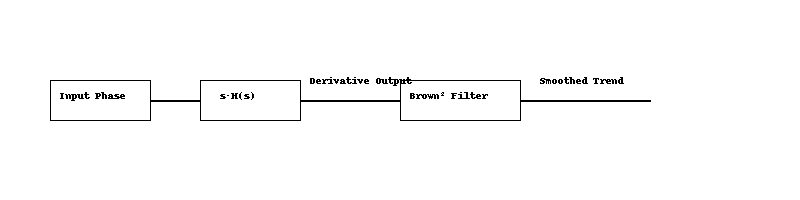
# PLL Derivative and Brown Filter Analysis

This document illustrates the sz x x Ztructure and frequency responses for a second-order PLL system's derivative output, with and without a second-order Brown low-pass filter applied.

## Block Diagram

The diagram below shows the signal flow and where each transfer function is measured:



• The input phase on the picture is the PLL input signal.  
• The pre-integrator output is modeled by s·H(s) where H(s) is defined as:

A black background with white text

AI-generated content may be incorrect.

The unit response for z=1, is:

A screen shot of a graph

AI-generated content may be incorrect.

And the frequency response and phase plot are:

A screenshot of a computer

AI-generated content may be incorrect.

• This output is optionally passed through a second-order Brown filter to produce a smoothed trend signal.

## Bode Plot

The plot below compares the frequency responses of the derivative output (red) and the filtered version (blue).

A screenshot of a computer

AI-generated content may be incorrect.

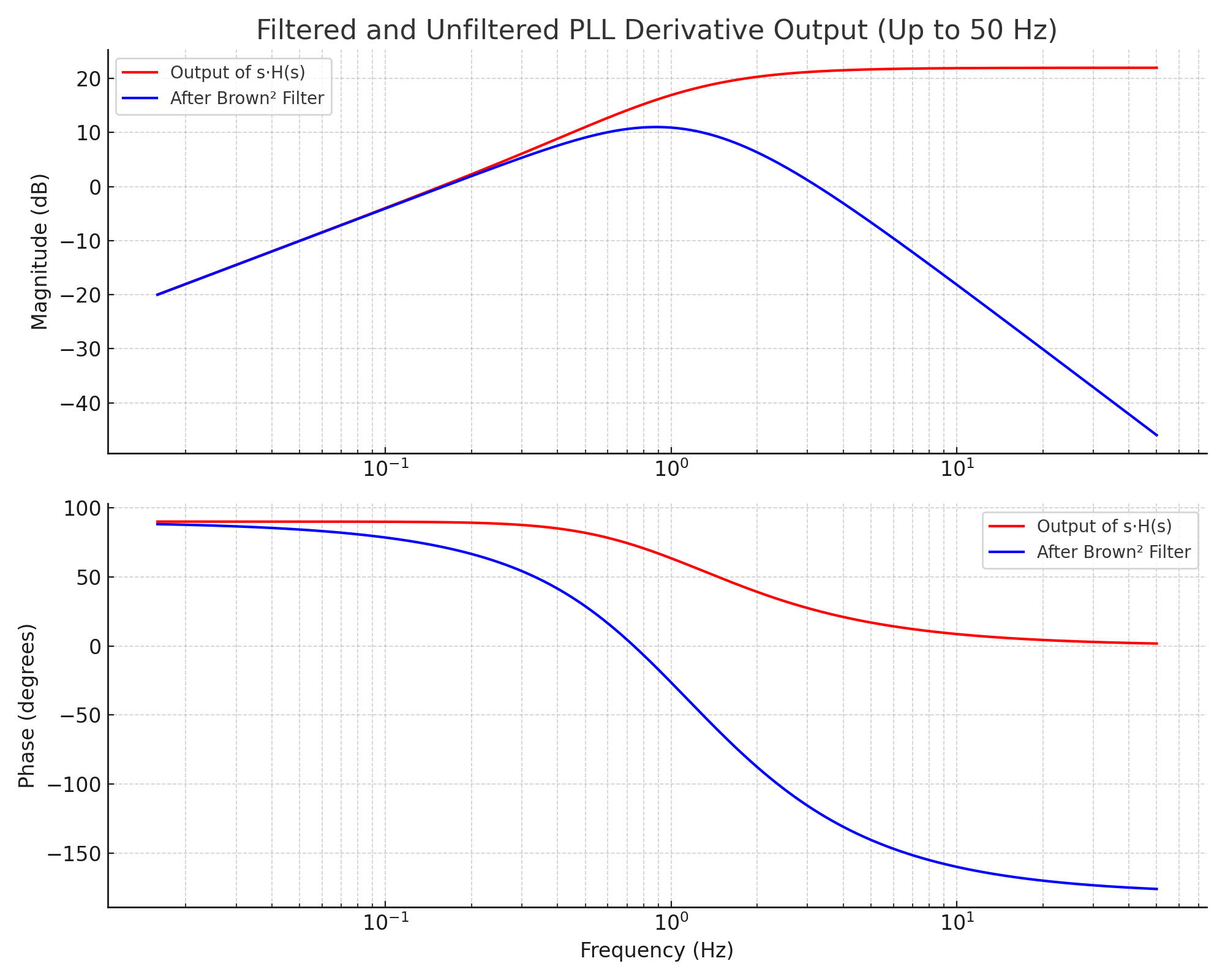
If the output derivative is filtered with a second order Brown low pass filter the system full response become a 4 order system with the following characteristic:

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.



• The raw derivative (s·H(s)) shows significant high-frequency amplification.  
• The Brown² filter reduces high-frequency noise while preserving trend responsiveness near the PLL's natural frequency.

