# A Fast, Robust Algorithm for Power Line Interference Cancellation in Neural Recording

**Mohammad Reza Keshtkaran and Zhi Yang**

Department of Electrical and Computer Engineering , National University of Singapore, 117583 Singapore

E-mail: keshtkaran@nus.edu.sg

**Abstract.**
*Objective* Power line interference may severely corrupt neural recordings at 50/60 Hz and harmonic frequencies. The interference is usually non-stationary and can vary in frequency, amplitude and phase. To retrieve the gamma-band oscillations at the contaminated frequencies, it is desired to remove the interference without compromising the actual neural signals at the interference frequency bands. In this paper, we present a robust and computationally efficient algorithm for removing power line interference from neural recordings. *Approach* The algorithm includes four steps. First, an adaptive notch filter is used to estimate the fundamental frequency of the interference. Subsequently, based on the estimated frequency, harmonics are generated by using discrete-time oscillators, and then the amplitude and phase of each harmonic are estimated through using a modified recursive least squares algorithm. Finally, the estimated interference is subtracted from the recorded data. *Main results* The algorithm does not require any reference signal, and can track the frequency, phase, and amplitude of each harmonic. When benchmarked with other popular approaches, our algorithm performs better in terms of noise immunity, convergence speed, and output signal-to-noise ratio (SNR). While minimally affecting the signal bands of interest, the algorithm consistently yields fast convergence ($< 100$ ms) and substantial interference rejection (output SNR $> 30$ dB) in different conditions of interference strengths (input SNR from $-30$ dB to 30 dB), power line frequencies (45–65 Hz), and phase and amplitude drifts. In addition, the algorithm features a straightforward parameter adjustment since the parameters are independent of the input SNR, input signal power, and the sampling rate. A prototype was fabricated in a 65-nm CMOS process and tested. The MATLAB implementation of the algorithm has been made available for open access at https://github.com/mrezak/removePLI. *Significance* The proposed algorithm features a highly robust operation, fast adaptation to interference variations, significant SNR improvement, low computational complexity and memory requirement, and straightforward parameter adjustment. These features render the algorithm suitable for wearable and implantable sensor applications, where reliable and real-time cancellation of the interference is desired.

## 1. Introduction

Extracellular neural recordings have made it possible to monitor single-neuron and population activities for studying various cognitive and motor functions. Due to various recording imperfections and experimental protocols, neural recordings are frequently superimposed with interferences and artefacts, which can cause erroneous data analysis. A more common cause of concern is the power line interference which is mainly due to the capacitive coupling between the subject and nearby electrical appliances and mains wiring [1, 2].

While high signal-to-noise ratio (SNR) (i.e. power of the clean neural signal divided by the power of the interference) is preferred for reliable data analysis, the interference pickup can be severe, degrading the SNR to as low as $-20$ dB (the interference is 100 times stronger than the signal). This is especially the case in some experiments where the operation of nearby electrical appliances is unavoidable, and the desired recording isolations cannot be obtained [1–5].

For studying field potentials at lower frequencies (e.g. $< 30$ Hz), a low-pass filter is sufficient to reject the power line interference. However, there is an increasing attention to the gamma band oscillations ($> 30$ Hz) due to their correlation with a wide range of cognitive and sensory processes [2, 6–15]. For example, the frequency bands of 80–500 Hz in [8], 40–180 Hz in [10], 76–150 Hz in [11], 0–200 Hz in [16], and 30–200 Hz in [14] have been shown useful for studying cognitive and motor processing. In this case, in addition to the fundamental harmonic at 50 Hz or 60 Hz, high order harmonics of the interference should also be removed before data analysis.

The interference is usually non-stationary and can vary in frequency, amplitude and phase. The frequency variations are usually small, and mainly originated from the AC power system [17, 18]. Nevertheless, the amplitude and phase variations can be large, which may significantly decrease the SNR of the recorded signal. These variations are mostly due to the subject movements, abrupt changes in nearby AC loads, and changes in capacitive coupling [1, 17, 19]. As a result, automatic cancellation of non-stationary power line interference would be advantageous for reliable data analysis.

A number of solutions are available for reducing the interference pickup. To attenuate the interference at hardware level, biopotential amplifiers are frequently designed to take differential input with large common mode rejection ratio and large isolated-mode rejection ratio. In addition, using active electrodes, shielding electrodes and the subject, and grounding the nearby electrical appliances are useful ways to further reduce the interference [1, 3, 20–22]. Despite these considerations, large residual interference may remain in the signal [1, 2, 5, 23], thus further signal processing is required to completely remove the interference.

Notch filtering has been widely used to attenuate the interference by rejecting its predetermined frequency components (i.e. at 50/60 Hz and harmonic frequencies). To avoid making excessive distortion, the filter should feature narrow notch bandwidth, small phase distortion, and negligible artificial oscillations [19, 24–26]. However, it is difficult to meet these specifications when the interference frequency is not stable and the filter is to accommodate the frequency variations. On the one hand, a very narrow notch may lead to an inadequate removal of the interference, especially when its frequency shifts outside the notch bandwidth. On the other hand, a wide notch can attenuated the interference, but it also results in the excessive removal of information-bearing signal components. These reasons have made notch filtering not a good candidate for power line interference removal in neural recording applications [2, 19].

Other techniques based on spectrum estimation have been used for detecting and removing the spectral peaks (thus the interference) [2]. A drawback is that, these methods require buffering a large number of samples, which slows down the signal processing and is not suitable for real-time implementation. Furthermore, they usually lose their effectiveness when the interference is non-stationary [23, 26].

Another popular approach is to use adaptive interference cancellation which addresses some of the drawbacks of notch filtering. When an auxiliary reference signal of the interference is available, the well-known adaptive noise canceller (ANC) can be utilized to remove the interference [19, 27, 28]. However, it may become ineffective when the interference contains higher order harmonics. Moreover, a reference signal may not always be available in practice. To address these limitations, several reference-free adaptive methods have been proposed, mainly tailored for electrocardiography (ECG) signal processing [23, 24, 26, 29]. Nevertheless, the performance and reliability of these methods have not been tested on neural recordings. In general, several issues might arise when applying the same algorithms to neural recordings. For example, in some algorithms [24, 26], the detection of QRS periods of the ECG signals is necessary to tackle non-stationarity; however, this method is not applicable to neural signals since the on/off period of neural oscillations cannot be easily detected in the presence of the interference. In addition, the power spectral density (PSD) of neural signals follows $1/f^{\alpha}{\scriptstyle (1<\alpha<3)}$ distribution [16, 30, 31] which is different from the that of the ECG; this might lead to inaccurate operation of the interference removal algorithms that are specifically tailored for ECG processing.

This paper proposes an algorithm which can reliably estimate and remove the 50/60 Hz line interference and its harmonics from neural recordings. The algorithm does not require any reference signal, and can track the variations in the frequency, phase, and amplitude of the interference at both the fundamental and the harmonic frequencies. The algorithm can reject the interference, while minimally affecting the signal band of interest, achieving output SNR (i.e. SNR after interference cancellation) of over 30 dB. When applied to neural signals, a performance comparison with two adaptive methods of [23] and [24] is carried out, where the proposed algorithm outperforms in terms of convergence behaviour and output SNR. The low computational complexity, low memory requirement, and adequate numerical behaviour of the algorithm make it suitable for real-time, low-latency hardware implementation. The algorithm is implemented in software as well as an application-specific integrated circuit (ASIC). The software source code and its user manuals are available for open access at [32]. The ASIC was fabricated in a 65-nm CMOS process, and its robust and real-time operation is verified. A preliminary version of this work has been presented in [33].

The rest of this paper is organized as follows. Section 2 details the proposed algorithm, its pseudocode, and parameter adjustment. Section 3 gives the experimental results based on both synthesized and real data, and presents a performance comparison with other methods. Section 4 presents the discussion, and section 5 concludes the paper. The mathematical derivation of the algorithm are given in Appendix A. The ASIC implementation and testing results are briefly described in Appendix B.

## 2. Proposed Algorithm

A recorded neural signal from one electrode can be represented by

$$x(n) = s(n) + p(n), \quad n \in \mathbb{Z}, \tag{1}$$

where $x(n)$ is the measured signal, $s(n)$ is the signal of interest (neural signal + neural noise), and $p(n)$ is the power line interference, all sampled at $f_s$ Hz. $x(n)$ is assumed to be zero-mean, $s(n)$ has a $1/f^{\alpha}{\scriptstyle (1<\alpha<3)}$ power spectrum, and $p(n)$ consists of a set of harmonic sinusoidal components with unknown frequencies, phases and amplitudes as

$$p(n) = \sum_{k=1}^{M} \underbrace{a_k \cos(k\omega_{\mathrm{f}} n + \phi_k)}_{h_k(n)} = \sum_{k=1}^{M} h_k(n). \tag{2}$$

Here, $\omega_{\mathrm{f}}$ is the fundamental frequency in rad/s, $a_k$ and $\phi_k$ are the amplitude and phase of the $k^{\mathrm{th}}$ harmonic, and $M$ is the number of harmonics present in the interference.

An ideal interference cancellation algorithm should eliminate the interference $p(n)$, while perfectly preserving the neural signal $s(n)$. Let $\hat{\omega}_{\mathrm{f}}$, $\hat{a}_k$, $\hat{\phi}_k$, $\hat{h}_k(n)$, and $\hat{p}(n)$ denote the

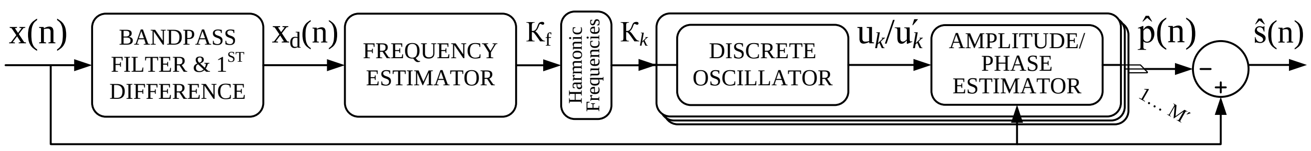

**Figure 1:** Functional block diagram of the proposed algorithm. $x(n)$ is the input signal contaminated by power line interference, $\hat{p}(n)$ is the estimated interference, and $\hat{s}(n)$ is the output interference-free signal.

estimate of $\omega_f$, $a_k$, $\phi_k$, $h_k(n)$, and $p(n)$, respectively. The clean (i.e. interference-free) signal $\hat{s}(n)$ is obtained as

$$\hat{s}(n) = x(n) - \hat{p}(n), \tag{3a}$$

where

$$\hat{p}(n) = \sum_{k=1}^{M'} \hat{h}_k(n). \tag{3b}$$

Here, $M'$ represents the desired number of harmonics to be removed from the recorded signal. It is chosen based on the bandwidth of interest, and its maximum value $M'_{\text{max}} = \lfloor \pi/\hat{\omega}_f \rfloor$ can be adopted if it is desired to remove all the harmonics up to the Nyquist frequency.

The following approach is proposed to cancel the interference. First, the interference fundamental frequency $\omega_f$ is estimated by using a fast and numerically well-behaved frequency estimator. Subsequently, based on the estimated frequency $\hat{\omega}_f$, each harmonic signal $h_k(n)$ is obtained by using discrete-time oscillators and then its amplitude and phase (i.e. $\hat{a}_k$ and $\hat{\phi}_k$, respectively) are estimated by using a simplified recursive least squares (RLS) algorithm. The cascaded stages of frequency and amplitude/phase estimation allow individually adjustable adaptation rates for each of these estimators, which helps to achieve a fast and reliable estimation of the interference. Finally, the estimated interference $\hat{p}(n)$ is subtracted from the input signal $x(n)$ to obtain the clean signal $\hat{s}(n)$. The structure of the proposed algorithm is shown in figure 1.

### *2.1. Fundamental Frequency Estimation*

For robust estimation of the fundamental frequency, first, the signal is preprocessed to enhance the fundamental harmonic of the interference. After that, the enhanced signal is used for frequency estimation. The preprocessing stage is described in section 2.1.1, followed by the frequency estimation stage in section 2.1.2.

*2.1.1. Preprocessing: Initial Band-pass Filtering and Spectrum Shaping* Since the input signal $x(n)$ has a coloured PSD $(1/f)$, the direct application of a typical adaptive frequency estimator would lead to a biased estimation of the frequency [34]. It is also possible that the inference $p(n)$ is weak at its fundamental frequency and more dominant at certain harmonic frequencies, especially with the use of a differential recorder that can largely attenuate odd-order harmonics. This may prevent the frequency estimator from converging to a correct frequency estimate. To address these issues and improve the frequency estimation, the input signal $x(n)$ is bandpass filtered with a 4th-order infinite-impulse-response (IIR) filter to enhance the fundamental harmonic of the interference and attenuate higher harmonics. This filtering is also useful for attenuating lower frequency artefacts and signal components which may negatively affect the frequency estimation. The filter passband

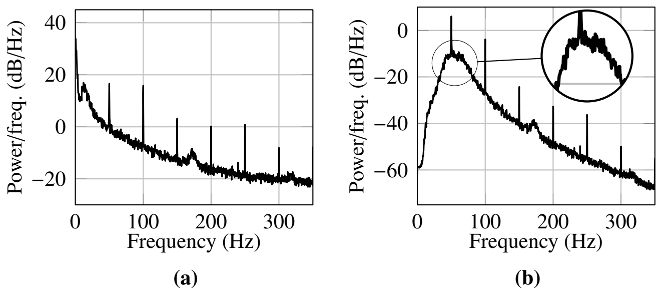

**Figure 2:** The effect of bandpass filtering and spectrum shaping. (a) PSD of a real ECoG signal. (b) PSD after bandpass filtering and spectrum shaping, where the fundamental harmonic is enhanced.

is by default set to 40–70 Hz to accommodate both 50 Hz and 60 Hz power line frequencies and their worst case variations, but it can be further customized; for example, to 55–65 Hz if the nominal power line frequency is known to be 60 Hz. Let $H(\cdot)$ be the realization of the bandpass filter, the filtered signal $x_f(n)$ is obtained as

$$x_f(n) = H(x(n)). \tag{4a}$$

To further reduce the estimation bias, a 1st-order differentiator is utilized which mitigates the effect of the power law spectrum of the input signal:

$$x_d(n) = x_f(n) - x_f(n-1). \tag{4b}$$

Here, $x_d(n)$ is the first difference signal fed into the next stage ANF for frequency estimation. The effect of 1st-order differentiation on the overall performance of the algorithm is not very significant; however, in practice, the first order differentiator can be incorporated into the bandpass filter with negligible computational overhead. Figure 2 shows the effects of bandpass filtering and spectrum shaping, where the fundamental harmonic of the interference is enhanced. It should be noted that signal $x_d$ is only used for frequency estimation, and not for amplitude/phase estimation.

*2.1.2. Frequency Estimation* The estimation of the instantaneous frequency of a single sinusoid buried in broadband noise has been largely investigated in the literature. Various well-established methods exist for frequency estimation differing in performance with regard to computational complexity, and estimation bias and variance [35–39]. In this work, a lattice adaptive notch filter (ANF)-based frequency estimator is utilized since it features instantaneous estimation of the frequency, desirable performance, low complexity, and suitability for real-time finite-precision implementation.

