ZEROPOLE: A NEW TOOL FOR TEACHING FILTER THEORY AND DESIGN

Christophe VIGNAT and Sébastien VALLÉRY

Laboratoire Systèmes de Communication
Université de Marne la Vallée
77454 Marne la Vallée Cedex 2 - France

ABSTRACT

In this paper, we present a new Matlab program called ZeroPole (ZP) that is intended to help teachers in their graduate level lectures on linear systems and digital filter design. This graphical and intuitive environment allows students to design filters and, more importantly, to understand the relationships between the basic notions associated with this field, eg frequency and impulse response, transfer function, zeros and poles: the graphical drag and drop capabilities of ZP allow to modify any of these features and to observe simultaneously the corresponding modifications on the other features. Some examples of applications of this toolbox are exhibited in this paper, as they were introduced with success in a Signal Processing tutorial at the university.

1. INTRODUCTION

Teaching digital filter theory and design requires, in addition to a rigorous theoretical lecture, experimental tools that allow students to understand the fundamental notions (frequency and impulse response, zeros and poles) that underlie this field. To our best knowledge, no such graphical and intuitive tool was available under Matlab until now: the **Zeropole** (ZP) program was designed to fill this gap. This toolbox was developed at Laboratoire des Systèmes de Communications at University of Marne la Vallée, by S. Valléry and the author. It was uploaded on the "users contribution" directory at Mathworks, and is currently available at **ftp.mathworks.com/pub/users/contrib/**, or at **www-syscom.univ-mlv.fr/~vignat/**.

The first part of this paper is dedicated to the review of the principal difficulties associated with teaching the theory of discrete linear systems, justifying the creation of this new toolbox. The second part shows in detail how these difficulties are raised by using ZP, through an extensive presentation of its features. In the same part of this paper, we explain how this toolbox can be used in a real tutorial session. For the sake of clarity, snapshots of the main screen of ZP are displayed throughout this article.

2. THE ISSUES OF DIGITAL FILTER THEORY TEACHING

Digital filters are often introduced as a corollary of continuous time filters and sampling theory. The first difficulty associated with the discrete time framework is the relationship between the (z domain) transfer function $H(z)$ and the (frequency domain) frequency response $h(f)$, as described by the following formula

$$h(f) = H(z)_{|z=e^{j2\pi\frac{f}{f_s}}} \qquad (1)$$

As a consequence, many students fail to link the z-transform to the frequency response, and especially to locate the "low-frequencies" and "high-frequencies" on the f_s-periodical graph of any frequency response. Using ZP, formula (1) can be easily understood since the function $h(f)$ is plotted in 3D as the intersection of the unit cylinder with function $H(z)$; this feature helps the student to understand the meaning of formula (1) as well as the f_s periodicity of the frequency response (see Fig. 1).

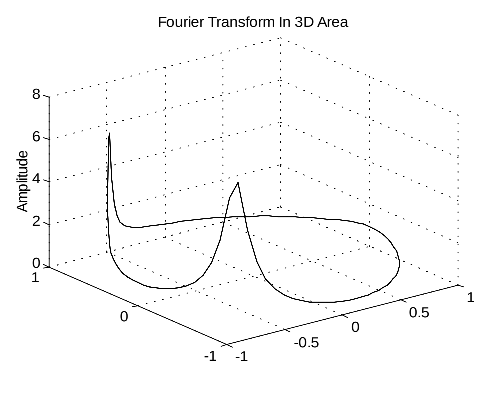

Figure 1: 3*D view of the frequency response*

A practical illustration of the relationship between the location of a pole or a zero of the filter and the shape of its frequency response is of importance. This is why ZP allows to modify graphically either the location of the poles/zeros, either the shape of the frequency response, and to check simultaneously the influence of these modifications on the other parameters. This tool allows at the same time to restate clearly the filter design problem, that is: how to place the zeros/poles in the complex plane so that a given shape of the frequency response is designed.

