Contents

Part I

Theoretical aspects

Chapter 1

Problem position

1. Objective

This study is devoted to the calibration of a sensor (assumed to be a linear time-invariant device). We mainly focus on the infrasonic stations but most results can extended to seismic stations.

[c1]The PTS specifications for infrasonic frequency response sensor are ±1% on the amplitude gain and ±5° on the phase in the frequency bandwidth going from 0.02 Hz up to 4 Hz. That represents a challenging issue.

[c2]tutu The method is based on the presence of a quasi permanent background acoustical sound all over the Word. The term ”acoustical” refers to the propagation velocity of this wave which is commonly around 340 m/s inducing in the whole frequency domain of interest large wavelengths compared to the size of the sensor. It follows that this signal appears coherently on the two sensors. In the following it is shortly called the coherent signal and is denoted $s(t)$. It has been observed and seems have a large enough frequency bandwidth to satisfy the PTS requirements. A general schema is reported figure 2.1. Unfortunately this signal is not observable. It follows that, in presence of additive noises, the problem is ill-conditioned. That means that they are an infinity of possible solutions.

A special mention concerns the noises denoted w_u and w_r (see figure 2.1). They are mainly due to the wind. Although the wind structure is very different of the ”acoustical” signal $s(t)$, we will see that a parameter

[c1] *Momo*: pas d’accord

[c2] *Momo*: ~~The method to estimate the response of a sensor, denoted by sensor under test (SUT), is based on the use of a second sensor, denoted reference sensor (SREF), whose the frequency response is perfectly~~

of interest is $\zeta = v/f$ where v is the wind velocity and f the frequency. ζ can be interpreted as a wavelength. For typical values of the wind velocity, ζ is much lower than the distances between air inlets of the noise reduction system (NRS).

2. Experimental testbed

An experimental testbed has been deployed in the IS26 located in Freyung in Germany:

- The sensors under test/reference are reported table 1.1. The reference sensors are deployed at a few meters of the sensors under test. These sensors have been checked before installation. We have two kinds of SREF, one is MB32005 with a short period, the other is MB3 with long period. The SUT are all MB3 with long period.
- The environment is particularly quiet, due to the Bavarian Forest where the station is located. It follows that the wind is almost absent during large periods of time.
- The data are recorded in continuous.

It is worth to notice that the calibration problem is a little bit different of the detection problem which consists to provide an alert when the system is out of specifications.

Site	SUT	info.	SREF	info.
	model	serie #	model	serie #
H1/C1	MB3	00020	MB2005	6046
H2/C2	MB3	00017	MB2005	5125
H3/C3	MB3	00014	MB2005	6039
H4/C4	MB3	00023	MB2005	5104
H5/C5	MB3	00012	MB2005	7095
H6/C6	MB3	00011	MB3	00007
H7/C7	MB3	00021	MB3	00008
H8/C8	MB3	00015	MB3	00022

Table 1.1: *sensor specifications: all sensors have the same theoretical sensitivity of* 0.02 *V/Pa. The MB3s have self noise lower than this of the MB2005s. The sites associated to the "Wind Noise Reduction System" are labelled H whereas the sites od reference sensors are labelled C.*

3. Main issue

The main issue is related to the under-determination of the problem (also known as blind identification). More specifically we have four unknowns for only two observations. The four unknowns are the frequency response of the SUT, the spectral shapes of the coherent signal and the two noises on the two sensors. The two observations are the output signals on both sensors.

A common way to solve this kind of problem is to introduce some *a priori* knowledges/assumptions. The drawback is what happens when the *a priori* knowledges are not well-verified.

Here a list of *a priori* knowledges

- the signals are stationary in the whole frequency bandwidth of interest. More specifically we have to precise what we mean in term of stationarity duration. The worst case is for the low frequency, i.e. for long period. For example if the accuracy requires to integrate the signal over five periods and if we want to analyze the spectral content around the frequency of 0.01 Hz, about 10 minutes of stationarity are needed.

  Let us notice that, even under the stationarity assumption the problem is still under-determined.

- the coherent signal and the noises can be modeled with parametrical models such as auto-regressive paradigm. In [?] a generalized autoregressive conditionally heteroskedastic (GARCH) model has been used to model the wind. This approach has not been investigated here.

- the noises on the two sensors are uncorrelated and white. This is the simplest parametrical model which depends on only one parameter. In this case the frequency response is identifiable, but observations and also the physical aspects of the wind noises deny this assumption.

- the system under test is a linear filter with *given* numbers of poles and zeroes (parametrical approach). This approach has not been already investigated. It follows that, for the calibration, imply the numbers of poles and zeroes could hide a singularity and then induce a bias. To illustrate saying that a system is of order 1 can hide in the estimate the presence of a non suspected resonance. But in the alert issue framework, we see that something is wrong, therefore we suspect that it could be an efficient way for test.

- assuming that the two noises are uncorrelated, the indetermination disappears if we assume that the ratio between the two noise levels is known. That is a realistic way in high frequency because the NRS works well and the ratio is about the inverse of the number of inlets.

- assuming that the two noises are uncorrelated, the indetermination disappears if we assume that one of the two noises is negligible, typically it could be the case for the SUT which is provided with a noise reduction system (NRS). But we also want to detect if this NRS is working well.

  The advantage of the assumption is the second noise could be very large, we have only to use the formula (2.5). The drawback is that no test does exist to provide an answer to the question: is the noise on the SUT negligible? Therefore if the assumption becomes wrong the error could be very large.

- assuming that the two noises are uncorrelated, the indetermination can be a fortiori removed if both noises are negligible. The advantage of this second assumption is double: (i) we can use any of the two formulas (2.5) or (2.6), and (ii) we have an efficient way to test if or not the two noises are negligible. The main drawback is it could be difficult to find time windows where the condition is verified for a longperiod, particularly in the high frequency band. A solution is to consider shorter window sizes in the high frequency ranges.

In summary, our conclusion is that, to get the expected accuracy, we have to consider that both noises are negligible. To test that the MSC (see below for specific definition) is used along time windows whose lengths vary in relation with the frequency ranges. From picturing manner, the MSC measures the fact that the estimated values exhibit very low dispersion, hence low noises, along a few consecutive windows.

Chapter 2

Spectral analysis

1. Observation model

The notations are reported figure 2.1. The input signals write:

$$\left\{ \begin{array}{rcl} e_u(t) & = & h(t) \star s(t) + w_u(t) \\ e_r(t) & = & s(t) + w_r(t) \end{array} \right. \tag{2.1}$$

where $h(t)$ would take into account some possible filtering effect. It is worth to notice that because $s(t)$ is unknown, the unknown filter $h(t)$ is not identifiable, even in the simple case where $h(t) \star s(t) = \alpha s(t)$. In the following we only consider the case where $h(t) = \delta(t)$, leading to:

$$\left\{ \begin{array}{rcl} e_u(t) & = & s(t) + w_u(t) \\ e_r(t) & = & s(t) + w_r(t) \end{array} \right. \tag{2.2}$$

The common signal $s(t)$ can be seen as the coherent part of the input signals. It is the key of the estimation. On the other hand the noises induce a loss of identifiability and appear as a nuisance factor which can be characterized by a loss of coherence (LOC). It follows that the observation signals write:

$$\left\{ \begin{array}{rcl} x_u(t) & = & g_u \star (s(t) + w_u(t)) \\ x_r(t) & = & g_r \star (s(t) + w_r(t)) \end{array} \right. \tag{2.3}$$

The stationarity and the absence of correlation between $s(t)$, $w_u(t)$ and $w_r(t)$ play a fundamental role in the identifiability of the response of the SUT. We assume that

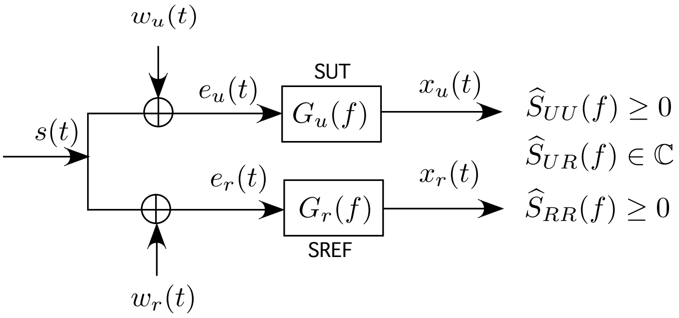
$w_u(t)$
$e_u(t)$
SUT
$G_u(f)$
$x_u(t)$
$\widehat{S}_{UU}(f) \geq 0$
$s(t)$
$\widehat{S}_{UR}(f) \in \mathbb{C}$
$e_r(t)$
$G_r(f)$
$x_r(t)$
$\widehat{S}_{RR}(f) \geq 0$
SREF
$w_r(t)$

Figure 2.1: *Basic observation schema.* $\widehat{S}_{UU}(f)$, $\widehat{S}_{RR}(f)$ *and* $\widehat{S}_{UR}(f)$ *represent the three estimated spectral components under stationary assumptions*

- $s(t)$ is a wide-sense stationary process with zero-mean and spectral density $\gamma_s(f)$,
- $w_u(t)$ and $w_r(t)$ are two stationary processes with zero-mean and respective spectral densities $\gamma_u(f)$ and $\gamma_r(f)$.
- $s(t)$, $w_u(t)$ and $w_r(t)$ are jointly independent. More specifically we have for any (t, t'):
$$\mathbb{E}\left[w_u(t)w_r(t')\right] = 0$$
- $G_u(f)$ represents the frequency response of the SUT which is the combination of the sensor under test, the frequency response of the anti-noise pipes and the digitalizer. It is the parameter to estimate.
- $G_r(f)$ represents the frequency response of the SREF. $H_r(f)$ is known and could be checked just before the test operation.

2. SUT estimation

If we assume that the SUT is well modeled by a linear filter, a common way to calibrate it is to estimate its impulse response or equivalently its frequency response. In the absence of *a priori* knowledges, i.e. non parametrical approac, the only parameters are the bounds of the frequency bandwidth. The lowest value is related to the largest period we want to analyze and the highest leads to the sampling frequency. In infrasonic system the largest period is about 50 seconds. Therefore to expect a good accuracy we can assume that at least 5 times this value is necessary leading to about 4 minutes. In frequency domain that is equivalent to about 0.004 Hz. In infrasonic system the largest frequency leads to choose the sampling frequency to 20 Hz. In summary impulse response consists of 4800 real values and frequency response consists of 2400 complex values.

There are basically two ways to estimate a linear filter either in time domain or in frequency domain. In both cases to get a good accuracy it is necessary to do averaging. The stochastic description is well-suited for that. The averaging can be applied using blocks of data or by adaptive approach (as Kalman filter or recursive least square). In the adaptive approach we adjust progressively the parameters of interest. This approach has been investigated leading to no interesting results. The main reason is the large variability of the observation. Indeed this kind of approach needs slow evolution of the stationarity. Numerical studies have shown that the best results are obtained if we take into account a small number of observation blocks and omit all the others.

In summary, in the following we only consider that the averaging is performed by block and we have adopted a frequency analysis. Basic tools for that is the spectral representation of wide sense stationary process (see annex 7).

If we adopt a stochastic approach and assume that the processes $s(t)$, $w_u(t)$ and $w_r(t)$ are jointly stationary and jointly uncorrelated, and the two sensors are time-invariant, the spectral content is described by the following spectral equations:

$$\left\{\begin{array}{lcl} S_{UU}(f) & = & |G_u(f)|^2(\gamma_{ss}(f)+\gamma_u(f)) \\ S_{RR}(f) & = & |G_r(f)|^2(\gamma_{ss}(f)+\gamma_r(f)) \\ S_{UR}(f) & = & G_u(f)G_r^*(f)\gamma_{ss}(f) \end{array}\right. \tag{2.4}$$

$S_{UU}(f)$, $S_{RR}(f)$ and $S_{UR}(f)$ are respectively the auto-spectrum of the signal of the SUT, the auto-spectrum of the signal of the SREF and the cross-spectrum between them. It is worth to notice that the cross-spectrum does not depend on the noises because the assumption of non correlation.

Under-determination

In the equations (2.4) the response $G_r(f)$ is perfectly known, the unknows are $G_u(f) \in \mathbb{C}$, $\gamma_{ss}(f) \in \mathbb{R}^+$, $\gamma_u(f) \in \mathbb{R}^+$ $\gamma_r(f) \in \mathbb{R}^+$.

It follows that, in absence of *a priori* knowledges, the system (2.4) is under-determined with 1 degree of freedom (d.o.f.). That means that for all f we can, for example, choose arbitrarily the ratio $\rho(f) = \gamma_u(f)/\gamma_r(f)$ from 0 to infinity, then solve the systems (2.4). If one of the two noises is zero, i.e. the noise ratio is either 0 or infinite, the solution becomes unique and is given by:

- if $\gamma_u(f) = 0$, i.e. $\rho(f) = 0$

$$G_u(f) = \frac{S_{UU}(f)}{S^*_{UR}(f)} G_r(f) \tag{2.5}$$

- if $\gamma_r(f) = 0$, , i.e. $\rho(f) = \infty$

$$G_u(f) = \frac{S_{UR}(f)}{S_{RR}(f)} G_r(f) \tag{2.6}$$

Unfortunately there is no way to test if one of the two noises is zero.

To circumvent this drawback we consider a more constrained condition: both noises $w_u(t)$ and $w_r(t)$ are almost negligible. In this case the two estimators given by (2.5) and (2.6) are almost equal and almost unbiased.

The assumptions of negligible noises can be tested by thresholding the Magnitude Square Coherence (MSC) defined by:

$$\text{MSC}(f) = \frac{|S_{UR}(f)|^2}{S_{UU}(f) S_{RR}(f)} \tag{2.7}$$

It can be shown that the MSC is 1, iff the two noises are zero.

In the model (2.4) MSC also writes:

$$\text{MSC}(f) = \frac{1}{(1 + \text{SNR}_u^{-1}(f))(1 + \text{SNR}_r^{-1}(f))} \tag{2.8}$$

where $\text{SNR}_r^{-1}(f) = \gamma_r(f)/\gamma_{ss}(f)$ and $\text{SNR}_u^{-1}(f) = \gamma_u(f)/\gamma_{ss}(f)$.

Remark on the deterministic approach (a few details page 74)

Considering only time intervals with almost zero noises, we can ask: why do we use stochastic approach. Indeed the simple ratio of the short time Fourier transforms of the two observations, where the term short time is related to the long period, is the searched response. But there is also an equivalent to the MSC test. In the deterministic approach we have to validate the assumption that there is no noise and that in some way by locking if the performed ratio is almost constant along successive windows. That can be expressed by saying that the two sequences of Fourier along a few windows are correlated, and that leads to the MSC-like test.

3. Spectral analysis

The classical moment method leads to replace in the formulas like (2.5) or (2.6) the true spectral components by consistent estimates. In the absence of *a priori* knowledges that is given by a non parametrical spectral analysis. More details about the statistics of these quantities, as the ratio of spectral components or the MSC, are presented in annexe 8.

Discrete time model

We consider that the frequency band of interest is $(1/\tau_c, F_M)$. F_M allows to replace the continuous time signals by a time series with a sampling frequency $F_s = 2F_M$. In infrasonic $F_s = 20$ Hz. τ_c implies to estimate the response up to this period. In the infrasonic context $\tau_c = 50$ seconds. That implies to reach in the normalized frequency domain the frequency of about $1/\tau_c F_s = 1/1000$. Taking a regular grid on the unit circle, as it is with FFT algorithm, the number L of frequency bins will be of an order of magnitude of a few times $\tau_c F_s$. If we consider an order of 4 to 5, that leads to take L around 4096.

In the following the index k refers to the frequency kF_s/L. For example $\gamma_{ss,k} = \gamma_{ss}(kF_s/L)$. Because the signal is real, all spectral representations have hermitian symmetry and it is only necessary to consider normalized frequency between 0 and 1/2. Finally the spectrum at frequency 0 is real valued and will be omitted. Therefore the spectral representation is restricted to the values of the frequency index k going from 1 to $K = L/2$.

Let us rewrite the spectral components

$$\left\{\begin{array}{lcl} S_{UU,k} &=& |G_{u,k}|^2(\gamma_{ss,k} + \gamma_{u,k}) \geq 0 \\ S_{RR,k} &=& |G_{r,k}|^2(\gamma_{ss,k} + \gamma_{r,k}) \geq 0 \\ S_{UR,k} &=& G_{u,k}G^*_{r,k}\gamma_{ss,k} \in \mathbb{C} \end{array}\right. \tag{2.9}$$

From (2.7) we derive the MSC expression in the discrete time case:

$$\mathrm{MSC}_k = \frac{|S_{UR,k}|^2}{S_{UU,k}S_{RR,k}} \tag{2.10}$$

and, for MSC_k very close to 1 we can use any of the two formulas (2.5) or (2.6), giving:

$$G_{u,k} = \frac{S_{UU,k}}{S^*_{UR,k}} G_{r,k} \tag{2.11}$$

or

$$G_{u,k} \;=\; \frac{S_{UR,k}}{S_{RR,k}}\, G_{r,k} \tag{2.12}$$

We have to keep in mind:

- that these equations assume stationarity,
- that we perform the value of $G_{u,k}$ during the period where the 2 noises are sufficiently low,
- that we have no access to the true values of the spectral components but only estimates, inducing dispersion.

Based on that, we can replace in expressions (2.10) and (2.11) the spectral components $S_{UU,k}$, $S_{RR,k}$ and $S_{UR,k}$ by their consistent estimates $\widehat{S_{UU,k}}$, $\widehat{S_{RR,k}}$ and $\widehat{S_{UR,k}}$.

These consistent estimates are obtained via a Welch's approach by averaging successive periodograms, with a typical overlap of 50% and Hann's window. It is worth to notice that the number of frequency bins is related to the long period signals whereas the accuracy of the estimates are related to the number of windows which are used for averaging the periodograms.

Also we can find in the annex 8 the details to calculate the probability distributions of $\widehat{\mathrm{MSC}}$ and of the ratio $\widehat{S_{UU,k}}/\widehat{S^*_{UR,k}}$. We also provide Matlab programs that performs these distributions.

MSC approach

It is worth noticing that we have only an estimate of the MSC not the true value. Therefore a test function has been determined to ensure with a given confidence level, typically 95%, to be over a target-value. The target-value is related to the accuracy we want on the estimation of the response of the SUT. In this case an important parameter is the stationarity duration of the signals. For example we see on figure 2.2 that with 5 windows, i.e. about 4 minutes (for long period of 50 seconds), the MSC must be over 0.97 to ensure an RMSE of 5% on the amplitude response. For such requirements the MSC threshold of the test is about 0.99, see figure 2.3.

