Kurtosis-Based Symbol Timing and Carrier Phase/Frequency Tracking

Todd K. Moon * and Jacob H. Gunther

Abstract: Kurtosis is known to be effective at estimating signal timing and carrier phase offset when the processing is performed in a “burst mode,” that is, operating on a block of received signal in an offline fashion. In this paper, kurtosis-based estimation is extended to provide tracking of timing and carrier phase, and frequency offsets. The algorithm is compared with conventional PLL-type timing/phase estimation and shown to be superior in terms of speed of convergence, with comparable variance in the matched filter output symbols.

Keywords: kurtosis; symbol timing estimation; carrier phase estimation; carrier frequency offset estimation

1. Introduction

In [1], a method was introduced for burst-mode symbol timing estimation and carrier phase estimation based upon complex and real kurtosis, respectively, of the received signal. The method involves computing kurtosis at several different parameter values (for both delay and phase) and is thus computationally expensive and more suited to offline computation than real-time implementations or parameter tracking. In this paper, kurtosis-based methods are extended to algorithms that track timing and phase. The carrier estimation is also extended to include both carrier phase and carrier frequency offset. As tracking algorithms (rather than burst-mode algorithms, which obtain one estimate for an entire burst), these algorithms are potentially amenable to real-time tracking application. In one development, tracking is accomplished by moving downhill on an objective function surface that operates without derivatives in a manner analogous to the Nelder–Mead simplex algorithm in one dimension [2]. In another development, a gradient descent algorithm is employed for phase/frequency estimation. The gradient descent method has higher complexity than the non-derivative method but has an otherwise similar performance. As shown in simulations, the kurtosis-based algorithms typically converge in fewer symbols than conventional PLL-type timing and phase tracking methods.

While [1] applied this kurtosis-based method only to QPSK constellations, in fact, it is agnostic with respect to signal constellation. Conventional synchronization algorithms typically employ knowledge of the signal constellation using training symbols and/or in a decision-directed mode (see, e.g., [3]). In some settings, such as in cognitive radio radio settings, in which an “intelligent receiver ... adapt[s] itself to a specific transmission context and blindly estimate[s] the transmitter parameters for self-reconfiguration purposes” [4], signals with unknown signal constellations may be employed. It would be helpful to be able to perform symbol and carrier sync without knowledge of the constellation, following which constellation identification may be more readily undertaken. The algorithms presented here serve that purpose. In addition, many detection problems require the symbols to be appropriately scaled, which often requires the use of automatic gain control (AGC) loops as part of the synchronization process. It may be advantageous to decouple the AGC loop from symbol timing and phase estimation, which the kurtosis-based approach
provides. The relatively fast convergence of the estimators may also make this useful in short blocklength scenarios for low latency communication systems.

The paper [1] did not contemplate the problem of residual carrier frequency offset, assuming instead that the Fourier transform-based approach to removing carrier offset is completely effective. In this paper, we account for residual carrier frequency offset.

Since kurtosis involves fourth powers of the data, outliers can have a significant effect that can lead an estimate astray or can result in higher variance of the estimated symbols. Another contribution of this paper is to introduce the use of a Huber function [5] to make the estimation more robust.

Kurtosis-based estimation has also been used for blind source separation [6–8], where the kurtosis is used as a measure of non-Gaussianity [9] (Section 8.2). Negentropy is also used as a measure of non-Gaussianity, where the negentropy of a random variable \( y \) is defined as [9] (Section 8.3)

\[
J(y) = H(y_{\text{gauss}}) - H(y)
\]

where \( y_{\text{gauss}} \) is a Gaussian random variable with the same covariance as \( y \). When the negentropy is approximated using higher-order cumulants, the negentropy can be expressed as

\[
J(y) \approx \frac{1}{12} E[y^3]^2 + \frac{1}{48} \kurt(y)^2.
\]

For a zero-mean variable with symmetric distribution, the negentropy is thus essentially equivalent to the kurtosis. This relationship between an entropy-related quantity and the kurtosis is what suggested this article as a venue of publication.

Parameter estimation for communication systems has been widely studied. Textbooks on this topic include [10–16]. Parameter estimation for communication is also covered in conventional digital communication textbooks, e.g., [3,17,18]. See also [19]. What distinguishes this work from all of these references is the use of complex and real kurtosis as a primary tool for adaptation toward the parameter estimates, which enables the parameter estimators to operate agnostic of the signal constellation. While there are some methods that can be applied without knowing the transmitted signal, such as taking powers of the received data to remove digital symbol information, those methods work only with some constellations. By contrast, the kurtosis-based methods are more general. As shown below, they can converge quickly to parameter estimates, more quickly than, for example, PLL-based methods. This suggests the possibility of kurtosis-based phase estimation to be used in phase acquisition and, where the constellation is known, switching over to a PLL-type technique for tracking.

This paper focuses on single-carrier linearly modulated digital communication signals. Extension to multicarrier signals is a topic for future research.

2. Signal Model

Digital information is transmitted at a rate of 1/\( T_s \) symbols per second according to the complex bandpass representation

\[
s(t) = \sum_k s_k p(t - kT_s) e^{j\omega_0 t},
\]

where \( p(t) \) is a unit-energy, pulse-shaping function satisfying the Nyquist zero ISI theorem (e.g., a square-root raised cosine, SRRC [3] (Appendix A)), \( s_k = a_k + jb_k \) is a complex point from the signal constellation, and \( \omega_0 \) is the carrier frequency. The pulse \( p(t) \) is assumed to be symmetric so that the matched filter is the same as \( p(t) \). The received signal is

\[
r(t) = s(t - \tau) + n(t)
\]

where \( \tau \) is delay resulting from transmission through the channel and \( n(t) \) is noise, assumed to be 0 mean. At the receiver, the signal is bandpass filtered and basebanded using a frequency \( \omega_1 \approx \omega_0 \). The resulting complex (nearly) basebanded signal is denoted as
\[ u(t) = e^{-j\omega_0 t} r(t) = \sum_k s_k p(t - kT_s - \tau) e^{j(\omega_c t + \phi)} + n(t), \]

where \( \omega_{\text{off}} = \omega_0 - \omega_1 \) is the residual carrier frequency offset and \( \phi \) accounts for the time delay and changes in index reference at the receiver.

This signal is rotated by an estimate of the offset frequency \( \hat{\omega}_{\text{off}} \) and passed through a matched filter with estimated delay \( \hat{\tau} \) to produce the signal \( x(t) = e^{-j\hat{\omega}_{\text{off}} t} (u(t) * p(t - \hat{\tau})) \), where * denotes convolution. The matched filter output can be expressed in terms of the pulse autocorrelation function

\[ x(t) = e^{j(\phi + (\omega_{\text{off}} - \hat{\omega}_{\text{off}}) t)} \sum_k s_k r_p(t - kT_s - (\tau - \hat{\tau})) + z(t). \tag{1} \]

where \( z(t) \) represents the noise filtered through the matched filter and \( r_p \) is the pulse autocorrelation function

\[ r_p(t) = \int_{-\infty}^{\infty} p(\lambda) p(\lambda - t) \, d\lambda. \]

The representation in (1) is accurate provided that the frequency offset does not exceed about 5% of the symbol rate [16] restriction was pointed out by a reviewer).

