EXPERIMENTAL IDENTIFICATION OF LTI SYSTEM USING WHITE NOISE EXCITATION

P. Šuránek[*], J. Czebe[**], V. Pavelka[***]

Abstract: *This paper presents the measurement of frequency response function on an analog RC filter. The filter is excited using a white noise signal from a PC sound output. Input and output signals are measured by a headphone input, which serves for two-channel data acquisition system. Matlab is used for collecting and processing of the measured data.*

Keywords: frequency response, RC filter, white noise, Matlab

1. Introduction

Identification of various kinds of systems is a key procedure for its understanding. The frequency response function (FRF) is, together with step response, impulse response and root locus, one of the most commonly used ways how to describe a linear time-invariant (LTI) system's dynamics. FRF is used for describing a wide range of systems: electronic, hydraulic, mechanical, etc.

Several kinds of test signals are used for excitation of the systems during identification. Sines, swept sines, steps and impulses are among the simplest ones, where the behaviour of the system is observable by a human eye from the output signal. In this paper, the identification using random signal will be described. In this case, we cannot say anything about an identified system from the time capture of output. However, this method is valuable for a spectral analysis, and after applying a fast Fourier transform, quality results are obtained in a short period of time.

In this paper, measurement on an RC filter is presented. This filter was chosen as an educational example (students can simply compare theoretical and real results), but FRF's are usually in this manner measured for identifying mechanical structures (Maia et al., 1997; Wrona and Pawelczyk, 2016).

2. Frequency response function

For an LTI system with an input in the form of $x_0\sin(\omega t)$ with an amplitude x_0 and frequency ω, the output is $y_0\sin(\omega t+\varphi)$ with the same frequency, new amplitude y_0 and a phase shift φ.

Input and output signals can be written in Euler form:

$$x(t) = x_0 e^{i\omega t} \tag{1}$$

$$y(t) = y_0 e^{i(\omega t+\varphi)} \tag{2}$$

---

[*] Ing. Pavel Šuránek, Ph.D.: Faculty of Mechanical Engineering, VSB – Technical University of Ostrava,17.listopadu 15/2172; 708 33, Ostrava; CZ, pavel.suranek@vsb.cz

[**] Ing. Jiří Czebe: Faculty of Mechanical Engineering, VSB – Technical University of Ostrava,17.listopadu 15/2172; 708 33, Ostrava; CZ, jiri.czebe@vsb.cz

[***] Ing. Václav Pavelka: Faculty of Mechanical Engineering, VSB – Technical University of Ostrava,17.listopadu 15/2172; 708 33, Ostrava; CZ, vaclav.pavelka@vsb.cz

The FRF then describes the relationship between the output and input:

$$H(\omega) = \frac{y_0 e^{i(\omega t + \varphi)}}{x_0 e^{i\omega t}} = A(\omega) e^{i\varphi(\omega)} \tag{3}$$

The frequency response function then describes the magnitude $A(\omega)$ (ratio y_0/x_0) and phase $\varphi(\omega)$ as functions of frequency; this is the most common interpretation and the simplest for human imagination. But the FRF is a complex function and therefore different ways, how to illustrate it can be found: real and imaginary parts drawn separately, Nyquist plot, rarely a 3D plot consisting of real, imaginary part and frequency can be found (Maia et al., 1997).

2.1 Computation of frequency response function

When the system is excited by a noise signal, the procedure of evaluating the frequency response function is the following. Two discretized signals are measured at the beginning. The output signal is marked as $y(kT)$ and the input signal as $x(kT)$. Both signals are then divided into data blocks with 2/3 overlay and M realizations of signals $y_1(kT) \dots y_M(kT)$, $x_1(kT) \dots x_M(kT)$ are extracted. All the realizations have N samples. N should be a power of 2. In the next step, each of the blocks is weighted by a Hanning window, and the obtained results are transformed using the Fast Fourier transform (FFT) into the frequency domain representations $Y_1(f) \dots Y_M(f)$, $X_1(f) \dots X_M(f)$. The complex conjugates are calculated by changing the sign of the imaginary part of complex numbers, and these representations are marked with superscript "*": $Y_1^*(f) \dots Y_M^*(f)$, $X_1^*(f) \dots X_M^*(f)$. Now, it is possible to calculate auto-spectrum S_{yy} of the output signal and a cross-spectrum S_{xy}.
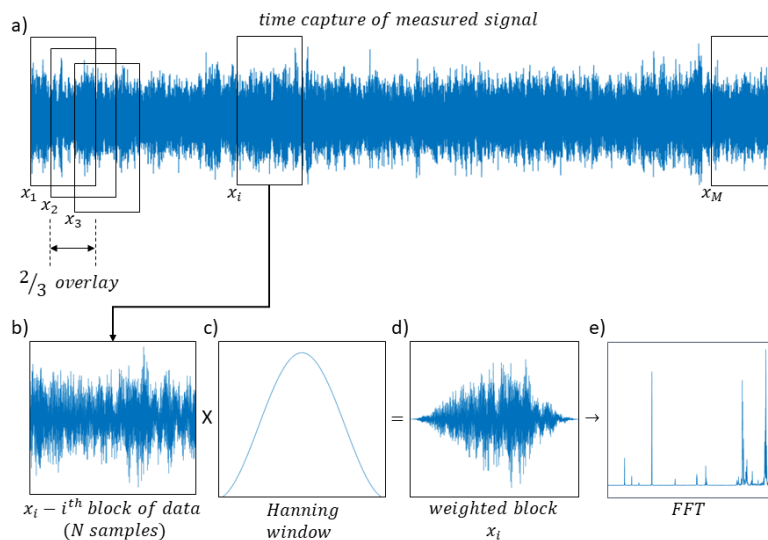