It is worth to notice that the approach is strongly conservative. Indeed we keep only a few periods of signals, but that could be largely enough if we considered that the calibration has to be done once a year.

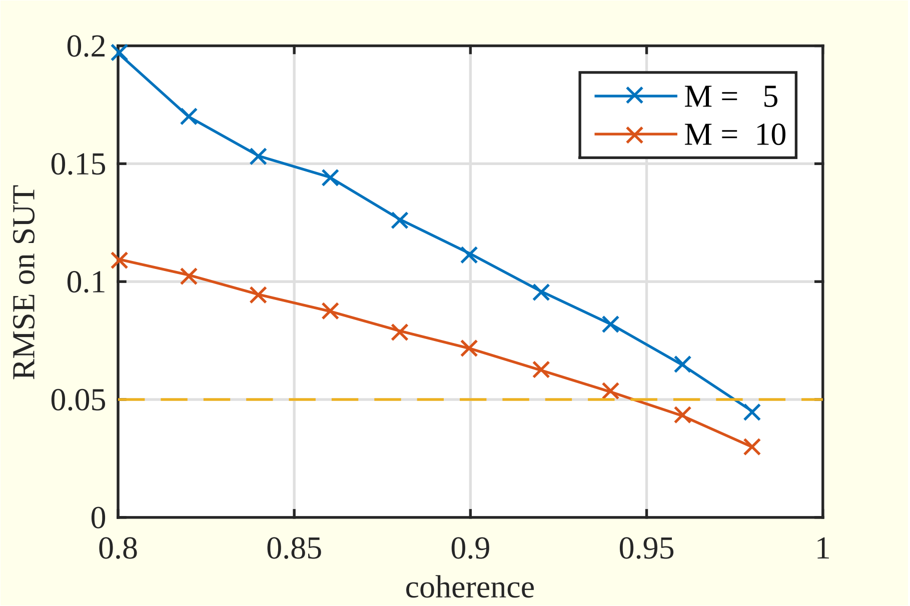

Figure 2.2: *Simulation: for simulating the sensors are two different IIR(2,2) with important resonances are used. The commun coherent signal $s(t)$ is white. The length of the frequency analysis is* 300 *seconds, i.e.* 6000 *points. The spectral components are performed on M times this duration. The root mean square error is performed by integrating on the full band. These curves have been obtained with the program* `estimHanalysis`.

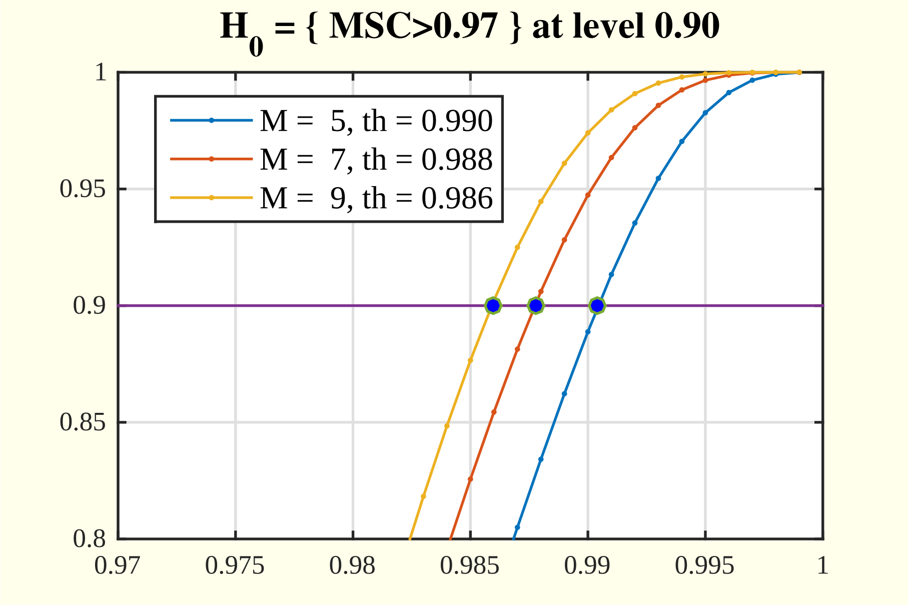

Figure 2.3: *Cumulative function of* $\widehat{\text{MSC}}$ *for different values of the number* M *of averaging under the hypothesis that the MSC is* 0.96*. These curves provide the threshold to test the hypothesis* $H_0 = \{\text{MSC} > 0.96\}$ *with a confidence level of* 90%*.*

It is important to remark that an RMSE of 5% means that the measurement has only a probability of 0.7 to be in this interval, if we assume gaussian asymptotic behavior. Therefore we could expect that we consider at first an accuracy of 10% and reduce in second step this number by aggregating many measurements. The problem is when the MSC is far from 1 a large indetermination appears which does not leads to a gaussian asymptotic behavior *around the true value*. That is reported on the theoretical curves of the figure 2.4. We see that the two ratios are distributed to a gaussian distribution but at a wrong location. It is not possible to correct this bias because that would assume that the noise ratio is exactly know.

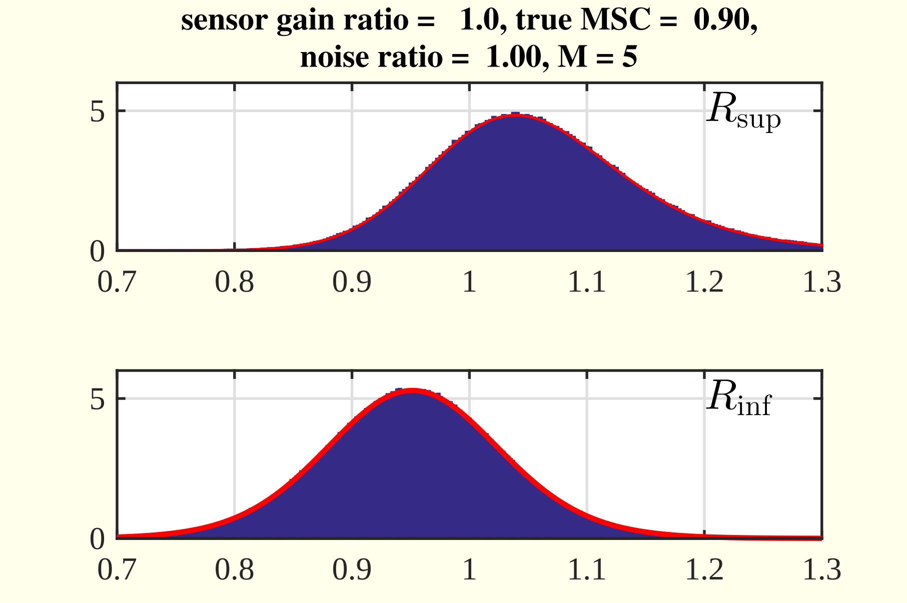

Figure 2.4: *Asymptotic distributions of the two ratios present in the formulas* (2.5) *and* (2.6)*. These distributions depend on the* MSC*, but also on the noise levels and that is not known in practical case. These curves are obtained with the program* `CIHestimate.m`*. The expressions are proved in the annex (see also the provided toolbox).*

SUT estimation

In each time window with an MSC above the selected threshold, we can use any of the formulas (2.11) or (2.12) replacing the true values by estimated values. The estimated values are distributed as it is reported figure 2.4. The theoretical expression of the distributions of both ratios are given in annex

8. In first approximation, we can assume for large values of MSC that the bias is zero and only stays the variance.

Therefore if we assume that the different values along a large period of time are statistically independent, we can access to a more accurate value of the useful ratio by averaging. But because we select time segments with different MSCs, we can use the variance of the estimated ratio and compute an averaging with weights in the inverse of this variance following:

$$\widehat{R} \;=\; \frac{1}{L}\sum_{\ell=1}^{L} w_\ell \widehat{R}_\ell \tag{2.13}$$

where L denotes the number of periods of time with MSC above the selected threshold,

$$\widehat{R}_\ell^{\text{sup}} = \frac{\widehat{S}_{UU,k}}{S^*_{UR,k}} \quad \text{or} \quad \widehat{R}_\ell^{\text{inf}} = \frac{\widehat{S}_{UR,k}}{S_{RR,k}} \tag{2.14}$$

and where w_ℓ equals the inverse of the variances, whose approximate values are given by (8.14) and (8.15) and replacing true values by estimated values:

$$\text{for } R^{\text{sup}} : \qquad 1/w_\ell = \frac{1}{2(2M+1)}\frac{\widehat{\Gamma}_{1,1}}{\widehat{\Gamma}_{2,2}}\frac{1-\widehat{\text{MSC}}}{\widehat{\text{MSC}}^2} \quad \text{or} \tag{2.15}$$

$$\text{for } R^{\text{inf}} : \qquad 1/w_\ell = \frac{1}{2(2M+1)}\frac{\widehat{\Gamma}_{1,1}}{\widehat{\Gamma}_{2,2}}(1-\widehat{\text{MSC}}) \tag{2.16}$$

That is implemented in the fonction `fbankanalysis.m`, look at the flag `weightingflag`.

Chapter 3

Simulation

To investigate the effect the approach, we focuse at first on a short simulation in such a way the groundtruth is available. For that, following the notations of (4.1) we take, for $s(t)$, about half an hour of a MB3 sensor record. The signal has a spectrum reported on figure 3.1. We see that the signal contains the most power in the low frequencies.

There is no filtering on the SREF signal, more specifically $G_{r,k} = 1$. The filter on the SUT signal is a FIR filter whose transfer function consists of a numerator with 3 coefficients and a denominator with 3 coefficients, whose values are reported on table 3.1

numerator	1	-1.0	0.35
denominator	1	-1.15	0.45

Table 3.1: *Coefficient of the IIR(2,2) used in the simulation for the SUT. The SREF has a gain 1. It follows a ratio which presents a large dynamic.*

The wind noise is modeled as the output of the a FIR filter whose input is a white sequence. The FIR filter impulse response consists of 4 one's. That corresponds to a low-pass-filter in such a way the synthetic noise does not introduce much poorer SNR on the high frequency band, regarding the spectrum of $s(t)$.

The spectra are performed using Welch's approach. The basic scheme considers the averaging of successive discrete Fourier transform (DFT) with 50% of overlapping. The maximal size of the DFT is related to the long period signals. Typically in our case that is of 50 seconds. The number of successive windows is related on the mean square error of the spectral estimation. Greater the number of windows better the accuracy, but to greater can lead to the loss of stationarity. We opt to a number of windows corresponding at 10 times the DFT size.

We definitively opt for a Hann's window which provides a good compromise between resolution and leakage. More specifically the leakage (which is the effect of transferring energy from high energy frequency band to low energy frequency band) is low enough to exhibit the presence of microbarom around 0.2 Hz.

1. Zero-noise on the SUT

On the simulation we see at first that the efficiency of the estimator given by the expression (2.5) does not present a visible bias. Also when we reduce the time period of estimation we improve the results in the high frequency range.

The drawback is when we introduce noise on the SUT, the formula is no more valid (underdetermination) and much more we can not test the presence/absence of the noise. Moreover if there is no noise on the SUT, hence the formula does not present a bias but does present a variance, which can be outside the PTS specifications.

As a conclusion this approach is not advised for the calibration.

2. Zero-noise on both SUT and SREF

There is negligible noises on both sensors. All results are reported on figures 3.1 and **??**.

We have chosen two estimation periods, 250 seconds and 50 seconds for white coherent signal and real coherent signal. For each the time are slotted in 9 windows with 50% of overlap to performe FFTs.

As expected we see figure 3.1 that the two periods provide similar results, regarding the coherent signal is stationary and white all over the frequency bandwidth.

Figure **??** we see some differences on the two periods. Indeed the coherent signal, extracted from the MB3 (H8/C8), may be viewed as consisting of a certain loss of stationarity. So it appears that, in the high frequencies, on the shorter period case the coherence increases, hence we can find several time slots with MSC over the threshold, not detected with longer time slots. However it is worth to notice that some high values of the MSC come from the variability of the estimator and not on the loss of stationarity.

In summary it is possible, if necessary, to reduce the period in high frequencies to detect more time slots if the stationarity duration is as the inverse of the frequency location. Greater the frequency, shorter the stationarity duration. We have not validate this assumption.

To use this approach we need an MSC above 0.99. What we see is very typical. If we use long period, to be able to estimate the ratio at low frequency, many values of the estimated MSC are under this threshold. But if we use two different length, we can access in high frequency domain to a greater number where the MSC is still over this threshold.

In conclusion we propose to combine long period/short period to estimate the ratio in the frequency range of interest. It is worth to notice that the short period is used to improve the MSC estimation but to avoid a loss of stationarity.

3. Comparison without/with filter bank

Also we have reported figures 3.2 and 3.3 10 simulations with/without filterb bank (see for filter bank table 4.1). The selected MSC threshold is 0.99. It appears that the accuracy seems to be better in the case with filter bank.

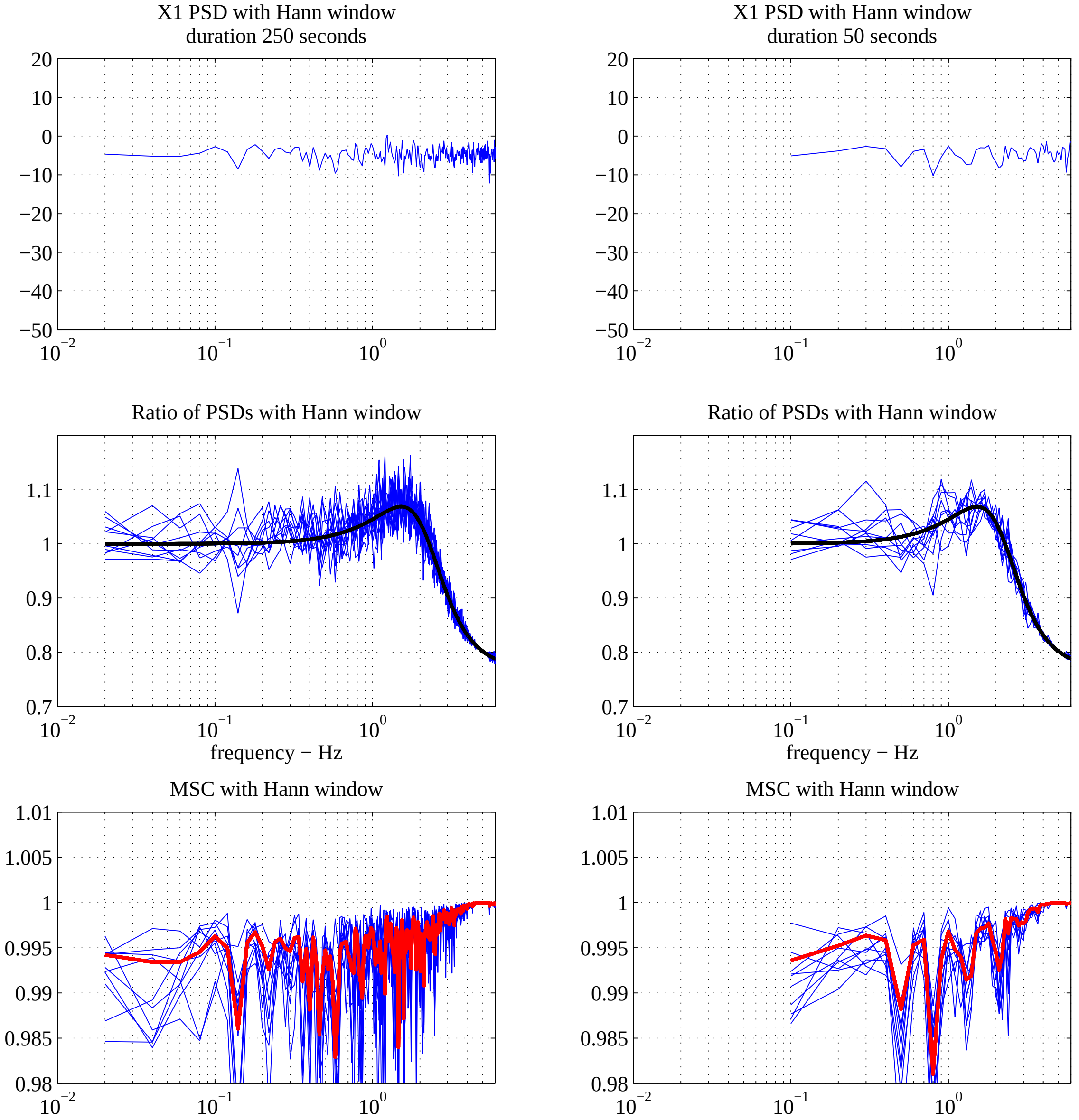

Figure 3.1: *White (`randn` from Matlab) signal with SNR of* 30 *dB on each sensor. Black curve, in the mid-curves is the true ratio. Red curve in the bottom is the true MSC. The true MSC curves present a large variability, because we use a real signal as signal of interest and we need, at first, to estimate its spectrum* γ_{ss} *and then use the formula* (2.8).

