# Chapter 1

# Measuring transfer functions

The transfer function of a linear time-invariant system completely describes the system. The transfer function can be measured using a large number of different excitation signals, and can be calculated using a number of different methods. In this lab assignment we will investigate the pros and cons of different excitation signals that can be used to evaluate the transfer function. As an example, the transfer function of a filter will be measured. The measurements will be performed as is explained in Chapter 3 of the theory notes of this course.

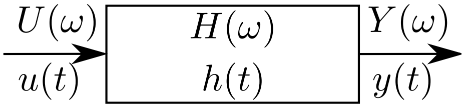


Figure 1.1: Representation of a Linear Time Invariant system.

## 1.1 Optimal excitation signals

### 1.1.1 Introduction

To allow for a proper selection of an excitation signal, it is first required to properly define what an optimal excitation signal is. The signal can be optimized with respect to a number of different criteria such as the required amount of excitation power, the time needed to perform the experiment, the cost of the experiment, the accuracy of the results, etc. [1]

Such an optimization will be highly dependent on the properties of the considered system. From a scientific point of view, the goal will always be to predict or explain the behaviour of a system using a mathematical model. To identify the parameters of the model with a low uncertainty, highly accurate measurements of the transfer function are required. This is the reason why the signals

[1] The diamonds (◇) are the questions that need to be answered or tasks you are supposed to do.

will be selected based on their ability to obtain accurate measurements for the transfer function.

### 1.1.2 Discretization of the measurement signals

Making measurements almost always requires one to use a computer these days. Therefore, it is mandatory to transform the analogue excitation and response signals in a stream of digital numbers. This process requires that the signal is discretized in time and in amplitude. Both discretization processes can degrade the measurement results significantly if they are not used very carefully, obeying all the theoretical rules.

#### 1.1.2.1 Discretization in time of the signal

To convert a continuous-time signal in a discrete time one requires that a sample and hold or a track and hold circuit is used. This circuit will freeze the instantaneous value of the analog signal taken at a time $t = nT_s, n \in \mathbb{N}$ and maintain it at the constant value $v(nT_s)$ during a complete sampling period.

#### 1.1.2.2 Quantization of the amplitude of the signal

The quantization of the amplitude of a signal is a process that maps the infinite (continuous) number of possible signal values on a discrete set of integer values. The circuit that performs this operation is called an Analog-to-Digital-Converter (ADC). A number of different possible operation principles are commonly used to construct ADC´s. They are discussed in some detail in the theory.

The transfer function of an ADC with 6 discretization levels is shown in Figure 1.2. The resolution of an ADC is typically specified as a number of bits; simple ADC´s have commonly a resolution of 10 bit (this means that 1024 different levels can be discerned) up to frequencies of a few Megahertz. The Least Significant Bit (LSB) is the voltage swing (in Volt) that corresponds to the minimal difference between two discretization levels of the discretized signal.

- Determine the voltage swing corresponding to 1 LSB for the NI ELVIS II acquisition hardware (16 bit), when the input voltage range is set to $[-10V\,,\,10V]$. ♢
- What is the dynamic range of the ADC used in the ELVIS II board? ♢

### 1.1.3 Factors that influence the accuracy of the measured signals

The accuracy of the measurement of the transfer function is determined by the joint influence of the measurement hardware and the algorithm that is used to calculate it. The errors that are introduced by the measurement process can severely be reduced by a proper choice of the excitation signal, however.

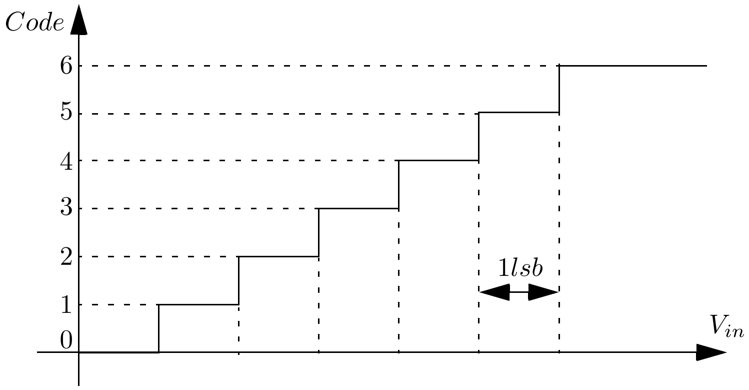


Figure 1.2: Transfer function of an ADC with 6 levels.

## Crest factor

Remember that the calculation of the transfer function is based on the discretization of the signals that are present at the ports of the device, and the accuracy of the discretization process depends on the time waveform associated to the excitation signal. If the ratio between the peak value and the mean value of the signal is very high, the accuracy will be adversely affected. Why is this the case? ♢

The crest factor has been defined to quantify the amplitude that a signal needs to produce a fixed excitation power level. It is defined as follows:

$$\text{Crestfactor} = \frac{\text{Peak value}}{\text{RMS value}} = \frac{\max_{t\in[0,T]}\left(\left|x\left(t\right)\right|\right)}{\sqrt{\frac{1}{T}\int_0^T x^2\left(t\right)dt}} \tag{1.1}$$

Different algorithms have been designed to minimize the crest factor of a periodic signal with a fixed power spectrum. As the waveform of the signal in the time domain is determined by the phase spectrum of the signal, those methods are mainly heuristics that help to choose the phase spectrum in such a way that the crest factor is lowered while the power spectrum is kept.

## Power spectrum

Of course the power spectrum of the signal can also be used to optimize the quality of the measurements. It is clear that it makes no sense to excite the signal in a frequency band where the transfer function is of no interest. Since the excitation of a system is almost always bound to a maximum safe power limit, concentration of the allowable power in the bandwidth of the measurement will increase the signal to noise ratio, and hence the quality of the measurement. When linear time invariant systems are considered, it is therefore always advantageous to increase the power level of the signal in the band of interest.

### Measurement errors

Measurement errors can in general be grouped in two classes, the stochastic ones and the systematic ones. The stochastic errors can be influenced by playing with the signal to noise ratio of the measurements. The systematic errors or bias errors are more insidious, as they do not manifest themselves as noisy additions to the measurement but they often result in a smooth shift of the measured characteristics. These errors will be looked at in the remainder of the lab.

## 1.2 The different excitation signals

The signals that will be used in this lab assignment are shortly described below.

### 1.2.1 The Impulse

Looking at the theoretic definition of the behaviour of a linear time invariant system in the time domain leads to the impulse response $h(t)$. Once this function of time is known, the response of the LTI system can be uniquely determined for all the possible excitation signals, as any signal can be written as the sum of an infinite number of Dirac impulses. The transfer function $H(\omega)$ of an LTI system is linked to the impulse response $h(t)$ through the Fourier transform,

$$H(\omega) = \mathcal{F}\{h(t)\}$$

where $h(t)$ is the impulse response of the system.

Using an impulse as an excitation signal therefore looks very attractive and simple to realize. For example, using a hammer to excite a mechanical system is an implementation of this method that is used in the everyday´s life of a mechanical engineer. However, there are some problems with the practical realization of this excitation.

- Remember that a Dirac impulse is both infinitely short and infinitely tall. Following the lines of the theory, it is impossible to realize such a signal in practice as this would require an infinite amount of energy and an infinite bandwidth. Again, we will have to get rid of the infinities of the theory and replace them by quantities that are large enough in order to keep the approximation errors that we make low enough. What rule of thumb would you propose to choose the duration of the impulse? Remember that the signal is to be applied to a certain system, and that the interaction between both produces the response. ♢

### 1.2.2 The multisine excitation

The general definition of the multisine is a sum of harmonically related sinewaves with a well chosen amplitude and phase spectrum,

$$x(t) = \sum_{i=1}^{N} X_i \sin(2\pi \nu_i f_0 t + \psi_i) \tag{1.2}$$

Note that the excited frequencies $\nu_i f_0$ are always an integer multiple of the basis frequency $f_0$ (the frequencies are therefore called commensurate). The number of excited frequency lines $N$, the amplitude of the excited lines $X_i$ and their respective phase $\psi_i$ can all be chosen freely to match the application and optimize some aspects of the measurement.

Figure 1.3 shows the time domain waveform of a multisine that consists of 1024 samples and contains $N = 100$ spectral components of equal amplitude. The phase spectrum has been calculated using Schröder´s equation to obtain a signal with a low crest factor without the need for an optimization algorithm. The amplitude spectrum is shown in Figure 1.3b. Note that since the multisine is a periodic signal, its spectrum consists of a number of discrete frequency lines only.

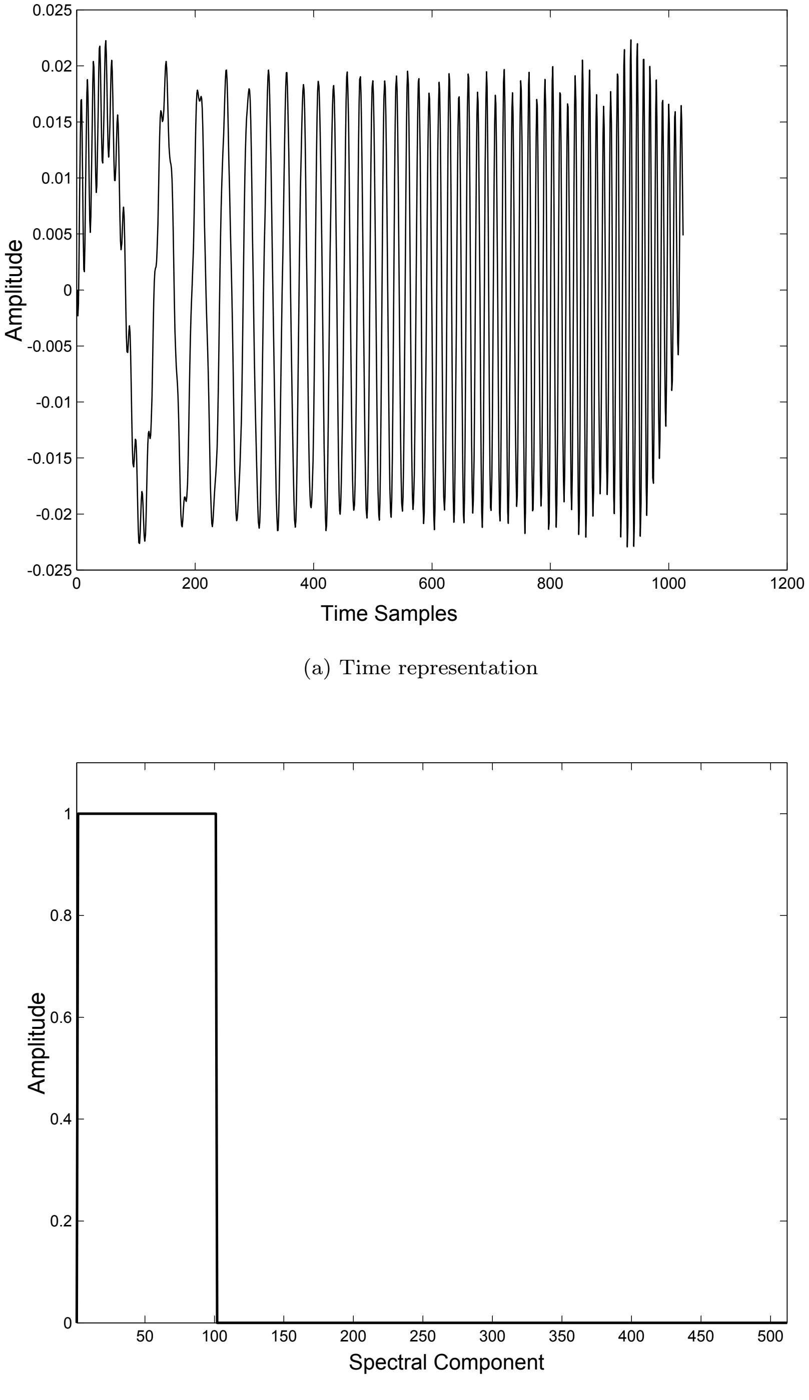
Now that we have taken a fast look at this signal, it is time for some action!

- Use MATLAB to calculate the samples of a multisine that consists of 4096 time samples and contains 100 excited spectral lines, located at the low end of the band (from line 1 to 100). Now calculate the waveform that is created when:
  - the multisine has a constant phase spectrum, ◇
  - the multisine has a random phase spectrum, ◇
  - the multisine has a Schröder phase spectrum; Choose the phases according to $\psi_i = \frac{i(i+1)\pi}{N}$, where $i$ is the line number (= the frequency of the line expressed in bins) and $N$ is the number of excited lines in the signal. ◇
- Calculate the crest factor of these signals. ◇
- Visualize the signals both in the time and the frequency domain then discuss differences and similarities. ◇

### 1.2.3 Noise excitation

Noise is a very popular choice for an excitation signal. Note that there is noise and noise in the literature. The noise that we consider here is the purely random noise, while the literature also contains lots of references to periodic noise. The difference is quit subtle, but results in a very different behaviour.

In the ´periodic´ noise case, the signal generator is loaded with one noise record and that same record is used for all the experiments. In fact, this boils



(a) Time representation

(b) Amplitude spectrum

Figure 1.3: Multisine with Schröder phase spectrum.