In modern practice, of course, the processing steps described above are implemented in discrete time and filters must be truncated to finite length. The signal \( u(t) \) is sampled at a rate of \( P \) (an integer) samples per symbol. Using \( T = T_s / P \), we write the basebanded sampled signal as \( u[n] = u(t)|_{t = nT} \). The pulse-shaping function \( p(t) \) is truncated to finite duration, which we take to be \( (Q - 1)T \), where \( Q \) is an odd integer and \( T \) is the sampling interval. The pulse-shaping function is thus represented by \( Q \) samples, \( p(nT) \). Different authors employ different conventions for the pulse shaping function, representing it either as a noncausal signal, centered around 0, or as a causal signal. We use a notation that accommodates both convention. If the pulse \( p(t) \) is centered around \( t = 0 \), then there are \( \lfloor (Q - 1)/2 \rfloor \) samples before and after \( n = 0 \) and the peak of \( r_p(nT) \) is at \( n = 0 \). If \( p(t) \) and its matched filter are causal, then the samples of interest of \( p(t) \) occur at indices for \( n = 0, 1, \ldots, Q - 1 \). In this case, the peak of the autocorrelation function occurs at time \( t = (Q - 1)T \), corresponding to sample \( n = (Q - 1) \). In either case, let \( O \) (“offset”) be the index offset at which the peak sample of \( r_p \) occurs: \( O = 0 \) for the pulse centered around the origin and \( O = Q - 1 \) for the causal pulse.

Due to the zero-ISI property, samples of the autocorrelation at shifts of multiples of \( T_s \) are 0:

\[ r_p(OT + \ell T_s) = \begin{cases} 1 & \ell = 0 \\ 0 & \ell \neq 0. \end{cases} \]

Let the downsampling be indexed so that the sample at \( n = 0 \) corresponds to the full matched filter response of the first symbol. That is, for the causal pulse-shaping function and its matched filter, \( x[n] = x(t)|_{t = (O+n)T} \) and \( z[n] = z(t)|_{t = (O+n)T} \). This results in

\[ x[n] = e^{j((\omega_{\text{off}} - \hat{\omega}_{\text{off}}) nT + \phi)} \sum_k s_k r_p(OT + nT - kT_s - (\tau - \hat{\tau})) + z[n]. \]

Thus, \( x[0] \) corresponds to the symbol \( s_0 \), etc. In what follows, the noise terms \( z[n] \) is omitted for brevity. (The phase change due to the difference in \( O \) was absorbed here into the phase \( \phi \).)

The matched filter output is downsampled to the symbol rate, taking samples at indices \( n = \ell P, \ell = 0, 1, \ldots \). The downsampling matched filter output is
\[ y[\ell] = x[\ell P] = \sum_k s_k r_p OT + (\ell - k) T_s - (\tau - \tilde{\tau}) \]

This can be decomposed into the term \( k = \ell \) and the other terms are

\[ y[\ell] = e^{i(\omega_{\text{off}} + \tilde{\omega}_P) \ell P} \sum_k s_k r_p \left( OT + (\ell - k) T_s - (\tau - \tilde{\tau}) \right) + e^{i(\omega_{\text{off}} + \tilde{\omega}_P) k P T + \phi} \sum_{k \neq \ell} s_k r_p \left( OT + ((\ell - k) T_s - (\tau - \tilde{\tau}) \right) + \text{ISI} \tag{2} \]

If \( \tilde{\tau} = \tau \), then the second sum, representing intersymbol interference (ISI), disappears and the downsampled matched filter output is \( y[\ell] = s \ell e^{i(\omega_{\text{off}} + \tilde{\omega}_P) \ell P} \), a single rotated (and rotating) symbol output. On the other hand, when \( \tilde{\tau} \neq \tau \), the sample is corrupted by the terms in the ISI sum. The goal of the timing delay estimation problem is to determine \( \tilde{\tau} \) to eliminate the ISI. The goal of the phase/carrier offset problem is to eliminate the complex rotation factor \( e^{i(\omega_{\text{off}} + \tilde{\omega}_P) \ell P + \phi} \).

3. Review of Kurtosis-Based Estimation

With this communication notation in place, we are now in a position to describe kurtosis-based parameter estimation. The kurtosis of a real zero mean random variable \( y \) is defined as \([20,21]\)

\[ K_r(y) = E[y^4] - 3E[y^2]^2, \tag{3} \]

The kurtosis of a complex random variable is \([21]\)

\[ K_c(y) = E[|y|^4] - 2E[|y|^2]^2 - |E[y^2]|^2. \]

The kurtosis (either real or complex) has the following key properties: (1) If \( y \) and \( z \) are independent random variables, then for constants \( a \) and \( b \), \( K(ay + bz) = |a|^4 K(y) + |b|^4 K(z) \), and (2) the kurtosis of a Gaussian random variable is 0. For non-Gaussian random variables, the kurtosis may be greater than zero or less than zero. Random variables representing points drawn from a symbol constellation have negative kurtosis (that is, they are sub-Gaussian).

For a stationary (or nearly stationary) sequence of random variables \( (y[1], y[2], \ldots, y[N_{\text{kurt}}]) \), the kurtosis may be estimated by using sample averages instead of expectations. Thus,

\[ \hat{K}_c(y) = \frac{1}{N_{\text{kurt}}} \sum_{n=1}^{N_{\text{kurt}}} |y(n)|^4 - 2 \left[ \frac{1}{N_{\text{kurt}}} \sum_{n=1}^{N_{\text{kurt}}} |y(n)|^2 \right]^2 - \left[ \frac{1}{N_{\text{kurt}}} \sum_{n=1}^{N_{\text{kurt}}} y(n)^2 \right]^2, \tag{4} \]

and, analogously, an estimate \( \hat{K}_r \) is computed for the real kurtosis. Note that \( \hat{K}_c(y) \) accepts an entire sequence of data as its argument, over which the averaging occurs to estimate the kurtosis.

The complex kurtosis of a sequence of matched filter outputs \( y[\ell] \) may be used to determine \( \tilde{\tau} \) as follows. Considering the matched filter output in (2), when \( \tilde{\tau} \neq \tau \), by the central limit theorem, the sum of terms due to ISI causes \( y[\ell] \) to tend toward having a Gaussian distribution, which has small absolute kurtosis. On the other hand, when \( \tilde{\tau} = \tau \), \( y[\ell] \) consists only of the phase-rotated point from the signal constellation. Since the phase rotation does not affect the complex kurtosis, this \( y[\ell] \) has negative kurtosis. This concept...
is used in [1] to synchronize a communication burst of many symbols (e.g., on the order of 100 symbols) using a method portrayed in Figure 1a. The received signal passes through a bank of $N\tau$ matched filters, each having a different delay. (The burst-mode synchronization operation implies that the matched filtering is computed over the entire received sequence.) The delays $\tau_0, \tau_1, \ldots, \tau_{N\tau-1}$ uniformly sample the range $[0, T_s)$. The kurtosis of each of the downsampled matched filter outputs is computed, and the signal having the lowest (most negative) kurtosis is selected. That is,

$$\hat{\tau} = \arg \min_{\tau} \hat{K}_c(u[n] \ast p(nT - \tau))$$

Since complex magnitudes are employed in the complex kurtosis, a lack of knowledge about phase and carrier offset does not affect the estimation of the delay. By contrast, in conventional time and phase estimation (e.g., [3]), timing and phase are jointly estimated in a joint computational structure, so the convergence of one estimate affects the convergence of the other.