**Title: Relationship Between Alpha and Cutoff Frequency in First-Order Exponential Filters**

1. **Introduction** A first-order exponential filter (also called a one-pole IIR low-pass filter or Brown's exponential smoother) is commonly used in digital signal processing for its simplicity and computational efficiency. The key design parameter, alpha (α\alpha), controls the responsiveness or smoothing level of the filter. Understanding the relationship between α\alpha and the filter's 3 dB cutoff frequency is important for both filter design In the frequency domain the IIR forsdt order filter is defined as:

=

**2. Filter Definition** The filter is defined by the recursive equation:

Where:

* x[n]: input signal
* y[n]: output signal
* : smoothing parameter

**3. Frequency Response and Cutoff Definition** The frequency response in the Z-domain is:

Taking the squared magnitude gives:

The cutoff frequency is defined as the frequency where the gain is reduced by 3 dB:

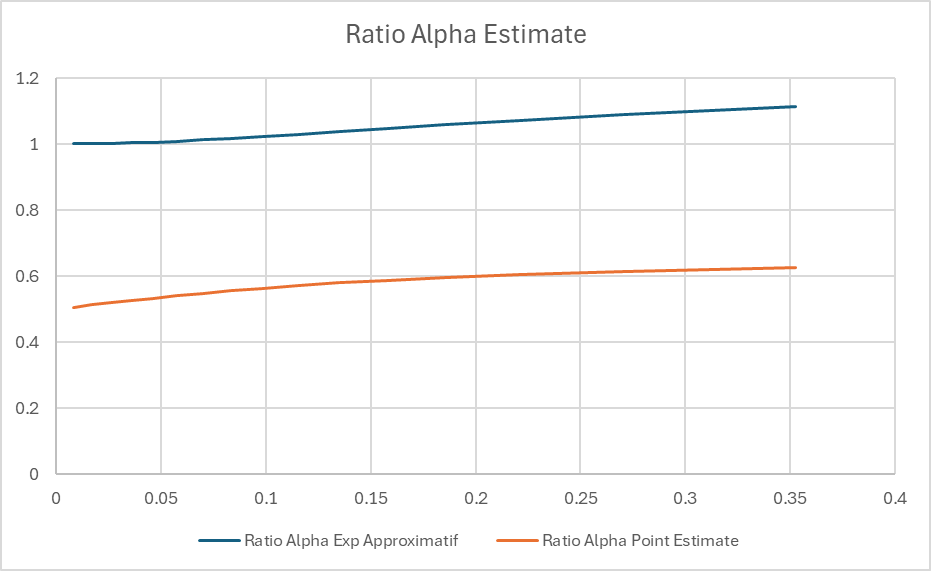
Solving this gives the **exact expression**:

**4. Approximate Formula** A commonly used approximation based on continuous-time filter equivalence via the bilinear transform is:

Where:

* α
* : desired cutoff frequency in Hz
* : sampling rate in Hz

Inverting this gives:



**7. Time to Reach 90% of Final Value (Step Response)** When a unit step is applied to a first-order exponential filter, the output follows:

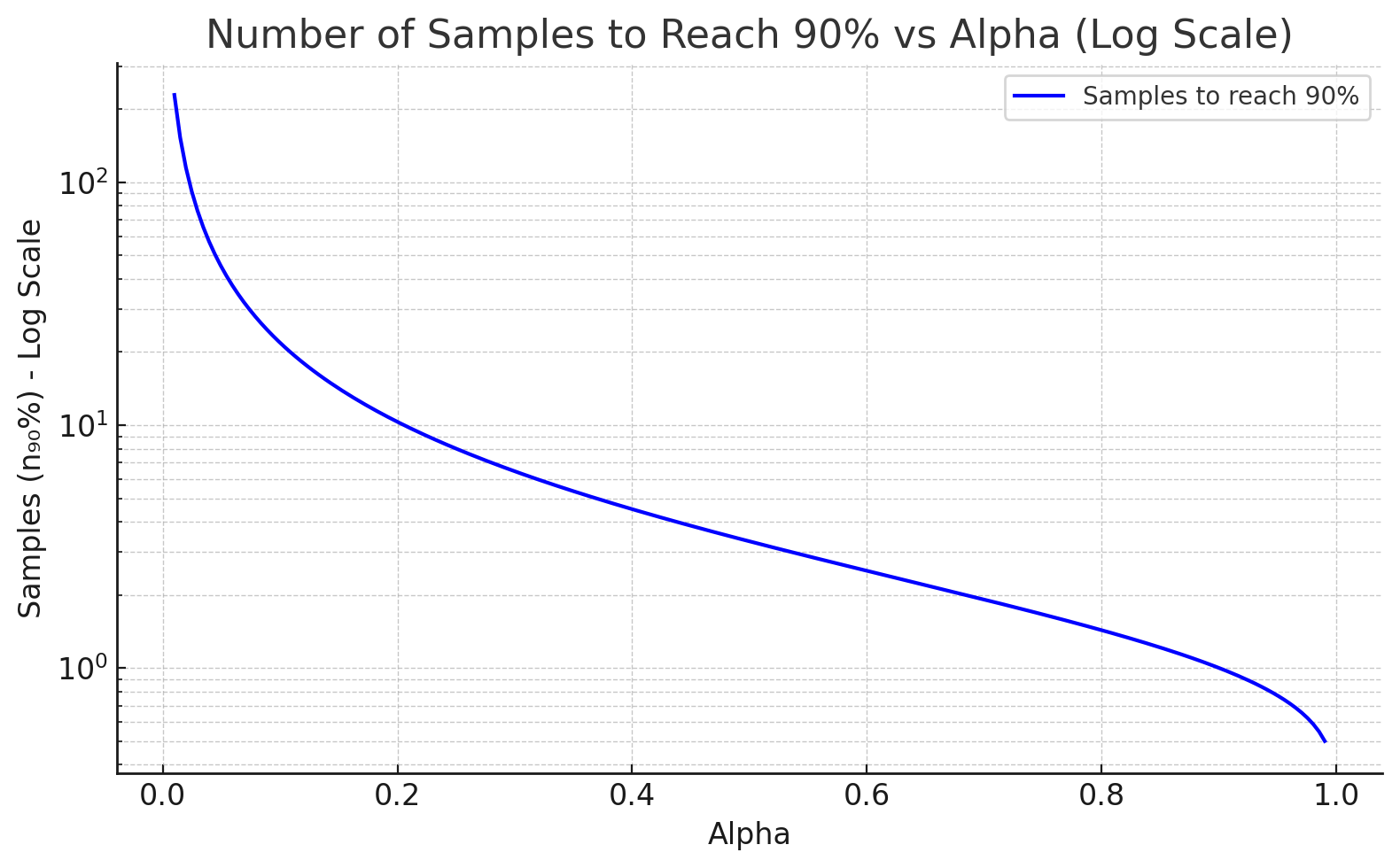
To determine how many samples it takes for the response to reach 90% of its final value:

x

Solving for n:

**Examples:**

|  | **Samples to 90% ()** |
| --- | --- |
| 0.01 | 230 |
| 0.05 | 45 |
| 0.1 | 22 |
| 0.3 | 8 |
| 0.5 | 4.3 |
| 0.9 | 1.05 |



**5. When to Use Each** **Exact Expression:**

* Use when precise control of cutoff is required
* Recommended for analytical work or design specifications where exact cutoff mapping is critical

**Approximation:**

* Sufficient for most practical filter implementations
* Easier to compute, especially when
* Useful when deriving from a desired analog cutoff frequency

**6. Conclusion** The relationship between α and the 3 dB cutoff frequency is fundamental to the effective use of exponential smoothing filters. Both the exact and approximate expressions have their place, depending on the required precision and computational constraints.