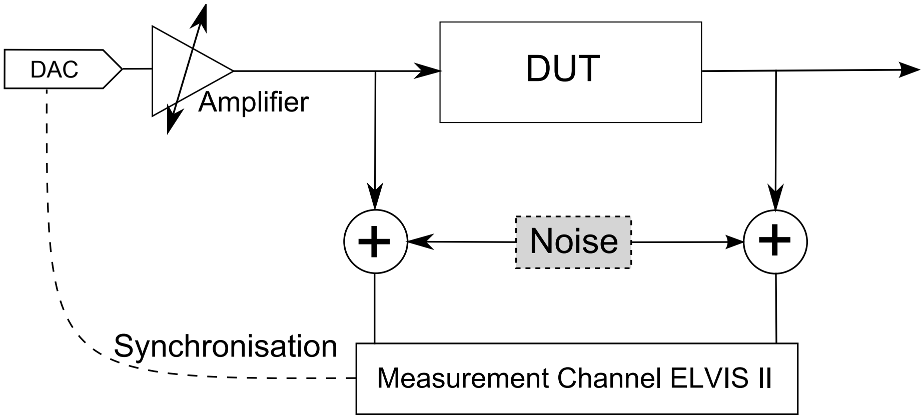


Figure 1.4: Measurement setup Lab 1.

down to the creation of a periodic excitation, a kind of multisine whose amplitude and phase spectra are both selected in a random way. Clearly, this is not what we call a random excitation in the context of this lab. Noise excitation requires that the signal generator is loaded with a fresh random signal for each experiment that is performed. Therefore, there is no periodicity at all and the signal will behave as predicted by the theoretical analysis for a noise excitation.

## 1.3 Experimental setup

Figure 1.4 shows the block diagram of the used measurement setup. The excitation signal is generated using an Arbitrary Waveform Generator (AWG or ARB). The voltages that are present at the input and the output of the DUT are both measured by the ELVIS II acquisition channel.

The internal operation of the acquisition channel is shown in Figure 1.5. It uses a multiplexer (MUX) followed by an Analog to Digital Converter (ADC). The multiplexer circuit ensures simultaneous sampling of the signals, even if their conversion is performed sequentially. The signals that are fed to the anti-alias (AA) filter are conditioned to increase the dynamic range of the setup.

To ease the operation, the AWG and the acquisition are integrated in the ELVIS II hardware. They use the same sampling clock. The software drivers that are needed to load and acquire signals are all provided in the Labview environment.

- What is the purpose of a trigger signal? In the introduction, a number of processing methods were discussed that show how to calculate the transfer function. Which methods require a trigger signal to operate properly? Explain what happens if the trigger is absent while needed. ♢

- If the clocks of the generator and the acquisition are not synchronized,

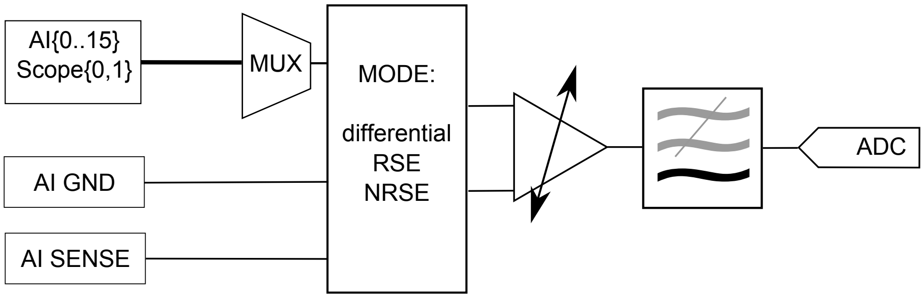


Figure 1.5: Measurement channel of ELVIS II.(More information can be found in the manual of ELVIS II)

which effect can you expect? Is this equally important for low and high frequencies? ♢

## 1.4 Now, it is up to you!

To avoid aliasing errors, determine the minimal sampling frequency that ensures these errors remain smaller than the resolution of the ELVIS II ADC. To get there, use the normalized transfer function of the anti-alias filter that is shown in Figure 1.6. ♢

A safe sampling frequency is now obtained and will be used in the remaining experiments.

### 1.4.1 Use different excitation signals

The lab setup contains a filter which has to be characterized. The transfer function is to be measured over the specified frequency band.

- Construct a multisine with a Schröeder phase. Use a record length of 4096 measurement points. The multisine should excite the frequencies from 1 Hz to 500 Hz with a frequency resolution of 1 Hz. Normalize the rms value of the input signal to 1 mV. Measure the system at a sample frequency of 8 kHz. What do you observe? ♢
- Repeat the previous measurement once the settings are corrected to ensure proper operation. Are all the measurement problems solved? What do you propose to increase the quality of the measurements? ♢
- Repeat the measurements for the excitation signals listed below. Use a record length of 4096 measurement points unless specified differently. Make sure you have 1 usable measurement without the influence of the transient. How can you know when the influence of the transient is gone?
  - A multisine with constant phase and excitation in the specified analysis band only ♢

- A multisine with an arbitrary phase and excitation in the specified analysis band only ◇
- A multisine with a Schröeder phase and excitation in the specified analysis band only ◇
- A multisine with a Schröeder phase and a data record length of 512 samples in the specified analysis band only ◇
- A multisine with a Schröeder phase and excitation outside the analysis band ◇
- A multisine with a Schröeder phase and a maximal amplitude of 0.5V ◇
- A multisine with a Schröeder phase and a maximal amplitude of 3V ◇
- A periodic noise signal ◇
- An aperiodic noise signal ◇
- A multisine that only contains excited lines at odd bin numbers. ◇
- Visualize the spectra of the input and the output using the `stem`-function. What does this measurement show you? ◇

Which effects do you expect to be responsible for changes in the transfer functions that result from a change in the excitation amplitude? How badly do you expect that these effects can influence the outcome of the experiment? ◇

## 1.5 Transfer function of the anti-alias filter

In this lab assignment, we use the Wavetek 432 programmable filter as an analog anti-alias filter. The frequency response of this device is shown in Figure 1.6.

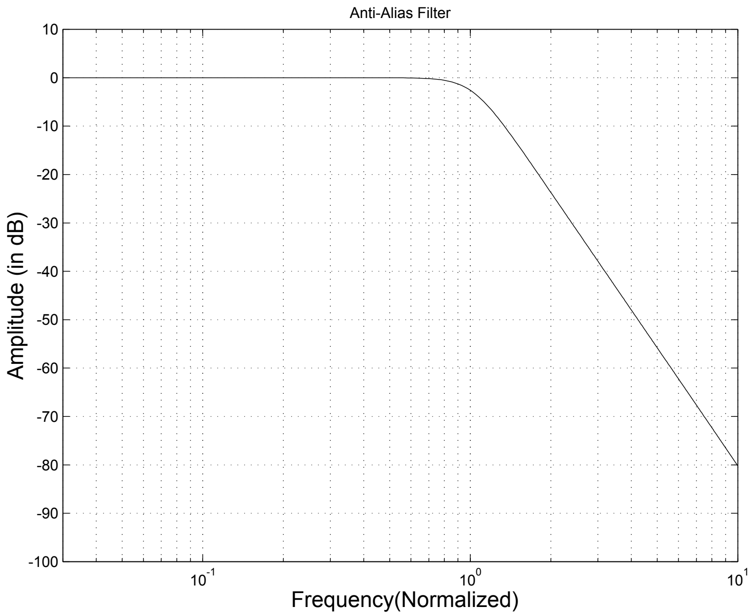


Figure 1.6: Amplitude-spectrum of the anti-alias filter.

## 1.6 The LabView-interface corner

Figure 1.7 shows the interface of the LabView Virtual Instrument (VI) that is used in the lab. The main parts are numbered on the figure and are explained below:

1. The connection that is used to connect a signal to the generator. Using specific arguments, such as `fgen` or `fgenBNC`, can lead to error conditions.
2. The connection mode of the acquisition. This connection can be specified on a channel by channel basis and allows for four different modes:

   - RSE or Referenced Single Ended connection. The reference terminal is the ground terminal of ELVIS II
   - NRSE or Non Referenced Single Ended. The reference terminal is labeled `AISENSE`
   - Fully differential connection
   - Pseudo-differential connection

   More information about the specific properties of these modes can be found in the manual of the ELVIS II board.

3. The sampling rate of the generator and the acquisition channels.

4. Limited frequency resolution can result in round-off errors between the requested and the actual value of the sampling rate that is used during the measurements. The actual value will be shown next to the inserted sampling frequency (3) during the measurements.

5. The excitation signal can be read using an exported CSV file. The first column of data in the file contains the actual information. CSV files can be generated in Excel and MATLAB (`csvwrite` or `dlmwrite`).

6. The connections for the measurement channels. Some combinations of connections can result in error conditions.

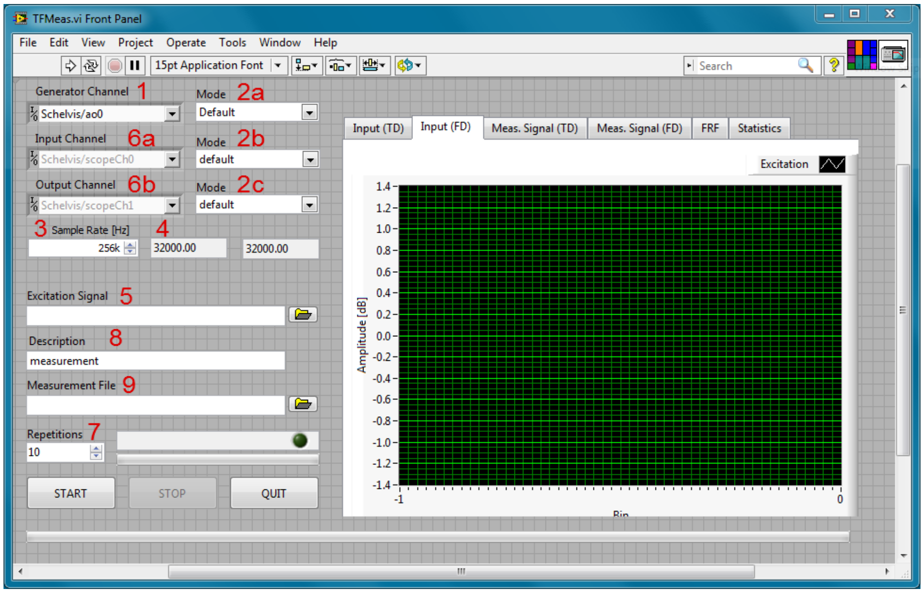
7. The integer number of periods of the signal that are to be acquired.

8. Title to be used for the measurement. This name is saved in the measurement data file.

9. Name of the file where the data have to be saved. Depending on the extension used in the filename, this is saved differently:

   **.MAT** MATLAB binary format,

   **else** CSV-file with a tab character used as a separator and some additional data put in the header.



TFMeas.vi Front Panel

Generator Channel 1: Schelvis/ao0 — Mode 2a: Default

Input Channel 6a: Schelvis/scopeCh0 — Mode 2b: default

Output Channel 6b: Schelvis/scopeCh1 — Mode 2c: default

3 Sample Rate [Hz]: 256k — 4: 32000.00 — 32000.00

Excitation Signal 5:

Description 8: measurement

Measurement File 9:

Repetitions 7: 10

Figure 1.7: LabView-interface.

To be able to perform a measurement all the settings need to be filled out properly. Next, the `start` button is used to fire up the measurement. A message that shows the actual status is shown during the measurement and the progress bar indicates the ratio of the measurements that are already done over the total count of requested measurements. The measurement can be stopped by using the `stop` button and the measurement program can be totally interrupted using the `quit` button.

## 1.7 MATLAB self-help corner

### 1.7.1 dB

```
function [result] = dB(input, ref, verbose);
```

dB calculates the dB value of a voltage input argument.

**Usage:**

- `result = dB(input)`
- `result = dB(input, ref)`
- `result = dB(input, ref, verbose)`

## 1.8 References

- ELVIS II manual http://doiop.com/elvismanual
- ELVIS II Datasheet http://doiop.com/elvisdatasheet

For any questions please email and arrange a date and time with one of the lab assistants.

| | |
|---|---|
| Maarten Schoukens | Maarten.Schoukens@vub.ac.be |
| Mark Vaes | Mark.Vaes@vub.ac.be |
| Maral Zyari | Maral.Zyari@vub.ac.be |
| Dieter Verbeke | Dieter.Verbeke@vub.ac.be |

# Chapter 2

# Measurement in the presence of noise

Unlike simulations, real life experiments will always be influenced by noise as seen in Lab1. This noise can lead to a biased estimation and/or high variability of your model if not taken into account. A solution to this problem is averaging over multiple periods to obtain the Frequency Response Function (FRF). In this lab assignment we study different methods to measure/estimate the FRF. The results obtained will be compared and evaluated.[1]

## 2.1 Different averaging methods to calculate the Frequency Response Function

Consider that the measured data records $u(kT_s)$ and $y(kT_s)$, taken at the input and the output of the device under test (DUT) are known. Herein,

$$\begin{array}{l} n = 0, 1, \ldots, N-1 \\ k = 0, 1, \ldots, N-1 \\ T_s = 1/F_s \end{array} \tag{2.1}$$

$N$ is the number of measured samples and $F_s$ is the sampling frequency at which the samples are taken.

Define $U\left(k\right)$ and $Y\left(k\right)$ as the discrete Fourier transform (DFT) of the measured time records:

$$U(k) = \text{DFT}\left\{u\left(nT_s\right)\right\} \tag{2.2}$$

$$Y(k) = \text{DFT}\left\{y\left(nT_s\right)\right\} \tag{2.3}$$

[1] The diamonds (◇) are the questions that need to be answered or tasks you are supposed to do.

