ICA 2013 Montreal
Montreal, Canada
2 - 7 June 2013

Engineering Acoustics
Session 2pEAa: Computational Methods in Transducer Design, Modeling, Simulation, and Optimization I

2pEAa7. Applications of network synthesis and zero-pole analysis in transducer modeling
Daniel M. Warren*

*Corresponding author's address: Knowles Electronics, 1151 Maplewood Dr, Itasca, IL 60143, daniel.warren@knowles.com

When a transducer is an immutable component of a larger system being simulated, it is sufficient that the transducer model correctly reproduce behavior only at the available ports of the transducer. The behavior of two-port electroacoustic transducer should be completely characterized by three transfer functions related to the electrical and acoustic termination impedance and the transfer impedance. To the extent that the transducer could be represented as a analog circuit of passive linear elements, the same circuit could be exactly represented by rational polynomials of the Laplace s, embodied as a set of poles and zeros of the transfer functions. This invites the reverse process of identifying poles and zeros by nonlinear curve fit of rational polynomials to measured transfer data, perhaps even synthesizing a circuit directly from the identified poles and zeros. Measured transducer transfer data have been fit demonstrating both the utility and the pitfalls of this method. Curve fit transfer functions can be a compact and faithful representation of complex data over frequency, but have no predictive value outside the given data. Judicious selection of the number of poles and zeros, initial values, proper constraints, and some physical insight are necessary for stable curve fits.

Published by the Acoustical Society of America through the American Institute of Physics
INTRODUCTION

Finite networks of linear, lumped parameter elements have long been used as a means of analyzing electroacoustic transducers [1]. To the extent that networks are sufficient representations of transducers, techniques used to analyze and synthesize networks can also be applied to transducers. Traditionally, white box techniques are used to synthesize transducer networks, wherein equations governing physical mechanisms within the transducer are derived from first principles and a network embodying the same equations is constructed. Other methods in use in control theory only observe the behavior of the system and attempt to derive equations describing this behavior through a process called system identification. In a black box approach, no prior knowledge of the system is assumed, and a general function with many free parameters is fit to the measured behavior [2]. In a gray box approach, prior knowledge is employed to varying degrees, from applying constraints to certain parameters to starting with a model that is known to represent the behavior of the system, before fitting the model to measured data [3].

The current project is to explore how far a gray box system identification can be pushed toward synthesizing a network model of a transducer. Is it possible to synthesize a network from nothing but behavioral data? Is it possible to find a matrix of functions which model the transducer within a larger system simulation, from nothing but behavioral data? Short of either of these, is it possible to start with a good network structure and derive the matrix of equations, eventually working toward populating the network component values in the white box network?

METHOD

Functions that define the input and output characteristics of a network will be generally referred to as network functions $H(s)$, where $s = j \omega$ is the complex frequency in continuous time. Network functions are ratios of output to input measured across exposed ports of the network. Network functions can be expressed as ratios of polynomials

$$H(s) = \frac{p(s)}{q(s)} = \sum_{k=0}^{n} \frac{a_k s^{n-k}}{b_m s^{m-\ell}}, \quad b_m = 1, \quad (1)$$

with real coefficients $a_k$ and $b_\ell$ [4]. The coefficient $b_m = 1$ to avoid ambiguity among the magnitudes of the coefficients. In a black box approach, one would choose arbitrary orders $n$ and $m$ and allow a least-squares curve fitting routine search for the coefficients.

An alternative formulation is found by factoring the polynomials

$$\frac{p(s)}{q(s)} = g \prod_{k=1}^{n} \frac{(s-z_k)}{(s-p_\ell)} = g \prod_{\ell=1}^{m} \frac{(s-z_{k})}{(p-p_{\ell})} = g \frac{(s-z_{k})}{(p-p_{\ell})} (s-z_k^2 + z_k^2) \ldots$$

$$= \frac{a_0}{b_0} \quad (2)$$

The roots $z_k$ of $p(s) = 0$ are the zeros of function and the roots $p_\ell$ of $q(s) = 0$ are the poles. Since the coefficients in Eq.(1) are real, the zeros must either be real valued $z_k = z_k^*$ or occur in conjugate pairs $z_k^* = z_k^* \pm jz_k^*$, as do the poles. The product of the conjugate pair factors are written in the form $((s-z_{k^*})^2 + z_{k^*}^2)$, which is a convenient form for calculation as it eliminates small spurious imaginary factors resulting from the numerical product of $(s-z_{k})(s-z_{k^*}^*)$.

