How to implement a digital biquadratic notch filter (in software)

by [Brandon Bickerstaff]

Background

As presented in the [complimentary white paper], the continuous-time transfer function of a biquadratic—or "biquad"—notch filter is …

$$H_{bn}(s) = \frac{s^2 + 2\zeta_{num}\omega_c s + \omega_c^2}{s^2 + 2\zeta_{den}\omega_c s + \omega_c^2}$$

Where s ≔ Laplace variable, ζ ≔ damping ratio (—), ω_c ≔ center/notch frequency (rad/s), and the subscripts num and den designate "numerator" and "denominator," respectively. ω_c is a design parameter, and the aforementioned white paper derives an algorithm for calculating ζ_{num} and ζ_{den}. That's great, and $H_{bn}(s)$ can be practically utilized with the "help" of third-party software such as [MathWorks' Control System Toolbox]. But what if the goal is to implement a notch filter in our *own* software—software running on an embedded system, for example? If that's the case, then a *digital* filter is needed, and its coefficients and corresponding difference equation must be determined.

Derivation

Bilinear transform

The [z-transform] is the discrete-time counterpart of the [Laplace transform], where ...

$$z \coloneqq e^{sT} = \frac{e^{sT/2}}{e^{-sT/2}}$$

And T ≔ sampling period (s), a design parameter. The first-order, [Maclaurin-series] approximation of e^x is $1 + x$. Thus, …

$$z \approx \frac{1 + sT/2}{1 - sT/2} \rightarrow s \approx \frac{2}{T} \cdot \frac{z-1}{z+1}$$

The latter result is known as the bilinear transform: replacing s with $\frac{2}{T} \cdot \frac{z-1}{z+1}$ allows a transfer function to be converted from continuous time to discrete time. Unfortunately, that transformation causes a "warping" of the frequencies—because, after all, it's based on an approximation (notably, a nonlinear one). For example, upon bilinearly transforming $H_{bn}(s)$, a Bode plot of the consequent, *digital* filter would show that ω_c—which was specified for the original, *analog* filter—shifted. (Try it for yourself!) Obviously, that's problematic, as the frequency the filter was designed to "notch out" won't properly be attenuated. Thankfully, that predicament can be remedied via frequency prewarping.

Frequency prewarping

Let …

- $H_a(s)$ be an arbitrary (continuous-time) transfer function

- $H_d(z)$ be the bilinearly transformed (discrete-time) equivalent of $H_a(s)$
- ω_a be the defining (analog) frequency of $H_a(s)$—e.g., the center frequency of a notch filter
- ω_d be the warped (digital) frequency corresponding to ω_a

The objective is to determine the relationship between ω_a and ω_d that makes the frequency responses of $H_a(s)$ evaluated at $j\omega_a$ and $H_d(z)$ evaluated at $e^{j\omega_d T}$—recalling the previous definition of z—equal, i.e., …

$$H_a(s = j\omega_a) = H_d(z = e^{j\omega_d T})$$

Employing the bilinear transform yields …

$$H_a\left(\frac{2}{T}\cdot\frac{z-1}{z+1}\right) = H_d\left(e^{j\omega_d T}\right)$$

Where the left-hand side equals …

$$H_a\left(\frac{2}{T}\cdot\frac{e^{j\omega_d T}-1}{e^{j\omega_d T}+1}\right) = \cdots = H_a\left(j\frac{2}{T}\tan\left(\frac{\omega_d T}{2}\right)\right)$$

(Refer to [Wikipedia]() for the redacted “mathematical magic.”) Therefore, …

$$H_a(j\omega_a) = H_a\left(j\frac{2}{T}\tan\left(\frac{\omega_d T}{2}\right)\right) \therefore \omega_a = \frac{2}{T}\tan\left(\frac{\omega_d T}{2}\right) \rightarrow \omega_d = \frac{2}{T}\tan^{-1}\left(\frac{\omega_a T}{2}\right)$$

Clearly, $\omega_a \neq \omega_d$: the former does indeed get “warped” (into the latter) upon application of the bilinear transform. But, in practice, those two frequencies *should* be identical: the defining frequency that the design of the initial transfer function is established on (in the s domain) *should* be accurately represented in the z domain. So, what if the bilinear transform could be multiplied by a constant, K, that *pre*warps the analog frequencies such that, upon digitization of $H_a(s)$, 1) $\omega_a = \omega_d = \omega_0$, and 2) the frequency-response equality—i.e., $H_a(s = j\omega_0) = H_d(z = e^{j\omega_0 T})$—is maintained? If that were possible (hint: it is), then, similar to above, …

$$H_a\left(K\frac{2}{T}\cdot\frac{z-1}{z+1}\right) = H_d\left(e^{j\omega_0 T}\right)$$