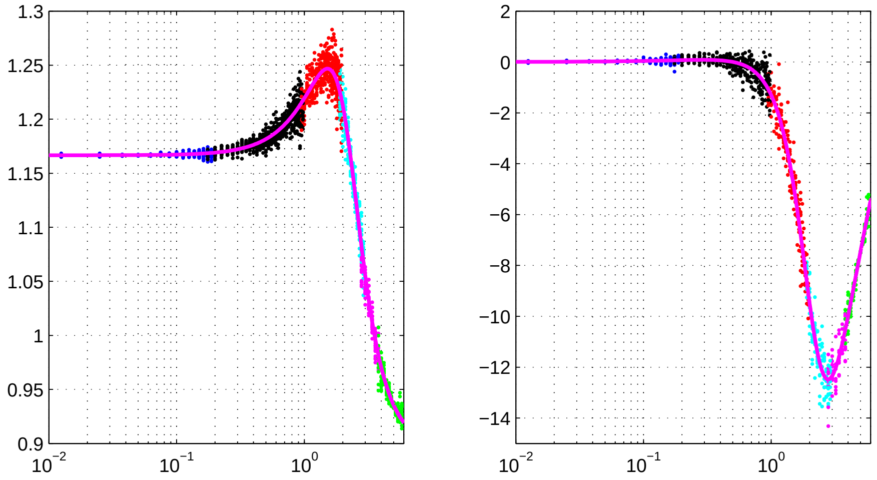

Figure 3.2: *Real SOI from MB3 with a six-filter bank. MSC threshold is* 0.99*. Solid lines are the ground truth. Points correspond to 10 simulations.*

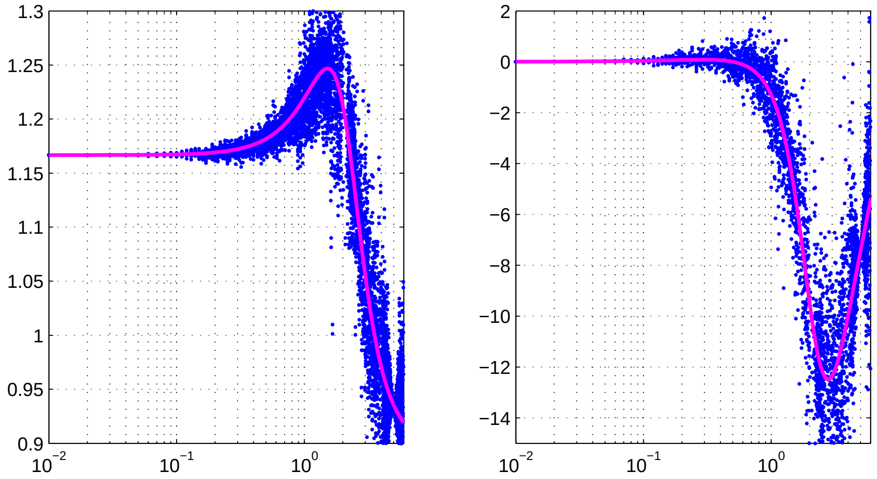

Figure 3.3: *Real SOI from MB3 with no filter bank. MSC threshold is* 0.99*. Solid lines are the ground truth. Points correspond to 10 simulations.*

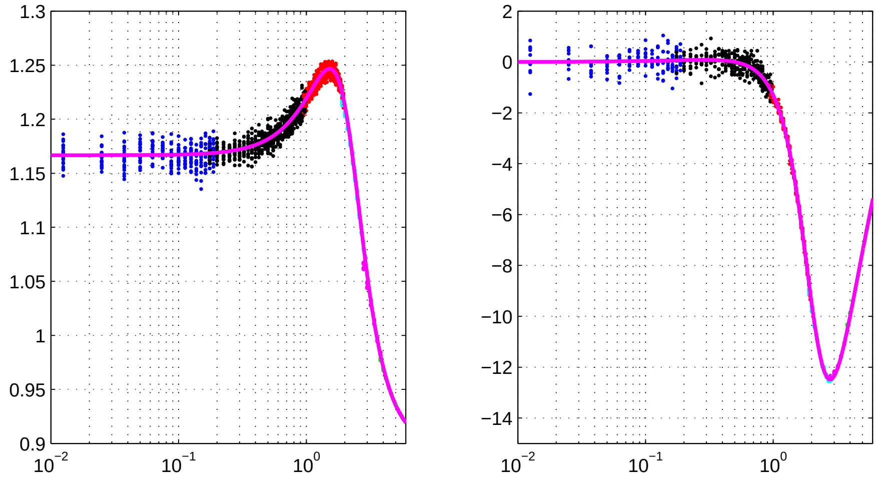

Figure 3.4: *White SOI (from* `randn`*) with a six-filter bank. MSC threshold is* 0.99*. Solid lines are the ground truth. Points correspond to 10 simulations.*

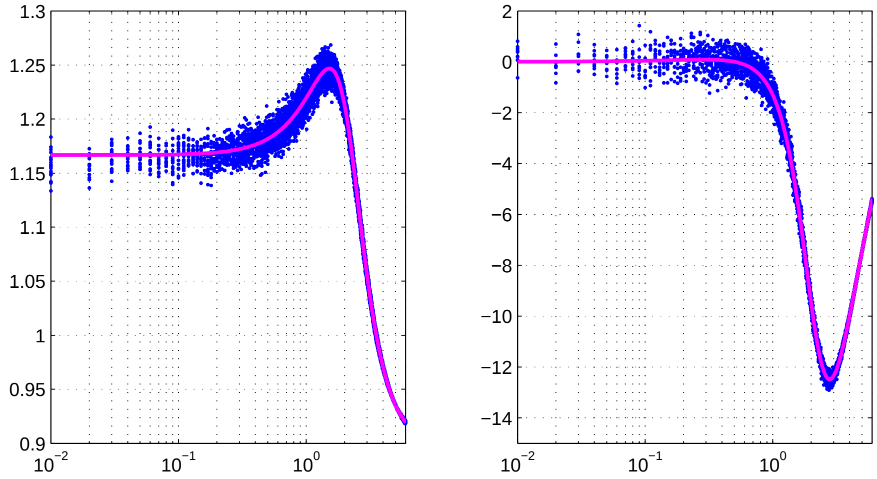

Figure 3.5: *White SOI (from* `randn`*) with no filter bank. MSC threshold is* 0.99*. Solid lines are the ground truth. Points correspond to 10 simulations.*

Chapter 4

Results on IS26 data (incomplete)

The results of this section report the estimation for the eight locations given in the table 1.1.

1. General pipeline for the estimation process

The general pipeline proposed for the estimation process is reported figure ??. It consists

- of a filter bank described in a file of settings which is characterized by a sequence of following descriptors: type, frequency lower bound, upper frequency bound, order, desired stationarity duration etc. Commonly used type is Butterworth (available on Matlab).
- At the output of a filter, the two signals (one for the SUT the other for the SREF) are shared in non-overlapping blocks to perform the spectral estimate. Longer the size of a block more accurate the spectral analysis. But that assumes stationarity, and the real signals do not exhibit permanent stationarity. Therefore we can choose in the setting file the desired stationarity duration in term of multiple of the longest period in the band (inverse of lower frequency bound). Typically we take 5 times the longest period, expecting that we can find such length with stationarity to estimate the spectral components. Even with that, the accuracy is not in accordance with the PTS requirements, but by averaging in a long period of times (many days), we can reach such requirements.

frequency band (Hz)	stationarity period (second)
[0.02 − 0.20]	400
[0.20 − 1.00]	200
[1.00 − 2.00]	100
[2.00 − 3.00]	50
[3.00 − 4.00]	25
[4.00 − 6.00]	25

Table 4.1:

- Each block is then shared in overlapping sub-blocks. We can choose the windowing and the overlap rate. Typically we consider Hann windowing and 50% of overlapping. The periodograms in the different sub-blocks are averaged to provide a spectral estimation.

  Only estimates in the bandwidth of the filter are retained.

- MSC is performed in each bandwidth. If the MSC is above the selected threshold, the value of the ratio $\widehat{S}_{UU,k}/\widehat{S}_{RU,k}$ is saved.

- along the recorded data, for a given frequency index k, the values $\widehat{S}_{UU,k}/\widehat{S}_{RU,k}$ are averaged with weighting factors in accordance with $\widehat{\text{MSC}}$ values.

- Finally the estimation of the SUT response is derived from the SREF response.

Filter bank analysis

Let us recall that in a first step, the signals are filtered in adjacent bands in such a way to use different periods whose the main interest is to consider short periods, if necessary, in the high frequency bands. The table 4.1 is an example of pavement, consisting of 5 bands with log-spaced filter parameters in the band $(0.01 - 5)$ Hz with a variable window length[c0]. Because the two signals are applied to the same filter we can use RII as Butterworth filter. Also because this process is for anlysis, we don't need to downsampling and/or to require perfect recontruction. Even more the bandpass filters can be overlap in the frequency domain. Although a decimation can be used to save computational time, indeed the bandwidths are lower than $F_s/2$, that operation is not considering in the following.

[c0] See also recent PMCC reports.

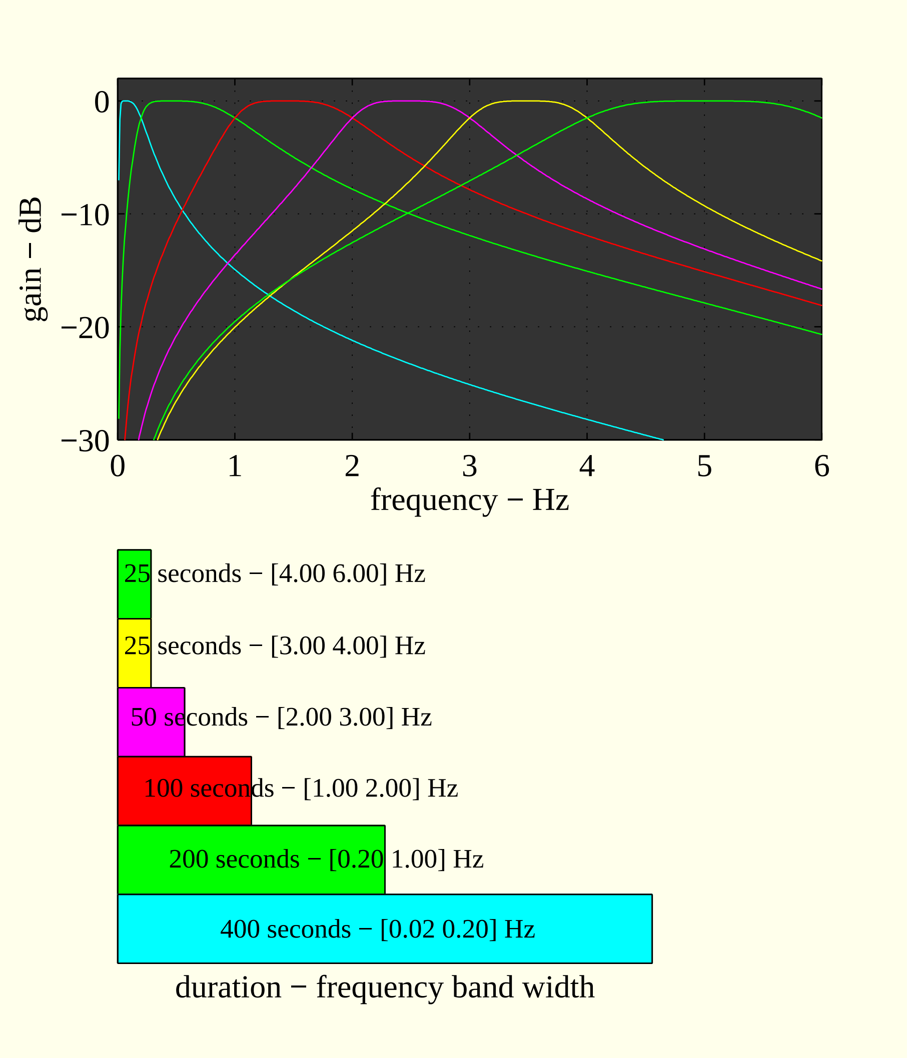

Figure 4.1: *Filter bank*

2. Numerical results

3. Comments and conclusion

For the figures 4.4 to 4.8 associated to the MB2005, an important dip around 0.1 Hz is observed. Also we have reported figure 4.2 a few days on the location H2C2. The different colors are for different days. The distribution seems to be uniform and does not depend on the day. The ratio seems to be different of 1.

If the MSC is about 0.99, meaning that the noises on the two sensors are negligible, and if the two sensors have the same response, the only reason to get a ratio less than 1, is that the acoustical SOI on the SUT is attenuated, may due to the noise system reduction. That would write: it exists $\alpha \in \mathbb{C}$ with $|\alpha| < 1$ s.t.:

$$\left\{\begin{array}{rcl} x_u(t) & = & g_u \star (\alpha s(t)) \\ x_r(t) & = & g_r \star s(t) \end{array}\right. \tag{4.1}$$

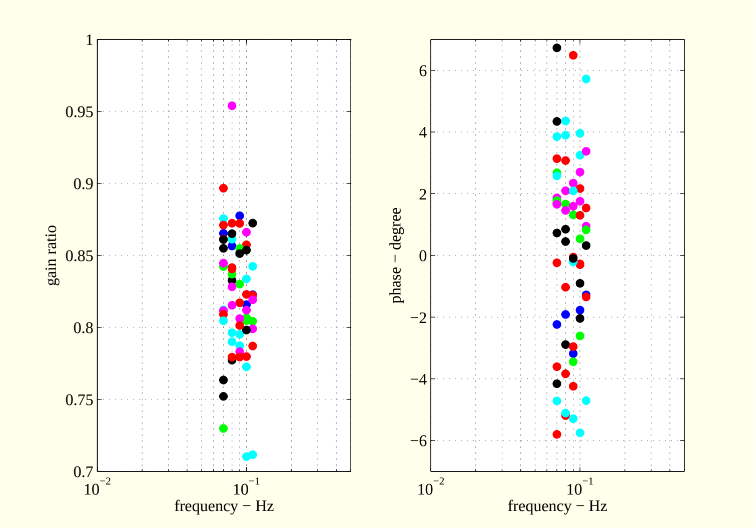

Figure 4.2: *a few days on H2C2. Only the band* $[0.08 - 0.12]$ *Hz is selected. The coherence is above* 0.99. *The different colors for the different days.*

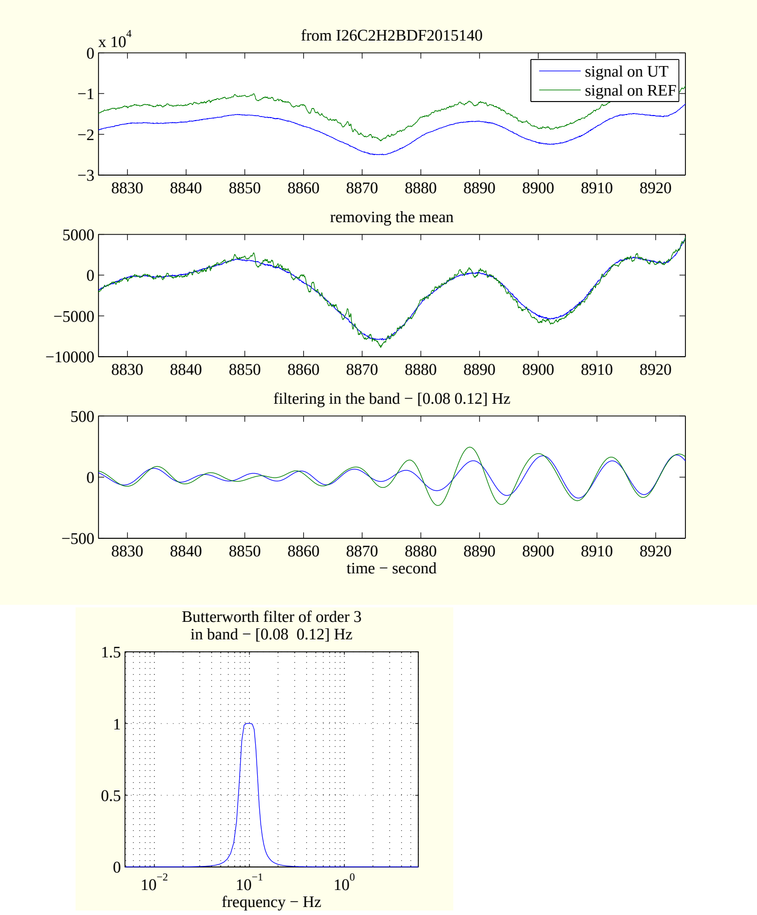

Figure 4.3: Filtered signals

Another way to see the response differences between the 2 sensors in the band $[0.08 - 0.12]$ Hz (gain ratio different of 1), appears figure 4.3. A zoom of the signals is plotted. It consists of about 1 minute, around a position where the observed coherence is above 0.99. We see that the signal on the SREF is bigger than this on the SUT. There is no way and no reason to reject this time window since the difference can be due either to the loss of gain or a unknown transfer function.