![Block diagram of burst-mode kurtosis-based estimation](image)

**Figure 1.** Block diagram of burst-mode kurtosis-based estimation.

The paper [1] recommends determining carrier frequency offset $\omega_{\text{off}}$ using Fourier transform techniques. This is still recommended when using the methods described in this paper when the carrier frequency offset exceeds the ability of the algorithm to track. However, even after removing the carrier frequency offset using Fourier transform
techniques, some residual carrier may remain to be tracked, which we still refer to as \( \omega_{\text{off}} \). The method presented below reliably perform such offset frequency tracking.

For the moment, however, assume that the frequency offset is removed, so that, when the symbol timing is correctly estimated, the matched filter output is \( y[\ell] = s[\ell]e^{j\phi} \). In terms of real and imaginary parts, \( y[\ell] = y_r[\ell] + jy_i[\ell] \) and \( s[\ell] = a[\ell] + jb[\ell] \). By stacking these components, the rotation can be written, and using

\[
s[\ell] = \begin{bmatrix} a[\ell] \\ b[\ell] \end{bmatrix} \quad G(\phi) = \begin{bmatrix} c(\phi) & -s(\phi) \\ s(\phi) & c(\phi) \end{bmatrix} \quad \text{and} \quad c(\phi) = \cos(\phi), \quad s(\phi) = \sin(\phi),
\]

the rotated symbols can be written as

\[
y[\ell] \triangleq \begin{bmatrix} y_r[\ell] \\ y_i[\ell] \end{bmatrix} = G(\phi)s[\ell] = \begin{bmatrix} c(\phi)a[\ell] - s(\phi)b[\ell] \\ s(\phi)a[\ell] + c(\phi)b[\ell] \end{bmatrix}. \tag{5}
\]

According to (5), the rotation \( G(\phi) \) produces a linear combination of the real and imaginary components of the symbol. Computing the kurtosis (separately) on the real and imaginary components \( y_r[\ell] \) and \( y_i[\ell] \), the kurtosis is smaller (nearer to zero) when \( \phi \neq 0 \) due to the mixture of \( a[\ell] \) and \( b[\ell] \). Remarkably, even though \( y_r \) and \( y_i \) are mixtures of only two random variables, the presence of the mixture can be detected using kurtosis.

A phase estimate \( \hat{\phi} \) is selected and minimizes the sum of the kurtosis of the real and imaginary parts:

\[
\hat{\phi} = \arg \min_{\phi} (\hat{K}_{r}(y_r) + \hat{K}_{i}(y_i))
\]

The phase offset can be removed by computing the product \( y_{\text{rot}}[\ell] = e^{-j\hat{\phi}}y[\ell] \). This can be expressed in matrix/vector form by stacking real and imaginary parts, leading to

\[
y_{\ell,\text{rot}} = G(-\hat{\phi})y[\ell] = G(\phi - \hat{\phi})s[\ell].
\]

When \( \hat{\phi} = \phi \), the components of \( y_{\ell,\text{rot}} \) are not mixed, so that the sum of the real kurtosis of the real and imaginary parts are maximally negative.

This kurtosis-based estimation is applied as shown in Figure 1b. There are \( N_{\phi} \) different rotations which span \([0, 2\pi]\). The time-synchronized matched filter output \( y[\ell] \) is rotated by each rotation, and the sum of the kurtosis of the real part and the kurtosis of the imaginary part are computed. The symbol having is determined from the kurtosis that is smallest (most negative) kurtosis.

### 4. Symbol Timing Tracking: Discrete Downhill Minimization

The kurtosis-based methods portrayed in Figure 1, representing the method in [1], require evaluation using \( N_\ell \) different matched filters in the symbol timing sync, followed by the same number of kurtosis computations, or \( N_{\phi} \) phase rotations, followed by the same number of kurtosis computations, where the kurtosis was computed using data over an entire burst. In [1], the expense of these computations is ameliorated by the fact that these computations are performed only once per burst. However, there is no provision in [1] to re-estimate or track the parameters. For this reason, the method is referred to as a “burst mode” algorithm, suitable for one-time estimation on a burst, without adaptation. The present paper computes only two kurtosis values for each symbol and computes the kurtosis over smaller segments of the signal, which reduces computational complexity (for each kurtosis). The authors found it surprising that, even using rather short segments to estimate the kurtosis (resulting in noisy estimates of the kurtosis), these kurtosis estimates were able to be used to estimate the timing and phase parameters.

This section describes how to use two kurtosis values computed for each symbol using a method that “slides downhill” toward the most negative kurtosis to estimate the timing offset. Only two kurtosis values are computed per symbol (as opposed to evaluating at \( N_\ell \) different values). This adaptive downhill slide is able to track changes in the parameters.
Let the (ostensibly) basebanded signal be denoted as \( u[n] \). It is assumed that it is available over the necessary length of indices (such as by buffering).

Let the sampled \( N_T \) delayed matched filters be denoted as follows:

\[
p_i[n] = p(nT - \tau_i), \quad n = 0, 1, \ldots, Q - 1, i = 0, 1, \ldots, N_T - 1. \tag{6}
\]

The index \( i \) must be in the range 0, 1, \ldots, \( N_T - 1 \). To ensure that is the case, we use the notation \( p_{i \% N_T}[n] \), where \( i \% N_T \) denotes \( i \mod N_T \). In (6), the range of \( n \) is for causal pulses, in which case, \( O \) is \( Q - 1 \), as discussed above. For 0-centered pulses, \( n = \left\lfloor (Q - 1)/2 \right\rfloor, \ldots, 0, \ldots, \left\lfloor (Q - 1)/2 \right\rfloor \), and \( O = 0 \).

In this paper, instead of evaluating kurtosis at \( N_T \) different timing offsets, at each symbol time, kurtosis is evaluated at two timing estimates. Let \( i_1 \) and \( i_2 \) be the indices of delay of the matched filters being used at the present time, where the indices refer to the delay estimates \( \tau_{i_1} \) and \( \tau_{i_2} \). In order to produce the downsampled matched filter outputs used to estimate kurtosis, \( L = Q + (N_{kurt} - 1)P \) input symbols are convolved with the matched filters \( p_{i_1} \) and \( p_{i_2} \), resulting in the matched filter outputs

\[
x_m[n] = u[n : N + L - 1] * p_{i_m \% N_T}[n], \quad m = 1, 2,
\]

where \(*\) denotes convolution. The downsampled matched filter samples are indexed so that the first retained matched filter sample is at \( n = 0 \) are

\[
y_m = x_m[O : P : O + (N_{kurt} - 1)P], \quad m = 1, 2.
\]

Let

\[
K_m = \text{complexkurtosis}(y_m), \quad m = 1, 2,
\]

denote the complex kurtosis estimated from the sequence \( y_m[n] \), computed as in (4).

