PULSE TRAIN DEINTERLEAVING: ALGORITHMS AND COST CRITERIA

Keith S.M. Lee, Michael J. Rowe and Vikram Krishnamurthy *

Department of Electrical and Electronic Engineering,
University of Melbourne,
Parkville, Victoria 3052, Australia.

ABSTRACT

Consider the problem where pulse trains transmitted from a known number of sources are received on a single communications channel. These pulses are corrupted with noise. The deinterleaving problem is to determine which source contributed which pulse and the periods and phases of each source. This paper explores the performance of a number of deinterleaving algorithms. We propose an alternative to the existing forward dynamic programming (FDP) technique: simulated annealing (SA). It can use either the same cost function as for FDP, or an L_1 or L_2 norm output error cost function. We also investigate modelling the noise by heavy-tailed distributions, in addition to white Gaussian noise (WGN).

1. INTRODUCTION

When pulse trains from a number of different sources are received on a single communications channel, the problem is then to identify which source contributed each pulse. This is known as deinterleaving. Here we consider the case where the sources are *periodic* and their number is *known*. Furthermore, it is assumed that the pulses have been corrupted by additive white Gaussian noise (WGN). By deinterleaving the received signal, the periods and phases of each source can then be estimated. Applications lie in radar detection and potentially in computer communications and neural systems (see [1], [2] and the references therein).

Proposed approaches to the deinterleaving problem include histogramming [1] and folding. However, they work best when the jitter noise is small and require *a priori* information regarding periods and phases in order to choose suitable initial conditions.

Note that this problem cannot be solved by first using continuous optimisation techniques and then checking all integer neighbours of the real-valued solution. This is because the cost function one obtains is not differentiable (it involves the function $\min$); nor can it be expressed in a closed form.

In [2], the pulse-train deinterleaving problem is formulated as a stochastic discrete-time dynamic linear model (DLM). This is a time-varying linear system using state-space form, in which the state and observation matrices at each time instant belong to a finite set of possible values.

When there are multiple sources, the optimal solution is to select from all source sequences, called paths, the one with the minimum prediction-error cost. The problem is $\mathcal{NP}$-hard, i.e. involving exponential complexity, so one must resort to some sub-optimal scheme. In [2], two such schemes are investigated: forward dynamic programming (FDP) with fixed look-ahead, and probabilistic teacher (PT). We investigate a third scheme: simulated annealing (SA), using both the DLM formulation and an alternative output error formulation. Simulated annealing is a useful method for tackling combinatorial optimisation problems. When minimising a function, it is able to climb out of a local minimum with some probability, in the hope of finding the global minimum.

The FDP algorithm does a full tree-search over a fixed look-ahead interval Δ. The prediction error cost of all of these sequences (each of length $\Delta + 1$) is computed. In this way improbable paths are eliminated and FDP updates the most probable sequences and costs terminating at each source.

The main contributions of this paper are as follows:

- comparison of performance of different deinterleaving algorithms;
- comparison of different cost criteria;
- investigation of different noise distributions.

We find that SA is superior to FDP in a number of conditions.

2. PROBLEM FORMULATION

2.1. Signal Model

Consider N sources, each generating periodic pulse trains. Define $T^i,\ \phi^i,\ i = 1, \ldots, N$, to be the period and phase of the ith source, respectively. Also, let $t_k,\ k = 1, 2, \ldots$ be the time of arrival (TOA) of the kth pulse at the receiver in the absence of measurement noise. Let $s_k = i$ mean that source i is active at pulse instant k. Let $e_i,\ i = 1, \ldots, N$, be the unit column vector in $\mathbf{R}^N$ with 1 in the ith position. Let $X_k \in \{e_1, \ldots, e_N\}$ be the state of the process, i.e.

$$X_k = e_i, \quad \text{if } s_k = i. \tag{1}$$

Consider an observation sequence $Y_k = (y_1, \ldots, y_k)$. Here, y_k is the *observed* TOA of the kth pulse at the receiver. Thus,

$$y_k = t_k + W_k, \quad \text{where } W_k \sim N[0, \sigma_w^2]. \tag{2}$$

In other words, the jitter noise is modelled as zero-mean WGN with known variance σ_w^2. It is assumed that the pulses summed at the receiver give no amplitude or width information which might help identify their sources. Although systems can use parameters such as direction of arrival (DOA) and carrier frequency to assist in the deinterleaving, we assume no knowledge of this information.