**Appendix: Common Approximate Values**

| α | **Approx** |
| --- | --- |
| 0.01 | 0.063 |
| 0.1 | 0.628 |
| 0.5 | 1.81 |
| 0.9 | 2.81 |
| 0.99 | 3.08 |

**5. Equivalent Noise Bandwidth (ENBW)** The Equivalent Noise Bandwidth (ENBW), also called represents the bandwidth of an ideal rectangular filter that passes the same noise power as the actual filter. For the exponential filter:

This is derived by integrating the squared frequency response over the Nyquist interval.

6. Comparison Between and

* Both are bandwidth measures, but serve different purposes:
  + : frequency at which the gain drops to -3 dB
  + : area-equivalent bandwidth for noise
* For small , both increase approximately linearly with
* < for small , but they converge for to 1

**Appendix: Common Approximate Values**

|  | **Approx. (rad/sample)** |
| --- | --- |
| 0.01 | 0.063 |
| 0.1 | 0.628 |
| 0.5 | 1.81 |
| 0.9 | 2.81 |
| 0.99 | 3.08 |

# Frequency Response of Brown's Double Exponential Smoothing Filter

Brown’s Double Exponential Smoothing filter is a two-stage filter designed to capture both the level and the trend in a time series. Unlike a simple exponential filter (single smoothing), it applies exponential smoothing twice, enabling it to adapt to linear trends more effectively.

## Filter Definition

The filter operates in two stages:  
 • First smoothing: S₁(t) = α x(t) + (1 - α) S₁(t-1)  
 • Second smoothing: S₂(t) = α S₁(t) + (1 - α) S₂(t-1)  
The forecast is computed using: x̂(t+1) = 2 S₁(t) - S₂(t)

## Transfer Function (Z-domain)

The Z-domain transfer function of the second smoother is:  
This is equivalent to a second-order IIR low-pass filter.

## Frequency Response

To understand the filter's behavior in the frequency domain, we evaluate the transfer function on the unit circle:  
The magnitude response becomes:

This formula shows that low frequencies (ω ≈ 0) pass through with near-unit gain, while high frequencies (ω ≈ π) are increasingly attenuated. The smaller the α, the stronger the smoothing and the steeper the attenuation.

## Slope and Prediction Insight

The difference between the first and second smoothing stages, S₁(t) - S₂(t), represents the recent trend or slope. This difference is scaled by α / (1 - α) to estimate the slope, allowing the filter to make a one-step forecast that incorporates both the level and trend of the signal:

This trend-following behavior is particularly useful in forecasting applications where the underlying data is expected to follow a consistent linear trajectory in the short term.

Figure: Frequency response (in dB) of Brown’s double exponential filter for various α values.

# Relation Between Laplace and Z Transform in Exponential Filters

This document explains how the Laplace-domain representation of an exponential filter relates to its discrete-time Z-transform equivalent, particularly in the context of Brown's exponential smoothing.

## Continuous-Time (Laplace Domain)

A first-order low-pass filter in the Laplace domain is represented as:  
 Where:  
 • τ is the time constant,  
 • s is the Laplace transform variable.

## Discrete-Time (Z Domain)

The equivalent Z-domain representation of a single exponential smoother is:  
 Where:  
 • α ∈ (0, 1] is the smoothing factor,  
 • (1 - α) = β is the decay factor (memory).

## Mapping Between Domains

The approximate transformation from Laplace to Z-domain using backward Euler is:  
 s ≈ (1 - z⁻¹) / T  
Substituting into the Laplace transfer function yields:  
 H(z) ≈ 1 / [ (τ/T)(1 - z⁻¹) + 1 ]  
Letting α = T / (τ + T), this simplifies to:  
 H(z) = α / [1 - (1 - α) z⁻¹]87

## Parameter Equivalence

To relate α and τ directly:

Thus, larger τ means slower response and smaller α (heavier smoothing).  
Smaller τ gives faster response and larger α (less smoothing).

## Interpretation of β

The term β = 1 - α represents the filter's memory. A larger β means the filter gives more weight to older samples, resulting in stronger smoothing.

## Example: Frequency Response

The graph below shows the frequency response of the Brown double exponential filter in dB for several α values. It demonstrates that smaller α values result in stronger attenuation of high-frequency components.

A graph of different colored lines

AI-generated content may be incorrect.

Figure: Frequency response (in dB) of Brown’s double exponential filter for various α values.

Generated by ChatGPT · OpenAI

**Brown Filter Response for the trend measurement:**

A screenshot of a computer

AI-generated content may be incorrect.

**Brown Filter Response for the Amplitude measurement:**

****

**Brown Filter Response for the Amplitude with no trend component:**

A screenshot of a computer

AI-generated content may be incorrect.

**Equivalent model for a derivative Brown Filter Response:**

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect.

**Frequency Response of PLL Equivalent digital Filter**

**What is the PLL Digital Frequency?**

First let define a signal *x*(*t*) with frequency *fo* signal period *To*. The

digital sequence *x*[*k*] is sampled at discrete times = , where is the sampling period.

Now the signal can be written as:

*x*(k) = *A* cos(ω+ *φ*) = *A* cos(ω+ *φ*)

If we define a new parameters called the digital frequency we can now write the above equation as:

*x*(k) =*A* cos(+ *φ*) = *A* cos(+ *φ*)

In other word, we are simply normalizing the frequency of the physical signal to how fast it was sampled. Note also that the digital frequency Ω is the angular frequency (rad/s) times the sampling period (s/sample), so the units of digital frequency are rad/sample.