The minimization algorithm seeks to minimize the kurtosis \( K_m \) over a series of symbol times. The minimization operates analogous to a one-dimensional Nelder–Mead algorithm [2], rolling downhill toward minimum kurtosis by moving in the direction of smaller kurtosis. This is referred to as discrete downhill minimization. It has been found by experimentation on communications data that the kurtosis is, in fact, a convex function of the delay.

At some symbol time of the two kurtoses \( K_1 \) and \( K_2 \), computed using \( p_{i_1} \) and \( p_{i_2} \), the lower (closer to \(-\infty\)) kurtosis is retained, the downsampled matched filter output is saved in \( y_{best} \), and the index of the other delay is adjusted by \( n_T \) steps to attempt to move toward lower kurtosis. The operation is detailed in Figure 2 and the logic is explicitly described in lines 48–60 of Algorithm 1 below. After execution of the steps, \( i_1 \) indexes the delay with lower kurtosis and \( i_2 \) is ready to be tested at the next time step. For example, in Case 1, since \( K_1 > K_2 \), \( i_1 \) is set to the value \( i_2 \) and \( i_2 \) is adjust by some number of steps \( n_T \). \( n_T \) is a small integer, say 2 or 3, indicating how fast to adapt. The other cases in Figure 2 correspond to other configurations of the kurtosis as a function of the delay indices \( i_1 \) and \( i_2 \).

If \( i_1 \) already indexes the delay of minimum kurtosis, \( i_2 \) bounces back and forth between \( i_1 - n_T \) and \( i_1 + n_T \), leaving \( i_1 \) at the delay \( \tau_{i_1} \) producing the lowest kurtosis. The descent algorithm moves at \( n_T \) at each step, so that the average number of steps to converge is \( N_T/(2n_T) \).

The sequence of matched filter outputs selected with the lowest kurtosis is referred to as \( y_{best} \).

The number of steps of adjustment, \( n_T \), is adjusted to provide a variable stepsize algorithm. The variance in the changes in index values is computed. When the variance is below a threshold (suggesting that the estimate is converging to a steady value) \( n_T \) is decremented, provided that it is > 1. This reduces the jitter of the estimate in steady state. This dynamic encourages rapid convergence. Tuning of the algorithm can be performed by
adjusting the decision thresholds. (While not totally satisfying, this is not so different from a PLL-based estimator, for which tuning may be required to obtain a desired performance.)

![Case 1](Figure 2. Discrete downhill minimization of kurtosis function.)

When an index falls outside the range 0, 1, . . ., \(N_\tau - 1\), it should be reduced modulo \(N_\tau\). However, in order to preserve the directional ordering between \(i_1\) and \(i_2\), this modulo reduction occurs only when both indices fall outside this range. Then both indices are reduced. These operations are described in lines 69–77 of Algorithm 1.

5. Carrier Frequency and Phase Tracking: Discrete Downhill Minimization

In this section, a minimization technique is developed for carrier frequency and phase tracking, similar to that used in the previous section for timing synchronization. This allows the algorithm to track these parameters.

The phase is estimated to have one of \(N_\phi\) different values \(\phi_0, \phi_1, \ldots, \phi_{N_\phi}\), which uniformly sample the range \([0, 2\pi)\). Let \(\Delta_\phi\) denote the increment in phase between adjacent steps, e.g., \(\Delta_\phi = \phi_1 - \phi_0\).

Phase is estimated by rotating \(y_{\text{best}}\) using two values of phase indexed by \(j_1\) and \(j_2\) and an estimate of \(\omega_{\text{off}}\), then by estimating the real kurtosis on the real and imaginary parts of the signal, and by using this information to move downhill. The rotating signals are produced by

\[
y_{\phi_1} = e^{-j(\hat{\omega}(0:N_\text{kurt}kurt - 1) + \phi_{j_1N_\phi})} \odot y_{\text{best}}
\]
\[
y_{\phi_2} = e^{-j(\hat{\omega}(0:N_\text{kurt}kurt - 1) + \phi_{j_2N_\phi})} \odot y_{\text{best}},
\]

where \(\odot\) denotes element-by-element multiplication. The real kurtosis is computed as

\[
J_1 = \text{realkurtosis}(\text{Re}(y_{\phi_1})) + \text{realkurtosis}(\text{Im}(y_{\phi_2}))
\]

and similarly for \(J_2\).
The frequency is estimated as follows. Let $j_{last}$ denote the phase index of the best phase at the previous time. The change in phase from the last to the current time is $\Delta \phi = (\delta \phi) (j_{last} - j_{last})$. With an index changes of 0 or ±1, $\Delta \phi$ may be thought of as a discrete-time point process taking values 0 and ±$\delta \phi$. The average value of this point process is the frequency estimate $\hat{\omega}$. The frequency estimate $\hat{\omega}$ taken as the output of a single-pole lowpass filter with unit DC gain and with input $\Delta \phi$. The pole of the filter is denoted by $\alpha$.

The following pseudocode (Algorithm 1) summarizes the timing and phase estimation algorithms.

---

**Algorithm 1: Kurtosis-based timing and phase tracking**

```plaintext
function [mfout, cumwraphtime] = tracker(u,n)

Internal (persistent or class) data:
P = number of samples per symbol (integer)
Nkurt = number of symbols used to estimate kurtosis
Q = number of samples in each SRRC pulse
L = Q + (Nkurt - 1)P = number of samples used in convolution

Variables for time estimation
Nτ = number of delay steps to consider
τ = array of Nτ delays in the range [0, T]
i, j = indices into τ. 0-based indexing is used. Init: i = 0, j = [Nτ/2]

Stored values of the delayed SRRC pulse $p[n] = p[nT - τ_i]$, for $n = 0, 1, \ldots, Q - 1$ and $i = 0, 1, \ldots, Nτ - 1$

cumwraphtime = cumulative wrap count. Init = 0

Variables for variable stepsize algorithm
n = number of steps. Init = 2 or 3
Llast = length of τ history (a circular buffer)
hist = buffer history of $i_1$ (length = Llast)
k = index into hist. Init = 0
Nhist = number of elements in hist. Init = 0
στ = variance of elements in hist.

thresh = variance threshold to reduce stepsize

Variables for frequency estimation
Nφ = number of phase steps to consider
$φ = \text{array of } Nφ \text{ phases in } [0, 2\pi)$
$\delta \phi = 2\pi / (Nφ - 1) =$ phase step size

$φ_{stepsize} = \text{number of index steps to move in phase adaptation. Init = 1}$

j1, j2 = indices into $φ$ array. Init: j1 = 0, j2 = [Nφ/2]
last1 = last value of j1 or j2. Init=0
cumwraphphase = cumulative phase wrap count. Init = 0

Variables for frequency estimation
$\hat{\omega} = \text{frequency offset estimate. Init = 0}$
lastomega = previous value of $\hat{\omega}$. Init = 0

a = pole location of single-pole LPF. a = 0.95

Inputs:
u = received basebanded signal (function uses L samples in u[n : n + L - 1])
n = starting sample of s[n] at this iteration (n increased by P before each call)

Output:
mfout = synched/rotated matched filter output
cumwraphtime = total number of timing wraparounds

(Timing estimation)
```
\( x_1 = u[n : N + L - 1] * p_{1\%N_T} \) (MF outputs (* = convolution))
\( x_2 = u[n : N + L - 1] * p_{2\%N_T} \)
\( y_1 = x_1[Q-1 : P : Q-1+L] \)
\( y_2 = x_2[Q-1 : P : Q-1+L] \)
\( K_1 = \text{complexkurt}(y_1) \) (Compute kurtosis using (4))
\( K_2 = \text{complexkurt}(y_2) \)

