Audio Restoration Techniques using Extended Kalman Filters: A Review and Evaluation

Affan Ul Ain
Department of Electrical Engineering and Automation, Aalto University.
Email: affan@aalto.fi

I. Introduction

In many different applications, including voice recognition, music composition, and telecommunications, the restoration of audio signals is an essential task. The existence of undesired noise, artifacts, or disruptions, which lower the signal's quality and have an impact on its comprehensibility or aesthetic appeal, is one of the most frequent problems in audio restoration. Here, we investigate state-of-the-art developments in audio restoration methods with an emphasis on the use of Extended Kalman Filtering (EKF) along with new noise estimates and filtering improvements.

The first section of the study provides a thorough literature overview of the numerous methods used for this purpose by different researchers. The problems with audio signal processing that are deteriorated by noise are then explained in detail, transforming them into an issue that can be estimated using Extended Kalman Filters (EKFs). For readers who are unfamiliar with EKFs, theory and a general grasp of EKFs are also provided.

The main difficulty in using EKF for audio restoration is then covered in the paper, which is figuring out the right noise estimates and filtering settings. We cover several approaches to this problem, such as quantile-based variance estimation of background noise and modifications to the Extended Kalman Filtering procedure itself.

Subsequently, various improvements to the Extended Kalman Filtering procedure are examined in order to boost audio restoration's capacity for noise reduction. These improvements include the use of forward/backward filtering to solve nonstationarity in audio signals, a bootstrap technique for filter initialization, and stability testing utilizing Levinson recursion. By incorporating these enhancements, parameter tracking, nonstationarity, and initialization issues in practical applications are addressed, improving the effectiveness of the EKF approach in noise reduction and signal restoration.

The efficiency of click detection and restoration algorithms in audio processing is then evaluated using a number of thorough assessment methodologies that are mentioned in the study. These techniques include assessing the effectiveness of click degradation restoration and detection, quantifying the improvement in Signal-to-Noise Ratio (SNR), performing perceptual evaluations with the Perceptual Evaluation of Audio Quality (PEAQ) approach, and examining the impact of click degradation amplitude on approach effectiveness. Through the use of both objective measures and subjective assessments, researchers can learn more about how well audio restoration methods function under different circumstances and in different degradation scenarios.

The conclusion, references, and an example Python code implementation of the equations described in this work are included in the final part. Extensive memory sizes and surroundings can replicate the results, which may help explain how audio restoration jobs utilizing Extended Kalman Filters are fundamentally implemented.

II. Literature Review

The restoration of deteriorated audio signals has drawn a lot of interest in the field of audio signal processing because of its applications in a number of industries, including telecommunications, music production, and archival preservation. Since the development of digital technology, audio signals have been vulnerable to a number of degradations, such as impulsive noise, quantization distortion, background noise, and other abnormalities. As a result, there is an urgent need for efficient restoration methods to improve the caliber of audio recordings and guarantee signal reproduction integrity.Many diverse approaches have developed over the years; we will go over the most widely used ones that have adopted filtering and predictive modelings of the situation at hand.

The Prediction-based Adaptive Median Filtering method is a unique technique to audio signal restoration presented in the work by Manikandan and Ebenezer [8] . This technique effectively suppresses degradations in audio signals, including clicks, crackles, impulses, and scratches, by integrating steps of prediction, detection, and adaptive median filtering. The suggested algorithm outperforms conventional techniques in the restoration of deteriorated audio signals because it integrates the Normalized Least Mean Square (NLMS) algorithm for prediction and adaptive window size selection for median filtering.

Similar to this, Avila and Biscainho[2] provide a framework for Bayesian restoration applied to audio signals that have been deteriorated by impulsive noise, which is represented as discrete pulses. The research presents a complete approach to impulsive noise removal and audio signal restoration by integrating the reversible-jump Metropolis-Hastings algorithm for parameter estimation with a hierarchical Bayesian model. The suggested approach shows promising results in restoring audio signals distorted by spurious noise and improves click removal accuracy.

Troughton [11] uses a sinusoidal model with autoregressive residuals to investigate Bayesian restoration methods for quantized audio signals. The goal of the research is to lessen quantization distortion and enhance signal quality by modeling the audio signal as the sum of sinusoids plus an autoregressive process. In order to restore quantized audio signals, the suggested Bayesian framework uses Markov Chain Monte Carlo (MCMC) techniques and suitable prior distributions for parameter estimation.

Islam and Saplakoglu [5] provide a Kalman filtering-based real-time flute note restoration technique for noisy observations. The research illustrates how Kalman filtering works well in real-time circumstances to restore flute notes that have been distorted by noise by using state-space modeling of the notes and event detection techniques. With respect to identified model changes, the suggested method offers adaptive tuning of Kalman filters, providing a complete solution for continuous music playback.

For audio restoration, Kutty and Murthy [7] present a Kalman filter technique with quantile-based noise estimation. The research allows real-time noise reduction while streamlining the noise estimation procedure by integrating quantile-based noise estimate into the Kalman filtering process. The suggested technique provides a workable solution for audio restoration applications by minimizing broadband noise in audio signals with satisfying results.

In their exploration of the Extended Kalman Filter (EKF) theory, Canazza et al. [3] pay particular attention to signal parameter tracking, impulsive noise reduction, and broadband noise filtering. The study begins with a summary of the increasing interest in digital audio restoration, classifies restoration algorithms, and highlights the effectiveness of time-domain techniques, especially the EKF. The foundation for the suggested solutions is laid forth in the problem statement, which highlights difficulties in representing audio signals as time-varying autoregressive processes tainted by noise. The work presents several enhancements to the EKF algorithm, such as the addition of both forward and backward filtering techniques, stability checks, and a bootstrap procedure for parameter initialization. The outcomes of the experiments confirm that the EKF technique is effective in various signal circumstances and noise levels.