* This work was supported by the Australian Research Council

2.2. Formulation as Dynamic Linear Model

By formulating the deinterleaving problem as a DLM, we can perform estimation using a Kalman filter (KF).

Let $\tau_k^i, \quad i = 1, 2, \ldots$ denote the last time source i was active up to and including the arrival of the kth pulse. τ_k^i is initialized to ϕ^i until source i first becomes active. Then

$$\begin{aligned} \tau_k^i &= \begin{cases} \tau_k^i + T^i, & \text{if } (k+1)\text{th pulse is due to source } i \\ \tau_k^i, & \text{otherwise} \end{cases} \\ \tau_1^i &= \phi^i. \end{aligned} \tag{3}$$

Define the $\mathbf{R}^N$ vectors

$$\begin{aligned} \tau_k' = (\tau_k^1, \ldots, \tau_k^N), \qquad T' = (T^1, \ldots, T^N), \\ \phi' = (\phi^1, \ldots, \phi^N). \end{aligned} \tag{4}$$

Now define the state vector x_k to be

$$x_k = \begin{pmatrix} T \\ \tau_k \end{pmatrix}, \quad x_1 = \begin{pmatrix} T \\ \phi \end{pmatrix}. \tag{5}$$

Finally, x_k can be written in state-space form as the following DLM

$$\begin{aligned} x_{k+1} &= F(X_{k+1})x_k + G(X_k)v_k, \quad v_k \sim N[0, Q] \\ y_k &= H'(X_k)x_k + W_k, \quad W_k \sim N[0, R]. \end{aligned} \tag{6}$$

Here v_k is independent WGN with variance $Q = \sigma_v^2$, $R = \sigma_w^2$, and

$$\begin{aligned} F(X_{k+1}) &= \begin{pmatrix} I_N & 0_{N\times N} \\ diag(X_{k+1}) & I_N \end{pmatrix}, \\ H'(X_k) &= \begin{pmatrix} 0_{1\times N} & X_k' \end{pmatrix}, \quad G(X_k) = 0_{2N\times 1}. \end{aligned} \tag{7}$$

Here, $0_{M\times N}$ represents the zero matrix with dimensions $M \times N$ and I_N is the identity matrix of dimensions $N \times N$.

2.3. Kalman Filter Estimator & Prediction Error Cost

Given a source sequence, Kalman filtering enables one to obtain an estimate of x_k. The source sequence $\chi_\kappa = S_\kappa^p$ is denoted as path p. Let F_k^p, G_k^p and H_k^p be the associated DLM matrices in (6). For given noise-corrupted observations Y_κ, let

$$\begin{aligned} x_{k+1|k}^p &= E\{x_{k+1}|Y_k, S_k^p\} \\ \Sigma_{k+1|k}^p &= E\{(x_{k+1} - x_{k+1|k})(x_{k+1} - x_{k+1|k})'|S_k^p\} \end{aligned} \tag{8}$$

be the predicted state estimate and the predicted state-covariance estimate at time $k+1$, given the path p.

Define e_{k+1} to be the one-step prediction error at time $k+1$ given the path p. The KF for the DLM (6), given the path p and observations Y_κ, is described by the standard KF equations [2], initialized with *a priori* estimates $x_{1|0}^p$ and $\Sigma_{1|0}^p$.

The KF one-step prediction error cost of the path p, denoted as J_κ^p, is then given by $J_\kappa^p = \sum_{k=1}^\kappa (e_k^p)^2$.

The optimal scheme is to calculate the one-step prediction error costs of all N^κ possible paths, given the observation sequence Y_κ. Then the optimal (MAP) path p^* is given by $p^* = \arg\min_p J_\kappa^p$, where J_κ^p is defined above. The KF on the optimal path p^* gives filtered estimates of the state and therefore periods.