(Move downhill toward best delay)

\[
\text{if } K_2 < K_1 \text{ and } i_1 < i_2 \quad \text{(Case 1)}
\]
\[
i_1 = i_2; K_1 = K_2; i_2 += \pi_T
\]
\[
y_{\text{best}} = y_2 \quad \text{(assign the entire sequence)}
\]
\[
\text{elseif } K_1 \leq K_2 \text{ and } i_2 < i_1 \quad \text{(Case 2)}
\]
\[
i_2 = i_1 + \pi_T
\]
\[
y_{\text{best}} = y_1
\]
\[
\text{elseif } K_2 < K_1 \text{ and } i_2 < i_1 \quad \text{(Case 3)}
\]
\[
i_1 = i_2; i_2 -= \pi_T; K_1 = K_2
\]
\[
y_{\text{best}} = y_2
\]
\[
\text{elseif } K_1 < K_2 \text{ and } i_1 < i_2 \quad \text{(Case 4)}
\]
\[
i_2 = i_1 - \pi_T
\]
\[
y_{\text{best}} = y_1
\]

End

(Adjust the step size)
\[
tau_{\text{hist}}[k_T] = i_1
\]
\[
k_T = (k_T + 1)\%L_{\text{hist}}
\]
\[
N_{\text{hist}} = \min(N_{\text{hist}}+1, L_{\text{hist}})
\]
\[
\text{if}(N_{\text{hist}} = L_{\text{hist}})
\]
\[
\sigma_T = \text{var}(\text{tau}_{\text{hist}})
\]
\[
\text{if}(\sigma_T < \text{thresh}_T \text{ and } \pi_T > 2) \pi_T -= 1
\]

End

(If both delays \( \geq N_T \) or both \(< 0 \), wrap around)
\[
\text{wrap} = 0
\]
\[
\text{if } i_1 \geq N_T \text{ and } i_2 \geq N_T
\]
\[
i_1 -= N_T; i_2 -= N_T
\]
\[
\text{wrap} = 1
\]
\[
\text{elseif } i_1 < 0 \text{ and } i_2 < 0
\]
\[
i_1 += N_T; i_2 += N_T
\]
\[
\text{wrap} = -1
\]

End

(At this point, \( y_{\text{best}} \) is an array containing \( N_{\text{hist}} \text{-time-aligned MF outputs} \))

Carrier offset/frequency estimation)
\[
despin_1 = e^{-j(\phi_{1\%N_q} + \hat{\phi}(0; N_{\text{kurt}}-1))}
\]
(array with \( N_{\text{kurt}} \text{ elements} \))
\[
despin_2 = e^{-j(\phi_{2\%N_q} + \hat{\phi}(0; N_{\text{kurt}}-1))}
\]
\[
y_{\phi_1} = y_{\text{best}} \odot \text{despin}_1
\]
\[
y_{\phi_2} = y_{\text{best}} \odot \text{despin}_2
\]
\[
f_1 = \text{realcomplexkurt}(y_{\phi_1}) \quad \text{(Sum kurtosis on real & imag)}
\]
\[
f_2 = \text{realcomplexkurt}(y_{\phi_2})
\]

(Move downhill toward best phase)

\[
\text{if } f_2 \leq f_1 \text{ and } f_1 < f_2
\]
\[
j_1 = j_2; K_1 K_2; j_2 += \phi_{\text{stepsize}}
\]
\[
\text{mfout} = y_{\phi_2}[0] \quad \text{(save matched filter output)}
\]
\[
\text{elseif } f_1 \leq f_2 \text{ and } f_2 < f_1
\]
\[ j_2 = j_1 + \phi \text{stepsize} \]
\[ m_{\text{fout}} = y_{\phi_1}[0] \]
\[ \text{elseif } j_2 \leq j_1 \text{ and } j_2 < j_1 \]
\[ j_1 = j_2; K_1 = K_2 \]
\[ j_2 = \phi \text{stepsize} \]
\[ m_{\text{fout}} = y_{\phi_2}[0] \]
\[ \text{elseif } K_1 < K_2 \text{ and } j_1 < j_2 \]
\[ j_2 = \phi \text{stepsize} \]
\[ m_{\text{fout}} = y_{\phi_1}[0] \]
\[ \text{end} \]
\[ \Delta j = j_1 - \text{last}; \quad \text{(phase difference in counts)} \]
\[ \Delta \phi = (\Delta j)(\delta\phi) \quad \text{(phase difference in rads)} \]
\[ \text{wrap} = 0 \quad \text{(Do phase wrap around)} \]
\[ \text{elseif } j_1 \geq N_\phi \text{ and } j_2 \geq N_\phi \]
\[ j_1 = N_\phi; j_2 = N_\phi; \]
\[ \text{wrap} = -1; \]
\[ \text{elseif } j_1 < 0 \text{ and } j_2 < 0 \]
\[ j_2 = N_\phi; j_2 = N_\phi; \]
\[ \text{wrap} = 1 \]
\[ \text{end} \]
\[ \text{last}j = j_1 \]
\[ \text{cumwrapphase} += \text{wrap} \]
\[ \hat{\omega} = \Delta \phi(1 - \alpha) + \alpha \text{lastomega} \]
\[ \text{lastomega} = \hat{\omega} \]
\[ \text{Adjust the phase adjustment step size} \]
\[ \phi \text{stepsize} = \lceil|\hat{\omega}|/\delta\phi \rceil + 1 \]
\[ \text{end function} \]

6. Huber Loss Function

The fourth moments computed in the kurtosis open the algorithm to the vulnerability that outlier events unduly affect the estimate. This effect can be mitigated by employing a Huber loss function [5], which is commonly used for robust estimation. This function is defined as follows:

\[ H_{\delta_H}(a) = \begin{cases} \frac{1}{2}a^2 & \text{if } |a| \leq \delta_H, \\ \delta_H(|a| - \frac{1}{2}\delta_H) & \text{otherwise.} \end{cases} \]

\( H_{\delta_H}(a) \) behaves quadratically for small values of \( a \) (when \(|a| \leq \delta_H\)) and linearly for larger values of \( a \), with the transition defined such that the transition from quadratic behavior to linear behavior at \(|a| = \delta_H\) is continuous and continuously differentiable. The real kurtosis of (3) is approximated using the Huber function by

\[ K_{\phi}^{H}(y) = E[H_{\delta_H}(H_{\delta_H}(y))] - 3H_{\delta_H}(E[H_{\delta_H}(y)]) \]