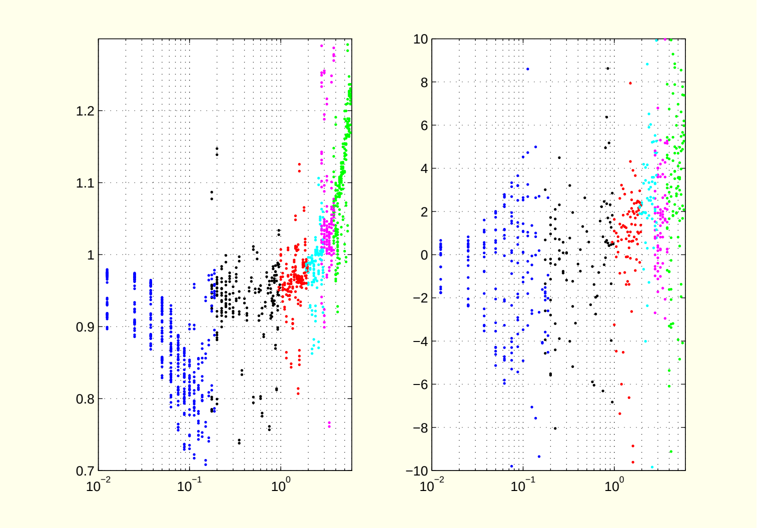

Figure 4.4: *Couple H2C2*

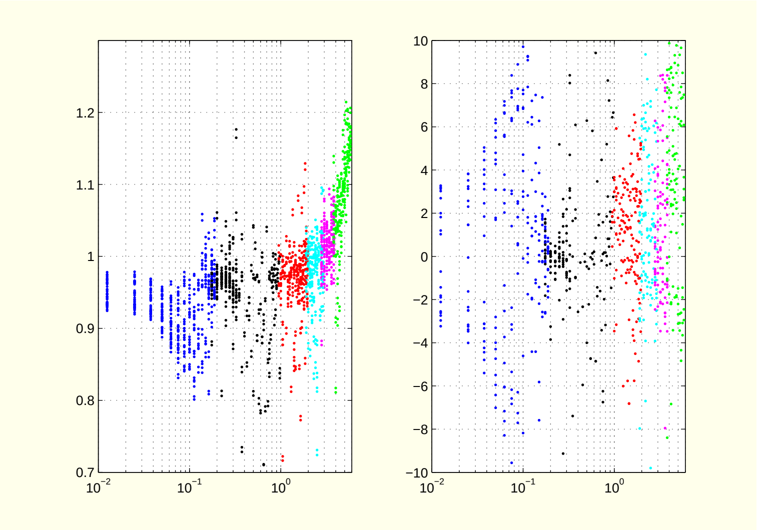

Figure 4.5: *Couple H3C3*

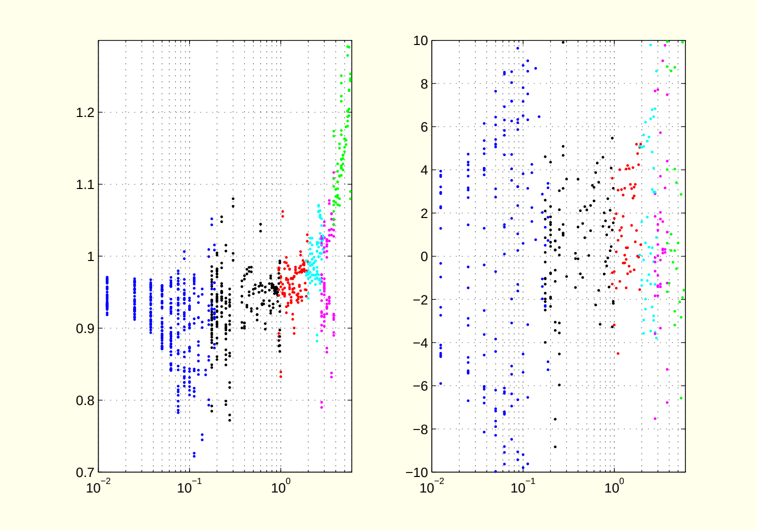

Figure 4.6: *Couple H4C4*

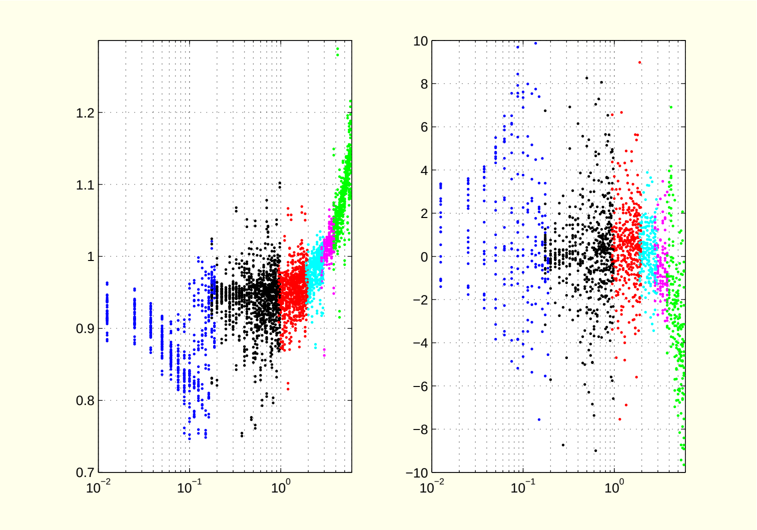

Figure 4.7: *Couple H5C5*

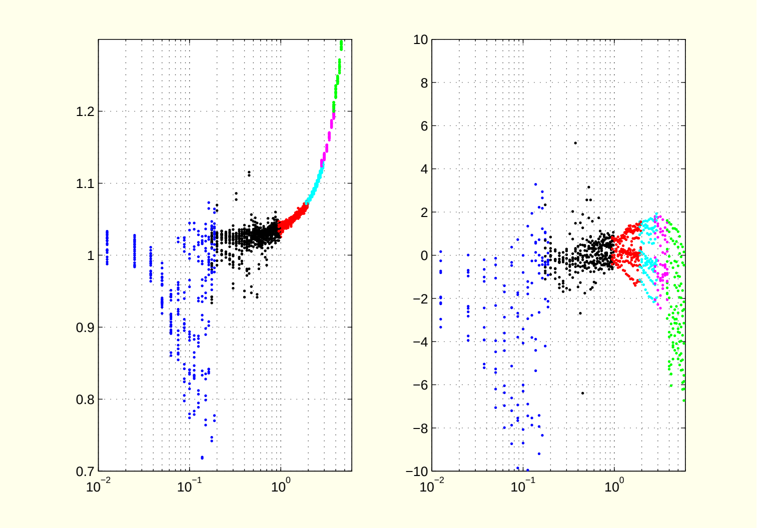

Figure 4.8: *Couple H6C6*

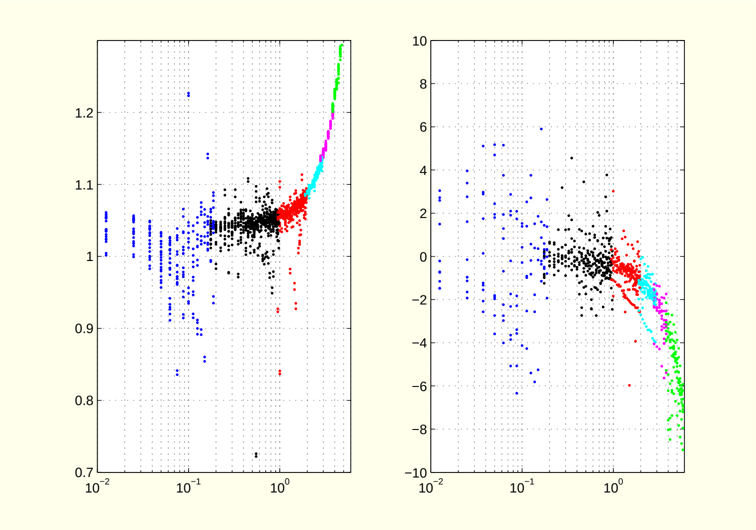

Figure 4.9: *Couple H7C7*

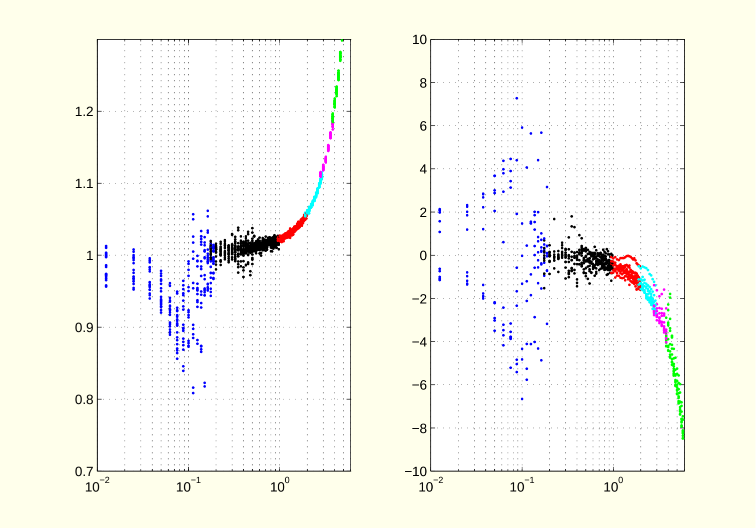

Figure 4.10: *Couple H8C8*

Chapter 5

Wind effect on NRS

1. Introduction

The noise reduction system (NSR) commonly used in front of the infrasonic sensors in the IMS stations is based on the property that the wind noise appears as spatially uncorrelated at the air inlets of the NRS. Therefore by summation, the SNR is improved in the ratio of the number of inlets.

Unfortunately this gain is reduced in low frequencies as it has been observed by Alcoverro and al. [1]. This effect is characterized by the parameter $\zeta = v/f$ where v denotes the wind velocity and f the frequency. ζ can be interpreted as a "wavelength" and must be compared to the inlet inter-distances. If ζ is of the same order of magnitude of the inter distances, spatial coherence increases. It is not a good news, at first because the noise reduction is lower. Another drawback occurs when we want to calibrate the sensor by using a second sensor without NRS. In this case the levels of the coherent part will be different and we have no way to identify it.

As a quantitative illustration for wind velocity above 5 m/s, measurement is no more possible. But for velocity around 1 m/s and for frequency under 0.01 Hz, $\zeta > 100$ m and the wind is seen as coherent by the full NSR. In the middle range for velocity around 1 m/s and frequency around 0.1 Hz, $\zeta \approx 10$ m and a part of the wind noise is seen as coherent just because the NRS. That can induce artefact in the *in-situ* calibration because the coherence level close to 1 does not guaranty that the system to solve is well determined.

This article proposes a model for the spatial coherency, as it observed in [1] and presents simulated and experimental results. This model is similar to the model proposed in [5] which is derived from the pioneer approach of Mack and Flinn [4].

2. Model for wind turbulences

Wind turbulence is a very complex air movement. Here we assume that it can be described (as a gray box) by a stationary random process whose coherence matrix depends on the distance between two spatial locations. At first we just summarize useful definitions of stationary M-ary process.

2.1 Stationary M-ary process

An M-ary process $x(t) = \begin{bmatrix} x_1(t) & \dots & x_m(t) & \dots & x_M(t) \end{bmatrix}^T$ is associated to a spatial-time description. More specifically, t denotes the time and m an index associated with the 3D spatial locations, usually related to a sensor array. Therefore we have to consider spatial and temporal properties. For example, an M-ary process could be:

- temporally white, meaning that $\mathbb{E}\left[x_m(t)x_m(t')\right] = 0$ for $t \neq t'$,
- spatially white, meaning that $\mathbb{E}\left[x_m(t)x_{m'}(t)\right] = 0$ for $m \neq m'$.

There is no relationship between these two notions. In the general case $\mathbb{E}\left[x_m(t)x_{m'}(t')\right]$ does depend on (m, m', t, t'). Temporal stationarity refers to the property that the statistics do depend on $(t - t')$. Spatial "stationarity" is a littlebit more complicate because of the 3D coordinates. In some simple cases $\mathbb{E}\left[x_m(t)x_{m'}(t')\right]$ could depend only on the distance between the 2 points indexed by m, m'. A particular case is when $\mathbb{E}\left[x_m(t)x_{m'}(t')\right] = 0$ for $m \neq m'$ which is called spatially white.

Spectral description

When the process is a wide sense stationary (WSS) process a spectral representation can be associated to the covariances. A zero-mean WSS process is defined by the property that the covariance matrix $R(h) = \mathbb{E}\left[x(t+h)x^T(t)\right]$ does not depend on t. It is also characterized by the spectral matrix which is the Fourier transform of the covariance matrix sequence:

$$\Gamma(f) = \int e^{-2j\pi fh} R(h) dh \tag{5.1}$$

where $\Gamma(f)$ is a square matrix of size M.

The most fundamental property says that, for any value of the frequency f, the spectral matrix is a non-negative matrix. Moreover if $x(t)$ is real, $\Gamma(f) = \Gamma(-f)$.

If the spectral matrix is of rank 1, the process is said spatially coherent. If the spectral matrix is diagonal we say that the process is spatially white. If the spectral matrix is rank deficient we say that the process is partially-coherent. A process whose spectral matrix $\Gamma(f) = \sigma^2 I_M$ is both spatially and temporally white, because $\Gamma(f)$ does not depend on f.

It is also common to consider the function

$$C_{km}(f) \;=\; \frac{\Gamma_{km}(f)}{\sqrt{\Gamma_{kk}(f)\Gamma_{mm}(f)}} \tag{5.2}$$

where it is assumed that $\Gamma_{kk}(f)\Gamma_{mm}(f) \neq 0$. We can also use the matricial notation

$$C(f) = P^{-1}(f)\Gamma(f)P^{-1}(f) \tag{5.3}$$

or

$$\Gamma(f) = P(f)C(f)P(f) \tag{5.4}$$

where

$$P(f) = \begin{bmatrix} \Gamma_{11}^{1/2}(f) & 0 & \cdots & 0 \\ 0 & \Gamma_{22}^{1/2}(f) & \cdots & 0 \\ \vdots & & & \\ 0 & \cdots & 0 & \Gamma_{MM}^{1/2}(f) \end{bmatrix} \tag{5.5}$$

The M-ary matrix

$$C(f) = \begin{bmatrix} 1 & C_{12}(f) & \dots & \\ C_{21}(f) & 1 & \dots & \\ \vdots & & & \\ \dots & \dots & C_{M,M-1}(f) & 1 \end{bmatrix} \tag{5.6}$$

is called the coherence matrix. The function

$$\mathrm{MSC}_{km}(f) = |C_{km}(f)|^2 \tag{5.7}$$

is called the magnitude square coherence, in short MSC, between $x_k(t)$ and $x_m(t)$. It is easy to show that $\mathrm{MSC}_{km}(f) \leq 1$ and

$$\mathrm{MSC}_{km}(f) = 1, \quad \forall f \tag{5.8}$$

if and only if $x_k(t)$ and $x_m(t)$ are related by a linear filter. When $\mathrm{MSC}_{km}(f) = 1$ for all pairs (k, m) the spectral matrix is of rank 1, hence

$$C(f) = \mathbb{1}_M \mathbb{1}_M^T \tag{5.9}$$

and the process is fully coherent.

WSS filtering

A fundamental property concerns the filtering. Let us consider an M-ary zero-mean WSS process $x(t)$ with the spectral matrix $\Gamma_x(f)$. Let $y(t)$ the N-ary process on the output of a multiple input multiple output (MIMO) filter whose frequency response is the matrix $K(f)$ of size N by M. Then $y(t)$ is a WSS process with zero-mean and whose the spectral matrix writes:

$$\Gamma_y(f) \quad = \quad K(f)\Gamma_x(f)K^H(f) \tag{5.10}$$

2.2 Wind noise model

Wind turbulence is a very complex air movement. Here we assume that it can be described by a stationary random process whose coherence matrix as the following entry form [5]:

$$C_{km}(f) = \int \exp(-2j\pi f(r_k - r_m)^T k)\, p_K(k) dk \tag{5.11}$$

where f is the frequency, r_k and r_m are any two 3D locations and k the slownwess vector. $p_K(k)$ is a probability distribution in $\mathbb{R}^3$ meaning that the slownwess vector is considered to be random. Let assume at first that k is deterministic, that writes $p_\Delta(u)du = \delta_0(du)$. Hence expression (5.11) can be written:

$$C_{km}(f) = e^{-2j\pi f(r_k - r_m)^T k_0}$$

That means that the displacement is a plane wave. It is worth to notice that the matrix C can be written:

$$C(f) = E(f)E^H(f), \quad \text{with} \quad E(f) = \begin{bmatrix} e^{-2j\pi r_1^T k_0} & \ldots & e^{-2j\pi r_M^T k_0} \end{bmatrix}^T$$

Hence the multivariate process is fully coherent.

The model, derived from the pioneer work by Mack and Flinn [4] for the loss of coherence, considers that the azimuth a, the elevation e and velocity v are random variables. More specifically, we assume that the 3D vector $\mu = (a, e, v)$ writes:

$$\mu = \mu_0 + \nu \tag{5.12}$$

where $\mu_0 = (a_0, e_0, v_0)$ is a deterministic part and ν a zero-mean Gaussian random vector of dimension 3, whose covariance is denoted Σ_μ. It is shown in the appendix that:

$$C_{m,\ell}(f) = \underbrace{e^{-2j\pi f(r_m - r_\ell)^T k_0}}_{\text{pure delay}} \underbrace{e^{-2\pi^2(f/v_0)^2(r_m - r_\ell)^T \tilde{\Sigma}_k (r_m - r_\ell)}}_{\text{coherence effect}} \tag{5.13}$$

where $\tilde{\Sigma}_k$ is given by (9.6). We let $\zeta = v_0/f$ that appears as a "wavelength". Using the MSC definition (5.7) we derive the following expression:

$$\text{MSC}_{km}(f) = e^{-4\pi^2(r_m - r_\ell)^T \tilde{\Sigma}_k (r_m - r_\ell)/\zeta^2} \tag{5.14}$$

The same parameter ζ has been proposed by Alcoverro and al. in [1] to characterize the coherence of the wind disturbances for closely located sensors. The expression (5.14) has a clear meaning: the coherence between any two spatial locations decreases when the distance between them is large compared to the "wavelength" ζ. It follows that when v_0 increases the wind signals on the air inlets can be considered as uncorrelated and therefore M works as a reduction factor. On the other hand when f is very low, the "wavelength" is large and all coherence matrix entries are approximatively equal to 1.

2.3 Coherence level as a function of v/d

To show that our model is in good agreement with the experimental results reported in [1] we consider a short simulation. We choose M random locations in such a way we obtain several different distances. Using the formula (9.7) we compute for each pair of locations the frequency f_c associated with the coherence level of 0.5 and also the ratio v/d where d is the distance between the two locations. The coordinate couple $(v/d, f_c)$ is reported figure 5.1. The obtained cloud shape figure is very similar to the observation values reported in figure 5 of [1]. It is worth to notice that the frequency f_c does not depend only on the distance. The reason is the coherence formula depends on the quadratic form $(r_m - r_k)^T \tilde{\Sigma}_k (r_m - r_k)$ and therefore on the angle between $(r_m - r_k)$ and the eigenvectors of the matrix $\tilde{\Sigma}_k$. This quadratic form reduces to the euclidian distance if $\tilde{\Sigma}_k \propto I_3$.

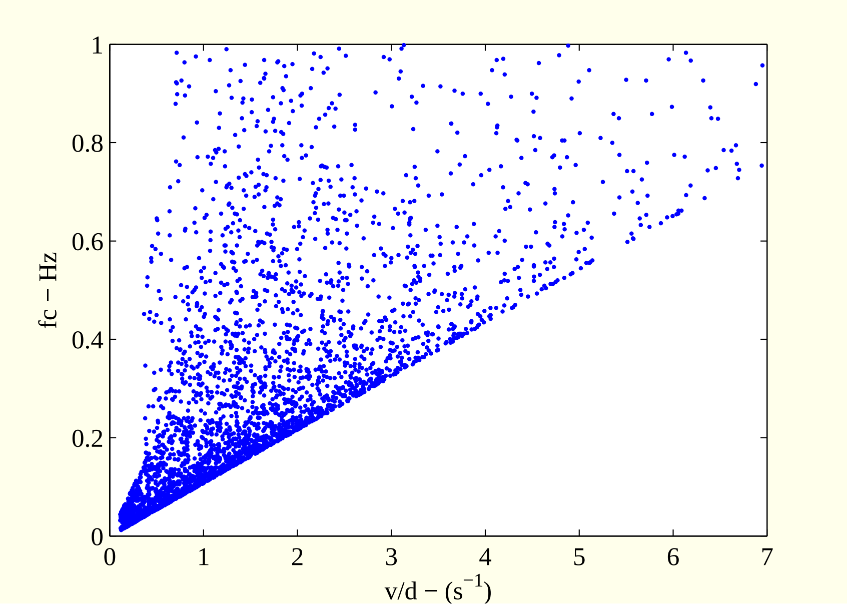

Figure 5.1: *Frequency f_c associated with the MSC level* 0.5 *as a function of v/d.*

3. Application to the *in-situ* calibration method

Coherence with/without NRS

We consider a sensor with its NRS, called sensor under test (SUT), and an other sensor with no NRS, called reference sensor (Sref). The NRS consists of $M = 96$ air identical inlets located at the ends of pipes. The scheme is reported on the figure 5.2. Thanks to the Gabrielson's algorithm [3], the transfer function $U(f)$ between any inlet and the cavity center of the NRS can be performed as a function of the geometrical structure of the NRS.

