A causal time domain filter for monitoring and subtraction of coherent signals in continuous data streams and control loops

E. J. Daw‡

University of Sheffield, Department of Physics and Astronomy, Hicks Building, Hounsfield Road, Sheffield S3 7RH, UK

E-mail: `e.daw@shef.ac.uk`

Abstract. Detectors and control systems frequently suffer from the contamination of data with coherent backgrounds. A previous paper [1] discussed an algorithm for subtraction of coherent backgrounds using an acausal filter. This algorithm (EFC) is useful for the removal of lines in the data outputs of the detectors, but its acausal nature and resultant group delay makes it unsuitable for incorporation into the control loops determining the states of the instruments. In this internal note I discuss a related causal algorithm, EAC, and give a case study on how this algorithm might be incorporated into a servomechanism to suppress coherent backgrounds. The algorithm also lends itself to efficient implementation of digital phase locked loops and the analysis of data having irregularly spaced samples, and thus has a broad range of applications in science, engineering and data analysis.

1. Introduction

The EFC algorithm [1] permits sample-by-sample estimation of the amplitude and phase of a signal at the frequency of a coherent background. The measured amplitude and phase can then be used to synthesize a signal having the same properties at that frequency, which can then be subtracted from the data to suppress its coherent oscillation component. The EFC algorithm is based upon an iteration equation for the Fourier coefficients of a data set, where each new iteration introduces a single data sample, x_N, to the end of the timeseries, and removes a single data sample, x_0 from the beginning. Using a previous estimate of the Fourier coefficient, $\mathcal{F}_k(x_0,\, x_1,\, x_2,\, \cdots,\, x_{N-1})$, the EFC iteration algorithm for the k^{th} Fourier coefficient is

$$\mathcal{F}_k(x_1,\ x_2, \cdots, x_N) = e^{\frac{+2\pi ik}{N}} \left(\mathcal{F}_k(x_0,\ x_1, \cdots, x_{N-1}) + (x_N - x_0)\right). \qquad (1)$$

Here the number N of samples per Fourier transform times the sampling period t_s should be greater than the timescale for signals that should be preserved, but less than the timescale for fluctuations in the amplitude and phase of the coherent background.

‡ To whom all correspondence should be addressed

This Fourier coefficient is the best estimator of the amplitude and phase of the coherent background at the mid point of the timeseries, so the estimator is subject to a group delay of approximately $Nt_s/2$ with respect to the data. This group delay limits the applicability of the algorithm in applications where background subtracted data might be used to control the instrument in real time as part of a closed loop servomechanism. We therefore seek a related algorithm with zero group delay.

2. The exponentially averaged coefficients (EAC) algorithm

The EFC iteration algorithm becomes a boxcar averager in the case where $k = 0$, and not surprisingly boxcar averagers are subject to the same group delay of half the data length of the box. An exponential average is a related average that has no group delay. The iteration algorithm of an exponential averager is

$$y_n = (1 - w)y_{n-1} + wx_n, \tag{2}$$

where the n^{th} input (output) data sample is x_n (y_n), and w is a number between 0 and 1. The exponential averager calculates the weighted sum of all previous input data samples,

$$y_n = \sum_{m=0}^{\infty} x_{n-m} e^{\frac{-(n-m)t_s}{\tau_w}}, \tag{3}$$

where τ_w is the averaging time constant of the filter, related to w by $\tau_w \simeq t_s/w$. By tuning w we can define a timescale for the average, though unlike the boxcar averager, all previous inputs have an influence on the current output.

Attempting to write down an iterative estimator for Fourier coefficients that, like the exponential averager, has zero group delay, leads us to the following iteration formula,

$$y_n = e^{i\Delta}\left((1 - w)y_{n-1} + wx_n\right), \tag{4}$$

where Δ is a real phase, related to the frequency of the coherent background by $\Delta = 2\pi f t_s$. In Appendix A we show that when the input is an oscillation of angular frequency $\omega = 2\pi f$, the output y_n of the filter tends after a time much larger than τ_w to points tracing out an elliptical path in the complex plane, centered on the origin. Figure 1 shows the output of the iteration algorithm for a choise of w and Δ such that the filter response time is a few cycles of the input sinusoid.

