2019 American Control Conference (ACC)
Philadelphia, PA, USA, July 10-12, 2019

A Comparison of the δ Parameterization and the τ Parameterization

Daniel Y. Abramovitch*

Abstract— **This paper continues the analysis of biquad structures – discretized at high sample rates relative to the dynamic frequencies of the biquad – by introducing the τ parameterization as an alternative to the venerable δ parameterization [1], [2]. The τ parameterization has slightly improved coefficient accuracy as the scale factor between sample frequency and biquad dynamic frequencies falls below the typical several orders of magnitude assumed in the use of the δ parameterization. The use of the τ parameterization, which is based on a Trapezoidal Rule equivalent, provides an intuitively appealing result. In the mid-range sample rates, there is a potential for significant decrease in controller latency. Furthermore, the use of a biquad based structure such as the MultiNotch [3] or Biquad State Space [4], [5] allows us to seriously consider doing away with "one size fits all" discretization, applying τ or δ parameterized biquads for low frequency dynamics and traditional discrete biquads for high frequency dynamics. Finally, a restructuring of the fixed point calculations of the τ^{-1} integrator allows for improved signal fidelity while maintaining the signal growth immunity of the δ parameterization [6].**

I. Introduction

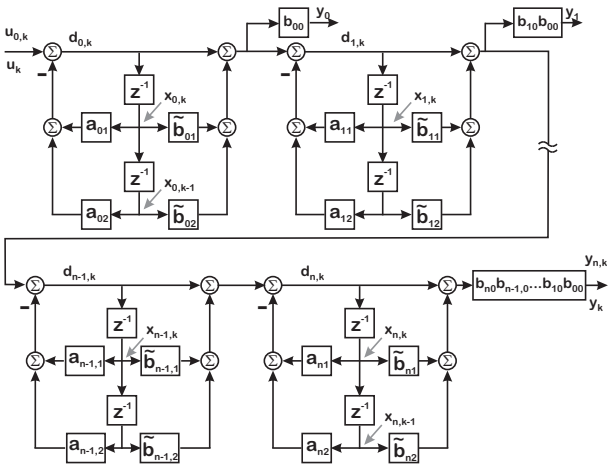

Fig. 1. Discrete biquad cascade, with factored out $b_{i,0}$ terms and scaling the output of each block.

The MultiNotch was introduced in [3] as a way to parameterize digital filters so as to preserve numerical fidelity of the filter while providing precalculation to reduce computational latency. Still, as the sample rate got large relative to the frequencies being filtered, the biquad coefficients in the MultiNotch got sensitive. A coefficient adjustment called Δ coefficients introduced in [7] worked extremely well on coefficient sensitivity, but did not address potential signal overflow problems. The δ parameterization [1], [2] is another method of adjusting digital filter coefficients to compensate for relatively high sample rates. A comparison of the two for coefficient accuracy was made in [8] and for signal growth in [6]. The results show that in a biquad cascade, such as the MultiNotch or the Discrete Time Biquad State Space [4], the main advantage of the δ parameterization is in limiting the internal signal growth when the sample rate is significantly higher than the frequencies of the dynamics in the biquad. In this paper, we explore using a slight improvement on the δ parameterization, the τ parameterization – tied closely to Tustin's rule, for the low frequency biquads. We will see that it keeps the scaling that limits signal growth at low frequency while retaining a bit more coefficient accuracy and with a potential for significant latency reduction in the middle frequencies.

Consider a controller design in the form of a continuous time filter,

$$C(s) = \frac{b_{0,c}s^n + b_{1,c}s^{n-1} + \cdots + b_{n-1,c}s + b_{n,c}}{s^n + a_{1,c}s^{n-1} + \cdots + a_{n-1,c}s + a_{n,c}} \qquad (1)$$

which has to be discretized for implementation on a real-time computer. A function of z^{-1}

$$C(z^{-1}) = \frac{b_0 + b_1 z^{-1} + \cdots + b_{n-1}z^{-n+1} + b_n z^{-n}}{1 + a_1 z^{-1} + \cdots + a_{n-1}z^{-n+1} + a_n z^{-n}}, \qquad (2)$$

allows us to express the output directly as a combination of past outputs and inputs:

$$\begin{aligned} u(k) \;=\; & -a_1 u(k-1) + \cdots - a_{n-1}u(k-n+1) \\ & -a_n u(k-n) + b_0 e(k) + b_1 e(k-1) + \\ & \cdots + b_{n-1}e(k-n+1) + b_n e(k-n). \end{aligned} \qquad (3)$$

This is all well known. It is also the case that for "high-Q" dynamics, such as those found in mechatronic systems, that is those characterized by one or more resonances or anti-resonances (notches) with very low damping ratios (and therefore high filter quality factors or Q's), such digital representations often fall short, particularly with multiple features (resonances/anti-resonances) spread across a wide frequency range and a sample frequency that is several orders of magnitude higher than some of the features.

The MultiNotch (Figure 1) puts (2) – (3) into a cascade of biquads (Figure 2) [3], [7] to exploit the improved numerical properties of having discrete coefficients of second order sections where those sections are selected so that the pole-zero pairs are as close as possible to each other.