Where the left-hand side equals …

$$H_a\left(K\frac{2}{T}\cdot\frac{e^{j\omega_0 T}-1}{e^{j\omega_0 T}+1}\right) = \cdots = H_a\left(jK\frac{2}{T}\tan\left(\frac{\omega_0 T}{2}\right)\right)$$

Therefore, …

$$H_a(j\omega_0) = H_a\left(jK\frac{2}{T}\tan\left(\frac{\omega_0 T}{2}\right)\right) \therefore \omega_0 = K\frac{2}{T}\tan\left(\frac{\omega_0 T}{2}\right) \rightarrow K = \frac{\omega_0 T}{2\tan\left(\frac{\omega_0 T}{2}\right)}$$

Consequently, the bilinear transform *with frequency prewarping* is given by …

$$s \approx K\frac{2}{T}\cdot\frac{z-1}{z+1} = \frac{\omega_0}{\tan\left(\frac{\omega_0 T}{2}\right)}\cdot\frac{z-1}{z+1}$$

That is, replacing s with $\frac{\omega_0}{\tan(\omega_0 T/2)} \cdot \frac{z-1}{z+1}$ allows a transfer function to be converted from continuous time to discrete time *while avoiding warping of the defining frequency, ω_0 (rad/s)*.

Bonus resource: Check out this video by Brian Douglas for an incredible explanation of this topic.

Filter coefficients

Now, apply the bilinear transform *with frequency prewarping* to $H_{bn}(s)$, noting that ω_c is the defining frequency of a notch filter:

$$H_{bn}(z) = H_{bn}(s)\big|_{s=\frac{\omega_c}{\tan(\omega_c T/2)}\cdot\frac{z-1}{z+1}} = \frac{\left(\frac{\omega_c}{\tan(\omega_c T/2)} \cdot \frac{z-1}{z+1}\right)^2 + 2\zeta_{num}\omega_c\left(\frac{\omega_c}{\tan(\omega_c T/2)} \cdot \frac{z-1}{z+1}\right) + \omega_c^2}{\left(\frac{\omega_c}{\tan(\omega_c T/2)} \cdot \frac{z-1}{z+1}\right)^2 + 2\zeta_{den}\omega_c\left(\frac{\omega_c}{\tan(\omega_c T/2)} \cdot \frac{z-1}{z+1}\right) + \omega_c^2}$$

$$= \frac{\frac{\omega_c^2}{\tan^2(\omega_c T/2)} \cdot \left(\frac{z-1}{z+1}\right)^2 + \frac{2\zeta_{num}\omega_c^2}{\tan(\omega_c T/2)} \cdot \frac{z-1}{z+1} + \omega_c^2}{\frac{\omega_c^2}{\tan^2(\omega_c T/2)} \cdot \left(\frac{z-1}{z+1}\right)^2 + \frac{2\zeta_{den}\omega_c^2}{\tan(\omega_c T/2)} \cdot \frac{z-1}{z+1} + \omega_c^2}$$

Canceling out ω_c^2 and multiplying the numerator and denominator by $\tan^2\left(\frac{\omega_c T}{2}\right)(z+1)^2$ yields …

$$H_{bn}(z) = \frac{(z-1)^2 + 2\zeta_{num}\tan\left(\frac{\omega_c T}{2}\right)(z-1)(z+1) + \tan^2\left(\frac{\omega_c T}{2}\right)(z+1)^2}{(z-1)^2 + 2\zeta_{den}\tan\left(\frac{\omega_c T}{2}\right)(z-1)(z+1) + \tan^2\left(\frac{\omega_c T}{2}\right)(z+1)^2}$$

$$= \frac{z^2 - 2z + 1 + 2\zeta_{num}\tan\left(\frac{\omega_c T}{2}\right)(z^2-1) + \tan^2\left(\frac{\omega_c T}{2}\right)(z^2+2z+1)}{z^2 - 2z + 1 + 2\zeta_{den}\tan\left(\frac{\omega_c T}{2}\right)(z^2-1) + \tan^2\left(\frac{\omega_c T}{2}\right)(z^2+2z+1)}$$

Collecting like terms yields …

$$H_{bn}(z) = \frac{\left[1 + 2\zeta_{num}\tan\left(\frac{\omega_c T}{2}\right) + \tan^2\left(\frac{\omega_c T}{2}\right)\right]z^2 + \left[-2 + 2\tan^2\left(\frac{\omega_c T}{2}\right)\right]z + \left[1 - 2\zeta_{num}\tan\left(\frac{\omega_c T}{2}\right) + \tan^2\left(\frac{\omega_c T}{2}\right)\right]}{\left[1 + 2\zeta_{den}\tan\left(\frac{\omega_c T}{2}\right) + \tan^2\left(\frac{\omega_c T}{2}\right)\right]z^2 + \left[-2 + 2\tan^2\left(\frac{\omega_c T}{2}\right)\right]z + \left[1 - 2\zeta_{den}\tan\left(\frac{\omega_c T}{2}\right) + \tan^2\left(\frac{\omega_c T}{2}\right)\right]}$$