The spectral matrix of the $(M+1)$-ary input signal writes:

$$S_x(f) = \gamma_s(f)\mathbb{1}_{M+1}\mathbb{1}_{M+1}^T + \Gamma(f) \tag{5.15}$$

where $\gamma_s(f)$ is the spectral density of the coherent part, which is usually acoustic. By acoustic we mean that the wave has a velocity c of the order of magnitude of 300 m/s. Let us notice that the wavelength of this acoustic part is greater than 150 meter for frequencies under 2 Hz.

The spectral matrix $\Gamma(f)$, whose size is $(M+1)$, is given by the expression

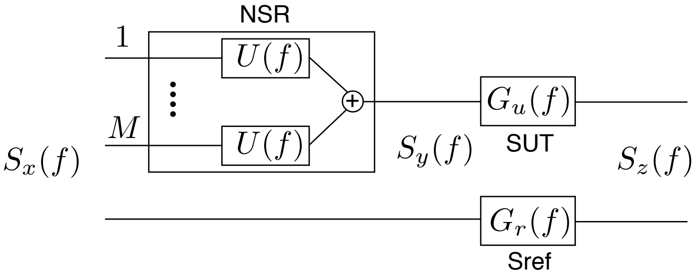

Figure 5.2: $S_x(f)$ *spectral matrix de dimension* $(M+1)$, $S_y(f)$ *spectral matrix de dimension* 2, $S_z(f)$ *spectral matrix de dimension* 2

(5.4) where $C(f)$ entries are given by (9.7) and

$$P(f) = \begin{bmatrix} \gamma_u^{1/2}(f) & 0 & \cdots & & 0 \\ 0 & \gamma_u^{1/2}(f) & \cdots & & 0 \\ \vdots & & & & \\ 0 & \cdots & 0 & \gamma_u^{1/2}(f) & 0 \\ 0 & \cdots & & 0 & \gamma_r^{1/2}(f) \end{bmatrix} \tag{5.16}$$

Here it is assumed that the noise has the same level $\gamma_u(f)$ on all NRS inlets but is different on the reference sensor denoted $\gamma_r(f)$.

It follows that the 2×2 spectral matrix associated to the signals at the inputs of the two sensors writes:

$$S_y(f) = K(f)S_x(f)K^H(f) \tag{5.17}$$

where

$$K(f) \quad = \quad \begin{bmatrix} U^*(f) & \dots & U^*(f) & 0 \\ 0 & \dots & 0 & 1 \end{bmatrix}$$

where we assume that the transfer function of the pipe of the reference sensor is 1. From (5.17) we derive that:

$$\begin{aligned} S_{y,11}(f) &= |U(f)|^2 \left(M^2\gamma_s(f) + \gamma_u(f)S_M(f)\right) \\ S_{y,22}(f) &= (\gamma_s(f) + \gamma_r(f)) \\ S_{y,12}(f) &= U^*(f)\left(M\gamma_s(f) + \gamma_u^{1/2}(f)\gamma_r^{1/2}(f)s_M(f)\right) \\ S_{y,21}(f) &= S^*_{y,12}(f) \end{aligned}$$

where

$$S_M(f) = \sum_{k=1}^{M}\sum_{k'=1}^{M} C_{k,k'} \quad \text{and} \quad s_M(f) = \sum_{k=1}^{M} C_{k,M+1}$$

Let us notice that for small values of ζ, the $(M+1)$ entries are spatially uncorrelated hence C is the identity leading to $S_M(f) = M$ and $s_M(f) = 0$.

We denote

$$S_z(f) = \begin{bmatrix} S_{UU}(f) & S_{UR}(f) \\ S_{RU}(f) & S_{RR}(f) \end{bmatrix}$$

with

$$\begin{aligned} S_{UU}(f) &= |G_u(f)|^2\,|U(f)|^2\,(M^2\gamma_s(f) + \gamma_u(f)S_M(f)) \\ S_{RR}(f) &= |G_r(f)|^2(\gamma_s(f) + \gamma_r(f)) \\ S_{UR}(f) &= G_u(f)G_r^*(f)U^*(f)\,(M\gamma_s(f) + \gamma_u^{1/2}(f)\gamma_r^{1/2}(f)s_M(f)) \\ S_{RU}(f) &= S_{UR}^*(f) \end{aligned}$$

Let us remark that a good NRS has $U(f) = 1/M$. That is not verified for high frequencies. Indeed it is known that the NRS presents a resonance above 2 Hz.

Consider at first that $U(f) = 1/M$, $S_M(f) = M$ and $s_M(F) = 0$ we obtain

$$\begin{aligned} S_{UU}(f) &= |G_u(f)|^2(\gamma_s(f) + \gamma_u(f)/M) \\ S_{RR}(f) &= |G_r(f)|^2(\gamma_s(f) + \gamma_r(f)) \\ S_{UR}(f) &= G_u(f)G_r^*(f)\gamma_s(f) \\ S_{RU}(f) &= S_{UR}^*(f) \end{aligned}$$

We recognize the classical expression which is the base of the *in-situ* calibration. We see also that the noise level is divided by M. Unfortunately, if this ratio $\gamma_r(f)/\gamma_r(f)$ is unknown, the system is underdetermined. The only way to circumvent the problem is to consider that the wind noise is very close to zero. Indeed if $\gamma_u(f) = \gamma_r(f) = 0$

$$G_u(f) = \frac{S_{UU}(f)}{S_{RU}(f)}\,G_r(f)$$

To test the zero-noise it is well-known that the MSC can be used but we see below that unfortunately in the frequency mid-range this test could fail, due to the factors $S_M(f)$ and $s_M(F)$.

MSC

The MSC writes:

$$\text{MSC}(f) = \frac{|S_{UR}(f)|^2}{S_{UU}(f)S_{RR}(f)} = \frac{|M\gamma_s(f) + \gamma_u^{1/2}(f)\gamma_r^{1/2}(f)s_M(f))|^2}{(M^2\gamma_s(f) + \gamma_u(f)S_M(f))(\gamma_s(f) + \gamma_r(f))} \quad (5.18)$$

We let $\rho_i(f) = \gamma_i(f)/\gamma_s(f)$.

- For high frequencies $s_M(f) \approx 0$ and $S_M(f) \approx M$, hence

  $$\mathrm{MSC}(f) \approx \frac{1}{(1+\rho_u(f)/M)(1+\rho_r(f))} \tag{5.19}$$

  and the level of noise on the NRS is divided by M,

- For very low frequencies $s_M(f) = M$ and $S_M(f) = M^2$, hence:

  $$\mathrm{MSC}(f) = \frac{(1+\sqrt{\rho_u(f)\rho_r(f)})^2}{(1+\rho_u(f))(1+\rho_r(f))} \tag{5.20}$$

  the MSC could be close to 1 even if $\rho_u(f)$ and $\rho_r(f)$ are different of 0.

- For mid-range, $s_M(f) \approx \lambda_1 M$ and $S_M(f) \approx \lambda_2 M^2$, hence

  $$\mathrm{MSC}(f) \approx \frac{(1+\lambda_1\sqrt{\rho_u(f)\rho_r(f)})^2}{(1+\lambda_2\rho_u(f))(1+\rho_r(f))} \tag{5.21}$$

  If λ_1 and λ_2 are close to 1, the MSC could be close to 1 even with $\rho_i(f) \neq 0$.

In conclusion MSC(f) close to 1 does not mean that $\rho_u(f)$ and $\rho_r(f)$ are close to 0 and in this case the system is still underdetermined.

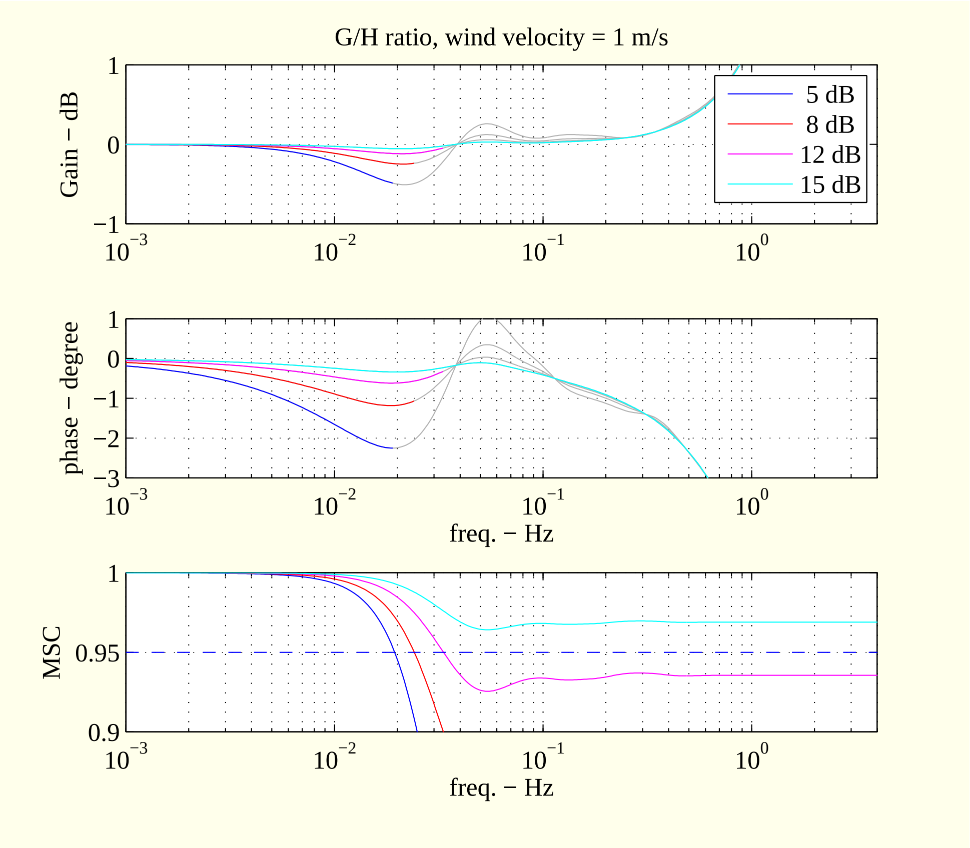

Figure 5.3: *$R(f)$ as a function of the frequency for different values of the* SNR*. The color curves report the values corresponding to an MSC greater than 0.95. For frequencies over 0.4 Hz up to 4 Hz and for low noise levels, the curves reproduce the response of the NRS. In particular on the RHS the curves are increasing, in relation with a known resonance effect. For frequencies under 0.08 Hz and even for low values of* SNR*, the coherence is very high, because the wind turbulences appear as spatially coherent.*

4. Results in I26

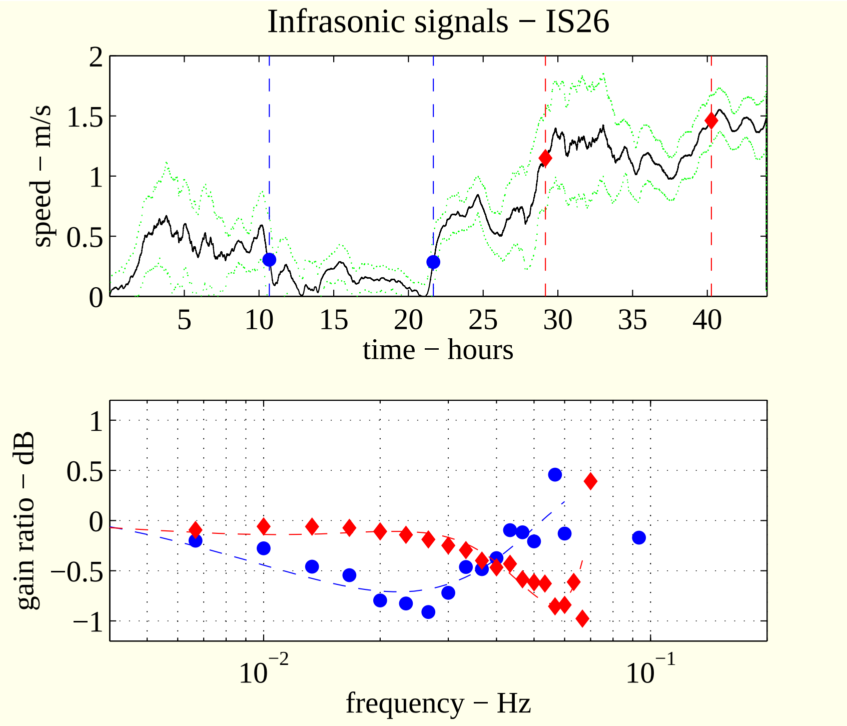

Figure 5.4: *Top figure: averaging of the wind speed on plain curve, standard deviation around the averaging in dotted curve. Markers indicate the two parts of interest. Bottom figure: curves associated to the two parts of interest. The shift between them has a ratio of about 4 which is about the ratio of the speed means od the two parts of interest.*

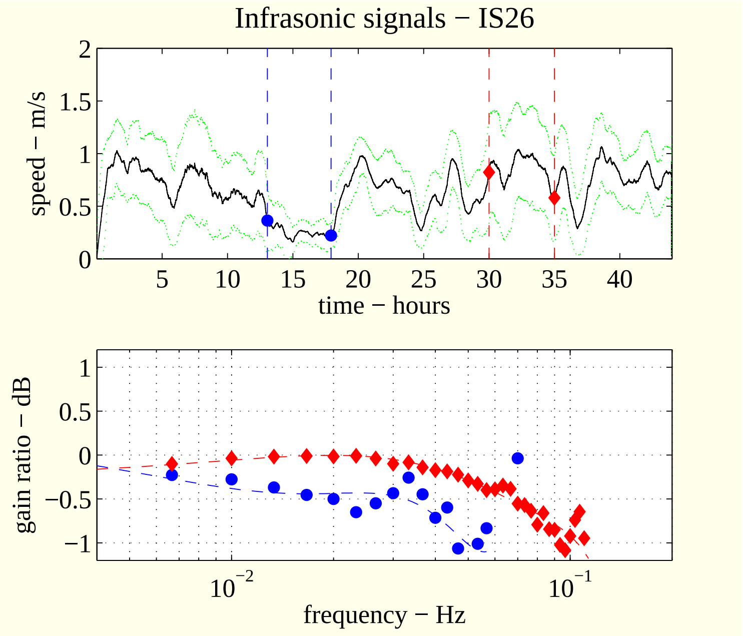

Figure 5.5: *see figure 5.4*

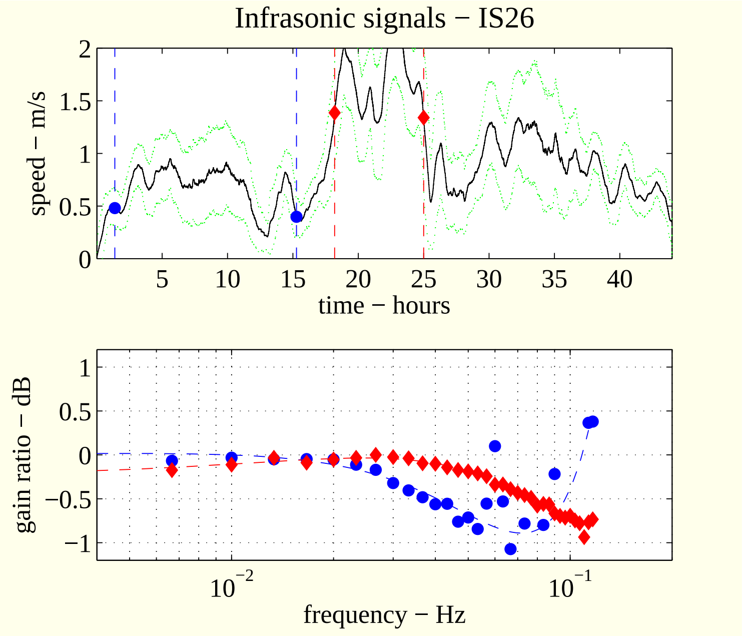

Figure 5.6: *see figure 5.4*

Chapter 6

Full process

The full process consists of

- move data from the IDC, using `execGPARSE.m` (thanks to P. Mialle). The data with Matlab format is saved. This data consists of a structure called `records` with the following items:
  - `data`: sequence of samples,
  - `Fs_Hz`: sampling frequency,
  - `stime`, `time`: start and end times,
  - `station`: I26C1–I26C8 or I26H1–I26H8
  - `channel`: 'BDF' or 'LKO' or 'LWD' or 'LWS'

  Typically in the following we use BDF on H and C. The three other channels are only provided on H1. Channel LWD yields the wind speed measurement. The wind sensor is located at about 2 meters above the infrasonic sensor.

- write the script `filtercharacteristics.m`, following the example given in Annex.
- run the program . . . . .   .