It should be noted that the ANF is merely used for frequency estimation and not for notch filtering, hence the all-zero section need not be used. Figure 3a shows the structure of the ANF where $x_d(n)$ is the input signal from the preprocessing stage and $f(n)$ is the output of the all-pole

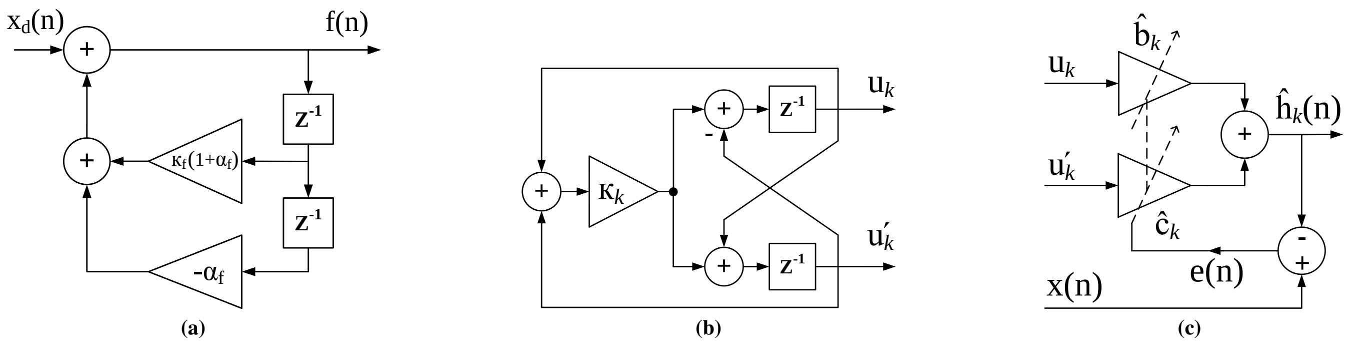

**Figure 3:** Signal flow graph of (a) all-pole lattice ANF structure. Notch frequency and bandwidth are determined by $\kappa_f$ and $\alpha_f$, respectively. (b) Discrete-time oscillator. The parameter $\kappa_k$ adjusts the oscillation frequency. $u_k$ and $u'_k$ represent orthogonal sinusoids at frequency $k\omega_f$. (c) Adaptive linear combiner, used for amplitude/phase adaptation. The weights $a_k$ and $b_k$ are adapted by the simplified RLS algorithm which minimizes the weighted least square error between $x(n)$ and $\hat{h}_k(n)$.

section. The transfer function of the all-pole section is given by

$$H(z) = \frac{1}{1 - \kappa_f(n)(\alpha_f + 1)z^{-1} + \alpha_f z^{-2}}, \tag{5}$$

where $\kappa_f(n)$ is the adaptive coefficient at time step $n$, which gives the frequency estimate $\hat{\omega}_f(n)$ through $\hat{\omega}_f(n) = \cos^{-1}\kappa_f(n)$, and $0 < \alpha_f < 1$ is the pole radii and determines the notch bandwidth. The lattice algorithm of [38] is employed to adjust $\kappa_f$ as follows.

$$c(0) = d(0) = \epsilon > 0,\ f(-1) = f(-2) = 0,\ \kappa_f(0) = 0, \tag{6a}$$
$$c(n) = \lambda_f c(n-1) + f(n-1)(f(n) + f(n-2)), \tag{6b}$$
$$d(n) = \lambda_f d(n-1) + 2f(n-1)^2, \tag{6c}$$
$$\kappa_t(n) = \frac{c(n)}{d(n)}, \tag{6d}$$
$$\kappa_t(n) = \begin{cases} \kappa_t(n) & \text{if } -1 < \kappa_t(n) < 1, \\ 1 & \text{if } \kappa_t(n) > 1, \\ -1 & \text{if } \kappa_t(n) < -1, \end{cases} \tag{6e}$$
$$\kappa_f(n) = \gamma\kappa_f(n-1) + (1-\gamma)\kappa_t(n), \tag{6f}$$

where $0 \ll \lambda_f < 1$ is the forgetting factor, $f(n)$ is the output of the all-pole section, $\kappa_f(n)$ is the estimated parameter ($\hat{\omega}_f(n) = \cos^{-1}\kappa_f(n)$), and $\gamma$ is the smoothing factor. Equation (6a) sets the initial condition, (6b)–(6d) form the frequency estimator, and (6e) is used to limit $\kappa_t$ in the range of $[-1, 1]$ to guarantee stability. (6f) is used to further smooth $\kappa_f(n)$. For simplicity in notation, in the rest of this paper, $\kappa_f$ is short for $\kappa_f(n)$ and $\hat{\omega}_f$ is short for $\hat{\omega}_f(n)$.

The parameters $\alpha_f$ and $\lambda_f$ control the speed and accuracy of frequency estimation. It is advantageous to use time-varying values for $\alpha_f$ and $\lambda_f$ due to several reasons. In initial convergence, if the notch is too narrow ($\alpha_f$ very close to 1), the ANF may not sense the presence of the input sinusoid, which in turn leads to a very slow initial convergence or even not converging to the correct frequency estimate. Similarly, an initial value of $\lambda_f$ very close to 1, significantly slows down the initial adaptation. On the other hand, smaller values of $\alpha_f$ and $\lambda_f$ increase the steady-state error. A solution is to start the algorithm with smaller values of $\alpha_f$ and $\lambda_f$ to reach a fast convergence, and after that gradually increase their values to obtain more accurate frequency estimation. For this purpose, $\alpha_f$ and $\lambda_f$ are updated in each iteration as

$$\alpha_f(n) = \alpha_{st}\alpha_f(n-1) + (1-\alpha_{st})\alpha_\infty, \tag{7a}$$
$$\lambda_f(n) = \lambda_{st}\lambda_f(n-1) + (1-\lambda_{st})\lambda_\infty, \tag{7b}$$

where $\alpha_\infty$ determines the asymptotic notch bandwidth and $\alpha_{st}$ sets the rate of change from the initial value $\alpha_f(0) = \alpha_0$ to the asymptotic value $\alpha_\infty$. Similarly, $\lambda_\infty$ determines the asymptotic forgetting factor and $\lambda_{st}$ sets the rate of change from initial value $\lambda_f(0) = \lambda_0$ to the asymptotic value $\lambda_\infty$. Detailed discussion on choosing proper values for the parameters are presented in section 2.4

### 2.2. Harmonic Estimation

Having estimated $\kappa_f$, the algorithm proceeds to estimate the harmonic components. Harmonic estimation comprises two sub-stages. First, a series of harmonic sinusoids with fundamental frequency $\hat{\omega}_f$ are generated. Subsequently, the amplitudes and the phases of the generated harmonics are estimated to match their corresponding components in the interference. Harmonic generation is explained in section 2.2.1, and amplitude/phase estimation is described in section 2.2.2.

*2.2.1. Harmonic Signal Generation* The harmonic sinusoids are generated through using discrete-time oscillators, which require less computation compared with the Taylor expansion method [40]. Among different oscillator structures, a digital waveguide oscillator is chosen (figure 3b). This structure provides orthogonal outputs, which are exploited to simplify the next stage RLS algorithm. More importantly, the oscillator output frequency can be directly controlled by $\cos k\hat{\omega}_f$, where $k\hat{\omega}_f$ is the oscillation frequency. This enables the output of the frequency estimator $\kappa_f$ to be directly employed for harmonic generation, thus avoiding the calculation of computationally expensive trigonometric functions. To further reduce the complexity, the frequency estimates of higher harmonics are obtained through the recurrence formulation in (8), which also avoid trigonometric function calculation. For each harmonic $k$, the frequency control parameter of the oscillator is denoted as $\kappa_k = \cos k\hat{\omega}_f$, and is recursively calculated through

$$\kappa_k = 2\kappa_1\kappa_{k-1} - \kappa_{k-2}, \quad \text{for } k = 2, 3, \cdots, M', \tag{8a}$$

where

$$\kappa_0 = 1, \kappa_1 = \kappa_f = \cos\hat{\omega}_f. \tag{8b}$$

The calculated parameter $\kappa_k$ is used to set the oscillation frequency of the oscillator.

Figure 3b shows the signal flow graph of the digital waveguide oscillator, which is represented by (9a).

$$\begin{bmatrix} u_k(n) \\ u'_k(n) \end{bmatrix} = \begin{bmatrix} \kappa_k & \kappa_k - 1 \\ \kappa_k + 1 & \kappa_k \end{bmatrix} \begin{bmatrix} u_k(n-1) \\ u'_k(n-1) \end{bmatrix}, \tag{9a}$$

$$G = 1.5 - \left( u_k(n)^2 - \frac{\kappa_k - 1}{\kappa_k + 1} u'_k(n)^2 \right), \tag{9b}$$

$$u_k(n) = G u_k(n), \quad u'_k(n) = G u'_k(n). \tag{9c}$$

Here, $u(n)$ and $u'(n)$ are state variables serving as sinusoidal outputs. The values of $u_k(0)$ and $u'_k(0)$ determine the initial phase and amplitude, which are arbitrarily chosen. (9b) and (9c) are used to apply gain control for stabilizing oscillation amplitude in dynamic frequency operation. The output of (9) can be generally expressed as

$$\begin{aligned} u_k(n) &= v_k \sin(k\hat{\omega}_{\mathrm{f}} n + \psi_k), \\ u'_k(n) &= v'_k \cos(k\hat{\omega}_{\mathrm{f}} n + \psi_k), \end{aligned} \tag{10}$$

where $v_k$ and $v'_k$ are the amplitudes of the generated sinusoids, and $\psi_k$ is the initial phase shift. The values of $\psi_k$s do not influence any further derivations and are neglected for simplicity.

*2.2.2. Amplitude and Phase Estimation* The amplitudes and phases (i.e. $\hat{a}_k$ and $\hat{\phi}_k$) of the generated harmonics are not necessarily the same with their corresponding power line interference components in (2); thus, an additional step is required to estimate them. The estimate of the $k^{\text{th}}$ harmonic, $\hat{h}_k(n)$, can be obtained via (2) by substituting $a_k$ and $\phi_k$ with their estimates that gives

$$\hat{h}_k(n) = \hat{a}_k \sin(k\hat{\omega}_{\mathrm{f}} n + \hat{\phi}_k) \tag{11a}$$

$$= \hat{b}'_k \sin(k\hat{\omega}_{\mathrm{f}} n) + \hat{c}'_k \cos(k\hat{\omega}_{\mathrm{f}} n). \tag{11b}$$

where

$$\hat{b}'_k = \hat{a}_k \cos \hat{\phi}_k, \quad \text{and} \quad \hat{c}'_k = \hat{a}_k \sin \hat{\phi}_k.$$

Here, instead of directly adapting $\hat{a}_k$ and $\hat{\phi}_k$ in (11a) we can equivalently adapt $\hat{b}'_k$ and $\hat{c}'_k$ in (11b) to obtain $\hat{h}_k(n)$. This transformation converts the non-convex search space in $\hat{a}_k$-$\hat{\phi}_k$ coordinates into a convex search space in rectangular coordinates. Using (10) and (11b), $\hat{h}_k(n)$ can be written as

$$\hat{h}_k(n) = \hat{b}_k u_k(n) + \hat{c}_k u'_k(n). \tag{12}$$

Here, $\hat{b}_k$ and $\hat{c}_k$ are defined as $\hat{b}'_k / v_k$ and $\hat{c}'_k / v'_k$, where $v_k$ and $v'_k$ merely scale the adaptive coefficients and do not affect the estimation performance. For each harmonic $k$, $\hat{b}_k$ and $\hat{c}_k$ are adapted by minimizing the exponentially weighted squared error between $\hat{h}_k(n)$ and $x(n)$. This is done by applying the simplified RLS algorithm, where $u_k(n)$ and $u'_k(n)$ serve as the input to an adaptive linear combiner (figure 3c). The following update equations are used to adapt $\hat{b}_k$ and $\hat{c}_k$.

$$\begin{aligned} r_{1,k}(-1) &= r_{1,k}(-1) = \hat{b}_k(-1) = \hat{c}_k(-1) = 0, \\ r_{1,k}(n) &= \lambda_{\mathrm{a}} r_{1,k}(n-1) + u_k(n)^2, \\ r_{4,k}(n) &= \lambda_{\mathrm{a}} r_{4,k}(n-1) + u'_k(n)^2, \\ \hat{b}_k(n) &= \hat{b}_k(n-1) + u_k(n) e_k(n) / r_{1,k}(n), \\ \hat{c}_k(n) &= \hat{c}_k(n-1) + u'_k(n) e_k(n) / r_{4,k}(n), \end{aligned} \tag{13}$$

where $e_k(n) = x(n) - \hat{h}_k(n)$ is the instantaneous error (figure 3c), and $0 \ll \lambda_{\mathrm{a}} < 1$ is the forgetting factor. A detailed description of the simplified RLS algorithm is described in Appendix A.1.

In each iteration, the most recent estimates $\hat{b}_k(n)$ and $\hat{c}_k(n)$ are used to obtain $\hat{h}_k(n)$ through (12). The interference-free neural signal is then obtained by

$$\hat{s}(n) = x(n) - \sum_{k=1}^{M'} \hat{h}_k(n). \tag{14}$$

**Algorithm 1:** Proposed Algorithm

```
Input: $x$
Output: $\hat{s}$
Constants:
$f_{\mathrm{s}}, M', N, \alpha_0, \alpha_{\mathrm{st}}, \alpha_\infty, \lambda_0, \lambda_{\mathrm{st}}, \lambda_\infty, \lambda_{\mathrm{a}}, \gamma$
$H(\cdot) \leftarrow$ 40–70 Hz IIR filter
Initialization:
$\kappa_0 \leftarrow 1, \kappa_{\mathrm{f}} \leftarrow 0$
$f_{-2} \leftarrow f_{-1} \leftarrow 0$
$c, d > 0$
$u_k, u'_k > 0$
$r_{1,k}, r_{4,k} > 0$
$\hat{b}_k \leftarrow \hat{c}_k \leftarrow 0$
$\alpha_{\mathrm{f}} \leftarrow \alpha_0, \lambda_{\mathrm{f}} \leftarrow \lambda_0$
Recursion:
for $n \leftarrow 1$ to $N$ do
    Bandpass filtering:
      $x_f \leftarrow H(x(n))$
    Frequency Estimation:
      $f_n \leftarrow x_f + \kappa_{\mathrm{f}}(1+\alpha_{\mathrm{f}}) f_{n-1} - \alpha_{\mathrm{f}} f_{n-2}$
      $c \leftarrow \lambda_{\mathrm{f}} c + f_{n-1}(f_n + f_{n-2})$
      $d \leftarrow \lambda_{\mathrm{f}} d + 2 f_{n-1}^2$
      $\kappa_{\mathrm{t}} \leftarrow c/d$
      if $\kappa_{\mathrm{t}} > 1$ then $\kappa_{\mathrm{t}} \leftarrow 1$   else if $\kappa_{\mathrm{t}} < -1$ then $\kappa_{\mathrm{t}} \leftarrow -1$
    $\kappa_{\mathrm{f}} \leftarrow \gamma \kappa_{\mathrm{f}} + (1-\gamma)\kappa_{\mathrm{t}}$
      $\alpha_{\mathrm{f}} \leftarrow \alpha_{\mathrm{st}} \alpha_{\mathrm{f}} + (1-\alpha_{\mathrm{st}})\alpha_\infty$
      $\lambda_{\mathrm{f}} \leftarrow \lambda_{\mathrm{st}} \lambda_{\mathrm{f}} + (1-\lambda_{\mathrm{st}})\lambda_\infty$
    Removing Harmonics:
      $\kappa_1 \leftarrow \kappa_{\mathrm{f}}$
      $e \leftarrow x(n)$
      for $k \leftarrow 1$ to $M'$ do
          Discrete Oscillator:
            $s_1 \leftarrow \kappa_k(u_k + u'_k)$
            $s_2 \leftarrow u_k$
            $u_k \leftarrow s_1 - u'_k$
            $u'_k \leftarrow s_1 + s_2$
            $G \leftarrow 1.5 - [u_k^2 - u_k'^2(\kappa_k - 1)/(\kappa_k + 1)]$
            if $G < 0$ then $G \leftarrow 1$   $u_k = G u_k, u'_k = G u'_k$
          Amplitude/Phase Estimation:
            $h_k \leftarrow (\hat{b}_k u_k + \hat{c}_k u'_k)$
            $e \leftarrow e - h_k$
            $r_{1,k} \leftarrow \lambda_{\mathrm{a}} r_{1,k} + u_k^2$
            $r_{4,k} \leftarrow \lambda_{\mathrm{a}} r_{4,k} + u_k'^2$
            $\hat{b}_k \leftarrow \hat{b}_k + e \cdot u_k / r_{1,k}$
            $\hat{c}_k \leftarrow \hat{c}_k + e \cdot u'_k / r_{4,k}$
          Harmonic Frequency Calculation:
            $\kappa_{k+1} \leftarrow 2\kappa_{\mathrm{f}} \kappa_k - \kappa_{k-1}$
      $\hat{s}(n) \leftarrow e$
```