2.4. Reformulation of Search Space

Reduction in the size of the search space occurs through:

- Formulation in terms of periods and phase differences, i.e. tuples $(T^1, \ldots, T^N, d^1, \ldots, d^{N-1})$, where $d^i = \phi^1 - \phi^{i+1}$, $i \in [1, N-1]$, since each source sends pulses periodically. Note that the source sequence depends on phase differences, not the actual phases themselves.
- Ordering periods, i.e. $T^1 \geq T^2 \geq \cdots \geq T^N$;
- Eliminating tuples whose greatest common divisor is not one (as the order of source assignments is unaffected if every element in the tuple is multiplied by a constant).

2.5. Output Error Formulation

An alternative, far simpler formulation is possible. Since each source sends pulses periodically, represent the (estimated) phase and period of each source in a $2N$-tuple or vector. (This corresponds to x_1 in the DLM formulation.) For a given tuple, one can generate the corresponding sequence of TOAs, $(\hat{t}_1, \ldots, \hat{t}_\kappa)$.

Now, define the least-squares (L_2 norm) cost function as:

$$\sum_{k=1}^{\kappa} (y_k - \hat{t}_k)^2, \tag{9}$$

where $(\hat{t}_1, \ldots, \hat{t}_\kappa)$ is the sequence of observations corresponding to the current state. The task is then to find the tuple which gives the minimum cost. Labelling of the sources is arbitrary, so let $T^1 \geq T^2 \geq \cdots \geq T^N$.

If the model of non-Gaussian noise is more appropriate than WGN, then the L_1 norm is also worth investigating.

3. SIMULATED ANNEALING

Simulated annealing is a probabilistic global optimisation technique. It is based on the physical annealing of solids, in which sufficiently slow cooling gives rise to a regular atomic structure with low energy.

When the algorithm moves to a configuration or 'state' which has a lower cost, it accepts it as the new state. However, unlike non-hill-climbing algorithms, if the cost of the new state is worse, it is not immediately rejected (which would result in getting trapped in a local minimum). Instead, it accepts the new state with probability $e^{-\Delta E/T}$, where $\Delta E = (High\_cost - Low\_cost)$.

This depends on the 'temperature' T, which is reduced according to a 'schedule'. Thus for the high starting temperatures, most new configurations are accepted, whereas for low temperatures good elements of configurations have been 'frozen' in. Since the moves and acceptance/rejection decisions are made probabilistically, SA is a stochastic method, which will not necessarily find the same configuration each time.

3.1. Move function for Prediction Error SA

The move functions chosen for the output error and DLM formulations are essentially the same, except that the former deals with phases and the latter with phase differences. Two types of move are used: a 'large' move for minimising the space diameter, and a 'small' move to achieve a smoother move landscape.

For the output error formulation, a new small move is chosen by taking one of the ϕ^i's at random, and choosing a new value,

ϕ_{new}, which satisfies the following restrictions: $\phi^i - w \leq \phi_{new} \leq \phi^i + w$ and $\phi_{min} \leq \phi_{new} \leq \phi_{max}$, where w is some window size and ϕ_{max} (ϕ_{min}) is an upper (lower) bound for the phases.

For the DLM formulation, phase differences d^i are changed in the same way as ϕ^i above.

The large move operates by changing a number of periods, chosen at random, again adjusting their value within some window size, but also maintaining the condition $T^1 \geq T^2 \geq \ldots \geq T^N$.

3.2. Cooling Schedule

In [3], it is shown that SA is guaranteed to converge to the *global* minimum if $T(i) \rightarrow 0$ no faster than $O(\frac{1}{\log i})$, where $T(i)$ is the ith temperature of the cooling schedule. In practice the annealing algorithm provides us with only a sub-optimal solution.

The cooling schedule is defined by the following parameters: the initial temperature $T(0)$; the stop criterion, determining the final temperature; the number of moves per temperature; some decrement rule for obtaining $T(i+1)$ from $T(i)$.

The initial temperature is chosen so that the ratio of *cost-increasing* moves accepted to total moves proposed is above some threshold. This means that initially the algorithm can move freely over the whole search space.