In the ideal periodic case, the FRF $H(jk\omega_0)$ is simply the division of the DFT spectra.

$$H(jk\omega_0) = \frac{Y(k)}{U(k)} \tag{2.4}$$

where the angular frequency $\omega_0 = 2\pi F_s/N$.

Under practical conditions, the measurements will always be distorted by measurement noise. As a result, the calculated FRF will only approximate the ideal one with a finite accuracy. To improve on the accuracy, one can measure more than one data record (spending more time and money). Those multiple records $u_i(nT_s)$ and $y_i(nT_s)$, $i = 1, 2, \ldots, M$, can then be averaged to improve the accuracy and precision. The problem is that this averaging can be performed using a number of different methods, and that these methods, even when fed by the same data, will not produce the same results. We will now look at the different methods and determine their advantages and disadvantages.

1. Averaging time domain data records:

$$\begin{aligned} u(nT_s) &= \frac{1}{M}\sum_{i=1}^{M} u_i(nT_s) && (2.5)\\ y(nT_s) &= \frac{1}{M}\sum_{i=1}^{M} y_i(nT_s) && (2.6)\\ H(jk\omega_0) &= \frac{\text{DFT}\left\{y(nT_s)\right\}}{\text{DFT}\left\{u(nT_s)\right\}} && (2.7)\end{aligned}$$

2. Averaging the linear DFT spectra:

$$\begin{aligned} U(k) &= \frac{1}{M}\sum_{i=1}^{M} U_i(k) && (2.8)\\ Y(k) &= \frac{1}{M}\sum_{i=1}^{M} Y_i(k) && (2.9)\\ H(jk\omega_0) &= \frac{Y(k)}{U(k)} && (2.10)\end{aligned}$$

3. Averaging the FRF:

$$\begin{aligned} H_i(jk\omega_0) &= \frac{Y_i(k)}{U_i(k)} && (2.11)\\ H(jk\omega_0) &= \frac{1}{M}\sum_{i=1}^{M} H_i(jk\omega_0) && (2.12)\end{aligned}$$

4. Averaging the auto-power of the input signal and the cross-power:

$$G_{yu}(k) = \frac{1}{M}\sum_{i=1}^{M} Y_i(k)U_i^*(k) \tag{2.13}$$

$$G_{uu}(k) = \frac{1}{M}\sum_{i=1}^{M} U_i(k)U_i^*(k) \tag{2.14}$$

$$H_1(jk\omega_0) = \frac{G_{yu}(k)}{G_{uu}(k)} \tag{2.15}$$

5. Averaging the auto-power of the output signal and the cross-power results in similar equations:

$$H_2(jk\omega_0) = \frac{G_{yy}(k)}{G_{uy}(k)} \tag{2.16}$$

For the first two methods to work properly the data records have to be identical up to the noise contribution. This calls for a periodic excitation signal and a triggering signal that ensures that each measurement starts at a fixed point in the period of the excitation. The last two methods are also applicable to the measurements obtained with noise excitation. They do not at all require a triggering of the records but come at the cost of a bias. Fortunately, this bias is proportional to the signal to noise ratio of the signal, and therefore it can often safely be neglected. However, when the signal to noise ratio is small or the required accuracy is high, it can be important. Remember that it can be removed completely without an increase in the measurement time if a different (periodic) excitation would be selected! The bias can be proven to be such that the true value of the FRF is bound by the $H_1$ and the $H_2$ estimate,

$$|H_1| \leq |H| \leq |H_2| \tag{2.17}$$

For a fast conversion of the signals between the time and the frequency domain the use of the FFT is preferred. Note that this algorithm works faster when the length of the data records, $N$, is a power of 2. Do you know why? ♢

## 2.2 Excitation signals

Three types of signals will be used in this lab:

- Periodic noise signals
- Aperiodic noise signals
- Multisine (Schroeder or random phase)
- Odd random phase multisine

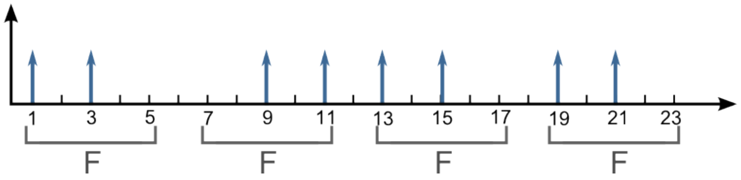


Figure 2.1: Odd random phase multisine

The first three signals have been generated and used in Lab 1. However if you have generated your multisine in the time domain you might have noted that it was quite tricky and time-consuming. Therefore in this lab we suggest that you generate the multisine from the frequency domain using the following hints: ♢

Hint 1 : Define the DFT spectrum $U(k)$ , $k = 0, 1, 2, ..., \frac{N}{2}$ and calculate $u = \text{ifft}(U)$. Apply a scale factor $N/2$ to compensate for internal scaling factor of the FFT. Notice that the FFT coefficient of a sinusoid with zero phase is $e^{(-j*\frac{\pi}{2})}$ for the positive frequency.

Hint 2: Make use of the fact that $U(k) = \overline{U}(N-k)$ and $xy+\bar{x}\bar{y} = 2\,\text{real}\,(xy)$

Next, generate the odd random phase multisine. The random harmonic grid is generated by grouping excited odd harmonics in $F = 3$ consecutive harmonics, and eliminating exactly 1 randomly selected odd harmonics in each group (see Figure 2.1). ♢

# Let's go practical!

Measure the system at the sampling frequency of 16kHz, using a periodic noise signal, an aperiodic noise signal, a full multisine of your choice (which one do you choose and why?) and a random phase multisine. ♢

- The record length consists of 8000 points.
- The multisine should excite frequencies from 1Hz to 2kHz.
- Make sure you have 32 usable measurements.

The measurement setup used in this lab is the same as in Lab 1, with the addition of a noise generator at the input and the output. (See Figure 1.1 )

### 2.2.1 Averaging the measurements

Apply the techniques that were described in Section 2.1 to the generated signals. Which techniques are applicable and which are not? What is the reason for this?

♢

- Determine the effect of the number of averaged records on the variability of the averaged result. Average over 2, 4, 16 and 32 signals and compare the standard deviation. ♢

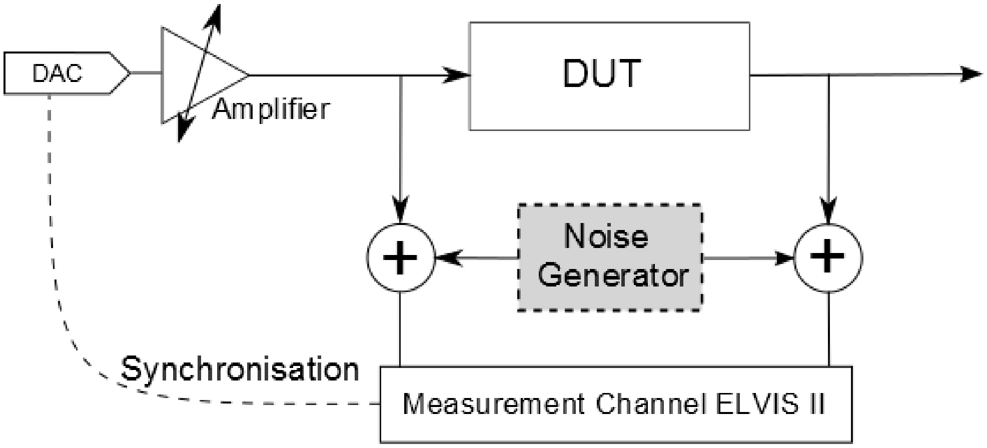


Figure 2.2: Measurement Setup

- The different averaging methods are now used on the same data. The lab files contain a function called `TransferFunc` that calculates the FRF and its standard deviation when fed with time records. This function does all the pre-processing and even calculates the standard deviation. However, there is a small part that is missing: the actual calculation of the FRF requires that the user provides a MATLAB function. The explanation of the technicalities that are needed are given in appendix at the end of the text. The idea is that you write the MATLAB functions that calculate the FRF using the proposed averaging methods when fed with repeated time domain records. [2]

- *Good to know*: the Standard Deviation (STD) is obtained as the sample variance of the FRFs that were calculated for each realization of the measurement. For example, if 32 measurements are available and one wants to obtain the variance based on the processing of 4 averaged data records per FRF calculation, only 8 repeated FRF measurements can be obtained. One of these FRFs is considered to be the result, while the standard deviation is evaluated using the 8 obtained FRFs. Note that as only 32 realizations of the measurements are present, there is only 1 FRF that can be calculated by taking the 32 repeated experiments into account and therefore the provided processing routine can not calculate the sample standard deviation of this FRF.

### 2.2.2 Data processing

Process the data of each measurement signal using all the averaging techniques. Discuss the differences and explain according to you, which will deliver the best

[2] For the cross- and auto-spectra methods, calculating either $H_1$or $H_2$ is sufficient.

result. Always average your results over the different measurements that you have gathered. Discuss the pros and cons of each excitation signal. ♢

## 2.3 MATLAB self-help corner

### 2.3.1 `dB`

`function [result] = db(input, ref, verbose);`
db calculates the dB value of a voltage input argument.

**Usage:**

- `result = db(input)`
- `result = db(input, ref)`
- `result = db(input, ref, verbose)`

**Input arguments:**

`input` The input value(s) to calculate the dB intensity.

`ref` The reference value that is used to calculate the dB value. If no `ref` is specified, 1 is used as reference.

`verbose` Makes the function verbose (1) or not silent (0), when omitted the function is silent.

### 2.3.2 `TransferFunc`

`function [H, stdH] = TransferFunc(u, y, Avgs, HFunction)`
TransferFunc calculates the FRF and its standard deviation, starting from a series of time domain measurements.

**Usage:**

- `[H, stdH] = TransferFunc(u, y, Avgs, @HFunction)`
- `[H, stdH] = TransferFunc(u, y, Avgs, ‘HFunction’)`

**Input arguments:**

`u,y` Matrices containing the time domain measurements of the input, resp. the output. Consecutive experiments are stored column-wise.

`Avgs` Specifies how many experiments must be averaged for the calculation of the FRF. If this is an array, all calculations are repeated for each element of this array and different FRFs are returned, one corresponding to each element.

`HFunction` MATLAB string containing the name of the function or function handle to be used for the actual calculation of the FRF. You must write such a function for every averaging method. See later for the requirements on these functions.

**Output arguments:**

`H` FRF of the system. If `Avgs` was an array, there is one FRF for each element of that array, and they are stored column-wise.

`stdH` Standard deviation on the different calculated FRFs, also stored column-wise.

### Requirements on the user specified function HFunction

**Usage:**

- `H = HFunction(u, y)`

**Input arguments:**

`u,y` Matrices containing the time domain measurements of the input, resp. the output. Consecutive experiments are stored column wise.

**Output arguments:**

`H` FRF from the measurements `u` and `y`, averaged over all provided experiments.

For any questions please email and arrange a date and time with one of the lab assistants.

| | |
|---|---|
| Maarten Schoukens | `Maarten.Schoukens@vub.ac.be` |
| Mark Vaes | `Mark.Vaes@vub.ac.be` |
| Maral Zyari | `Maral.Zyari@vub.ac.be` |
| Dieter Verbeke | `Dieter.Verbeke@vub.ac.be` |

# Chapter 3

# Distortion measurements

The goal of this lab assignment is to characterize the (nonlinearities) in the response of a system. In this lab we first study the response of a nonlinear system using a single sinewave as input. Next the same system is analysed using the multisine excitation. Using a specific type of multisine it is possible detect and qualify the (even or odd) nonlinear distortions. [1]

## 3.1 The concepts of Distortion and Nonlinearity

### 3.1.1 Single sine wave excitation

Any system that obeys the superposition principle is a linear system per definition. As a consequence, all the systems that do not obey the superposition principle are called nonlinear. This negative definition has the disadvantage that it contains almost anything, and therefore we will restrict the class towards systems that are close to linear. The main properties of the class of nonlinear systems Periodic In Same Period Out (PISPO) that we consider will be discussed generally.

A STATIC NONLINEAR SYSTEM is a nonlinear system where the relation between the input signal $u$ and the output signal $y$ can be described by a polynomial series expansion of the response with respect to the input signal in a given interval:

$$y\left(t\right) = \sum_{k=0}^{\infty} K_k u\left(t\right)^k = K_0 + K_1 u\left(t\right) + K_2 u\left(t\right)^2 + \ldots \tag{3.1}$$

For DYNAMIC NONLINEAR SYSTEMS this relation is no longer sufficient to describe the output signal as the system contains memory effects. The polynomial expansion then needs to be generalised to a Volterra series. We will not consider Volterra series here as this would lead us too far off track, they will be

[1] The diamonds (◇) are the questions that need to be answered or tasks you are supposed to do.

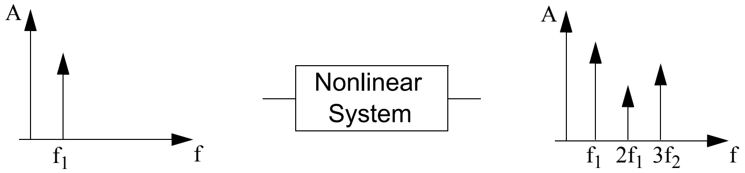


Figure 3.1: Input and output of a nonlinear system excited by a sinewave.

considered in detail in the course called "Measuring and Modelling of Nonlinear Systems".