Fig. 1: Dividing the measured signal into M blocks and weighting by Hanning window.

The auto-spectrum (also known as power spectrum) of the signal weighted block is computed as a product of signal's Fast Fourier transform and its complex conjugate. It is calculated as an average over M measurement realizations.

$$S_{yy}(f) = \frac{1}{M} \sum_{i=1}^{M} Y_i(f) \cdot Y_i^*(f) \tag{4}$$

The averaging smooths the resulting spectrum but also consumes measurement time, because more samples are needed for the experiment.

The cross-spectrum is computed as a multiplication between the Fourier transform of the input signal and the corresponding complex conjugate of the output signal's Fourier transform.

$$S_{xy}(f) = \frac{1}{M} \sum_{i=1}^{M} X_i(f) \cdot Y_i^*(f) \tag{5}$$

The frequency response function is simply the cross-spectrum divided by the auto-spectrum of the reference signal:

$$H(f) = \frac{S_{xy}(f)}{S_{xx}(f)} \qquad (6)$$

2.2 Matlab experiment with RC filter

A Matlab script was created for the evaluation of the FRF on an RC filter. In the beginning, the sampling frequency of the sound card is set. The random signal is prepared and sent to the RC filter. Input and output signals are measured using microphone input, and then the FRF is computed according to the previous chapter.

```
%===============================================================================
%                    MEASUREMENT OF FREQUENCY RESPONSE FUNCTION
%-------------------------------------------------------------------------------
% This script measures frequency response function using white noise. PC sound card is used for
% noise generation, microphone input is used for measurement of two channels.
%===============================================================================

%-------------------------------------------------------------------------------
% Generating the noise sound
%-------------------------------------------------------------------------------
sf = 44100;                        % sampling frequency
sd = 5;                            % length of sound (seconds)
n = sf * sd;                       % number of samples
md = 3;                            % length of measurement (seconds)
noise = randn(1, n);              % noise generation
noise = noise / max(abs(noise));  % normalizing noise to max 1V
sound(noise, sf);                  % plays signal at the PC headphones output
pause(1);                          % 1 second pause

%-------------------------------------------------------------------------------
% Recording the filter's input and output signals
%-------------------------------------------------------------------------------
recorder = audiorecorder(sf,16,2); % microphone input setting (sampling, 16 bits, 2 channels)
recordblocking(recorder,md);       % recording (object, time)
output=getaudiodata(recorder);     % saving the measured data into variable output
X=output(:,2);                     % saves the second channel into variable X (RC filter input)
Y=output(:,1);                     % saves the first channel into variable Y (RC filter output)
clear ('recorder','output')

%-------------------------------------------------------------------------------
% Dividing the signals to blocks with 2/3 overlay
%-------------------------------------------------------------------------------
lW=4096;                           % length of one block (samples)
nW=floor(sf*md/lW*3-2);            % number of blocks
XW=zeros(lW,nW);                   % empty matrix for input blocks
YW=zeros(lW,nW);                   % empty matrix for output blocks
% dividing the signal and time weighting by Hanning window, one column includes one block
for k=1:nW
    XW(:,k)=(X(1+(k-1)*floor(lW/3):1+(k-1)*floor(lW/3)+lW-1)).*hann(lW);
    YW(:,k)=(Y(1+(k-1)*floor(lW/3):1+(k-1)*floor(lW/3)+lW-1)).*hann(lW);
end

%-------------------------------------------------------------------------------
% Fast Fourier transform
%-------------------------------------------------------------------------------
FFTX=fft(XW)/lW;                   % FFT of input blocks
FFTY=fft(YW)/lW;                   % FFT of output blocks

%-------------------------------------------------------------------------------
% Auto-spectrum of the otput XX and cross-spectrum XY
%-------------------------------------------------------------------------------
YY=sum(FFTY.*conj(FFTY),2)/nW;     % auto-spectrum of output
XY=sum(FFTX.*conj(FFTY),2)/nW;     % cross-spectrum

%-------------------------------------------------------------------------------
% Frequency response function
%-------------------------------------------------------------------------------
H1=YY./XY;                         % values
freq=(0:(sf/lW):(sf-sf/lW));       % frequency
```