Finally, to improve the quality of signals damaged by additive white noise, Kutty et al. [6] suggest an Extended Kalman Filter (EKF) with quantile-based noise variance estimation. The research provides real-time noise reduction with signal quality preservation by integrating quantile-based noise estimate into the EKF framework. Based on simulations and comparisons with current methods, the suggested method provides an effective way to restore the quality of signals affected by additive white noise.

In summary, by putting forth creative methods for repairing damaged audio signals, these works enhance the field of audio signal processing. These techniques provide potential answers to a range of problems in audio signal restoration, from impulsive noise removal to broadband noise reduction and signal parameter tracking, by utilizing adaptive filtering, Bayesian inference, and Kalman filtering methodologies.

III. Research Problem

Let a time-varying autoregressive (AR) model of order p represent the audio signal $s(t), t = 1, 2, \ldots,$:

$$\begin{aligned} s(t+1) &= \sum_{i=1}^{p} a_i(t)s(t-i+1) + e(t) \\ &= a_1(t)s(t) + a_2(t)s(t-1) + \ldots \\ &\quad + a_p(t)s(t-p+1) + e(t) \end{aligned} \tag{1}$$

driven by the variance σ_e^2 of the Gaussian zero-mean white noise sequence $e(t)$. The random walk model describes the evolution of the time-varying coefficients $a_i(t)$:

$$a_i(t+1) = a_i(t) + w_i(t), \quad i = 1, \ldots, p \tag{2}$$

$w_i(t)$ are zero-mean Gaussian white processes with variance σ_w^2, which are independent of $e(t)$ and mutually uncorrelated, i.e., $E[w_i(t)w_j(t)] = 0$ for $i \neq j$. Furthermore, assuming that a combination of broadband noise $z(t)$ (a noise that has sound energy dispersed throughout a large portion of the audible range [1]) and impulsive noise $v(t)$ (a class of acoustic noise that includes unwanted, unwelcome sounds that happen almost instantly (like clicks and pops)[12]) corrupts the original signal $s(t)$ (regardless of $e(t)$ and $w_i(t)$), the resulting available signal $y(t)$ may be expressed as follows:

$$y(t) = s(t) + z(t) + v(t) \tag{3}$$

While $v(t)$ is assumed to be Gaussian zero-mean noise with $\sigma_v^2 = \infty$ if a click is present, or $\sigma_v^2 = 0$ otherwise, the noise $z(t)$ is assumed to be Gaussian zero-mean white noise with variance σ_z^2. Because it lacks information on $s(t)$, if a click is revealed at time t, the corresponding sample $y(t)$ must be deleted, and $s(t)$ needs to be recovered from $\{\ldots, y(t-1), y(t+1), \ldots$.

Thus, the problem of estimating the noise-free signal $s(t)$ and the model parameters a_p reduces to a nonlinear filtering problem in state space. The theory of Extended Kalman Filter (EKF) can provide a suboptimal solution to the problem, which is gained after linearization. Thus, let us first examine the concept of the Extended Kalman Filter.

Extended Kalman Filter Overview

The classical Kalman filter's ability to handle non-linear models is expanded by the Extended Kalman Filter (EKF) [13]. Although many real-world systems behave non-linearly, the Kalman filter works best for linear systems with Gaussian noise. We can represent these non-linear processes by linearizing the system around the estimate that the EKF provides.

Key Concepts and Equations

1) **Non-Linear Models**: Conventional Kalman filters predict and update the state using linear models. Non-linear functions can be used for state prediction using the EKF.
   - **Linear Model**: $\hat{x}_{t+1}^- = A\hat{x}_t + B$
   - **Non-Linear Model**: $\hat{x}_{t+1}^- = f(\hat{x}_t)$
2) **Model Linearization**: The EKF approximates the non-linear function $f(\cdot)$ with its linear approximation (tangent line) around the current estimate $\hat{x}_t$.

- **Tangent Line Approximation**: $f(x) \approx f(\hat{x}_t) + f'(\hat{x}_t)(x - \hat{x}_t)$
- **Linearized Model**: $\hat{x}_{t+1}^- = f'(\hat{x}_t)\hat{x}_t + [f(\hat{x}_t) - f'(\hat{x}_t)\hat{x}_t]$

3) **Jacobian Matrix**: The Jacobian matrix represents the first derivative of the non-linear function with respect to the state vector. It permits the non-linear function to be linearized.
   - **Jacobian Matrix**: $F = \frac{\partial f}{\partial x}$
4) **State Update**: The EKF's corrective step takes the linearized model into account and uses sensor readings to update the state estimate.
   - **State Update Equation**: $\hat{x}_t = (I - k_t)\hat{x}_t^- + k_t z_t$
5) **Kalman Gain**: Taking into account the covariance matrices, the Kalman gain modifies the impact of the predicted state and sensor measurements in the update step.
   - **Kalman Gain**: $k_t = P_t^-(P_t^- + R)^{-1}$
6) **Covariance Matrices**: Instead of scalar variances, the EKF uses covariance matrices to represent uncertainties in state predictions and sensor measurements.
   - **Variance Prediction**: $P_{t+1}^- = F \cdot P_t \cdot F^T + Q$
   - **Variance Update**: $P_{t+1} = (I - k_t) \cdot P_{t+1}^-$

With its ability to provide more precise estimation in systems with non-linear dynamics, the EKF finds application in a multitude of real-world situations. Although it necessitates careful consideration of matrix operations and numerical calculation of derivatives, it provides greater performance when addressing non-linear systems [13].