Starting with a continuous time biquad,

$$B_{i,C} = b_{i0,C}\left(\frac{s^2 + \tilde{b}_{i1,C}s + \tilde{b}_{i2,C}}{s^2 + a_{i1,C}s + a_{i2,C}}\right), \qquad (4)$$

*Daniel Y. Abramovitch is a system architect in the Mass Spec Division at Agilent Technologies, 5301 Stevens Creek Blvd., M/S: 3U-WT, Santa Clara, CA 95051 USA, danny@agilent.com

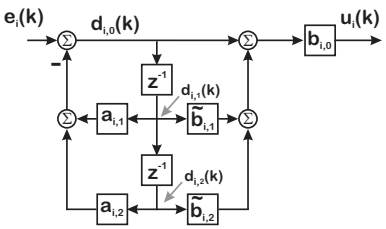

Fig. 2. A digital biquad filter with the b_0 term factored out.

we discretize through some chosen method (Figure 2) to yield

$$B_{i,D}(z^{-1}) = b_{i0}\left(\frac{1+\tilde{b}_{i1}z^{-1}+\tilde{b}_{i2}z^{-2}}{1+a_{i1}z^{-1}+a_{i2}z^{-2}}\right). \quad (5)$$

We will assume that we have factored out $b_{i,0}$ from the numerator in both forms, and will restrict the discussion to resonance/anti-resonance pairs, since they are most illustrative of the issues we are trying to examine. We will not dwell on the discretization of the biquad itself since that has been discussed in [3], [4], [5], [6], [7], and [8].

The work of [2], [9], [10] made obvious the issue that as the sample frequency goes up relative to the feature being controlled, the poles/zeros of the compensator approach the point $z = 1$. This means that the coefficients of (2) – (3) do not change much even when the physical parameters that they are supposed to represent change significantly. A difference of several hundred Hertz in resonance frequency of the physical system may be represented by a few bits worth of variation of the filter parameters.

This has been addressed via both the Δ coefficients [7] and the δ parameterization, the main advantage of the latter being less sensitive to signal growth. The δ parameterization, which modifies the digital filter by mapping:

$$\delta = \frac{z-1}{\Delta} \text{ or } z = 1 + T_S\delta \quad (6)$$

where $T_S = 1/f_S$ is the sample period. As this is the form of the familiar Forward Rectangular Rule integration discrete equivalent [11], we know that one of the effects is to map the inside of the unit circle back towards a line so that at very small T_S, $\delta \longrightarrow s$.

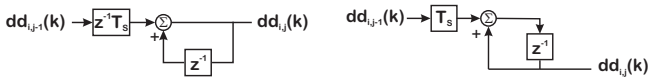

Fig. 3. Block diagram for implementing δ^{-1} block in a filter. On the left is a traditional view of the forward rule integrator. On the right is a form allowing us to compare it to the τ^{-1} integrator more readily.

The δ parameterization starts with coefficients that are already from the discrete form. As noise considerations generally make it more prudent to integrate rather than differentiate when possible, the δ parameterization is implemented using δ^{-1} form shown in Fig. 3:

$$\delta^{-1} = \frac{T_S}{z-1} = \frac{T_S z^{-1}}{1-z^{-1}}. \quad (7)$$

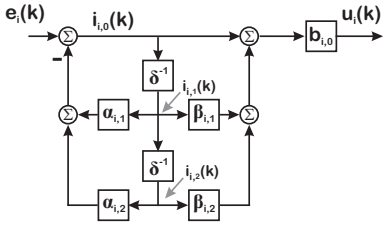

Fig. 4. A digital biquad reparameterized with the δ parameterization.

A δ parameterized biquad is shown in Figure 4 where the biquad's δ parameters are obtained from the discrete biquad parameters [8] by evaluating (5) as a function of δ as defined in (6). Proceeding directly to the δ^{-1} form, we get

$$B_{i,\delta}(\delta^{-1}) = b_{i0}\left(\frac{1+\tilde{\beta}_{i1}\delta^{-1}+\tilde{\beta}_{i2}\delta^{-2}}{1+\alpha_{i1}\delta^{-1}+\alpha_{i2}\delta^{-2}}\right), \quad (8)$$

where

$$\alpha_{i,1} = \frac{a_{i,1}+2}{T_S}, \quad \alpha_{i,2} = \frac{1+a_{i,1}+a_{i,2}}{T_S^2}, \quad (9)$$

$$\tilde{\beta}_{i,1} = \frac{\tilde{b}_{i,1}+2}{T_S}, \text{ and } \tilde{\beta}_{i,2} = \frac{1+\tilde{b}_{i,1}+\tilde{b}_{i,2}}{T_S^2}. \quad (10)$$

Three things to note here:

- This block – a Forward-Rectangular Rule discrete integrator [11] – *only* produces reasonable results when T_S is small compared to the dynamics being integrated.
- The δ^{-1} blocks maintain their own state, and as digital integrators, one has to be aware of the number of bits needed to prevent overflow.
- As was pointed out in [8], the δ parameters approach the continuous parameters and as shown in Table I (repeated from [8]), these require more bits to represent even modest frequencies. A 1 kHz resonance or anti-resonance will require at least 26 bits while a 2 kHz resonance or anti-resonance will require at least 28 bits. Thus, the larger word size requirement has been moved from the signal to the coefficients.

Format	Maximum Number for Format	$f_{0,max}$ (Hz)
s18.0 (18-bit signed integer)	$2^{17}-1$	57.6200
s25.0 (25-bit signed integer)	$2^{24}-1$	651.8986
s27.0 (25-bit signed integer)	$2^{26}-1$	1.3038e+003
s32.0 (32-bit signed integer)	$2^{31}-1$	7.3754e+003

TABLE I
FIXED POINT NUMBER FORMATS AND THE MAXIMUM FREQUENCIES THEY CAN HOLD.