### 2.3. Algorithm Implementation

The algorithm is implemented in software as well as an ASIC. The pseudocode of the algorithm is presented in algorithm 1 with the MATLAB source code available online at [32]. An explanatory list of the symbols and parameters is shown in

**Table 1:** List of symbols and parameters

| Symbol | Explanation |
|---|---|
| $N$ | Number of samples |
| $f_s$ | Sampling rate (Hz) |
| $M'$ | Number of harmonics to remove |
| $B_0$ | Initial notch bandwidth of the frequency estimator (Hz) |
| $B_\infty$ | Asymptotic notch bandwidth of the frequency estimator (Hz) |
| $B_{st}$ | Settling time from $B_0$ to $B_\infty$ (s) |
| $\alpha_f$ | Pole radii of the adaptive notch filter (ANF) |
| $\alpha_0$ | Initial pole radii of the ANF |
| $\alpha_\infty$ | Asymptotic pole radii of the ANF |
| $\alpha_{st}$ | Rate of change from $\alpha_0$ to $\alpha_\infty$ |
| $P_0$ | Initial settling time of the frequency estimator (s) |
| $P_\infty$ | Asymptotic settling time of the frequency estimator (s) |
| $P_{st}$ | Settling time from $P_0$ to $P_\infty$ (s) |
| $\lambda_f$ | Forgetting factor of the frequency estimator |
| $\lambda_0$ | Initial forgetting factor of the frequency estimator |
| $\lambda_\infty$ | Asymptotic forgetting factor of the frequency estimator |
| $\lambda_{st}$ | Rate of change from $\lambda_0$ to $\lambda_\infty$ |
| $\gamma$ | Smoothing parameter of the frequency estimator |
| $\gamma'$ | Cut-off frequency of the smoothing filter; set at 90 Hz |
| $W$ | Settling time of amplitude/phase estimator (s) |
| $\lambda_a$ | Forgetting factor of the amplitude/phase estimator |
| $H(\cdot)$ | 40–70 Hz 4th order IIR bandpass filter |

table 1, and the proper parameter values can be obtained through the guidelines in section 2.4.

The ASIC was fabricated in a 65-nm CMOS process, and consumes 0.11 mm$^2$ of silicon area. It was tested against a reference model, where its robust and real-time operation was experimentally verified. For validating the chip output, we used a full-precision MATLAB implementation of the algorithm with the same structure and parameter values used in the chip design. This implementation is referred to as 'reference model' in the rest of this paper. Further discussion of hardware implementation and testing results are presented in Appendix B.

### 2.4. Parameter Setting

The performance of the algorithm is mainly controlled by three basic parameters including notch filter pole radii ($\alpha_f$), frequency estimator's forgetting factor ($\lambda_f$) and amplitude/phase estimator's forgetting factor ($\lambda_a$). Since, the proper values of these parameters depend on the signal sampling rate ($f_s$), the parameter adjustment become less intuitive. To alleviate this issue, we have chosen other representative characteristics such as notch bandwidth (related to the pole radii) and settling time (related to the forgetting factors) which can be alternatively used for parameter adjustment. The alternative parameters are displayed in (15).

$$\begin{aligned}\mathcal{A}_1 &= \{\alpha_{st}, \lambda_0, \lambda_\infty, \lambda_{st}, \lambda_a\},\\ \mathcal{B}_1 &= \{B_{st}, P_0, P_\infty, P_{st}, W\},\\ \mathcal{A}_2 &= \{\alpha_0, \alpha_\infty, \gamma'\},\\ \mathcal{B}_2 &= \{B_0, B_\infty, \gamma/2\},\end{aligned} \tag{15a}$$

$$\begin{aligned}\mathcal{A}_1 &= \exp\frac{\ln(0.05)}{\mathcal{B}_1 f_s + 1},\\ \mathcal{A}_2 &= \frac{1-\tan(\pi\mathcal{B}_2/f_s)}{1+\tan(\pi\mathcal{B}_2/f_s)}.\end{aligned} \tag{15b}$$

Here, $\mathcal{B}_1$ and $\mathcal{B}_2$ contain the alternative parameters which are independent of the sampling rate and have intuitive units. The actual parameters, defined in $\mathcal{A}_1$ and $\mathcal{A}_2$, can be obtained through (15b). It should be noted that improper parameter setting may lead to inadequate removal of the interference. Some guidelines on the proper adjustment of the parameters are discussed as follows.

The notch bandwidth of the frequency estimator affects both the tracking speed and the estimation bias. A wide notch allows faster tracking of the frequency at the expense of an increased estimation bias and variance. On the other hand, a narrow notch leads to a more accurate frequency estimate, but it causes very slow frequency adaptation if the desired sinusoidal component falls out of the notch bandwidth. To address this trade-off, the notch bandwidth is initially widened to allow fast initial convergence and then gradually narrowed down to achieve a lower steady-state error (described in (7a)). In the alternative form, $B_0$ is associated with $\alpha_0$ and controls the initial notch bandwidth. Larger values of $B_0$ are preferred (e.g. tens of Hz) to achieve a faster initial convergence. Similarly, $B_\infty$ is associated with $\alpha_\infty$ and controls the asymptotic notch bandwidth. Small values of $B_\infty$ are preferred (e.g. tenths of Hz) to achieve more accurate estimation of the frequency. $B_{st}$ controls the rate of transition between initial notch bandwidth $B_0$ and the asymptotic notch bandwidth $B_\infty$, and indicates the time in seconds, in which $\alpha_f$ reaches $0.95\alpha_\infty$ in (7a). When the algorithm is used to remove a large number of harmonic components, $B_\infty$ should be set small enough to minimize the bias in the frequency estimates of higher harmonics. For example, if it is desired to remove harmonics up to 100th order, setting $B_\infty = 0.001$ would be an adequate choice. In this case, although small values of $B_\infty$ lead to slow frequency adaptation, it would not be problematic, since in practice, the drifts in the power line frequency are usually slow and the algorithm can still reasonably track the variations.

The forgetting factor of the frequency estimator $\lambda_f$ is initially small to achieve a fast convergence and is gradually increased to achieve a more accurate estimate (described in (7b)). In the alternative form, $P_0$ is associated with $\lambda_0$ and controls the initial settling time of the frequency estimation algorithm. Smaller values of $P_0$ are preferred (e.g. tenths of seconds) to achieve a faster initial convergence. Similarly, $P_\infty$ is associated with $\lambda_\infty$ and controls the asymptotic settling time of the frequency estimation algorithm. Considering the fact that the power line frequency drifts are slow, larger values of $P_\infty$ are preferred (e.g. a few seconds) to obtain a more accurate estimation of the power line frequency. $P_{st}$ controls how fast the settling time changes from the initial value of $P_0$ to its final value of $P_\infty$ and indicates the time in seconds, in which $\lambda_f$ reaches $0.95\lambda_\infty$ in (7b). This transition time should be set large enough (e.g. a few seconds depending on the notch bandwidth) to allow global convergence.

The settling time of the amplitude/phase estimator ($W$) controls how fast it responds to the fluctuations in the amplitudes and phases of the harmonics. In the alternative form, $W$ is associated with $\lambda_a$, and indicates the time in

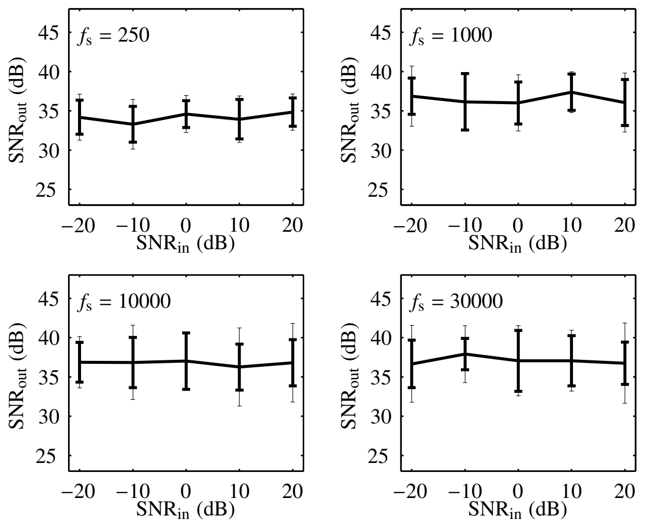

**Figure 4:** SNR$_{out}$ vs. SNR$_{in}$. Each figure is obtained at a different sampling rate. The horizontal plots indicate the mean, the thick bars show the standard deviation and the thin bars indicate minimum and maximum SNR$_{out}$ values over 50 runs on real ECoG signals with synthetic interference containing 3 harmonics at 61 Hz, 122 Hz, and 183 Hz (2 harmonics for $f_s$=250 Hz). Consistent high values of SNR$_{out}$ indicate the robust operation of the algorithm with regard to different SNR$_{in}$ and sampling rates. Parameter setting: $\{B_0 = 50, B_{st} = 1, B_\infty = 0.1, P_0 = 0.1, P_{st} = 1, P_\infty = 2, W = 2\}$.

which the estimates of amplitude and phase reach 95% of their asymptotic values. The interference frequency bands (e.g. near 50/60 Hz and multiples) contain both the interference components as well as useful neural signals which should be preserved. For this purpose, $W$ should be selected reasonably large to obtain an accurate estimation of the interference, thus avoiding the excessive removal of neural signals, while small enough to allow tracking of the interference amplitude fluctuations. Depending on the recording environment and subject movements, $W$ may be selected from a few tenths of seconds to a few seconds. A recommended set of parameter values are suggested in table 2 which could be initially used for further tuning.

**Table 2:** Recommended values of parameters

| Parameter | Recommended Range |
|---|---|
| $B_0$ (Hz) | 10 – 50 |
| $B_\infty$ (Hz) | 0.01 – 0.1 |
| $B_{st}$ (s) | 0.5 – 10 |
| $P_0$ (s) | 0.01 – 0.5 |
| $P_\infty$ (s) | 1 – 5 |
| $P_{st}$ (s) | 1 – 10 |
| $W$ (s) | 0.5 – 5 |

## 3. Results

Extensive simulations are carried out to quantitatively evaluate the performance of the proposed algorithm under various signal and parameters conditions. The algorithm performance is also compared with other popular interference removal methods. Furthermore, the algorithm is tested on extracellular, electrocorticography (ECoG) and electroencephalography

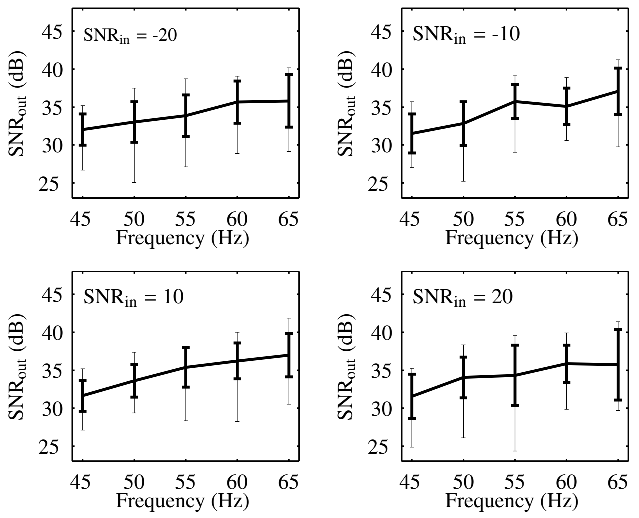

**Figure 5:** SNR$_{out}$ vs. power line frequency. The horizontal plots indicate the means, the thick bars show the standard deviation and the thin bars indicate minimum and maximum SNR$_{out}$ values over 50 runs on real ECoG signals with synthetic interference containing 3 harmonics. Consistent high values of SNR$_{out}$ are achieved in the wide range of power line frequencies and sampling rates. The reason for the slight increase of mean SNR$_{out}$ with frequency is mainly due to the $1/f$ PSD of neural signals. With a fixed SNR$_{in}$, at higher frequencies, the power of neural signals are less, leading to a more accurate estimation of the interference (neural signals are seen as noise to the interference estimation algorithm), hence resulting in a better cancellation and slightly higher SNR$_{out}$ compared with the lower frequencies. Parameter setting is the same as that of figure 4.

(EEG) recordings to illustrate its performance on real neural data. The results of the performance evaluation using synthetic data are described in section 3.1, the performance comparison results are reported in section 3.2, and the results on real data are presented in section 3.3. In case the reader wishes to reproduce the paper's results, the parameter setting in each simulation is provided.

### *3.1. Performance Evaluation on Synthetic Data*

Synthetic data are used to quantitatively evaluate the important characteristics of the proposed algorithm under various signal conditions. Each test sequence was synthesized by adding a synthetic interference containing 3 harmonics, to a random portion of real ECoG and extracellular recordings that were recorded in a controlled condition with negligible amount of power line interference. The frequency and power of the interference components are specified in each simulation.

In the rest of this paper, SNR$_{in}$ and SNR$_{out}$ are used to denote the SNRs of the algorithm input and output signals, i.e. $x(n)$ and $\hat{s}(n)$, respectively. It should be noted that, SNR$_{out}$ values are calculated after the algorithm reaches its steady-state, unless otherwise stated.

*3.1.1. Sensitivity to SNR$_{in}$* The variations in the power of the picked-up interference are usually significant, leading to different SNR$_{in}$ values from as low as −20 dB (severe interference), to as high as 30 dB (negligible interference). To ensure proper interference cancellation, the algorithm is desired to work reliably under various SNR$_{in}$ conditions. To evaluate this aspect, we generated synthetic sequences whose SNR$_{in}$ ranged from −20 dB to 20 dB. For each SNR$_{in}$ value, 50 sequences were generated, the algorithm was applied to cancel the interference, and the resultant SNR$_{\text{out}}$s were recorded. In addition the simulation was repeated with different sampling rates for reliability resting.

Figure 4 shows the mean, variance, minimum and maximum of the resultant SNR$_{\text{out}}$ for each SNR$_{\text{in}}$ condition and sampling rate. It can be seen that, consistent high values of SNR$_{\text{out}}$ are observed in all the conditions, indicating that the performance of the algorithm is highly insensitive to SNR$_{\text{in}}$.

*3.1.2. Sensitivity to Power Line Frequency* Since the accurate value of power line frequency is a priori unknown, and may also change over time [17, 18], it is important to test the performance of the algorithm with regard to different power line frequencies. For this purpose, synthetic sequences with fundamental frequencies ranging between 45 Hz to 65 Hz were used as the input to the algorithm, and output SNRs were measured to test the performance. This frequency range covers the worst case power line frequency deviations [17, 18]. As can be seen in figure 5, high values of SNR$_{\text{out}}$ (> 30 dB) were consistently achieved for different power line frequencies in all the SNR$_{\text{in}}$ conditions. The results demonstrate the robust operation of the algorithm even in worst case power line frequency deviations. Furthermore, it can be seen that the algorithm can automatically detect the interference at 50 Hz or 60 Hz, and no a priori setting of the nominal power line frequency is required.