Fig. 2: Matlab script for the processing of frequency response function

Finally, the measurement results are presented. The RC filter consisting of 7.724 kΩ and 0.1 µF was measured as a low-pass and a high-pass filter. Measured FRFs are compared with analytical ones in the following figures.
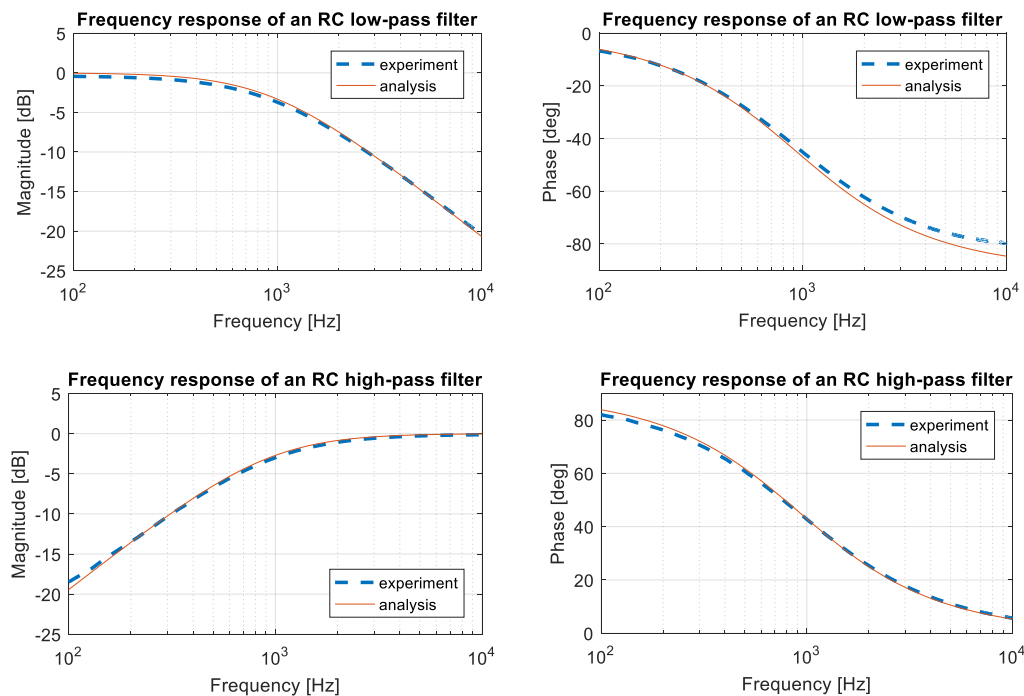

Fig. 3: High-pass filter results

3. Conclusions

The identification of an RC filter was presented in this paper. White noise signal was used for the excitation of this simple electronic system. This signal was achieved by generating random number signal. Noise signal was sent to an RC filter using PC sound output and the input and the output signal was measured using microphone input. A Matlab script was made to evaluate the frequency response function of the filter. It was achieved in several steps. The measured signals were divided into blocks with 2/3 overlay, these blocks were weighted by Hanning window, and the fast Fourier transform was applied to each block. The Auto-spectrum and cross-spectrum were able to be calculated after the averaging process. Finally, the FRF was computed as the auto-spectrum divided by the cross-spectrum.

Acknowledgement

References

Maia, N. M., Silva, J. M., He, J., Lieven, N. A., Lin, R. M., William, S. G. Urgueira, A. P. (1997). *Theoretical and Experimental Modal Analysis*. Research Studies Press, Baldock, Hertfordshire, England.

Wrona, S. and Pawelczyk, M. (2016). Shaping frequency response of a vibrating plate for passive and active control applications by simultaneous optimization of arrangement of additional masses and ribs. Part I: Modeling. *Mechanical Systems and Signal Processing*, Vol 70-71, pp. 682-698.

Wrona, S. and Pawelczyk, M. (2016). Shaping frequency response of a vibrating plate for passive and active control applications by simultaneous optimization of arrangement of additional masses and ribs. Part II: Optimization. *Mechanical Systems and Signal Processing*, Vol 70-71, pp. 699-713.