The rest of this paper will proceed as follows. Section II will present the reasoning behind using different discretization methods on different biquads. Section III will introduce the τ Parameterization. Section IV will discuss how we compare three low frequency forms, while Section V will present the results. Finally, Section VI proposes a method of increasing the accuracy of fixed point τ^{-1} integrators while mitigating issues of signal overflow.

Block diagrams in this paper use the same "mixed-metaphor" combinations of time and frequency notation used in [12], [13], and [14]. z^{-1} blocks have signals with time shift notation going in and out of them, e.g. $x_k, x(k)$. In difference equations the z^{-1} becomes the unit delay operator, similar to q^{-1}, but we still recognize the that we can also get a frequency response from the structure with z^{-1}. Although, inexact, this usage is common and well understood.

II. Biquad by Biquad Discretization

The final result of the comparisons between Δ coefficients and δ parameters as applied to a single biquad on coefficient accuracy [8] and signal growth [6] is that Δ coefficients keep their accuracy across all frequency bands while the δ parameterization is most accurate when the sample frequency is orders of magnitude higher than the frequencies of the biquad dynamics. On the other hand, for signal growth, the Δ coefficients do nothing, while the δ parameterization is largely immune to the signal growth due to large differences between the sample rate and the frequencies of the biquad dynamics.

The logic and results of [6] indicates that we might want to use different parameterizations for biquads operating in different frequency ranges. Of course, this is not practical if we choose one discretization model for an entire filter (which happens in typical polynomial form filters or state space forms). In the case of the MultiNotch [3] and the Biquad State Space [4], [5], the discretization happens one biquad at a time, and so it is straightforward – albeit not initially intuitive – to select different discretization methods for different biquads. Doing away with "one size fits all" discretization presents a tremendous opportunity to match the controller subsection implementation to the physical system subsection.

III. The τ Parameterization

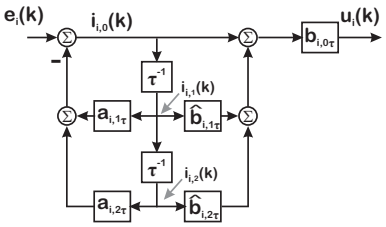

Fig. 5. A digital biquad reparameterized with the τ parameterization.

When using a δ parameterization at low frequency due to how the δ^{-1} integrator handles signal growth, we might naturally ask, "Why not use a better integrator?" One such integrator would be a τ^{-1} integrator where:

$$\tau^{-1} = \frac{T_S}{2}\left(\frac{z+1}{z-1}\right) = \frac{T_S}{2}\left(\frac{1+z^{-1}}{1-z^{-1}}\right) \tag{11}$$

This is clearly the Trapezoidal, bilinear, or Tustin rule approximation [12] to s, i.e.

$$s \approx \frac{2}{T_S}\left(\frac{z-1}{z+1}\right) = \tau \text{ or } z = \frac{1+\frac{T_S}{2}\tau}{1-\frac{T_S}{2}\tau}. \tag{12}$$

The τ parameterization sets $\tau = \frac{2}{T_S}\frac{z-1}{z+1}$ to yield:

$$B_{i,\tau}(\tau^{-1}) = b_{i,0,\tau}\left(\frac{1+\tilde{b}_{i,1,\tau}\tau^{-1}+\tilde{b}_{i,2,\tau}\tau^{-2}}{1+a_{i,1,\tau}\tau^{-1}+a_{i,2,\tau}\tau^{-2}}\right), \tag{13}$$

where the τ parameters are obtain from the discrete biquad parameters of Equation 5 via

$$a_{i,1\tau} = \frac{4}{T_S}\left(\frac{1-a_{i,2}}{\Delta_{A,i}}\right),\ a_{i,2\tau} = \frac{4}{T_S^2}\left(\frac{1+a_{i,1}+a_{i,2}}{\Delta_{A,i}}\right), \tag{14}$$

$$\tilde{b}_{i,1\tau} = \frac{4}{T_S}\left(\frac{1-\tilde{b}_{i,2}}{\Delta_{B,i}}\right),\ \tilde{b}_{i,2\tau} = \frac{4}{T_S^2}\left(\frac{1+\tilde{b}_{i,1}+\tilde{b}_{i,2}}{\Delta_{B,i}}\right). \tag{15}$$

$$\text{and } \tilde{b}_{i,0\tau} = b_{i,0}\left(\frac{\Delta_{B,i}}{\Delta_{A,i}}\right). \tag{16}$$

Here,

$$\Delta_{A,i} = 1 - a_{i,1} + a_{i,2} \text{ and } \Delta_{B,i} = 1 - \tilde{b}_{i,1} + \tilde{b}_{i,2}. \tag{17}$$

Part of the allure of the δ parameterization is that as $T_S \longrightarrow 0$, $\alpha_i \longrightarrow a_{i,C}$ and $\tilde{\beta}_i \longrightarrow \tilde{b}_{i,C}$ [2]. As the Trapezoidal Rule is more accurate than the Forward Rectangular Rule, we expect the mapping from traditional discrete parameters to τ parameters to be very close to the original continuous time parameters. If the original discretization done via a Trapezoidal Rule equivalent, we can make the trip from s to τ using both parts of (12):

$$s \approx \frac{2}{T_S}\left(\frac{1+\frac{T_S}{2}\tau-1+\frac{T_S}{2}\tau}{1+\frac{T_S}{2}\tau+1-\frac{T_S}{2}\tau}\right) = \tau. \tag{18}$$

Even if we don't use the Trapezoidal Rule to discretize our biquad, the coefficients should be pretty close to the continuous time ones.