The rotation period of the y_n is equal to the period $2\pi/f$ of the sinusoidal input. We show that the output y_n can be used to generate an oscillation having the same amplitude and phase as the input x_n. As for the EFC algorithm, signals not at the frequency f are suppressed in the filter output, as are signals at frequency f but varying in amplitude or phase on a timescale significantly less than τ_w. The filter is causal in that only the current input data and current plus a single previous sample of the output data is used to synthesize the sine wave.

The EAC algorithm consists of the above iteration formula and in addition the method for using the algorithm output to synthesize a real sinusoidal oscillation having

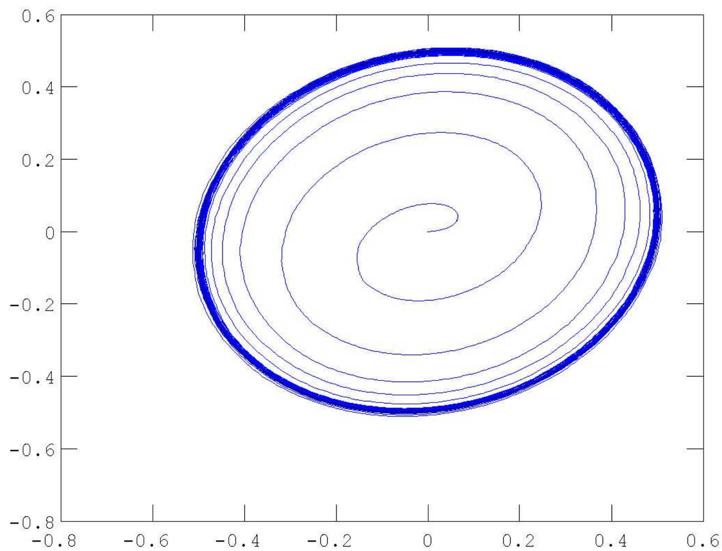

Figure 1. The output of the iteration algorithm for input $x(n) = \cos(n\Delta)$ plus some gaussian noise, starting at n=0, and zero beforehand. The filter output is in blue, the predicted limiting ellipse is shown in green with good agreement.

the same amplitude and phase as the input. The method is to make a linear combination of the real and imaginary parts of y_n. We define the following quantities:

$$\begin{aligned}
A_f &= \tfrac{1}{2}; \\
A_b &= \frac{1}{2\sqrt{1-2(1-w)\cos\Delta+(1-w)^2}}; \\
\phi &= \Delta + \arctan\left(\frac{(1-w)\sin 2\Delta}{1-(1-w)\cos 2\Delta}\right); \\
\alpha &= \frac{\Delta-\phi}{2}; \\
\beta &= \frac{\Delta+\phi}{2}.
\end{aligned} \tag{5}$$

It is shown in Appendix A that the following linear combination of the real and imaginary parts of y_n is a real sinusoidal oscillation having the same amplitude, phase and frequency as the input signal x_n.

$$y_n^R = \frac{\cos\alpha}{A_f + A_b}\Re\{e^{-i\beta}y_n\} + \frac{\sin\alpha}{A_f - A_b}\Im\{e^{-i\beta}y_n\}. \tag{6}$$

Here $\Re$ and $\Im$ denote the real and imaginary parts of their arguments.

The synthesized cosine wave can be subtracted from the input drive, which must include a sinusoid at the filter frequency, but could also include broadband noise. Figure 2 shows the result of the subtraction for the input signal and Gaussian noise used to generate Figure 1. Subtraction is by eye quite effective after the ring-up time of the iteration algorithm.