As a concept one major reason for using Ω instead of is that signal processing techniques fundamentally take in a sequence of numbers *x*[*n*] and operate on them in the same way regardless of the sampling period. By generalizing the sequences to the parameters k the time become an integer index and we can generalize signal processing techniques from that.

Going back to the equation of a second order ADPLL as presented in

A close-up of a computer screen

AI-generated content may be incorrect.

We have the following transfert fonction et parameters defined as:

A math equations and formulas

AI-generated content may be incorrect.

Another aspect to consider for a ADPLL is that there is three frequencies of concern for system performance consideration. We have the resonance frequency , which appear directly as a parameters in the transfer function H(s), the 3 dB bandwidth and the noise bandwidth equivalent related to the measure in the energy of the noise at the output relative to a perfect windows type bandwidth this parameter (NEB) can be called Bn but it is generally better to define it as as to distinguish it from the parameter .The relation for or can be shown to be dependent of and the damping factor ζ as:

A mathematical equation with numbers and symbols

AI-generated content may be incorrect.

**Or**

=

The reference from the equation above is R. Best, Phase Locked Loops - Design, Simulation and Applications (6th Edition), McGraw Hill, 2007

Also the the 3dB bandwidth can be approximated with this formula extracts from a visual graphic interpretation from ChatGPt:

# 3 dB Bandwidth vs. Damping in Second-Order ADPLL

This document presents an improved model for estimating the 3 dB bandwidth of a second-order analog or digital phase-locked loop (ADPLL) system as a function of the damping factor ζ.

## Measured Bandwidth Data

Measured 3 dB bandwidths (normalized by natural frequency fn):

|  |  |
| --- | --- |
| Damping Factor (ζ) | 3 dB Bandwidth / fn |
| 0.70 | 2.04 |
| 1.00 | 2.46 |
| 1.25 | 2.86 |
| 1.50 | 3.28 |

## Improved Analytical Fit

Rather than relying on a linear approximation, the following analytical model provides a better match:

This expression reflects the natural shape of the transfer function and improves accuracy across a wider range of damping factors compared to simple linear regression.

## Comparison Graph

The following graph compares the measured 3 dB bandwidths to values predicted by the improved formula:

A graph with a red line

AI-generated content may be incorrect.

Generated by ChatGPT · OpenAI

**Frequency Response Curves**

**This figure shows the frequency response of the second-order ADPLL system for different damping factors ζ. The 3 dB points are visually confirmed in each case.**

**A diagram of a frequency response

AI-generated content may be incorrect.**

# Estimating PLL Parameters from Unit Step Response

This sextion explains how to estimate the parameters of a second-order system or phase-locked loop (PLL) based on its response to a unit step input. This method is useful for validating filter performance or estimating internal settings from external observations.

## Applicable System

The method applies to systems with the following transfer function:

This represents the output of a PLL just after the integrator. It is the 'velocity-type' output of a second-order system that can track changing input related to speed.

## Step Response Features to Measure

You can extract the following key features from the unit step response:

* • Peak Value:
* • Final Value:
* • Overshoot:
* • Time to Peak: when the peak occurs

## Formulas to Estimate Parameters

Estimate the damping factor (ζ) from overshoot:

ζ ≈ -ln() / sqrt(π² + ln²())

Estimate the natural frequency (ωₙ) from time to peak:

ωₙ ≈ π / ( \* sqrt(1 - ζ²))

If needed, convert ωₙ to frequency (Hz):

fₙ = ωₙ / (2π)

## Example

Measured response:  
• Overshoot ≈ 16%  
• Time to Peak ≈ 6 seconds  
  
→ ζ ≈ 0.503  
→ ωₙ ≈ 0.53 rad/s  
→ fₙ ≈ 0.085 Hz

## Usage in Code

You can automate this analysis in VB.NET by:  
• Collecting step response data  
• Computing overshoot and time to peak  
• Applying the above formulas  
  
This is useful for validating PLL behavior in live systems or during development.

Generated by ChatGPT · OpenAI

**How to design the PLL second order system based on the first order IIR standard response.**

For the first order IIR filter we use this factor to define the term α:

This is the value of α that will give similar statistics for average and sigma equivalent to a pure FIR filter with a square data window of N points.

The transfer function H(s) is defined as:

With

For that single pole filter the 3 dB bandwidth is also given by:

(rad/sec)

The 3 dB bandwidth of that filter is defined as:

or

**Origin of the Term in First-Order IIR Filters:**

For a discrete-time first-order IIR low-pass filter defined as:

The transfer function in the Z-domain is:

Substituting to analyze the frequency response:

The -3 dB cutoff frequency ​ occurs where the magnitude drops to of the DC gain. Solving the magnitude equation:

Leads to:

This logarithmic relationship arises because the pole of the filter is , and the frequency response depends on how close this pole is to the unit circle in the Z-plane. The term expresses the -3 dB bandwidth in radians/sample based on this pole location.

**Digital Frequency Ω vs. Physical Frequency in IIR Filters**

In digital signal processing, we use:

Where:

* z is the complex frequency variable in the Z-transform
* is the **digital angular frequency**, measured in **radians per sample**

**Relationship Between Digital and Physical Frequency:**

Where:

* = digital frequency (angular)
* f = physical (analog) frequency in Hz
* = sampling rate in Hz

**Frequency Range in Digital Systems:**

* : DC (0 Hz)
* : Nyquist frequency (half of the sampling rate)
* for single-sided spectrum
* for full spectrum

**π**

**Matching Cutoff Frequencies Across Filter Types**

When designing filters of different orders (e.g., first-order and second-order), it is important to compare their cutoff frequencies **in the same domain**.

**Digital Filters:**

If both filters are implemented digitally (using the Z-transform), compare their -3 dB points using **digital angular frequency** . This ensures they exhibit equivalent attenuation characteristics at the same point in the digital spectrum.

**Analog Filters:**

If you're comparing analog prototypes (in the Laplace domain), use analog angular frequency ω in. This applies before any conversion to digital.