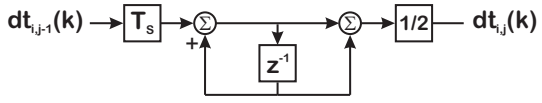

Fig. 6. Implementing the τ^{-1} integrator as a filter. In this layout, we can compare it to the δ^{-1} integrator block of the right side of Figure 3.

The τ^{-1} integrator block of Figure 6 can be compared to the δ^{-1} integrator block on the right side of Figure 3. This structure makes it obvious that there should be no difference in internal scaling between the two integrators. For this reason, we need not compare the two with respect to internal signal growth. They should be identical.

The difference is all at the output of the blocks. The τ^{-1} integrator block averages the current and delayed integrator state while the δ^{-1} integrator only gives the delayed state. Thus, there is an average of half a sample delay more with a δ^{-1} integrator than a τ^{-1} integrator. For the very high sample rates ($\sim$100$\times$ the biquad dynamic frequencies) associated with the prescribed use of the δ parameterization, this may not be an issue. However, for biquad dynamic frequencies (BDFs) only 10–20 times smaller than the sample frequency, a 10$^{\text{th}}$ order controller with δ^{-1} integrators would have an average of $5T_S$ more delay than the same controller with τ^{-1}

integrators. **This may be the most significant difference between the two parameterizations: the τ parameterization allows the use of the τ^{-1} integrator which reduce controller delay.** We can also look at the accuracy for the two coefficient parameterizations in fixed point, but as they both approach the continuous time parameters, the clipping issues described in [8] should affect them in similar ways.

IV. Comparing δ and τ Forms Methodology & Results

Generally, these discussions fall into three frequency ranges: signals far below the biquad dynamic frequencies (BDFs), signals far above the BDFs, and signals near the BDFs. As discussed in [6], the signals near the dynamic frequencies of the biquad, particularly that of the denominator are those that are most important for considering the upper bound on signal growth. Relative sample frequencies also have three ranges with respect to a particular biquad: those which are approximately 10 times that of the BDFs, those that are on the order of 100 times that of the BDFs, and those that were much higher, say 1000 times those of the BDFs. We saw in [6] that the scaling of the δ^{-1} integrator by T_S which kept the δ parameterized biquads insensitive to the increases in f_S. Comparing Figures 3 and 6 makes it obvious that this will also hold for the τ parameterized biquads. They are different in their output signals in that the δ^{-1} integrator outputs the delayed value of the accumulator, while the τ^{-1} integrator outputs the average of the current and delayed value of the integrator. Thus, we expect no significant differences in the outputs of the differently parameterized biquads, as we observed with all the parameterizations in [6].

When the sample frequency, f_S, is several orders of magnitude above the BDFs we would expect the signal and coefficient accuracy of the δ and τ parameterized biquads to be high. For a fixed f_S, we would expect that accuracy of the δ to fall off as the BDFs rose to the mid to high range (say within $1/20$ – $1/10$ of f_S). As the Trapezoidal Rule integration is more accurate than a Forward Rectangular Rule integrator (especially for short integrals), the τ^{-1} integrator should produce better results over a higher frequency range.

We will look at the fidelity of the coefficients to the numbers that they are supposed to represent. In particular using floating point math, or sufficiently high sample frequencies, we might see little difference. We would expect that the chief differences would be in the middle frequency ranges, where τ parameterization might show greater coefficient fidelity for fixed point math than the δ parameterization. Following the discussion of Section II, moving to a biquad-by-biquad discretization would allow us to use the the τ parameters on low to mid frequency biquads, while using standard discretization on the higher frequency ones. In doing so, the greater fidelity of the τ parameterization would allow us to use it over a greater frequency range than that of the δ parameterization.

In comparing the δ^{-1} and τ^{-1} forms, we still have the issues with biquad coefficients that approximate the continuous time ones described in [8]: the size of the coefficients when implemented in fixed point math. We are once again facing a tradeoff between limiting signal growth and having accurate computations in fixed point math, and we care about fixed point operations because they take approximately half the time and one quarter the programmable logic resources of floating point computations. Thus, when the physical system pushes computation rates, this becomes an important distinction.

The results of this study were that the coefficient accuracy differences were small, owing particularly to the need to truncate the lower bits of the $\alpha_2, \beta_2, a_{\tau,2},$ and $b_{\tau,2}$ coefficients for high biquad frequencies. We choose a single example to illustrate this with parameters in Table II. The resulting τ parameters are shown in Table III and these can be directly compared with the δ parameters of Example 2 (Table IV) in [8].

Example Parameters			
$f_{N,n}$ **(Hz)**	Q_n	$f_{N,d}$ **(Hz)**	Q_d
1000	40	2000	40

TABLE II

Analog biquad parameters for a single biquad example.

