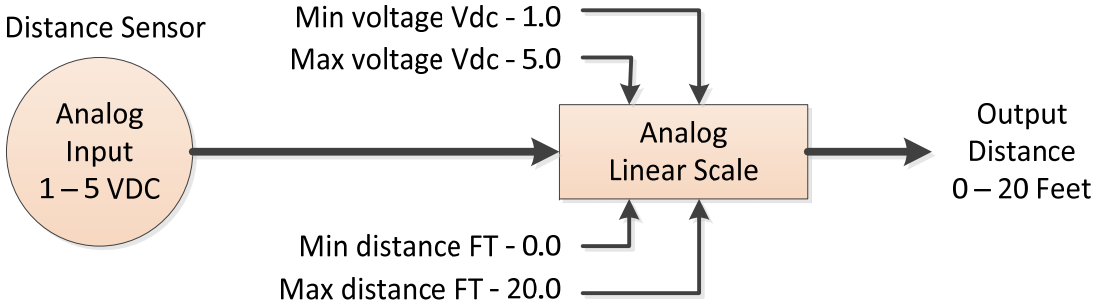
Control System Training

Module 9 – Analog Data Acquisition

Copyright Notice

Analog Values – Acquire and Scale

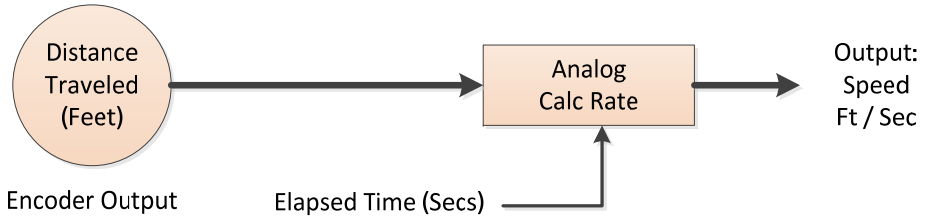
- **Values are often read as:**
  - Voltages, Counts, other meaningless numbers
- **Convert to usable units**
  - Feet, FT/Sec, Degrees
    - These values may be shown to drivers on dashboard
- **Could have a "Standard" module to do conversion**
  - Inputs are High and Low "engineering" units values corresponding to High and Low voltage values./
  - Inputs could be M and B for equation:
    - Y = M * X + B

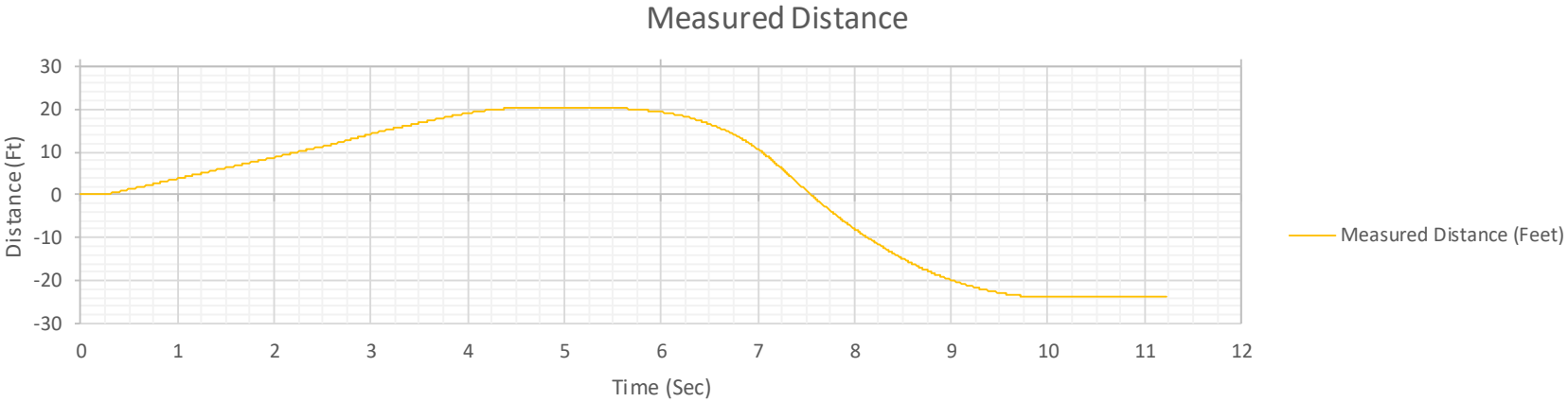


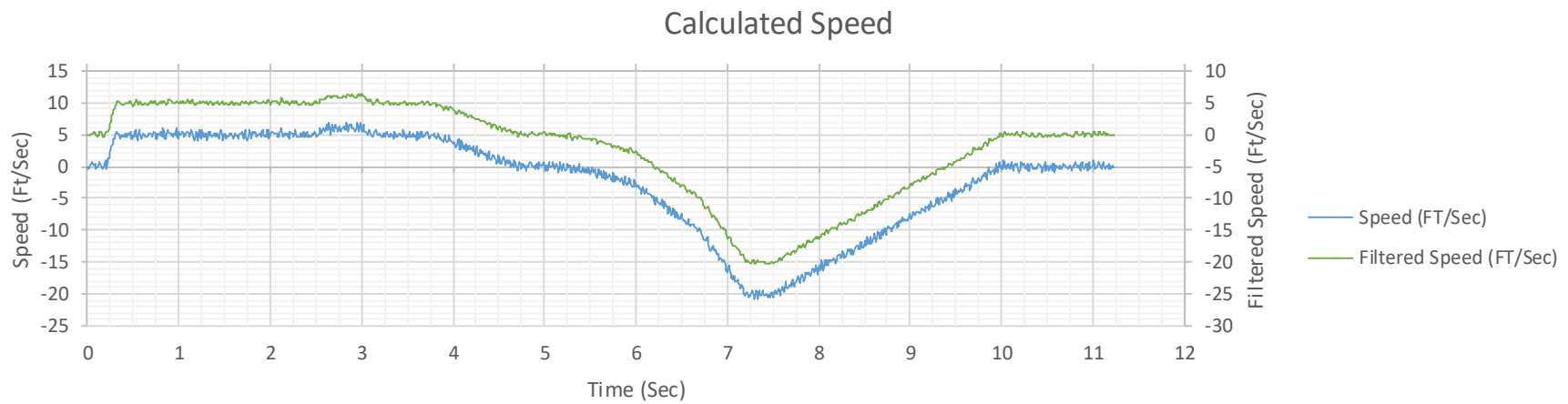
Analog Values – Calc Speed From Distance

- **Numerical differentiation.**
- **Speed = d Distance / dt**
  - Numerically: Speed = (D2 – D1) / (T2 – T1)
- **Acceleration = d (Speed) / dt**
- **Could have a "Standard" module to do this**
  - Input current distance and current time
- **Differentiation amplifies the noise in a signal and can introduce additional noise from time inaccuracy. Consider filtering the resulting rate.**
- **Note: It appears that the built in encoder distance outputs do NOT seem to work very well. The signal to noise ratio is not very good.**

