{x}_t)$, and updates of the mean and variance of the marginal distribution of the parameters, $\boldsymbol{\theta}$, are obtained by recursive relationships derived from the matrix inversion lemma.

If $\boldsymbol{\theta}$ cannot be marginalised we may proceed in the following manner. The joint prediction density may be factorised:

$$p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1}, \boldsymbol{\theta} | \mathbf{x}_{0:t}, \mathbf{y}_{0:t}) = p(\mathbf{x}_{t+1} | \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1}) \times$$
$$p(\mathbf{y}_{t+1} | \mathbf{x}_t, \boldsymbol{\theta})\, p(\boldsymbol{\theta} | \mathbf{x}_{0:t}, \mathbf{y}_{0:t}) \tag{3}$$

3.1.1. Method

Let us assume we have available at time t, N samples $\mathbf{x}_t^{(j)}$ from the distribution $p(\mathbf{x}_t \mid \mathbf{y}_{0:t})$, possibly with associated weights $w^{(j)}$. We may update to the distribution $p(\mathbf{x}_{t+1} \mid \mathbf{y}_{0:t+1})$ as follows:

1. For each j:
   (a) Draw $\boldsymbol{\theta}^{(j)}$ from $p(\boldsymbol{\theta} | \mathbf{x}_{0:t}^{(j)}, \mathbf{y}_{0:t})$.
   (b) Draw $\mathbf{x}_{t+1}^{(j)}$ from $p(\mathbf{x}_{t+1} | \boldsymbol{\theta}^{(j)}, \mathbf{x}_t^{(j)}, \mathbf{y}_{t+1})$.
   (c) Calculate incremental importance weights from: $w_{t+1}^{(j)} = p(\mathbf{y}_{t+1} | \boldsymbol{\theta}^{(j)}, \mathbf{x}_t^{(j)})$.
2. If the effective sample size is too small, resample the trajectories with probability proportional to the importance weights and reset the weights to equal values.

At time t, we calculate the marginal posterior state probabilities at time $t - p$ as given by equation 2. We may now decide on the state by choosing the one with marginal maximum posterior probability (a marginal MAP estimate). This estimate is for output only — this does not affect the state trajectories or their weights in further iterations of the algorithm.

3.2. Lagged time filtering density

To simplify the expressions, we will shift our time-base forward by p samples ($t \to t + p$). Hence at time t we will make estimates of $\mathbf{x}_t$ using information from $\mathbf{y}_{0:t+p}$.

The joint prediction density may be factorised as follows:

$$p(\mathbf{y}_{t+1}, \mathbf{x}_{t+1}, \boldsymbol{\theta} \mid \mathbf{x}_{0:t}, \mathbf{y}_{0:t}) = p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1}) \times$$
$$p(\mathbf{y}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \mathbf{x}_{0:t}, \mathbf{y}_{0:t}) \tag{4}$$

By using this factorisation, we may make new draws for the parameters at each time without having to draw the states from the predictive density, $p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \mathbf{y}_t)$, as suggested by [10]. It is possible to sample directly from the first term; however we have chosen instead to use the importance function:

$$\pi(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1}) = p(\mathbf{x}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta}, \mathbf{y}_{t+1:t+p+1}) \tag{5}$$

with the hope of using the additional data to bias the draw towards states that will be kept at later iterations. This may be accomplished by using techniques described by [2], [5] and [3], discarding the unwanted imputed values $\mathbf{x}_{t+2:t+p+1}$. This introduces the importance weight:

$$w_{t+1} = \frac{p(\mathbf{y}_{t+1} \mid \mathbf{x}_{t+1}, \boldsymbol{\theta})\, p(\mathbf{y}_{t+1:t+p+1} \mid \mathbf{x}_t, \boldsymbol{\theta})}{p(\mathbf{y}_{t+1} \mid \mathbf{x}_t, \boldsymbol{\theta})\, p(\mathbf{y}_{t+1:t+p+1} \mid \mathbf{x}_{t+1}, \boldsymbol{\theta})} \tag{6}$$

The first term of both numerator and denominator are easy to calculate. The second terms may be simply calculated from the terms evaluated in a forward pass of the sampling algorithm through the data [3], which has already been made for the numerator when sampling from equation 5.

Note that in this case the draw for the parameters does not include any information from the extra received data, $y_{t+1:t+p+1}$. However, in the case of fixed parameters, the prior rapidly becomes quite strong so any additional information in these bits does not affect the posterior for $\boldsymbol{\theta}$ very much.