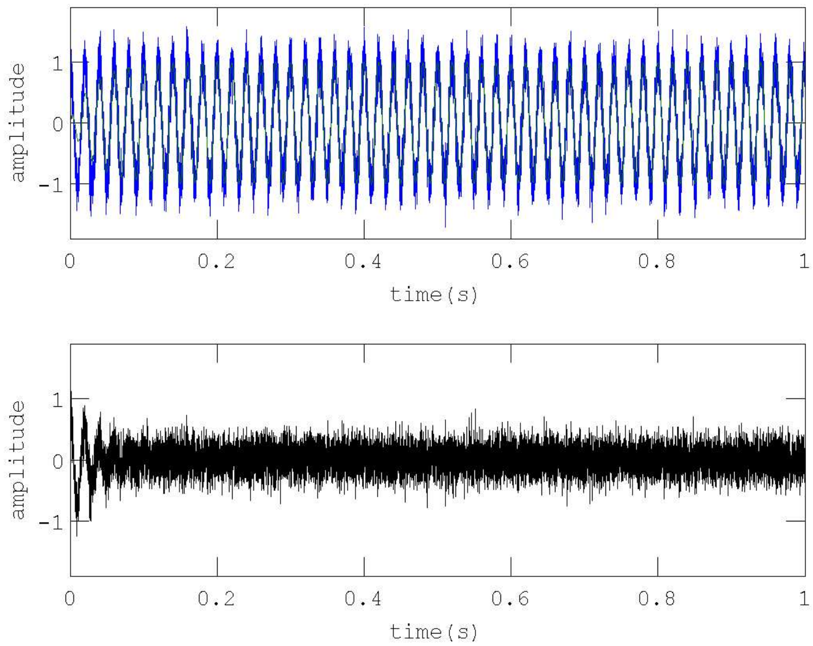

Figure 2. The upper pane in this figure shows (in blue) the input signal for a test of the line subtraction with EAC. The blue curve is a line plus Gaussian noise. The green curve is the I phase output of the algorithm as defined in Appendix A. The lower pane shows the difference between the two, with the line clearly suppressed after the response time of the filter, a few cycles for this choice of w and Δ.

3. Application to feedback control systems

Figure 3 shows a generic feedback control loop operating on some plant. Blocks labelled D denote the plant, blocks labelled C denote portions of the feedback controller. We consider the case where a coherent background N_i enters the plant at some point. Portions of the plant before (after) the point of entry of the noise are denoted by D_B (D_A). The feedback controller is split into two components, a conventional frequency dependent controller C_F, and a new controller, C_N whose purpose is to minimize the coherent signal, N_o appearing at the plant output.

The coefficient w is made sufficiently small so that the transfer function $C_N(\omega)$ is sharply peaked at $2\pi f$. Therefore this portion of the controller only has a significant effect on the controller at frequency f. The transfer function $G(\omega)$ for the coherent signal is given by

$$G(\omega) = \frac{N_o(\omega)}{N_i(\omega)} = \frac{D_A(\omega)}{1 - D_A\omega D_B\omega \left(C_N(\omega) + C_F(\omega)\right)}. \quad (7)$$

Here $D_B(\omega)$, $D_A(\omega)$, $C_F(\omega)$ and $C_N(\omega)$ are the complex transfer functions of the different elements of the loop at frequency ω. We assume that $D_B(\omega)$, $D_A(\omega)$ and $C_F(\omega)$ are known. Errors in the estimates of these quantities will degrade the performance of