Rate Calc Sample







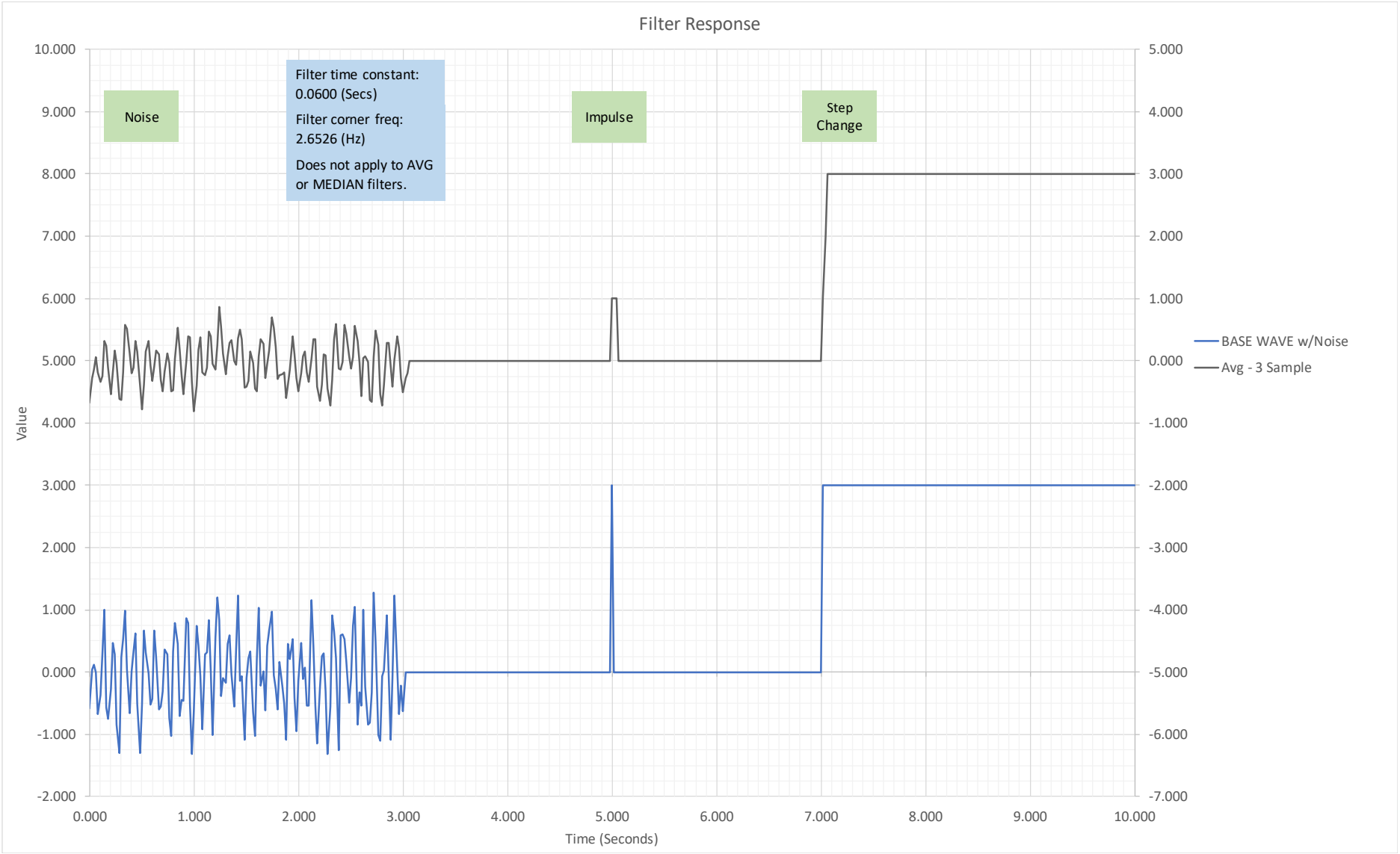
Filtering Analog Values – Why and What

- **Analog values can be noisy**
- **Noise comes from many sources**
  - Wiring induced electrical noise
  - Sensor induced noise
  - Timing jitter induced noise when calculating speed or other differential
- **Filters reduce noise at the expense of latency (time delay).**
  - Some hardware may have filtering, but it might not be enough.
  - Increased filtering means less noise, but more phase shift (time delay) of actual value.
    - All filtering uses previous values as part of the smoothing calculation.
  - Usually choose trade-offs to allow some noise, but not delay signal too much.
  - One option is to scan the inputs twice as fast as they need to be used to reduce phase shift.

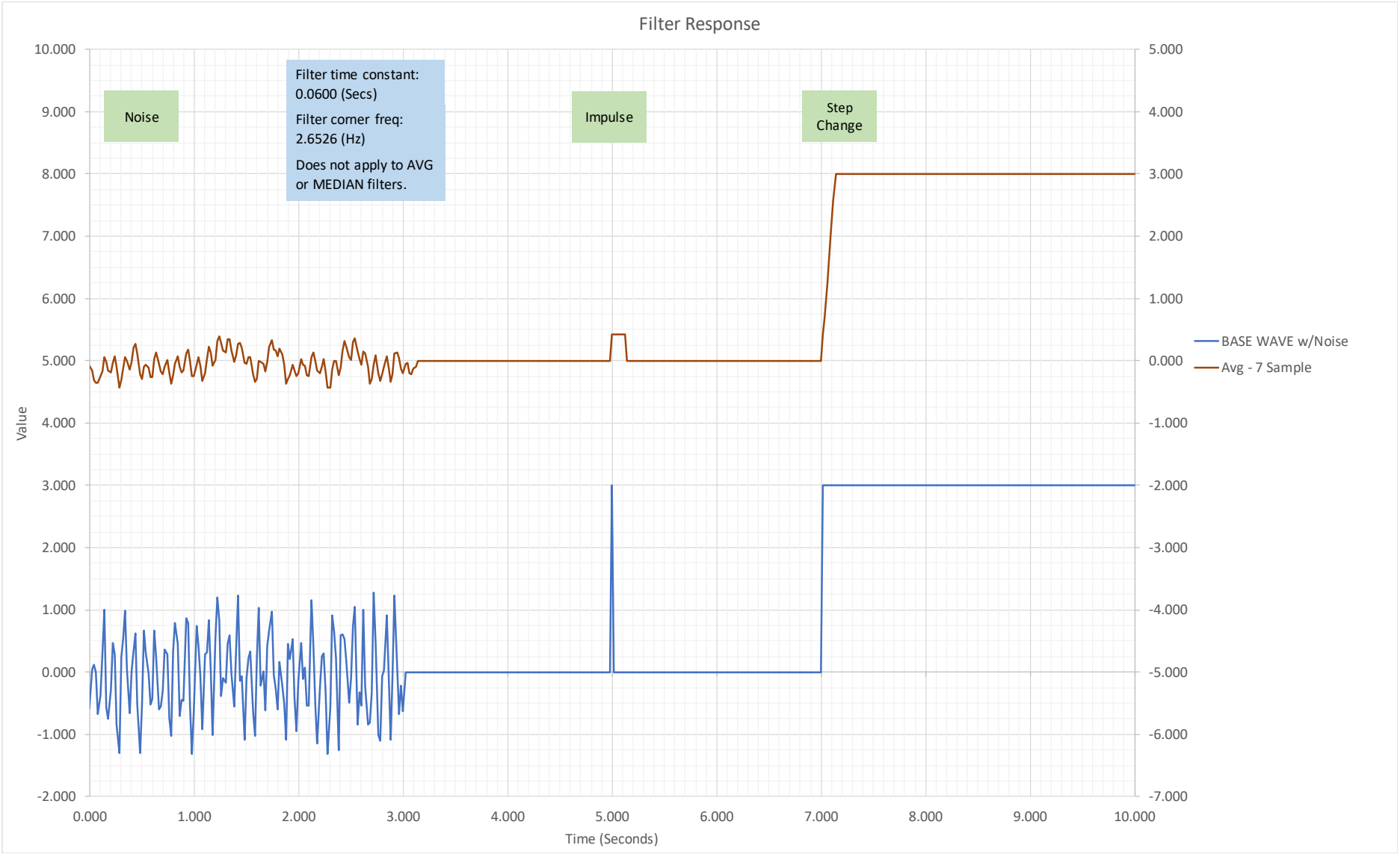
Average Filter

- **Average filter**
  - Calculate the average of the last *n* scans
  - Use *n* between 3 and 7 (Can use others.)
  - Performs data smoothing, but doesn't exclude data "spikes"