In general, network functions are complex in value, and it is best to work with complex data when available. Many curve fitting routines are available to fit real-valued functions to real
data, and these could be used to fit the magnitude of Eq.(1) to the magnitude of the data. The quality and robustness of the fit can be improved by fitting to the complex data values, but the techniques for fitting to complex data are less well known. In Mathematica, the FindFit function can perform complex curve fitting when given the correct NormFunction option, as

\[
\text{FindFit}[\text{data}, \text{expr}, \text{pars}, \text{vars}, \text{NormFunction} \rightarrow (\text{Abs}\@\text{Norm}[#] \&)].
\]

A practical consideration when fitting curves is scaling the data. If the free parameters in a curve fit have widely disparate orders of magnitude, then the gradient matrix is ill-conditioned and the smaller parameters may get lost in the round-off. At the very least, the smaller parameters are not determined to the same precision as the larger parameters, if they are found at all.

First, we scale the frequency of the data. Choose the maximum frequency of interest \(f_{\text{max}}\). It is advisable that this frequency be greater than the highest frequency in the data. In fact, if the eventual goal is to create a discrete-time filter to represent the data, the Nyquist Theorm requires that \(f_{\text{max}}\) be at least twice the highest frequency in the data. Next, choose a value \(v_{\text{mid}}\) somewhere in the middle of the range. Normalize each data set as

\[
\{[\Omega_i, \tilde{v}_i], \ldots\} = \left\{\left\{\frac{\pi f_i}{f_{\text{max}}}, \frac{v_i}{|v_{\text{mid}}|}\right\}, \ldots\right\},
\]

where \(0 \leq \Omega \leq \pi\) is the normalized circular frequency.

Although the network function is complex valued, it is required that the free parameters and function independent variable are real. Complex parameters are broken into their real and imaginary parts as two separate parameters as in Eq.(2). Eqs.(1, 2) are functions of the complex frequency \(s\), so for the curve fit we set \(s = j\Omega\), and fit the function to the normalized data. Finally, recover the original scale by mapping \(\Omega\) back to the complex frequency \(s\) and multiplying by the magnitude of the original data set

\[
|v_{\text{mid}}| H(j\Omega) = \frac{s}{2f_{\text{max}}},
\]

Test Case

To demonstrate curve fitting techniques, we examine the network in Figure 1. We chose a network of two high-pass filters cascaded with two resonators with the characteristics shown in Table 1. The voltage gain across the network is determined by standard network analysis techniques [5] to have the form given by

\[
\frac{V_2}{V_1} = g \frac{s^2}{(s-p_{1,r})(s-p_{2,r})(s-p_{3,r})(s-p_{4,r})^2 + p_{3,j}^2 + p_{3,j}^2},
\]

with the poles and zeros tabulated in Table 1. We derive a set of test data by sampling the voltage gain function at 50 Hz intervals from 50 Hz to 10 kHz. The intent is to have some features of the gain function just outside the band of the sampled data, with one high-pass filter corner frequency on the low end and one resonance peak on the high end. A small, random, complex value is added to each datum to simulate the noise floor of the measurement. Since the random values are added to frequency domain data, it is not a true representation of time-domain random noise, but suffices for a curve-fitting demonstration.

Black Box

Eq.(1) with \(n = m = 8\) is fit to the generated noisy data using the described method. No a priori knowledge is applied about the system except that the gain must be 0 at DC, which is
FIGURE 1: A network to demonstrate system identification techniques. Element values are given in Table (1).