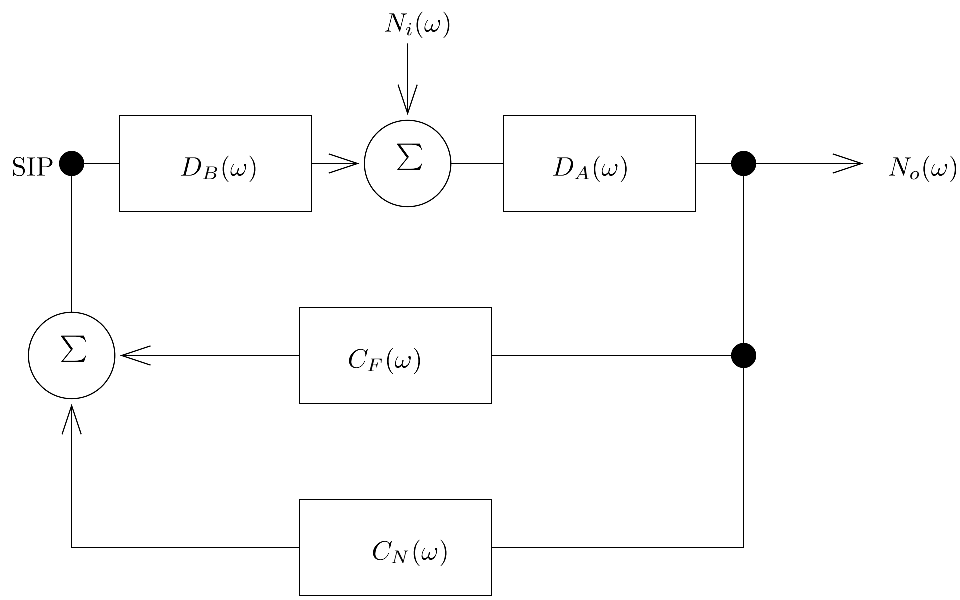

Figure 3. A controlled system with feedback and noise coupling in to a point in the plant at frequency $f = 2\pi\omega$. The input to the plant is at the label SIP, then noise injected is $N_i(\omega)$. The noise content of the plant output is $N_o(\omega)$. The feedback controller is split into two components, C_F providing broadband control and $C_N(\omega)$ the additional new controller component aiming to suppress the component at frequency f at the plant output.

the suppression algorithm.

Good suppression of the coherent component at the servo output is achieved when $A = 1/G \gg 1$. We also require that G is real, so that the remaining coherent signal at the output is in phase with the input signal. The requirement of a particular A and a real transfer function G are sufficient to constrain the amplitude and phase of $C_N(\omega)$. The amplitude and phase of $C_N(\omega)$ are denoted by C_N and ε respectively, so that $C_N(\omega) = C_N \exp(i\varepsilon)$. The amplitude and phase of the response of the EAG algorithm can be set by inserting extra amplitude and phase factors into Equation 6, yielding

$$y_n^R = \frac{C_N \cos\alpha}{A_f + A_b}\Re\{e^{(-i\beta+i\varepsilon)}y_n\} + \frac{C_N \sin\alpha}{A_f - A_b}\Im\{e^{(-i\beta+i\varepsilon)}y_n\}. \tag{8}$$

This signal leads the input waveform by a phase of ε radians and has an amplitude C_N times the amplitude of the input waveform. To determine C_N and ε that achieves the required attenuation of coherent background at the output, we first split the transfer functions of the other elements of the control system into their magnitudes and phases.

$$\begin{aligned} D_A(\omega) &= D_A e^{i\rho} \\ D_B(\omega) &= D_B e^{i\sigma} \\ C_F(\omega) &= C_F e^{i\lambda} \\ D_A(\omega)D_B(\omega) &= De^{i(\rho+\sigma)}. \end{aligned} \tag{9}$$

The gain G of the servo may now be expressed as

$$\left[\frac{G}{D_A}\right] = \frac{1}{A} = \frac{1}{e^{-i\rho} - DC_F e^{i(\sigma+\lambda)} - DC_N e^{i(\sigma+\varepsilon)}}. \tag{10}$$

Equating the real and imaginary parts of Equation 10 we obtain

$$\begin{aligned} \sin\rho + DC_F\sin(\sigma+\lambda) + DC_N\sin(\sigma+\varepsilon) &= 0 \\ \cos\rho - DC_F\cos(\sigma+\lambda) - DC_N\cos(\sigma+\varepsilon) &= A. \end{aligned} \tag{11}$$

Solving simultaneously we obtain expressions for the amplitude and phase of the EAC algorithm

$$\begin{aligned} \varepsilon &= \arctan\left(\tfrac{DC_F\sin(\sigma+\lambda)+\sin\rho}{A+DC_F\cos(\sigma+\lambda)-\cos\rho}\right) - \sigma \\ C_N &= \tfrac{1}{D}(A^2 + D^2C_F^2 + 1 - 2DC_F\cos(\rho+\sigma+\lambda) + \\ &\quad 2A(DC_F\cos(\sigma+\lambda) - \cos\rho))^{\frac{1}{2}}. \end{aligned} \tag{12}$$

If we set C_N and ε using Equations 12 then we expect the amplitude of the plant output at frequency $2\pi\omega$ to be $1/A$ times the amplitude of the input N_i.