**Mixed Domain Designs:**

If you design in the analog domain and then convert to digital (e.g., using the bilinear transformation), keep in mind that:

This non-linear relationship means:

* Equal analog cutoff frequencies do **not** map directly to equal digital cutoff frequencies
* Matching must account for **frequency warping** introduced by the transformation

**Summary Table:**

| **Goal** | **Compare In** | **Frequency Type** |
| --- | --- | --- |
| Analog design |  | Analog frequency |
| Digital design |  | Digital frequency |
| Mixed design (analog and digital). | Use bilinear transform | Frequency warping applies |

**Comparison: First-Order IIR vs. Second-Order DPLL Cutoff Alignment** Matching Cutoff Frequencies Across Filter Types

When designing filters of different orders (e.g., first-order and second-order), it is important to compare their cutoff frequencies in the same domain.

Digital Filters:

Use digital angular frequency Ω to compare -3 dB points.

Analog Filters:

Use analog angular frequency ω in .

Mixed Domain Designs:

Matching must account for frequency warping due to the non-linear transform.

Summary Table:

|  |  |  |
| --- | --- | --- |
| Goal | Compare In | Frequency Type |
| Analog design |  | Analog frequency |
| Digital design |  | Digital frequency |
| Mixed design (analog → digital) | Use bilinear transform | Frequency warping applies |

Comparison: First-Order IIR vs. Second-Order DPLL Cutoff Alignment

First-Order IIR Filter:

For that single pole filter the 3 dB bandwidth is also given by:

(rad/sec)

The 3 dB bandwidth of that filter is defined as:

or

Second-Order DPLL Loop Filter:

This equation can also be represented as the paper by Shayan and Lee-Ngoc “All digital PLL: Concepts, design and applications”

With and related to the S frequency domain (jof a second order analog PLL filter

Where we have:

Solve for Ω where:

Conclusion:

- Use Ω to compare frequency behavior in digital systems.

- Second-order filters offer steeper roll-off but must be aligned in bandwidth.

**Comparison: First-Order IIR vs. Second-Order DPLL Cutoff Alignment**

When comparing a first-order IIR filter with a second-order DPLL-based loop filter, it is crucial to define the target -3 dB cutoff frequency in terms of digital frequency Ω to ensure consistent behavior.

**First-Order IIR Filter:**

A standard one-pole low-pass IIR filter is defined by:

The pole is at z = 1 - α, and the -3 dB cutoff occurs approximately at:

The equation becomes:

The term N here is the number of points that the user needs for data filtering. The relation ensure that the statistics at the output of the filter will be equivalent to the one of an equivalent FIR first order filter using a square window of N points. This establishes the statistical relation to the equivalent FIR filter.

This gives also a smooth single-pole roll-off at -20 dB/decade.

And another relation that is needed is the equivalence in bandwidth between the IIR filter and the second order IIR DPLL filter. The number of point N for the second order filter should be scaled or adjusted to ensure the 3 dB bandwidth equivalence. The statistic between the first order and second order may diverge slightly but the difference will become mainly related to the order of the filter and the sharper roll-off filtering of -40 dB/decade.

**Second-Order DPLL Loop Filter:**

As defined in the work of Shayan and Le-Ngoc ("All-Digital PLL: Concepts, Design and Applications"), the second-order digital loop filter has the transfer function:

Where C₁ and C₂ are also related to the analog S-domain transfer function:

(S) =

.

.

defined as:

C₂ = 2ζωₙTₛ = 2ζ1

C₁ = C₂² / (4ζ²)

To match the bandwidths of the first and second order filters, solve for Ω where:

Additionally, the 3 dB bandwidth for the second-order analog PLL filter in the analog domain can be approximated by:

This 3 dB cut-off frequency can be mapped to the digital filter equivalent using the bilinear transformation. We then have:

Or

and by substitution it follows:

As a first approximation the amplitude of the analog second order filter at the resonnace frequency is always very close to unity even with a damping factor variation of between 0.7 to 1.2. This is demonstrated in the following graph

A screen shot of a graph

AI-generated content may be incorrect.

Next is the frequency response for the same filter for a damping factor of 1

A graph on a screen

AI-generated content may be incorrect.

In the current application the DPLL filter is expected to be used mostly as a fast roll off low pass filter with the main advantage being its fast response and relatively small delay. Under this consideration it appears to be reasonable to assume that the resonance frequency of the second order filter should match the 3 dB bandwidth of the first order filter i.e.

From the discussion above for the first order filter we have:

Again, the value for here is to ensure statistical equivalence to a square windows FIR filter. After some transformation we the equation for the digital frequency becomes:

Since we need we can write the following relation between the first and second order filter:

Using the linear transformation this yield to the 3 dB bandwidth of the ADPLL defined as:

From this we can evaluate the parameters C2 in function of N as:

C₂ = 2ζωₙTₛ = 2ζ

# Taylor Series Expansion of

We aim to find the Taylor series expansion of the logarithmic expression:

This can be rewritten as:

Using the known identity for the logarithm of a ratio:

Set x = 1/N, which is valid for N > 1. Then:

Therefore, the Taylor series expansion is:

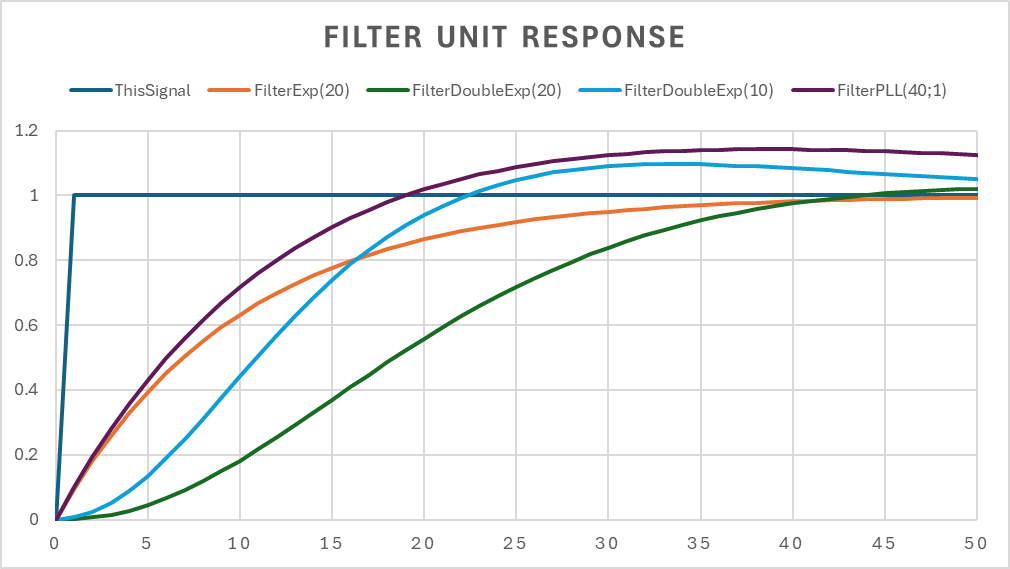
This expansion becomes very accurate for large N. and the equation above for C2 could be written as:

C₂ = 2ζωₙTₛ = 2ζ

The previous filter code used the approximation factor 2/N for estimating the factor C₂ but new code has been updated to reflect the exact equation formula for the factor C2. The effect for the filter response is minimal but now matches the exact theoretical formula equivalence between the 3 dB bandwidth of the IIR filter and the resonance frequency of the second order filter.

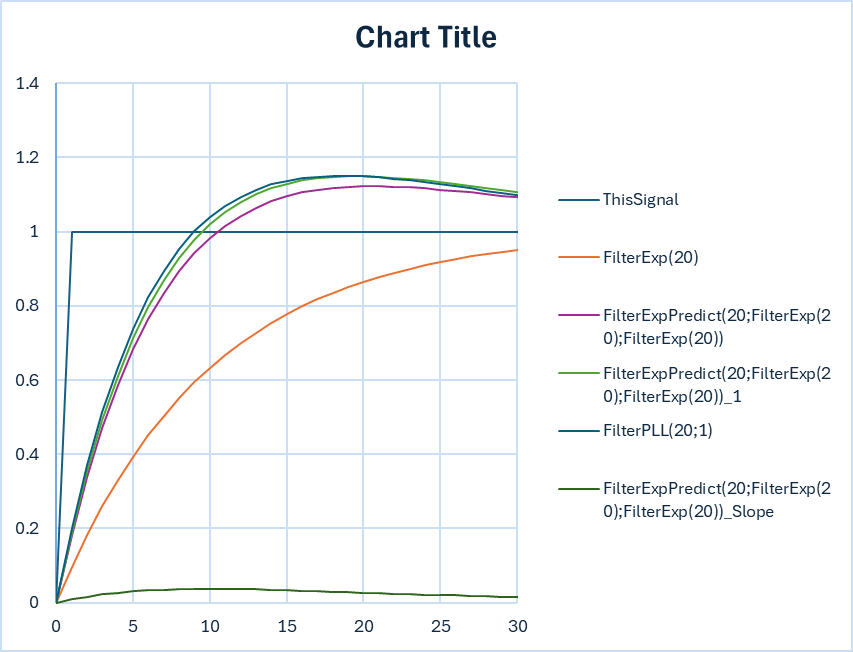
Note that the implication is that the corresponding 3 dB bandwidth for the second order filter is larger than the equivalent IIR first order filter of the current implementation. In case the frequency bandwidth at 3 dB is needed, it could be achieved approximatively by doubling the number of samples parameters N for the DPLL filter. The result for a unit response time will become very similar to the single pole filter as the dominant factor for the pulse response is the bandwidth of the filter. The difference is the expected overshoot of the DPLL and the response of the unit signal becomes very similar as demonstrated by the following response of the two filters:

Another aspect to consider is the double filter exponential build by cascading two exponential first order filter with the same 3 dB bandwidth or parameters. The unit response is shown in the figure below compared to the single IIR filter with N=20 for all filter except the DPLL set a 2N. The result shows the slower response that can be improved by reducing the factor N by factor 2 as shown below. We then have an equivalent second order filter with slightly slow response speed but reduced overshoot.:

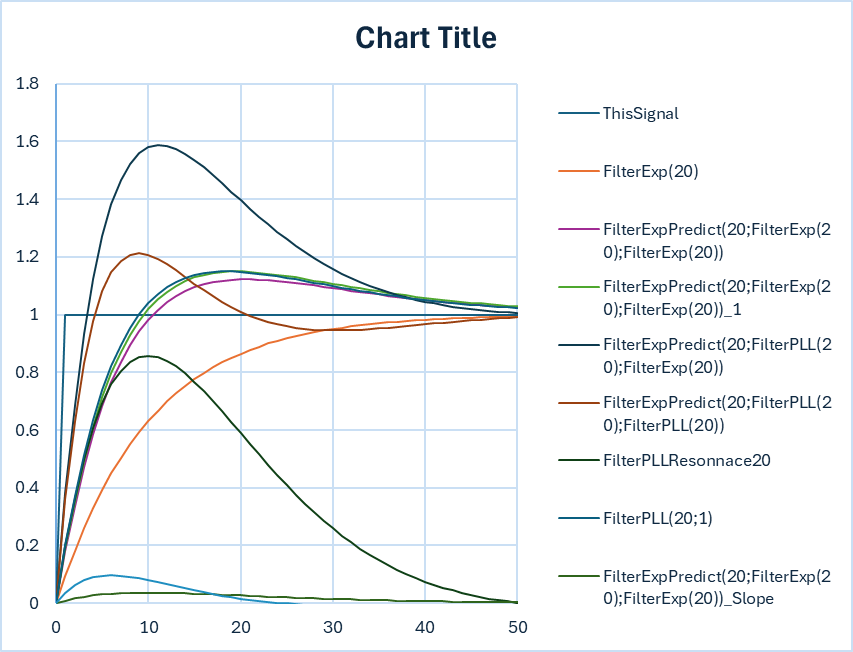


The other filter of interest is the FilterExpPredict based on the Brown prediction model of two IIR first order exponential filter. In this case we have a response that is very similar to the equivalent second order DPLL filter as shown below:

Next on this filter an interesting aspect is that the Brown exponential filter with a prediction of 1 (see line orange vs the brown line is essentially and in all practical cases identical to the DPLL response. In practice, filter could be used with practically the same result i.e. :



Some other more unusual filter are possible if one combine the various filter with the Brown FilterExp standard filter are shown here



**Amplitude Response for FilterExpPredict**

This filter amplitude is close in response to the PLL second order but with a smaller overshoot, and an attenuation at frequency 10 of -14dB.