We use a fixed number of temperatures, chosen so that the last few temperatures are low. Also fixed is the number of moves at each temperature. The exponential decrement rule (first proposed in [4, 5]), is used, i.e. $T(i+1) = \alpha T(i)$, for $\alpha \in [0.86, 0.92]$.

3.3. Additions to Output Error SA Algorithm

When using the output error formulation, two additions to the basic SA algorithm were made: a local or restricted search at the end of the annealing, and a conditional reheat.

3.3.1. Restricted Search and Reheat

At the end of the schedule, the algorithm does a local search in the neighbourhood of the best configuration found (using the SA algorithm again), i.e. if the best tuple found is $(\hat{T}^1, \ldots, \hat{T}^N, \hat{\phi}^1, \ldots, \hat{\phi}^N)$ then the search is restricted to periods in the range $(\hat{T}^1 \pm w_l, \ldots, \hat{T}^N \pm w_l)$ for some w_l. However the phases are free to take any value within the original phase range. In this way, when configurations close to the correct one are found by the main run, there is a good chance of the local search then finding the correct one.

A reheat condition (e.g. [6]) is employed to deal with those cases when the SA originally fails to find a low score and hence a good configuration. The correct periods and phases result in $\hat{t}_k = t_k$. By (2) and (9), the average cost of the correct configuration (i.e. the global minimum for κ sufficiently large) for WGN is:

$$\begin{aligned} \overline{C_{true}} &= \sum_{k=1}^{\kappa} \frac{1}{\sqrt{2\pi}\sigma_w} \int_{-\infty}^{\infty} w_k^2 e^{-w_k^2/2\sigma_w^2} \, dw_k \\ &= \kappa \sigma_w^2. \end{aligned} \tag{10}$$

When the best cost found is above some threshold, say $2\overline{C_{true}}$, the reheat is activated. As a precaution, the total number of reheats allowed is prespecified to ensure termination of the algorithm.

The expected values of the 'true' cost for Gaussian noise, using the L_1 and L_2 norm cost functions are $\sqrt{\frac{2}{\pi}}\kappa\sigma_w$ and $\kappa\sigma_w^2$ respectively. For Rayleigh noise (with parameter α), the L_1 and L_2 norms give average 'true' costs of $\sqrt{\frac{\pi}{2}}\kappa\alpha$ and $2\alpha^2\kappa$ respectively.

3.4. Comparison of SA Cost Functions and FDP

The L_2 and L_1 norm cost functions have the advantage of robustness, i.e. they do not break down when the *a priori* estimates of periods and phases ($x^p_{1|0}$ in the DLM formulation) are far from their true values. For κ sufficiently large, the correct tuple always has the lowest cost, unlike the prediction error cost function. The norms have no parameters that must be determined, whereas a suitable value of $\Sigma^p_{1|0}$ must be determined in the prediction error case.

For SA, the prediction error cost function results in a computational cost of $(8N^2 + 8N + 4)M\kappa = O(N^2)M\kappa$ floating point operations (FLOPs), where N is the number of sources, M is the number of moves and κ is the number of pulse instants in the sequence. This is due to the structure of F^p_{k+1} and H^p_k, which means that the KF requires only $O(N^2)$ FLOPs, instead of $O(N^3)$.

A run of the FDP algorithm with a lookahead of Δ requires that $N^{\Delta+1}$ sequences of length Δ be examined for each pulse instant. Thus, $\Delta(8N^2 + 8N + 4)N^{\Delta+1}\kappa = \Delta O(N^{\Delta+3})\kappa$ FLOPs are required, where κ in this case is the number of pulse instants required for convergence.

In contrast, the L_1 and L_2 norm cost functions *per se* are not the computational bottleneck of the output error SA algorithm, unlike in the case of the prediction error cost function. Instead, the most computation time is spent constructing the new configurations every time a move is made. The number of FLOPs for a simulation is $(N\kappa + 3\kappa - 1)M$, i.e. linear in N and κ.

4. SIMULATION STUDIES

4.1. Estimation of Multiple Sources with WGN

SA is computationally more efficient than FDP in the following scenarios: *(i)* high corrupting noise (e.g. Fig. 1); *(ii)* a large number of sources (e.g. Fig. 2 and Table 1).