4. Application to monitoring of irregularly sampled data

This application is not yet developed. It applies to data where we have data samples at an arbitrary set of times, but we know the timestamps of each data sample. In this case, we may set the phase shift Δ between successive data samples in the sequence according to the known phase delay between the samples at a certain frequency. The result will be an oscillator at that frequency driven by the frequency component of the data at that frequency. This may be useful for determining a particular Fourier coefficient of an irregularly sampled data set from a set of irregularly sampled data points. Applications in LIGO include deriving information from channels when the interferometer is falling in and out of lock frequently, and possibly to slow controls data which is irregularly sampled due to the nature of the slow controls DAQ.

5. Application to digital phase locked loops

From the theory in Appendix Appendix A a sinusoidal input to the iteration algorithm at the frequency $f = \Delta/2\pi t_s$ can be used to synthesize a phasor whose argument advances by Δ at each sampling time. The output $y(n)$, multiplied by a phase $e^{-i\Delta}$ would therefore be expected to have a stationary phase. We could imagine measuring the phase shift of $y(n)e^{-in\Delta}$ between successive samples and using this variable to feed back to the phase in the iteration algorithm. This is effectively a phase locked loop as long as the frequency shift in the line driving the iteration algorithm is sufficiently slow to allow locking to the line to be maintained. The advantage of this idea over conventional digital phase locked loops (DPLL) is that no externally synthesized oscillator is necessary. The investigation of this possibility for DPLLs will naturally follow the analysis of conventional phase locked loops in a treatment such as that of Gardener. Applications of this idea include tracking and subtraction of lines having frequencies that drift significantly with time.

6. Conclusions

We have shown that an iteration algorithm implemented as a complex time-independent IIR filter on an input data stream can be used to synthesize and subtract sinusoids having arbitrary amplitude and phase from noisy input data containing coherent features at high amplitude. By analogy with the closely related exponential averaging algorithm, the response time τ of the iteration algorithm is given by $\tau = t_s/w$, where t_s is the sampling time.

Implementation and testing with real data remains to be done. Testing on simulated data has proceeded as far as verification that the algorithm can be used to subtract a single fixed frequency sinusoid from Gaussian noise, and verification that the oscillator $y(n)$ traces out the ellipse in the stationary state that is predicted by the analysis of Appendix A. No testing of the performance of EAC in a feedback loop has yet been attempted on either simulated or real data. The algorithm is sufficiently fast that testing on simulated data can be done entirely in matlab with no speed or CPU load problem. For testing on real data, the approach I have taken with EFC, where the algorithm is coded up as ANSI C code with wrappers into the DMT (as a derived filter class inheriting `FilterBase()` and `Pipe()`) would seem to be a good first step. Following this, testing of the algorithm on data could proceed using the DMT and some variable frequency line target, perhaps a violin mode resonance.

This document has been assigned LIGO DCC number LIGO-T080244-00-K.

Appendix A. Coefficients for output phase ellipse eccentricity, rotation, and phase shift

Consider the algorithm of Equation 4 as a difference equation for a circuit with an applied driving term. The driving term is an oscillatory input $x(n) = \cos(\omega t)$. We express this input in terms of a basis of complex exponentials in the usual way,

$$x(n) = \frac{1}{2}e^{+in\Delta} + \frac{1}{2}e^{-in\Delta}. \tag{A.1}$$

We consider the effect of the two exponential components of the oscillation as two separate driving terms. The left hand term is easier, because it turns out that the response of the difference equation to this drive has a phase shift with respect to the input drive that is independent of w. Write the solution in the presence of this drive as $y_n = A_F e^{in\Delta}$, where A_F may be complex. Substitute in to the iteration equation to obtain

$$\begin{aligned} A_F e^{in\Delta} &= e^{+i\Delta}\left((1-w)A_F e^{i(n-1)\Delta} + w e^{in\Delta}\right) \\ &= (1-w)A_F e^{+in\Delta} + w e^{in\Delta} e^{+i\Delta} \\ A_F &= (1-w)A_F + w e^{i\Delta} \\ A_F &= e^{+i\Delta}. \end{aligned} \tag{A.2}$$