Part II

Annexes

Chapter 7

Wide sense stationary process

A multivariate process x_n is said to be a zero-mean wide-sense stationary (WSS) of second order iif $\mathbb{E}\left[x_n\right] = 0$, trace $\left(\mathbb{E}\left[x_n x_n^H\right]\right) < +\infty$ and the sequence of covariance matrices[c0] $R(h) = \mathbb{E}\left[x_{n+h} x_n^H\right]$ does not depend on n. Under very general conditions, its Fourier transform

$$\Gamma(f) = \sum_h R(h) e^{-2j\pi h f}, \quad \text{with} \quad f \in (0,1)$$

exists which is called the spectral matrix sequence. A fundamental property says that

$$\forall f, \quad \Gamma(f) \geq 0$$

The non-negativity of $\Gamma(f)$ implies that $\Gamma_{ij}(f) = \Gamma^*_{ji}(f)$ and $\Gamma_{ii}(f) \geq 0$. If x_n is real alors $\Gamma(f) = \Gamma(-f)$. For example for a bivariate process we have

$$\Gamma(f) = \begin{bmatrix} \Gamma_{11}(f) & \Gamma_{12}(f) \\ \Gamma_{21}(f) & \Gamma_{22}(f) \end{bmatrix}$$

where $\Gamma_{11}(f)$ and $\Gamma_{22}(f)$ are both positive functions. The coherence matrix is defined by

$$\begin{aligned} C(f) &= \begin{bmatrix} \Gamma_{11}^{-1/2}(f) & 0 \\ 0 & \Gamma_{22}^{-1/2}(f) \end{bmatrix} \begin{bmatrix} \Gamma_{11}(f) & \Gamma_{12}(f) \\ \Gamma_{21}(f) & \Gamma_{22}(f) \end{bmatrix} \begin{bmatrix} \Gamma_{11}^{-1/2}(f) & 0 \\ 0 & \Gamma_{22}^{-1/2}(f) \end{bmatrix} \\ &= \begin{bmatrix} 1 & C_{12}(f) \\ C_{21}(f) & 1 \end{bmatrix} \end{aligned}$$

with $C_{12}(f) = C^*_{2,1}(f)$. Let

$$\eta_{12}(f) = \frac{\Gamma_{12}(f)}{\sqrt{\Gamma_{11}(f)\Gamma_{22}(f)}} \tag{7.1}$$

[c0] The superscript H denotes the transposition-conjugaison, T the transposition and $*$ the conjugaison.

which is called the normalized cross spectral density and

$$\mathrm{MSC}(f) = \frac{|\Gamma_{12}(f)|^2}{\Gamma_{11}(f)\Gamma_{22}(f)} \tag{7.2}$$

which is called the Magnitude Squared Coherence (MSC) or shortly the coherence. A fundamental property says that

$$\forall f, \quad \mathrm{MSC}(f) \leq 1$$

and the equality occurs iff it exists a filter s.t. one signal of the bivariate process is the filtered expression of the other. In this case $\Gamma(f)$ is, up to a a multiplicative positive value, a projector of rank 1. A fundamental example is given by

$$\left\{ \begin{array}{rcl} x_{1,n} & = & g_{1,n} \star s_n \\ x_{2,n} & = & g_{2,n} \star s_n \end{array} \right. \quad \Leftrightarrow \quad x_n = \begin{bmatrix} x_{1,n} \\ x_{2,n} \end{bmatrix} = h_n \star s_n$$

where s_n is a monovariate WSS process with spectral density $\gamma_s(f)$. Then the spectral matrix of the bivariate process writes

$$\Gamma(f) = \gamma_s(f) G(f) G^H(f)$$

and

$$C(f) = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \end{bmatrix}$$

In the noisy case if the spectral matrix of the two noises is diagonal with respective spectral densities $\gamma_1(f)$ and $\gamma_2(f)$, the spectral matrix writes:

$$\Gamma(f) = \gamma_s(f) \begin{bmatrix} |G_1(f)|^2 & G_1(f)G_2^*(f) \\ G_1^*(f)G_2(f) & |G_2(f)|^2 \end{bmatrix} + \begin{bmatrix} \gamma_1(f)|G_1(f)|^2 & 0 \\ 0 & \gamma_2(f)|G_2(f)|^2 \end{bmatrix}$$

then

$$\begin{aligned} \mathrm{MSC}(f) &= \frac{|\Gamma_{1,2}(f)|^2}{\Gamma_{1,1}(f)\Gamma_{2,2}(f)} \\ &= \frac{1}{(1+\mathrm{SNR}_1^{-1}(f))(1+\mathrm{SNR}_2^{-1}(f))} \end{aligned}$$

where

$$\mathrm{SNR}_1(f) = \frac{\gamma_s(f)}{\gamma_1(f)} \quad \text{and} \quad \mathrm{SNR}_2(f) = \frac{\gamma_s(f)}{\gamma_2(f)}$$

All these expressions can be generalized for more than 2 dimensions.

Chapter 8

Theoretical results on spectral estimation

1. Non parametrical spectral estimation

Let us consider $x_{n=0:N-1}$ N successive values of a real WSS bivariate process with zero-mean and spectral matrix $\Gamma(f)$. Its DFT writes

$$X_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x_n e^{-2j\pi kn/N}$$

It can be shown that, for large N, and for $k \neq 0$ and $k \neq N/2$:

$$X_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}_c(0, \Gamma_k)$$

where $\Gamma_k = \Gamma(k/N)$ and where $\mathcal{N}_c$ denotes the complex circular gaussian distribution. The periodogram is defined by

$$P_k = X_k X_k^H$$

In the following section 2., it is shown that the expectation of the periodogram P_k is Γ_k, but unfortunately the dispersion around Γ_k never goes to 0 when N goes to infinity. The periodogram is not a consistent estimate of Γ_k. A fundamental way to have consistency is to smooth the periodogram (smoothed peridogram). An other approach consists to segment the data in $(2M + 1)$ 50%-overlapping blocks and averaging the $2M + 1$ periodograms, applying a window on each block. That is called Welch's approach in the literature. The 2 approaches are very similar.

For the smoothed periodogram estimator at any frequency f writes:

$$\widehat{\Gamma}_k = \sum_{m=0}^{2M} W_m P_{k-M+m} \tag{8.1}$$

where k is an integer s.t. $\frac{k-1/2}{N} \leq f < \frac{k+1/2}{N}$. That leads to a mean square error (MSE) that writes

$$\text{MSE}_k \;=\; \mathcal{G}_k + \mathcal{B}_k \mathcal{B}_k^H \tag{8.2}$$

where the bias

$$\mathcal{B}_k = \Gamma_k - \sum_{k=0}^{2M} W_m \Gamma_{k-M+m} \tag{8.3}$$

and the covariance

$$\mathcal{G}_k = \sum_{k=0}^{2M} W_m^2 \text{diag}\left(\Gamma_{k-M+m}\right) \text{diag}\left(\Gamma_{k-M+m}\right)^H \tag{8.4}$$

The choice of the window and its length results from a compromise between bias and variance. Unfortunately this compromise is related to the shape of the spectrum which is *unknown.* Using (8.2) we derive a confidence interval replacing deterministic values by their estimates and assuming a gaussian distribution.

2. More periodogram properties

Let us consider $x_{n=0:N-1}$ a sequence of N consecutive values of a real wide-sense stationary bivariate process with zero-mean and spectral matrix $\Gamma(f)$. An fundamental property says that $\Gamma(f) \geq 0$ for any value of f. The DFT of the sequence writes

$$X_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x_n e^{-2j\pi kn/N}$$

It can be shown that, for large N, and for $k \neq 0$ and $k \neq N/2$:

$$X_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}_c(0, \Gamma_k)$$

where $\Gamma_k = \Gamma(k/N)$ and where $\mathcal{N}_c$ denotes the complex circular gaussian distribution whose the probability density writes:

$$p_{X_k}(x_k) = \frac{1}{\pi^2|\Gamma_k|} e^{-x_k^H \Gamma_k^{-1} x_k}$$

Recall that for any complex gaussian random vector with zero-mean and covariance Γ we have by definition:

$$\mathbb{E}\left[XX^H\right] = \Gamma$$

and

$$\mathbb{E}\left[XX^T\right] = 0$$

and if $X_{1\cdots 4}$ denote 4 gaussian complex zero-mean r.v., we have

$$\begin{aligned} \mathbb{E}\left[X_1^{\beta_1} X_2^{\beta_2} X_3^{\beta_3} X_4^{\beta_4}\right] &= \mathbb{E}\left[X_1^{\beta_1} X_2^{\beta_2}\right] \mathbb{E}\left[X_3^{\beta_3} X_4^{\beta_4}\right] \qquad (8.5) \\ &+\mathbb{E}\left[X_1^{\beta_1} X_3^{\beta_3}\right] \mathbb{E}\left[X_2^{\beta_2} X_4^{\beta_4}\right] + \mathbb{E}\left[X_1^{\beta_1} X_4^{\beta_4}\right] \mathbb{E}\left[X_2^{\beta_2} X_3^{\beta_3}\right] \end{aligned}$$

where β_i is either “star” or “not-star”.

The formula (8.5) is very useful to perform the second order moments of the periodogram defined by

$$P_k = X_k X_k^H = \begin{bmatrix} X_1 X_1^* & X_1 X_2^* \\ X_1^* X_2 & X_2 X_2^* \end{bmatrix} \qquad (8.6)$$

Let us remark that if we multiply X_1 and X_2 by $e^{j\theta}$ the matrix P_k is unchanged. Then given P_k we can determine X_1 and X_2 up to a constant of modulus 1. It is possible to derive the distributions of P_k entries from the distribution of X_k. Alleviating the notations by omitting the index k, the distribution of these two complex r.v. X_1, and X_2 writes

$$\begin{aligned} p_{X_1,X_2}(x_1,x_2) &= \frac{1}{\pi^2(\Gamma_{1,1}\Gamma_{2,2} - |\Gamma_{1,2}|^2)} \\ &\exp\left(-\frac{|x_1|^2\Gamma_{2,2} + |x_2|^2\Gamma_{1,1} - 2|x_1|\,|x_2||\Gamma_{2,1}|\cos(\alpha)}{(\Gamma_{1,1}\Gamma_{2,2} - |\Gamma_{1,2}|^2)}\right) \end{aligned}$$

where $\alpha = \arg x_1 - \arg x_2 - \arg \Gamma_{1,2}$.

In practical cases we often need only the second order moments.

Example 1 (Variance of $P_{1,1,k}$) *Let us recall that $P_{1,1,k} \geq 0$. It comes*

$$\mathbb{E}\left[P_{1,1,k}\right] = \mathbb{E}\left[X_1 X_1^*\right] = \Gamma_{1,1} \geq 0$$

and

$$\mathbb{E}\left[P_{1,1,k}P^*_{1,1,k}\right] = \mathbb{E}\left[X_{1,k}X^*_{1,k}X^*_{1,k}X_{1,k}\right] = 2\mathbb{E}\left[X_{1,k}X^*_{1,k}\right]\mathbb{E}\left[X_{1,k}X^*_{1,k}\right] + 0 = 2\Gamma^2_{1,1,k}$$

then

$$\mathrm{var}\left(P_{1,1,k}\right) = \mathbb{E}\left[P_{1,1,k}P^*_{1,1,k}\right] - \mathbb{E}\left[P_{1,1,k}\right]\mathbb{E}\left[P_{1,1,k}\right]^* = \Gamma^2_{1,1,k}$$

Example 2 (Variance of $P_{1,2,k}$) *Consider we want to determine the second order moments of $P_{1,2,k}$ which is complex. It comes*

$$\mathbb{E}\left[P_{1,2,k}\right] = \mathbb{E}\left[X_{1,k}X^*_{2,k}\right] = \Gamma_{1,2}$$

and

$$\begin{aligned}\mathbb{E}\left[P_{1,2,k}P^*_{1,2,k}\right] &= \mathbb{E}\left[X_{1,k}X^*_{2,k}X^*_{1,k}X_{2,k}\right] \\ &= \mathbb{E}\left[X_{1,k}X^*_{1,k}\right]\mathbb{E}\left[X_{2,k}X^*_{2,k}\right] + \mathbb{E}\left[X_{1,k}X^*_{2,k}\right]\mathbb{E}\left[X_{2,k}X^*_{1,k}\right] + 0 \\ &= \Gamma_{1,1,k}\Gamma_{2,2,k} + \Gamma_{1,2}\Gamma^*_{1,2}\end{aligned}$$

then

$$\mathrm{var}\left(P_{1,2,k}\right) = \mathbb{E}\left[P_{1,2,k}P^*_{1,2,k}\right] - \mathbb{E}\left[P_{1,2,k}\right]\mathbb{E}\left[P^*_{1,2,k}\right] = \Gamma_{1,1,k}\Gamma_{2,2,k}$$

On the other hand

$$\mathbb{E}\left[P_{1,2,k}P_{1,2,k}\right] = \mathbb{E}\left[X_1X^*_2X_1X^*_2\right] = 2\Gamma^2_{1,2,k}$$

which is complex.

Example 3 *We want to determine the variance of $P_{1,2,k,R}$. To alleviate the notations we omit the index k and let $G = P_{1,2}$, $G = P_{1,2}$ and $G_R = P_{1,2,R}$. Then*

$$G_R = \frac{1}{2}(G + G^*)$$

at first

$$\mathbb{E}\left[G_R\right] = \frac{1}{2}(\Gamma_{1,2} + \Gamma^*_{1,2}) = \Gamma_{1,2,R}$$

then

$$\mathbb{E}\left[G^2_R\right] = \frac{1}{4}\mathbb{E}\left[GG + G^*G^* + 2GG^*\right]$$

at first

$$\begin{aligned}\mathbb{E}\left[G^2\right] &= \mathbb{E}\left[X_1X^*_2X_1X^*_2\right] \\ &= \mathbb{E}\left[X_1X^*_2\right]\mathbb{E}\left[X_1X^*_2\right] + 0 + \mathbb{E}\left[X_1X^*_2\right]\mathbb{E}\left[X_1X^*_2\right] \\ &= 2(\Gamma_{1,2})^2\end{aligned}$$

similarly

$$\mathbb{E}\left[G^*G^*\right] = 2(\Gamma_{1,2}^*)^2$$

and

$$\begin{aligned} 2\mathbb{E}\left[GG^*\right] &= 2\mathbb{E}\left[X_1X_2^*X_1^*X_2\right] \\ &= 2|\Gamma_{1,2}|^2 + 2\Gamma_{1,1}\Gamma_{2,2} + 0 \end{aligned}$$

then

$$\mathrm{var}\left(G_{1,2,R}\right) = \underbrace{\frac{1}{2}(\Gamma_{1,2})^2 + \frac{1}{2}(\Gamma_{1,2}^*)^2}_{\Gamma_{1,2,R}^2 - \Gamma_{1,2,I}^2} + \frac{1}{2}|\Gamma_{1,2}|^2 + \frac{1}{2}\Gamma_{1,1}\Gamma_{2,2} - \Gamma_{1,2,R}^2$$

Hence

$$\mathrm{var}\left(P_{1,2,k,R}\right) = \frac{1}{2}\Gamma_{1,1}\Gamma_{2,2} + \frac{1}{2}\Gamma_{1,2,R}^2 - \frac{1}{2}\Gamma_{1,2,I}^2$$

Similarly

$$\mathrm{var}\left(P_{1,2,k,I}\right) = \frac{1}{2}\Gamma_{1,1}\Gamma_{2,2} + \frac{1}{2}\Gamma_{1,2,I}^2 - \frac{1}{2}\Gamma_{1,2,R}^2$$

Properties of the periodogram

Similarly we do in the previous examples, we have

$$\mathbb{E}\left[P_{k,1,1}\right] = \Gamma_{k,1,1}, \qquad \mathrm{var}\left(P_{k,1,1}\right) = \Gamma_{k,1,1}^2 \tag{8.7}$$

$$\mathbb{E}\left[P_{k,2,2}\right] = \Gamma_{k,2,2}, \qquad \mathrm{var}\left(P_{k,2,2}\right) = \Gamma_{k,2,2}^2 \tag{8.8}$$

$$\mathbb{E}\left[P_{k,1,2}\right] = \Gamma_{k,1,2} \qquad \mathrm{var}\left(P_{k,1,2}\right) = \Gamma_{k,1,1}\Gamma_{k,2,2} \tag{8.9}$$

That means that the expectation of the periodogram P_k is Γ_k, that is well! But, unfortunately, the dispersion around Γ_k not only never goes to 0 when N goes to infinity, but it is of the order of magnitude of what we want to estimate. Consequently the periodogram is not a good estimator of Γ_k.

But all good estimators of Γ_k are built on the periodogram either by smoothing or by averaging. For example if we use a smoothing approach the estimator writes

$$\widehat{\Gamma}_k = \sum_{m=0}^{2M} W_m P_{k-M+m} \tag{8.10}$$

with the following properties:

- $\sum_{m=0}^{2M} W_m = 1$
- $\overline{W}_M^2 = \sum_{m=0}^{2M} W_m^2 \to 0, \quad M \to \infty$
- $M = N^\beta$ with $\beta \in (0,1)$, typically β is between 0.2 and 0.3. That induces that both M and N/M go to infinity when N goes to infinity.

It follows that

$$\mathbb{E}\left[\widehat{\Gamma}_k\right] = \sum_{k=0}^{2M} W_m \Gamma_{k-M+m}$$

and therefore the estimator is biased with bias given by

$$\mathcal{B}_k = \Gamma_k - \sum_{k=0}^{2M} W_m \Gamma_{k-M+m} \tag{8.11}$$

From the properties of P_k we derive that the variance of the 4 components of $\widehat{\Gamma}_k$ are given by the 4 elements of

$$\mathcal{G}_k = \sum_{k=0}^{2M} W_m^2 \text{diag}\left(\Gamma_{k-M+m}\right) \text{diag}\left(\Gamma_{k-M+m}\right)^H$$

It results that the mean square error (MSE) writes as the 2 by 2 matrix

$$\text{MSE}_k \quad = \quad \mathcal{G}_k + \mathcal{B}_k \mathcal{B}_k^H$$

An estimate of the MSE can be obtained replacing the values of Γ_k by their estimates. Generally the variance decreases when the bias increases. Therefore the choice of a smoothing window results in a trade-off between the bias and the variance.

It is worth to noting that $\mathcal{B}_k$, given by (8.11) represents the bias of the estimators. Naively you could imagine to subtract the bias to the estimator, replacing in (8.11) the unknown values by the observed values. Of course the expectation of this quantity is 0 but for one outcome it could large and therefore by subtracting it we will re-introduce more variance.

It is worth to noting that the spectrum estimator variance decreases in $1/(2M+1)$ where M is the length of the smoothing window for smoothed periodogram and the number of windows for Welch's approach, but it is not related to the data number N which is assumed to be *infinite*. In the practical cases, we derive the value of M and the window shape from a given accuracy. Then N is chosen to be huge w.r.t. M as for example $N = M^3$.

3. Second order moments for the smoothed periodogram

We omit the index k and let

$$\widehat{\Gamma} = \frac{1}{2M+1}\sum_{n=0}^{2M+1} X_n X_n^H = \begin{bmatrix} \widehat{\Gamma}_{1,1} & \widehat{\Gamma}_{1,2} \\ \widehat{\Gamma}^*_{1,2} & \widehat{\Gamma}_{2,2} \end{bmatrix} \to \Gamma = \begin{bmatrix} \Gamma_{1,1} & \Gamma_{1,2} \\ \Gamma^*_{1,2} & \Gamma_{2,2} \end{bmatrix} \tag{8.12}$$

where $X_{0\cdots 2M}$ are i.i.d. bivariate gaussian complex vector with zero-mean and covariance Γ. $\Gamma_{1,1}$ and $\Gamma_{2,2}$ are real. We let $\Gamma_{1,2} = \Gamma_{1,R} + j\Gamma_{1,2,I}$.