Consider the example of an interleaved pulse train with parameters $N = 2, T' = (80, 11), \phi' = (4, 3)$. The initial estimates for periods and phases were $x'_{1|0} = (96, 60, 1, 2)$. [1] Fig. 1 shows that the prediction error and L_2 norm SA are computationally less expensive than FDP for $\sigma_w^2 \geq 56$. Whereas FDP fails to correctly estimate the periods for $\sigma_w^2 \geq 90$, prediction error SA copes for σ_w^2 up to 100. L_2 norm SA appears to be able to cope with any physically realistic variance. e.g. for $\sigma_w^2 = 225$ (not shown on graph), correct estimation requires 6.73×10^7 FLOPs.

Fig. 2 refers to the example $N = 3, T' = (11, 30, 80)$, $\phi = (3, 4, 6)$, with $x'_{1|0} = (39, 40, 90, 3, 4, 2)$. The L_2 norm SA is more economical computationally than FDP for $\sigma_w^2 \geq 16$.

Table 1 compares FDP and the L_2 norm SA for the interleaved pulse train with parameters $N = 4$, $T' = (11, 30, 56, 80)$, $\phi' = (3, 4, 1, 6)$, and *a priori* estimate $x'_{1|0} = (20, 40, 46, 90, 3, 4, 3, 2)$. In this case, SA is superior to FDP for noise with variance only $\sigma_w^2 = 4$.

4.2. Non-Gaussian Noise

The Rayleigh and Cauchy distributions are heavy-tailed, i.e. they satisfy $P[U \geq u] \sim u^{-\alpha}h(u)$, as $u \rightarrow \infty$, in which $0 < \alpha < 2$ and h is slowly varying at infinity. SA is computationally more efficient than FDP for heavy-tailed noise.

Fig. 3 compares the performance of FDP and SA for the example $N = 2, T' = (80, 11)$, $\phi' = (4, 3)$, with varying levels of

[1] Since SA is a probabilistic algorithm, for each simulation numerous runs of SA were carried out to obtain its average performance.

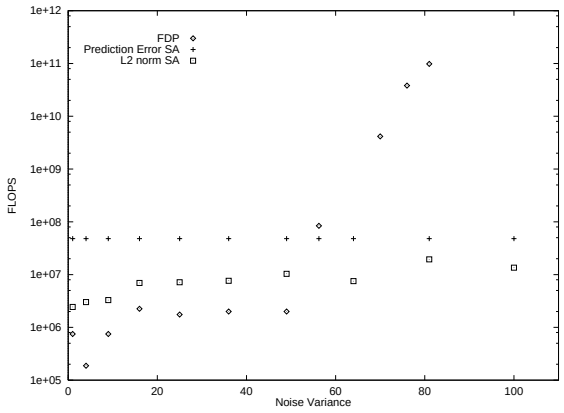

Figure 1: Comparison of FLOPS required to obtain the correct periods for SA and FDP. The example used is $T' = (80, 11), \phi' = (4, 3), x'_{1|0} = (96, 60, 1, 2)$ with Gaussian noise.

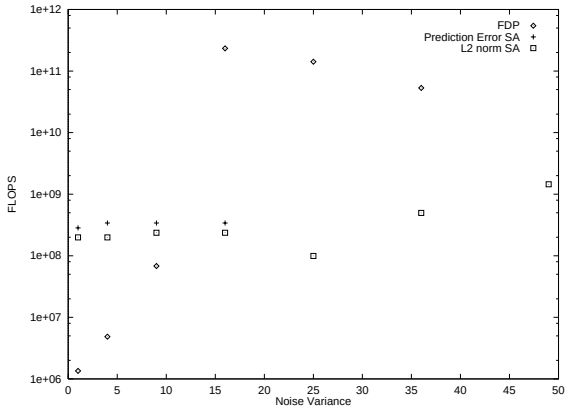

Figure 2: Comparison of FLOPS required to obtain the correct periods for SA and FDP. The example used is $T' = (11, 30, 80), \phi' = (3, 4, 6), x'_{1|0} = (39, 40, 90, 3, 4, 2)$ with Gaussian noise.