*3.1.3. Trade-off between Settling Time and SNR$_{out}$* As discussed in section 2.4, there is a trade-off between SNR$_{\text{out}}$ and the amplitude settling time ($W$). To track the abrupt changes in the interference power, fast settling time is desired. Typically, when $W$ is set small to have a fast tracking response, the SNR$_{\text{out}}$ would decrease. On the other hand, when $W$ is set large to achieve a higher SNR$_{\text{out}}$, then the settling time will increase. It is desirable to achieve a high SNR$_{\text{out}}$ along with a reasonably fast settling time. Figure 6 shows the average values of SNR$_{\text{out}}$ versus different settling time values ($W$). It can be seen that high values of SNR$_{\text{out}}$ ($\approx 30$ dB) can be obtained with a reasonably low settling time (< 1 s).

*3.1.4. Tracking of Amplitude and Frequency Fluctuations* The drifts of the power line frequency are typically small, while the fluctuations of the harmonics amplitudes can be quite large [17, 18]. In order to effectively reject the power line interference, the algorithm should be able to adequately track the frequency and amplitude variations.

To illustrate the amplitude tracking performance, the harmonic amplitudes were increased to twice their initial values and the algorithm was applied with different settling time values ($W$). Figure 7 displays the first three interference harmonics, where the they underwent a ramp change and a step change. It can be seen that, the estimates of the amplitudes properly tracked the actual values.

To illustrate the frequency tracking performance, two simulations were done. In the first simulation, the fundamental frequency of the synthetic harmonics was swept from 59 Hz to 61 Hz. It can be seen in figures 8a and 8c that, for all the parameter conditions, the frequency estimates accurately track the actual values. In the second simulation, the fundamental frequency underwent a step change from 50 Hz to 60 Hz. Figures 8b and 8d show that, the frequency estimate converges to the actual frequency with different settling times which depend on the parameters $B_\infty$ and $P_\infty$. It should be noted that, due to the use of time-varying parameters in (7a), the initial convergence is much faster than is in the operating condition.

*3.1.5. Initial Convergence* To illustrate the convergence behaviour of the algorithm, two synthetic sequences with interference fundamental frequency of 50 Hz and 60 Hz were used. The interference contained 3 harmonics. Figure 9 shows the frequency convergence, where the frequency estimates converged to the actual frequencies (i.e. 50 Hz and 60 Hz) in less than 100 ms, while maintaining a high SNR$_{\text{out}}$. This fast convergence speed is mainly due to adopting time-varying $\alpha_{\text{f}}$ and $\lambda_{\text{f}}$. In other words, the initial convergence is controlled by the parameters $B_0$, $B_{\text{st}}$, $P_0$ and $P_{\text{st}}$, whereas the parameters $B_\infty$, $P_\infty$ and $W$ determine SNR$_{\text{out}}$. The convergence of the three estimated harmonics is displayed in figure 10, where a quick (< 100 ms) convergence to actual harmonics is observed.

## 3.2. Comparison with Other Methods

The algorithm is compared with narrow- and wide-band notch filtering, and two adaptive algorithms proposed by Ziarani et al. [23] and Martens et al. [24].

A performance comparison in terms of SNR improvement, mean square error (MSE) and convergence speed is presented in section 3.2.1. A comparison between the effects of different interference removal methods on synthetic neural oscillations is made in section 3.2.2.

*3.2.1. Performance Comparison* Figure 11 shows the effect of wide- and narrow-band notch filtering on a synthetically corrupted ECoG signal. The interference fundamental frequency was slightly deviated from 60 Hz which translated to even higher deviations in higher harmonics (figure 11b). As can be seen in figure 11c, narrow-band notch filters fail to adequately remove the interference with changing frequency. On the other hand, wide-band notch filters distort the signal PSD (figure 11d). The result of interference cancellation using the proposed method is displayed in figure 11e. It can be seen that, the interference is adequately removed while the signal frequency bands are highly preserved.

The adaptive methods of Ziarani et al. [23] and Martens et al. [24] have been widely applied to electrocardiography (ECG) signals and shown effective in removing non-stationary power line interference. Here, we compare the convergence behaviour and the asymptotic performances of these methods against the proposed algorithm.

The first simulation is done to evaluate the asymptotic performances of the algorithms in terms of SNR$_{\text{out}}$ versus SNR$_{\text{in}}$. For this purpose, randomly selected portions of an interference-free ECoG recording were used and each of which was superimposed with interference containing a single stationary sinusoid at $f_{\text{I}} = 59$ Hz with a random phase and a determined amplitude. The algorithms were allowed to fully converge to their steady states and the values of SNR$_{\text{out}}$ were calculated for t > 60 s. As can be seen in figure 12, for all SNR$_{\text{in}}$ values, the proposed algorithm achieves significantly higher SNR$_{\text{out}}$ compared with other methods. Furthermore,

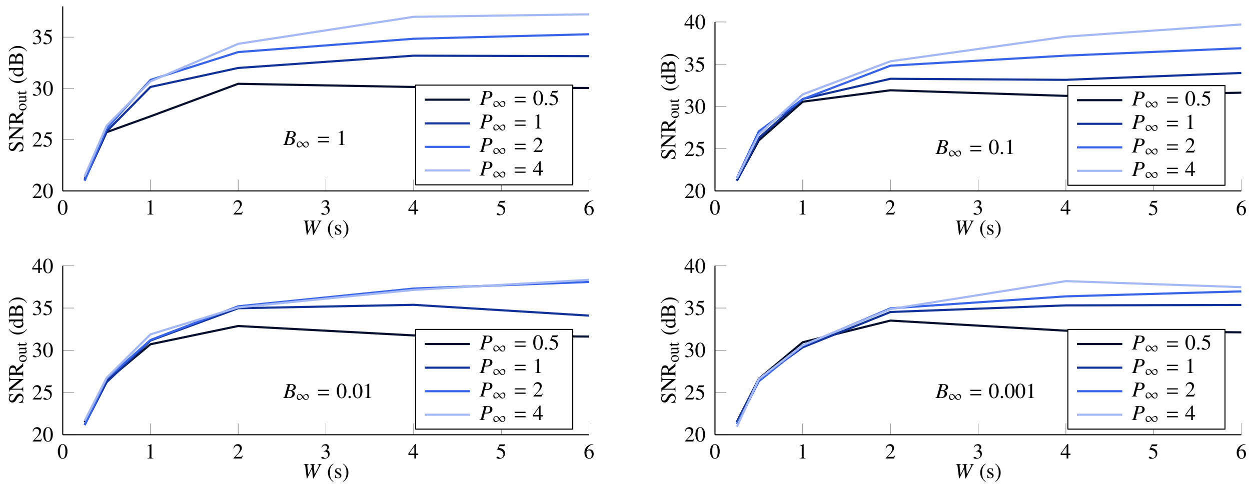

**Figure 6:** Trade-off between amplitude settling time and $\text{SNR}_{\text{out}}$. The plots display the $\text{SNR}_{\text{out}}$ versus amplitude settling time $W$, for different $P_\infty$ and $B_\infty$. $\text{SNR}_{\text{in}}$ is set to 0 dB for all the cases. The results show that high $\text{SNR}_{\text{out}}$ values (> 30 dB) can be achieved along with a reasonably fast settling time (< 1 s at $W = 1$). Parameter setting: $\{f_s = 1$ kHz, $B_0 = 50$, $B_{\text{st}} = 1$, $P_0 = 0.1$, $P_{st} = 1\}$

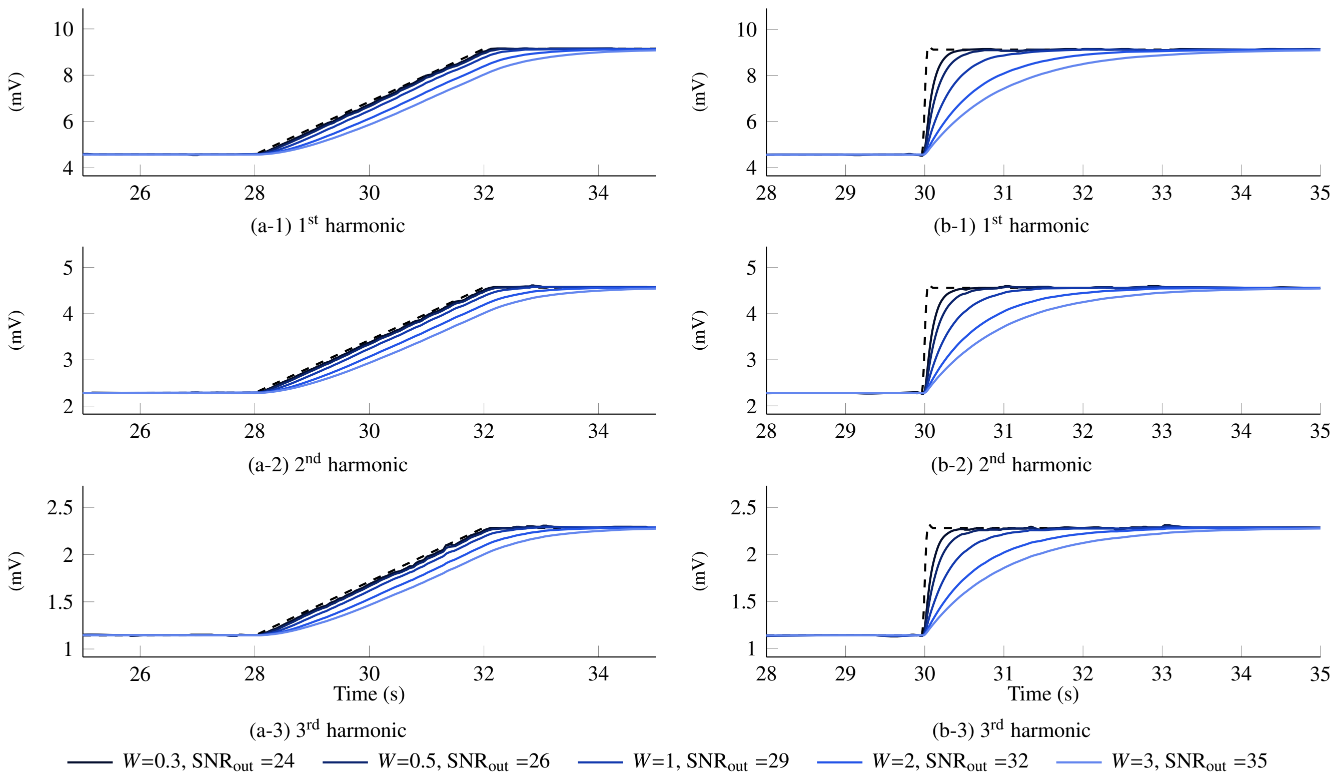

**Figure 7:** Amplitude tracking. In (a-1)-(a-3), the amplitudes of the harmonics were gradually increased to twice their initial values. In (b-1)-(b-3), the amplitudes of the harmonics underwent a step jump. The actual amplitude is displayed by (- - -). The algorithm was applied with different values of $W$ which led to different settling times and $\text{SNR}_{\text{out}}$ values. The input SNR was set to $\text{SNR}_{\text{in}} = 0$ dB, and $\text{SNR}_{\text{out}}$ values were calculated after convergence ($t > 35$ s). As can be seen, smaller values of $W$ have led to faster amplitude tracking, however yielded lower $\text{SNR}_{\text{out}}$. On the other hand, larger values of $W$ resulted in a slower amplitude tracking but yielded higher $\text{SNR}_{\text{out}}$. Parameter setting: $\{f_s = 1$ kHz, $B_0 = 50$, $B_{\text{st}} = 1$, $B_\infty = 0.1$, $P_0 = 0.1$, $P_{st} = 1$, $P_\infty = 1\}$

the small minimum and maximum deviations—shown by the error bars—indicate the reliable convergence and consistent performance of the proposed algorithm. In this simulation, large lower error bars indicate that the algorithm under test may fail to converge.

The second simulation is carried out to evaluate the convergence behaviour of the adaptive algorithms in the mean sense. For this purpose, For this purpose, random signals with $1/f$ PSD (mimicking neural signal PSD) were generated, each of which was superimposed with a single sinusoid (mimicking the interference) whose frequency was slightly deviated from 60 Hz. Subsequently, the MSEs between the output of each algorithm and the actual random signal (without the interference) were calculated. The simulation was then repeated with different values of $\text{SNR}_{\text{in}}$ and interference frequency ($f_I$). In this evaluation, faster convergence and lower MSE values are desirable factors. Figure 13 shows that the proposed algorithm consistently achieves faster convergence and lower MSE compared with the other methods. Furthermore, it can reasonably achieve its

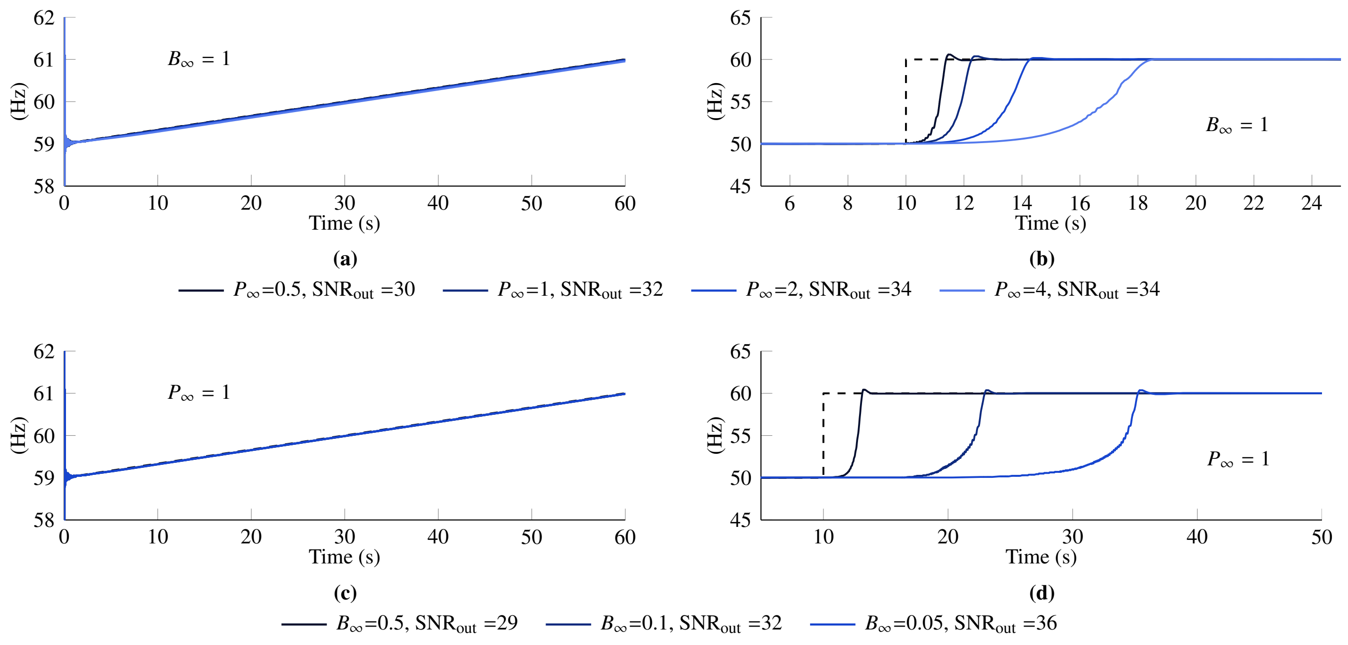

**Figure 8:** Frequency tracking. The actual frequency is shown by (- - -). In (a) and (c) the fundamental frequency is swept from 59 Hz to 61 Hz. It can be seen that, the estimated values properly track the changes of the frequency, with the minimum SNR$_{out}$ = 26 dB during tracking. In (b) and (d), the fundamental frequency is abruptly changed from 50 Hz to 60 Hz, and the frequency estimates have well track the change. In this simulation, SNR$_{in}$ = 0 dB and SNR$_{out}$ values are calculated after convergence (t > 20 s in (b) and t > 40 s in (d)). In (a) and (b), $B_\infty$ = 1 and $P_\infty$ was varied. In (c) and (d), $P_\infty$ = 1 and $B_\infty$ was varied. Parameter setting: $\{f_s = 1$ kHz, $B_0 = 50$, $B_{st} = 1$, $P_0 = 0.1$, $P_{st} = 1$, $W = 1\}$