Continuous Time			
a_{1c}	a_{2c}	b_{1c}	b_{2c}
3.141593e+02	1.579137e+08	1.570796e+02	3.947842e+07
9 bits	28 bits	8 bits	26 bits
τ **param,** $f_S = 1e4$			
$a_{1,\tau}$	$a_{2,\tau}$	$\tilde{b}_{1,\tau}$	$\tilde{b}_{2,\tau}$
4.799329e+02	2.111198e+08	1.736590e+02	4.222817e+07
9 bits	28 bits	8 bits	26 bits
τ **param,** $f_S = 1e5$			
$a_{1,\tau}$	$a_{2,\tau}$	$\tilde{b}_{1,\tau}$	$\tilde{b}_{2,\tau}$
3.154025e+02	1.583301e+08	1.572347e+02	3.950440e+07
9 bits	28 bits	8 bits	26 bits
τ **param,** $f_S = 1e6$			
$a_{1,\tau}$	$a_{2,\tau}$	$\tilde{b}_{1,\tau}$	$\tilde{b}_{2,\tau}$
3.141717e+02	1.579178e+08	1.570812e+02	3.947868e+07
9 bits	28 bits	8 bits	26 bits

TABLE III

Coefficients computed for the biquad example, with analog and τ parameter coefficients. This can be directly compared with the δ parameterization of Example 2 in [8].

A quick look at Table III indicates that the biquad coefficients corresponding to the τ parameterization do get close to the continuous-time coefficients and can be large numbers. In the analog biquad form, the largest coefficients will likely be related to the $\omega_{N,n}^2$ or $\omega_{N,d}^2$ terms. As we can see from Table I, frequencies in the neighborhood of 10 kHz require 32 bits or more to hold the coefficients. This matters because FPGAs have, for their fastest operations, fixed size multipliers, typically 18×18 or 18×25 bits in the case of Xilinx [15] or 18×18 or 27×27 bits in the case of Altera [16]. The fastest, lowest latency computation that we can do involves a single hardware multiplier, and so we want to compare

these. Furthermore, most of these multipliers use signed, two's complement arithmetic, so the unsigned formats are not available in hardware. For mechatronic systems, with some natural frequencies in the tens of kHz, it is not reasonable to use τ coefficients in their raw form. This is a major issue.

V. Comparing Numeric Accuracy: Results

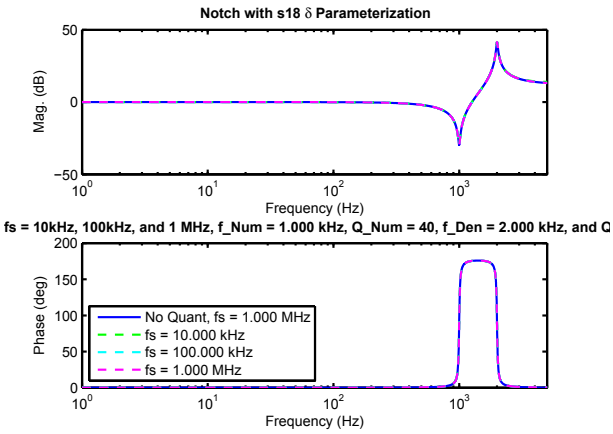

Fig. 7. Bode plot of example, with coefficients from a shifted δ parameterization.

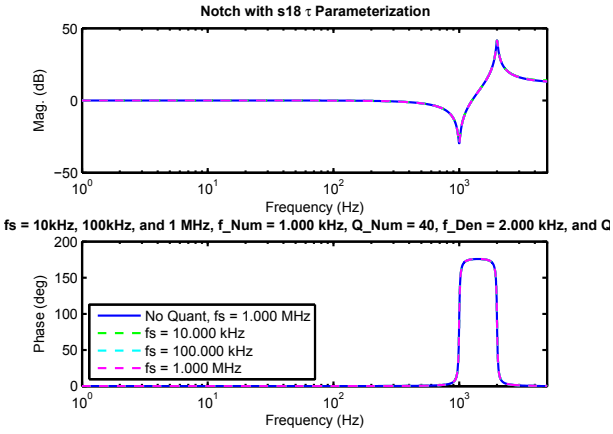

Fig. 8. Bode plot of example, with coefficients from a shifted τ parameterization.

The comparisons here will mirror many of the comparisons between conventional digital biquad coefficients,the Δ coefficients, and the δ parameter coefficients in [7] and [8]. For simplicity and space considerations, we will restrict our comparisons to signed 18-bit coefficients. The rationale is we would like to compare the behavior of 18-bit δ parameters and τ parameters.

The issue is that the continuous-time coefficients, and therefore the biquad coefficients associated with the δ and τ parameterizations, often require far more than 18 bits to represent. It would be easy enough to state that this limitation disqualifies these parameterizations, but since we want an "apples-to-apples" comparison of the accuracy of the three formats, we borrow a method used in [2], [7], [8].

We will use the method of [8] for computing the effects of finite word length on the δ parameterization and adapt

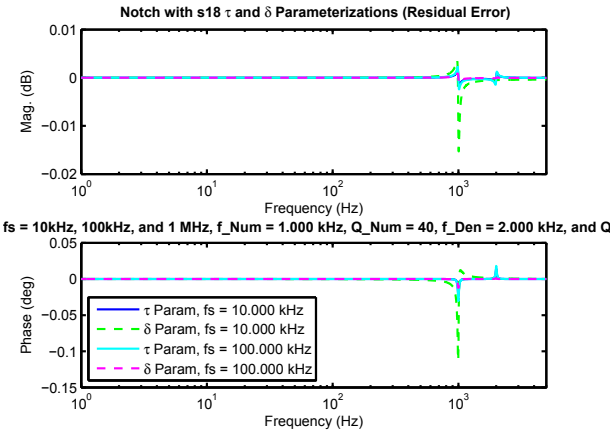

Fig. 9. Bode plot of example, showing differences between floating point, $s18$ δ, and $s18$ τ parameters.