For a linear time invariant system whose transfer function is labelled $H(f)$, it has been proven before that excitation with an input signal $u(t) = A_1 \cos(2\pi f_1 \cdot t)$ results in a response that is again a sinewave at the same frequency, but with a different amplitude and a different phase with respect to the input signal:

$$y(t) = |H(f_1)|\, A \cos(2\pi f_1 \cdot t + \angle H(f_1))$$

When nonlinear systems are considered this is no longer true. Consider that we have a static nonlinear system containing only a quadratic and a cubic response term. The output response $y(t)$ of this system to an excitation $u(t)$ is then given below:

$$y(t) = u(t) + u^2(t) + u^3(t) \tag{3.2}$$

Whenever $u(t) = \cos(\omega_0 t)$, the output $y(t)$ can be rewritten as:

$$y(t) = \sum_{k=1}^{3} \left[\frac{e^{j\omega_0 t} + e^{-j\omega_0 t}}{2}\right]^k = [\cos(\omega_0 t)] + \left[\frac{1}{2} + \frac{1}{2}\cos(2\omega_0 t)\right] + \left[\frac{1}{2}\cos(\omega_0 t) + \frac{1}{4}\cos(3\omega_0 t)\right] \tag{3.3}$$

The linear part of the response results in an output at the frequency $\omega_0 = 2\pi f_0$. The quadratic part of the response creates a DC component and a component at the second harmonic $2\omega_0$. The cubic term finally creates energy at the fundamental frequency $\omega_0$ and the third harmonic $3\omega_0$ (see Figure 3.1). Hence, the response now contains energy at frequencies that are an integer multiple of the basic frequency $\omega_0$. This type of behaviour is called HARMONIC DISTORTION.

If the input signal is a sum of two sinewaves, the response becomes more complex, as $x(t) = \cos(\omega_1 t) + \cos(\omega_2 t)$, results in an output signal as is shown below:

$$\begin{aligned} y(t) &= [\cos(\omega_1 t) + \cos(\omega_2 t)] \ldots \\ &+ \left[1 + \frac{1}{2}\cos(2\omega_1 t) + \frac{1}{2}\cos(2\omega_2 t) + \cos((\omega_1 + \omega_2)t) + \cos((\omega_1 - \omega_2)t)\right] \ldots \\ &+ \left[2\cos(\omega_1 t) + 2\cos(\omega_2 t) + \frac{1}{2}\cos(3\omega_1 t) + \frac{1}{2}\cos(3\omega_2 t) + \frac{3}{4}\cos(\omega_2 t - 2\omega_1 t) \ldots\right. \\ &+ \left.\frac{3}{4}\cos(\omega_2 t + 2\omega_1 t) + \frac{3}{4}\cos(\omega_1 t - 2\omega_2 t) + \frac{3}{4}\cos(\omega_1 t + 2\omega_2 t)\right] \end{aligned} \tag{3.4}$$

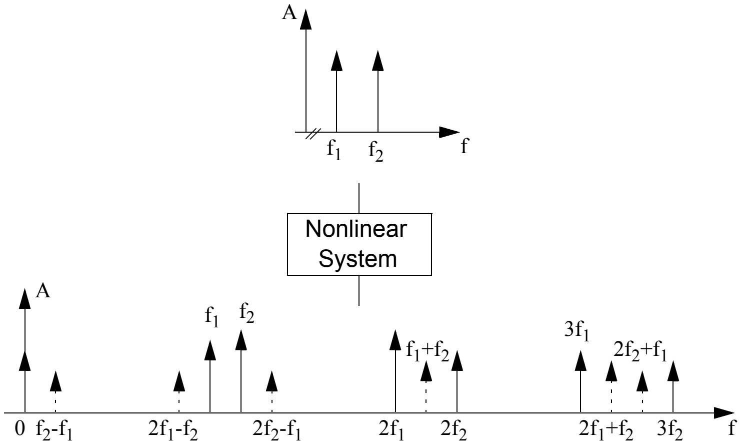


Figure 3.2: Input and output spectrum of a nonlinear system (full line = harmonic distortion, dashed line = intermodulation distortion).

The linear term results now in energy that appears at the frequencies $\omega_1$ and $\omega_2$. The quadratic term will not only produce a contribution at DC and at the frequencies $2\omega_1$ and $2\omega_2$, but also at the sum and difference frequencies $\omega_1 + \omega_2$ and $\omega_1 - \omega_2$. The cubic term creates energy at the frequencies $\omega_1$, $\omega_2$, $3\omega_1$ and $3\omega_2$ as could be expected, but in addition to that, contributions also appear at $\omega_2 - 2\omega_1$, $\omega_2 + 2\omega_1$, $\omega_1 - 2\omega_2$ en $\omega_1 + 2\omega_2$. Besides the harmonic distortion that was expected to be present at multiples of $\omega_1$ and $\omega_2$, a new type of contribution pops up. It is called the intermodulation distortion and appears at frequencies that are a combination of $\omega_1$ and $\omega_2$ simultaneously.

Determining the frequencies where energy is expected to pop up at the output fortunately does not always require to fully work out the equations (3.3) and (3.4). A simple rule of thumb can be used, and it is possible to derive it based on the simple example given above. An important point to be remembered is that a sinewave does not only have a contribution at positive frequencies, but also conveys energy at the negative frequency. Consider again a sinewave input as in (3.3). The linear term $u^1$, only creates energy in the output signal at frequencies that match the frequency of the input signal,

| Input | Output |
|---|---|
| $f$,$-f$ | $f$,$-f$ |

To evaluate the frequencies at which the quadratic term $u^2$ creates contributions, we must synthesize all possible outcome of the sum of 2 frequencies that are present in the initial signal,

| Input | Output | |
|---|---|---|
| $f, -f$ | $-f - f$ | $-2f$ |
| | $-f + f$ | DC |
| | $f + f$ | $2f$ |

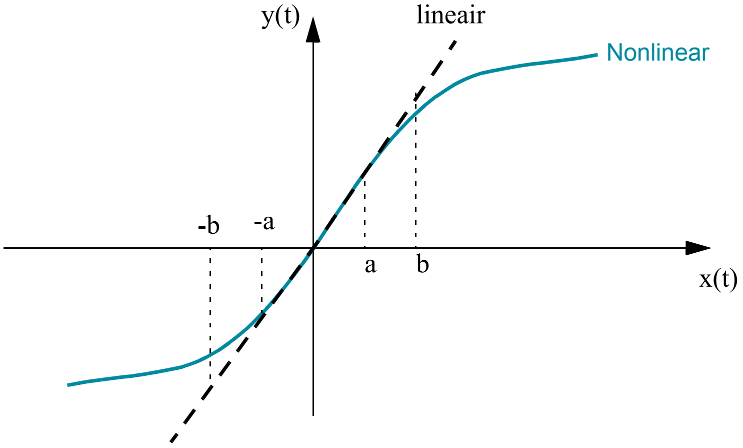


Figure 3.3: A static nonlinear response.

For $u^3$ we now have to combine 3 frequencies , taking the negative frequencies in consideration.

| Input | Output | |
|---|---|---|
| $f, -f$ | $-f-f-f$ | $-3f$ |
| | $-f-f+f$ | $-f$ |
| | $f+f-f$ | $f$ |
| | $f+f+f$ | $3f$ |

To obtain the output contributions for a two-tone input signal, the reasoning is completely similar. The linear term does only create contributions at the excited frequencies, the quadratic term creates energy at all frequencies that can be reached by the sum of 2 frequencies that are present in the input signal and the cubic term reaches all frequencies that can be written as the sum of 3 input frequencies.

A second important property of a nonlinear system is that, unlike a linear time invariant system, the amplitude size of the input signal influences the energy conversion of the system. The phase difference between components also influences the result, but the analysis of this behaviour falls out of the scope of this lab.

Figure 3.3 clearly illustrates that for an amplitude that lies between $a$ and $-a$ the nonlinear curve is very close to its linear component, while for a range of amplitudes between $b$ en $-b$ the difference with the linear term is way bigger. Stated differently, one sees that the nonlinearity is way less excited by the first range of excitations and hence the harmonic contributions will remain smaller.

### 3.1.2 Odd multisine excitation

As seen above, looking at the response of a single sine wave allows us to detect what kind of nonlinear distortions are present: even $(u^2, u^4, ..)$ , odd $(u^3, u^5, ..)$

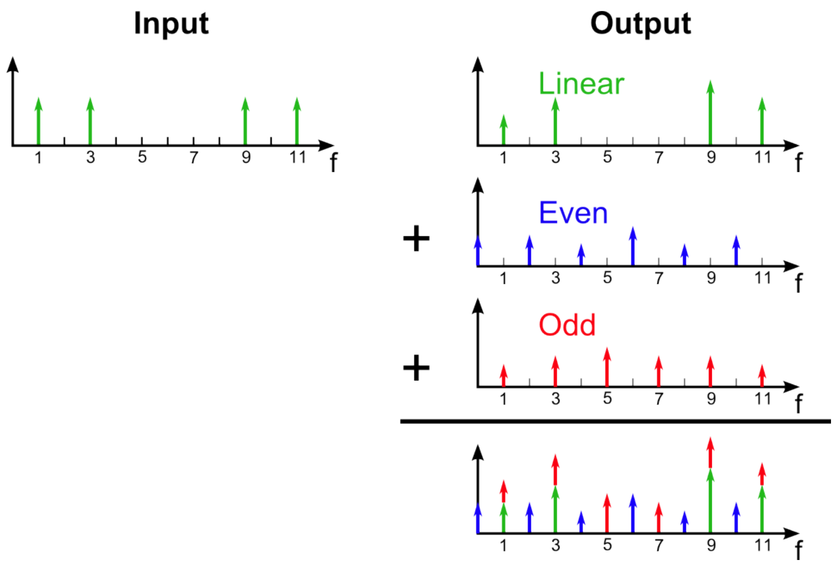


Figure 3.4: Response of an odd random phase multisine

or a combination of both. This is called qualification of a system. Knowing wether the system contains odd or even nonlinear distortions can be a great advantage as odd nonlinear distortions tend to change the dynamics of the system (example: resonance frequency shifts to the left or to the right). It would therefore be interesting to be able to combine the detection and qualification of the system while calculating the transfer function within the band of interest at the same time. This can be done using an odd random phase multisine with random harmonic grid as seen in Figure 3.4. By leaving the even harmonics out, it is possible to detect the even nonlinearities because an even sum of odd harmonics is an even number. The randomly eliminated odd harmonics allow us to detect the odd nonlinear distortions because an odd sum of odd harmonics is an odd number. More information can be found in detail in the course called “Measuring and Modelling of Nonlinear Systems”.

### 3.1.3 The Total Harmonic Distortion (THD) using a single sine wave

Assume that the response of a nonlinear system to a sinusoid can be written as follows:

$$y\left(t\right)=\sum_{k=0}^{\infty}y_k\cos\left(k\omega_0 t\right)=y_0+y_1\cos\omega_0 t+y_2\cos 2\omega_0 t+\ldots \tag{3.5}$$

The $K^{\mathrm{TH}}$ HARMONIC DISTORTION (in %) is defined as the ratio of the amplitude of the $k^{\mathrm{th}}$ harmonic (the term located at frequency $k\omega_0$) and the amplitude of the fundamental tone located at $\omega_0$:

$$HD_k = \frac{|y_k|}{|y_1|} \qquad k = 2, 3, \ldots \tag{3.6}$$

If the system is described by 3.1 and the excitation signal $u\left(t\right) = A_1 \cos\left(\omega_0 t\right)$, this expression for $HD_2$ becomes:

$$HD_2 = \frac{A_1 K_2}{2K_1}$$

The TOTAL HARMONIC DISTORTION (in %) is defined by the following relation:

$$THD = \sqrt{\sum_{k=2}^{\infty} HD_k^2} \tag{3.7}$$

Consider an input signal $U(\omega)$ which is a sinewave that results in a fundamental response with amplitude $Y(\omega_0) = |Y_1|$ located at a frequency $\omega_0$. The rest of the Fourier spectrum $Y\left(f\right)$ contains other frequency components that can be harmonically related with the excitation or not. The TOTAL DISTORTION is then defined as:

$$TD = \sqrt{\frac{\int_0^{\infty} \left|Y\left(f\right)\right|^2 df - \left|Y_1\right|^2}{\left|Y_1\right|^2}} \tag{3.8}$$

These equations will be used in the remainder of the text to assess the total distortion of the measured signals.

### 3.1.4 1dB expansion or compression points

When a linear system is excited with an input signal whose amplitude is gradually increased starting from zero, it can be expected that there will be very little influence of the nonlinearity for the very low amplitudes. The key issue here is to know what a low amplitude is for a certain considered system. To quantify the amplitude level at which the system becomes significantly nonlinear, a number of figures of merit have been defined. Here we will take a look at the 1dB compression point and the intercept points.