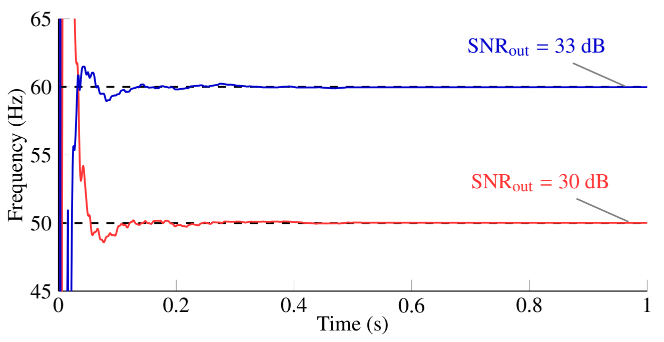

**Figure 9:** The initial convergence to two interference frequencies at 50 Hz and 60 Hz. A quick initial frequency convergence can be observed. In this simulation SNR$_{in}$ = 0 dB, and the values of SNR$_{out}$ were calculated after $t$ = 1 s. Parameter setting: $\{f_s = 1$ kHz, $B_0 = 50$, $B_\infty = 0.05$, $B_{st} = 0.5$, $P_0 = 0.1$, $P_\infty = 2$, $P_{st} = 0.5$, $W = 1\}$

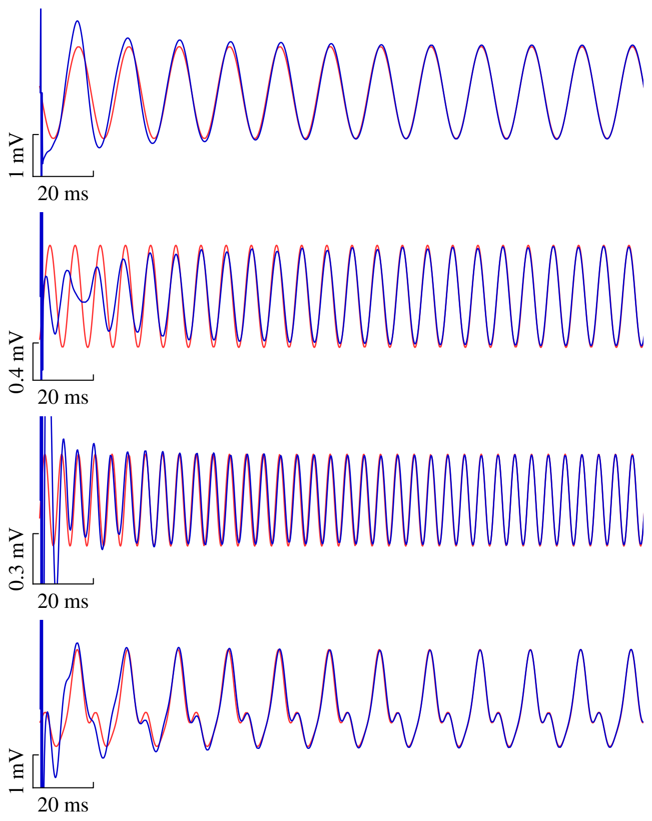

**Figure 10:** Initial convergence of the estimated harmonics. The method shows a fast adaptation of frequency, phase and amplitude. From top to bottom, 1st, 2nd, 3rd harmonic and total interference. The plots show actual components (——) and the estimates (——). In this simulation, SNR$_{in}$ = 0 dB and SNR$_{out}$ = 33 dB (for $t$ > 1 $s$). Parameter setting is the same as that of figure 9.

optimum performance regardless of the initial deviation of the interference frequency from its nominal value.

In the simulations, we observed that the two other adaptive methods were sensitive to the large amplitude artefacts—which are usually present in neural recording—such as electrode displacement and movement artefacts. In addition, since the performance of the algorithms depend on their parameter setting, we fine tuned the parameters of each algorithm to achieve its best performance—in terms of lower MSE and faster convergence—at SNR$_{in}$ = 0 dB and $f_I$ = 59 Hz. Furthermore, the adaptation blocking in the Martens' algorithm is not applicable to ECoG signals, thus their SAC 2 method was used.

*3.2.2. Effects on Synthetic Oscillations* Neural oscillations (bio-markers) can appear at, or in the vicinity of, the interference frequency bands. Since these oscillations are useful for information decoding, it is important to ensure that

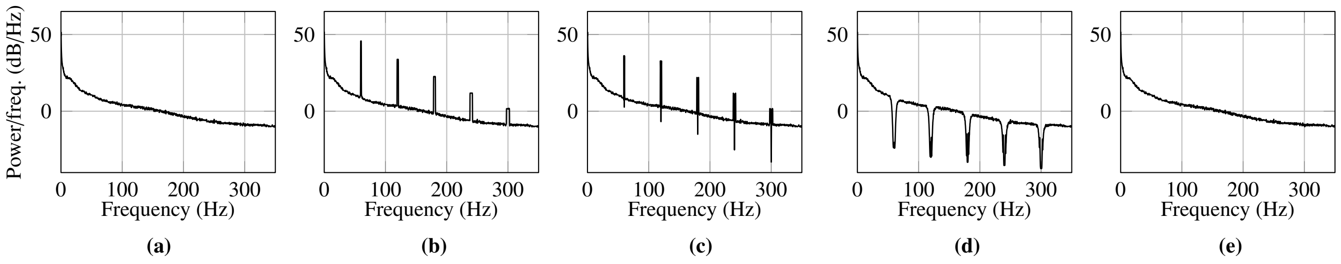

**Figure 11:** The effect of notch filtering on ECoG signal corrupted with power line interference. (a) PSD of the actual ECoG signal. (b) PSD of the synthetically corrupted signal. The interference fundamental frequency is slightly deviated from 60 Hz. (c) PSD after applying narrow-band (1 Hz) IIR notch filters centred at 60 Hz and multiples. (d) PSD after applying wide-band (8 Hz) notch filters. (e) PSD after interference cancellation with the proposed algorithm, where the interference is completely removed while the signal frequency bands are minimally affected. The narrow-band filter fails to adequately remove the interference due to its changing frequency. The wide-band filter distorts the signal spectrum at the rejection bands.

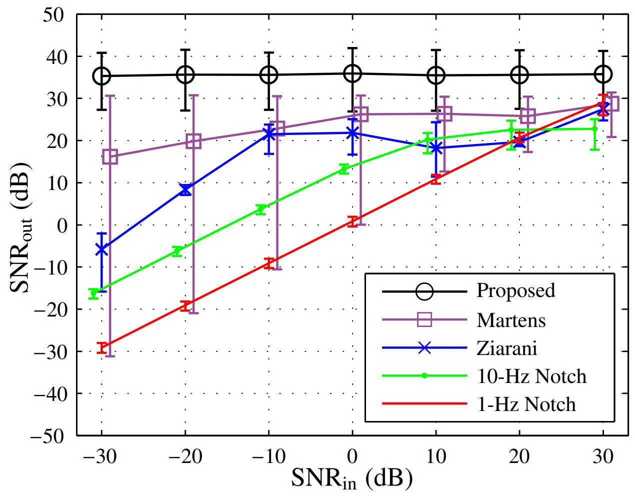

**Figure 12:** Comparison of asymptotic performances of different interference removal methods. The plots represent the average and the error bars indicate the maximum and minimum SNR$_{out}$. For each SNR$_{in}$, the SNR$_{out}$ values are obtained throughout 200 independent runs on synthesized sequences of corrupted ECoG signals. The interference consists of a stationary sinusoid fixed at 59 Hz. The SNR$_{out}$ values are calculated after t = 60 s to ensure the full convergence of the algorithms. When applied on ECoG signals, the proposed algorithm consistently yields high SNR$_{out}$. The Martens' algorithm performs well in the mean sense; however, large deviations of minimum SNR$_{out}$ imply that it may not always converge. The Ziarani's algorithm is sensitive to the input signal power, thus same parameters cannot be used to achieve an optimal performance for different values of SNR$_{in}$. The 10-Hz and 1-Hz notch filters are centred at 60 Hz. The parameter setting is the same as that of figure 13.

they are well preserved and/or undergo minimal distortion during interference cancellation. To illustrate the performance of the algorithm in this regard, it is tested on synthetic oscillations contaminated with interference. For this purpose, a sequence of patterned oscillations in the range of 50–70 Hz was generated and then added to a background random signal with $1/f$ PSD (figure 14a). This sequence represents a synthetic neural signal. Subsequently, a sinusoid (representing the interference) was synthesized and added to the signal. The frequency of this sinusoid was swept from 59 Hz to 61 Hz, and its amplitude was logarithmically increased, setting SNR$_{in}$ from 10 dB to -20 dB (figure 14b).

Different methods of interference removal were applied to the synthesized signal. Figure 14c illustrates that the proposed algorithm has tracked and removed the interference while reasonably preserving the signal components. Figure 14d shows that in the Martens' algorithm, the phase-locked loop (PLL) has become out of lock due to the oscillations (e.g. 5 < t < 15 s). Figure 14e shows that the Ziarani's algorithm is sensitive to the interference power, thus same parameters cannot be used to obtain adequate performance for different power of the signal and/or interference. Furthermore, it has distorted the signal near the interference frequency band. Figure 14f indicates that 10-Hz notch filter has excessively removed the signal components. Figure 14g shows that the 1-Hz notch filter has only attenuated the interference near $t$ = 30 s when its frequency was close to 60 Hz, and failed to remove the interference otherwise. The resultant SNR$_{out}$s (calculated for $0 < t < 60$) are displayed in table 3.

**Table 3:** Results of simulation with synthetic oscillations

| Methods | SNR$_{out}$ (dB) |
|---|---|
| Proposed | 12.06 |
| Martens | 8.60 |
| Ziarani | 7.90 |
| 10-Hz Notch | 2.20 |
| 1-Hz Notch | −7.85 |

### *3.3. Performance Evaluation on Real Data*

Three types of biosignals including extracellular, ECoG and EEG recordings were used to demonstrate the performance of the algorithm on real data. These signals were recorded in ordinary environments, thus containing a significant amount of power line interference. The PSD of the recorded signals are displayed in figures 15a, 15d and 15g where the presence of the power line interference can be clearly seen. It can be observed that, the harmonics' power can be tens of dB higher than the signal power at the contaminated bands. Furthermore, both odd and even harmonics may be present in the recorded signals.

The extracellular, ECoG and EEG recordings were sampled at 40 kHz, 1 kHz and 128 Hz, respectively. The algorithm was applied to the recorded signals to cancel the interference. Figures 15b, 15e and 15h show the PSDs after interference cancellation, where the harmonics have been removed. In addition, the algorithm does not distort the signal frequency components where no harmonic is present. For example in figure 15b, the PSD remained unchanged at the

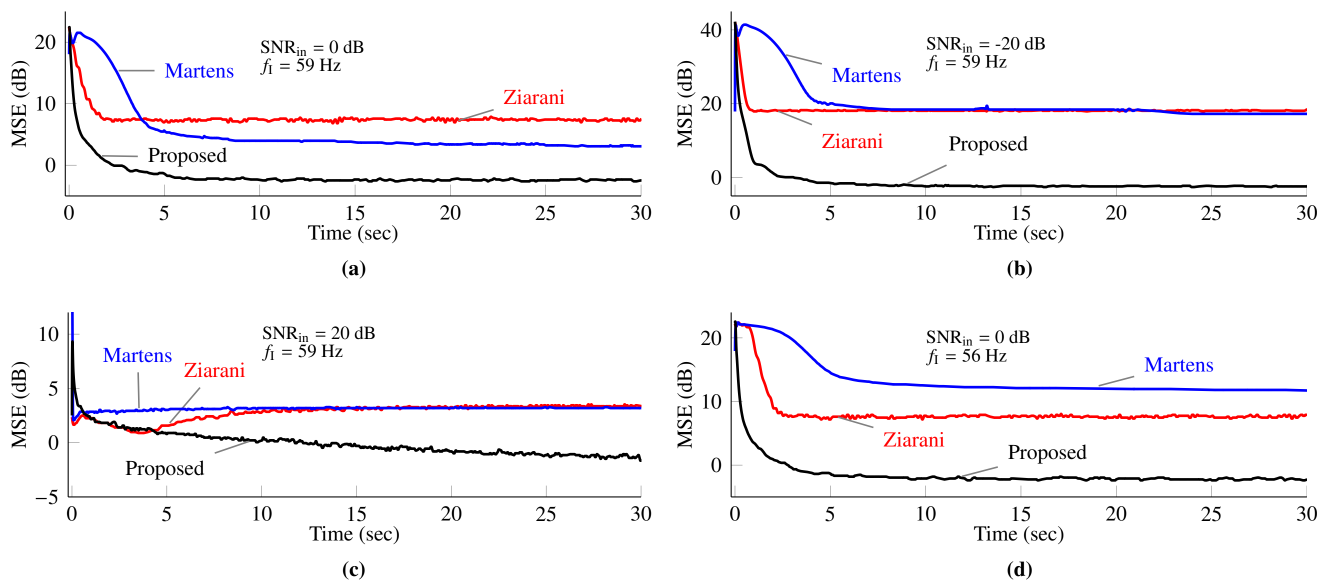

**Figure 13:** Learning curves of the proposed, Ziarani's [23] and Martens' [24] algorithms when applied to random signals with $1/f$ PSD superimposed with a single stationary sinusoid at $f_I$ Hz. The mean square errors (MSEs) are calculated through 1000 independent runs. The parameters of the algorithms were fine tuned to achieve their optimum performance at $\mathrm{SNR_{in}} = 0$ dB and $f_I = 59$ Hz which is shown in (a). (b) In low $\mathrm{SNR_{in}}$, the proposed algorithm still yields low MSE, whereas the increased MSE using the other methods. (c) In high $\mathrm{SNR_{in}}$, the proposed algorithm can further achieve lower MSE. (d) At $f_I = 56$ Hz; unlike the two other methods, the performance of the proposed algorithm is highly insensitive to the initial frequency deviations. As can be seen in (a)-(d) the proposed algorithm consistently yields faster convergence along with lower MSE compared with the other methods. The nominal frequency is set to 60 Hz in the Ziarani's and Martens' algorithms; however, the proposed algorithm does not require a priori setting of nominal frequency. Parameter setting, $f_s = 1$ kHz, Proposed:$\{B_0 = 50, B_{\mathrm{st}} = 0.5, B_\infty = 0.1, P_0 = 0.1, P_{st} = 0.5, P_\infty = 1, W = 1\}$, Ziarani: $\{\mu_1 = 8, \mu_2 = 1000, \mu_3 = 0.02\}$, Martens:$\{\tau = 200, \zeta = 1, \omega_n/\omega_p^n = 0.02\}$