For the sake of simplicity, because the numerator and denominator have the same form, let …

$$P_{bn}(z) = \left[1 + 2\zeta\tan\left(\frac{\omega_c T}{2}\right) + \tan^2\left(\frac{\omega_c T}{2}\right)\right]z^2 + \left[-2 + 2\tan^2\left(\frac{\omega_c T}{2}\right)\right]z + \left[1 - 2\zeta\tan\left(\frac{\omega_c T}{2}\right) + \tan^2\left(\frac{\omega_c T}{2}\right)\right]$$

Which is merely the numerator/denominator of $H_{bn}(z)$ with ζ_{num}/ζ_{den} replaced by ζ. Applying the half-angle and power-reduction formulas for tangent (recalling that $\tan\theta = \frac{\sin\theta}{\cos\theta}$) yields …

$$P_{bn}(z) = \left[1 + 2\zeta\frac{1-\cos(\omega_c T)}{\sin(\omega_c T)} + \frac{1-\cos(\omega_c T)}{1+\cos(\omega_c T)}\right]z^2 + \left[-2 + 2\frac{1-\cos(\omega_c T)}{1+\cos(\omega_c T)}\right]z + \left[1 - 2\zeta\frac{1-\cos(\omega_c T)}{\sin(\omega_c T)} + \frac{1-\cos(\omega_c T)}{1+\cos(\omega_c T)}\right]$$

Multiplying both sides by $\sin(\omega_c T)\,[1+\cos(\omega_c T)]$ yields …

$$
\begin{aligned}
&P_{bn}(z)\sin(\omega_c T)\,[1+\cos(\omega_c T)] \\
&\qquad = \{\sin(\omega_c T)\,[1+\cos(\omega_c T)] + 2\zeta[1-\cos(\omega_c T)][1+\cos(\omega_c T)] \\
&\qquad + [1-\cos(\omega_c T)]\sin(\omega_c T)\}z^2 \\
&\qquad + \{-2\sin(\omega_c T)\,[1+\cos(\omega_c T)] + 2[1-\cos(\omega_c T)]\sin(\omega_c T)\}z \\
&\qquad + \{\sin(\omega_c T)\,[1+\cos(\omega_c T)] - 2\zeta[1-\cos(\omega_c T)][1+\cos(\omega_c T)] \\
&\qquad + [1-\cos(\omega_c T)]\sin(\omega_c T)\} = \cdots \\
&\qquad = [2\sin(\omega_c T) + 2\zeta - 2\zeta\cos^2(\omega_c T)]z^2 + [-4\sin(\omega_c T)\cos(\omega_c T)]z \\
&\qquad + [2\sin(\omega_c T) - 2\zeta + 2\zeta\cos^2(\omega_c T)]
\end{aligned}
$$

Dividing both sides by 2 and applying the Pythagorean identity yields …

$$
\begin{aligned}
&P_{bn}(z)\frac{\sin(\omega_c T)\,[1+\cos(\omega_c T)]}{2} \\
&\qquad = [\sin(\omega_c T) + \zeta\sin^2(\omega_c T)]z^2 + [-2\sin(\omega_c T)\cos(\omega_c T)]z \\
&\qquad + [\sin(\omega_c T) - \zeta\sin^2(\omega_c T)]
\end{aligned}
$$

Dividing both sides by $\sin(\omega_c T)$ yields …

$$
P_{bn}(z)\frac{1+\cos(\omega_c T)}{2} = [1+\zeta\sin(\omega_c T)]z^2 + [-2\cos(\omega_c T)]z + [1-\zeta\sin(\omega_c T)]
$$

Such that …

$$
P_{bn}(z) = \frac{2\{[1+\zeta\sin(\omega_c T)]z^2 + [-2\cos(\omega_c T)]z + [1-\zeta\sin(\omega_c T)]\}}{1+\cos(\omega_c T)}
$$

Reconstructing $H_{bn}(z)$ as …

$$
H_{bn}(z) = \frac{P_{bn}(z)|_{\zeta=\zeta_{num}}}{P_{bn}(z)|_{\zeta=\zeta_{den}}}
$$