it for the τ parameterization as follows. We compute the τ parameterization biquad coefficients from Equations 13–16, and then find a common number of bits to shift, N, such that we replace $a_{i,\tau 1}$, $a_{i,\tau 2}$, $\tilde{b}_{i,\tau 1}$, and $\tilde{b}_{i,\tau 2}$ with s18.0 format versions such that:

$$a_{i,\tau 1} \approx 2^N \times a_{i,\tau 1,s18}, \quad (19)$$
$$a_{i,\tau 2} \approx 2^N \times a_{i,\tau 2,s18}, \quad (20)$$
$$\tilde{b}_{i,\tau 1} \approx 2^N \times \tilde{b}_{i,\tau 1,s18}, \text{ and} \quad (21)$$
$$\tilde{b}_{i,\tau 2} \approx 2^N \times \tilde{b}_{i,\tau 2,s18}, \quad (22)$$

and then convert these back to conventional biquad coefficients by reversing Equations 13–16.

$$a_{i,1,s18} = \frac{2^N a_{i,\tau 2,s18} \frac{T_S^2}{2} - 2}{\Delta_{A\tau}}, \quad (23)$$
$$a_{i,2,s18} = \frac{1 - 2^N a_{i,\tau 1,s18} \frac{T_S}{2} + 2^N a_{i,\tau 2,s18} \left(\frac{T_S}{2}\right)^2}{\Delta_{A\tau}} \quad (24)$$
$$\tilde{b}_{i,1,s18} = \frac{(2^N) b_{i,\tau 2,s18} \frac{T_S^2}{2} - 2}{\Delta_{B\tau}}, \quad (25)$$
$$\tilde{b}_{i,2,s18} = \frac{1 - 2^N b_{i,\tau 1,s18} \frac{T_S}{2} + 2^N b_{i,\tau 2,s18} \left(\frac{T_S}{2}\right)^2}{\Delta_{B\tau}} \quad (26)$$

where $\Delta_{A\tau} = 1 + 2^N a_{i,\tau 1,s18} \frac{T_S}{2} + 2^N a_{i,\tau 2,s18} \left(\frac{T_S}{2}\right)^2$ and $\Delta_{B\tau} = 1 + 2^N b_{i,\tau 1,s18} \frac{T_S}{2} + 2^N b_{i,\tau 2,s18} \left(\frac{T_S}{2}\right)^2$.

The coefficients from Equations 23 – 26 can now be used to form new conventional biquads for generating discrete Bode plots, and these can be compared in a fair way. This has been done for a single example (the most illustrative) and the results are shown in Figures 7 – 9.

This particular example was chosen because the biquad dynamic frequencies (BDFs) were within $10\times$ that of the sample frequency (for the lower sample rate). We can make some inferences by looking at the notches with δ and τ parameter coefficients. Correcting for an error in [8], only the coefficients which were too large for the $s18$ format were actually shifted, truncated, and shifted back, and these were only the α_2, β_2, $a_{\tau,2}$ and $b_{\tau,2}$ coefficients. The plots show

remarkably little difference between the δ and τ parameterizations, especially as the sample rate goes up relative to the biquad dynamic frequencies. For low sample rates (only $10\times$ the BDFs), we do see a slight improvement in accuracy for the τ coefficients, but this is on a pretty small scale.

VI. Rearranging τ^{-1} Integrators for Higher Accuracy

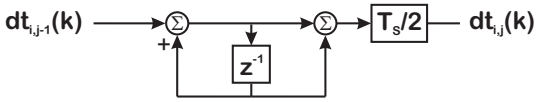

Fig. 10. Implementing the τ^{-1} integrator as a filter. Comparing this to Figure 6, we have moved the value of $\frac{T_S}{2}$ to after the integrator so that the integrator gets full accuracy.

T_S	$\log_2(T_S)$	**Bits Lost**
1e-3	-9.96	10
1e-4	-13.29	14
1e-5	-16.61	17
1e-6	-19.93	20

TABLE IV
Bits lost by multiplying by T_S.

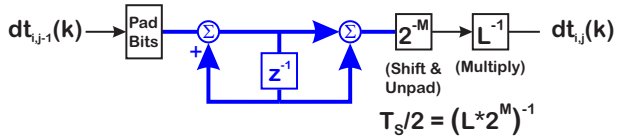

Fig. 11. This implementation of the τ^{-1} integrator allows the internal signals of the integrator to have higher bit count, maintaining a more accurate state.

In all cases, the scaling of T_S immunizes the system against massive signal growth as f_S goes up. However, premultiplying our signal with a tiny T_S as it enters the integrator could nearly zero out the input of even a 25-bit fixed point number, leading to an inaccurate integral. The number of bits lost for representative values of T_S are shown in Table IV. Recalling the growth analysis of [6], we would look at a signal that was constant over a half interval of the biquad denominator to give an estimate of the needed number of bits to hold the signal growth.