Rayleigh-distributed noise and initial estimate $x'_{1|0} = (96, 60, 1, 2)$. Since the noise is heavy-tailed, the L_1 norm cost function was used as well as the L_2 norm cost function. SA performs far better than FDP, even when the noise is low, despite there being only two sources. The L_1 and L_2 norm cost functions perform comparably.

Similar results are also obtained when Cauchy-distributed noise is used. Refer to the full paper [7] for examples of: Cauchy noise, larger numbers of sources and bad *a priori* estimates.

5. CONCLUSIONS

This paper shows that simulated annealing using the L_2 norm (least-squares) cost function outperforms forward dynamic programming in all of the following cases: *(i)* high variance of the corrupting noise; *(ii)* bad *a priori* estimates of the periods and phases, $x_{1|0}$; *(iii)* heavy-tailed noise; *(iv)* a large number of sources, N. In addition, the L_2 norm cost function avoids the need to choose a suitable value for $\Sigma_{1|0}$. This *a priori* estimate causes problems for FDP, as

Table 1: Comparison of FDP and SA for the 4-source case of $T' = (11, 30, 56, 80), \phi' = (3, 4, 1, 6), x'_{1|0} = (20, 40, 46, 90, 3, 4, 3, 2)$ with Gaussian noise. $\Sigma_{1|0}$ = 100.

σ^2	FDP			SA (L_2 norm)	
	Δ	κ	FLOPs	κ	FLOPs
1	2	200	4.198×10^6	100	4.386×10^9
2.5	4	700	4.702×10^8	100	1.168×10^{10}
4	11	150	4.540×10^{12}	150	5.083×10^{10}

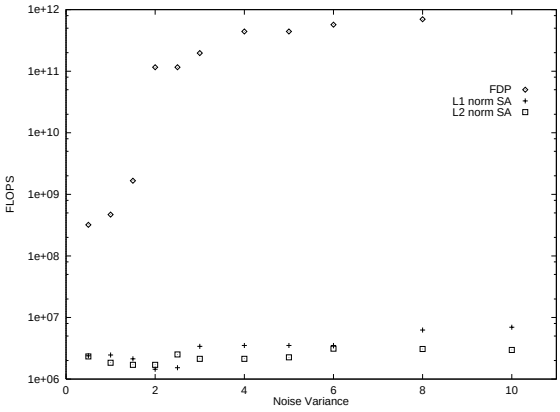

Figure 3: Comparison of FLOPS required to obtain the correct periods for SA and FDP. The example used is $T' = (80, 11), \phi' = (4, 3), x'_{1|0} = (96, 60, 1, 2)$. The noise is Rayleigh-distributed.

there is no reliable rule for choosing it, and if chosen incorrectly FDP fails.

6. REFERENCES

[1] H. K. Mardia, "New Techniques for the Deinterleaving of Repetitive Sequences," *IEEE Proceedings*, vol. 136, no. 4, August 1989.

[2] J.B. Moore, V. Krishnamurthy, "Deinterleaving Pulse Trains Using Discrete-Time Stochastic Dynamic-Linear Models," *IEEE Trans. on Signal Processing*, vol. 42, no. 11, Nov. 1994.

[3] P. J. M. van Laarhoven, E. H. L. Aarts, *Simulated Annealing: Theory and Applications*, The Netherlands: D. Reidel Publishing, 1988.

[4] S. Kirkpatrick, C. D. Gelatt Jr., M. P. Vecchi, "Optimization by Simulated Annealing," *IBM Research Report RC 9355,* 1982.

[5] S. Kirkpatrick, C. D. Gelatt Jr., M. P. Vecchi, "Optimization by Simulated Annealing," *Science* vol. 220, no. 4598, 13 May 1983.

[6] D. T. Connolly, "General Purpose Simulated Annealing," *Journal of Operational Research*, **43,** 1992.

[7] K. S. M. Lee, M. J. Rowe, V. Krishnamurthy, "Pulse Train Deinterleaving: Algorithms and Cost Criteria," to be submitted.