A third difficulty about teaching the discrete filter theory concerns the stability issue: the stability condition in the (causal) discrete time case ($|z_i| < 1$ for all poles z_i) is often confused with the stability condition in the continuous time case ($Re(p_i) < 0$ for all poles p_i). In ZP, the poles are not permitted to be placed outside the unit circle; moreover, the presence of near unit-circle poles translates immediately into high amplitude impulse and frequency response, hereby warning the student about the instability issue and allowing to memorize the stability condition.

The essential role of sine and cosine signals - as eigenfunctions of linear time invariant systems - is of outmost importance in the theory of digital filters. It appears thus essential that a student can visualize the action of a filter on a sine signal: ZP includes, as a part of the main screen, a graphical tool that draws the sine input with the output of the currently studied filter: both amplitude and frequency of the sine input can be graphically modified, illustrating the following important results:

• the frequency invariance of a sine signal through linear filtering

• the role of the poles/zeros and frequency response on the amplitude and phase of the output sine

As a last issue, we have found it interesting to highlight the relationship between the pole/zero location and the impulse response: the computation and graphical update of the impulse response simultaneously to any zero/pole modification allows to make understand the notions of impulse response, and the distinction between IIR and FIR filters.

3. THE ZEROPOLE UTILITY

ZP is a Matlab toolbox that involves, for the sake of clarity, a single, main graphical screen, as illustrated in Fig. 2.

The main screen is divided into five areas:

• the "z-transform" area (upper left) represents the complex plane, the unit circle and the zeros (x) and poles (o) of the filter currently designed by the student

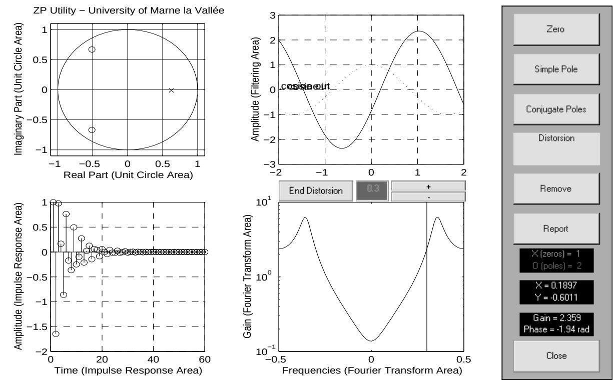

Figure 2: *the main screen*

• the "impulse response / phase" area (lower left) represents either the impulse response or the phase of the current filter

• the "sine in / sine out" area (upper right) shows either the sine output of the current filter superposed with the sine input signal, either the 3D frequency response as shown on Fig.1

• the "frequency response" area (lower right) displays the frequency response of the current filter

• the "user" area (right) includes the selection buttons, and informations about the filter.

Let us now present the different educational goals that can be reached using this program.

3.1. understanding the relationship zeros-poles / frequency response

ZP allows naturally to understand how a given frequency response can be synthesized using the poles and zeros location, and reciprocally, how this location influences the frequency response. The student can either:

• put simple zeros, simple or complex conjugated poles inside the unit circle and check the shape of the associated frequency and impulse response

• drag and drop these poles and zeros and check instantaneously the corresponding deformation upon the frequency and impulse response

• reciprocally modify the frequency response by dragging its shape: the zeros and poles move then according to the designed filter

A typical tutorial involving this feature is the following:

1− compute the frequency response of a filter having only one zero $z = \rho e^{i\theta}$ with $\rho = 0.8$ and $\theta = 0.3$ and verify the result using ZP

2− study the limit cases $\rho \simeq 0$ and $\rho \simeq 1$ for $\theta = 0.3$ and conclude about the role of ρ; verify the result by dragging the zero inside the unit circle while keeping θ constant

3− study the influence of θ upon the frequency response; check the result by rotating the zero in the unit circle while keeping ρ constant

4− do the same study using one simple pole, and a pair of complex conjugated poles (see Fig. 3- 4 and Fig. 5- 6)

5− consider a filter with one zero and one pole, and describe the behavior of its frequency response as the pole gets close to the zero. What happens when both get superimposed ?