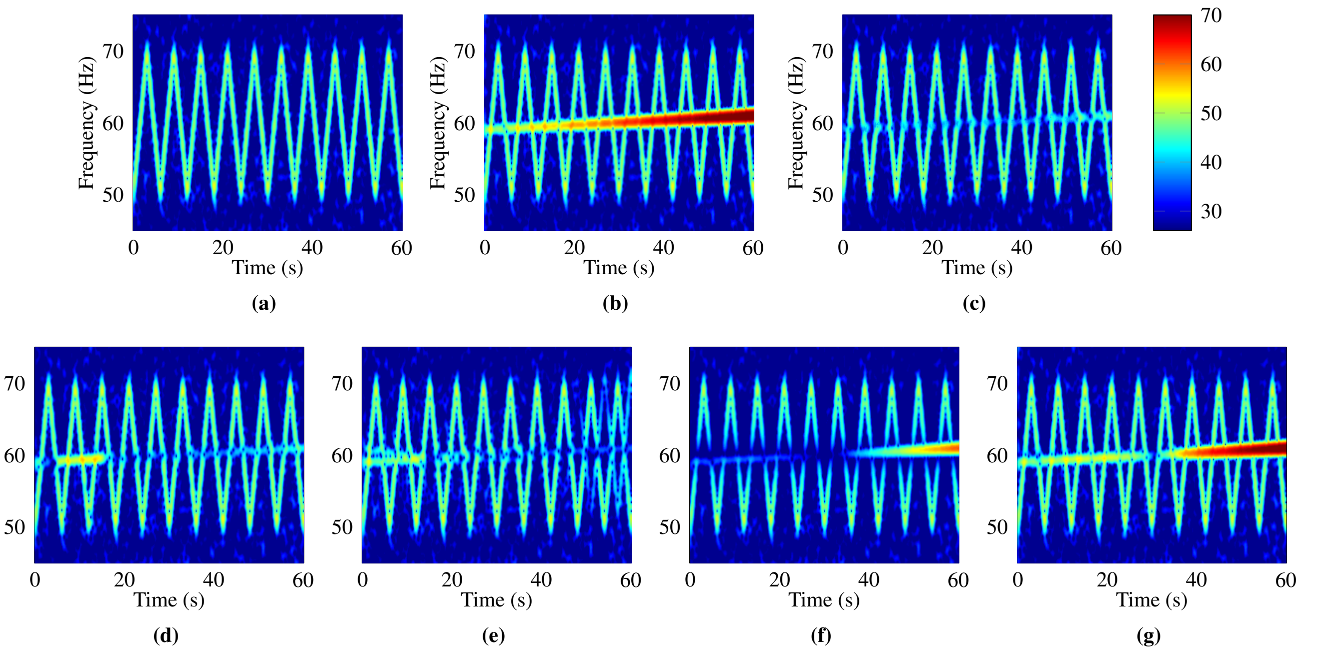

**Figure 14:** Simulation with synthetic oscillations. Time-frequency plots: (a) Synthesized signal consisting of bidirectional chirp between 50–70 Hz (representing the signal of interest) superimposed with a $1/f$ PSD background signal (representing neural noise). (b) After adding a sinusoidal interference whose frequency was swept from 59 Hz to 61 Hz, and its amplitude was logarithmically increased, setting $\mathrm{SNR_{in}}$ from 10 dB to −20 dB. (c) The proposed algorithm has tracked and removed the interference while reasonably preserving the signal components. (d) In the Martens' algorithm, the phase-locked loop (PLL) has become out of lock due to the oscillations (e.g. $5 < \mathrm{t} < 15$ s). (e) The Ziarani's algorithm is sensitive to the interference power, thus same parameters cannot be used to obtain adequate performance for different power of the signal and/or interference. Furthermore, it has distorted the signal near the interference frequency band. (f) 10-Hz notch filter has excessively removed the signal components. (g) 1-Hz notch filter has only attenuated the interference near $t = 30$ s when its frequency was close to 60 Hz, and failed to remove the interference otherwise. The resultant $\mathrm{SNR_{out}}$ values (for $0 < t < 60$) are displayed in table 3. Parameter setting, $f_s = 1$ kHz, Proposed:$\{B_0 = 20, B_{\mathrm{st}} = 0.5, B_\infty = 0.1, P_0 = 0.2, P_{st} = 1, P_\infty = 0.5, W = 1\}$, Ziarani: $\{\mu_1 = 12, \mu_2 = 0.001, \mu_3 = 0.2\}$, Martens:$\{\tau = 100, \zeta = 1, \omega_n/\omega_p^n = 0.01\}$

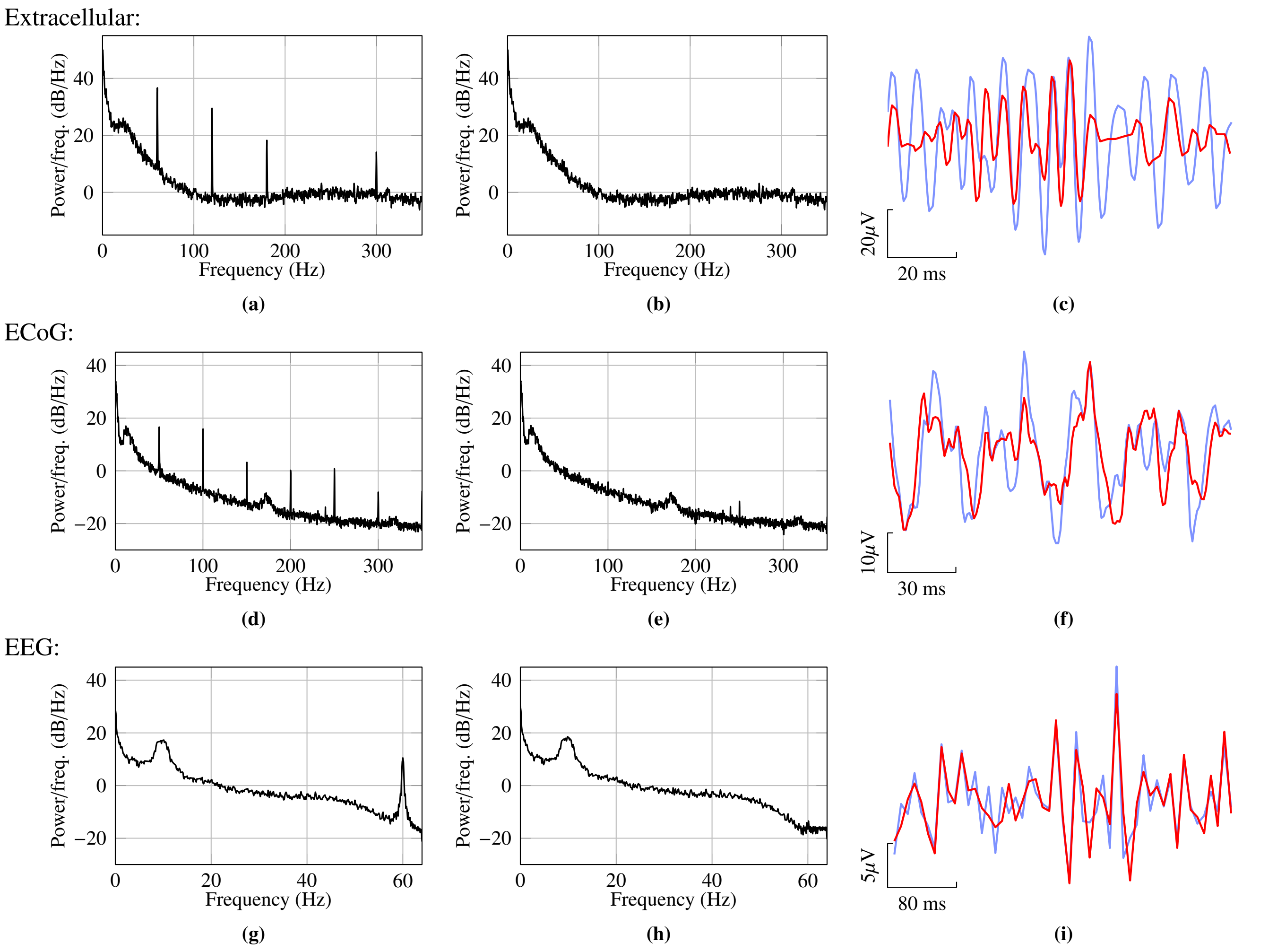

**Figure 15:** Results of the experiment with real data. First column: PSD of the actual recorded signals containing power line interference. Second column: after applying the proposed algorithm, where the interference harmonics are removed. Third column: the actual recorded signal (—), and after interference cancellation by the proposed algorithm (—), displayed in the gamma band (> 30 Hz). Note that in (b), the PSD remained minimally affected at 240 Hz (4th harmonic), where no harmonic was present (compare with (a)). In this experiment common parameter setting is $\{B_0 = 50, B_\infty = 0.05, B_{st} = 1, P_0 = 0.1, P_\infty = 4, P_{st} = 1\}$; and specific parameters settings are, extracellular: $\{f_s = 40$ kHz, $W = 2\}$, ECoG: $\{f_s = 1$ kHz, $W = 1\}$, EEG: $\{f_s = 128$ Hz, $W = 0.5\}$.

frequency of the 4th harmonic. Figures 15c, 15f and 15i display portions of the gamma band signals before and after interference cancellation.

It is worth mentioning that, a favorable property of the proposed algorithm is that the proper parameter values remained the same (except $W$) for various signal modalities and sampling rates, confirming the usefulness of defining the alternative parameters. This property makes the algorithm easy to apply on various types of biopotential recordings, with only a slight adjustment of its parameters.

## 4. Discussion

In the design of the algorithm, a number of techniques are used to reduce its computational complexity. First, trigonometric function calculations are avoided in several parts including frequency estimation, harmonic frequencies calculation, and harmonic sinusoids generation. Second, the RLS algorithm is simplified by diagonal approximation of its covariance matrix. These considerations are particularly important in hardware implementation, and significantly reduce the circuit area. To optimize the area further, a number of resource sharing techniques are used to share dividers, multipliers and some reusable circuit blocks at the cost of an increased clock frequency. In general, the circuit blocks of the discrete oscillator and the amplitude/phase estimator can be reused to remove $M$ number of harmonics at a $M$ times higher clock rate while requiring $7M'$ registers to store the state variables. Furthermore, the same circuitry can be reused to implement the two RLS update equations with 2 times higher clock rate. In this case, the blocks operate at different clock rates at $f_s$, $2f_s$, $M'f_s$ and $2M'f_s$.

The algorithm provides instantaneous interference cancellation, implying a zero phase shift between the input and output signals. In hardware implementation, the delay between input and output signals is determined by the circuit propagation delay which depends on several factors including the critical path of the circuit, fabrication technology and temperature. In our implementation in a 65-nm technology, this delay is negligible ($\ll 1\ \mu$s) considering the typical sampling rates used in biopotential recording (i.e. up to tens of kHz).

Although the algorithm is mainly proposed for power line interference cancellation, it can also be used to cancel other types of harmonic interferences which may present in the recording. For this purpose, the corner frequencies of the bandpass filter should be adjusted accordingly. Furthermore, the algorithm is applicable to other types of biopotential recordings including ECG and electromyography (EMG) with no modification; however, the test results on these recordings are not presented in this paper. Moreover, we tested the algorithm on signals corrupted with different types of artefacts such as muscle, eye movement, electrode displacement, and other low and high frequency artefacts. We observed that the algorithm is highly robust to such artefacts. It is important to note that the input signal is assumed to be zero-mean, thus any DC bias should be removed before applying the algorithm.

It should be noted that the algorithm relies on the first harmonic of the interference for frequency estimation. In some situations, however, the first harmonic is suppressed by the recording amplifier, but higher harmonics are still present in the signal. In such situations, the bandpass filter can be accordingly adjusted (e.g. to 90–130 Hz) to estimate the frequency of the second harmonic ($\kappa_2$), and the fundamental frequency estimate $\kappa_f$ can be obtained through $\kappa_f = \sqrt{(\kappa_2+1)/2}$ and subsequently be used for harmonic estimation.

## 5. Conclusion

A robust and efficient algorithm is proposed to remove non-stationary 50/60 Hz interference and its harmonics, from neural recordings. It is highly insensitive to the power of the interference, maintaining high output SNR (> 30 dB) in a wide range of signal and interference conditions. It can effectively track the variations in the frequencies, amplitudes and phases of the harmonics to cancel the interference without compromising the actual neural signals at the interference frequency bands. Furthermore, it features low computational and memory requirements despite using no reference signal for estimation. This property makes the algorithm suitable for real-time applications and hardware implementation.

The convergence, tracking, and estimation accuracy of the algorithm can be controlled through several parameters. An alternative form of these parameters are introduced which have intuitive meaning, and make the parameter adjustment straightforward.

The performance of the algorithm is quantitatively evaluated in terms of output SNR, trade-off between settling time and SNR$_{\text{out}}$, and the convergence behaviour. High SNR$_{\text{out}}$ (> 30 dB) is consistently achieved in different conditions of SNR$_{\text{in}}$ (−30 dB to 30 dB), power line frequencies (45 Hz to 65 Hz), and sampling rates (as low as 100/120 Hz). Test results on the trade-off between settling time and SNR$_{\text{out}}$ as well as the tracking behaviour, demonstrate the fast adaptation of the algorithm to interference variations, while maintaining a high SNR$_{\text{out}}$. This makes the algorithm highly suitable for practical applications such as in wearable recording systems, where the interference power undergos large variations. Moreover, the algorithm features quick (<100 ms) initial convergence.

The comparative performance evaluation between the proposed algorithm with two other adaptive methods shows the improved performance of the proposed algorithm in terms of noise immunity, output SNR and convergence behaviour.

The algorithm is tested on real extracellular, ECoG and EEG recordings, where almost complete removal of the interference—while preserving the neural signals—is observed. It is also applicable to other types of biopotential recordings including ECG and electromyography (EMG), with no modification.

The MATLAB implementation of the proposed algorithm is provided at [32], which caters to various biosignal recording applications. Furthermore, the chip implementation demonstrates the suitability of the algorithm for ASIC and real-time applications.

## Acknowledgments

The ECoG and EEG data were downloaded from `http://neurotycho.org/` [13] and `http://physionet.org/` [41], respectively. The authors would like to acknowledge Edward Keefer at Plexon, and Victor Pikov at Huntington Medical Research Institute for providing extracellular recording data.

The authors would also like to thank lab members Tong Wu for assisting in chip design and testing, and Azam Khalili and Amir Rastegarnia for their comments on an earlier draft of this paper.

The authors acknowledge the funding support by A*STAR PSF Grant R-263-000-699-305, A*STAR-CIMIT Grant R-263-000-A32-305, and NUS Grant R-263-000-A29-133.