Remark 1 *Let us remark that the expression* (8.12) *is as an approximation of the smoothed periodogram. Indeed (i) the window is assumed to be the rectangular window and more important (ii) the r.v. X_n are assumed to have the same variance whereas in the equation* (8.10) *the r.v. have different variance, saying Γ_k. For non identically distributed r.v. the limit central theorem is a little bit more complicate.*

Here, for the two first order moments we have to perform the two first order moments of $X_n X_n^H$ and then divide by $1/2M+1$ for the second order moments. We have

$$\mathbb{E}\left[\widehat{\Gamma}\right] = \Gamma$$

$$\mathrm{var}\left(\widehat{\Gamma}_{1,1}\right) = \frac{1}{2M+1}\Gamma^2_{1,1}$$

$$\mathrm{var}\left(\widehat{\Gamma}_{2,2}\right) = \frac{1}{2M+1}\Gamma^2_{2,2}$$

$$\mathrm{var}\left(\widehat{\Gamma}_{1,2}\right) = \frac{1}{2M+1}\Gamma_{1,1}\Gamma_{2,2}$$

$$\mathrm{var}\left(\widehat{\Gamma}_{1,2,R}\right) = \frac{1}{2M+1}\frac{1}{2}\left(|\Gamma_{1,2}|^2 + \Gamma_{1,1}\Gamma_{2,2}\right)$$

and

$$\mathrm{var}\left(\widehat{\Gamma}_{1,2,I}\right) = \frac{1}{2M+1}\frac{1}{2}\left(|\Gamma_{1,2}|^2 + \Gamma_{1,1}\Gamma_{2,2}\right)$$

It can be shown that $\Gamma_{1,2,R}$ and $\Gamma_{1,2,R}$ are correlated with

$$\mathrm{cov}\left(\widehat{\Gamma}_{1,2,R}, \widehat{\Gamma}_{1,2,I}\right) = \Gamma_{1,2,R}\Gamma_{1,2,I}$$

4. Approximate variances

δ-method

The so called δ-method allows to derive the covariance of

$$Z = f(Y)$$

from the covariance of Y where $f : \mathbb{R}^m \mapsto \mathbb{R}^q$. We let $\mu_Y = \mathbb{E}\left[Y\right]$. Using the first order Taylor expansion of f in the neighborhood of μ_Y, we write

$$Z \approx f(\mu_Y) + J_{\mu_Y}(Y - \mu_Y)$$

where J_{μ_Y} is the Jacobian of f, which is a matrix of size $m \times q$, performed in $y = \mu_Y$. Therefore at first order

$$\mathbb{E}\left[Z\right] \approx f(\mu_Y) + J_{\mu_Y}\mathbb{E}\left[Y - \mu_Y\right] = f(\mu_Y) + 0$$

then

$$Z - \mathbb{E}\left[Z\right] \approx J_{\mu_Y}(Y - \mu_Y)$$

therefore, according to the definition $\operatorname{cov}(Z) = \mathbb{E}\left[(Z - \mathbb{E}\left[Z\right])(Z - \mathbb{E}\left[Z\right])^H\right]$, we have

$$\operatorname{cov}(Z) \approx J_{\mu_Y}\operatorname{cov}(Y)\, J_{\mu_Y}^H$$

Approximate variance of $\arg\widehat{\lambda}_k$

$$\tan(\phi) = \frac{\Gamma_{1,2,I}}{\Gamma_{1,2,R}}$$

Then

$$d\phi = \frac{\Gamma_{1,2,R}d\Gamma_{1,2,I} - \Gamma_{1,2,I}d\Gamma_{1,2,R}}{|\Gamma_{1,2}|^2}$$

Therefore

$$\operatorname{var}\left(\widehat{\phi}\right) = \frac{1}{|\Gamma_{1,2}|^4}\left(\Gamma_{1,2,R}^2\operatorname{var}\left(\widehat{\Gamma}_{1,2,I}\right) + \Gamma_{1,2,I}^2\operatorname{var}\left(\widehat{\Gamma}_{1,2,R}\right) - 2\Gamma_{1,2,R}\Gamma_{1,2,I}\operatorname{var}\left(\widehat{\Gamma}_{1,2,R}, \widehat{\Gamma}_{1,2,I}\right)\right)$$

Using that

$$\operatorname{var}\left(\widehat{\Gamma}_{1,2,R}\right) = \frac{1}{2}\Gamma_{1,1}\Gamma_{2,2} + \frac{1}{2}\Gamma_{1,2,R}^2 - \frac{1}{2}\Gamma_{1,2,I}^2$$

$$\operatorname{var}\left(\widehat{\Gamma}_{1,2,I}\right) = \frac{1}{2}\Gamma_{1,1}\Gamma_{2,2} + \frac{1}{2}\Gamma_{1,2,I}^2 - \frac{1}{2}\Gamma_{1,2,R}^2$$

and

$$\operatorname{cov}\left(\widehat{\Gamma}_{1,2,R}, \widehat{\Gamma}_{1,2,I}\right) \;=\; \Gamma_{1,2,R}\Gamma_{1,2,I}$$

we get

$$\operatorname{var}\left(\widehat{\phi}\right) = \frac{1-\mu}{2\mu}$$

Now if we smooth on a rectangular window with weights $1/(2M+1)$ see equation (**??**), the variance is given by

$$\operatorname{var}\left(\widehat{\phi}\right) = \frac{1}{2M+1}\frac{1-\mu}{2\mu}$$

We can approximate the distribution of $\widehat{\phi}$ by a gaussian with mean ϕ and variance $\frac{1}{2M+1}\frac{1-\mu}{2\mu}$. The green curve of the third figure of **??** uses this approximation.

Approximate variance of $|\widehat{\lambda}_k^{(1)}|$

We have

$$R_1 = \frac{\sqrt{\Gamma_{1,2,R}^2 + \Gamma_{1,2,I}^2}}{\Gamma_{2,2}}$$

Then

$$\begin{aligned} dR_1 &= \frac{\Gamma_{1,2,R}d\Gamma_{1,2,R} + \Gamma_{1,2,I}d\Gamma_{1,2,I}}{\Gamma_{2,2}\sqrt{\Gamma_{1,2,R}^2 + \Gamma_{1,2,I}^2}} - \frac{\sqrt{\Gamma_{1,2,R}^2 + \Gamma_{1,2,I}^2}}{\Gamma_{2,2}^2}d\Gamma_{2,2} \\ &= \frac{\Gamma_{2,2}\Gamma_{1,2,R}d\Gamma_{1,2,R} + \Gamma_{2,2}\Gamma_{1,2,I}d\Gamma_{1,2,I} - |\Gamma_{1,2}|^2 d\Gamma_{2,2}}{\Gamma_{2,2}^2|\Gamma_{1,2}|} \end{aligned}$$

Therefore we have to determine the expression of the variances of $\widehat{\Gamma}_{1,2,R}$, $\widehat{\Gamma}_{1,2,I}$, $\widehat{\Gamma}_{2,2}$, and the 3 covariances $\operatorname{cov}\left(\widehat{\Gamma}_{1,2,R}, \widehat{\Gamma}_{1,2,I}\right)$, $\operatorname{cov}\left(\widehat{\Gamma}_{1,2,R}, \widehat{\Gamma}_{2,2}\right)$ and $\operatorname{cov}\left(\widehat{\Gamma}_{1,2,I}, \widehat{\Gamma}_{2,2}\right)$.

$$\mathbb{E}\left[\widehat{\Gamma}_{1,2,R}, \widehat{\Gamma}_{2,2}\right] = \frac{1}{2}\mathbb{E}\left[(\widehat{\Gamma}_{1,2} + \widehat{\Gamma}_{1,2}^*)\widehat{\Gamma}_{2,2}\right] = \frac{1}{2}\mathbb{E}\left[(X_1X_2^* + X_1^*X_2)X_2X_2^*\right]$$

$$\frac{1}{2}\mathbb{E}\left[X_1X_2^*X_2X_2^*\right] + \frac{1}{2}\mathbb{E}\left[X_1^*X_2X_2X_2^*\right] = \Gamma_{1,2}\Gamma_{2,2} + \Gamma_{1,2}^*\Gamma_{2,2} = 2\Gamma_{1,2,R}\Gamma_{2,2}$$

$$\operatorname{cov}\left(\widehat{\Gamma}_{1,2,R}, \widehat{\Gamma}_{2,2}\right) = \Gamma_{1,2,R}\Gamma_{2,2}$$

Similarly

$$\operatorname{cov}\left(\widehat{\Gamma}_{1,2,I}, \widehat{\Gamma}_{2,2}\right) = \Gamma_{1,2,I}\Gamma_{2,2}$$

Then

$$\operatorname{var}(R_1) = \frac{1}{\Gamma_{2,2}^4|\Gamma_{1,2}|^2}(A+B+C+D+E+F) \tag{8.13}$$

$$A = \Gamma_{2,2}^2\Gamma_{1,2,R}^2\left(\frac{1}{2}\Gamma_{1,1}\Gamma_{2,2} + \frac{1}{2}\Gamma_{1,2,R}^2 - \frac{1}{2}\Gamma_{1,2,I}^2\right)$$

$$B = \Gamma_{2,2}^2\Gamma_{1,2,I}^2\left(\frac{1}{2}\Gamma_{1,1}\Gamma_{2,2} + \frac{1}{2}\Gamma_{1,2,I}^2 - \frac{1}{2}\Gamma_{1,2,R}^2\right)$$

$$C = |\Gamma_{1,2}|^4\Gamma_{2,2}^2$$

$$D = 2\Gamma_{2,2}^2|\Gamma_{1,2}|^2\Gamma_{1,2,R}\Gamma_{1,2,I}$$

$$E = -2|\Gamma_{1,2}|^2\Gamma_{2,2}^2\Gamma_{1,2,R}^2$$

$$F = -2|\Gamma_{1,2}|^2\Gamma_{2,2}^2\Gamma_{1,2,I}^2$$

$$E+F = -2\Gamma_{2,2}^2|\Gamma_{1,2}|^4$$

$$A+B = \Gamma_{2,2}^2\left(\frac{1}{2}\Gamma_{1,1}\Gamma_{2,2}|\Gamma_{1,2}|^2 + \frac{1}{2}|\Gamma_{1,2}|^4 - 2\Gamma_{1,2,R}^2\Gamma_{1,2,I}^2\right)$$

$$A+B+D+E+F = \Gamma_{2,2}^2\left(\frac{1}{2}\Gamma_{1,1}\Gamma_{2,2}|\Gamma_{1,2}|^2 + \frac{1}{2}|\Gamma_{1,2}|^4 - 2|\Gamma_{1,2}|^4\right)$$

$$\operatorname{var}\left(|\Gamma_{1,2}|/\widehat{\Gamma}_{2,2}\right) = \frac{1}{2}\left(\frac{\Gamma_{1,1}\Gamma_{2,2} - |\Gamma_{1,2}|^2}{\Gamma_{2,2}^2}\right)$$

Approximate variance of $|\widehat{\lambda}_k^{(2)}|$

From the previous section we have

$$\operatorname{var}\left(|\widehat{\Gamma}_{2,1}|/\widehat{\Gamma}_{1,1}\right) = \frac{1}{2}\left(\frac{\Gamma_{1,1}\Gamma_{2,2} - |\Gamma_{1,2}|^2}{\Gamma_{1,1}^2}\right)$$

then

$$\operatorname{var}\left(\widehat{\Gamma}_{1,1}/|\widehat{\Gamma}_{2,1}|\right) = \frac{1}{2}\left((\Gamma_{1,1}\Gamma_{2,2} - |\Gamma_{1,2}|^2)\frac{\Gamma_{1,1}^2}{|\Gamma_{1,2}|^4}\right)$$

Summary

We can rewritten the 2 last expressions using

$$\mu = \frac{|\Gamma_{1,2}|^2}{\Gamma_{1,1}\Gamma_{2,2}}$$

and the averaging on $(2M+1)$:

$$\operatorname{var}\left(|\widehat{\Gamma}_{1,2}|/\widehat{\Gamma}_{2,2}\right) = \frac{1}{2(2M+1)}\frac{\Gamma_{1,1}}{\Gamma_{2,2}}(1-\mu) \tag{8.14}$$

and

$$\operatorname{var}\left(\widehat{\Gamma}_{1,1}/|\widehat{\Gamma}_{2,1}|\right) = \frac{1}{2(2M+1)}\frac{\Gamma_{1,1}}{\Gamma_{2,2}}\frac{1-\mu}{\mu^2} \tag{8.15}$$

Where μ is close to 1, the 2 formulas are equivalent.

Estimator based on eigen-decomposition (uncompleted)

$$\lambda_k^{(3)} = \frac{\Gamma_{1,1} - \Gamma_{2,2} + ((\Gamma_{1,1} - \Gamma_{2,2})^2 + 4|\Gamma_{1,2}|^2)^{1/2}}{2\Gamma_{1,2}^*}$$

5. Some distributions related to the smoothed periodogram

Wiskart's distribution

We consider the sequence of $2M+1$ bivariate random gaussian complex independent variables

$$X_m \overset{\text{i.i.d.}}{\sim} \mathcal{N}_c(0, \Gamma)$$

We let

$$\Gamma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 e^{j\theta} \\ \rho\sigma_1\sigma_2 e^{-j\theta} & \sigma_2^2 \end{bmatrix}$$

where $\sigma_1 \geq 0$, $\sigma_2 \geq 0$, $\rho \geq 0$ and $\theta \in (0, 2\pi)$, and

$$\widehat{\Gamma} = \frac{1}{2M+1}\sum_{m=1}^{2M+1} X_m X_m^H = \begin{bmatrix} A_1 & Re^{j\Phi} \\ Re^{-j\Phi} & A_2 \end{bmatrix}$$

where $A_1 \geq 0$, $A_2 \geq 0$, $R \geq 0$ and $\Phi \in (0, 2\pi)$. We also have $R^2 \leq A_1A_2$ thanks to the positivity of $\widehat{\Gamma}$.

We let $K = 2M + 1$. The distribution of (A_1, A_2, R, Φ) writes:

$$p_{A_1A_2R\Phi}(a_1, a_2, r, \phi) = \frac{r}{\pi(K-1)!(K-2)!|\Gamma|^K}(a_1a_2 - r^2)^{K-2} \tag{8.16}$$

$$\exp\left\{-\frac{1}{|\Gamma|}\left(\sigma_2^2 a_1 + \sigma_1^2 a_2 - 2r\rho\sigma_1\sigma_2\cos(\theta - \phi)\right)\right\} \tag{8.17}$$

$$\mathbb{1}(a_1 \geq 0) \times \mathbb{1}(a_2 \geq 0) \times \mathbb{1}(\sqrt{a_1a_2} \geq r \geq 0) \times \mathbb{1}(\phi \in (0, 2\pi)) \tag{8.18}$$

Ratio distributions

We consider at first the ratio $|\widehat{\Gamma}_{1,2}|/\widehat{\Gamma}_{2,2}$, expression (**??**). Integrating expression (8.23) w.r.t. ϕ gives

$$\begin{aligned} p_{A_1A_2R}(a_1, a_2, r) &= \frac{2r}{(K-1)!(K-2)!|\Gamma|^K}(a_1a_2 - r^2)^{K-2} \\ &\quad \exp\left\{-\frac{1}{|\Gamma|}\left(\sigma_2^2 a_1 + \sigma_1^2 a_2\right)\right\} I_0(2r\rho\sigma_1\sigma_2/|\Gamma|) \end{aligned}$$

It is remarkable to notice that this expression does not depend on θ. We let

$$\left\{\begin{array}{rcl} Y_1 &=& A_1 \\ Y_2 &=& A_2 \\ T &=& R/A_2 \end{array}\right. \Leftrightarrow \left\{\begin{array}{rcl} A_1 &=& Y_1 \\ A_2 &=& Y_2 \\ R &=& TY_2 \end{array}\right. \tag{8.19}$$

whose Jacobian is Y_2 which is positive. Because $R^2 \leq A_1A_2$ it follows that $Y_1 - T^2Y_2 \geq 0$. Then

$$\begin{aligned} p_{Y_1Y_2T}(y_1, y_2, t) &= \frac{2ty_2^2}{(K-1)!(K-2)!|\Gamma|^K}(y_1y_2 - t^2y_2^2)^{K-2}\mathbb{1}(y_1 - t^2y_2 \geq 0) \\ &\quad \exp\left\{-\frac{1}{|\Gamma|}\left(\sigma_2^2 y_1 + \sigma_1^2 y_2\right)\right\} I_0(2ty_2\rho\sigma_1\sigma_2/|\Gamma|) \end{aligned}$$

Integrating w.r.t. y_1 writes

$$\begin{aligned} p_{Y_2T}(y_2, t) &= \int_0^{+\infty} \frac{2ty_2^K}{(K-1)!(K-2)!|\Gamma|^K}(y_1 - t^2y_2)^{K-2}\mathbb{1}(y_1 - t^2y_2 \geq 0) \\ &\quad \exp\left\{-\frac{1}{|\Gamma|}\left(\sigma_2^2 y_1 + \sigma_1^2 y_2\right)\right\} I_0(2ty_2\rho\sigma_1\sigma_2/|\Gamma|)dy_1 \end{aligned}$$

We let $u = y_1 - t^2y_2$, it get

$$\begin{aligned} p_{Y_2T}(y_2, t) &= \int_0^{+\infty} \frac{2ty_2^K}{(K-1)!(K-2)!|\Gamma|^K}u^{K-2} \\ &\quad \exp\left\{-\frac{1}{|\Gamma|}\left(\sigma_2^2 u + (t^2\sigma_2^2 + \sigma_1^2)y_2\right)\right\} I_0(2ty_2\rho\sigma_1\sigma_2/|\Gamma|)du \end{aligned}$$

and

$$\begin{aligned} p_{Y_2T}(y_2,t) &= \frac{2ty_2^K}{(K-1)!(K-2)!|\Gamma|^K}\exp\left\{-\frac{1}{|\Gamma|}(t^2\sigma_2^2+\sigma_1^2)y_2\right\} I_0(2ty_2\rho\sigma_1\sigma_2/|\Gamma|) \\ & \int_0^{+\infty} u^{K-2}\exp\left\{-\frac{1}{|\Gamma|}\sigma_2^2 u\right\} du \end{aligned}$$

and

$$\begin{aligned} p_{Y_2T}(y_2,t) &= \frac{2ty_2^K}{(K-1)!(K-2)!|\Gamma|^K}\times(K-2)!\frac{|\Gamma|^{K-1}}{\sigma_2^{2(K-1)}} \\ & \exp\left\{-\frac{1}{|\Gamma|}(t^2\sigma_2^2+\sigma_1^2)y_2\right\} I_0(2ty_2\rho\sigma_1\sigma_2/|\Gamma|) \end{aligned}$$

$$p_{Y_2T}(y_2,t) = \frac{2ty_2^K}{(K-1)!|\Gamma|\,\sigma_2^{2(K-1)}}\exp\left\{-\frac{1}{|\Gamma|}(t^2\sigma_2^2+\sigma_1^2)y_2\right\} I_0(2ty_2\rho\sigma_1\sigma_2/|\Gamma|)$$

We let $x = 2ty_2\rho\sigma_1\sigma_2/|\Gamma|$ then $dx = 2t\rho\sigma_1\sigma_2 dy_2/|\Gamma|$ and

$$\begin{aligned} p_T(t) &= \frac{1}{(K-1)!}\frac{\lambda}{\rho}\frac{1}{(2t)^K}\frac{1}{\lambda^K}\frac{(1-\rho^2)^K}{\rho^K} \\ & \int_0^{+\infty} x^K\ \exp\left\{-\frac{1}{|\Gamma|}\frac{t^2\sigma_2^2+\sigma_1^2}{2t\rho\sigma_1\sigma_2}x\right\} I_0(x)dx \end{aligned} \tag{8.20}$$

rewritten as

$$p_T(t) = \frac{\lambda}{\rho}\int_0^{+\infty}(\zeta x)^K\, e^{-\xi x} I_0(x)dx \tag{8.21}$$

where

$$\zeta = \frac{1-\rho^2}{2t\rho\lambda}\frac{1}{((K-1)!)^{1/K}}$$

$$\xi = \frac{1+t^2\lambda^2}{2t\rho\lambda}$$

The last integration w.r.t. x has been done numerically. Also we remark that

$$\zeta^K = \exp\left(K\log(\mu) - \sum_{k=1}^{K-1}\log(k)\right)$$

where

$$\mu = \frac{1-\rho^2}{2t\rho\lambda}$$

which is more accurate to use than factorial function. Also it is more accurate to use $\tilde{I}_0(x) = e^{-x}I_0(x)$ which is usually proposed in several numerical toolboxes.