In complex kurtosis, for the terms involving the magnitude \(|y|\), the Huber function can be applied directly. For the term involving \(y^2\), the Huber function is applied to the magnitude, leaving the phase quadratically varying. The complex Kurtosis is approximated using the Huber function as follows:

\[ K_{\phi}^{H}(y) = E[H_{\delta_H}(H_{\delta_H}(|y|))] - 2H_{\delta_H}(E[H_{\delta_H}(|y|)]) - H_{\delta_H}\left(E[H_{\delta_H}(|y|)|y|^{2/2}]\right). \]

This is estimated from a sequence of observations:
\[
\hat{K}_c^H(y) = \frac{1}{N_{\text{kurt}}} \sum_{n=1}^{N_{\text{kurt}}} H_{\delta_H}(H_{\delta_H}(|y(n)|)) - 2H_{\delta_H}\left(\frac{1}{N_{\text{kurt}}} \sum_{n=1}^{N_{\text{kurt}}} H_{\delta_H}(|y(n)|)\right) - H_{\delta_H}\left(\frac{1}{N_{\text{kurt}}} \sum_{n=1}^{N_{\text{kurt}}} H_{\delta_H}(|y(n)|)e^{2j2\pi y(n)}\right).
\]

Huber functions can be used in Algorithm 1 by simply replacing the computations of \(K_1\) and \(K_2\) at lines 45 and 46 with the complex Huber function, and the computations of \(J_1\) and \(J_2\) at lines 85 and 86 with the real Huber function.

7. Gradient Descent for Phase Estimation

As an alternative to optimizing over a fixed number of alternatives, gradient descent may also be employed. We illustrate the method here for phase estimation; this can be modified for timing estimation.

Let the real and imaginary components of the rotated signal be denoted as

\[
\begin{bmatrix}
y_{r,\text{rot}} \\
y_{i,\text{rot}}
\end{bmatrix} = \begin{bmatrix} c(\hat{\phi})y_r + s(\hat{\phi})y_i \\ -s(\hat{\phi})y_r + c(\hat{\phi})y_i \end{bmatrix}
\]

where \(c(\hat{\phi}) = \cos(\hat{\phi})\) and \(s(\hat{\phi}) = \sin(\hat{\phi})\). The objective function is the sum of the real kurtoses of the real and imaginary parts. Expanding this and identifying the moment terms using the variables \(A\) through \(D\), we find

\[
J(\hat{\phi}) = K_r(y_{r,\text{rot}}) + K_r(y_{i,\text{rot}}) = E[(c(\hat{\phi})y_r + s(\hat{\phi})y_i)^2] - 3E[(c(\hat{\phi})y_r + s(\hat{\phi})y_i)^2]^2 + E[(-s(\hat{\phi})y_r + c(\hat{\phi})y_i)^2] - 3E[(-s(\hat{\phi})y_r + c(\hat{\phi})y_i)^2]^2
\]

\[
= c(\hat{\phi})^4 \left( E[y_r^4] - 3E[y_r^2]^2 + 4E[y_i^4] - 3E[y_i^2]^2 \right)
\]

\[
+ c(\hat{\phi})^3 s(\hat{\phi}) \times \left( 4E[y_r^2 y_i] - 4E[y_r y_i^2] - 12E[y_r^2]E[y_r y_i] + 12E[y_i^2]E[y_r y_i] \right) \tag{A}
\]

\[
+ c(\hat{\phi})^2 s(\hat{\phi})^2 \left( 12E[y_r^2 y_i^2] - 12E[y_r y_i]E[y_i^2] \right) \tag{B}
\]

\[
+ c(\hat{\phi}) s(\hat{\phi})^3 \times \left( 4E[y_r y_i^3] - 12E[y_r y_i]E[y_r^3] - 4E[y_i^3]E[y_r y_i] + 12E[y_i^3]E[y_r y_i] \right) \tag{C}
\]

\[
+ s(\hat{\phi})^4 \left( E[y_r^4] + E[y_i^4] - 3E[y_r^2]^2 - 3E[y_i^2]^2 \right) \tag{D}
\]

The gradient of \(J(\hat{\phi})\) is

\[
\frac{\partial J(\hat{\phi})}{\partial \hat{\phi}} = A(4s(\hat{\phi})^3c(\hat{\phi}) - 4c(\hat{\phi})^3s(\hat{\phi}))
\]

\[
+ B(c(\hat{\phi})^4 - 3s(\hat{\phi})^2c(\hat{\phi})^2) +
\]

\[
C(-2c(\hat{\phi})s(\hat{\phi})^3 + 2c(\hat{\phi})^3s(\hat{\phi})) + D(3c(\hat{\phi})^2s(\hat{\phi})^2 - s(\hat{\phi})^4)
\]
8. Experimental Results

Kurtosis-based estimation was compared with the PLL-based timing and phase sync algorithms of [3] (Sections 7.4 and 8.4.4). In these algorithms, the timing interpolator is governed by the fractional symbol \( \mu \). The phase estimate is represented by the DDS (direct digital synthesizer) value. Two examples are presented here, one with \( \omega_{\text{off}} = 0 \) and one with \( \omega_{\text{off}} = 0.01 \).

**Example 1.** QPSK, SNR = 10 dB, excess bandwidth=0.2. \( \omega_{\text{off}} = 0 \). Algorithm parameters: \( N_{\text{kurt}} = 20, N_\tau = 60, N_\phi = 40, Q = 101. \omega\)-filtering parameter \( \alpha = 0.99 \).

Figure 3a shows the results of the delay estimation. The inset in the figure shows the first 30 samples, illustrating that convergence has occurred in about 10 symbols periods. Figure 3b shows the phase estimation. The phase estimate (top, in radians) shows the phase estimate. The inset shows that the phase estimate has converged in less than 10 symbols. The \( \Delta\)-phase (middle) shows \( \Delta \phi \). The bottom plot shows the estimate of \( \omega \).

By comparison, Figure 4a shows the PLL-based estimates, with a filter time-bandwidth product \( B_N T = 0.01 \). The top plot shows the fractional symbol \( \mu \), demonstrating convergence somewhere around 200 symbols. The bottom plot shows the phase converges in about 100 symbols.

As another method of comparing performance, the matched filter outputs were clustered (according to nearest signal constellation point) and the average variance of the clusters was computed. This variance was compared with the variance that would occur if the only source of noise were the AWGN. These variance results are shown in Table 1. In the column labeled “Noise Var”, the variance due to the AWGN at that SNR is shown. For example, in the first row where \( E_b / N_0 = 10 \text{ dB} \), the noise variance is 0.05. The column labeled “No Offsets” shows the result of estimating the variance of the clustered matched filter outputs when there are no phase or timing offsets. When \( E_b / N_0 = 10 \text{ dB} \), this is 0.05. This value should be close to the “Noise Var.” The columns labeled “\( \delta_H \)” and “Grad \( \mu \)” indicate the settings for the parameters. “\( \delta_H \)” set to a value indicates that the Huber function was used. “Grad \( \mu \)” set to a value indicates that gradient descent was used. The column labeled “Kurtosis” indicates the variance of the clustered matched filter outputs for the various kurtosis-based estimation algorithms after convergence. For example, the first row with \( \delta_H = 0.5 \) shows that the variance is 0.061. This can be compared with the variance in the “No Offsets” column, indicating that using the estimate does, in fact, increase the variance of the matched filter outputs compared to not having to estimate the parameters at all. The column labeled “PLL-based” shows the variance of the clustered matched filter outputs for the PLL-based estimators. For \( E_b / N_0 = 10 \text{ dB} \), this variance is 0.06. Finally, the column “dB(Kurt/PLL)” shows the comparison (in dB) between the kurtosis and PLL-based estimators, where negative numbers indicate superiority of kurtosis-based vs. PLL-based. For example, in the first row, kurtosis-based estimation is 0.06 dB worse (in variance) than PLL-based estimation.