3 Sample Average Filter Response



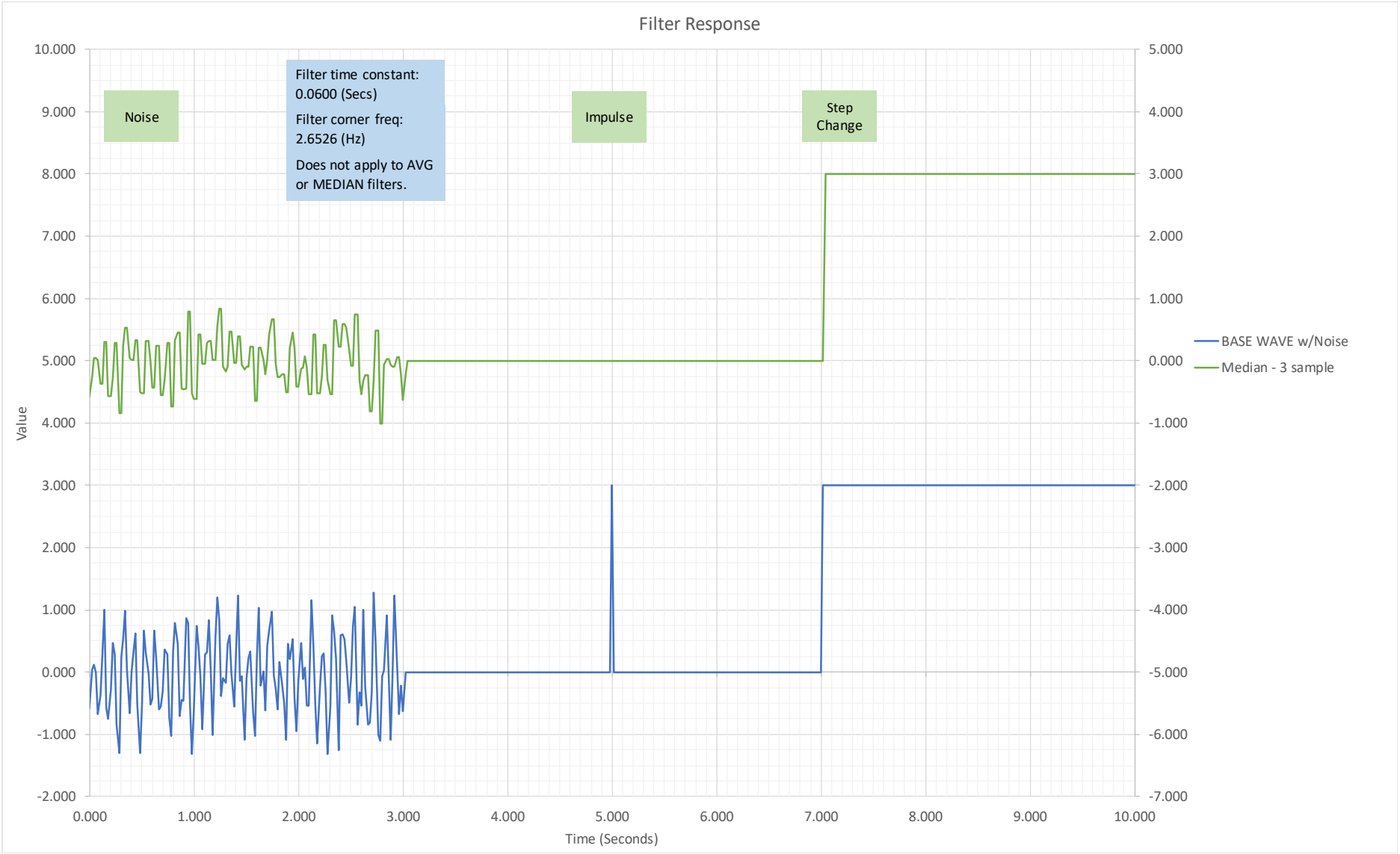
7 Sample Average Filter Response



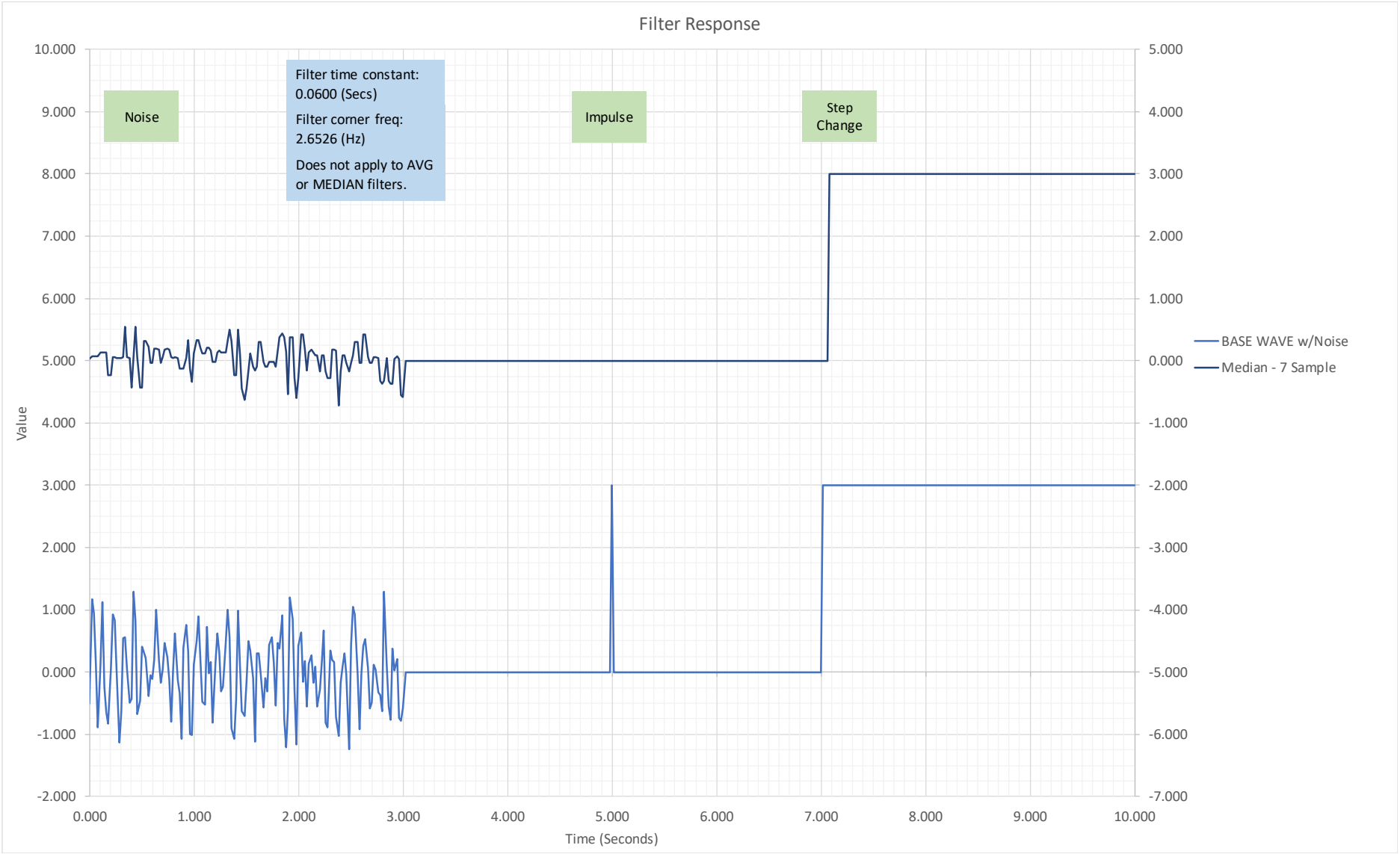
Median Filter

- **Median filter**
  - Use the median of the last *n* scans
  - Use *n* between 3 and 7 (Can use others.)
  - Better spike rejection, but risk of step changing values instead of smooth movements.