3.2.1. Method

Let us assume we have available at time t, N samples $\mathbf{x}_t^{(j)}$ from the distribution $p(\mathbf{x}_t \mid \mathbf{y}_{0:t})$,[1] possibly with associated weights $w^{(j)}$. At time $t+1$ we have available the additional observation $\mathbf{y}_{t+p+1}$. We may update these samples to the distribution $p(\mathbf{x}_{t+1} \mid \mathbf{y}_{0:t+1})$ as follows:

1. For each j:
   (a) Draw $\boldsymbol{\theta}^{(j)}$ from $p(\boldsymbol{\theta} \mid \mathbf{x}_{0:t}^{(j)}, \mathbf{y}_{0:t})$.
   (b) Draw $\mathbf{x}_{t+1}^{(j)}$ from equation 5.
   (c) Calculate importance weights from equation 6.
2. If the effective sample size is too low, resample as before.

4. FIXED-LAG SIMULATION USING THE SMOOTHING DENSITY

Rather than tracking the filtering density, $p(\mathbf{x}_{0:t} \mid \mathbf{y}_{0:t})$, we may track the smoothing density, $p(\mathbf{x}_{0:t-p} \mid \mathbf{y}_{0:t})$, to achieve fixed-lag smoothing. Ideologically this is a better approach, although it introduces some practical problems.

The fixed lag smoothing distribution is given by:

$$p(\mathbf{x}_{0:t-p+1} \mid \mathbf{y}_{0:t+1}) = p(\mathbf{x}_{0:t-p} \mid \mathbf{y}_{0:t}) \times \frac{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t}, \mathbf{x}_{t-p+1})\, p(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{t-p+1:t}, \mathbf{x}_{0:t-p})}{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t})} \tag{7}$$

We define the importance distribution in the usual manner and the unnormalised incremental importance weight is given by:

$$w_{t+1} = \frac{p(\mathbf{y}_{t+1} \mid \mathbf{y}_{t-p:t}, \mathbf{x}_{0:t-p+1})\, p(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{t-p+1:t}, \mathbf{x}_{0:t-p})}{\pi(\mathbf{x}_{t-p+1} \mid \mathbf{x}_{0:t-p}, \mathbf{y}_{0:t+1})\, p(\mathbf{y}_{t+1} \mid \mathbf{y}_{0:t})} \tag{8}$$

For a general distribution, there are two problems:

1. The term $p(\mathbf{y}_{t+1} \mid \mathbf{y}_{t-p:t}, \mathbf{x}_{0:t-p+1})$ cannot be evaluated.
2. The distribution $\pi(\mathbf{x}_{t-p+1} \mid \mathbf{x}_{0:t-p}, \mathbf{y}_{0:t+1}) = p(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{t-p+1:t}, \mathbf{x}_{0:t-p})$ is difficult to sample from.

In some models we may assume that for a sufficiently large value of p the term $p(\mathbf{y}_{t+1} \mid \mathbf{y}_{t-p:t}, \mathbf{x}_{0:t-p+1})$ will be approximately constant.[2]

To address the second problem we consider the case where the state $\mathbf{x}$ is discrete and dependent on some (continuous) parameters $\boldsymbol{\theta}$. We now consider generating samples from the joint distribution $p(\mathbf{x}, \boldsymbol{\theta} \mid \mathbf{y})$ by using the following factorisation:

$$p(\mathbf{x}_{t-p+1}, \boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}) = p(\mathbf{x}_{t-p+1} \mid \mathbf{x}_{0:t-p}, \mathbf{y}_{t-p+1:t}, \boldsymbol{\theta})\, p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}) \tag{9}$$

Note that we can easily sample from the distribution:

$$p(\mathbf{x}_{t-p+1}, \widetilde{\mathbf{x}}_{t-p+2:t} \mid \mathbf{x}_{0:t-p}, \mathbf{y}_{t-p+1:t}, \boldsymbol{\theta}) = p(\mathbf{x}_{t-p+1:t} \mid \boldsymbol{\theta}, \mathbf{y}_{t-p+1:t}, \mathbf{x}_{t-p}) \tag{10}$$

as before. The imputed values, $\widetilde{\mathbf{x}}_{t-p+2:t}$, are retained for use in the next iteration. We cannot easily sample from $p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p})$; however we may sample from the importance distribution:

$$\pi(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}) = p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}, \widetilde{\mathbf{x}}_{t-p+1:t}) \tag{11}$$

This introduces the (incremental) importance weight:

$$w_{t+1} = \frac{\sum_{\widetilde{\mathbf{x}}_{t-p+1:t}} p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}, \widetilde{\mathbf{x}}_{t-p+1:t})}{p(\boldsymbol{\theta} \mid \mathbf{y}_{0:t}, \mathbf{x}_{0:t-p}, \widetilde{\mathbf{x}}_{t-p+1:t})} \tag{12}$$

At first glance it may appear that the numerator of equation 12 is expensive to calculate, however all the terms will be calculated when sampling from equation 10: only a small number of additional calculations are required.