As Table 1 shows that the variance performance between kurtosis-based and PLL-based estimation is generally quite close.

**Example 2.** This example shows behavior typical of the kurtosis-based phase estimate. In this case, \( N_\phi = 100 \), and SNR = 24 dB. Figure 5 demonstrates an example of the phase estimate. After convergence, the estimator tends to jitter around the bottom of the kurtosis “bowl”, because the kurtosis is only estimated. The phase variance thus largely determined by the steps size, essentially \( 2\pi / N_\phi \).
(a) Delay estimation. Inset: first 30 symbols.

(b) Phase estimation. Top: $\hat{\phi}$. Middle: $\hat{\phi}(n) - \hat{\phi}(n - 1)$. Bottom: $\hat{\omega}$.

**Figure 3.** Example 1: Kurtosis-based timing and phase estimation.
Figure 4. Example 1: PLL-based timing and phase estimation.
Table 1. Cluster variances

| Example | $E_b/N_0$ (dB) | Noise Var | No Offsets | $\delta_H$ | Grad $\mu$ | Kurtosis | PLL-Based | dB(Kurt/PLL) |
|---------|----------------|-----------|------------|------------|------------|----------|-----------|--------------|
| 1-9 1   | 10             | 0.05      | 0.05       | 0.5        | —          | 0.061    | 0.06      | 0.06         |
|         |                |           |            |            | —          | 0.077    | 1.08      |              |
|         |                |           |            | 0.5        | 0.01       | 0.054    | —0.47     |              |
| 1-9 8   | .079           | .079      | 0.5        | —          |            | 0.086    | 0.94      | −0.38        |
|         |                |           |            |            | —          | 0.105    | 0.48      |              |
|         |                |           |            | 0.5        | 0.01       | 0.128    | −0.60     |              |
| 1-9 6   | .126           | .124      | 0.5        | —          |            | 0.13     | 0.15      | −0.47        |
|         |                |           |            |            | —          | 0.15     | 0.016     |              |
|         |                |           |            | 0.5        | 0.01       | 0.127    | −0.64     |              |
| 1-9 4   | .2             | .19       | 0.5        | —          |            | 0.19     | 0.22      | −0.55        |
|         |                |           |            |            | —          | 0.22     | 0.056     |              |
|         |                |           |            | 0.5        | 0.01       | 0.19     | −0.62     |              |
| 1-9 2   | 10             | 0.05      | 0.05       | 0.5        | —          | 0.057    | 0.6       | −0.20        |
|         |                |           |            |            | —          | 0.058    | −0.07     |              |
|         |                |           |            | 0.5        | 0.2        | 0.21     | 5.4       |              |
| 1-9 8   | .079           | .079      | 0.5        | —          |            | 0.085    | 0.093     | −0.42        |
|         |                |           |            |            | —          | 0.088    | −0.28     |              |
|         |                |           |            |            | —          | 0.23     | 3.91      |              |
| 1-9 6   | .079           | .079      | 0.5        | —          |            | 0.13     | 0.15      | −0.49        |
|         |                |           |            |            | —          | 0.15     | 0.16      |              |
|         |                |           |            |            | —          | 0.25     | 2.3       |              |
| 1-9 4   | .2             | .19       | 0.5        | —          |            | 0.2      | 0.22      | −0.51        |
|         |                |           |            |            | —          | 0.2      | −0.49     |              |
|         |                |           |            |            | —          | 0.28     | 1.04      |              |

Figure 5. Example 2: Illustration of phase estimate jitter, $N_\phi = 100$.

Example 3. QPSK, SNR = 10 dB, excess bandwidth = 0.2. $\omega_{\text{off}} = 0.02$. Algorithm parameters: $N_{\text{kurt}} = 20$, $N_T = 60$, $N_\phi = 40$. $\omega$-filtering parameter $\alpha = 0.95$.

Figure 6a shows the results of the delay estimation. The inset in the figure shows the first 30 samples, illustrating that convergence occurred in less than 20 symbols. Figure 6b shows the phase estimation. The phase estimate (top, in radians) shows the phase estimate. The inset shows that the phase estimate converged in less than 10 symbols. The $\Delta$-phase (middle) shows $\Delta \phi$. The bottom plot shows the estimate of $\omega$.

Figure 7 shows results for the PLL-based methods in this setting. While the phase tracking appears smoother in the PLL-based estimate, as the results for Example 3 in Table 1 show, there is less variance around the constellation points at the matched filter outputs for the kurtosis-based methods than the PLL-based methods when step descent is used. Interestingly, the gradient descent performs significantly worse than PLL-based estimation at all SNRs.
Figure 8 shows results for the gradient descent estimation of the phase. The convergence time is about the same as the step descent, but there is higher variance on the estimate of $\omega$.

(a) Delay estimation. Inset: first 30 symbols

(b) Phase estimation.

Figure 6. Example 3: Kurtosis-based estimates.
Figure 7. Example 3: PLL-based timing and phase estimation.
9. Comparison with Modified Cramer–Rao Lower Bound

Evaluating the performance in terms of how the recovered signal points are clustered, as in Table 1, is a natural way to evaluate the performance of these estimation algorithms, since it demonstrates how all of the estimators work together to achieve what is desired in the receiver: good signal detection. Another way of evaluating performance of estimators is to compare the estimator variance against lower bounds such as the Cramer–Rao lower bound (CRLB) or modified CRLB. The modified CRLB provides a bound when the observed data depends on multiple parameters, but only one parameter at a time is estimated [22]. The modified CRLB is, in general, lower than the CRLB.

The modified CLRB for the estimate of $\phi$ is [22] (Equation (31))

$$\text{MCRB}(\phi) = \frac{B_L T_s}{E_s/N_0}$$

In [22], $B_L = 1/(2LT_s)$, where $LT_s$ is the length of window over which the estimator operates. The modified CLRB for the estimate of $\tau$ is [22] (Equation (32))

$$\text{MCRB}(\tau) = \frac{B_L T_s T_s^2}{4\pi^2 \xi E_s/N_0}$$

where

$$\xi = \frac{\int_{-\infty}^{\infty} f^2 |P(f)|^2 df}{\int_{-\infty}^{\infty} |P(f)|^2 df},$$

where $P(f)$ is the Fourier transform of the pulse-shaping function $p(t)$. 
The kurtosis is computed over $N_{\text{kurt}}$ symbols (matched filter outputs). The first matched filter output occurs at sample $QT_s/P$, with symbol outputs occurring every $T_s$ seconds thereafter. Thus, the duration over which a kurtosis is computed is $QT_s/P + (N_{\text{kurt}} - 1)T_s$. We take this as the value of $B_L T_s$ over which the estimate is computed. Since the estimator steps over several symbol times to converge, this does not apply initially but should be applicable after convergence.