Problem Formulation as an Extended Kalman Filter System

According to the aforementioned hypotheses, the EKF can ideally manage the problem of recovering the signal $s(t)$ from the noisy measurements $Y(t) = \{y(t), y(t-1), \ldots, y(1)\}$ as demonstrated in [10]. For this reason, the non-minimal state-space form of the signal $s(t)$ in (1) is a useful representation:

$$s_q(t+1) = A_q[a_p(t)]s_q(t) + b_q e(t) \tag{4}$$

where $s_q(t) = [s(t), \ldots, s(t-p), \ldots, s(t-q+1)]^\top$, $q \geq p$ is the signal vector, $a_p(t) = [a_1(t), \ldots, a_p(t)]^\top$ is the vector of the AR model coefficients $b_q(t) = [1, 0, \ldots, 0]$, and $A_q(t)$ is the companion matrix associated with the extended parameter vector $a_q(t) = [a_p(t), 0^{q-p}]^\top$.

Now we have a non-minimal state-space description where the dimension of the state vector q is greater than the order of the autoregressive model p, it means we're including more information in our state vector than just the bare minimum needed for modeling the system. In this context, $q - p$ represents the difference between the length of the state vector q and the order of the autoregressive model p.

Now, because $q > p$, it implies that we're capturing more past information in our state vector than what is strictly necessary for modeling the system. This additional past information allows us to potentially reconstruct more samples that have been affected by impulsive noise, not just those immediately preceding or following the contaminated sample. By having this broader context from both sides of the affected samples, we increase our chances of effectively removing the impulsive noise and reconstructing the original signal accurately. Thus, having a non-minimal state-space description enables us to handle impulsive noise contamination more effectively by considering a wider temporal context.

Note that a reliable signal model can only be created when a noiseless signal is present, and that noise must be removed using an accurate signal model. Filtering and parameter tracking are hence closely related problems that require joint solution. The solution to their combined treatment can be obtained by combining the unknown AR model coefficients with the signal vector in a $p + q$ state.

$$\begin{cases} x(t+1) = f[x(t)] + u(t) \\ y(t) = c^\top x(t) + \zeta(t) \end{cases} \tag{5}$$

where:

$$\begin{aligned} f\left[\mathbf{x}(t)\right] &= \begin{bmatrix} \mathbf{A}_q(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_p \end{bmatrix} x(t) \\ \mathbf{u}(t) &= \begin{bmatrix} \mathbf{b}_q e(t) \\ \mathbf{w}(t) \end{bmatrix} \\ \mathbf{w}^\top(t) &= \begin{bmatrix} w_1(t) & \cdots & w_p(t) \end{bmatrix}^\top \\ \zeta(t) &= z(t) + v(t) \\ \mathbf{c}^\top &= \left[\mathbf{b}_q^\top, \mathbf{0}^\top\right] = [1, 0, \ldots, 0]\,. \end{aligned}$$

Linearization Step

We'll start by linearizing the nonlinear transformation f around the state estimate $\hat{x}(t|t-1)$.

Given: $f\left[\mathbf{x}(t)\right] = \begin{bmatrix} \mathbf{A}_q(t) & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_p \end{bmatrix} x(t)$

The Jacobian matrix $\frac{\partial f}{\partial \mathbf{x}}$ is:

$$\frac{\partial f}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial \mathbf{s}_q}{\partial \mathbf{x}} \\ \frac{\partial \mathbf{a}_p}{\partial \mathbf{x}} \end{bmatrix}$$

Since $\frac{\partial \mathbf{s}_q}{\partial \mathbf{x}} = A_q(t)$ and $\frac{\partial \mathbf{a}_p}{\partial \mathbf{x}} = 0$, the Jacobian matrix $\frac{\partial f}{\partial \mathbf{x}}$ evaluates to:

$$\frac{\partial f}{\partial \mathbf{x}} = \begin{bmatrix} A_q(t) & 0 \\ 0 & 0 \end{bmatrix}$$

Now, substituting $\mathbf{x} = \hat{\mathbf{x}}(t|t)$ into $\frac{\partial f}{\partial \mathbf{x}}$, we obtain the desired result:

$$\mathbf{F}(t) = \left.\frac{\partial f[\mathbf{x}]}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{x}(t|t)} = \begin{bmatrix} \mathbf{A}_q(t|t) & \mathbf{s}_p^\top(t|t) \\ \mathbf{0}_{(q-1)\times p} & \mathbf{I}_p \end{bmatrix} \tag{6}$$

where $\mathbf{x}^\top(t|t) = [\mathbf{s}_q^\top, \mathbf{a}_p^\top(t|t)]$ represents the filtered state trajectory obtained from the Extended Kalman Filter (EKF) algorithm, $\mathbf{A}_q(t|t) = \mathbf{A}_q[\mathbf{a}^p(t|t)]$, and $\mathbf{s}_p(t|t)$ denotes the vector composed of the first p components of $\mathbf{s}_q(t|t)$. Moreover, let

$$\Omega = \frac{\mathrm{cov}[\mathbf{u}(t)]}{\sigma_e^2} = \begin{bmatrix} \mathbf{b}_q\mathbf{b}_q^\top & \mathbf{0} \\ \mathbf{0} & \xi\mathbf{I}_p \end{bmatrix} \tag{7}$$

with $\xi = (\sigma_w^2/\sigma_e^2)$

Define the Process and Measurement Models

Prediction Step:

- **Predicted State Estimate**: The predicted state estimate $\hat{\mathbf{x}}(t|t-1)$ is obtained by applying the process model f to the previous estimated state $\hat{\mathbf{x}}(t-1|t-1)$.

$$\hat{\mathbf{x}}(t|t-1) = f\left[\hat{\mathbf{x}}(t-1|t-1)\right] \tag{8}$$

- **Covariance Estimate**: The covariance estimate $\Sigma(t|t-1)$ is computed by propagating the uncertainty from the previous time step using the state transition matrix $\mathbf{F}(t-1)$ and adding the process noise covariance Ω.

$$\Sigma(t|t-1) = \mathbf{F}(t-1)\Sigma(t-1|t-1)\mathbf{F}^{\top}(t-1) + \Omega \tag{9}$$

Update Step:

- **Updated State Estimate**: The updated state estimate $\hat{\mathbf{x}}(t|t)$ is obtained by combining the predicted state estimate with the innovation $\varepsilon(t)$ scaled by the Kalman gain.

$$\hat{\mathbf{x}}(t|t) = \hat{\mathbf{x}}(t|t-1) + \mathbf{K}(t)\varepsilon(t) \tag{10}$$

- **Updated Covariance Estimate**: The updated covariance estimate $\Sigma(t|t)$ is computed to reflect the reduction in uncertainty after incorporating the measurement information. It is adjusted by the Kalman gain and the measurement matrix $\mathbf{c}$.

$$\Sigma(t|t) = \left(\mathbf{I}_{p+q} - \mathbf{K}(t)\mathbf{c}^{\top}\right)\Sigma(t|t-1) \tag{11}$$

where $\Sigma(t|t) = E[(x(t) - \hat{x}^{(t|t)})(x(t) - \hat{x}^{(t|t)})^{\top}]$ is the state estimation error covariance and $\Sigma(t|t-1) = E[(x(t) - \hat{x}^{(t|t-1)})(x(t) - \hat{x}^{(t|t-1)})^{\top}]$ is the state prediction error covariance. Moreover,

$$\varepsilon(t) = y(t) - \mathbf{c}^{\top}\hat{\mathbf{x}}(t|t-1) = y(t) - \hat{\mathbf{s}}(t|t-1) \tag{12}$$

is the prediction error (Kalman filter innovation), and $K(t)$ is the Kalman gain, whose value depends on the click indicator function $\hat{d}^{(t)}$, which takes the value 1 if a click is detected at time t:

$$\begin{cases} \frac{\Sigma(t|t-1)\mathbf{c}}{\mathbf{c}^{\top}\Sigma(t|t-1)\mathbf{c}+l(t)}, & \text{if } \hat{d}^{(t)} = 0 \\ 0, & \text{if } \hat{d}^{(t)} \neq 0 \end{cases}$$

and $l(t) = \frac{\sigma_z^2}{\sigma_e^2(t)}$.

The lack of scale invariance of the parameter tracker in the update equations is a severe issue; for fixed ξ, the identification results depend on signal scaling (the parameter estimations change when measurements are multiplied by $\beta \neq 1$ unless ξ is reduced β^2 times).

The Exponentially Weighted Least Squares (EWLS) algorithm addresses the lack of scale invariance in the parameter tracker by introducing exponentially weighted updates. Unlike the fixed parameter tracker, EWLS dynamically adjusts to changes in signal scaling. In EWLS, the state estimate $\mathbf{x}^{(t|t)}$ is updated using a weighted sum of the previous estimate and the prediction error $\epsilon(t)$, with weights determined by the gain matrix $\mathbf{L}(t)$. The covariance matrix $\Sigma(t|t)$ is updated based on the difference between the previous covariance estimate $\Sigma(t|t-1)$ and the outer product of the gain vector $\mathbf{L}(t)$ and the observation vector $\mathbf{c}(t)$, with a scaling factor γ controlling the weighting of the update.

$$\hat{\mathbf{x}}(t|t) = \hat{\mathbf{x}}(t|t-1) + L(t)\varepsilon(t) \tag{13}$$

$$\Sigma(t|t) = \frac{1}{\gamma}\left(\Sigma(t|t-1) - L(t)\Sigma^{\top}\mathbf{c}^{\top}\right) \tag{14}$$

where $0 < \gamma < 1$ [3].

Click Detection

The detection of clicks is based on the value assumed at each t by the prediction error:

$$\hat{d}(t) = \begin{cases} 0 & \text{if } |\varepsilon(t)| \leq \mu\hat{\sigma}_{\varepsilon}(t) \\ 1 & \text{if } |\varepsilon(t)| > \mu\hat{\sigma}_{\varepsilon}(t) \end{cases} \tag{13}$$

where:

- $\hat{d}(t)$ is the detected click at time t.
- μ is the parameter determining the threshold for detection of impulsive noise.
- $\hat{\sigma}_{\varepsilon}(t)$ is the estimated innovation variance.
- $\varepsilon(t)$ is the prediction error.

The estimated innovation variance $\sigma_{\varepsilon}^2(t)$ is given by:

$$\hat{\sigma}_e^2(t) = \begin{cases} \lambda\hat{\sigma}_e^2(t-1) + (1-\lambda)\frac{\varepsilon^2(t)}{\eta(t)} & \text{if } \hat{d}(t) = 0 \\ \hat{\sigma}_e^2(t-1) & \text{if } \hat{d}(t) \neq 0 \end{cases} \tag{14}$$

where:

- λ determines the adaptation speed.
- $\eta(t) = c^T\Sigma(t|t-1)c + k(t)$.
- c is a vector.
- $\Sigma(t|t-1)$ is a covariance matrix.
- $k(t)$ is the Kalman gain.