TABLE 1: Numerical values characterizing the test case network in Figure 1.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Network</th>
<th>Poles and Zeros</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Pass 1</td>
<td>$C_1 = 400 \text{nF}$</td>
<td>$g = 1.37 \times 10^{18}$</td>
</tr>
<tr>
<td>$f_c \approx 40 \text{Hz}$</td>
<td>$R_1 = 10 \text{k}\Omega$</td>
<td>$z_1 = 0$</td>
</tr>
<tr>
<td>High Pass 2</td>
<td>$C_2 = 2 \mu\text{F}$</td>
<td>$z_2 = 0$</td>
</tr>
<tr>
<td>$f_c \approx 800 \text{Hz}$</td>
<td>$R_2 = 100 \text{\Omega}$</td>
<td>$p_1 = -41.4$</td>
</tr>
<tr>
<td>Peak 1</td>
<td>$L_3 = 21.6 \text{H}$</td>
<td></td>
</tr>
<tr>
<td>$f_0 \approx 3 \text{kHz}$</td>
<td>$R_3 = 10 \text{k}\Omega$</td>
<td>$p_3 = -3175. \pm j12293.$</td>
</tr>
<tr>
<td>$Q \approx 3$</td>
<td>$C_3 = 1.8 \text{nF}$</td>
<td>$p_4 = -3919. \pm j92100.$</td>
</tr>
<tr>
<td>Peak 2</td>
<td>$L_4 = 127 \text{mH}$</td>
<td></td>
</tr>
<tr>
<td>$f_0 \approx 10 \text{kHz}$</td>
<td>$R_4 = 1 \text{k}\Omega$</td>
<td></td>
</tr>
<tr>
<td>$Q \approx 8$</td>
<td>$C_4 = 2 \text{nF}$</td>
<td></td>
</tr>
</tbody>
</table>

enforced by setting $a_n = 0$. Bode plots of two trials are shown in Figure 2. Neither trial captured the out-of-band high-pass or resonance features, but the resulting curves are a very good fit to the data on hand. The curves also tend to diverge sharply outside the frequency band of the data, indicating that they have no extrapolative value.

FIGURE 2: Bode plot of the voltage gain of the test case network with two trials of the black box curve fit.

Figure 3 shows some of the problems of the black box approach. There is no consistency between the two trials; the eye sees only a random spray of poles and zeros across the complex plane. More troubling are the poles and zeros in the right half-plane. The presence of poles in the right half-plane indicates that the function is unstable and would explode in the time domain. The presence of either poles or zeros in the right half-plane indicates that the function is not minimum phase [6]. These are important, and problematic, properties that the original network does not possess, so it is clear that one can draw incorrect conclusions about the basic nature of a system from a black box system identification.

This technique still has its uses. It is fast and easy to set up, and the resulting function is a faithful representation of the data. The result is a simple, compact representation of complex data over a wide frequency range, but this function cannot be extrapolated outside the domain of the data, and no insight into the underlying network can be surmised. While the coefficients of the resulting function cannot be directly applied, for example, to derive a compensation filter,
the function can represent the data in a simulation of a system in which compensation filter coefficients are to be optimized, as long as the system is limited to the bandwidth of the original data.

**Gray Box**

It is evident that the black box procedure fails for two main reasons: the curve fit has too many degrees of freedom and is not properly constrained. Eq.(5) has only 7 unknown parameters but the curve fit is working with 16. The curve fit routine is trying to get rid of those additional parameters. In Figure 3, you can see attempts to cancel zeros and poles by placing them on top of each other. Extra zeros and poles are being distributed anywhere else on the complex plane that will not perturb the function fit within the data band, without concern that the resulting function be stable or minimum-phase.

In order to have more control over the fitting process, a software tool was developed in *Mathematica*, shown in Figure 4. The tool allows the user to interactively place poles and zeros in the third quadrant of the s-plane. Poles or zeros placed on the origin are locked to the origin, and those placed on the negative real axis are locked to, but free to vary along, the real axis. Poles and zeros placed elsewhere represent complex conjugate pairs and are free to vary within the constrained plane

$$\{ s : \Re(s) < 0 \land 0 < \Im(s) \leq \pi \}.$$  

The tool dynamically creates a function of the form Eq.(2) and displays its Bode plot with the normalized data. Once the user is satisfied that the function has the right structure and reasonable initial conditions, he can push a button and the curve fit finds finishes the job.