4.1. Method

Let us assume we have available at time t, N samples $\mathbf{x}_{t-p}^{(j)}$ from the distribution $p(\mathbf{x}_{t-p} \mid \mathbf{y}_{0:t})$, possibly with associated weights $w^{(j)}$. At time $t+1$ we have available the additional observation $\mathbf{y}_{t+1}$. We may update these samples to the distribution $p(\mathbf{x}_{t-p+1} \mid \mathbf{y}_{0:t+1})$ as follows:

1. For each j:
   (a) Draw $\boldsymbol{\theta}^{(j)}$ from equation 11.
   (b) Draw $\mathbf{x}_{t-p+1}^{(j)}$, $\widetilde{\mathbf{x}}_{t-p+2:t}^{(j)}$ from equation 10 and draw an additional $\widetilde{\mathbf{x}}_{t+1}^{(j)}$ from $p(\mathbf{x}_{t+1} | \mathbf{y}_{t+1}, \widetilde{\mathbf{x}}_t^{(j)})$.[3]
   (c) Calculate importance weights from equation 12.
2. If necessary, resample as before.

5. SIMULATIONS

5.1. Problem Formulation

Most digital communications systems transmit a signal $\{b_t\}$ where the value of each of the b_t are taken from a finite alphabet of q symbols. This is transmitted over a channel which may introduce distortion and noise. The channel model used here is an FIR filter with additive Gaussian noise:

$$y_t = \sum_{i=0}^{n-1} b_{t-i} h_i + v_t \tag{13}$$

$$b_t \in \{S_0, \ldots, S_{q-1}\} \tag{14}$$

$$v_t \overset{i.i.d.}{\sim} N(0, \sigma_v^2) \tag{15}$$

where $\{y_t\}$ is the observed signal, $\{b_t\}$ is the transmitted signal, S_i a symbol from the alphabet, $\mathbf{h} = \{h_0, \ldots, h_{n-1}\}$ is the channel filter, n is the number of filter taps and t is an integer.

This may be represented in an equivalent state-space form:

$$\begin{cases} \mathbf{x}_t = \left[\begin{array}{c|c} \mathbf{0}^{\mathbf{T}} & 0 \\ \hline \mathbf{I} & \mathbf{0} \end{array}\right] \mathbf{x}_{t-1} + \begin{bmatrix} 1 \\ \mathbf{0} \end{bmatrix} b_t \\ y_t = \mathbf{h}^{\mathbf{T}} \mathbf{x}_t + v_t \end{cases} \tag{16}$$

where $\mathbf{x}_t = [b_t, b_{t-1}, \ldots, b_{t-n+1}]^{\mathbf{T}}$ is the state (a vector of length n) and $\mathbf{0}$ is the null vector of length $n-1$. A collection of states, $\mathbf{x}_{1:t}$, can be exactly determined by $\{b_1, \ldots, b_t\}$ and some initial conditions, given equation 16.

[1] Initial samples could be created by any batch-based MCMC method.

[2] A similar approximation is made in the communications literature concerning trace-back in a Viterbi equalizer [13]. After a fixed delay the paths stored in the Viterbi algorithm are assumed to have joined, or if not the most likely path is chosen at this time.

[3] This will be used in the next iteration.

5.2. Results

To compare methods we tried them on communications channels simulated directly from equation 13 using a binary amplitude modulation scheme where bits are encoded as ± 1. All methods require an estimate of the initial density: this was obtained using a Gibbs sampler (see [3] for details) which was run over a frame of length 200 samples. The channel was chosen to be $\mathbf{h} = [0.766, 0.575, 0.287]^{\mathbf{T}}$ and the lag $p = 15$, five times the channel length. In all cases 151 particle streams were chosen. Rejuvenation/resampling took place when the effective sample size fell below 135. The input bit at time t was determined by summing over all states at time t which would have that bit set — this is a marginal maximum a posteriori estimate.

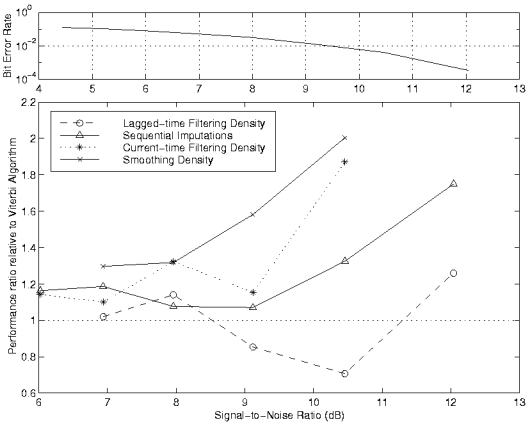

Figure 1: Top Graph shows Bit Error Rate vs. SNR for trained Viterbi Algorithm. Bottom graph shows relative performance ratio (ratio of Bit Error Rates) for proposed algorithms.