Smoothing and Reconstruction

Given all measurements available up to time t, the best smoothed estimate $\hat{s}(t|t)$ is the smoothed estimate of $s(t|t)$. It is convenient to use $\hat{s}(t-q+1|t)$ as an estimate of $s(t-q+1)$ at time t, introducing a delay of q samples, in order to fully utilize the available information. For signal smoothing, then, $q = p$ suffices. It can be demonstrated that in the presence of clicks that, for a pth order AR process, a block with at least p "good" subsequent consecutive samples is necessary for good reconstruction [10]: a value $q \geq p + n$ is needed for a group of n consecutive samples corrupted by a click. One can use the aforementioned factor to obtain a variable state.

IV. Noise Estimation and Filtering Enhancements

The primary challenge in applying EKF to this audio restoration use-case is determining the appropriate noise estimation and filtering parameters. Several studies have revealed a variety of methods for choosing these parameters. An overview of the many implementation strategies that have produced the best restoration outcomes will be given in this section.

A. Quantile Based Estimation of Background Noise Variance

This subsection presents a new method for assessing background noise variance using a quantile-based algorithm, as described by Kutty et al. [6]. In order to estimate noise power, one often assumes that speech activity is absent and calculates the noise spectrum straight from an initial section of the noisy signal. Unfortunately, because speech identification is not performed during noise updates, this approach is not very good in monitoring non-stationary noise. A quantile-based strategy that draws inspiration from Martin's minimum statistic algorithm is suggested as a solution to this drawback.

1) Algorithm Description: Assuming there is no speech activity, the simplest method to estimate the noise power is to compute the noise spectrum straight from the first segment of the noisy signal. Unfortunately, because speech identification is not performed during noise updates, this approach is not very good in monitoring non-stationary noise. A quantile-based strategy that draws inspiration from Martin's minimum statistic algorithm is suggested as a solution to this drawback.

2) Quantile Calculation: Sorting the dataset and choosing the value that corresponds to a specific quantile yields the dataset's quantile. Within this particular context, the quantile denotes the lower portion of the noisy voice power spectrum. The quantile value was selected according to experimental findings that indicated the likelihood of a silent duration within particular segment lengths.

The calculation of the quantile q for a dataset $\{x_i, i = 0, \ldots, N\}$ is expressed as:

$$x_0 \leq x_1 \leq \ldots \leq x_N$$

The quantile value q is obtained as $x_{\text{int}(Nq)}$, where int() rounds to the nearest integer. For example, $q = 0$ corresponds to the minimum, $q = 1$ corresponds to the maximum, and $q = 0.5$ corresponds to the median.

3) Performance Considerations: The algorithm's capacity to follow non-stationary noise is affected by the window length selection; shorter windows provide greater tracking performance but may result in fewer accurate estimations. The arithmetic mean of the lower percentage (e.g., $q < 0.2$) of the noisy speech power spectrum is used as the noise estimation in each frequency band, with the average over all frequency bands providing the noise variance estimate, as opposed to directly using the median or other quantile values.

The quantile-based noise estimation algorithm, which incorporates adaptive tracking of noise features during signal processing, addresses the shortcomings of conventional methods and provides a reliable way for predicting background noise variance overall.

B. Improvements to Extended Kalman Filtering

This subsection mentions the enhancements to the Extended Kalman filter aimed at improving noise reduction capabilities, as introduced by Canazza et al. in [3].

1) Bootstrap Procedure: : The problem of filter initialization can be solved by implementing a bootstrap process. When the algorithm is restarted, there is usually a brief initial phase during which the noise reduction capabilities of the filter are impaired. We can use a bootstrap technique, in which the first section of the signal is put into the filter after being time-reversed, to lessen this. This makes it possible to estimate the parameters for appropriate model initialization, which improves signal restoration.

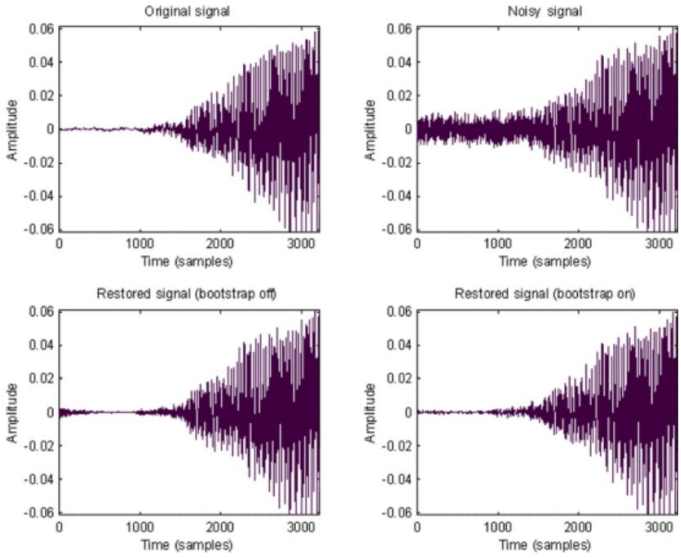

Fig. 1. Original signal (top-left); noisy signal (top-right); signal restored without (bottom-left) and with bootstrap (bottom right).

2) Stability Check: : Instability in the time-varying AR-model that the filter estimates is possible, although unusual. We can use the Levinson recursion [9] to perform a stability check at each time step to guarantee strong performance. We identify and skip parameter updates in the event of an unstable AR-model by measuring the magnitude of reflection coefficients, which avoids interfering with click detection.