## Appendix A. Mathematical Derivations

### *Appendix A.1. RLS algorithm*

The RLS algorithm is used to adapt the weights of the adaptive linear combiner in figure 3c. The weighted least squares cost function is defined as

$$\begin{aligned} e_k(i) &= x(i) - \hat{h}_k(i), \\ E_k &= \sum_{i=0}^{n} \lambda_a^{n-i} e_k^2(i), \end{aligned} \tag{A.1}$$

where $e_k(i)$ is the instantaneous error and $0 < \lambda_a \ll 1$ is the forgetting factor. The RLS algorithm is described as follows. Let $\mathbf{U}_k(n) = [u_k(n)\ u'_k(n)]^{\mathrm{T}}$ be the input sample vector and $\mathbf{W}_k(n) = [\hat{b}_k(n)\ \hat{c}_k(n)]^{\mathrm{T}}$ be the parameter vector. The input sample correlation matrix is defined as

$$\begin{aligned} \mathbf{R}_k(n) &= \sum_{i=0}^{n} \lambda_a^{n-i} \mathbf{U}_k(n) \mathbf{U}_k{}^T(n) \\ &= \begin{bmatrix} r_{1,k}(n) & r_{2,k}(n) \\ r_{3,k}(n) & r_{4,k}(n) \end{bmatrix}. \end{aligned} \tag{A.2a}$$

$\mathbf{R}_k(n)$ can be recursively calculated through

$$\begin{aligned} &\mathbf{R}_k(0) = \epsilon I_2, \epsilon > 0 \quad \text{for } k = 1, \cdots, M', \\ &\mathbf{R}_k(n) = \lambda_a \mathbf{R}_k(n-1) + \mathbf{U}_k(n)\mathbf{U}_k^{\mathrm{T}}(n). \end{aligned} \tag{A.2b}$$

Now, $\mathbf{W}_k$ can be adapted through the following RLS update equation [42].

$$\mathbf{W}_k(n+1) = \mathbf{W}_k(n) - \mathbf{R}_k^{-1}\mathbf{U}_k(n)e_k(n) \tag{A.2c}$$

In the standard RLS method, matrix inversion lemma is used to obtain $\mathbf{R}_k^{-1}$ in order to avoid inverse matrix calculation. In this work, we suggest a simplification on $\mathbf{R}_k$ which leads to a less computational parameter adaptation. It is assumed that the forgetting factor parameter $\lambda_a$ is selected close to the recommended values (i.e $0.5 < W < 5$). In this case, it is shown in Appendix A.2 that $r_{2,k}$ and $r_{3,k}$ would become very small compared with $r_{1,k}$ and $r_{4,k}$, thus they can be neglected and $\mathbf{R}_k$

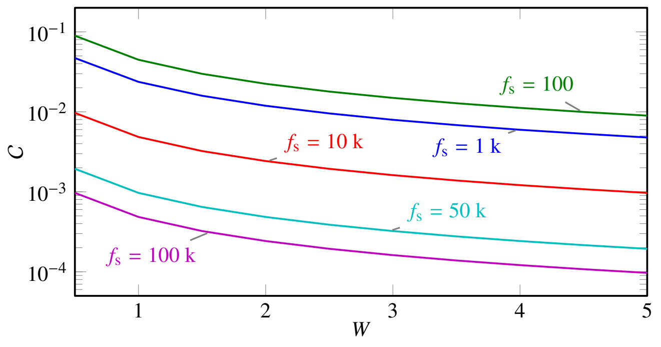

**Figure A1:** The values of $C$ at different sampling rates. It can be seen that at the minimum sampling rate $f_s = 100$, the maximum value of $C$ is less than $0.1 \ll 1$. The value of $C$ monotonously decreases with increasing $f_s$ and $W$

becomes a diagonal matrix. This leads to simplified update equations as

$$\begin{aligned}
&r_{1,k}(-1) = r_{1,k}(-1) = \hat{b}_k(-1) = \hat{c}_k(-1) = 0,\\
&r_{1,k}(n) = \lambda_a r_{1,k}(n-1) + u_k(n)^2,\\
&r_{4,k}(n) = \lambda_a r_{4,k}(n-1) + u'_k(n)^2,\\
&\hat{b}_k(n) = \hat{b}_k(n-1) + u_k(n)e_k(n)/r_{1,k}(n),\\
&\hat{c}_k(n) = \hat{c}_k(n-1) + u'_k(n)e_k(n)/r_{4,k}(n).
\end{aligned} \tag{A.3}$$

These update equations are much simpler than that of (A.2b) with regards to the number of arithmetic operations. We also investigated the effect of this simplification on the performance of the algorithm by comparing the results with the case that the standard RLS algorithm was used, and no significant difference in performance was observed, which validates the proposed approximation.

### *Appendix A.2. Simplification of the RLS algorithm*

We propose a simplification on the RLS algorithm used in the phase and amplitude estimation stage of the algorithm. This simplification is based on approximating the RLS sample correlation matrix $\mathbf{R}_k(n)$ in (A.2a) with a diagonal matrix. The elements of $\mathbf{R}_k(n)$ can be expanded as

$$r_{1,k} = \sum_{i=0}^{n} \lambda_a^{n-i} v^2 \sin^2(k\omega_f i), \tag{A.4a}$$

$$r_{4,k} = \sum_{i=0}^{n} \lambda_a^{n-i} v'^2 \cos^2(k\omega_f i), \tag{A.4b}$$

$$r_{2,k} = r_{3,k} = \sum_{i=0}^{n} \lambda_a^{n-i} v v' \sin(k\omega_f i)\cos(k\omega_f i). \tag{A.4c}$$

If it can be shown that for typical parameters values and sampling rates the coefficients $|r_{2,k}|$ and $|r_{3,k}|$ are much less than $|r_{1,k}|$ and $|r_{4,k}|$, then the sample correlation matrix $R_k(n)$ can be well-approximated by a diagonal matrix (i.e. $r_{2,k} = r_{3,k} = 0$). This subsequently leads to much less computational RLS update equations. In the following derivations, we first show that $|r_{2,k}|, |r_{3,k}| \ll r_{1,k}$; the inequality $|r_{2,k}|, |r_{3,k}| \ll r_{4,k}$ can be similarly derived and is not presented here. We would like to show that the following inequality holds for typical parameter values:

$$\left|\sum_{i=0}^{n} \lambda_a^{n-i} v v' \sin(k\omega_f i)\cos(k\omega_f i)\right| \ll \sum_{i=0}^{n} \lambda_a^{n-i} v^2 \sin^2(k\omega_f i), \tag{A.5a}$$

or equivalently

$$\frac{vv'}{v^2} \frac{\left|\sum_{i=0}^{n} \lambda_a^{n-i} \sin(k\omega_f i)\cos(k\omega_f i)\right|}{\sum_{i=0}^{n} \lambda_a^{n-i} v^2 \sin^2(k\omega_f i)} \ll 1. \tag{A.5b}$$

Note that the right-hand side of (A.5a) is always positive and equal to its absolute value. The magnitudes $v$ and $v'$ are not initially included for the following derivation and will be considered in the last stage. After a little manipulation of (A.5a) using trigonometric identities we get

$$\frac{\sqrt{2}}{2}\left|\sum_{i=0}^{n} \lambda_a^{n-i} \sin(2k\omega_f i + \frac{\pi}{4})\right| \ll \frac{1}{2}\sum_{i=0}^{n} \lambda_a^{n-i} \tag{A.6}$$

which can be expressed in the complex domain as

$$\sqrt{2}\cdot\left|\mathcal{I}m\left\{e^{j\frac{\pi}{4}} \frac{\lambda_a^{n+1} - e^{j2k\omega_f(n+1)}}{\lambda_a - e^{j2k\omega_f}}\right\}\right| \ll \frac{\lambda_a^{n+1}-1}{\lambda_a - 1}. \tag{A.7}$$

Using the property $|\mathcal{I}m\{z\}| \le |z|$, where $z$ is a complex number, we can alternatively show that

$$\begin{aligned}
&\sqrt{2}\cdot\left|e^{j\frac{\pi}{4}} \frac{\lambda_a^{n+1} - e^{j2k\omega_f(n+1)}}{\lambda_a - e^{j2k\omega_f}}\right| = \\
&\left[2\frac{1+\lambda_a^{2(n+1)} - 2\lambda_a^{n+1}\cos(2k\omega_f(n+1))}{1+\lambda_a^2 - 2\lambda_a\cos(2k\omega_f)}\right]^{\frac{1}{2}} \ll \frac{\lambda_a^{n+1}-1}{\lambda_a-1}.
\end{aligned} \tag{A.8}$$

We define

$$\mathcal{A} = \frac{\left[2\frac{1+\lambda_a^{2(n+1)} - 2\lambda_a^{n+1}\cos(2k\omega_f(n+1))}{1+\lambda_a^2 - 2\lambda_a\cos(2\omega_f)}\right]^{\frac{1}{2}}}{\frac{\lambda_a^{n+1}-1}{\lambda_a-1}}, \tag{A.9}$$

$$\mathcal{B} = \max\{\frac{v}{v'}, \frac{v'}{v}\}, \tag{A.10}$$

$$C = \max_{k,n} \mathcal{A}\cdot\mathcal{B}, \tag{A.11}$$

where $v/v'$ is given by (see [40])

$$\frac{v}{v'} = \sqrt{\frac{1+\cos(k\omega_f)}{1-\cos(k\omega_f)}}. \tag{A.12}$$

If we can show that $C \ll 1$, then (A.5b) holds. To show this, numerical simulation is used to obtain the upper bound values on all the harmonics and in the operating condition where $n \gg 1$. For this purpose, the values of $\lambda_a$ are obtained from $W$ through (15b), where $W$ is swept in the range of 0.5–5 which covers the recommended range. Figure A1 displays the values of $C$ in different sampling rates. As can be seen, in all the conditions, $C < 0.1 \ll 1$ indicating that the inequalities $|r_{2,k}|, |r_{3,k}| \ll r_{1,k}$ and $|r_{2,k}|, |r_{3,k}| \ll r_{4,k}$ hold.

### *Appendix A.3. Forgetting factors, Settling time, and Notch bandwidth*

Here, we describe the relation between forgetting factors and settling time as well as the relation between pole radii and notch bandwidth. The proper values of the forgetting factors depend on the sampling rate, making it difficult to adjust their values in general condition. On the other hand, settling time

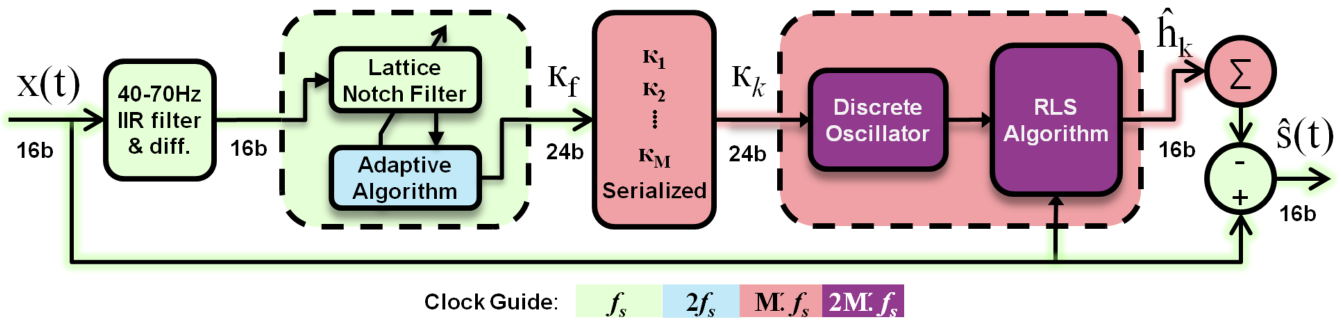

**Figure B1:** System architecture for implementation of power line interference cancellation algorithm. The system operates at multiple clock rates which are indicated in different colours. The slowest clock is equal to the signal sampling rate $f_s$ and the fastest clock depends on the number of harmonics to be removed and is equal to $2M' f_s$.

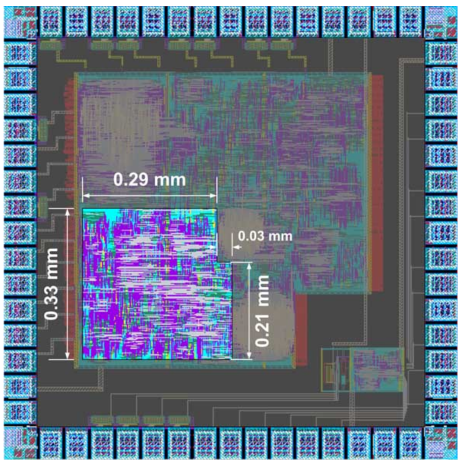

**Figure B2:** The chip layout photo. The area consumed by the interference removal module is approximately 0.11 mm$^2$.

is independent of the sampling rate and has a more intuitive meaning that makes the parameter tuning straightforward. In certain parts of the algorithm such as frequency estimation and phase/amplitude adaptation, settling time can be associated with the forgetting factor by the following formulation

$$0.95\frac{1}{1-\lambda} = \frac{1-\lambda^{n_{\text{set}}+1}}{1-\lambda} \tag{A.13}$$

$$\Rightarrow \lambda = \exp\frac{\ln(0.05)}{t_{\text{set}} f_s + 1}, \tag{A.14}$$

where $n_{\text{set}} = f_s t_{\text{set}}$, $\lambda$ is the forgetting factor, $t_{\text{set}}$ is the desired settling time, and $f_s$ is the sampling rate. The transformation of (A.14) is used in (15b) to adjust $\alpha_{\text{st}}$, $\lambda_0$, $\lambda_{\text{st}}$, $\lambda_\infty$, and $\lambda_a$.

Notch bandwidth is independent of the sampling rate and can be alternatively adjusted instead of the pole radii. Given the notch bandwidth $B$, the pole radii $\alpha$ is obtained by

$$\alpha = \frac{1-\tan \pi B/f_s}{1+\tan \pi B/f_s}. \tag{A.15}$$

## Appendix B. Hardware Implementation

A prototype of the proposed algorithm was fabricated in a 65-nm CMOS process. All the circuit modules are coded in Verilog and implemented in fixed-point arithmetic. The implementation achieves real-time performance with instantaneous interference cancellation (only negligible propagation delay). To optimize the area, a number of resource sharing techniques are used to share dividers, multipliers and some reusable circuit blocks, at the cost of an increased clock frequency by at least 2 times the sampling rate. To account for higher harmonics, same circuit blocks are reused through multiplexing the inputs; hence, considerably saving the circuit area. In general, the cost for processing $M'$ harmonics is $7M'$ number of 24-bit registers as well as an increased system maximum clock rate of $2M' f_s$. The system requires four clock inputs with frequencies $f_s$, $2f_s$, $M' f_s$ and $2M' f_s$.

In the hardware implementation we set $f_s = 1250$ Hz and $M' = 3$ (i.e. removing the first three harmonics), hence requiring a maximum clock rate of 7.5 kHz. The input/output signals are represented by 16-bit signed integers. The clock rates and word lengths of the hardware modules are shown in figure B1. The chip layout photo is displayed in figure B2.

The functionality of the chip was verified against a reference model which was implemented in MATLAB and used full-precision arithmetic. The chip input signal was synthesized by adding a pre-recorded interference-free ECoG signal with a synthetic interference. We carried out three tests to demonstrate the chip functionality. In all the tests, the lengths of the input signals were 1 minute, and the SNR$_{\text{out}}$ values were calculated for $t > 20$ s.

In Test 1, SNR$_{\text{in}}$ = 0 dB and the interference consisted of three stationary harmonics with the fundamental frequency of 59 Hz. The signal was then fed into the chip in real-time and the output was recorded. Figure B3 shows the result of Test 1. The PSD of the chip output signal can be seen in figure B3c showing that the interference is significantly attenuated. Furthermore, the signals traces in the time domain clearly show that the chip output properly follows the reference model output.

In Test 2, the interference power was suddenly increased, changing the SNR$_{\text{in}}$ from 0 dB to −10 dB. Figure B4 shows the chip input and output signals. It can be seen that, the chip output consistently follows the reference model output. Moreover, after the step jump in the interference power at (- - -), the chip output adapted to the change to reject the residual amount of the interference.

In Test 3, SNR$_{\text{in}}$ = 0 dB and the interference fundamental frequency was changed from 60 Hz to 60.2 Hz. The chip input and output signals are shown in figure B5. It can be seen that, the chip output consistently follows the reference model output. Moreover, after the frequency change at (- - -), the chip output adapted to the frequency change and approached the actual signal.