The performance of the various algorithms are shown in figure 1 relative to the Viterbi algorithm using 20 training symbols in a frame of 200, the channel estimate being obtained by least mean squared error. A value of 2 for performance indicates that the bit error rate is reduced by a factor of 2. Note that no training is given to our own algorithms: they are performing blind. However, if training symbols are available, it is straightforward to make use of them by incorporating them as prior information.

Differential encoding was not employed, but the bits were inverted when the phase ambiguity was incorrectly resolved (choosing $\mathbf{h} = [-0.766, -0.575, -0.287]^{\mathbf{T}}$ is an equally probable solution). In a practical system, either differential encoding or a small number of known bits are required.

5.3. Comparison of Algorithms

All algorithms performed well, especially at high Signal-to-Noise ratios. As expected, tracking the smoothing density directly produced the best results. The method using the lagged-time filtering density was surprisingly poor. Since this ran almost a factor of 2 slower (it requires 2 forward passes — the most computationally intensive part of the algorithms), it has limited use. If a reduced complexity algorithm is required, one may use sequential imputations or the current-time filtering density method.

As with all implementations of the particle filter, the computations for the propagation of each state trajectory may be performed in parallel, allowing significant speed-up in practical systems. This propagation is the most computationally intensive part of the algorithm: the complexity for each trajectory is similar to that of the Viterbi algorithm.

Errors typically occurred in bursts: the burst could last up to 4 symbols (not necessarily all would be in error), though 2 was most common.

Rejuvenation occurred most iterations on the sequential imputations algorithm, although the gap increased to about every 10 iterations with less noise present. When using the smoothing density, typical values for the time gap between rejuvenation varied between 150 and 700 as noise levels decreased. The other algorithms lie somewhere in between these two extremes.

6. REFERENCES

[1] C. Berzuini, N. G. Best, W. R. Gilks, and C. Larizza. Dynamic conditional independence models and Markov chain Monte Carlo methods. *J. Amer. Statist. Assoc.* , 92(440):1403–1412, 1997.

[2] C. K. Carter and R. Kohn. Gibbs sampling for state space models. *Biometrika*, 81(3):541–553, 1994.

[3] T. Clapp and S. J. Godsill. Bayesian Blind Deconvolution for Mobile Communications. In *IEE Colloqium on Adaptive Signal Processing for Mobile Communication Systems*, pages 9/1–9/6, Savoy Place, London, 29th October 1997. Reference No: 1997/383.

[4] A. Doucet. On Sequential Simulation-Based Methods for Bayesian Filtering. Technical Report CUED/F-INFENG/TR. 310, Cambridge University Department of Engineering, 1998.

[5] A. Doucet and P. Duvaut. Fully Bayesian Analysis of Hidden Markov models. In *Proceedings of EUSIPCO*, 1996.

[6] N. Gordon, D. Salmond, and A. F. M. Smith. Non-linear / non-gaussian bayesian state estimation. *IEE Proceedings of Radar and Signal Processing, Pt.F*, 140(2):107–113, 1993.

[7] J.E. Handschin and D.Q. Mayne. Monte Carlo Techniques to Estimate the Conditional Expectation in Multi-stage Non-Linear Filtering. In *Int. J. Cont.*, volume 9, pages 547–559, 1969.

[8] J. A. Heller and I. M. Jacobs. Viterbi Decoding for Satellite and Space Communication. In *IEEE Transactions on Communcation Technology*, volume COM-19, pages 835–848, October 1971.

[9] A. Kong, J.S. Liu, and W. H. Wong. Sequential Imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, March 1994.

[10] J. S. Liu and R. Chen. A note on Monte Carlo methods for dynamical systems. *unpublished*, 1996.

[11] J.S. Liu and R. Chen. Blind Deconvolution via Sequential Imputations. *Journal of the American Statistical Association*, 90(430):567–576, 1995.

[12] M.K. Pitt and N. Shephard. Filtering via simulation: Auxiliary particle filters. October 1997.

[13] J. G. Proakis. *Digital Communications*. Electrical engineering series. McGraw-Hill, 3rd edition, 1995.

[14] V.S. Zaritskii, V.B. Svetnik, and L.I. Shimelevich. Monte Carlo Technique in Problems of Optimal Data Processing. In *Auto. Remo. Cont.*, volume 12, pages 95–103, 1975.