One measure of potential internal signal growth is how much a signal at the denominator natural frequency ($f_{D,i}$) grows. If we inject a signal at $f_{D,i}$, it will essentially behave like an integrator over the half period of the denominator frequency before the oscillation turns around. Thus, one bound on signal growth inside the biquad is to ask how many steps there are in the half period of that input signal. For a signal of frequency f_0 with a period T_0 there are

$$\text{Steps in}\frac{T_0}{2} = N_{step} = \frac{\frac{T_0}{T_S}}{2} = \frac{\frac{f_S}{f_0}}{2}. \tag{27}$$

By this simple relation, the relative increase in the sample rate versus the denominator frequency results in significant increase in the number of values that might be added up. For relatively high sample rates, the denominator of a step biquad approaches a double integrator [17] and so we can approximate:

$$\begin{aligned} d_{i,k} &= -a_{i,1}d_{i,k-1} - a_{i,2}d_{i,k-2} + u_{i,k} \\ &\approx 2d_{i,k-1} + d_{i,k-2} + u_{i,k}. \end{aligned} \tag{28}$$

If we start the half interval at $k = 0$ and assume a constant $u_{i,k} = 1$ on the half interval (say for a square wave), then a few steps reveal that:

$$d_{i,k} \approx ku_{i,0} + (k-1)u_{i,1} + \ldots + 1u_{i,k}, \tag{29}$$

$$= \frac{k(k+1)}{2} = \frac{N_{step}(N_{step}+1)}{2}. \tag{30}$$

This is an upper bound on the growth over a half interval. Thus, for a filter with a resonance at 100 Hz and a 10 kHz sample rate, $N_{step} = 50$, which means that on the half interval, an upper bound of the growth would be $G = 25(51) = 1275$ which requires an extra $log_2(1275) = 10.3163$ or 11 bits of head space. For a 100 kHz sample rate, $G = 250(501) = 125,250 \approx 12.5e5$, which requires $log_2(125250) = 16.9345$ or 17 bits. Finally a 1 MHz sample rate makes $N_{step} = 5000$ and $G = 2500(5001) = 12,502,500 \approx 12.5e6$, which requires an extra 24 bits of head room. In [6] we argued that for standard multiply-accumulate blocks of field programmable gate arrays (FPGAs), such as a Xilinx DSP48E [18], [19] can compute 25 bit $\times$ 18 bit multiplies in 5 fabric clock cycles. These math blocks have 48 bit output registers, so there are only 5 bits of headroom. In two's complement math, an overflow rolls over erroneously and so we go from the maximally high value to the maximally low value (or vice-versa) which is unacceptable for control or signal processing applications. This means that the operation outputs must be pre-saturated to avoid this error, but a saturated value in a linear filter means that the filter is no longer linear and at best, is no longer filtering as it should. Following this stream of logic, one would have to provide either extra wide operators (using a combination of 2 or more DSP48Es) or move to floating point operations (using a combination of 2-4 DSP48Es, depending upon the signal width)[20]. However, the move to floating point essentially doubles the computational delay *and* means that we need to convert the signals in our block to floating point. We might do this for the entire filter, but for very high speed applications such as high speed control of an atomic force microscope (AFM) [21], this could be the limit. This might motivate a change in FPGA hardware, since Altera's (Intel) most recent FPGAs include hardware that can do single precision IEEE floating point in a single DSP block [22]. On the other hand, high speed applications with only high speed filters in them, such as AFM demodulation [23], [24] would not run into these headroom problems.

However, if we modify the τ^{-1} integrator structure to have a trailing multiplication by T_S we get Figure 10. What is important here is that the rest of the integral is done simply with a register and addition. In an FPGA, these functions can be done without the aid of specialty math blocks, and so for our purposes, we can pad the input signal with enough bits to account for our maximum growth and

do the integration/accumulation on those extra wide registers, as shown in Figure 11. This means that even the 24 bits of head room described above provide little hindrance to our integrators. A wide register can not only hold the final sum, but there would have been no loss of signal accuracy to that point. We still need to post-multiply by $T_S/2$ and for this we can employ a trick described in the FPGA based stepped-sine demodulator of [25].

As T_S is known in advance, on a host computer we compute $T_S/2 = L^{-1}(2^{-(M+1)})$ or $T_S = L^{-1}2^{-M}$. The $1/2^M$ will be done by the right shift by M bits and the last part of the normalization will be done by a multiply by $1/L$. Now, $1 < L < 2$ so $0.5 < 1/L < 1$, and we can represent $1/L$ as s1.17 two's complement number where the number looks like 0.1XXXXXXXXXXXXXXX, where the "X" bits can be either 0 or 1. The smaller T_S, the larger M, but once the right shifts by M resulting in a division of $1/2^M$ has been done, most of the padding top bits of the register can be discarded before the multiply by L. Thus, we can achieve high accuracy with no loss of speed, while maintaining our immunity to overall signal growth in the biquad.

While this may save us for low frequency signals with time constants several orders of magnitude times T_S, it does nothing for high frequency signals that tend to be averaged out by the integrators. Even a high fidelity multiplication of these small integrals by a minuscule T_S results in few or no bits of accuracy when the lower bits are removed. This quantization effectively creates an unintended low pass filter – which is not something we would want in our design.

VII. Conclusions

This paper has compared the common δ parameterization of biquad filters with a new, τ parameterization. The results of Section IV, show that the τ parameterization produces a slight improvement on coefficient accuracy when sample rates are not super high compared to the block dynamics. However, more significant is an understanding that implementing biquad chains with τ^{-1} integrators in place of δ^{-1} integrators may yield a substantial decrease in controller latency. A way of improving the precision of the δ^{-1} or τ^{-1} integrators for high speed, fixed point calculations was also described in Section VI. However, both the δ^{-1} and τ^{-1} parameterization can produce an unintended low pass filter due to the quantization effects.

