A Note on Gaussian Steps in openEMS

Ted Yapo

May 20, 2019

1 Motivation

While many examples use openEMS [1] to produce frequency-domain analyses, in testing printed-circuit board structures, we may wish to use time-domain techniques as well. This corresponds to the time-domain-reflectometry (TDR) methods PCB designers may use in the lab. The openEMS code contains an abrupt unit step excitation, accessed through the octave function SetStepExcite(), but the resulting excitation is not bandwidth-limited, and can excite transmission-line modes above those of interest, producing quite confusing results. Instead of this sharp unit step, it's preferable to use a softer step function, specifically a Gaussian step with controllable rise-time (and hence bandwidth). This note presents the derivation of an approximation to the Gaussian step function which can be used with the existing SetCustomExcite() openEMS function. Future versions of openEMS may include such functions built-in, but until then, this approximation may suffice.

2 The Gaussian Step

The Gaussian step function, $G_\sigma(t)$ is the result of passing the unit step signal:

$$f(t) = u(t)$$

through a Gaussian filter with an impulse response of:

$$g(t) = \frac{1}{\sigma\sqrt{\pi}} e^{-\frac{t^2}{\sigma^2}}$$

The resulting convolution can be expressed as:

$$\begin{aligned} G_\sigma(t) &= f(t) * g(t) \\ &= \frac{1}{\sigma\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{\frac{-\tau^2}{\sigma^2}} u(t-\tau)\, d\tau \\ &= \frac{1}{\sigma\sqrt{\pi}} \int_{-\infty}^{t} e^{\frac{-\tau^2}{\sigma^2}}\, d\tau \\ &= \frac{1}{\sigma\sqrt{\pi}} \left[\frac{\sigma\sqrt{\pi}}{2} + \frac{\sigma\sqrt{\pi}}{2} \operatorname{erf}\left(\frac{t}{\sigma}\right) \right] \\ &= \frac{1}{2} + \frac{1}{2} \operatorname{erf}\left(\frac{t}{\sigma}\right) \end{aligned} \tag{1}$$

Where erf() is the commonly-known Gauss error function. A plot of $G_\sigma(t)$ is shown in Figure 1.

2.1 Choosing σ

The rise-time of the Gaussian step is determined by the scaling parameter, σ, but choosing this parameter directly is inconvenient. Instead, we wish to specify a rise time for the signal and calculate the required σ from that. In this case, we choose the $10\% - 90\%$ rise time, although the method is equally applicable to other metrics.

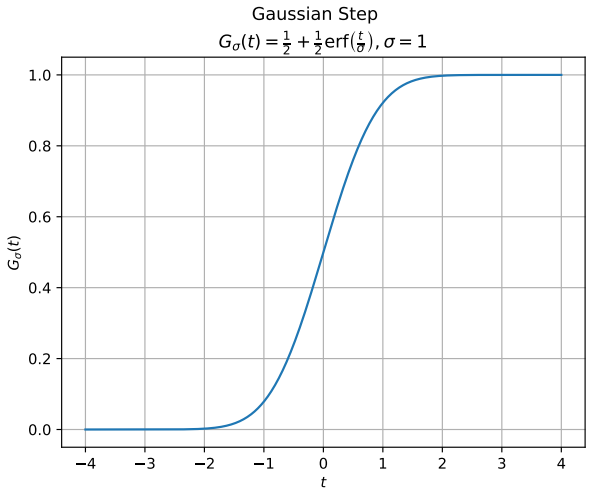

Figure 1: Gaussian Step Function ($\sigma = 1$)

Using the result of (1), we can find the time at which the signal rises to 90% of the final amplitude:

$$\begin{aligned} 0.9 &= \frac{1}{2} + \frac{1}{2}\,\mathrm{erf}\left(\frac{t_{90}}{\sigma}\right) \\ 0.8 &= \mathrm{erf}\left(\frac{t_{90}}{\sigma}\right) \\ \mathrm{erf}^{-1}(0.8) &= \frac{t_{90}}{\sigma} \\ t_{90} &= \sigma\,\mathrm{erf}^{-1}(0.8) \end{aligned}$$

Given the symmetry of $G_\sigma(t)$ around $t = 0$, we can double this to find the rise time of the signal, and solve for σ as a function of t_{rise}:

$$\begin{aligned} t_{rise} &= 2\,t_{90} = 2\sigma\,\mathrm{erf}^{-1}(0.8) \\ \sigma &= \frac{t_{rise}}{2\,\mathrm{erf}^{-1}(0.8)} \end{aligned} \tag{2}$$

3 Time Shifting

While the abrupt unit step function, $u(t)$ is zero for all time $t < 0$, this is not the case for the Gaussian step. Accordingly, we typically want to shift the step so that the rise is centered on some $t_c > 0$. This is easily accomplished by subtracting the offset from the time parameter:

$$G'_\sigma(t) = \frac{1}{2} + \frac{1}{2}\,\mathrm{erf}\left(\frac{t - t_c}{\sigma}\right)$$

This shift does not affect the rise time or frequency characteristics of the step.

