Orthogonal Matching Pursuit based Linear prediction for Real-time Packet Loss Concealment

Leonardo Severi
Department of Electronics and Telecommunications
Politecnico di Torino
Torino, Italy
leonardo.severi@polito.it

***Abstract*—We present an Orthogonal Matching Pursuit based sparse linear prediction algorithm for real-time audio packet loss concealment. The method iteratively selects non-contiguous lags to model periodic signals, using efficient correlation updates and combining Burg initialization with sparse autoregressive prediction via optimized crossfading. Implementation on Raspberry Pi4 achieves an average of $\sim 245\mu s$ for model fitting and $< 20ns$ per sample prediction, suitable for networked music performance.**

***Index Terms*—Packet Loss Concealment, Networked Music Performance, Linear Prediction, Autoregressive, Orthogonal Matching Pursuit**

I. Introduction

This report presents a sparse Linear Prediction algorithm for real-time packet loss concealment in audio applications. This method selects a sparse subset of lags $\{d_1, d_2, ..., d_k\}$, enabling modeling of periodic signals while maintaining computational efficiency suitable for real-time constraints. The algorithm combines a Burg model for initial sample prediction with sparse autoregressive (SAR) for the main signal reconstruction, ensuring smooth transitions through optimized crossfading.

II. Algorithm Description

A. Problem Formulation

Given a discrete signal s_t, we seek to model it as:

$$s_t = \sum_{i=1}^{k} \phi_i s_{t-d_i} + \epsilon_t \tag{1}$$

where $\{d_i\}_{i=1}^k$ are sparse lags and $\{\phi_i\}_{i=1}^k$ are the corresponding coefficients with ϵ_t being an error term. The key idea is the iterative selection of lags based on their correlation with the prediction residual and heuristics aimed at keeping the process fast and accurate enough.

B. Iterative Lag Selection

The algorithm follows an Orthogonal Matching Pursuit (OMP) approach [1] adapted for time series. The mathematical procedure is shown in Algorithm 1.

Leonardo Severi's PhD Programme is funded by the European Union in the framework of the Resiliency and Recovery Plan (RRP), within the NextGenerationEU initiative.

Algorithm 1 Sparse Autoregressive via Iterative Lag Selection

1: **Input:** Signal s_t, max iterations K, max lag L
2: **Initialize:** $\mathcal{D} = \emptyset$, $\hat{s}_t = 0$, $\epsilon_t = s_t$
3: **for** $j = 1$ to K **do**
4: **Compute correlations:**
5: **for** $d = 1$ to L **do**
6: **if** $d \notin \mathcal{D}$ **then**
7: $\rho_{\epsilon,s}(d) = \mathbb{E}[\epsilon_t s_{t-d}]$
8: **end if**
9: **end for**
10: **Select best lag:**
11: $d_j = \arg\max_d |\rho_{\epsilon,s}(d)|$
12: $\mathcal{D} = \mathcal{D} \cup \{d_j\}$
13: **Re-optimize all coefficients:**
14: Construct $\mathbf{R}_{ij} = r_{|d_i - d_j|}$
15: Construct $\mathbf{r}_i = r_{d_i}$
16: Solve $\mathbf{R}\boldsymbol{\phi} = \mathbf{r}$
17: **Update residual:**
18: $\hat{s}_t = \sum_{i=1}^{j} \phi_i s_{t-d_i}$
19: $\epsilon_t = s_t - \hat{s}_t$
20: **if** stopping criterion met **then**
21: **break**
22: **end if**
23: **end for**
24: **Output:** Lags $\mathcal{D}$, coefficients $\boldsymbol{\phi}$

C. Correlation Computation

The key insight is that the correlation between the residual and a candidate lag can be computed efficiently:

$$\rho_{\epsilon,s}(d) = r_d - \sum_{i=1}^{j} \phi_i r_{|d-d_i|} \tag{2}$$

where $r_l = \mathbb{E}[s_t s_{t-l}]$ is the autocorrelation at lag l. This formulation substitutes line 7, exploits the stationarity assumption and avoids explicit residual computation (lines 18 and 19).

D. Coefficient Optimization

After selecting a new lag d_j, all coefficients are re-optimized by solving the Yule-Walker-like system with Tikhonov regularization:

$$(\mathbf{R} + \lambda \mathbf{I})\boldsymbol{\phi} = \mathbf{r} \tag{3}$$

where $\mathbf{R}$ is a $j \times j$ symmetric matrix with elements $R_{il} = r_{|d_i - d_l|}$, $\mathbf{r}$ is a vector with elements $r_i = r_{d_i}$, and $\lambda = 0.01 \cdot r_0$ provides numerical stability.

E. Hybrid Prediction Strategy

The algorithm employs a two-stage prediction approach:

1) Burg Model for Initial Samples: To ensure continuity at packet boundaries, we compute a traditional Burg model of order $p_b \leq 8$ using the precomputed autocorrelation values (following the optimized implementation in [2]). This model predicts the first $m_b = 20$ samples:

$$\hat{s}_t^{Burg} = \sum_{i=1}^{p_b} a_i s_{t-i}, \quad t \in [1, m_b] \tag{4}$$

2) Sparse AR for Main Prediction: The sparse AR model handles the remaining samples:

$$\hat{s}_t^{SAR} = \sum_{i=1}^{k} \phi_i s_{t-d_i}, \quad t > m_b \tag{5}$$

3) Crossfade Optimization: An optimal crossfade point c^* is determined by minimizing the squared difference over a sliding window:

$$c^* = \arg\min_c \sum_{i=0}^{w-1} (\hat{s}_{c+i}^{\text{Burg}} - \hat{s}_{c+i}^{\text{SAR}})^2 \tag{6}$$

where w is the crossfade window size. The final signal combines both predictions using smooth fade curves. The implementation uses $w = 12$ for Burg-to-SAR transitions and $w = 32$ for right boundary crossfades, with smooth weighting functions.