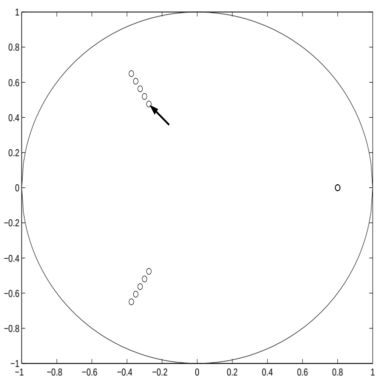

Figure 3: *moving radially a pair of conjugated poles*

3.2. understanding the relationship zeros-poles / impulse response

The same kind of study can be performed about the impulse response. However, contrary to the preceding case, the impulse response can not be modified graphically: the computation load required to compute the corresponding zeros and poles is too heavy so that the

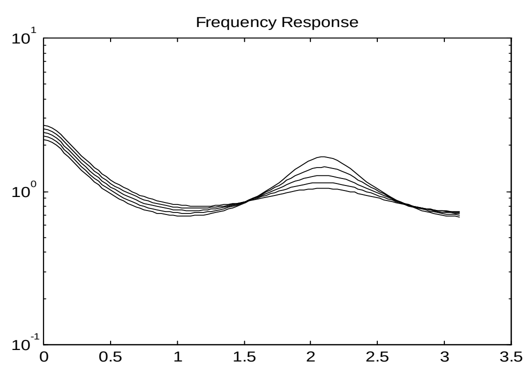

Figure 4: *the resulting frequency responses*

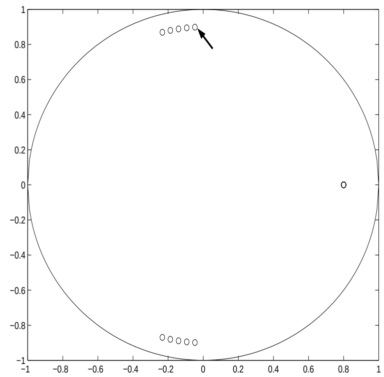

Figure 5: *moving the conjugated poles* ($\rho = K$)

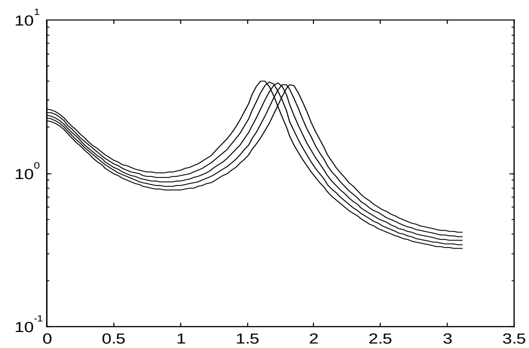

Figure 6: *the resulting frequency responses*

"real-time" update of the zeros and poles is not convincing. We are currently working on an approximation method for bridging this gap.

However, the dragging of any pole/zero instantaneously modifies the impulse response. This allows the student to understand, for example, the influence of the modulus of a pole upon this response. A typical tutorial writes as follows:

1− recall the definition of the impulse response

2− explain theoretically the influence of a single zero on the impulse response and check the result using ZP

3− explain theoretically why, in the case of single zeros and poles, the impulse response can be considered as a weighted sum of elementary responses, each corresponding to one pole/zero and check experimentally the result in a simple case

4− how does the presence of high-frequency components in the frequency response translate on the impulse response (which is its Fourier transform) ? Check the result on ZP

3.3. understanding the amplitude behavior of a filter

The significance of the frequency response can be easily understood thanks to the "sine in/sine out" area that represents both sine input and output of the filter. The frequency of the input can be chosen according to three ways:

- entering its numerical value in a dialog window
- dragging the input sine waveform until its shape corresponds to the desired frequency
- dragging the vertical line that represents the input frequency in the "frequency response" window

The third solution particularly highlights the significance of the frequency response, since the frequency gain can be read directly as the intersection of the vertical line (representing the input frequency) with the shape of the frequency response (see Fig.2 and 5b right bottom).