3 Sample Median Filter Response



7 Sample Median Filter Response



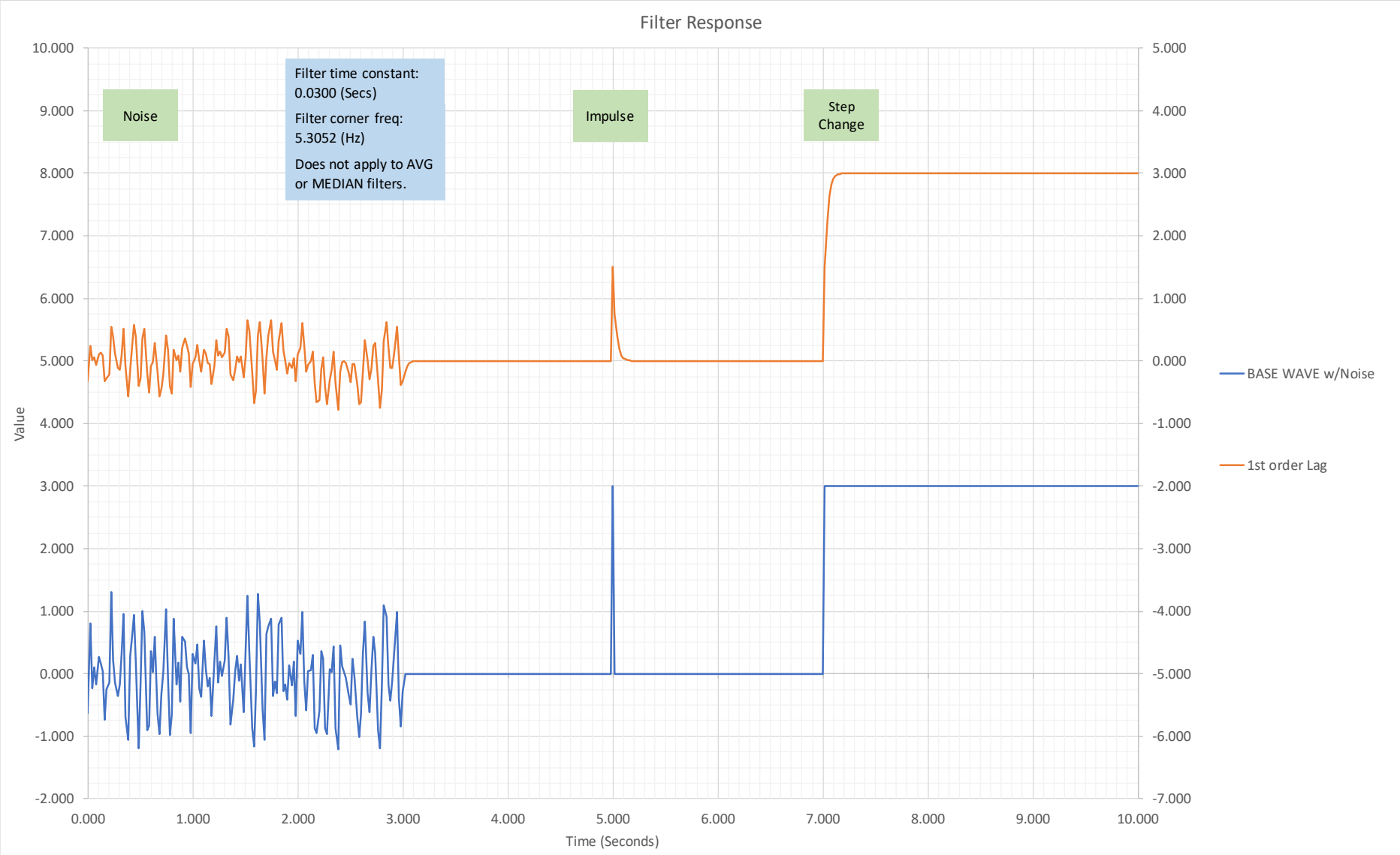
1st Order Lag Filter

- **These are "Low Pass Filter" - Filters out higher frequency noise**
  - Filtering based on time constant (Seconds). Translate to "cutoff frequency" (Hertz)
- **When a step change occurs**
  - Reaches 63% of value in 1 time constants
  - Reaches 99% of value in 5 time constants

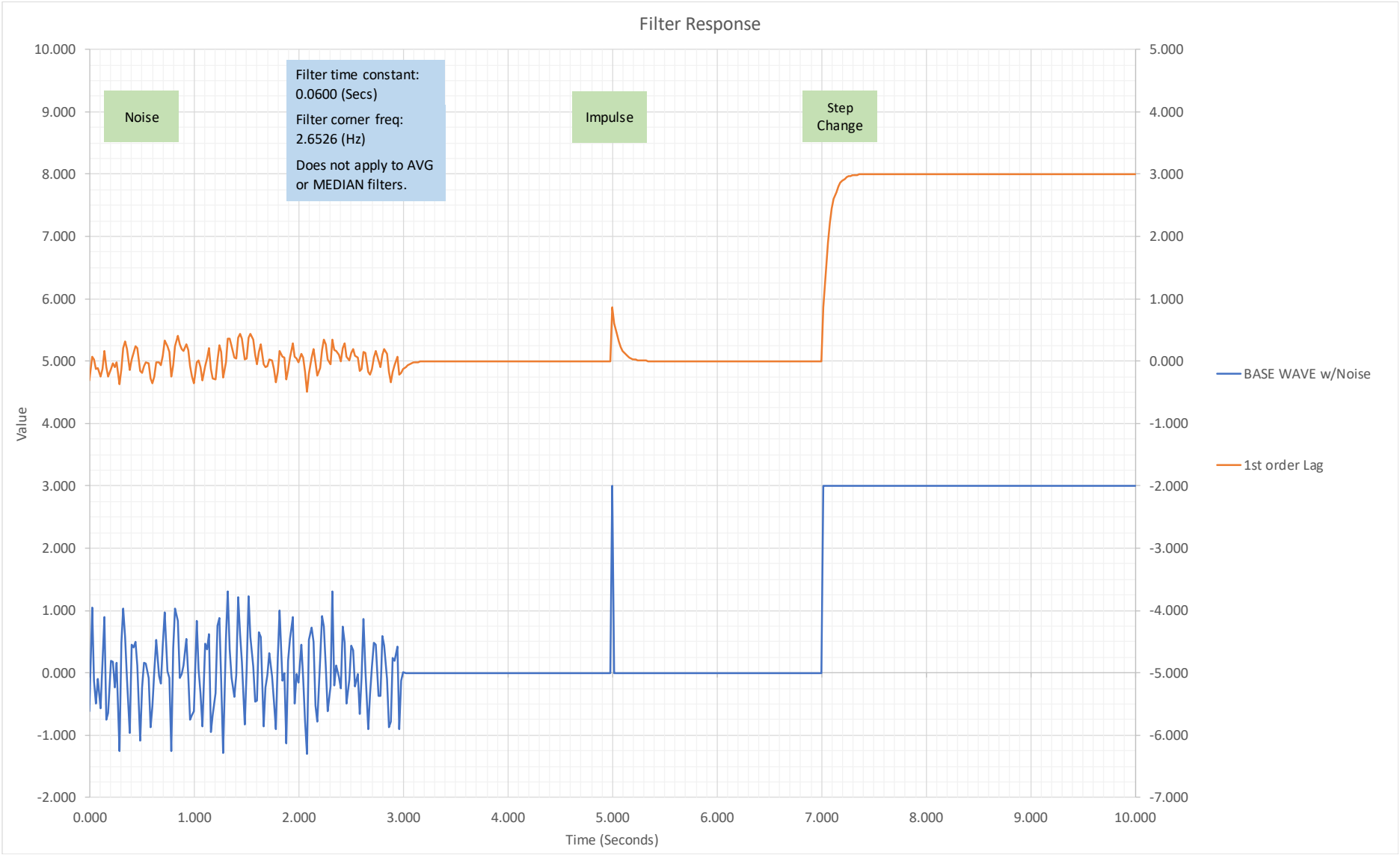
1st Order Lag Filter

- **Equation (numerical implementation):**
  - Tc (Secs) = Time Constant, specified by user
  - Ts (Secs) = Sampling time, specified or calculated
  - Wd (Hz) = Cutoff (corner) Frequency = 1 / (2 * PI * Tc)
  - A0 = 2 * Ts / (Ts + 2 * Tc)
  - B1 = (2 * Tc – Ts) / (2 * Tc + Ts)
  - $Y_n = A0 * X_n - B1 * Y_{n-1}$
- **Trivia:**
  - This equation is created from a bilinear transformation of the laplace transform representation of the transfer function equation. (Think differential equations…)
  - The sum of the constants, A0, A1, -B1, -B2, etc. has to be = 1.0