III. Theoretical Justification

A. Orthogonality Property

The lag selection criterion ensures that each newly selected lag has a component orthogonal to the span of previously selected lags. If $\mathbf{X} = [s_{t-d_1}, ..., s_{t-d_{j-1}}]$ represents the matrix of selected lag vectors and $\mathbf{e} = \mathbf{y} - \mathbf{X}\boldsymbol{\phi}$ is the residual, then from the normal equations:

$$\mathbf{X}^T \mathbf{e} = \mathbf{0} \tag{7}$$

A new lag s_{t-d} with $\langle s_{t-d}, \mathbf{e} \rangle \neq 0$ must have a component $\perp$ span$(\mathbf{X})$, guaranteeing that it brings new information.

B. Relationship to OMP

The algorithm is equivalent to OMP applied to the dictionary $\mathcal{D} = \{s_{t-1}, s_{t-2}, ..., s_{t-L+1}\}$, with L being the maximum count of computed lags. The correlation maximization step corresponds to finding the dictionary atom most correlated with the residual, while the coefficient re-optimization ensures the optimal projection onto the selected subspace.

IV. Computational Complexity

The algorithm's efficiency stems from computing the optimization function without the need to explicit compute the error vector.

- **Autocorrelation computation:** $\Theta(Nlog(N))$ per iteration using a Fast Fourier Transform (FFT) - based approach.
- **Lag Selection:** $O(kL)$ for k iterations
- **System solution:** $O(k^3)$ for k systems
- **Total complexity:** $O(k^2L + k^4 + Nlog(N))$

For typical applications with $k \leq 5$ and $L = 1024$, and $N = 2048$ being the considered window size.

V. Implementation Considerations

A. Stopping Criteria

The implementation uses several criteria:

- Maximum of $k = 3$ non-contiguous lags
- Minimum correlation threshold: $|\rho| > 0.2 \cdot r_0$
- Early termination if single-lag correlation exceeds 0.55
- Skip regions around selected lags (± 80 samples)

B. Signal Conditioning

1) Compression and Limiting: To prevent instability, predicted samples undergo soft compression:

- Linear region: $|s| < 1.01 \cdot \max(|s_{\text{hist}}|)$
- Compression region with smooth polynomial transition using a cubic spline
- Hard limiting at $1.5 \cdot \max(|s_{\text{hist}}|)$ with gradual decay when triggered the first time

C. Model Fitting Heuristics

The current implementation uses:

- L1-norm normalization of coefficients when $\sum_i |\phi_i| < 1$, sacrificing accuracy in favor of preserving signal amplitude.
- A shock detection mechanism that monitors the variance ratio between consecutive windows of 512 samples, limiting the history when $\log_{10}(\sigma_i^2/\sigma_{ref}^2) \geq 1$, where σ_{ref}^2 is the variance of the most recent 512 samples (closest to the lost packet) and σ_i^2 is computed for each preceding window moving backward, thus preventing model contamination from abrupt signal changes such as note onsets or transients.

VI. Preliminary performance analysis

A. Qualitative assessment

For note onsets that express a strong periodicity, the algorithm usually stops after selecting a single lag, which boils down to a form of pattern replication (per section V-C). However, this is perceptually transparent, as the signal is prolonged according to its periodical form (see fig. 1).

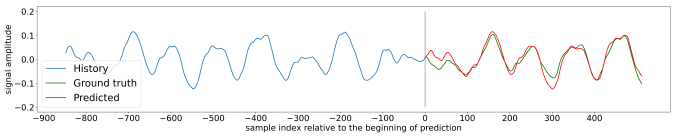

Fig. 1. Example showing single-lag SAR prediction (pattern replication) with Burg initialization on a piano signal

B. Computational Efficiency

The implementation achieves sub-millisecond execution through:

- Optimized C++ code, using float32 as main numerical type.
- Efficient computation of FFT-based autocorrelations (use of the *FFTW* library [3] with precomputed wisdom)
- Single-core elaboration with no need for synchronization mechanisms
- Vector operations through the *Eigen3* library [4]

Preliminary analyses conducted on a Raspberry Pi4 (standard ARM64 hardware) with $N \leq 2048$, $L \leq 1024$, $k \leq 3$ show:

- Model fitting: $\approx 245\mu s$ average
- Sample prediction: $< 20ns$ per sample

REFERENCES

[1] Y. Pati, R. Rezaiifar, and P. Krishnaprasad, "Orthogonal matching pursuit: recursive function approximation with applications to wavelet decomposition," in *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pp. 40–44 vol.1, 1993.

[2] K. Vos, "A fast implementation of burg's method," *OPUS codec*, 2013.

[3] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005. Special issue on "Program Generation, Optimization, and Platform Adaptation".

[4] G. Guennebaud, B. Jacob, *et al.*, "Eigen v3." http://eigen.tuxfamily.org, 2010.