For the linear time invariant systems it is quite easy to understand that the output power and the input power are linearly related:

$$\begin{aligned}
y &= K_1 \cdot u \qquad \text{taken from 3.1} \\
P_y &= y^2 \\
&= \left(K_1 u\right)^2 \\
&= K_1^2 u^2 \\
P_y &= K_1^2 P_u = \tilde{K_1} P_u
\end{aligned} \tag{3.9}$$

For a nonlinear system the response will gradually start to deviate from this linear law when the power of the input signal is increased. When the output

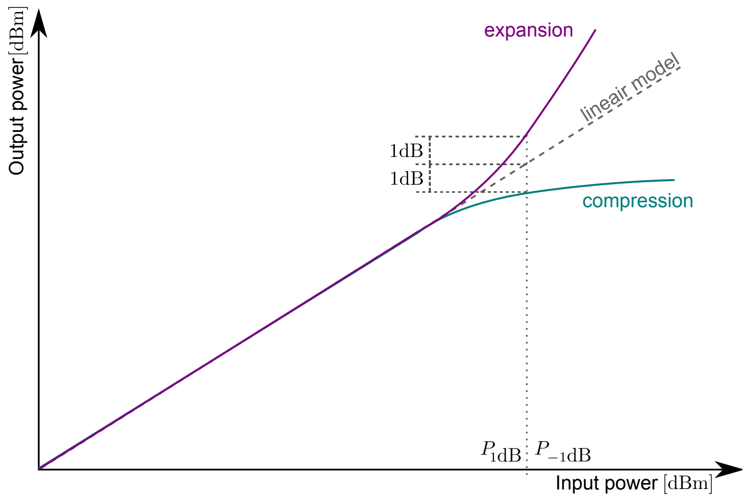


Figure 3.5: Compression and expansion with the 1dB points.

power increases slower than predicted by the linear law 3.9, COMPRESSION occurs. If the output power increases faster than the predicted law, the nonlinear behaviour is called EXPANSION.

Since the compression or expansion behaviour increases gradually, a certain level of deviation has to be set in order to condense the characteristic to a single number. Often, a 1dB level is selected and the corresponding points are called the 1dB compression point ($P_{-1\mathrm{dB}}$) for a system that compresses or the 1dB expansion point ($P_{+1\mathrm{dB}}$) for a system that expands the amplitude.

In Figure 3.5 the linear law as taken from 3.9 is represented by a dashed line. Around it, a system that exhibits compression and one that exhibits expansion are shown on log-log amplitude in versus amplitude out scale. The 1dB compression and expansion points are then easily obtained as the input powers where the difference between the nonlinear and the linear characteristic reaches 1 dB (dotted line).

### 3.1.5 Response of a system

During this lab assignment a system response will be measured. The system contains many potential sources of nonlinearity, so the response is often more complex than the examples shown in the graphs above. Note also that this response will not only depend on the system itself, but also on the impedances that are presented to the amplifier and the frequency at which it is excited.

#### 3.1.5.1 Evaluating the linear and the nonlinear response

We know that most real systems will exhibit harmonic distortion. When excited with a pure sinewave at frequency $f_0$ the output will contain energy at the harmonics of this frequency, namely at $(0,\ f_0,\ 2f_0,\ 3f_0,\ ...)$.

- Consider the nonlinear static system whose response is given by: $y = x - \frac{1}{2}x^3 - \frac{1}{4}x^4$. What are the frequencies at which energy will appear at the output when the input is excited by the sum of 2 sinewaves, one at frequency 4 and one at frequency 11? ♢

## 3.2 Time to go practical!

In this lab assignment all measurements need to be taken at a sampling frequency of 10 kHz and records that contain 4096 data are acquired. Remember that real systems are dynamic, hence they are bound to transient behaviour and this has to be removed before the processing happens. Therefore, the measurements are repeated several times (with the generator left on) to allow that the transients damp. Check that the transients are indeed gone before you start the data processing!

1. Design and calculate an excitation sinewave with a frequency of 77 Hz and an amplitude of 1V. Measure the signals that are present at the input and the output of the DUT. What do you observe? How can you remove the unexpected behaviour? Give a list of possible solutions and a measurement that shows that the proposed solution does indeed work. ♢

2. Remeasure the DUT with a sinewave that has a frequency of approximately 77 Hz. Use approximately 10 different amplitudes that are spaced properly to allow a determination of the compression and intercept points (from 100mV to 5V ampltitude, logarithmic spacing). Do not forget to save the values of these amplitudes, as you will need them afterwards.

   (a) What are the frequencies where you expect that distortion will appear? Are they all present in reality? Can you explain this? ♢

   (b) Determine the harmonic distortion $HD_k$ and the total harmonic distortion $THD$. Check the values you obtain using the DSA contained in ELVIS (only for 2nd and 3rd harmonic). ♢

   (c) Determine the different $K_k$-values (see equation 3.1) starting from the measured data (only for 2nd and 3rd harmonic). ♢

   (d) Plot the measurements on an amplitude in -amplitude out plot. Do you see compression or expansion? ♢

3. Measure the 1 dB compression or expansion point using the same sinewave as above. Adjust the amplitude until the 1 dB point is reached.

## 3.3 Extending to Multisines

The second part of the lab consists of measuring two systems using an odd random phase multisine as excitation signal. First start by measuring one of the systems.

- Is your result noisy? If so, try using the methods of Lab 2 to average out the noise. Does applying the averaging methods help to improve your results? ♢
- Analyse the obtained results. How does the system behave? ♢
- Now try to measure with 3 different power levels. What is the impact of increasing the power of the input signal on the distortions? ♢
- How does the system behave when the power of the input signal has been increased? ♢
- Now repeat the same procedure for the second system and compare the results. ♢

# Additional Information

## Intercept points

The Intercept Points of the $k^{\text{th}}$ order are also expressed as a power, often specified in dBm. These new figures of merit are obtained as the intersection of the extrapolation of the evolution of the linear term of the response at the fundamental frequency (the one used in the 1dB compression points) and the extrapolated value of the harmonic distortion at the $k^{\text{th}}$ harmonic as is seen on Figure 3.6. The power at which the intersection occurs can be referred to the input (Input Intercept Point: $IIP_k$) or to the output (Output Intercept Point: $OIP_k$).

The most commonly used intercept points are the second order one $IP_2$ or $SOI$ and the third order one labelled $IP_3$ or $TOI$. Note that neither the $IP_2$ nor the $IP_3$ can be measured directly without destroying the DUT. It will therefore be mandatory to obtain these points by linear regression of the two curves used, as is illustrated on Figure 3.6.

## Go Practical!

1. For a single tone using the previouse measurements determine the second and third order intercept points. ♢

2. Replace the sinewave with a two-tone, i.e. a sum of two sinewaves that are closely spaced. Use approximately 1kHz and 1.1kHz for the frequencies and use the same amplitudes as the ones used in the previous experiment.

   (a) At which frequencies do you expect to find energy in the output signal? Compare expected and measured behaviour, and explain the differences if any. ♢

   (b) Determine $IIP_2$, $OIP_2$, $IIP_3$ and $OIP_3$ based on the measurements. ♢

For any questions please email and arrange a date and time with one of the lab assistants.

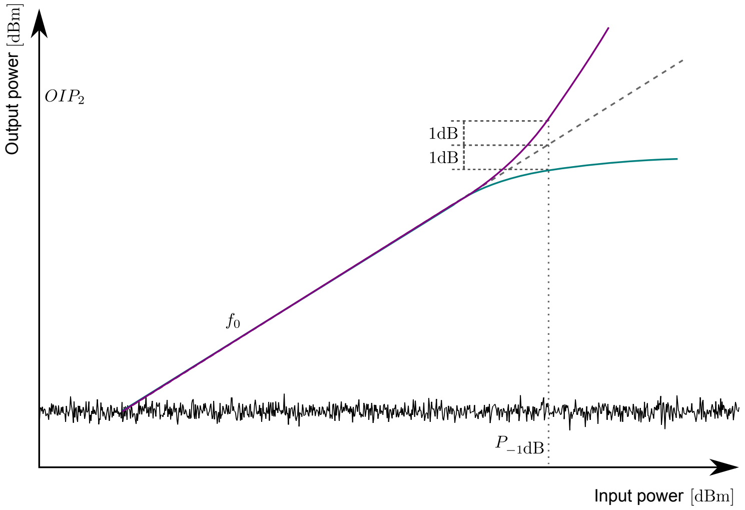


Figure 3.6: Amplitude-amplitude plot of the harmonic response of a nonlinear system excited by a sinewave.

| | |
|---|---|
| Maarten Schoukens | `Maarten.Schoukens@vub.ac.be` |
| Mark Vaes | `Mark.Vaes@vub.ac.be` |
| Maral Zyari | `Maral.Zyari@vub.ac.be` |
| Dieter Verbeke | `Dieter.Verbeke@vub.ac.be` |

# Chapter 4

# Model selection

## 4.1 Lab objectives

The goal of this lab is to illustrate:

- the selection of a “good” model (number of model parameters),
- the effect of the number of model parameters on the value of the cost functions and the quality of the model.
- that the optimal model order can depend on the signal-to-noise ratio of the data and the number of data points.

## 4.2 Introduction

In this lab we will be estimating a finite impulse response of two systems, both, in the presence of output noise. First, we will perform a simulation on a discrete-time system to generate the input ($u_0(t)$) and output ($y(t)$) time samples (Section 4.3) and then we will perform experimental measurements on a bandpass analog filter (Section 4.4).

The general simulation and experimental layout is shown in Figure 4.1.

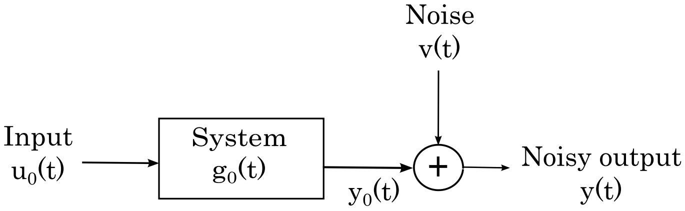


Figure 4.1: Measurement setup.

The input $u_0(t)$ is computer generated and filtered through the system to obtain a noise free output $y_0(t)$ to which noise ($v(t)$) is added giving the final noisy output $y(t)$. Through a convolution summation, the input and noisy output can be related via the system's impulse response ($g_0(t)$) as given below:

$$\begin{aligned} y_0(t) &= \sum_{\tau=0}^{\infty} u_0(t-\tau)g_0(\tau) \quad t = 0, 1, 2, \ldots \\ y(t) &= y_0(t) + v(t) \end{aligned} \tag{4.1}$$

The systems we will be working with have in theory an impulse response that is infinitely long (IIR-system), which is why the summation in equation (4.1) goes to infinity.

We will approximate this with a *finite impulse response* (FIR-system) of length $I$ (the model order) and investigate how an optimum value for $I$ can be obtained. Therefore, our FIR model is as given below:

$$\hat{y}(t) = \sum_{\tau=0}^{I} u_0(t-\tau)\hat{g}(\tau) \quad t = 0, 1, 2, \ldots \tag{4.2}$$

Equation (4.2) is now the model that we will fit on to the input and output data. Notice that the summation goes up to $I$ and this acts as the model order. The impulse response coefficients, $\hat{g}(\tau)$, as the model parameters to be estimated. Equation (4.2) can be written in a compact matrix form (**work this out yourself**) as:

$$\hat{Y} = H\theta \tag{4.3}$$

| | | | |
|---|---|---|---|
| | $\hat{Y}$: | The vector of *model output* values | size $N \times 1$ |
| With: | $H$: | The *regressor* or *observation* matrix | size $N \times (I+1)$ |
| | $\theta$: | The parameter vector. $\theta = [\hat{g}(0), \hat{g}(1), \ldots, \hat{g}(I)]'$ | size $(I+1) \times 1$ |

Models that can be written as in equation (4.3) are said to be *linear-in-the-parameters*.

Starting from the input-output measurements, an estimate of the impulse response coefficients can be obtained by minimizing the following least squares cost function:

$$V(\theta, N) = \frac{1}{N}\sum_{t=0}^{N-1} |y(t) - \hat{y}(t)|^2 \tag{4.4}$$

Since this model is linear-in-the-parameters, the optimum parameter vector ($\hat{\theta}$) that minimises the cost function can be calculated explicitly as follows:

$$\hat{\theta} = (H^T H)^{-1} H^T Y \tag{4.5}$$

In equation (4.5), $Y$ is the vector (size $N \times 1$) of measured *noisy output* samples, and in practice a numerically stable method is used to evaluate equation (4.5), (see Appendix 2.C of the lecture notes).

The quality of the estimates $\hat{\theta}$ will strongly depend on the model order $I$, the length of the data $N$, and the signal-to-noise-ratio of the data. Increasing the order $I$ will improve the quality of the approximation at a cost of using more parameters to describe the system. This will increase the uncertainty on the estimated model. In order to find the "best" model, a balance should be made between the model errors and the noise errors. To achieve this, we will use the robustified AIC cost function given as:

$$V_{AIC} = V(\hat{\theta}, N)\left(1 + 2\frac{I+1}{N}\right) \tag{4.6}$$

Remarks:

1. In practice, if starting the measurements at $t = 0$, the inputs before $t = 0$ are unknown (while they also appear in $H$). To circumvent this problem, $N_t$ additional samples (called *transient points*) are measured before $t = 0$. In total, $N_t + N$ points are measured, from which only the last $N$ output points are used to construct the matrix equation (4.3) and the $N_t$ first output samples are discarded.

2. To assess the model quality and the effectiveness of the AIC method, in this lab, a set of *validation data* is measured for comparison purposes. These validation data are not used for estimation of the parameters, but for validating the model. The data used to estimate the model are called *estimation data*. The lengths of these data sets are denoted as $N_t + N_v$ and $N_t + N_e$, respectively.