3) Forward/Backward Filtering: : Conventional filtering techniques are challenged by the nonstationarity of audio signals. We can use forward and backward EKFs that operate on the signal in different directions to address this. We can accomplish greater noise reduction and enhanced detection of impulsive disturbances by merging the output from both filters by using a weighting strategy,

All these enhancements add to the overall efficacy of the EKF method in terms of noise reduction and signal restoration, providing answers to problems associated with parameter tracking, nonstationarity, and initialization in real-world applications. We show that these improvements are effective in enhancing signal quality and lowering noise artifacts through testing and comparison with current techniques.

V. Proposed Evaluation Methods

A number of approaches are put out in the paper [4] to assess how well click detection and restoration methods work in audio processing. One method is to evaluate how well click degradation can be detected at the beginning and how long it lasts. This assessment aids in determining how well the techniques identify the beginning and duration of audio disruptions.

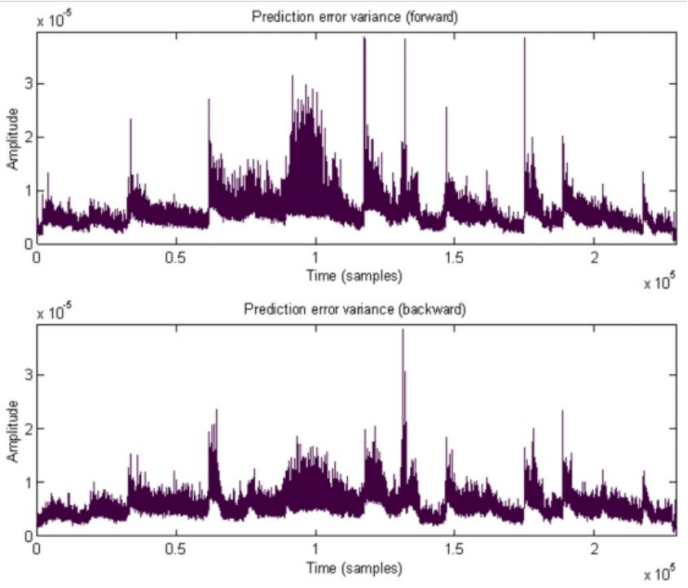

Fig. 2. Comparison between prediction error variance $\sigma^2_{\hat{\varepsilon}}(t)$ corresponding to forward-EKF and to backward-EKF.

Measuring the unified detection and restoration performance is another evaluation component that assesses the methods' overall efficacy without requiring prior knowledge of the click position or duration. This thorough evaluation sheds light on the techniques' capacity to identify and reduce click artifacts in audio sources.

In addition, the assessment involves measuring the enhancement in Signal-to-Noise Ratio (SNR) attained using the restoration techniques in contrast to the deteriorated audio. This objective metric provides a numerical representation of the improvement in audio quality brought about by the restoration procedure. One can compute the SNR improvement (ΔSNR) in the following way:

$$\Delta\text{SNR} = \text{SNR}_{\text{restored}} - \text{SNR}_{\text{degraded}}$$

A perceptual assessment of audio quality utilizing the Perceptual Evaluation of Audio Quality (PEAQ) approach is included in the evaluation in addition to objective measures. Researchers can learn more about the perceived improvements in audio fidelity by assessing the quality of the restored audio signal for both speech and music fragments subjectively.

Finally, an analysis is conducted on the influence of click degradation amplitude on the efficacy of approaches for detection and restoration. This assessment looks at how different click degradation amplitudes impact the approaches' capacity to identify and recover audio artifacts, offering important information about how reliable and efficient they are in various degradation conditions.

VI. Conclusion

This research concludes with a thorough review of the latest developments in audio restoration methods, with an emphasis on the application of Extended Kalman Filtering (EKF) in conjunction with cutting-edge noise estimates and filtering improvements. We have synthesized the different approaches used by researchers to address the problems of noise reduction and signal restoration in audio processing after a thorough examination of the literature.

We have illustrated the difficulties in obtaining ideal outcomes in practical situations by looking at the use of EKF in audio restoration and talking about the difficulties in determining filtering parameters and noise estimate. We have also investigated improvements to the EKF process itself, such as bootstrap methods for filter initialization and stability tests with Levinson recursion, with the goal of enhancing noise reduction performance.

Additionally, we have provided thorough evaluation methods that cover a wide range of quantitative metrics—such as an improvement in Signal-to-Noise Ratio (SNR)—as well as subjective assessments that employ perceptual assessment tools like PEAQ to gauge the efficacy of click detection and restoration procedures. These assessment methodologies shed light on how well audio restoration technologies work in different environments and deterioration scenarios.

For scholars and practitioners in the field of audio signal processing, this research task report is a useful resource even if its main focus is the consolidation of previous research and approaches. Although the appendix section offers a Python code that implements this audio restoration problem practically using Extended Kalman Filtering, future work may involve putting these techniques into practice and validating their results, further improving and optimizing the suggested methodologies for improved signal restoration and noise reduction in real-world applications.

References

[1] Acoustic glossary. Online. Accessed: April 7, 2024.

[2] Francisco R Avila and Luiz W Biscainho. Bayesian restoration of audio signals degraded by impulsive noise modeled as individual pulses. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(9):2470–2481, Nov 2012.

[3] Sergio Canazza, Giovanni De Poli, and Gian Antonio Mian. Restoration of audio documents by means of extended kalman filter. *IEEE Transactions on Audio, Speech, and Language Processing*, 18(6):1107–1115, Aug 2010.