4 Approximating $\mathrm{erf}(x)$

The openEMS SetCustomExcite() function uses an expression parser in the underlying C++ code to evaluate the user-supplied string as a function of time. Unfortunately, the parsing function does not know about the error function $\mathrm{erf}(x)$, so an approximation must be made. Luckily, Abromowitz and Stegun [2] have tabulated a number of good approximations to $\mathrm{erf}(x)$. In this case, we choose the following approximation (A&S section 7.1.25):

$$\mathrm{erf}(x) \approx 1 - (a_1 t + a_2 t^2 + a_3 t^3)e^{-x^2}$$

where

$$t = \frac{1}{1 + px},$$

$$p = 0.47047, a_1 = 0.3480242,$$

$$a_2 = -0.0958798, a_3 = 0.7478556$$

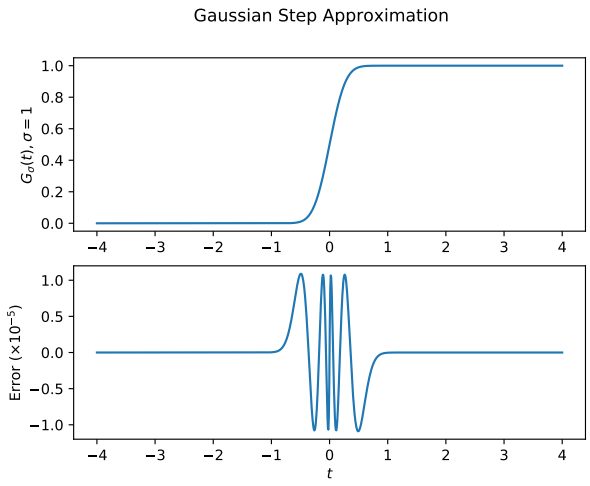

Figure 2: Gaussian Step Approximation ($\sigma = 1$), and corresponding error

A&S report this approximation has an error of:

$$|\epsilon(x)| \leq 2.5 \times 10^{-5}$$

but due to the $\frac{1}{2}$ scaling of $\mathrm{erf}(t)$, this error is reduced by half in the apprximation. The resulting Gaussian Step approximation and the computed error are shown in Figure 2.

5 Nyquist Sampling Rate

The openEMS SetCustomExcite() function takes a Nyquist frequency parameter which is used to determine how often the output of the model is sampled to avoid aliasing issues. While an exact formulation would consider the frequency content of (1), it is sufficient to consider just the response of the Gaussian filter. Given the filter impulse response:

$$g(t) = \frac{1}{\sigma\sqrt{\pi}} e^{\frac{-t^2}{\sigma^2}}$$

the frequency response is:

$$|G(f)| = \sigma\sqrt{\pi} e^{-\pi^2\sigma^2 f^2}$$

(the Fourier transform of a Gaussian is itself a Gaussian). Using this, we can solve for the frequency at which the response is down by A dB – note that the leading coefficient, $\sigma\sqrt{\pi}$, drops out in the ratio:

$$\begin{aligned} 10^{\frac{-A}{20}} &= e^{-\pi^2\sigma^2 f^2} \\ \frac{A}{20}\ln 10 &= \pi^2\sigma^2 f^2 \\ f &= \sqrt{\frac{\frac{A}{20}\ln 10}{\pi^2\sigma^2}} \end{aligned}$$

The Nyquist frequency is twice this, or:

$$f_{\text{Nyquist}} = 2\sqrt{\frac{\frac{A}{20}\ln 10}{\pi^2\sigma^2}} \tag{3}$$

6 GNU Octave Function

The GNU Octave function shown in Figure 3 creates the necessary string to pass to SetCustomExcite() and calculates the Nyquist frequency for a given cutoff level in dB.

7 References

References

[1] openEMS code GitHub repository `https://github.com/thliebig/openEMS`

```
function [string, f_nyquist] = GaussianStep(rise_time, center_time, dB_cutoff)
  C = center_time;
  sigma = rise_time / (2*erfinv(0.8));
  K = 1/sigma;
  f_nyquist = 2*sqrt(dB_cutoff/20 * log(10) / (pi*pi*sigma*sigma));
  %
  % Gaussian step using erf() approximation 7.1.25 from Abramowitz and Stegun
  % http://people.math.sfu.ca/~cbm/aands/page_299.htm
  %
  string = ['0.5+0.5*(', num2str(K), '*(t-', num2str(C), ')>=0)*(1 - exp(-', num2str(K), ...
            '*(t-', num2str(C), ')*', num2str(K), '*(t-', num2str(C), '))*(0.3480242/(1+0.47047*', ...
            num2str(K), '*(t-', num2str(C), ')) - 0.0958798/((1+0.47047*', num2str(K), '*(t-', ...
            num2str(C), '))*(1+0.47047*', num2str(K), '*(t-', num2str(C), ...
            '))) + 0.7478556/((1+0.47047*', num2str(K), '*(t-', num2str(C), '))*(1+0.47047*', ...
            num2str(K), '*(t-', num2str(C), '))*(1+0.47047*', num2str(K), '*(t-', num2str(C), ...
            ')))))-0.5*(', num2str(K), '*(t-', num2str(C), ')<0)*(1 - exp(-', num2str(K), '*(t-', ...
            num2str(C), ')*', num2str(K), '*(t-', num2str(C), '))*(0.3480242/(1-0.47047*', ...
            num2str(K), '*(t-', num2str(C), ')) - 0.0958798/((1-0.47047*', num2str(K), '*(t-', ...
            num2str(C), '))*(1-0.47047*', num2str(K), '*(t-', num2str(C), ...
            '))) + 0.7478556/((1-0.47047*', num2str(K), '*(t-', num2str(C), '))*(1-0.47047*', ...
            num2str(K), '*(t-', num2str(C), '))*(1-0.47047*', num2str(K), '*(t-', num2str(C), ')))))'];
end
```

Figure 3: GaussianStep() GNU Octave Function

[2] Abramowitz and Stegun, Handbook of Mathematical Functions, Section 7.1.25, http://people.math.sfu.ca/~cbm/aands/page_299.htm

8 License

8.1 Document License

8.2 Code License

THE USE OR OTHER DEALINGS IN THE
SOFTWARE.

9 Revsion History

Date	Version	Comment
May 19, 2019	1.0	First Version
May 20, 2019	1.1	moved constants between f(t) and g(t)