The SNR$_{\text{out}}$ values are displayed in table B1. Slightly less SNR$_{\text{out}}$ values of the chip output compared with the reference model output are due to the precision loss caused by fixed-point

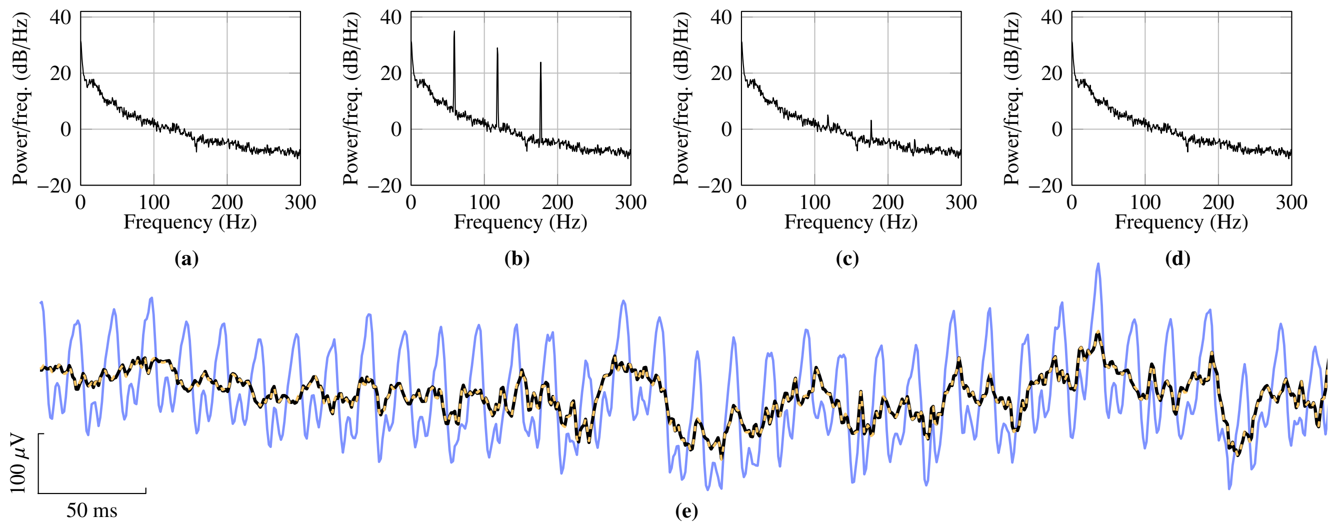

**Figure B3:** Testing result with real-time data. (a) PSD of the clean ECoG signal used for simulation. (b) PSD of the signal contaminated with synthetic interference with the fundamental frequency of 59 Hz, which served as the chip input signal ($\text{SNR}_{\text{in}}$ = 0 dB). (c) PSD of the chip output signal ($\text{SNR}_{\text{out}}$ = 28.9 dB). (d) PSD of the output from the full-precision reference model ($\text{SNR}_{\text{out}}$ = 31.1 dB). The slight difference between (c) and (d) is due the precision loss in fixed-point implementation. (e) Plots of contaminated signal (——), chip output (——) and reference model output (- - - -), where the interference is removed, and the chip output accurately follows the reference model output.

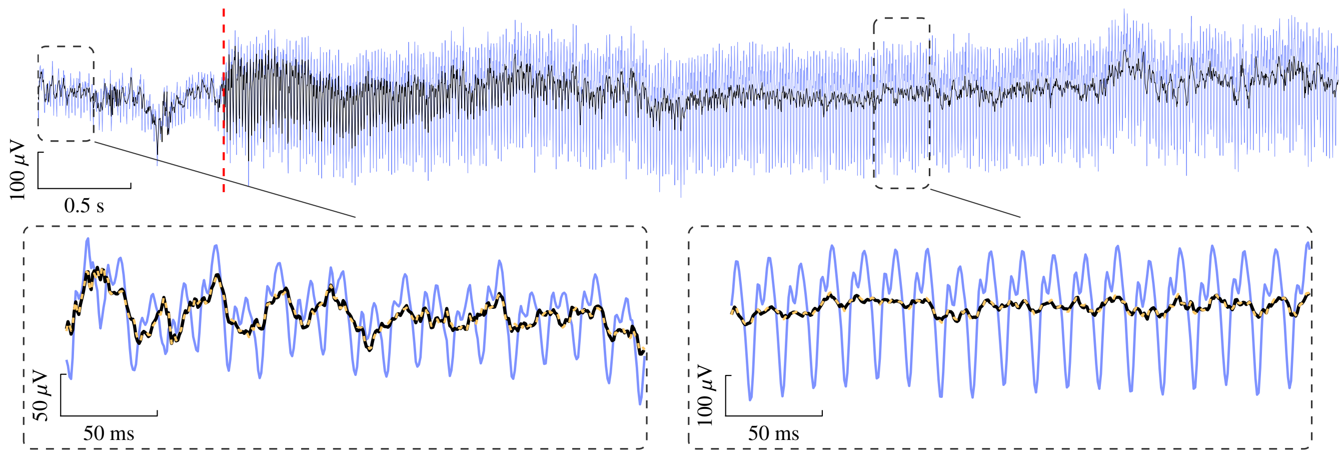

**Figure B4:** Response to a step change in interference power. The chip input signal (——) was synthesized by adding a synthetic interference, containing three harmonics, to a pre-recorded clean ECoG signal. The power of the increased at the time instance indicated by (- - -) ($\text{SNR}_{\text{in}}$ = 0 dB before (- - -) and $\text{SNR}_{\text{in}} = -10$ dB after (- - -)). The plots show the contaminated input signal (——), the chip output signal (——) and the full-precision reference model output (- - - -), where the interference is removed, and the chip output closely follows the reference model output.

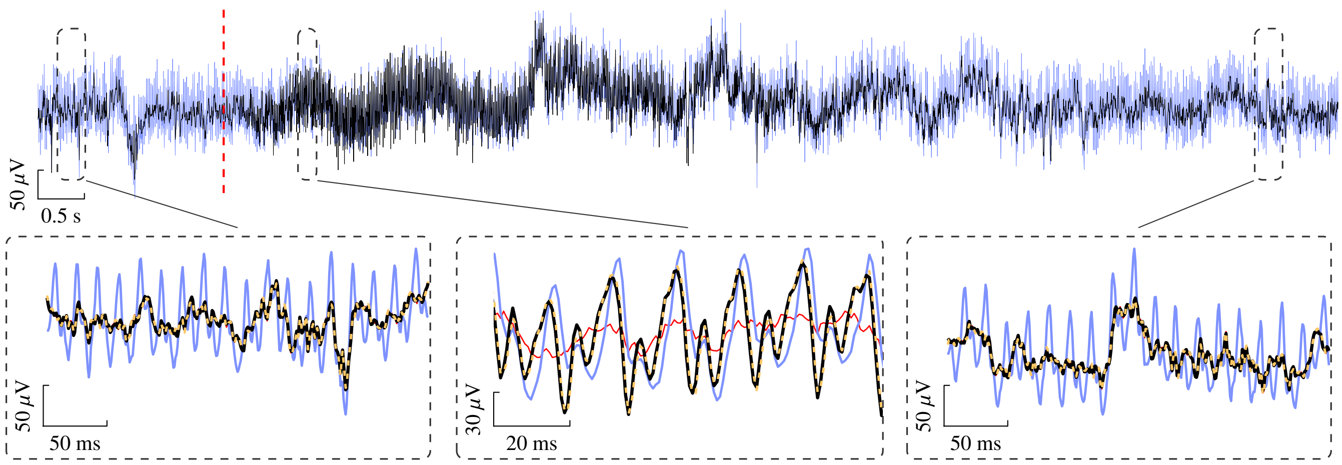

**Figure B5:** Response to a step change in the interference fundamental frequency. The fundamental frequency is changed from 60 Hz to 60.2 Hz at the time instance indicated by (- - -). The plots show the original signal (——), the contaminated signal serving as the chip input (——), the chip output signal (——) full-precision reference model output (- - - -). It can be seen that the chip output adapts to the change and closely follows the output of the reference model.

arithmetic used in the chip implementation.

**Table B1:** Reference model and Chip $\text{SNR}_{\text{out}}$

| Test | $\text{SNR}_{\text{in}}$ (dB) | $\text{SNR}_{\text{out}}$ (dB) | |
|---|---|---|---|
| | | Reference model | Chip |
| Test 1 | 0 | 31.1 | 28.9 |
| Test 2 | −10 | 29.5 | 27.8 |
| Test 3 | 0 | 29.2 | 28.3 |

## References

[1] Chimene M F and Pallas-Areny R 2000 A comprehensive model for power line interference in biopotential measurements *IEEE Trans. Instrum. Meas.* **49** 535–540

[2] Mitra P and Bokil H 2008 *Observed Brain Dynamics* (Oxford: Oxford University Press)

[3] Metting van Rijn A, Peper A and Grimbergen C 1990 High-quality recording of bioelectric events *Med. Biol. Eng. Comput.* **28** 389–397

[4] Teplan M 2002 Fundamentals of EEG measurement *Meas. Sci. Rev.* **2** 1–11

[5] Thorp C K and Steinmetz P N 2009 Interference and noise in human intracranial microwire recordings *IEEE Trans. Biomed. Eng.* **56** 30–36

[6] Buzsaki G, Horvath Z, Urioste R, Hetke J and Wise K 1992 High-frequency network oscillation in the hippocampus *Science* **256** 1025–1027

[7] Jones M S, MacDonald K D, Choi B, Dudek F E and Barth D S 2000 Intracellular correlates of fast (>200 hz) electrical oscillations in rat somatosensory cortex *J. Neurophysiol.* **84** 1505–1518

[8] Staba R J, Wilson C L, Bragin A, Fried I and Engel J 2002 Quantitative analysis of high-frequency oscillations (80–500 hz) recorded in human epileptic hippocampus and entorhinal cortex *J. Neurophysiol.* **88** 1743–1752

[9] Chao Z C, Nagasaka Y and Fujii N 2010 Long-term asynchronous decoding of arm motion using electrocorticographic signals in monkeys *Front. Neuroeng.* **3** 3

[10] Leuthardt E C, Schalk G, Wolpaw J R, Ojemann J G and Moran D W 2004 A brain–computer interface using electrocorticographic signals in humans *J. Neural Eng.* **1** 63–71

[11] Miller K, Shenoy P, den Nijs M, Sorensen L, Rao R and Ojemann J 2008 Beyond the gamma band: The role of high-frequency features in movement classification *IEEE Trans. Biomed. Eng.* **55** 1634–1637

[12] Moran D 2010 Evolution of brain–computer interface: action potentials, local field potentials and electrocorticograms *Curr. Opin. Neurobiol.* **20** 741–745

[13] Shimoda K, Nagasaka Y, Chao Z C and Fujii N 2012 Decoding continuous three-dimensional hand trajectories from epidural electrocorticographic signals in japanese macaques *J. Neural Eng.* **9** 036015

[14] Liang N and Bougrain L 2012 Decoding finger flexion from band-specific ecog signals in humans *Front. Neurosci.* **6** 91

[15] Hwang E J and Andersen R A 2013 The utility of multichannel local field potentials for brain–machine interfaces *J. Neural Eng.* **10** 046005

[16] Miller K J, Zanos S, Fetz E E, Nijs M d and Ojemann J G 2009 Decoupling the cortical power spectrum reveals real-time representation of individual finger movements in humans *J. Neurosci.* **29** 3132–3137

[17] Dugan R C, McGranaghan M F, Santoso S and Beaty H W 2012 *Electrical Power Systems Quality* (New York: McGraw-Hill)

[18] Baggini A 2008 *Handbook of Power Quality* (Chichester: John Wiley & Sons)

[19] Wang Z and Roe A W 2011 Trial-to-trial noise cancellation of cortical field potentials in awake macaques by autoregression model with exogenous input (ARX) *J. Neurosci. Methods* **194** 266–273

[20] Degen T and Jackel H 2004 Enhancing interference rejection of preamplified electrodes by automated gain adaption *IEEE Trans. Biomed. Eng.* **51** 2031–2039

[21] Spinelli E and Mayosky M 2005 Two-electrode biopotential measurements: power line interference analysis *IEEE Trans. Biomed. Eng.* **52** 1436–1442

[22] Alzaher H, Tasadduq N and Mahnashi Y 2013 A highly linear fully integrated powerline filter for biopotential acquisition systems *IEEE Trans. Biomed. Circuits Syst.* **7** 703–712

[23] Ziarani A K and Konrad A 2002 A nonlinear adaptive method of elimination of power line interference in ECG signals *IEEE Trans. Biomed. Eng.* **49** 540–547

[24] Martens S M, Mischi M, Oei S G and Bergmans J W 2006 An improved adaptive power line interference canceller for electrocardiography *IEEE Trans. Biomed. Eng.* **53** 2220–2231

[25] Luck S J 2005 *An Introduction to the Event-Related Potential Technique (Cognitive Neuroscience)* (Cambridge, MA: MIT Press)

[26] Levkov C, Mihov G, Ivanov R, Daskalov I, Christov I and Dotsinsky I 2005 Removal of power-line interference from the ECG: a review of the subtraction procedure *Biomed. Eng. Online* **4** 50

[27] Widrow B, Glover J R, McCool J M, Kaunitz J, Williams C S, Hearn R H, Zeidler J R, Eugene Dong J and Goodlin R C 1975 Adaptive noise cancelling: Principles and applications *Proc. IEEE* **63** 1692–1716

[28] Hamilton P 1996 A comparison of adaptive and nonadaptive filters for reduction of power line interference in the ECG *IEEE Trans. Biomed. Eng.* **43** 105–109

[29] Ferdjallah M and Barr R E 1994 Adaptive digital notch filter design on the unit circle for the removal of powerline noise from biomedical signals *IEEE Trans. Biomed. Eng.* **41** 529–536

[30] Heldman D A, Wang W, Chan S S and Moran D W 2006 Local field potential spectral tuning in motor cortex during reaching *IEEE Trans. Neural Syst. Rehab. Eng.* **14** 180–183

[31] Miller K J, Leuthardt E C, Schalk G, Rao R P N, Anderson N R, Moran D W, Miller J W and Ojemann J G 2007 Spectral changes in cortical surface potentials during motor movement *J. Neurosci.* **27** 2424–2432

[32] Keshtkaran M R 2013 Adaptive Power Line Interference Canceller Source Code *Available Online* https://github.com/mrezak/removePLI

[33] Keshtkaran M R and Yang Z 2012 Power line interference cancellation in in-vivo neural recording *Conf. Proc. IEEE Eng. Med. Biol. Soc.* pp 5214–5217

[34] Cho N I, Choi C H and Lee S U 1989 Adaptive line enhancement by using an IIR lattice notch filter *IEEE Trans. Acoust., Speech, Signal Processing* **37** 585–589

[35] Nehorai A 1985 A minimal parameter adaptive notch filter with constrained poles and zeros *IEEE Trans. Acoust., Speech, Signal Processing* **33** 983–996

[36] Kay S 1989 A fast and accurate single frequency estimator *IEEE Trans. Acoust., Speech, Signal Processing* **37** 1987–1990

[37] Regalia P A 1992 Stable and efficient lattice algorithms for adaptive IIR filtering *IEEE Trans. Signal Processing* **40** 375–388

[38] Cho N and Lee S 1993 On the adaptive lattice notch filter for the detection of sinusoids *IEEE Trans. Circuits Syst. II* **40** 405–416

[39] Klein J 2006 Fast algorithms for single frequency estimation *IEEE Trans. Signal Processing* **54** 1762–1770

[40] Turner C 2003 Recursive discrete-time sinusoidal oscillators *IEEE Signal Process. Mag.* **20** 103–111

[41] Goldberger A L, Amaral L A N, Glass L, Hausdorff J M, Ivanov P C, Mark R G, Mietus J E, Moody G B, Peng C K and Stanley H E 2000 PhysioBank, PhysioToolkit, and PhysioNet components of a new research resource for complex physiologic signals *Circulation* **101** e215–e220

[42] Farhang-Boroujeny B 1999 *Adaptive Filters: Theory and Applications* (Chichester: John Wiley & Sons)