A typical tutorial writes as follows:

1− recall the significance of the frequency response of a filter, and the way to compute it from the transfer function

2− explain how a sine signal $s_n = A\cos\left(2\pi n \frac{f_0}{f_S}\right)$ is transformed by a linear filter with frequency response $h(f) = |h(f)|\, e^{j\phi(f)}$

3− check the result using ZP: design graphically a single zero filter such that $H(f_0) = 0.5$ for $\frac{f_0}{f_s} = \frac{1}{4}$ and verify the shape of the output

3.4. understanding the phase behavior of a filter

In the same fashion, ZP allows to understand the meaning of the phase response of a filter: the phase response of the studied filter can be plotted as a function of the frequency in the "Impulse Response" area; the student can thus check that the delay between the input and output sine simultaneously corresponds to the represented phase. A typical tutorial writes as follows:

1− study the theoretical phase response of a filter having (i) one simple zero, (ii) one simple pole and (iii) one pair of complex conjugated poles and check the result experimentally in each case

2− explain the influence of the angular position of a simple zero upon the phase response and check the result experimentally

3.5. understanding the design filter issue

We have added to ZP some functions that allow to interface it with the famous Matlab "Filter Design" program: this graphical program allows to synthesize a filter according to a set of constraints, and to some predefined method (Kaiser, Remez...). The filter, once computed, can be loaded into ZP as a starting point for further study: its poles/zeros can then be modified, some of them deleted or added in order to understand how the filter was designed. This is an interesting feature for educational purpose since it helps the student to differentiate between several classical methods of design. As an example, here is a typical study:

1− design a low-pass I.I.R. filter having the following characteristics: $F_s = 2kHz$, $F_{pass} = 500Hz$, $F_{stop} = 600Hz$, $ripple_{pass} = 3dB$, $ripple_{stop} = 50dB$ using the Chebyshev Type 1 method (see Fig. 7)

2− load the designed filter under ZP and check its characteristics, the location of its zeros and poles (see Fig. 8)

3− retry with other design methods and compare the advantages and drawbacks of each.

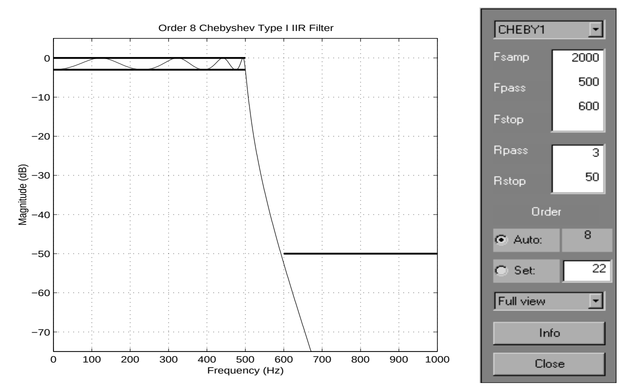

Figure 7: *Order 8 Chebyshev I filter*

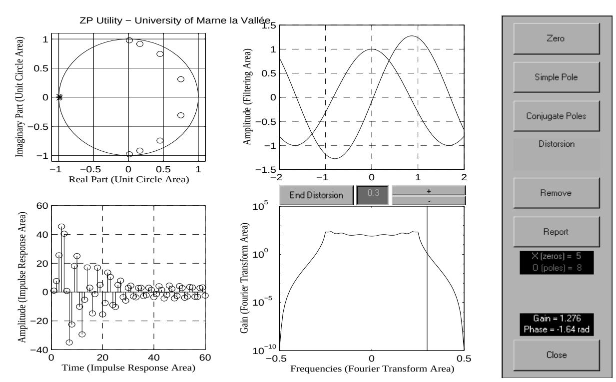

Figure 8: *the same filter under ZP*

4. CONCLUSION

The proposed utility was tested during tutorials in signal processing at University of Marne la Vallée. Its main benefits are:

- a graphical approach of filter design (no extra Matlab programming is required)
- an intuitive way to modify the filter parameters (essentially using the drag and drop technique)
- the clear highlighting of relationships between the basic notions underlying the discrete filtering theory: impulse and frequency responses, poles and zeros.

We are currently enhancing this program in order to increase its performances and bring new features, such as the real-time computation of the poles and zeros while the impulse response is modified.

REFERENCES

[1] Oppenheim, A.V., and R.W. Schafer. Discrete-Time Signal Processing. Prentice Hall, 1989