## 4.3 Simulation

The transfer function ($G_0(z)$) of the discrete-time system we will be simulating is:

$$G_0(z) = \frac{\sum_{k=0}^{n_b} b_k z^{-k}}{\sum_{k=0}^{n_a} a_k z^{-k}}, \tag{4.7}$$

with $a = [a_0, a_1, ..., a_{n_a}]$ and similarly for $b$ the filter coefficients and they are:

$a = [2.765e\text{-}001\ \text{-}3.464e\text{-}001\ 6.141e\text{-}001\ \text{-}4.371e\text{-}001\ 4.410e\text{-}001\ \text{-}1.645e\text{-}001\ 9.9619e\text{-}002]$
$b = [8.002e\text{-}004\ 1.9427e\text{-}002\ \text{-}4.5489e\text{-}002\ 1.245e\text{-}002\ 3.050e\text{-}002\ \text{-}1.928e\text{-}002\ 1.902e\text{-}003]$

We will generate the input samples and noise samples via the Matlab `randn` function[1] and will generate two input-output data sets, one for model estimation and the other for model validation.

Note: Make sure that in your Matlab code, signal lengths, standard deviations and model order are set as variables and **not hard coded**, so that you can examine different settings easily and comment your code sufficiently for later reference.

[1] type »`help randn` to get help on `randn` function, or similarly on any other function

### Task 1: data generation

- Using the **randn** function generate two input signals $u_e(t)$ (estimation) and $u_v(t)$ (validation), of lengths 2000 and 11000, respectively. These lengths include $N_t = 1000$ transient points. The standard deviation of both signals is $\sigma_u = 1$.
- Similarly, using the **randn** function generate two noise signals $v_e(t)$ and $v_v(t)$ of the same lengths but with a standard deviation of $\sigma_v = 0.5$.
- Filter both $u_e(t)$ and $u_v(t)$ via the `filter` function to generate the noise free outputs $y_{0e}(t)$ and $y_{0v}(t)$. $y_{0e} = \texttt{filter}(b, a, u_e)$, similarly for $y_{0v}$.
- Add the corresponding noise signals to $y_{0e}(t)$ and $y_{0v}(t)$ to obtain the final noisy outputs $y_e(t)$ and $y_v(t)$. $y_e = y_{0e} + v_e$, similarly for $y_v(t)$.

Note: In practical problems the validation set is selected (much) smaller than the estimation set. Here we selected many more validation points than estimation points in order to illustrate that the method works fine.

### Task 2: impulse response estimation

Using the linear least squares solution (equation 4.5), we will now estimate the parameter vector (impulse response coefficients) with varying model orders, $I = 0, 1, \ldots, 100$.

- Set $I$ in your Matlab code as a variable so that it can be changed as required.
- For a given $I$, using the estimation input $u_e(t)$ set up the observation matrix $H$[2] and using the noisy output $y_e(t)$ set up the output vector $Y$. Don't forget to take the transient points into account.
- Obtain the linear least squares solution $\hat{\theta}$ (equation 4.5). To avoid numerical ill-conditioning problems, it is recommended to use the Matlab backslash operator (\) to obtain a least squares estimate $\hat{\theta} = H \setminus Y$. This is an estimate of the optimum impulse response coefficients for the given model order $I$.

### Task 3: model selection

Having obtained an estimate of the model parameters ($\hat{\theta}$), we can now simulate outputs for both inputs $u_e(t)$ and $u_v(t)$ and evaluate the corresponding cost functions. This will give an indication on the quality of our model.

- Using the `filter` function obtain model outputs $\hat{y}_e(t)$ and $\hat{y}_v(t)$ as follows: $\hat{y}_e(t) = \texttt{filter}(\hat{\theta}, 1, u_e)$ and $\hat{y}_v(t) = \texttt{filter}(\hat{\theta}, 1, u_v)$. With these commands, you perform the convolution in (4.2).

[2] Hint: You can construct the observation matrix $H$ for $I = 100$ and when you need $H$ for some model order e.g. $I = 10$, just use the first 10 columns of the full matrix.

- For both the estimation and validation data, evaluate the cost function (4.4):
  $V_e(\hat{\theta}, N_e) = \frac{1}{N_e}\sum_{t=0}^{N_e-1} |y_e(t) - \hat{y}_e(t)|^2$
  $V_v(\hat{\theta}, N_v) = \frac{1}{N_v}\sum_{t=0}^{N_v-1} |y_v(t) - \hat{y}_v(t)|^2$
- Also obtain the AIC cost function which is based on the estimation cost function.
  $V_{AIC} = V_e(\hat{\theta}, N_e)\left(1 + 2\frac{I}{N_e}\right)$
- Now evaluate the three cost functions for an increased model order. In your Matlab code, this should be done via a `for` loop where $I$ goes from 0 to 100.
- Normalise $V_e$, $V_v$ and $V_{AIC}$ by dividing them with the noise variance $\sigma_v^2$ and plot the normalised cost functions against the model order.

### Task 4: reduced noise level

The above analysis has been performed for a noise standard deviation of $\sigma_v = 0.5$. We will reduce the noise level by setting $\sigma_v = 0.1$and $\sigma_v = 0$ to investigate how the cost functions behave and the choice of our optimum model order.

Repeat tasks 1, 2 and 3 for $\sigma_v = 0.1$ and $\sigma_v = 0$.

### Task 5: analysis

We now have three cost functions ($V_e$, $V_v$, $V_{AIC}$) which can be used to evaluate the goodness of our model.

- Which of the cost functions decrease monotonically with the model order? Is this expected?
- Which of the cost functions decrease in value and then begin to increase above a certain model order? This indicates that there is an optimum model order. Why might this be?
- In real life applications, we may at times not have a validation data set. Which cost function can we then use to evaluate an optimum model order?
- What was the effect of reducing the noise level on the optimum model order?

## 4.4 Experimental measurement

We will now perform experimental measurements on a passive analogue bandpass filter[3]. The central bandpass frequency can be adjusted and the data acquisition has a sampling frequency of 8kHz.

[3] This experiment will be done with the help of a lab assistant

### Task 6: data generation and acquisition

Similar to the simulation task the experimental task is to acquire an estimation ($y_e(t)$ and $u_e(t)$) and validation ($y_v(t)$ and $u_v(t)$) data set and then estimate and validate an optimum finite impulse response model for the bandpass filter. The input will again be a random input signal generated using the `randn` Matlab function. You will do the Matlab coding on the computer by the measuring set-up.

- Using the `randn` function generate two (zero mean) input signals $u_e(t)$ and $u_v(t)$, of lengths 2000 and 11000, respectively. These lengths include $N_t = 1000$ transient points. Set the standard deviation of both signals as $\sigma_u = 1$ (you might have to increase the $\sigma_u$ value if the noise level is too high).
- Save each input signal as a column vector on the local drive with some file name, e.g. input1.csv and input2.csv. To create a csv (*comma separated value*) file use the following code: `save(‘input1.csv’,‘u’,‘-ascii’)` where the input signal is variable $u$.
- Apply $u_e(t)$ and $u_v(t)$ on the bandpass filter with some output noise and acquire the corresponding noisy output signals $y_e(t)$ and $y_v(t)$.
- Adjust (increase or decrease) the noise level on the noise generator and measure a new estimation and validation data set for this noise setting.

### Task 7: data processing

Repeat tasks 2, 3 and 5 for the acquired data. The normalisation of the cost functions need not be applied since the noise standard deviation is unknown.

For any questions please email and arrange a date and time with one of the lab assistants.

| | |
|---|---|
| Maarten Schoukens | `Maarten.Schoukens@vub.ac.be` |
| Mark Vaes | `Mark.Vaes@vub.ac.be` |
| Maral Zyari | `Maral.Zyari@vub.ac.be` |
| Dieter Verbeke | `Dieter.Verbeke@vub.ac.be` |

# Chapter 5

# Estimation of transfer functions

## 5.1 Objectives

The goal of the lab is to illustrate:

- how parameters of a transfer function can be estimated
- when a nonlinear optimisation routine is needed
- the performance of three optimisation routines

## 5.2 Introduction

In this lab we will estimate the parameters of a transfer function defined in the s-domain (continuous time). Unlike in Lab 1, where the model (convolution summation) was linear in the parameters, the transfer function model is not. As such, a nonlinear optimisation routine is required. In this lab you will not perform any measurements but you will be provided with input and output time data from the analogue band pass filter which you used in Lab 1.

A transfer function is of the form:

$$G(s) = \frac{\sum_{i=0}^{n_b} b_i s^i}{1 + \sum_{i=1}^{n_a} a_i s^i} = \frac{B(s)}{A(s)} = \frac{B(s)}{1 + A'(s)} \tag{5.1}$$

In equation (5.1), $B(s)$ and $A'(s)$ are polynomials in $s$ with degree $n_b$ and $n_a$ respectively, and the coefficients are the parameters to be estimated.

$$\begin{aligned} B(s) &= b_{n_b} s^{n_b} + \cdots + b_0 \\ A'(s) &= a_{n_a} s^{n_a} + \cdots + a_1 s \end{aligned}$$

The optimum coefficients are obtained by minimising the following quadratic cost function:

$$V(\theta) = \sum_{k=1}^{F} \frac{\|G_m(s_k) - G(s_k,\theta)\|^2}{\hat{\sigma}_G^2} \tag{5.2}$$

| | | | |
|---|---|---|---|
| | $G_m(s_k)$: | Frequency response at $s_k$, measured from the data | $G_m(s_k) \in \mathbb{C}$. |
| | $G(s_k,\theta)$: | Model response at $s_k$ given the parameters $\theta$ | $G(s_k,\theta) \in \mathbb{C}$ |
| where: | $\theta$: | Parameter vector. $\theta = [b_{n_b}, \ldots, b_0, a_{n_a}, \ldots, a_1]^T$ | $(n_a + n_b + 1) \times 1$ |
| | $F$: | Number of frequency measurements | |
| | $s_k$: | Discretised Laplace variable. $s_k = 2j\pi f_k$, ($f_k$ in Hz) | $s_k \in \mathbb{C}$ |

If the model could be written as $G(s_k,\theta) = H(s_k)\theta$, where $H(s_k)$ is some regressor row vector, it would be a linear-in-the-parameters model and the optimum parameter vector is then the linear least squares solution. Since this is not the case, we can either modify the cost function and model or use a nonlinear optimisation routine. These methods are explained next.

## 5.2.1 Levy method

### 5.2.1.1 Levy cost function

Note that the cost function we need to minimise is:

$$V(\theta) = \sum_{k=1}^{F} \left\| G_m(s_k) - \frac{B(s_k,\theta)}{A(s_k,\theta)} \right\|^2 \tag{5.3}$$

The Levy method redefines the cost function such that the initial nonlinear-in-the -parameters problem becomes linear-in-the-parameters.

$$V_{Levy}(\theta) = \sum_{k=1}^{F} \|G_m(s_k)A(s_k,\theta) - B(s_k,\theta)\|^2 \tag{5.4}$$

with $A(s_k,\theta) = 1 + A'(s_k,\theta)$ this becomes:

$$\begin{aligned}
V_{Levy}(\theta) &= \sum_{k=1}^{F} \|G_m(s_k) + G_m(s_k)A'(s_k,\theta) - B(s_k,\theta)\|^2 \\
V_{Levy}(\theta) &= \sum_{k=1}^{F} \|G_m(s_k) - (-G_m(s_k)A'(s_k,\theta) + B(s_k,\theta))\|^2 \\
V_{Levy}(\theta) &= \sum_{k=1}^{F} \|G_m(s_k) - G_{Levy}(s_k,\theta)\|^2 \qquad (5.5)
\end{aligned}$$

By modifying the cost function, the Levy method leads to a new model $G_{Levy}(s_k,\theta)$ where the parameters of the transfer function appear linearly in it.

$$\begin{aligned}
G_{Levy}(s_k,\theta) &= -G_m(s_k)A'(s_k,\theta) + B(s_k,\theta) \\
G_{Levy}(s_k,\theta) &= H_{Levy}(s_k)\theta \qquad (5.6)
\end{aligned}$$

In equation (5.6) $H_{Levy}(s_k)$ is a regressor vector of dimension $1 \times (n_a + n_b + 1)$. If we have $F$ frequency points, equation (5.6) can be written in matrix form as:

$$G_{Levy} = H_{Levy}\theta \tag{5.7}$$

|  |  |  |  |
|---|---|---|---|
| where: | $G_{Levy}$: | Vector of model outputs | size $F \times 1$ |
|  | $H_{Levy}$: | Levy regressor matrix | size $F \times (n_a + n_b + 1)$ |

The $H_{Levy}$ matrix is composed of the measured frequency response $G_m(s_k)$ and the discretised Laplace variable $s_k$. **Work out the Levy regressor matrix in terms of $G_m(s_k)$ and $s_k$ for model orders $n_a$ and $n_b$.**

#### 5.2.1.2 Levy cost function minimisation

Since we have a linear-in-the-parameters model, the optimum parameter vector minimising equation (5.5) is the linear least squares solution. Let $G_m$ be the *measured vector* of frequency responses $G_m \in \mathbb{C}^{F\times 1}$. Because the Levy regressor and the measured vector are complex, direct use of the '\' operator ($\hat{\theta} = H_{levy} \setminus G_m$) will give complex optimum parameters. To obtain real valued parameters we split the measured vector and regressor matrix into real and imaginary parts and then concatenate it.

$$G_{m,c} = \begin{bmatrix} \mathrm{Re}\{G_m\} \\ \mathrm{Im}\{G_m\} \end{bmatrix} \quad H_{Levy,c} = \begin{bmatrix} \mathrm{Re}\{H_{Levy}\} \\ \mathrm{Im}\{H_{Levy}\} \end{bmatrix} \tag{5.8}$$

$G_{m,c}$ and $H_{Levy,c}$ are now our new measured vector and regressor matrix, all elements are real valued and the $c$ in the superscript is to indicate concatenation. The estimated parameter vector will still remain of size $(n_a + n_b + 1) \times 1$ but real valued. **Can you explain why this is**.