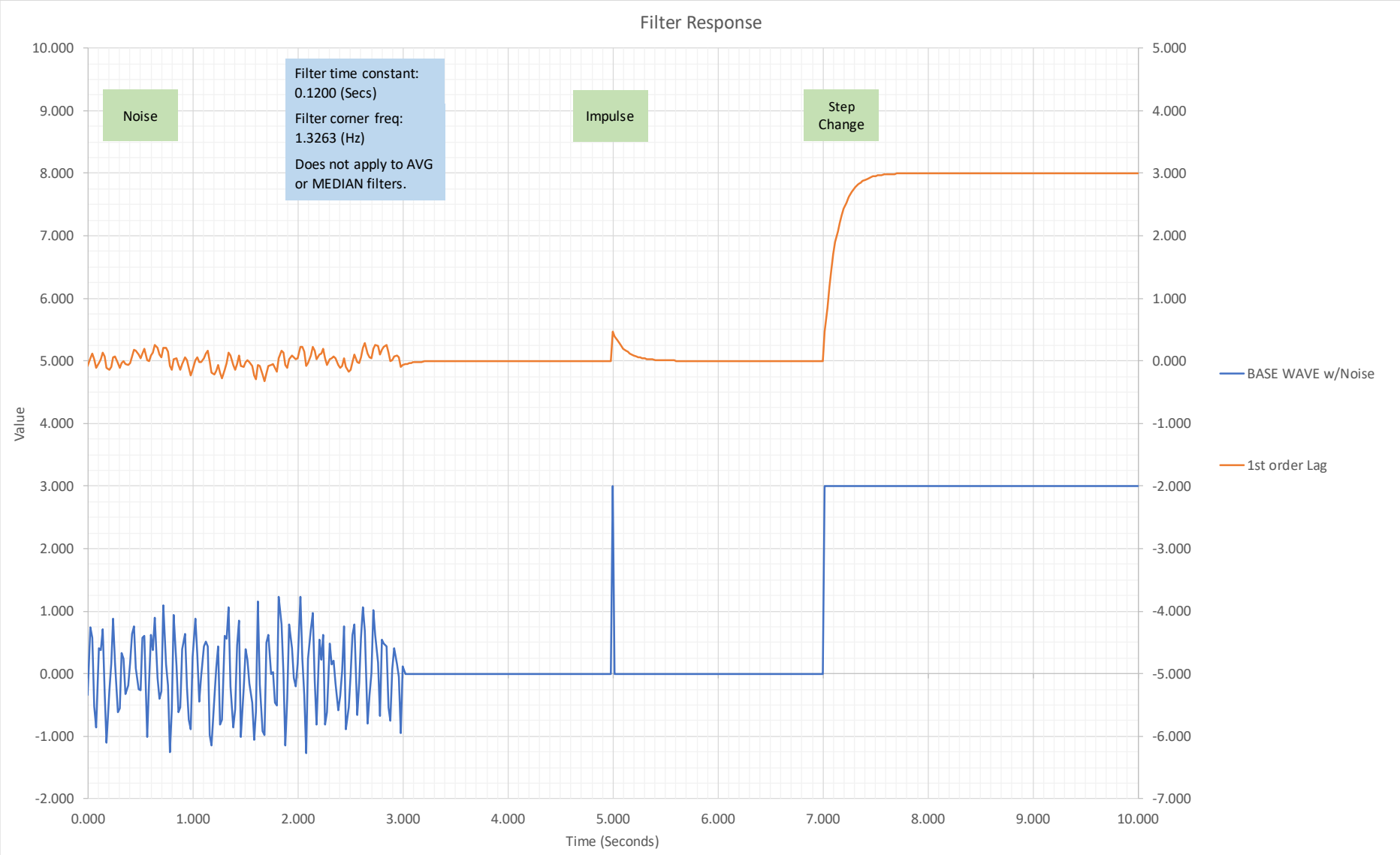
1st Order Lag Filter Response – Tc = 0.03



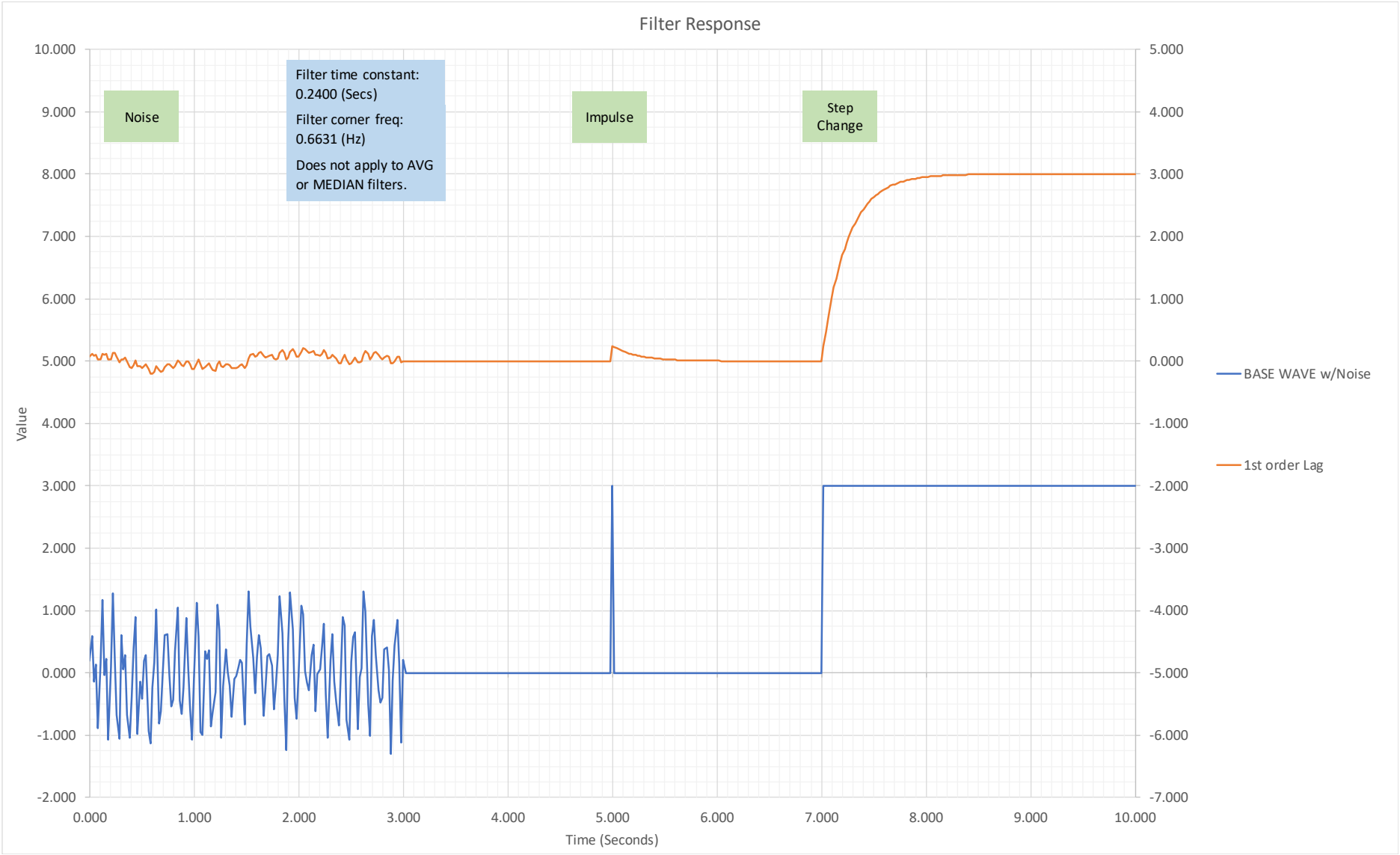
1st Order Lag Filter Response – Tc = 0.06