[4] Bisrat Derebssa, Eyerusalem Adugna, Koen Eneman, and Toon Van Waterschoot. Detection and restoration of click degraded audio based on high-order sparse linear prediction. *Journal of EEA*, 40, 2022. Accessed: Apr. 11, 2024.

[5] Md Kamrul Islam and Gokhan Saplakoglu. Detection and restoration of sound of flute embedded in noise using real-time kalman filter. In *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 1993.

[6] PP Kutty, K Venkatesh, HR Nagendra, TS Nanjundaswamy, and AS Murthy. Quality improvement of signals corrupted by additive white noise using extended kalman filter with quantile based noise variance estimation. In *2011 3rd International Conference on Electronics Computer Technology*, Apr 2011.

[7] PP Rayan Kutty and A Sreenivasa Murthy. Kalman filter using quantile based noise estimation for audio restoration. In *2011 International Conference on Emerging Trends in Electrical and Computer Technology*, Mar 2011.

[8] S Manikandan and D Ebenezer. A new prediction based adaptive median filter for restoration of degraded audio signals. In *2008 International Conference on Signal Processing, Communications and Networking*, 2008.

[9] Richard Martin. Minimum statistic algorithm for noise power estimation in speech enhancement. *IEEE Transactions on Speech and Audio Processing*, 9(8):849–856, 2001.

[10] M. Niedwiecki and K. Cisowski. Adaptive scheme for elimination of broadband noise and impulsive disturbances from ar and arma signals. *IEEE Transactions on Signal Processing*, 44(3):967–982, Mar. 1996.
[11] Paul T Troughton. Bayesian restoration of quantised audio signals using a sinusoidal model with autoregressive residuals. In *Proceedings of the 1999 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics. WASPAA'99 (Cat. No.99TH8452)*, 1999.
[12] Wikipedia contributors. Impulse noise (acoustics), 6 2022. Accessed: April 7, 2024.
[13] Alan Zucconi. The extended kalman filter. https://www.alanzucconi.com/2022/07/24/extended-kalman-filter/, 2022. Accessed Apr. 2, 2024.