The Levy cost function is only an approximation of the original cost function (equation 5.2 or 5.3). Because of the multiplication with the denominator $(A(s_k, \theta))$, the high frequency errors gets more weight compared to the low frequency errors. This is because $|A(s_k, \theta)|$ becomes large at high frequencies. This overweighting of the high frequency errors will, in many cases, result in poor estimates. This effect can be reduced with the method of Sanathanan.

### 5.2.2 Sanathanan method

#### 5.2.2.1 Sanathanan cost function

We want to obtain a method which compensates for the weighting problems of the Levy method. To do so, the Levy cost function is modified and an iterative procedure known as the Sanathanan method is obtained:

$$V_{San}(\theta_l) = \sum_{k=1}^{F} \frac{1}{\left\|A(s_k, \hat{\theta}_{l-1})\right\|^2} \left\|G_m(s_k)A(s_k, \theta_l) - B(s_k, \theta_l)\right\|^2$$

$$V_{San}(\theta_l) = \sum_{k=1}^{F} \left\| \frac{G_m(s_k)}{A(s_k, \hat{\theta}_{l-1})} - \frac{G_{Levy}(s_k, \theta_l)}{A(s_k, \hat{\theta}_{l-1})} \right\|^2 \tag{5.9}$$

where $l$ is the iteration index. Note that the Sanathanan cost function error is the Levy cost function error (equation 5.5) divided by the denominator $A(s_k, \hat{\theta}_{l-1})$. $A(s_k, \hat{\theta}_{l-1})$ is the polynomial evaluation for the previous parameter estimate $\hat{\theta}_{l-1}$. In equation (5.9), $G_m(s_k)/A(s_k, \hat{\theta}_{l-1})$ is now our "new measured vector" and $G_{Levy}(s_k, \theta_l)/A(s_k, \hat{\theta}_{l-1}) = G_l^{San}(s_k, \theta_l)$.

$$\begin{aligned} G_l^{San}(s_k, \theta_l) &= \frac{-G_m(s_k)A'(s_k, \theta_l)}{A(s_k, \hat{\theta}_{l-1})} + \frac{B(s_k, \theta)}{A(s_k, \hat{\theta}_{l-1})} \\ G_l^{San}(s_k, \theta_l) &= H_l^{San}(s_k)\theta_l \end{aligned} \tag{5.10}$$

Note that we still have a linear-in-the-parameters-model (equation 5.10), with $F$ frequency points equation (5.10) can be written in matrix form:

$$G_l^{San} = H_l^{San}\theta_l \tag{5.11}$$

#### 5.2.2.2 Sanathanan cost function minimisation

Here $H_l^{San}$ is the regressor matrix of size $F \times (n_a + n_b + 1)$ at index $l$. It is similar to the Levy regressor matrix except that the elements are divided by $A(s_k, \hat{\theta}_{l-1})$. Again, to obtain real valued estimates, the real and imaginary parts of $H_l^{San}$ and the new measured vector need to be split. **Work out the Sanathanan regressor matrix in terms of $G_m(s_k)$, $s_k$ and $A(s_k, \hat{\theta}_{l-1})$ and the new measured vector in terms of $G_m(s_k)$ and $A(s_k, \hat{\theta}_{l-1})$.**

Therefore, by defining a maximum number of iterations (let this be $L$) and using $F$ frequency points, we solve a linear least squares problem and the new parameters provide the denominator value for the next iteration. To start off the iteration, the parameters estimated from the Levy method can be used for $A(s_k, \hat{\theta}_0)$.

In many cases, an improvement is obtained compared to the Levy method, but it is possible that the Sanathanan method doesn't arrive at the original cost function. This makes it inevitable to use a nonlinear optimisation routine to solve the original cost function and we will use the Gauss-Newton method.

### 5.2.3 Gauss-Newton

The Gauss-Newton (GN) method is an algorithm for solving nonlinear least squares problems. The cost function we started with is:

$$V(\theta) = \sum_{k=1}^{F} \|G_m(s_k) - G(s_k, \theta)\|^2 = \sum_{k=1}^{F} \|E(s_k, \theta)\|^2 \tag{5.12}$$

where $E(s_k, \theta) = G_m(s_k) - G(s_k, \theta)$ and $G(s_k, \theta) = B(s_k, \theta)/A(s_k, \theta)$. The GN algorithm is also an iterative procedure and requires an initial parameter value to start. At each iteration the next parameter is calculated as:

$$\hat{\theta}_{l+1} = \hat{\theta}_l + \Delta\hat{\theta}_l \tag{5.13}$$

and the parameter update ($\Delta\hat{\theta}_l$) is obtained by solving a linear least squares problem as shown below:

$$\begin{aligned}(J_l^T J_l)\Delta\hat{\theta}_l &= J_l^T E_l \\ \Delta\hat{\theta}_l &= (J_l^T J_l)^{-1} J_l^T E_l \end{aligned} \tag{5.14}$$

In equation (5.14), $E_l$ is a vector of the errors $E_l = [E(s_1, \hat{\theta}_l), \ldots, E(s_F, \hat{\theta}_l)]^T$ (size $F \times 1$) and $J_l$ is the *Jacobian matrix* (size $F \times (n_a + n_b + 1)$). The Jacobian is a matrix of the partial derivatives of the model with respect to each parameter.

$$J_l = \left.\frac{\partial G(s_k, \theta)}{\partial \theta}\right|_{\theta = \hat{\theta}_l}$$

$$J_l = \begin{pmatrix} \frac{\partial G(s_1)}{db_{n_b}} & \cdots & \frac{\partial G(s_1)}{db_0} & \frac{\partial G(s_1)}{da_{n_a}} & \cdots & \frac{\partial G(s_1)}{da_1} \\ \vdots & & \vdots & \vdots & & \vdots \\ \frac{\partial G(s_F)}{db_{n_b}} & \cdots & \frac{\partial G(s_F)}{db_0} & \frac{\partial G(s_F)}{da_{n_a}} & \cdots & \frac{\partial G(s_F)}{da_1} \end{pmatrix} \tag{5.15}$$

**Show that the partial derivative of $G(s_k, \theta)$ with respect to the numerator and denominator coefficients is of the following form.**

$$\frac{\partial G(s_k, \theta)}{db_i} = \frac{s_k^i}{1 + \sum_{p=1}^{n_a} a_p s_k^p} \qquad \text{with } i = 0, \ldots, n_b \tag{5.16}$$

$$\frac{\partial G(s_k, \theta)}{da_i} = \frac{-\sum_{p=0}^{n_b} b_p s_k^p}{\left(1 + \sum_{p=1}^{n_a} a_p s_k^p\right)^2} s_k^i \qquad \text{with } i = 1, \ldots, n_a \tag{5.17}$$

Therefore, for a parameter value $\hat{\theta}_l$ we can evaluate equations (5.16) and (5.17) and construct the Jacobian matrix $J_l$ and having a model, $G(s_k, \hat{\theta}_l)$, also construct the error vector $E_l$ for iteration $l$. The parameter update, via the Matlab '\' operator is then:

$$\Delta\hat{\theta}_l = J_l \setminus E_l \tag{5.18}$$

Note: Once $J_l$ and $E_l$ are constructed, split and concatenate their real and imaginary parts to avoid getting complex update parameters.

Similar to the Sanathanan method, we can have a predefined number of iterations $L$ for terminating the GN method. **Can you think of other stopping criterion?**[1] 1]Hint: Perhaps define a threshold for the cost function or its relative variation in successive iterations; at each iteration $E_l^H E_l$ is the value of the cost function $V(\hat{\theta}_l)$. $E_l^H$ is the Hermitian or conjugate transpose.

[1] [

## 5.3 Implementation

We will now implement the three (Levy, Sanathanan and Gauss-Newton) optimisation routines on the data provided (this is a `.mat` Matlab file). This consists of the input and output time data arranged as two matrices `xdata` (the input) and `ydata` (output) of size 1024 × 32. Each column is one period, so we have 32 periods with $N = 1024$ samples per period. The input signal is a swept sine (plot one period and visualise) and the response when applied to the band pass filter is the output sampled at $f_s = 8$kHz.

Make sure that in your Matlab code, model orders, harmonic selection and normalising factor (described later) are set as variables and **not hard coded**, so that you can examine different settings easily and comment your code sufficiently for later reference.

### 5.3.1 Estimating the frequency response

We first need to get an estimate of the frequency response ($G_m(s_k)$) to which we can fit the transfer function model.

- Average the input and output data across periods to reduce any measurement noise, e.g. `xm=mean(xdata,2)` and similarly for the output.

- Compute the Fast Fourier Transform (FFT) of the averaged signals, `Xm=fft(xm)` and similarly for the averaged output `ym`. Note: you could also have taken the FFT of the matrices first, `Xdata=fft(xdata)`, and then averaged over the periods `Xm=mean(Xdata,2)`. **Why does reversing the two operations make no difference?**

- We will do the fitting on 114 frequency points or harmonics, $F = 114$. Define a column vector called `harm=[15:128]'`. These are the excited harmonics in the signal.

- Note: When we perform FFT's, the value at the first index is the dc value or harmonic zero, so the Matlab index for a given harmonic is `harm+1`. Obtain the frequency response estimate as `Gm=Ym(harm+1)./Xm(harm+1)`.

- Create the frequency vector `freq=harm*fs/N`, where $f_s = 8$kHz, $N = 1024$.

- A plot of the magnitude of $G_m(s_k)$ (in dB) and its phase (in rad) against frequency is shown in Figure 5.1, `plot(freq,db(Gm))`, `plot(freq,unwrap(angle(Gm)))`.

### 5.3.2 Matrix conditioning

A point to consider is the condition number[2]1]Condition number of a matrix is the ratio of its maximum and minimum singular values. of the regressor and Jacobian matrix. A condition number of 1 means the least squares method will find

[2] [

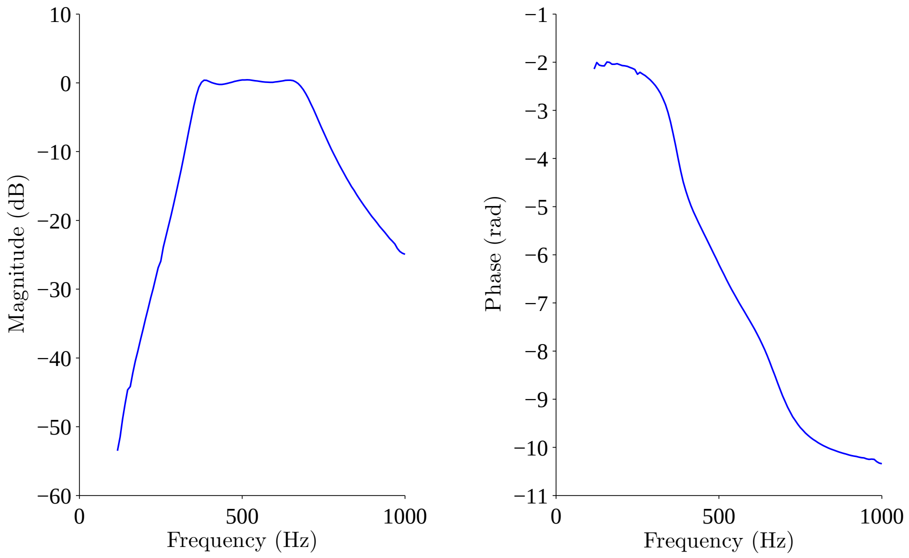


Figure 5.1: Estimated magnitude and phase response of the bandpass filter, $G_m(s_k)$

.

the optimum parameters with an arbitrary precision, whereas a condition number of infinity means the least squares solution is very unreliable. Ill-conditioned matrices are usually handled via scaling or normalising the matrices. We will define a normalised frequency vector to improve ill-conditioning.

- We will use the mean of the smallest and largest frequency as the normalising factor: $\alpha$=`(min(freq)+max(freq))/2`.
- Define the new normalised frequency vector `fnorm=freq/`$\alpha$.
- Now define the Laplace variable as a vector, `snorm=j*2*pi*fnorm`, which will be used in composing the regressor and Jacobian matrices.

### 5.3.3 Levy implementation

In Section 5.2.1.1 you should have worked out the Levy regressor matrix $H_{Levy}$ defined for some given model orders.

- Define the model orders (polynomial degrees) $n_b$ and $n_a$[3]1]Hint: Use $n_b > 4$, $n_a > 5$.
- Using `Gm` and `snorm` setup the Levy regressor matrix, size $F \times (n_a + n_b + 1)$.

[3] [

- Split and concatenate the real and imaginary terms of the measured vector and the Levy regressor matrix, `Gm_c=[real(Gm);imag(Gm)]`, `HLevy_c=[real(HLevy);imag(HLevy)]`.
- Use the Matlab backslash operator (\) to obtain the linear least squares estimate $\hat{\theta}^n$=`HLevy_c\Gm_c`, the superscript $n$ implies that the parameters are based under normalised conditions.
- Record the condition number of the split regressor matrix for later analysis.
  `condLevy=cond(HLevy_c)`.
- Extract the numerator ($\hat{\theta}_b^n = [b_{n_b}, \ldots, b_0]^T$) and denominator ($\hat{\theta}_a^n = [a_{n_a}, \ldots, a_1]^T$) coefficients from the parameter estimate $\hat{\theta}^n$.

