ISBN: 978-3-95450-241-7 ISSN: 2673-5350 doi:10.18429/JACoW-IBIC2022-MOP43

WEB-BASED APPLICATION FOR CABLE SIMULATION MODELS*

M. C. Paniccia†, S. Clark, D. M. Gassner, R. Hulsart, P. Thieberger
Brookhaven National Laboratory, Upton, USA

Abstract

Signal attenuation in a coaxial cable increases with cable length, and the amount of attenuation and signal distortion is dependent on the signal's spectral frequency content. A model of these variations can help predict a signal's expected loss and distortion. This paper describes a free web-based application developed to provide accurate SPICE models for various coaxial cable types. The user can specify a length, select between different cable types, or upload a cable's attenuation curve, and receive a SPICE model for that cable. These simulation models have been used to assist the design and development of new instrumentation systems for the future Electron-Ion Collider (EIC).

INTRODUCTION

The new Electron-Ion Collider (EIC) will include over a thousand new beam position monitors (BPMs) and many other instrumentation systems spread throughout the complex. Signals generated at the various sensors must traverse cable lengths as great as 250 m. Understanding the signal loss and distortion as they propagate over these distances is a key aspect of the system design process. To assist in the design process, analog circuits are modeled with the Simulation Program with Integrated Circuit Emphasis (SPICE) [1]. The SPICE simulation provides a transient and steady-state response of the system allowing for the first-pass representation of expected performance. Modeling cables in SPICE poses a unique challenge as cable performance changes over distance and frequency content. Since most signals from instrumentation systems have a wide frequency content, it is imperative to accurately capture the frequency-dependent performance of the cabling.

A procedure for developing a SPICE simulation model for frequency-dependent cable loss is described in [2]. The frequency-dependent attenuation performance is simulated by creating a multistage passive filter with poles and zeros mapped along the cable's transfer function. A custom Python script was developed to implement the pole-zero approximation method and several simulation models were generated for different coaxial cables. To test the validity of the models, spare cable assemblies were tested in the lab with narrow pulses, and the SPICE model performance matched closely to the measured data. The Python scripts were integrated into a web application to make the cable simulation process accessible to the entire EIC team and the broader instrumentation community.

* Work supported by the US Department of Energy under contract No. DE-AC02-98CH10886.
† mpaniccia@bnl.gov

SPICE MODEL

The creation of the SPICE model starts with defining the cable's frequency-dependent attenuation characteristic curve. The characteristic curve acts like a filter's transfer function and defines the cable's response in the frequency domain. Cable manufacturers typically provide attenuation values at specific frequencies at a specified length; most often 100 ft or 100 meters. If not provided by the manufacturer, the characteristic curve can be measured using a network analyzer. Once established, the characteristic curve can be defined as an equation that can provide attenuation values for any given frequency. This equation takes the form of Eq. (1).

$$Atten = a_0\sqrt{MHz} + a_1\,MHz + a_2. \qquad (1)$$

In generating the SPICE model, an equation for each cable type is determined using the least squares approximation of Eq. (1) to the attenuation versus frequency values provided by the manufacturer. The cable's characteristic curve is then scaled by the distance at which it was measured, assuming attenuation is proportional over distance and frequency. A pair of spare cables of different lengths were measured to verify this assumption using a network analyzer. The measurements were compared to the estimated attenuation curve using the proportional approximation and are shown in Fig. 1. For both cables, the proportional approximation proved to be conservative.

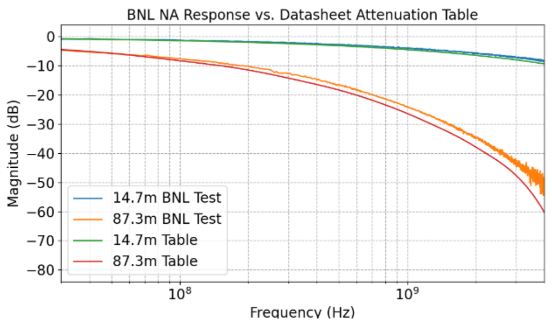


Figure 1: Comparison of LMR240 frequency-dependent attenuation characteristic curve and measured BNL network analyzer data.

The next step in the derivation of the SPICE model is to use the pole-zero approximation process described in [2]. This process recreates the characteristic curve for frequency-dependent attenuation by creating a circuit with a transfer function with a similar slope. The transfer function is created from multiple poles and zeros that are set along the characteristic curve. The poles and zeros are generated using resistors and capacitors configured in a low-pass filter. Multiple stages are connected using the SPICE Voltage

Controlled Source to prevent loading. The multistage poles-zeros transfer function is calculated using a non-linear least squares regression fit provided by the Python SciPy.Optimize library [3]. The results from the fit are translated into resistor and capacitor values for each filter stage. The number of filter stages can vary based on accuracy and runtime performance. In the web application, the stages are set to 6 poles and 5 zeros, and the circuit is shown in Fig. 2.