Figure 9a shows the variance of estimate of $\tau$ using the kurtosis-based method (red) and the conventional (loop-based) method (yellow) compared with the modified CRLB (blue), as a function of $E_b/N_0$. The variance of the kurtosis-based method was obtained by computing the variance of $\hat{\tau}$ after convergence, such as seen in Figure 3a, averaged over 10 independent runs. The variance of the loop-estimated variance was computed as the variance of $\mu$ (the fractional timing offset in the synchronization algorithm [3]), averaged over 10 runs. The kurtosis-based method performs significantly better than the loop-based method and demonstrates variance decreases with SNR. However, it does not decrease as fast as the modified CRLB. Additionally, it appears that the kurtosis-based method meets the modified CRLB, which is unexpected. It may be that the value for $B_L T_s$ used to compute the bound in (7) is larger than it should be.

Figure 9b shows the variance of the estimate of $\phi$ using kurtosis (red) and PLL (yellow) compared with the modified CRLB (blue). The variance of the kurtosis-based method was obtained by computing the variance of $\hat{\phi}$ after convergence, such as seen in Figure 3b, averaged over 10 independent runs. The variance of the kurtosis method hardly changes with SNR (although it can be seen to vary slightly). The continued variation is due to the fixed step size, and the fact that the kurtosis estimate has some variance associated with it causing jitter in the indices. Depending on the value of $N_{\phi}$, this can achieve variances less than those of the PLL-based method. This suggests that a variable $N_{\phi}$ algorithm would be useful: use a larger step size to converge quickly, then reduce the step size to reduce the variance.

(These plots were performed using QPSK, $Q = 101$, $P = 3$, $\alpha = 0.2$, and $N_{\text{kurt}} = 40$ and using the Hubert function to estimate the kurtosis.)

10. Discussion and Conclusions

We demonstrated that kurtosis-based methods can be applied to symbol timing and phase estimation. These offer potential advantages such as being able to synchronize without knowledge of the signal constellation in a scale-invariant way and, generally, convergence as fast as or faster than PLL-based methods. The experiments showed that the variance of the matched filter outputs for kurtosis-based methods is about the same as or slightly better than PLL-based methods. The gradient descent phase estimation, however, does not generally perform as well as discrete minimization methods. We made the following observations:
• Using the Huber function instead of the full kurtosis reduces the variance of the estimators.
• For lower SNRs, the gradient-based method performs significantly worse than the discrete minimization.
• The gradient-based method has a higher complexity than the discrete minimization.
• Kurtosis-based methods converge more quickly than PLL-based methods. The number of steps is determined primarily by $N_\tau$, $N_\phi$, and $n_r$. (There is a secondary effect due to the bandwidth of the lowpass filter for $\hat{\omega}_{\text{off}}$.)

The fast convergence time suggests that these kurtosis-based methods may be useful in situations where short packets are used, such as in the Internet of Things.

Author Contributions: Conceptualization, T.K.M. and J.H.G.; methodology, T.K.M. and J.H.G.; software, T.K.M.; validation, T.K.M.; formal analysis, T.K.M.; writing—original draft preparation, T.K.M.; writing—review and editing, T.K.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not Applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Gunther, J.H.; Moon, T.K. Burst mode synchronization of QPSK on AWGN channels using kurtosis. IEEE Trans. Comm. 2009, 57, 2453–2462. [CrossRef]
2. Nelder, J.A.; Mead, R. A Simplex Method for Function Minimization. Comput. J. 1965, 7, 308–313. [CrossRef]
3. Rice, M. Digital Communications: A Discrete-Time Approach, 2nd ed.; Prentice Hall: Hoboken, NJ, USA, 2017; ISBN 9781790588545.
4. Marazin, M.; Gautier, R.; Burel, G. Dual Code Method for Blind Identification of Convolutional Encoder for Cognitive Radio Receiver Design. In Proceedings of the 2009 IEEE Globecom Workshops, Honolulu, HI, USA, 30 November–4 December 2009; pp. 1–6; [CrossRef]
5. Huber, P.J. Robust Estimation of a Location Parameter. Ann. Math. Statist. 1964, 35, 73–101. [CrossRef]
6. Delfosse, N.; Loubaton, P. Adaptive blind separation of convolutive mixtures. In Proceedings of the 1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings, Atlanta, GA, USA, 9 May 1996; pp. 2940–2943.
7. Hyvarinen, A. Fast and robust fixed-point algorithms for independent component analysis. IEEE Trans. Neural Netw. 1999, 10, 626–634. [CrossRef] [PubMed]
8. Hyvarinen, A.; Oja, E. A fast fixed-point algorithm for independent component analysis. Neural Comput. 1997, 9, 1483–1492. [CrossRef]
9. Hyvärinen, A.; Karhunen, J.; Oja, E. Independent Component Analysis; Wiley: New York, NY, USA, 2001.
10. Viterbi, A. Principles of Coherent Communication; McGraw-Hill: New York, NY, USA, 1966.
11. Stiffler, J. Theory of Synchronous Communication; Prentice-Hall: Englewood Cliffs, NJ, USA, 1971.
12. Lindsey, W. Synchronization Systems in Communication and Control; Prentice-Hall: Englewood Cliffs, NJ, USA, 1972.
13. Lindsey, W.; Simon, M. Telecommunication Systems Engineering; Prentice-Hall: Englewood Cliffs, NJ, USA, 1973.
14. Gardner, F. PhaseLock Techniques; Wiley: New York, NY, USA, 1979.
15. Meyr, H.; Asheid, G. Synchronization Techniques in Digital Communications; Wiley: New York, NY, USA, 1990.
16. Mengali, U.; D’Andrea, A. Synchronization Techniques for Digital Receivers; Springer Science+Business Media: New York, NY, USA, 1997.
17. Proakis, J.G. Digital Communications; McGraw Hill: New York, NY, USA, 1995.
18. Barry, J.R.; Lee, E.A.; Messerschmitt, D.G. Digital Communication, 3rd ed.; Kluwer Academic: Boston, MA, USA, 2004.
19. Franks, L. Carrier and Bit Synchronization in Data Communication—A Tutorial Review. IEEE Trans. Comm. 1980, 28, 1107–1121. [CrossRef]
20. Ding, Z.; Li, Y. Blind Equalization and Identification; Marcel Dekker: New York, NY, USA, 2001.
21. Shalvi, O.; Weinstein, E. New criteria for blind deconvolution of non-minimum phase systems. IEEE Trans. Info. Theory 1990, 36, 312–321. [CrossRef]
22. D’andrea, A.; Mengali, U.; Reggiannini, R. The Modified Cramer-Rao Bound and Its Application to Synchronization Problems. IEEE Trans. Comm. 1994, 42, 1391–1399. [CrossRef]