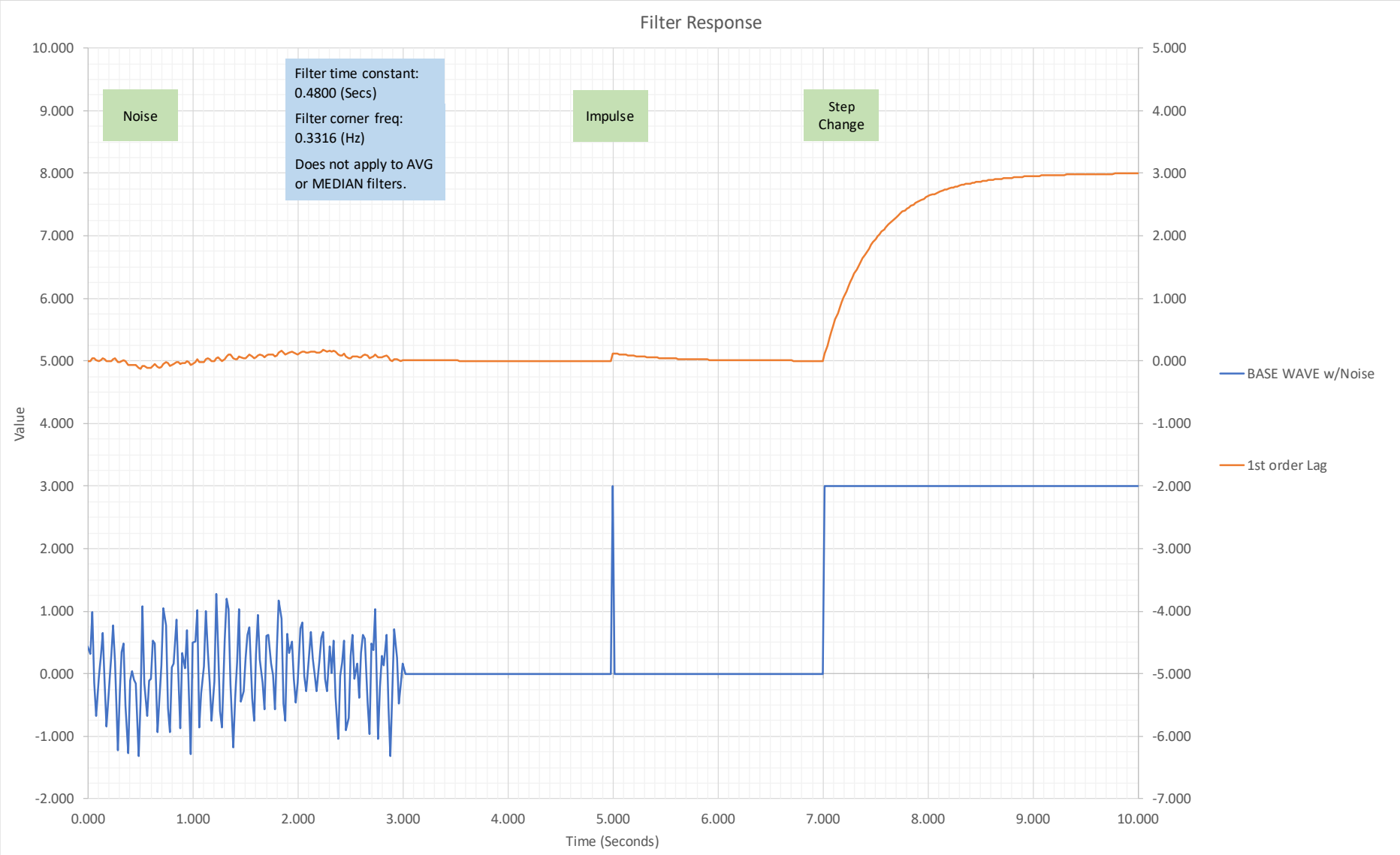
1st Order Lag Filter Response – Tc = 0.12



1st Order Lag Filter Response – Tc = 0.24



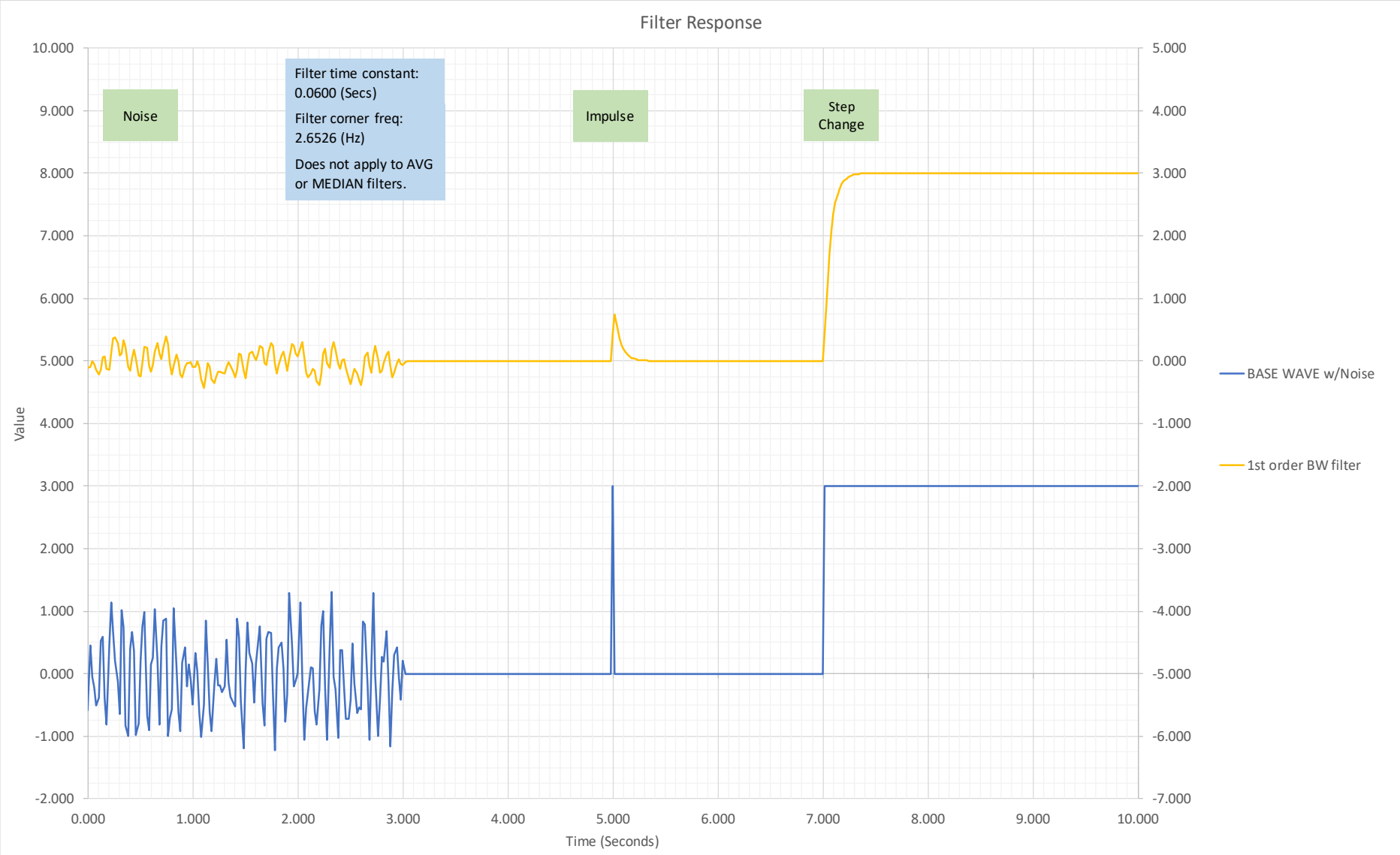
1st Order Lag Filter Response – Tc = 0.48



1st Order Butterworth Filter

- **Response is very similar to 1st order lag filter**
- **Equation (numerical implementation):**
  - Tc (Secs) = Time Constant, specified by user
  - Ts (Secs) = Sampling time, specified or calculated
  - Wd (Hz) = Cutoff Frequency = 1 / (2 * PI * Tc)
  - C = Cotangent(Ts / (Tc * 2))
  - D0 = (1 + C)
  - A0 = 1 / D0
  - A1 = 1 / D0
  - B1 = (1 – C) / D0
  - $Y_n = A0 * X_n + A1 * X_{n-1} - B1 * Y_{n-1}$