A black rectangular sign with white text

AI-generated content may be incorrect.

A graph with lines and numbers

AI-generated content may be incorrect.

This filter is easily modified to extract the trend of the filtering process. The Trend amplitude response for FilterExpPredict is given below. Note the -20 dB drop at frequency 0.1 and 10. However there is an attenuation at the frequency 1 of -8.5 dB and if taken as a reference the drop at frequency 10 is effectively only of -11.5. The trend represents an estimate of the slope of the filter center around frequency 1 essentially acting as a bandpass filter operating with an approximative loss of -8.5 dB at the center frequency.

In a digital filter the trend output of the filter is also adjusted for calibration in unit/sample by the factor 2/(N-1). The filter output then represents the best slope estimate that can be used if needed for extrapolation to the next samples. When the trend is used the signal output becomes very similar to the result that would be obtained using a second order DPLL operating with a damping factor of 1. As demonstrated below in the FilterPLL section

A screenshot of a computer

AI-generated content may be incorrect.

FilterExpPrediction slope response to a Unit Step

FilterExpPrediction slope response to a Ramp Step (converge toward 1)

A screenshot of a computer

AI-generated content may be incorrect.

FilterExpPrediction Frequency Response

A graph with a blue line

AI-generated content may be incorrect.

Note that if we test this system with a Unit ramp the long-term response tends toward 1 i.e. the slope per sample is one:

**Amplitude Response for the FilterPLL**

The following graph represents the response of a standard PLL with a resonance frequency and damping factor respectively of 1. We observe a slight overshoot of approximately 15% and a drop at frequency 10 of about 14.5 dB similar the brown exponential filter presented above.

A graph on a black background

AI-generated content may be incorrect.

This filter derivative is also easily obtained if the signal is taken just before the input to the integrator. In that case the frequency response becomes:

A graph on a black background

AI-generated content may be incorrect.

As expected, this filter becomes a high pass filter with a high frequency response of +6 db. In most cases this filter can be use in a very similar way that the Brown filter if the output of the input to the integrator is filtered with a double exponential filter as indicated below. We note a very sinusoidal shaped pulse response with a slight undershoot. The High frequency attenuation is large with a factor of -28 db relative to the center frequency of the filter. The rejection a the low frequency is not as goo at -14dB relative to the center frequency. This design can be vey good for high frequency rejection.

The ramp test indicates that this system can also be used to measure the slope with accuracy as indicated below with the long-term response showing a slope of one as expected. The stabilization period, however, appear to be slightly better than the brown filter with a stabilization at around 4-5 samples versus 5-6 for the brown filter.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A graph with a blue line

AI-generated content may be incorrect.

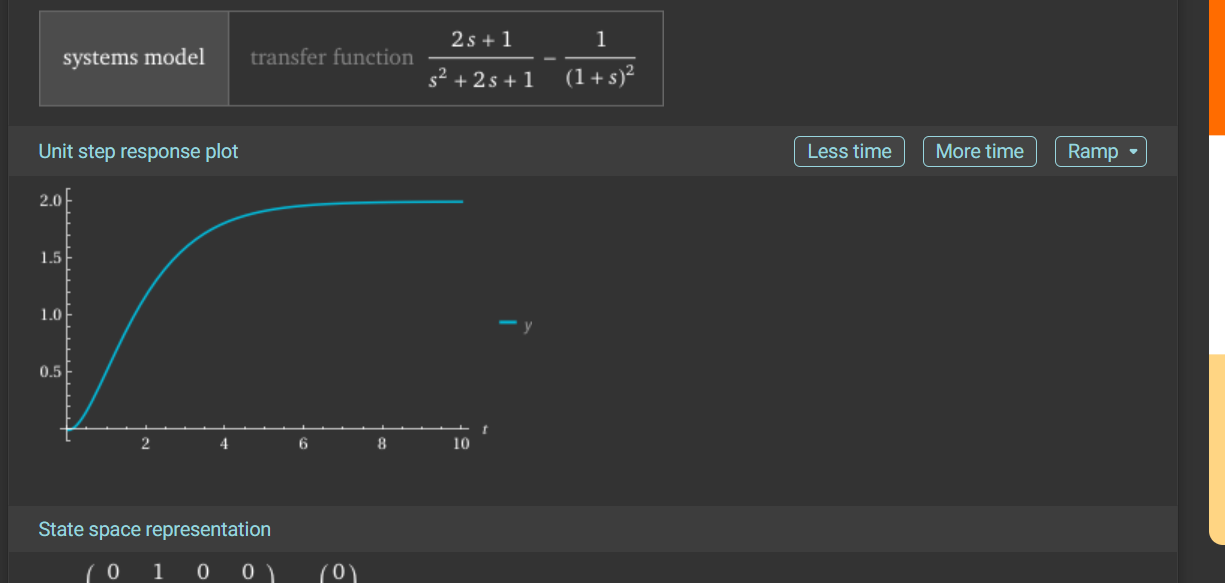
The result obtained when we compare the DFPP response to the Brown exponential filter when a ramp is applied to the input in essentially very close with as expected a slight overshoot for the DPLL response.

**The resonance PLL filter**

A variation of the PLL filter is slightly different than the brown exponential filter can be obtained if only the derivative portion of the PLL response is kept in the function. The filter can be implemented by subtracting the PLL response using a double exponential filter. The result of the transfer function of that filter is typically of that form:

This filter is easy to implement, and the formula is easily matched and exact if the damping factor is defined as 1. However, even for a damping factor of between 0.7 and 1.5 the result is still acceptable in an application where a perfect signal response is required. The aspect to consider is that the slope estimate for this filter should be divided by 2 for an exact slope measurement. The following graph presents the different results:





**Theory related to the Laplace Transform and the method used to estimate the initial and final value of a transform H(s)**

# Initial and Final Value Analysis using Laplace Transform

This document demonstrates how to compute the initial slope and final value of a system's response using Laplace Transform. These techniques are useful in control systems and signal analysis to determine system behavior at the beginning and end of the response.