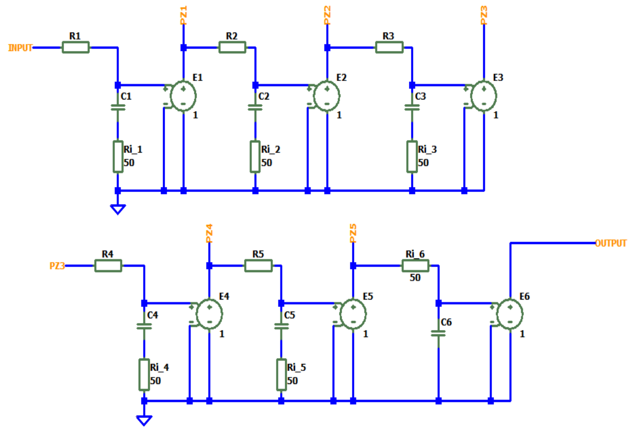


Figure 2: Graphical representation of the SPICE subcircuit. This subcircuit has 6 poles and 5 zeros and uses a 50 Ω impedance.

After the resistor values are calculated from the fit, the values are applied directly to the subcircuit SPICE library file. To do this, a generic library with the predefined netlist for the circuit is stored on the web page server and read into the application as a string. Using the Python string.format method, the resistors, and capacitors are assigned to each stage and the updated string is written to a new library file.

TEST AND VALIDATION

To verify the validity of the simulation models, cable assemblies were tested at various lengths with narrow pulses. The cable assemblies included two LMR240 50 Ω coaxial cables, and two several-meter custom Tefzel 50 Ω semi-rigid coaxial cables. The length of each cable assembly was measured with a Mohr CT100 Time-Domain Reflectometer (TDR). An arbitrary pulse generator was configured to provide unipolar pulses with pulse widths between 200 ps to 10 ns. Each pulse was measured directly on the oscilloscope and was exported into a CSV file. Then each cable assembly was connected to the pulse generator and the responses were again measured on an oscilloscope and exported into a CSV file.

SPICE models were developed for each cable type at the measured lengths. The captured input pulses that were exported from the oscilloscope were loaded into the SPICE simulation file as a piece-wise linear voltage source and connected to the cable SPICE models. The results from the SPICE simulation for both types of cables matched closely to the cable output pulses measured on the oscilloscope, examples are shown in Figs. 3 and 4.

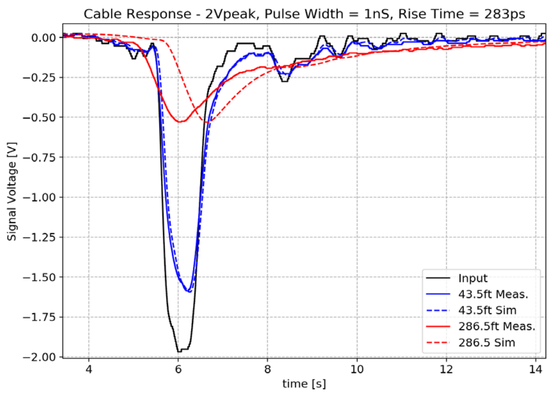


Figure 3: LMR240 50 Ω coaxial cables at 14 and 83 m. The input pulse width is 1 ns FWHM and a rise time of 283 ps.

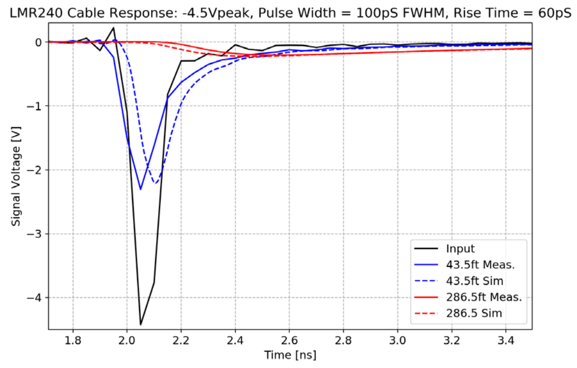


Figure 4: LMR240 50 Ω coaxial cables at 14 and 83 m. The input pulse width is 100 ps FWHM and a rise time of 60 ps.

WEB APPLICATION

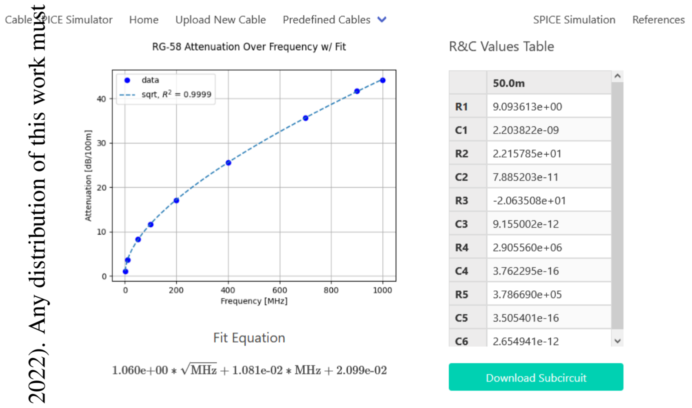
The web application acts as an online calculator for generating the SPICE subcircuits for user-specified cables and lengths. Within the application, the user can select SPICE libraries for multiple cable types over a range of common lengths or generate a custom subcircuit model for a specific cable length or upload a new cable profile. The SPICE subcircuits can be added directly to any SPICE simulation system. When selecting a predefined library, an assembly file is included for use with LTSPICE. The subcircuit consists of pole-zero low-pass filter elements and an ideal transmission line that adds a delay based on the cable's velocity of propagation. New uploaded cables use the default velocity of propagation value of 85% unless otherwise specified. The application utilizes HTML and the Python Flask library to build the web page. Flask is a lightweight framework that was easily integrated with the Python scripts used to create the SPICE models.