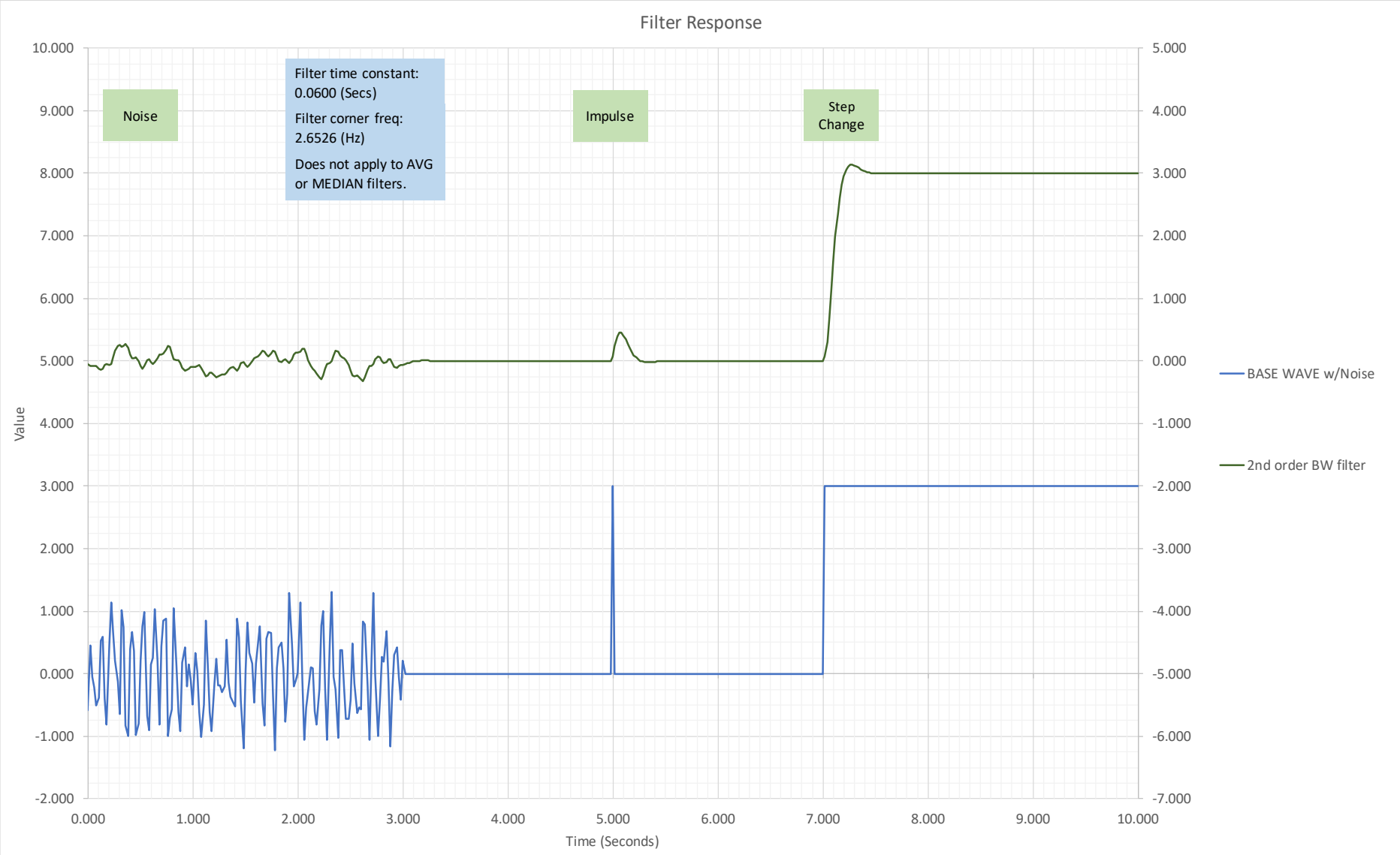
Butterworth 1st Order Response – Tc = 0.06



2nd Order Butterworth Filter

- **Much better filtering**
  - Also a larger phase shift
- **Equation (numerical implementation):**
  - Tc, Ts, Wd, C – same as 1st order equation
  - $D0 = C^2 + 2^{0.5} * C + 1$
  - $A0 = 1 / D0$
  - $A1 = 2 / D0$
  - $A2 = 1 / D0$
  - $B1 = (-2 * C^2 + 2) / D0$
  - $B2 = (C^2 - 2^{0.5} * C + 1) / D0$
  - $Y_n = A0 * X_n + A1 * X_{n-1} + A2 * X_{n-2} - B1 * Y_{n-1} – B2 * Y_{n-2}$

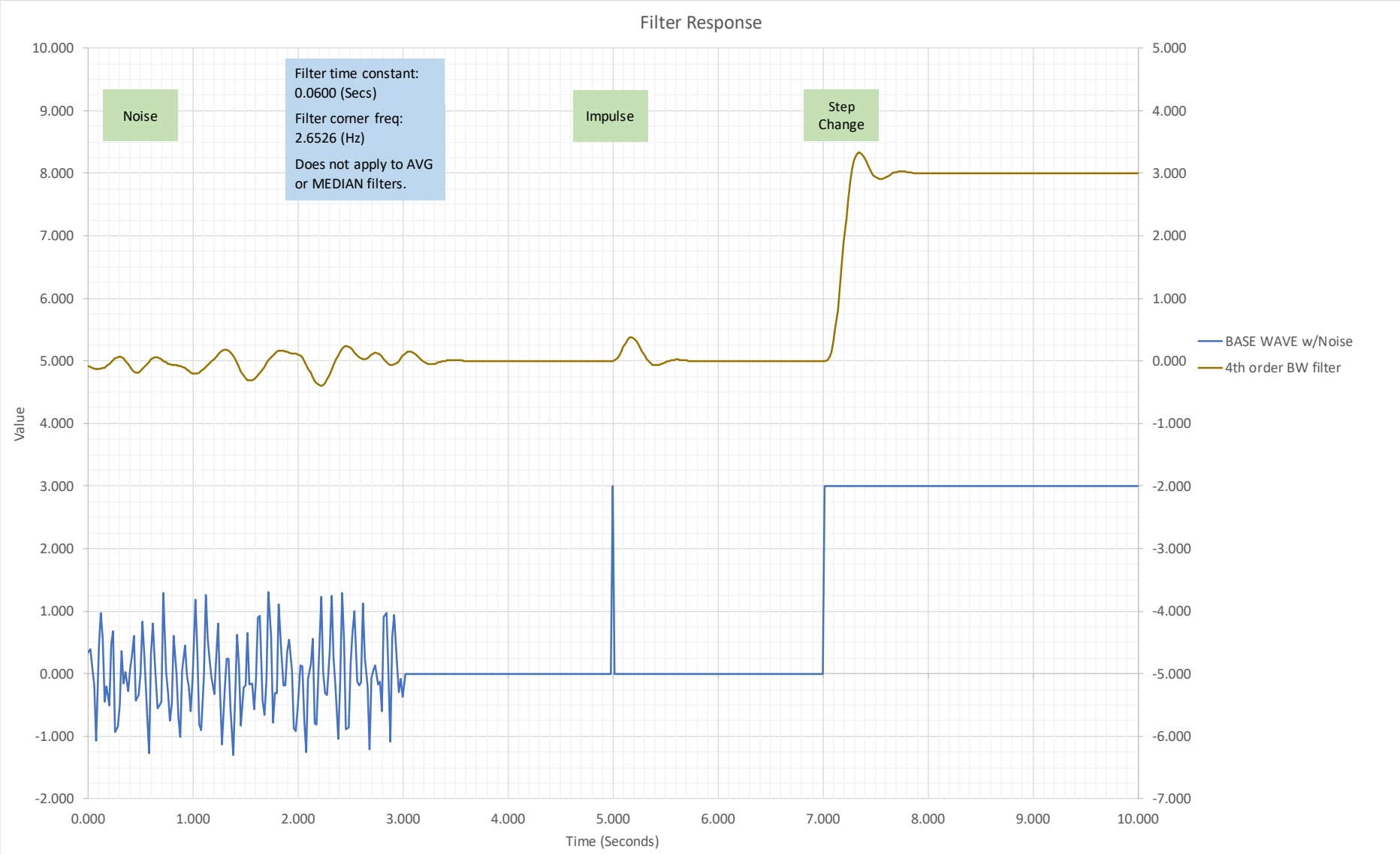
Butterworth 2nd Order Response – Tc = 0.06