Therefore the response of the iteration equation to the driving term $x(n) = e^{+in\Delta}$ may be written

$$y_F(n) = e^{+i(n+1)\Delta}. \tag{A.3}$$

The driving term $x(n) = e^{-in\Delta}$ is more troublesome. Define the response of the iteration equation to this driving term as $y(n) = A_B e^{-in\Delta+i\phi}$, where A_F and ϕ are real. Substitute in to the iteration equation again to yield

$$\begin{aligned} A_B e^{-in\Delta} e^{+i\phi} &= e^{+i\Delta}\left(A_B(1-w)e^{-i(n-1)\Delta} e^{+i\phi} + w e^{-in\Delta}\right) \\ A_B e^{+i\phi} &= A_B(1-w)e^{+2i\Delta} e^{+i\phi} + w e^{+i\Delta} \\ A_B &= A_B(1-w)e^{+2i\Delta} + w e^{+i(\Delta-\phi)}. \end{aligned} \tag{A.4}$$

Equating real and imaginary parts yields

$$\begin{aligned} A_B &= A_B(1-w)\cos 2\Delta + w\cos(\Delta-\phi) \\ 0 &= A_B(1-w)\sin 2\Delta + w\sin(\Delta-\phi) \end{aligned} \tag{A.5}$$

Eliminating A_B by division between these two equations yields

$$\begin{aligned} \tan(\phi-\Delta) &= \tfrac{(1-w)\sin 2\Delta}{1-(1-w)\cos 2\Delta} \\ \phi &= \Delta + \tan^{-1}\left(\tfrac{(1-w)\sin 2\Delta}{1-(1-w)\cos 2\Delta}\right). \end{aligned} \tag{A.6}$$

Squaring and adding Equations A.5 yields

$$\begin{aligned} w^2 &= A_B^2\left(1 - 2(1-w)\cos 2\Delta + (1-w)^2\right) \\ A_B &= \tfrac{w}{\sqrt{1-2(1-w)\cos 2\Delta + (1-w)^2}}. \end{aligned} \tag{A.7}$$

Therefore the response of the iteration equation to a driving term $x(n) = e^{-in\Delta}$ may be written

$$y_B(n) = A_B e^{-in\Delta+i\phi}, \tag{A.8}$$

with ϕ and A_B given in Equations A.6 and A.7. Therefore the response of the iteration equation to a real cosine wave $x(n) = \cos(n\Delta)$, through its expression in terms of exponentials through Equation A.1 is

$$y(n) = A_f e^{+in\Delta+i\Delta} + A_b e^{-in\Delta+i\phi}, \tag{A.9}$$

where $A_f = 1/2$ and $A_b = A_B/2$, A_B and ϕ being dependent on w through Equations A.2 and A.7. This derivation justifies the definitions of the first three quantities given in Equation 5.

We now show that the locus of the y_n in the complex plane tends to an ellipse centered on the origin in the steady state, and calculate the parameters of this ellipse in terms of w and Δ. Factor $e^{+i\Delta/2}$ and $e^{+i\phi/2}$ out of Equation A.9 to obtain

$$y(n) = e^{i\left(\frac{\Delta}{2}+\frac{\phi}{2}\right)} \left(A_f e^{+in\Delta} e^{+i\left(\frac{\Delta}{2}-\frac{\phi}{2}\right)} + A_b e^{-in\Delta} e^{-i\left(\frac{\Delta}{2}-\frac{\phi}{2}\right)} \right). \tag{A.10}$$

Using the definitions of α and β written down in Equation 5 and rearranging, this becomes

$$\begin{aligned} y_n e^{-i\beta} &= A_f e^{+in\Delta+\alpha} + A_b e^{-in\Delta-\alpha} \\ &= (A_f + A_b)\cos(n\Delta+\alpha) + i(A_f - A_b)\sin(n\Delta+\alpha). \end{aligned} \tag{A.11}$$

These Equations explain the final state ellipse traced out by the output of the iteration equation. The angle β is the angle between the major axis of the ellipse formed by the locus of $y(n)$ with respect to the real axis. Rotating the ellipse through angle $-\beta$ about the origin yields an ellipse with its semimajor axis along the real axis. This is shown explicitly by the second equation in A.11, the right hand side of which is the parameteric equation for such an ellipse. The angle α is the argument of the phase rotated ellipse at $n = 0$, when the excitation $x(n)$ is pure real, and is therefore the phase lead of the phase rotated output with respect to the phase of the drive waveform.