Predefined Page

The predefined page provides users with libraries for cables previously characterized. The page includes a table with the frequency-dependent attenuation values for several common lengths along with the characteristic curve at 100 m. The corresponding resistor and capacitor values required to recreate the pole-zero cable model are also included in a separate table. A custom library of SPICE models for several common lengths and an LTSPICE assembly file are available for download at the bottom of the page. The user can create a custom SPICE subcircuit by specifying the desired length in the length text field. After hitting the Calculate button a new SPICE library file will become available for download.

Upload Page

On the Upload Page, the user can add new cables and generate custom SPICE models for them. New cable profiles are uploaded as either a CSV, TXT, or XLS file containing the frequency-dependent attenuation values or manually added directly to a table. A characterization curve is generated for the uploaded data along with the representative function and a SPICE model for download. When uploading the profile, the user can specify the cable impedance and velocity; if left blank they will default to 50 Ω and 85%, respectively. An example of the output from the Upload Page is shown in Fig. 5.



R&C Values Table

	50.0m
R1	9.093613e+00
C1	2.203822e-09
R2	2.215785e+01
C2	7.885203e-11
R3	-2.063508e+01
C3	9.155002e-12
R4	2.905560e+06
C4	3.762295e-16
R5	3.786690e+05
C5	3.505401e-16
C6	2.654941e-12

Fit Equation

$$1.060\text{e}{+}00 * \sqrt{\text{MHz}} + 1.081\text{e}{-}02 * \text{MHz} + 2.099\text{e}{-}02$$

Figure 5: Output from the Upload New Cable section of the Cable SPICE Simulator web application. The SPICE subcircuit modeled is for a 50-meter RG-58 cable.

USE IN EIC BPM DESIGN

Determining the distribution and location of the EIC Beam Position Monitor (BPMs) electronics has been an important focus of the instrumentation preliminary design. In the presently proposed design, none of the BPM electronics will be located inside the tunnel, to reduce potential radiation related problems. Instead, the plan is to utilize external service buildings spread around the ring. As a result, the cable lengths can vary from 100-240 m long. At these relatively large cable lengths, the variation in the level of signal degradation between cables can have a significant impact on the system performance and cost.

To study the performance of the various cable types, BPM responses are generated using CST Studio. The BPM responses are then imported into a SPICE program as a piecewise linear voltage source and connected to the cable model under test. A comparison between several 50 Ω cable types at the maximum distance of 240 m is shown in Fig. 6.

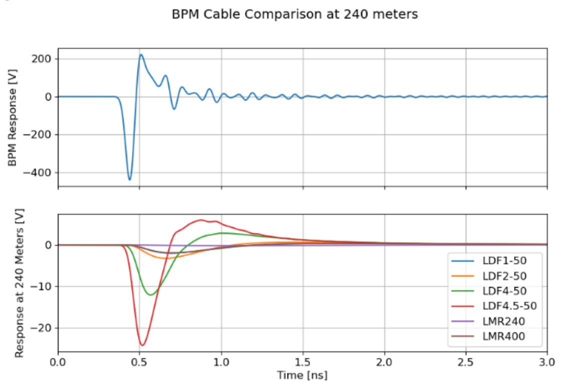


Figure 6: Response to BPM signal across various 50 Ω cables. All cables use SPICE models developed with the procedure discussed herein.

CONCLUSION

Accurate modeling of coaxial cables in SPICE presents a unique challenge. Using the models created with pole/zero approximation, designers can easily test the performance of complex signals over long-distance cables and receive reasonable results.

The models presented in this paper and the ones available on the web application are not a complete representation of the cable performance. These models are not bidirectional and will not model the performance of impedance mismatches nor do they include variations in temperature and humidity. However, the SPICE models derived using the pole-zero approximation provide an accurate simulation of the frequency-dependent attenuation and can assist in understanding the signal loss in complex signals.

REFERENCES

[1] L. W. Nagel and D. O. Pederson, “SPICE (Simulation Program with Integrated Circuit Emphasis)”, University of California, Berkeley EECS Department, UCB/ERL M382, Apr. 1973.

[2] B. Hyland., “An Improved and Simple Cable Simulation Model”, *EDN*, Jul. 2012, `https://www.edn.com/improve-and-simplify-cable-simulation`

[3] P. Virtanen, R. Gommers, *et al.*, “SciPy 1.0: fundamental algorithms for scientific computing in Python”, *Nat. Methods*, vol. 17, pp. 261-272, 2020. doi:10.1038/s41592-019-0686-2