We can think of the parameters $\hat{\theta}_b^n$ and $\hat{\theta}_a^n$ as the parameters of a system measured over our normalised frequency interval defined in `fnorm`. But since the actual system was measured over the bandwidth defined in `freq` we need to adapt the estimated parameters.

- Adapt the parameters by dividing each parameter by the normalising factor raised to the corresponding power. $\hat{\theta}_b = [b_{n_b}/\alpha^{n_b}, \ldots, b_0/1]^T$ and $\hat{\theta}_a = [a_{n_a}/\alpha^{n_a}, \ldots, a_1/\alpha]^T$. **Can you explain this adaptation method?**

We will now plot the Levy model and measured frequency response. For this, the Matlab function `freqs` is used. Note: the `freqs`[4]2]Type `help freqs` for more details. function requires the coefficients to be in decreasing order, so make sure your parameters in $\hat{\theta}_b$ and $\hat{\theta}_a$ are in decreasing order.

- Define the angular frequency vector `w=2*pi*freq`.
- Recall that $A(s) = A'(s) + 1$, so include the constant 1 along with $\hat{\theta}_a$ and generate the Levy model output. `GLevy=freqs(`$\hat{\theta}_b$`,[`$\hat{\theta}_a; 1$`],w)`.
- Plot `Gm`, `GLevy` and the error (`Gm-GLevy`), all in dB, against frequency.

### 5.3.4 Sanathanan implementation

Recall that the Sanathanan method is an iterative procedure and we will use the Levy solution to start the iteration. The function `polyval` will be used to evaluate the polynomial $A(s_k, \theta)$ for any value of $s_k$. Similar to `freqs`, the polynomial coefficients must be in decreasing order when using `polyval`.

- Use the *normalised* denominator coefficients from Levy as the initial denominator coefficients for Sanathanan. $\hat{\theta}_{a0}^n = \hat{\theta}_a^n$.
- Evaluate the polynomial $A(s_k, \hat{\theta}_{a0}^n)$ for the values defined in the `snorm` vector.
  `A=polyval([`$\hat{\theta}_{a0}^n; 1$`],snorm)`

[4] [

- Construct the Sanathanan regressor matrix ($H_l^{San}$) and also the new measured vector $G_m(s_k)/A(s_k, \hat{\theta}_0)$ which is `GmSan=Gm./A`.
- Again, split and concatenate the Sanathanan regressor ($H_l^{San}$) and the new measured vector (`GmSan`) into real and imaginary terms and calculate the linear least squares solution $\hat{\theta}_l^n$. Extract the normalised numerator and denominator coefficients from $\hat{\theta}_l^n$ and record the condition number of the split regressor matrix.
- We now have the normalised estimates $\hat{\theta}_{b1}^n$ and $\hat{\theta}_{a1}^n$ from the first iteration $l = 1$. Repeat the above steps using $\hat{\theta}_{a1}^n$ for the next iteration. Perform this in a `for` loop and set the maximum number of iterations ($L$) as a variable so that you can try different settings.

After $L$ iterations we have estimated the normalised parameters via the Sanathanan method. Adapt the parameters with the normalising factor as before and obtain the Sanathanan model output.

- Adapt the parameters: $\hat{\theta}_b = [b_{n_b}/\alpha^{n_b}, \ldots, b_0/1]^T$ and $\hat{\theta}_a = [a_{n_a}/\alpha^{n_a}, \ldots, a_1/\alpha]^T$.
- Evaluate the Sanathanan model: `GSan=freqs(`$\hat{\theta}_b$`,`$[\hat{\theta}_a; 1]$`,w)` and plot `Gm`, `GSan` and the error (`Gm-GSan`) in dB against frequency.

### 5.3.5 Gauss-Newton implementation

To initiate the GN method we need initial values for the numerator and denominator coefficients. Similar to initiating the Sanathanan method, use the normalised parameters obtained from the Levy method to start.

- Use the *normalised* coefficients from Levy as the initial coefficients for the GN method. $\hat{\theta}_{b0}^n = \hat{\theta}_b^n$ and $\hat{\theta}_{a0}^n = \hat{\theta}_a^n$. Let $\hat{\theta}_0^n = [\hat{\theta}_{b0}^n; \hat{\theta}_{a0}^n]$
- Evaluate the polynomials $B(s_k, \hat{\theta}_{b0}^n)$ and $A(s_k, \hat{\theta}_{a0}^n)$ for the values defined in the `snorm` vector. `B=polyval(`$\hat{\theta}_{b0}^n$`,snorm)`, `A=polyval([`$\hat{\theta}_{a0}^n$`;1],snorm)`, and compute the model output as `GGN=B./A`.
- Construct the error vector $E_l$, ($E_l$`=Gm-GGN`), and the Jacobian matrix $J_l$ as explained in Section 5.2.3.
- Split and concatenate the real and imaginary parts of $E_l$ and $J_l$ and obtain the parameter update $\Delta\hat{\theta}_l$ as the least squares solution and record the condition number of the split Jacobian matrix.
- Update the parameter vector, $\hat{\theta}_{l+1}^n = \hat{\theta}_l^n + \Delta\hat{\theta}_l$. Extract the normalised numerator and denominator coefficients from $\hat{\theta}_{l+1}^n$.
- We now have the normalised estimates $\hat{\theta}_{b1}^n$ and $\hat{\theta}_{a1}^n$ from the first iteration $l = 1$. Repeat the above steps using $\hat{\theta}_{b1}^n$ and $\hat{\theta}_{a1}^n$ for the next iteration. Perform this until your termination condition is met.

- Once complete, adapt the normalised parameters: $\hat{\theta}_b = [b_{n_b}/\alpha^{n_b}, \ldots, b_0/1]^T$ and
$\hat{\theta}_a = [a_{n_a}/\alpha^{n_a}, \ldots, a_1/\alpha]^T$.
- Evaluate the GN model: `GGN=freqs(`$\hat{\theta}_b$`,`$[\hat{\theta}_a; 1]$`,w)` and plot `Gm`, `GGN` and the error (`Gm-GGN`) in dB against frequency.

## 5.4 Analysis

### 5.4.1 Questions

We have now estimated the transfer function parameters using three routines (Levy, Sanathanan and Gauss-Newton). Each has its merits and demerits which you can analyse from the results.

- **What model orders $n_b$ and $n_a$ gave a good fit? We only had data for model estimation, what would be the more complete method of verifying the model orders?[5]1]Hint: What was our procedure in Lab 1?**
- **Did the parameter estimates improve when going from Levy to Sanathanan and to Gauss-Newton?**
- **How did the matrix conditioning in the three routines compare? Imagine we didn't normalise, set $\alpha = 1$ and run your code, how do the results and the condition number compare?**
- **What is the advantage of the Levy method? Does the Sanathanan and GN method converge to good estimates with some other starting parameter values?**

### 5.4.2 Jacobian approximation

One of the drawbacks of the GN method is the analytical evaluation of the Jacobian. For complex models it may be difficult to obtain the true Jacobian matrix. In such cases an approximate of the Jacobian can be obtained numerically. Since $G_m$ (the frequency response based on the data) does not depend on $\theta$, starting with the definition of the error, one row of the Jacobian matrix is:

$$E(s_k, \theta) = G_m(s_k) - G(s_k, \theta) \tag{5.19}$$

$$\left. \frac{\partial E(s_k, \theta)}{\partial \theta} \right|_{\theta = \hat{\theta}_l} = - \left. \frac{\partial G(s_k, \theta)}{\partial \theta} \right|_{\theta = \hat{\theta}_l} = -J_l(s_k)$$

$$\therefore J_l(s_k) = - \left[ \frac{\partial E(s_k, \theta)}{d\theta(1)}, \cdots, \frac{\partial E(s_k, \theta)}{d\theta(n_\theta)} \right] \tag{5.20}$$

[5] [

with $n_\theta = n_a + n_b + 1$. The complete Jacobian matrix has dimension $F \times n_\theta$. All entries of the Jacobian can be approximated via finite differences. The entry of the $k$th row and $i$th column is then:

$$-\frac{\partial E(s_k,\theta)}{d\theta(i)}\bigg|_{\theta=\hat{\theta}_l} \approx -\frac{E(s_k,\hat{\theta}_l+\triangle\theta_i)-E(s_k,\hat{\theta}_l)}{\epsilon\hat{\theta}_l(i)} \qquad i=1,2,\dots,n_\theta$$

with,

$$\triangle\theta_i = \epsilon\hat{\theta}_l(i)e_i \tag{5.21}$$

here $e_i \in \mathbb{R}^{n_\theta}$ is a canonical basis vector (all zeros but the $i$th component is 1) and $\epsilon$ an arbitrarily small constant. As such, for every column, an increment of $\epsilon\hat{\theta}_l(i)$ is added to a different element of $\hat{\theta}_l$. This means that relative to $\hat{\theta}_l$, we are looking at the change in error ($E$) when each parameter increments, one at a time, by an amount of $\epsilon\hat{\theta}_l(i)$. The complete numerical approximation of the Jacobian is:

$$J_l \approx -\begin{bmatrix} \frac{\triangle E(s_1,\hat{\theta}_l+\triangle\theta_1)}{\epsilon\hat{\theta}_l(1)} & \frac{\triangle E(s_1,\hat{\theta}_l+\triangle\theta_2)}{\epsilon\hat{\theta}_l(2)} & \cdots & \frac{\triangle E(s_1,\hat{\theta}_l+\triangle\theta_{n_\theta})}{\epsilon\hat{\theta}_l(n_\theta)} \\ \frac{\triangle E(s_2,\hat{\theta}_l+\triangle\theta_1)}{\epsilon\hat{\theta}_l(1)} & \frac{\triangle E(s_2,\hat{\theta}_l+\triangle\theta_2)}{\epsilon\hat{\theta}_l(2)} & \cdots & \frac{\triangle E(s_2,\hat{\theta}_l+\triangle\theta_{n_\theta})}{\epsilon\hat{\theta}_l(n_\theta)} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\triangle E(s_F,\hat{\theta}_l+\triangle\theta_1)}{\epsilon\hat{\theta}_l(1)} & \frac{\triangle E(s_F,\hat{\theta}_l+\triangle\theta_2)}{\epsilon\hat{\theta}_l(2)} & \cdots & \frac{\triangle E(s_F,\hat{\theta}_l+\triangle\theta_{n_\theta})}{\epsilon\hat{\theta}_l(n_\theta)} \end{bmatrix} \tag{5.22}$$

The numerator term in the above matrix is:

$$\triangle E(s_k,\hat{\theta}_l+\triangle\theta_i) = E(s_k,\hat{\theta}_l+\triangle\theta_i) - E(s_k,\hat{\theta}_l) \tag{5.23}$$

and from equation (5.19) follows that it is equal to:

$$\triangle E(s_k,\hat{\theta}_l+\triangle\theta_i) = G(s_k,\hat{\theta}_l) - G(s_k,\hat{\theta}_l+\triangle\theta_i) \tag{5.24}$$

Therefore, for some fixed $\epsilon$ value, at every GN iteration we can evaluate equations (5.21) and (5.24) and construct the approximated Jacobian (matrix equation 5.22).

You can verify this implementation as an additional exercise. In practice, it is strongly adviced to verify the exactly calculated jacobian matrix by comparing it with the numerical calculation to check for calculation and programming errors.

- Set $\epsilon = 1E{-}5$ and declare a vector of increments for each of the parameters `deltaTheta=`$\epsilon * \hat{\theta}_l$.

We will start a `for` loop which goes from $i = 1$ to $n_\theta$, in which we will implement the canonical basis vector $e_i$ and the approximated Jacobian matrix. Note: this `for` loop should be within your main loop that goes over the iteration index $l$.

- Declare the $e_i$ vector as a vector of zeros; `evec=zeros(`$n_\theta$`,1)` and set the corresponding element as 1. `evec(i)=1`.

- Extract the corresponding parameter increment and add it to $\hat{\theta}_l$. `theta_inc=`$\hat{\theta}_l$`+deltaTheta(i)*evec`. By doing so, we have evaluated $\hat{\theta}_l + \triangle\theta_i$.

- Extract the numerator and denominator coefficients from `theta_inc`, let this be $\hat{\theta}_b^{inc}$ and $\hat{\theta}_a^{inc}$, and evaluate the polynomials. `B_inc=polyval(`$\hat{\theta}_b^{inc}$`,snorm)`, `A_inc=polyval(`$[\hat{\theta}_a^{inc}; 1]$`,snorm)`.

- Compute the GN model output at the incremented parameters, `G_inc=B_inc./A_inc`, and the corresponding error increment (as in equation 5.24). `DeltaE=GGN-G_inc`. Divide each element of `DeltaE` with $\epsilon\hat{\theta}_l(i)$, this gives us one column of the matrix and by saving each column in the loop we have the approximated Jacobian. `Japprox(:,i)=-DeltaE./deltaTheta(i)`.

Run your GN routine by using the approximated Jacobian and evaluate the error on the estimated parameters, also compare the true Jacobian and approximated Jacobian matrices.

- **Is the effort spent on constructing the analytical Jacobian worth it when compared with the approximated Jacobian?**

- **Try using a larger value for $\epsilon$, does the estimate get worse?**

For any questions please email and arrange a date and time with one of the lab assistants.

| | |
|---|---|
| Maarten Schoukens | `Maarten.Schoukens@vub.ac.be` |
| Mark Vaes | `Mark.Vaes@vub.ac.be` |
| Maral Zyari | `Maral.Zyari@vub.ac.be` |
| Dieter Verbeke | `Dieter.Verbeke@vub.ac.be` |