This might finally be the feature that requires floating point calculations. New FPGA hardware [22] might cut the latency cost of such a move. Furthermore, it's worth pointing out that while a move to floating point calculations effectively does away with any numerical advantage of the δ or τ parameterizations, or the Δ coefficients [7], there are still considerable advantages to a biquad formulation of filters or state space, both in understandability and in numerical sensitivity [3], [4], [5]. Even in floating point, δ or τ parameterized biquads may be helpful in managing the rapidly sampled system as if it were a continuous time system. Finally, these types of frequency specific discretization decisions are only possible when the controller, filter, or state-space model is broken down into well defined subcomponents, such as the MultiNotch or Biquad State Space.

References

[1] G. C. Goodwin, J. I. Yuz, J. C. Agüero, and M. Cea, "Sampling and sampled-data models," in *Proceedings of the 2010 American Control Conference*, (Baltimore, MD), AACC, IEEE, June 2010.

[2] J. Kauraniemi, T. I. Laakso, I. Hartimo, and S. J. Ovaska, "Delta operator realizations of direct-form IIR filters," *IEEE Transactions on Circuits and Systems–II: Analog and Digital Signal Processing*, vol. 45, pp. 41–52, January 1998.

[3] D. Y. Abramovitch, "The Multinotch, Part I: A low latency, high numerical fidelity filter for mechatronic control systems," in *Proceedings of the 2015 American Control Conference*, (Chicago, IL), pp. 2161–2166, AACC, IEEE, July 2015.

[4] D. Y. Abramovitch, "The discrete time biquad state space structure: Low latency with high numerical fidelity," in *Proceedings of the 2015 American Control Conference*, (Chicago, IL), pp. 2813–2818, AACC, IEEE, July 2015.

[5] D. Y. Abramovitch, "The continuous time biquad state space structure," in *Proceedings of the 2015 American Control Conference*, (Chicago, IL), pp. 4168–4173, AACC, IEEE, July 2015.

[6] D. Y. Abramovitch, "A comparison of Δ coefficients and the δ parameterization, Part II: Signal growth," in *Proceedings the 2018 American Control Conference*, (Milwaukee, WI), pp. 5231–5237, AACC, IEEE, June 2018.

[7] D. Y. Abramovitch, "The Multinotch, Part II: Extra precision via Δ coefficients," in *Proceedings of the 2015 American Control Conference*, (Chicago, IL), pp. 4137–4142, AACC, IEEE, July 2015.

[8] D. Y. Abramovitch, "A comparison of Δ coefficients and the δ parameterization, Part I: Coefficient accuracy," in *Proceedings of the 2017 American Control Conference*, (Seattle, WA), AACC, IEEE, May 2017.

[9] G. C. Goodwin, J. I. Yuz, J. C. Agüero, and M. Cea, "Sampling and sampled-data models," in *Proc. 2010 Amer. Control Conf.*, IEEE, 2010.

[10] J. Wu, S. Chen, G. Li, R. Istepanian, and J. Chu, "Shift and delta operator realisations for digital controllers with finite word length considerations," *IEE Proc. Ctrl. Theory Appl.*, vol. 147, no. 6, 2000.

[11] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*. Add. Wesl. Long., 3rd ed., 1998.

[12] G. F. Franklin, J. D. Powell, and M. L. Workman, *Digital Control of Dynamic Systems*. Menlo Park, California: Addison Wesley Longman, third ed., 1998.

[13] G. F. Franklin, J. D. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems*. Upper Saddle River, New Jersey: Prentice Hall, fifth ed., 2006.

[14] T. Kailath, *Linear Systems*. Englewood Cliffs, N.J. 07632: Prentice-Hall, 1980.

[15] Xilinx, *7 Series DSP48E1 Slice Users Guide*, ug479 (v1.6) ed., August 7 2013.

[16] Altera, "Arria v Cyclone v DSP block," 2016. [Online; accessed September 17, 2016].

[17] D. Y. Abramovitch, "The Multinotch, Part II: Extra precision via Δ coefficients," in *Proc. Amer. Ctrl. Conf.*, (Chicago), IEEE, 2015.

[18] *Virtex-5 FPGA XtremeDSP Dsgn. Consid. UG193 (v3.5)*, 2012.

[19] Xilinx, *7 Series DSP48E1 UG479 (v1.6)*, 2013.

[20] Xilinx, *LogiCORE IP Float.-Pt Opr. DS816 (v1.2)*, 2012.

[21] D. Y. Abramovitch, S. B. Andersson, L. Y. Pao, and G. Schitter, "A tutorial on the mechanisms, dynamics, and control of atomic force microscopes," in *Proc. Amer. Ctrl. Conf.*, (New York, NY), July 2007.

[22] Intel, *Intel Stratix 10 Variable Precision DSP Blocks User Guide*, ug-s10-dsp ed., September 24 2018.

[23] D. Y. Abramovitch, "Low latency demodulation for atomic force microscopes, Part I: Efficient real-time integration," in *Proc. Amer. Ctrl. Conf.*, (San Francisco), IEEE, 2011.

[24] D. Y. Abramovitch, "Low latency demodulation for atomic force microscopes, Part II: Efficient calculation of magnitude and phase," in *Proc. IFAC World Congr.*, (Milan), IFAC, 2011.

[25] D. Y. Abramovitch, "Built-in stepped-sine measurements for digital control systems," in *Proceedings of the 2015 Multi-Conference on Systems and Control*, (Sydney, Australia), pp. 145–150, IEEE, IEEE, September 2015.