Yields …

$$
H_{bn}(z) = \frac{[1+\zeta_{num}\sin(\omega_c T)]z^2 + [-2\cos(\omega_c T)]z + [1-\zeta_{num}\sin(\omega_c T)]}{[1+\zeta_{den}\sin(\omega_c T)]z^2 + [-2\cos(\omega_c T)]z + [1-\zeta_{den}\sin(\omega_c T)]}
$$

Rewriting the above in standard form yields …

$$
\begin{aligned}
H_{bn}(z) &= \frac{\beta_0 + \beta_1 z^{-1} + \beta_2 z^{-2}}{\alpha_0 + \alpha_1 z^{-1} + \alpha_2 z^{-2}} \\
&= \frac{[1+\zeta_{num}\sin(\omega_c T)] + [-2\cos(\omega_c T)]z^{-1} + [1-\zeta_{num}\sin(\omega_c T)]z^{-2}}{[1+\zeta_{den}\sin(\omega_c T)] + [-2\cos(\omega_c T)]z^{-1} + [1-\zeta_{den}\sin(\omega_c T)]z^{-2}}
\end{aligned}
$$

Where …

- $\beta_0 = 1 + \zeta_{num}\sin(\omega_c T)$
- $\beta_1 = -2\cos(\omega_c T)$
- $\beta_2 = 1 - \zeta_{num}\sin(\omega_c T)$
- $\alpha_0 = 1 + \zeta_{den}\sin(\omega_c T)$

- $\alpha_1 = \beta_1$
- $\alpha_2 = 1 - \zeta_{den} \sin(\omega_c T)$

Finally, let …

$$H_{bn}(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Such that the filter coefficients (below) are normalized by α_0 for the sake of implementation:

- $b_0 = \beta_0 / \alpha_0$
- $b_1 = \beta_1 / \alpha_0$
- $b_2 = \beta_2 / \alpha_0$
- $a_1 = b_1$
- $a_2 = \alpha_2 / \alpha_0$

Difference equation

$$H_{bn}(z) = \frac{Y_{bn}(z)}{X_{bn}(z)}$$

Where $Y_{bn}(z)$ and $X_{bn}(z)$ ≔ output and input, respectively, of $H_{bn}(z)$. Thus, …

$$\frac{Y_{bn}(z)}{X_{bn}(z)} = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

$$\rightarrow Y_{bn}(z) + a_1 z^{-1} Y_{bn}(z) + a_2 z^{-2} Y_{bn}(z) = b_0 X_{bn}(z) + b_1 z^{-1} X_{bn}(z) + b_2 z^{-2} X_{bn}(z)$$

Applying the time-shifting property of the z-transform yields …

$$y_{bn}[n] + a_1 y_{bn}[n-1] + a_2 y_{bn}[n-2] = b_0 x_{bn}[n] + b_1 x_{bn}[n-1] + b_2 x_{bn}[n-2]$$

Where y_{bn} and x_{bn} ≔ *discrete-time* output and input, respectively, of the *digital* biquad notch filter, and n ≔ sample number such that …

- $y_{bn}[n]$ ≔ current output
- $y_{bn}[n-1]$ ≔ previous output
- $y_{bn}[n-2]$ ≔ second-to-last output
- $x_{bn}[n]$ ≔ current input
- $x_{bn}[n-1]$ ≔ previous input
- $x_{bn}[n-2]$ ≔ second-to-last input

Ultimately, the following function can be implemented in software to “notch” a discrete-time signal:

$$y_{bn}[n] = b_0 x_{bn}[n] + b_1 x_{bn}[n-1] + b_2 x_{bn}[n-2] - a_1 y_{bn}[n-1] - a_2 y_{bn}[n-2]$$

[Note that the above is the direct-form-I realization of $H_{bn}(z)$.]

Summary

The (software) implementation of a digital biquad notch filter is as follows:

$$y_{bn}[n] = b_0 x_{bn}[n] + b_1 x_{bn}[n-1] + b_2 x_{bn}[n-2] - a_1 y_{bn}[n-1] - a_2 y_{bn}[n-2]$$

Where …

- $b_0 = \beta_0 / \alpha_0$
- $b_1 = \beta_1 / \alpha_0$
- $b_2 = \beta_2 / \alpha_0$
- $a_1 = b_1$
- $a_2 = \alpha_2 / \alpha_0$

And …

- $\beta_0 = 1 + \zeta_{num} \sin(\omega_c T)$
- $\beta_1 = -2 \cos(\omega_c T)$
- $\beta_2 = 1 - \zeta_{num} \sin(\omega_c T)$
- $\alpha_0 = 1 + \zeta_{den} \sin(\omega_c T)$
- $\alpha_2 = 1 - \zeta_{den} \sin(\omega_c T)$