This tool automatically constrains the function to be stable and minimum-phase, which are expected properties of a transducer network, and constraints chosen by the user are adhered to during the curve fit. It also allows the user to apply knowledge of the problem domain to system identification. At one level of the gray box, the user has limited knowledge of the problem domain, but he can visually analyze the data and make guesses as to the structure of the function. The test case data (Figure 5) visibly has a 6 dB/octave rising slope and a peak, indicating a high pass (1 zero at the origin, 1 real pole) and a peak (1 complex conjugate pole pair).

The fit is fairly good, although the given function could not fit the slight rise of data at the high end of the range due to the out-of-band peak. In this case, the user might guess the presence of the second peak and add another pole pair, but the out-of-band high-pass is not...
visibly discernible. However, if the user has knowledge of the underlying model and chooses the correct structure of 2 zeros at the origin, 2 real poles, and 2 conjugate pole pairs, both high-pass filters and both peaks are well fit and the original pole/zero structure recovered (Figure 6).

**FIGURE 5:** Bode plots of system identification curve fits guided by visual analysis of the data (Visual) or by foreknowledge of the underlying model (Known Model).

### Two-Port Network Functions

A deeper layer of knowledge can be applied to network functions solely from the fact that they are derivable from a set of linear equations embodying the Kirchoff Voltage Laws of the network [4]. It is **not** true that any function constructed of poles and zero can represent a network function. In addition, multiple network functions derived from the same network must be interrelated. It is not sufficient to perform individual curve fits on different network functions of a system and expect to have a characterization of the system.

The nature of these structural constraints and interrelations is known in the field of analog network synthesis, but not yet well understood by this author. It seems reasonable to work the problem forward by starting with a network that we believe represents the transducer, derive
the network functions, and fit them to measured data. To explore this method, we follow the same steps as in the previous section by generating data from a known network and attempting to reconstruct the pole, zero, and gain values of the network functions.

Figure 7 shows a network representing a hearing aid receiver. The model has been reduced to the minimum number of components which properly preserve the relationships between the input and output. While it is shown as an electrical network, it should be understood that the left side represents the electrical input and the right side the acoustic output. The presence of the gyrator ensures the correct anti-reciprocal behavior as noted in Eq.(8). The elements of the impedance function matrix derived directly from this network are

\[
Z_{11} = \frac{V_1}{I_1} \bigg|_{I_2=0} = g_{11} \frac{(s-z_{1r})(s-z_{2r})(s-z_{3r})^2 + z_{3r}^2}{(s-p_{1r})(p_{2r}^2 + (s-p_{2r})^2)} \tag{6}
\]

\[
Z_{12} = \frac{V_1}{I_2} \bigg|_{I_1=0} = g_{12} \frac{s}{(s-p_{1r})(p_{2r}^2 + (s-p_{2r})^2)} \tag{7}
\]

\[
Z_{21} = \frac{V_2}{I_1} \bigg|_{I_2=0} = -Z_{12} \tag{8}
\]

\[
Z_{22} = \frac{V_2}{I_2} \bigg|_{I_1=0} = g_{22} \frac{(s-z_{4r})(s-z_{5r})^2 + z_{5r}^2}{s(s-p_{1r})(p_{2r}^2 + (s-p_{2r})^2)} \tag{9}
\]

This gives us three equations, the electrical input impedance \((Z_{11})\), the transfer impedance \((Z_{12})\), and the acoustic output impedance \((Z_{22})\). The magnitudes and poles and zeros of the three impedance functions are plotted in Figure 8 and tabulated in Table 2. It is interesting to note that this set of poles and zeros completely characterizes the linear behavior of the receiver, at least as well as the network representation does.

Eqs.(6–9) reveal that the impedance functions share common poles. Our first impulse is to fit the impedance functions individually then reconcile the common pole values. To test this, we use simulate the network in Figure 7 under the appropriate drives and shorting conditions to
generate three sets of test data, this time without adding noise. We individually fit each impedance function to the generated data. The results, along with the pole and zero values derived directly from the network, are tabulated in Table 2. What we see here is startling. The electrical input impedance $Z_{11}$ is almost completely insensitive to the free parameters $p_1$ and $z_1$. Their values are essentially arbitrary as long as they are relatively close together. There is no way to find the value of $p_1$ without the other curve fits, and without $p_1$ a value of $z_1$ cannot be found.