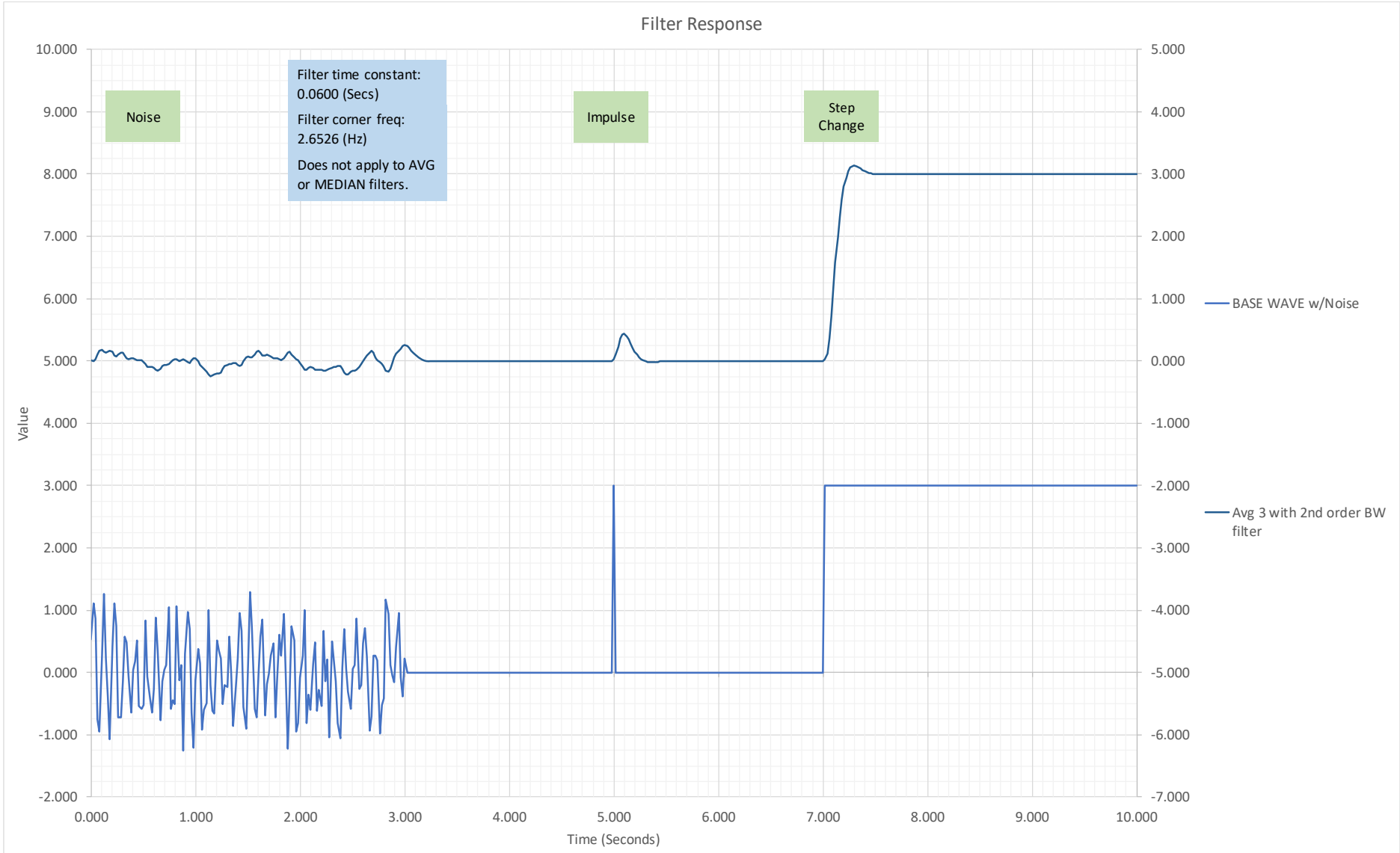
Other Low Pass Filters

- **Higher orders**
- **Cascaded Combinations**
  - Average with 1st order lag, etc.
- **Many other types**
  - Chebyshev
  - Sallen-Key
  - Bessel
  - Optimum “L” (Legendre-Papoulis)
  - Exponentially decaying average
  - Kalman
  - Etc., etc., etc.

Butterworth 4th Order Response – Tc = 0.06

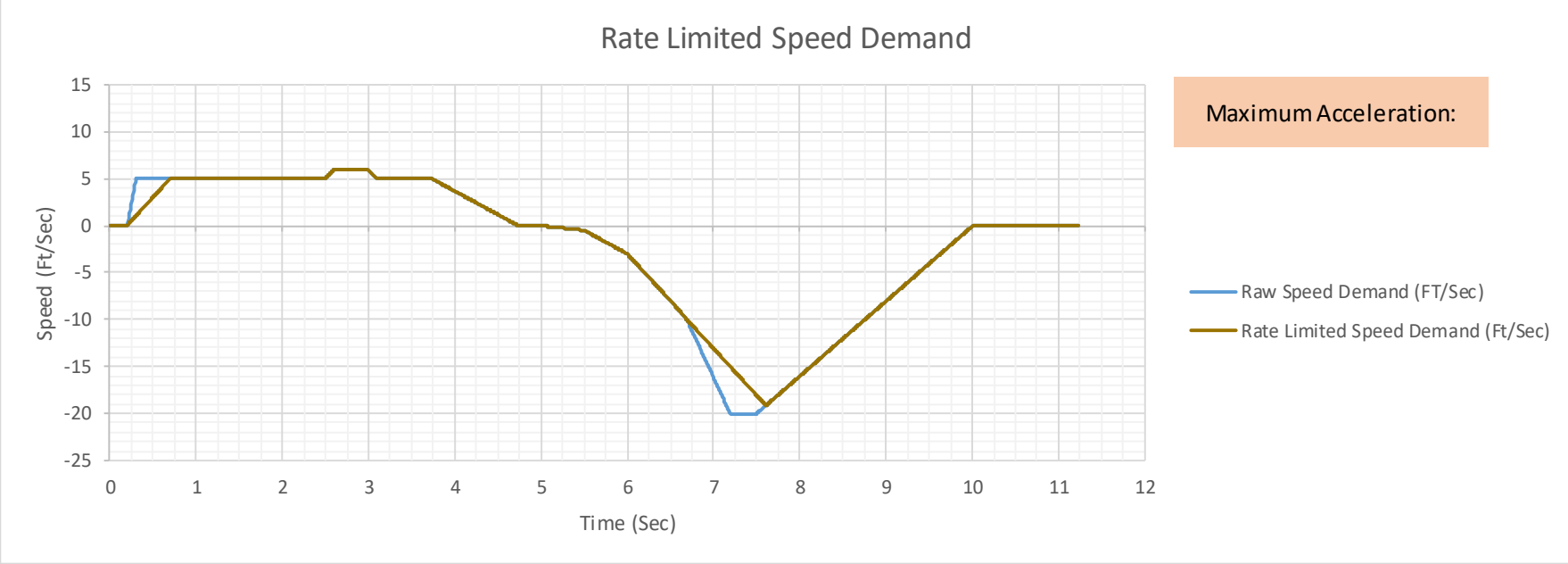


Cascaded 3 Sample Avg and 2nd Order Butterworth



Other Analog Value Processing

- **Rate Limiting**
  - Prevent over anxious drivers from toppling robot !
  - “Clamp” how much the value can change each scan cycle
- **Sample (also see demo app) …**



Other Analog Value Processing

- **High / Low Value Limiting**
  - Clamps value within pre-defined limits
- **High Select or Low Select**
  - Select largest value of multiple inputs
  - Select smallest value of multiple inputs
  - Choose between multiple sensors
- **Function Generator**
  - Y = F(X)
  - F(x) is a piece-wise linear function consisting of pairs of X, Y values, ordered by increasing X.
  - Open loop speed control and other characterizations.
  - This is one of the most used control system algorithms!

Other Analog Value Processing

- **Selecting Between Two Analog Values**
  - Analog Switch
    - Auto / Teleop Demand
    - Auto / Manual Output
  - Incorporate “bumpless” transfer between two values by rate limiting the difference between the two at the time of transfer.
    - Note: Do not rate limit the change of the input…

Exercise 9.1 (1/2)

- **Draw an analog block diagram to process left and right drive wheel desired speed signals from input to motor controller output.**
- **There are two inputs (left and right) and two input sources:**
  - One for Teleop. This is +/- 1. Scale to +/- 15 ft/sec. The output should be zero when the input is within the deadband of +/- 0.08. Apply a filter to scaled input, with a very small time constant.
  - One for autonomous. This one is in FT/Sec. Ensure this signal is always within +/- 15 ft/sec
- **Switch between the two inputs based on a digital signal that indicates "teleop mode"**
- **Rate limit the selected signal so it doesn't change any faster than 35 ft/sec/sec.**

Exercise 9.1 (2/2)

- **The motor controller needs an output of +/-1.**
- **When tested the robot responds as follows on level ground when it isn't carrying any cubes:**

Motor Output	Speed
0.0	0
0.1	0.02
0.2	0.1
0.3	0.25
0.4	0.38
0.5	.52
0.6	.65
0.7	.77
0.8	.85
0.9	.94
1.0	1.0