APPENDIX

Listing 1. Sample Python Code for implementation of Audio Restoration problem as that of an Extended Kalman Filter

```
def process_model(x, A_q, b_q, e, w):
    """
    Process model function

    Parameters:
        x (numpy array): State vector at time t
        A_q (numpy array): Companion matrix
            associated with extended parameter
            vector a_q
        b_q (numpy array): Coefficient vector for
            the noise term
        e (float): White noise sequence e(t)
        w (numpy array): White noise sequence w(t)

    Returns:
        numpy array: Predicted state vector at time
            t+1
    """
    p = len(w)
    x_pred = np.dot(A_q, x) + np.concatenate((b_q *
        e, w))
    return x_pred

def measurement_model(x, c, zeta):
    """
    Measurement model function

    Parameters:
        x (numpy array): State vector at time t
        c (numpy array): Measurement coefficient
            vector
        zeta (float): Combined noise term

    Returns:
        float: Measured output at time t
    """
    return np.dot(c, x) + zeta

def linearize_process_model(x_hat, A_q, s_p, p):
    """
    Linearization of the process model

    Parameters:
        x_hat (numpy array): Filtered state
            trajectory (x^T(t|t))
        A_q (numpy array): Companion matrix
            associated with extended parameter
            vector a_q
        s_p (numpy array): Vector composed of the
            first p components of s_q(t|t)
        p (int): AR model order

    Returns:
        numpy array: Jacobian matrix representing
            the linearized process model (F)
    """
    q = len(x_hat)
    F = np.zeros((q, q))
    F[:p, :p] = A_q
    F[p:, :p] = np.eye(p)
    F[p:, p:] = np.eye(p)
    return F


def calculate_Omega(b_q, sigma_e, sigma_w):
    """
    Calculate the covariance matrix Omega

    Parameters:
        b_q (numpy array): Coefficient vector for
            the noise term
        sigma_e (float): Variance of white noise
            sequence e(t)
        sigma_w (float): Variance of white noise
            sequence w(t)

    Returns:
        numpy array: Covariance matrix Omega
    """
    p = len(b_q)
    xi = sigma_w ** 2 / sigma_e ** 2
    Omega = np.zeros((p + 1, p + 1))
    Omega[0, 0] = np.dot(b_q, b_q.T)
    Omega[1:, 1:] = xi * np.eye(p)
    return Omega


def predict(x_hat_prev, Sigma_prev, F, Omega):
    """
    Perform the prediction step of the Kalman filter
        .

    Parameters:
        x_hat_prev (numpy array): Previous state
            estimate (x_hat(t|t-1))
        Sigma_prev (numpy array): Previous state
            covariance estimate (Sigma(t|t-1))
        F (numpy array): Jacobian matrix
            representing the linearized process
            model
        Omega (numpy array): Covariance matrix
            representing the process noise

    Returns:
        numpy array: Predicted state estimate (x_hat
            (t|t))
        numpy array: Predicted state covariance
            estimate (Sigma(t|t))
    """
    x_pred = np.dot(F, x_hat_prev)
    Sigma_pred = np.dot(np.dot(F, Sigma_prev), F.T)
        + Omega

    return x_pred, Sigma_pred

def update(x_pred, Sigma_pred, y, c, sigma_z, d_hat)
    :
    """
    Perform the update step of the Kalman filter.

    Parameters:
        x_pred (numpy array): Predicted state
            estimate (x_hat(t|t))
        Sigma_pred (numpy array): Predicted state
            covariance estimate (Sigma(t|t))
        y (float): Measurement at time t
        c (numpy array): Measurement matrix
        sigma_z (float): Variance of measurement
            noise
        d_hat (int): Click indicator function

    Returns:
        numpy array: Updated state estimate (x_hat(t
            |t))
        numpy array: Updated state covariance
```

```
        estimate (Sigma(t|t))
    """
    epsilon = y - np.dot(c, x_pred)

    if d_hat == 0:
        K = np.dot(Sigma_pred, c) / (np.dot(np.dot(c
            , Sigma_pred), c.T) + sigma_z)
    else:
        K = np.zeros_like(Sigma_pred)

    x_update = x_pred + np.dot(K, epsilon)
    Sigma_update = np.dot(np.eye(len(x_pred)) - np.
        dot(K, c), Sigma_pred)

    return x_update, Sigma_update


def detect_click(epsilon, mu, sigma_epsilon_hat):
    """
    Detect clicks based on the prediction error.

    Parameters:
        epsilon (float): Prediction error at time t
        mu (float): Threshold parameter for
            detection of impulsive noise
        sigma_epsilon_hat (float): Estimated
            innovation variance

    Returns:
        int: Detected click (0 for no click, 1 for
            click)
    """
    if abs(epsilon) <= mu * sigma_epsilon_hat:
        return 0  # No click
    else:
        return 1  # Click detected

def update_innovation_variance(sigma_epsilon_prev,
    epsilon, c, Sigma_prev, k):
    """
    Update the estimated innovation variance.

    Parameters:
        sigma_epsilon_prev (float): Previous
            estimated innovation variance
        epsilon (float): Prediction error at time t
        c (numpy array): Measurement matrix
        Sigma_prev (numpy array): Previous state
            covariance estimate (Sigma(t|t-1))
        k (numpy array): Kalman gain

    Returns:
        float: Updated estimated innovation variance
    """
    eta_t = np.dot(np.dot(c, Sigma_prev), c.T) + k

    if detect_click(epsilon, mu, sigma_epsilon_prev)
         == 0:
        sigma_epsilon_t = lambda_val *
            sigma_epsilon_prev + (1 - lambda_val) *
            (epsilon ** 2 / eta_t)
    else:
        sigma_epsilon_t = sigma_epsilon_prev

    return sigma_epsilon_t


def ewls_update(x_hat_prev, Sigma_prev, epsilon, c,
    gamma):
    """
    Perform the update step of the Exponentially
        Weighted Least Squares (EWLS) algorithm.

    Parameters:
        x_hat_prev (numpy array): Previous state
            estimate (x_hat(t|t-1))
        Sigma_prev (numpy array): Previous state
            covariance estimate (Sigma(t|t-1))
        epsilon (float): Prediction error (Kalman
            filter innovation) at time t
        c (numpy array): Measurement matrix
        gamma (float): EWLS parameter (0 < gamma <
            1)

    Returns:
        numpy array: Updated state estimate (x_hat(t
            |t))
        numpy array: Updated state covariance
            estimate (Sigma(t|t))
    """
    K = np.dot(Sigma_prev, c) / (np.dot(np.dot(c,
        Sigma_prev), c.T))
    x_update = x_hat_prev + np.dot(L, epsilon)
    Sigma_update = (1 / gamma) * (Sigma_prev - np.
        outer(K, np.dot(Sigma_prev, c.T)))

    return x_update, Sigma_update

audio_path = 'arp-space-190694.mp3'
y, sr = librosa.load(audio_path)

sigma_e = 0.5  # Variance of white noise sequence e(
    t)
sigma_w = 0.2  # Variance of white noise sequence w(
    t)
sigma_z = 0.1  # Variance of broadband noise z(t)

T = len(y)
e = np.random.normal(0, sigma_e, T)
w = np.random.normal(0, sigma_w, T)
z = np.random.normal(0, sigma_z, T)

v = np.zeros(T)
click_probability = 0.05
for t in range(T):
    if np.random.rand() < click_probability:
        v[t] = np.random.choice([-np.inf, np.inf])

y_noisy = y + z + v

time = librosa.times_like(y, sr=sr)

q = len(y)
x_hat = np.zeros(q)
Sigma = np.eye(q)

restored_audio = []

for t in range(T):
    x_pred, Sigma_pred = predict(x_hat, Sigma, F,
        Omega)
    y_pred = measurement_model(x_pred, np.array([1])
        , 0)
    d_hat = detect_click(epsilon_list[t], mu, np.
        sqrt(Sigma_pred[0, 0]))

    if d_hat == 0:
        x_update, Sigma_update = update(x_pred,
            Sigma_pred, y_noisy[t], np.array([1]),
            sigma_z, d_hat)
    else:
        x_update, Sigma_update = ewls_update(x_pred,
             Sigma_pred, epsilon_list[t], np.array
            ([1]), gamma)

    restored_audio.append(y_pred)

    x_hat = x_update
    Sigma = Sigma_update

plt.figure(figsize=(10, 8))

plt.subplot(3, 1, 1)
plt.plot(time, y, color='blue', alpha=0.7, label='
```

```
    Original␣Audio')
plt.xlabel('Time␣(s)')
plt.ylabel('Amplitude')
plt.legend()

plt.subplot(3, 1, 2)
plt.plot(time, y_noisy, color='red', alpha=0.7,
    label='Noisy␣Audio')
plt.xlabel('Time␣(s)')
plt.ylabel('Amplitude')
plt.legend()

plt.subplot(3, 1, 3)
plt.plot(time, restored_audio, color='green',
    linestyle='--', label='Restored␣Audio')
plt.xlabel('Time␣(s)')
plt.ylabel('Amplitude')
plt.legend()

plt.tight_layout()
plt.show()
```