**TABLE 2:** Poles and zeros of the network in Figure 7, derived directly from the network, by individual curve fits, or by simultaneous curve fit.

<table>
<thead>
<tr>
<th>Target</th>
<th>$Z_{11}$</th>
<th>$Z_{12}$</th>
<th>$Z_{22}$</th>
<th>$Z$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1$</td>
<td>$-328.$</td>
<td>$-239119.$</td>
<td>$-328.$</td>
<td>$-328.$</td>
</tr>
<tr>
<td>$p_2$</td>
<td>$-801. \pm j20594.$</td>
<td>$-809. \pm j20595.$</td>
<td>$-801. \pm j20594.$</td>
<td>$-801. \pm j20594.$</td>
</tr>
<tr>
<td>$z_1$</td>
<td>$-309.$</td>
<td>$-248200.$</td>
<td></td>
<td>$-309.$</td>
</tr>
<tr>
<td>$z_2$</td>
<td>$-2639.$</td>
<td>$-2564.$</td>
<td></td>
<td>$-2639.$</td>
</tr>
<tr>
<td>$z_3$</td>
<td>$-1132. \pm j23633.$</td>
<td>$-1126. \pm j23642.$</td>
<td></td>
<td>$-1132. \pm j23638.$</td>
</tr>
<tr>
<td>$z_4$</td>
<td>$-145.$</td>
<td></td>
<td>$-145.$</td>
<td>$-145.$</td>
</tr>
<tr>
<td>$z_5$</td>
<td>$-727. \pm j14262.$</td>
<td></td>
<td>$-727. \pm j14262.$</td>
<td>$-727. \pm j14262.$</td>
</tr>
<tr>
<td>$z_6$</td>
<td>$-2665. \pm j51105.$</td>
<td></td>
<td>$-2665. \pm j51105.$</td>
<td>$-2665. \pm j51105.$</td>
</tr>
</tbody>
</table>

One can envision a decoupled means of solving this problem, for example by fitting the transfer impedance and putting the determined value of $p_1$ back into the input impedance function curve fit as a fixed parameter. A more robust and general solution is to fit all three functions simultaneously with a single set of parameters. The first challenge is that these three functions have very different magnitudes, so the biggest one will tend to dominate over the others. The data normalization process previously described handles this nicely, and puts all three data sets on the same scale. The next step is to combine the three data sets with independent variable $\Omega$ into a single data set with two independent variables, the index of the data set $i$ and the normalized frequency $\Omega$, visualized in Figure 9. Finally, the three functions are combined into a single function

$$Z = \delta_{1-i}Z_{11} + \delta_{2-i}Z_{12} + \delta_{3-i}Z_{22},$$

(10)

where $\delta_i$ is the Kronecker delta function

$$\delta_i = \begin{cases} 0, & \text{if } i \neq 0 \\ 1, & \text{if } i = 0 \end{cases}.$$  

(11)
The multi-function curve fit performed smoothly in *Mathematica* and successfully reconstructed the original impedance matrix functions, as shown in the final column of Table 2.

![Figure 9: Combining three data sets of one independent variable to a single data set in two independent variables.](image)

**CONCLUSIONS**

The ultimate goal of this project is to determine if a network representation of a transducer could be determined directly from measured data, or at least if a matrix of transfer functions could be fit to measured data to characterize the two-port behavior of the receiver. This report shows significant progress but only partial success. Deriving a rational polynomial from a set of measured data has been successfully demonstrated, but the result can only be considered an approximating function and not a physical model. If one starts with a network model, however, it has been demonstrated in concept that the two-port network functions of that model can be simultaneously fit to generated data. The next step of this work is to fit a network model to actual data measured on a physical transducer, and to relate the pole, zero, and gain values thus determined back to network component values.

**REFERENCES**