We now use the iterative equation output to synthesize oscillations at the original frequency and in phase with the drive. From Equation A.11 we can find the expressions for $\cos(n\Delta + \alpha)$ and $\sin(n\Delta + \alpha)$

$$\cos(n\Delta+\alpha) = \frac{\Re(y_n e^{-i\beta})}{A_f+A_b} \quad \text{and} \quad \sin(n\Delta+\alpha) = \frac{\Im(y_n e^{-i\beta})}{A_f-A_b}. \tag{A.12}$$

Addition formulae allow us to make linear combinations of these expressions as follows

$$\begin{aligned} \cos(n\Delta+\alpha) &= \cos(n\Delta)\cos\alpha - \sin(n\Delta)\sin\alpha \\ \sin(n\Delta+\alpha) &= \cos(n\Delta)\sin\alpha + \sin(n\Delta)\cos\alpha. \end{aligned} \tag{A.13}$$

Eliminating the terms in $\sin(n\Delta)$ between these two simultaneous equations yields

$$\begin{aligned} \cos(n\Delta) &= \cos\alpha\cos(n\Delta+\alpha) + \sin\alpha\sin(n\Delta+\alpha) \\ &= \frac{\cos\alpha}{A_f+A_b}\Re\{e^{-i\beta}y_n\} + \frac{\sin\alpha}{A_f-A_b}\Im\{e^{-i\beta}y_n\}. \end{aligned} \tag{A.14}$$

This is equation 6 for the synthesized replica of the real cosine drive input formed entirely from the output of the iterative equation. We may also eliminate terms in $\cos(n\Delta)$ to obtain a synthesized sinusoid that is in the Q phase quadrature with respect to the drive,

$$\begin{aligned} \sin(n\Delta) &= \cos\alpha\sin(n\Delta+\alpha) - \sin\alpha\cos(n\Delta+\alpha) \\ &= \frac{\cos\alpha}{A_f-A_b}\Im(e^{-i\beta}y_n) - \frac{\sin\alpha}{A_f+A_b}\Re(e^{-i\beta}y_n). \end{aligned} \tag{A.15}$$

Finally we give an expression for a sinusoid with an arbitrary phase lead ε with respect the drive $x(n)$. To this end we note that in Equation A.14, multiplying the output y_n by a phase factor $e^{-i\beta}$ rotated the phasor through an angle $-\beta$ with respect to the real axis. Therefore multiplying it by a phase factor $e^{+i\varepsilon}$ should add a phase lead of ε to the synthesized output. This means that an output of arbitrary phase lead ε with respect to the drive can be synthesized using the following linear combination of the real and imaginary parts of the iteration algorithm output.

$$\cos(n\Delta + \varepsilon) = \frac{\cos\alpha}{A_f + A_b}\Re\{e^{-i\beta+i\varepsilon}y_n\} + \frac{\sin\alpha}{A_f - A_b}\Im\{e^{-i\beta+i\varepsilon}y_n\}. \quad \text{(A.16)}$$

As a check, if we set $\varepsilon = -\pi/2$ we should obtain the Q phase result of Equation A.15. We get a factor of $-i$ from each of these substitutions into Equation A.16, and the real (imaginary) part of $-iz$ is $+(-)$ the imaginary (real) part of z. These transformations turn Equation A.16 into Equation A.15, which is what we would expect if retarding the I phase by $\pi/2$ yields the Q phase. This is Equation 8 used to incorporate EAC into a feedback controller for in loop line subtraction.

References

[1] LIGO DCC P080013.
[2] A.V. Oppenheim, R.W. Schafer, "Digital Signal Processing", ISBN 0-13-214635-5, Prentice Hall, 1975.
[3] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, "Numerical Recipes in C", ISBN 0-521-43108-5, Cambridge University Press, 1992.
[4] B. Willke *et al.*, Class. Quant. Grav. **19** (2002) 1377.