## System Description

Given Transfer Function:

H(s) = (2s + 1) / (s² + 2s + 1)

Input: Unit Step Function

U(s) = 1 / s

## Output in Laplace Domain

Y(s) = H(s) × U(s) = (2s + 1) / [s × (s² + 2s + 1)]

## Initial Slope (dy/dt at t = 0⁺)

Using the Initial Derivative Theorem:  
 dy(t)/dt|ₜ₌₀⁺ = limₛ→∞ [s² × Y(s)] = limₛ→∞ [s²(2s + 1) / (s × (s² + 2s + 1))]

Simplifying:  
Numerator: s²(2s + 1) = 2s³ + s²  
Denominator: s(s² + 2s + 1) = s³ + 2s² + s  
As s → ∞, divide numerator and denominator by s³:  
limₛ→∞ [(2 + 1/s) / (1 + 2/s + 1/s²)] = 2

→ Initial slope at t = 0⁺ is \*\*2\*\*.

## Final Value (y(t) as t → ∞)

Using the Final Value Theorem:  
y(∞) = limₛ→₀ [s × Y(s)] = limₛ→₀ [s × (2s + 1) / (s × (s² + 2s + 1))]

Canceling s:  
limₛ→₀ (2s + 1) / (s² + 2s + 1) = 1 / 1 = 1

→ Final value as t → ∞ is \*\*1\*\*.

## Summary

• Initial Slope at t = 0⁺: 2

• Final Value as t → ∞: 1

**Discrete-Time (Z-domain) Equivalent Concepts**

In the Z-domain, equivalent theorems exist for analyzing the initial and final behavior of discrete-time systems. These theorems mirror the Laplace-domain analysis but are adapted for sampled systems.

**Final Value Theorem (Z-domain)**

For a stable causal system with output Y(z) and input X(z), the final value of y[n] is given by:

limₙ→∞ y[n] = lim\_{z→1} (1 - z⁻¹) Y(z)

This is analogous to taking the limit as s → 0 in the Laplace domain. It assumes all poles of (1 - z⁻¹)Y(z) lie inside the unit circle, except possibly a simple pole at z = 1.

**Initial Value Theorem (Z-domain)**

The initial value of y[n] (i.e., y[0]) is given by:

limₙ→0⁺ y[n] = lim\_{z→∞} Y(z)

This matches the intuition that early behavior is governed by the high-frequency (large z) characteristics.

**About z = -1 (Nyquist Frequency)**

The point z = -1 corresponds to e^{jπ}, i.e., the Nyquist frequency in discrete-time systems. This point is important in frequency response analysis but is not used for determining time-domain behaviors such as initial value, final value, or slope. Therefore, it is not applicable to initial/final value theorems.

**Discrete-Time Approximation of Slope**

There is no direct analogue in the Z-domain for computing the initial slope as in Laplace. Instead, the slope at the start can be approximated using finite differences:

dy[n]/dt ≈ (y[1] - y[0]) / Tₛ

Where Tₛ is the sampling period. This gives an estimate of the rate of change at the beginning of the signal.

# Analysis of Final Slope Behavior

While initial and final values can be directly extracted from Laplace and Z-domain transforms, the slope at the final value (i.e., the derivative as time approaches infinity) cannot be directly extracted.

## Laplace Domain Perspective

The Final Value Theorem provides:  
 lim\_{t→∞} y(t) = lim\_{s→0} s × Y(s)  
To compute the slope at t → ∞ (i.e., dy/dt as t → ∞), one might consider:  
 lim\_{s→0} s² × Y(s)  
However, this expression actually yields the initial slope (as t → 0⁺), not the final slope. Laplace transforms emphasize initial behavior due to the high-frequency weighting of s.

## Z-Domain Perspective

Similarly, in the Z-domain, initial and final values are computed using limits at z → ∞ and z → 1 respectively. There is no direct method to extract the final slope (i.e., the difference y[n] - y[n−1] as n → ∞) from Y(z).

## Interpretation for Stable Systems

For most stable systems, the output y(t) (or y[n]) converges to a steady value as time increases. This implies that the slope (rate of change) at large time approaches zero:

lim\_{t→∞} dy(t)/dt = 0

This is typically the case for causal, asymptotically stable systems. If a system exhibits exponential decay like y(t) ≈ A·e^(−αt) near the end, then the final slope also decays exponentially to zero.

## Conclusion on the use of the Laplace and Z transform to obtain the initial and final value analysis.

While Laplace and Z-transforms provide valuable tools for analyzing system dynamics, extracting the slope at the final time directly from these domains is not possible. Such information typically requires knowing the explicit time-domain form of y(t) or its long-term decay behavior.

**Other consideration related to the use of the Laplace transform:**

We often use Laplace Transform for evaluating the response of an analog system equivalent which corresponds in realities to an infinite sampled system. In practice we are dealing with a sampled system often better described via the Z transform. The Z transform talke into account the limit imposed by the sampling. When we use the s transform in a sampled system the goal is to use the known properties of certain filter in the analog world to understand the system and then if possible adapt the digital system in an equivalent often very similar response via the Z transform.

In all this document the s transform is often use to extract key information and concept and then the Z transform relation is use to extract the final relation for any filter. An example is the use of the standard second order analog PLL response to understand the quasi-equivalent comportment of the digital system and for this we use:

H(s) = (2s + 1) / (s² + 2s + 1)

Based on the analysis done in other section of this document with the filter above we have from the corresponding relation to the Z transform, assuming =1 we can write as:

This function H(s) above corresponds to a digital resonance frequency in the z domain of . In that situation we can extract the value of N and show that it is defined as:

Or environ 2 samples. It is important to realize that the property derived from this H(s) are still valid in the z transform or the samples system but only if we used a larger sampling rate normally greater than 5 or the DPLL become unstable. The current code detects the stability range and automatically gives an error if the parameters N is too small. It is recommended to use at least N=5 as the minimum filter rate for the DPLL filter.

+-

**Conclusion:**

- For comparing different filter structures in a digital system, digital frequency Ω is the correct axis of comparison

- Use established PLL formulas (e.g., Shayan & Le-Ngoc) for second-order bandwidth estimation

- Solve numerically or graphically for frequency where both filter magnitudes reach 1/√2 to ensure matched 3 dB performance

More detailed plots and step responses can further clarify their behavior, especially in time-domain tracking or stability analysis.