We consider the ratio $\widehat{\Gamma}_{1,1}/|\widehat{\Gamma}_{1,2}|$, expression (**??**).

For the ratio $V = A_1/R$, we compute at first the distribution of $W = R/A_2$ which can be obtained using the expression (8.21) reversing the role of A_1 and A_2. Then using the transformation $V = 1/W$ whose the Jacobian is $1/V^2$ we get

$$p_V(v) \;=\; \frac{\lambda}{\rho}\,v^{-2}\int_0^{+\infty}(\zeta x)^K\,e^{-\xi x}I_0(x)dx \tag{8.22}$$

where

$$\zeta = \frac{v(1-\rho^2)}{2\rho\lambda}\frac{1}{((K-1)!)^{1/K}}$$
$$\xi = \frac{v^2+\lambda^2}{2v\rho\lambda}$$

Phase distribution (not completed)

We consider the sequence of $2M+1$ bivariate random gaussian complex independent variables

$$X_m \overset{\text{i.i.d.}}{\sim} \mathcal{N}_c(0,\Gamma)$$

We let

$$\Gamma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 e^{j\theta} \\ \rho\sigma_1\sigma_2 e^{-j\theta} & \sigma_2^2 \end{bmatrix}$$

where $\sigma_1 \geq 0$, $\sigma_2 \geq 0$, $\rho \geq 0$ and $\theta \in (0, 2\pi)$, and

$$\widehat{\Gamma} = \frac{1}{2M+1}\sum_{m=1}^{2M+1} X_m X_m^H = \begin{bmatrix} A_1 & Re^{j\Phi} \\ Re^{-j\Phi} & A_2 \end{bmatrix}$$

where $A_1 \geq 0$, $A_2 \geq 0$, $R \geq 0$ and $\Phi \in (0, 2\pi)$. We also have $R^2 \leq A_1A_2$ thanks to the positivity of $\widehat{\Gamma}$.

We let $K = 2M+1$. The distribution of (A_1, A_2, R, Φ) writes:

$$\begin{aligned} p_{A_1A_2R\Phi}(a_1,a_2,r,\phi) \;=\; & \frac{r(a_1a_2-r^2)^{K-2}}{\pi(K-1)!(K-2)!|\Gamma|^K} \\ & \exp\left\{-\frac{1}{|\Gamma|}\left(\sigma_2^2a_1+\sigma_1^2a_2-2r\rho\sigma_1\sigma_2\cos(\theta-\phi)\right)\right\} \\ & \mathbb{1}(a_1\geq 0)\times\mathbb{1}(a_2\geq 0)\times\mathbb{1}(\sqrt{a_1a_2}\geq r\geq 0)\times\mathbb{1}(\phi\in(0,2\pi)) \end{aligned} \tag{8.23}$$

We have

$$\int_0^{\sqrt{a_1 a_2}} P(r) e^{-\mu r} dr = \left[\frac{-1}{\mu} e^{-\mu r} \sum_{s=0}^{2K-3} \frac{1}{\mu^s} P^{(s)}(r) \right]_0^{\sqrt{a_1 a_2}}$$

Here $\mu = 2\rho\sigma_1\sigma_2 \cos(\theta - \phi)/|\Gamma|$ and

$$\begin{aligned}
P(r) &= r(a_1a_2 - r^2)^{K-2} = \sum_{q=0}^{K-2} (-1)^q (a_1a_2)^{K-2-q} r^{2q+1} C_{K-2}^q \\
P^{(1)}(r) &= (2q+1) \sum_{q=1}^{K-2} (-1)^q (a_1a_2)^{K-2-q} r^{2q} C_{K-2}^q \\
P^{(2)}(r) &= (2q+1)2q \sum_{q=1}^{K-2} (-1)^q (a_1a_2)^{K-2-q} r^{2q-1} C_{K-2}^q \\
&\vdots \\
P^{(2K-4)}(r) &= -(K-2)(a_1a_2)^{K-3} r^3 \\
P^{(2K-3)}(r) &= (-1)^{K-2}
\end{aligned}$$

$$\int_0^{+\infty} a_1^{m+1/2} e^{-\alpha_1 a_1} da_1$$

and

$$\int_0^{+\infty} a_1^{m} e^{-\alpha_1 a_1} da_1$$

We use

$$\int_0^{+\infty} x^{t-1} e^{-\mu x} dx = \Gamma(t)\frac{1}{\mu^t}$$

...

(uncompleted ...)

6. MSC distribution

For the MSC defined in (7.2) an estimator is:

$$\widehat{\mathrm{MSC}}_k = \frac{|\widehat{\Gamma}_{k,1,2}|^2}{\widehat{\Gamma}_{k,1,1}\widehat{\Gamma}_{k,2,2}} \tag{8.24}$$

Its statistics have been given in [2]. The density probability writes

$$\begin{aligned} p(c, \mathrm{MSC}) &= (N_d - 1) \times \left[\frac{(1-c)(1-\mathrm{MSC})}{(1-c \times \mathrm{MSC})^2}\right]^{N_d} \times \frac{1-c \times \mathrm{MSC}}{(1-c)^2} \\ & {}_2F_1(1-N_d, 1-N_d; 1; c \times \mathrm{MSC}) \end{aligned} \tag{8.25}$$

ant the cumulative function writes:

$$\begin{aligned} F(c, \mathrm{MSC}) &= \mathbb{P}\left(\widehat{\mathrm{MSC}}_k \leq c; \mathrm{MSC}\right) \\ &= c \times \left(\frac{1-\mathrm{MSC}}{1-c \times \mathrm{MSC}}\right)^{N_d} \times \\ & \sum_{k=0}^{N_d-2} \left(\frac{1-c}{1-c \times \mathrm{MSC}}\right)^k {}_2F_1(-k, 1-N_d; 1; c \times \mathrm{MSC}) \end{aligned} \tag{8.26}$$

where ${}_2F_1$ is the hypergeometric function.

The integer N_d is either the size of the window for smoothed periodogram or the number of blocks for the averaged peridograms (Welch method). We will call it the degree of smoothing, in short d.o.s.

Fortunately the expression (8.26) is easy to compute using that

$${}_2F_1(-k, 1-N_d; 1; c \times \mathrm{MSC}) = \sum_{m=0}^{k} T_m$$

where $T_0 = 1$ and the recursion:

$$T_m = T_{m-1} \frac{(m-1-k)(m-N_d) \times \text{MSC}}{m^2} c \tag{8.27}$$

To show the accuracy of this expression we did a simple simulation. We generate a sequence of the bivariate process

$$x_n = G w_n$$

where w_n is a bivariate process with 0 mean and covariance I_2. Therefore x_n is a bivariate process with 0 mean and covariance $\Gamma = GG^H$. Therefore the MSC of the 2 components is

$$\frac{|\Gamma_{1,2}|^2}{\Gamma_{1,1}\Gamma_{2,2}}$$

Cumulative function has been performed from Monte-Carlo simulation and compared to the expression (8.26). The results are reported figure 8.1. The theoretical values are in very good agreement with those obtained from Monte-Carlo simulation.

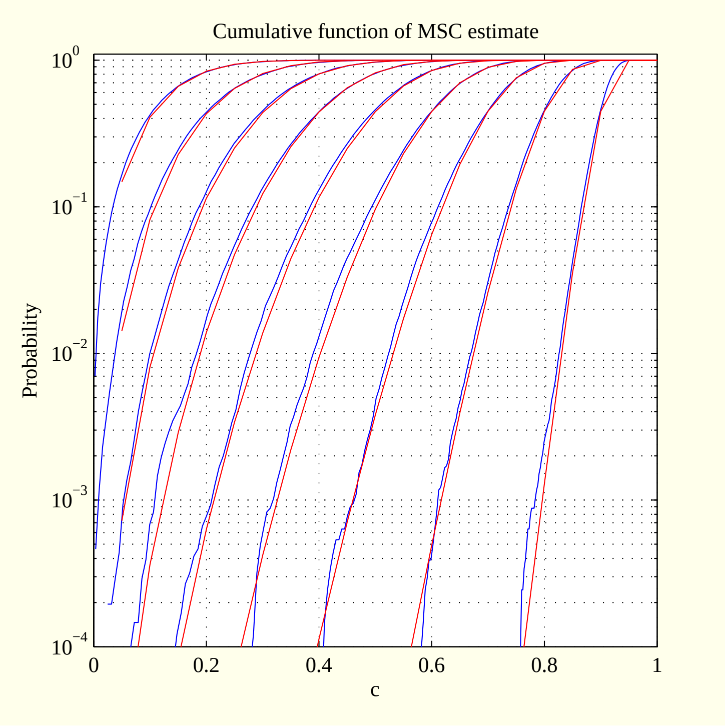

Figure 8.1: *Cumulative function of the MSC estimate given by expression* (8.24)*. The true values of the MSC are from* 0.1 *to* 0.9 *by step* 0.1 *from the left to the right. In blue, the values obtained by simulation on* 40960 *samples. In red the theoretical expression* (8.26).

7. MSC detection

The analytical expressions of the probability density functions (8.25) and the cumulative function (8.26) of the MSC estimator. Therefore we have a way to perform numerically the inverse of the cumulative function and then determine a detection threshold. Indeed, a simple dichotomy can be applied because of the monotony of the cumulative function. Then we can derive a confidence interval and attest on the MSC value.

We consider the statistical model

$$\{p(\mu, \mathrm{MSC}_0), \quad \mathrm{MSC}_0 \in (0,1)\}$$

where p is the probability density function of $\widehat{\mathrm{MSC}}$ for a given value of the d.o.s. N_d. Its expression is given in equation (8.25). We want to test the hypothesis $H_0 = \{\mathrm{MSC}_0 > C\}$ against the counter hypothesis $H_1 = \{\mathrm{MSC}_0 \leq C\}$ with $C \in (0,1)$. The GLRT writes

$$\Lambda(\mu) = \frac{\max_{\mathrm{MSC}_0 \in (0,1)} p(\mu, \mathrm{MSC}_0)}{\max_{\mathrm{MSC}_0 \in H_0} p(\mu, \mathrm{MSC}_0)}$$

The property that $\Lambda(\mu)$ is a monotonic increasing function of c is not proven but we admit that $\widehat{\mathrm{MSC}}$ is a good function of test. In this case we can fix the threshold η such that

$$\mathbb{P}\left(\widehat{\mathrm{MSC}} \leq \eta; \mathrm{MSC}_0\right) = 0.95$$

Figure 8.2 we have reported the η values at 95% for $\mathrm{MSC}_0 \in (0,1)$ for $N_d = 20, 30, 40, 50$. It is worth noticing that the curves is largely over the first bissectrix even for large N_d. For example for MSC $= 0.8$ the threshold is given on the zoom of the figure for different values of N_d.

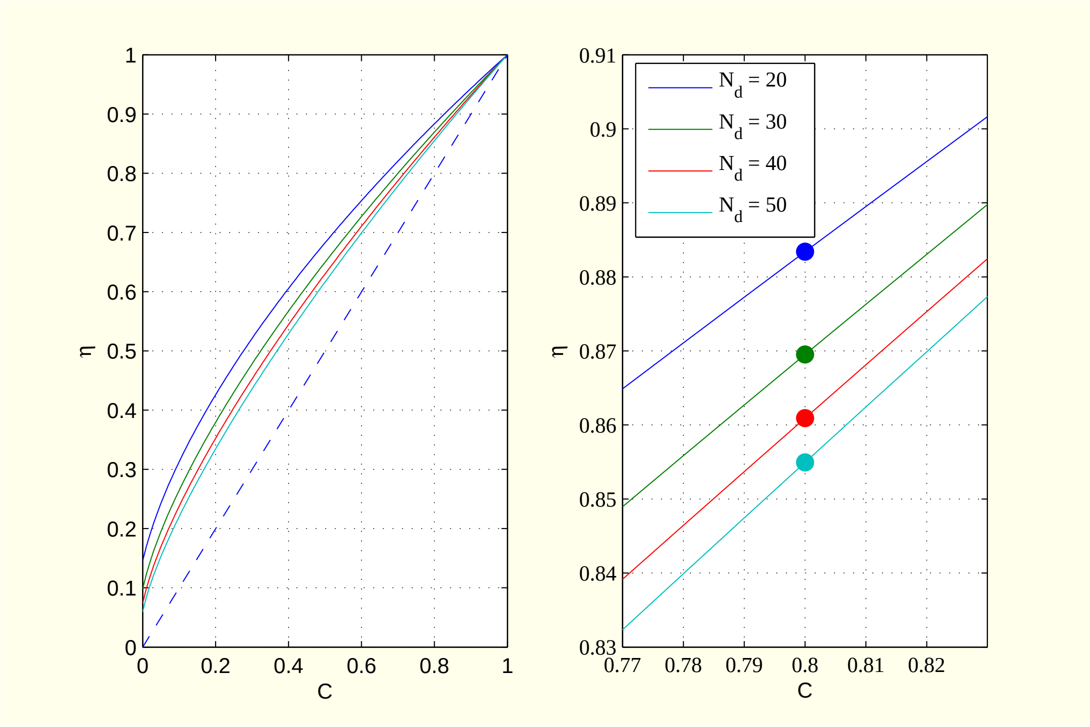

Figure 8.2: *Threshold at* 95% *and* $N_d = 20, 30, 40, 50$ *as a function of the value on test.*

A simulation has been done with a white SOI leading to a level of MSC of 0.7. We have reported figure 8.3 the numerical estimates of the MSC and the threshold at 95%. We verified that only 5% are over the threshold. That means that, if we observe a value of MSC greater than around 0.8, the probability that the MSC is under 0.7 is less than 5%. That is a very conservative attitude.

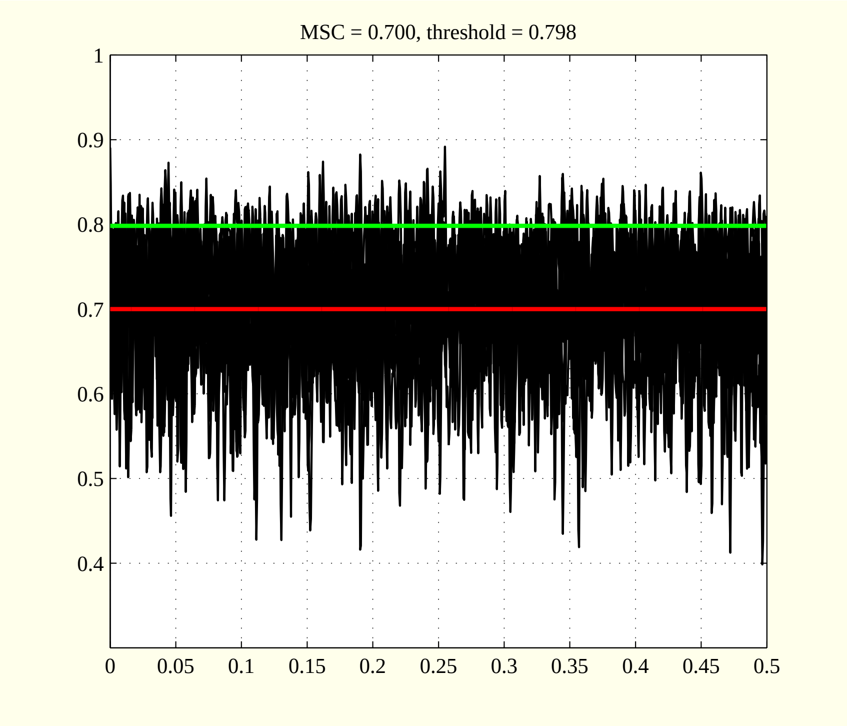

Figure 8.3: *Monte-Carlo simulations on 1000 runs. In red the true value of the MSC. In green, the threshold derived from the analytical expression in such a way that the probability to be over this value is less than* 5%*. In black the Monte-Carlo estimates. It has been verified that* 5% *are over the threshold.*

Chapter 9

LOC

Let us consider the equation:

$$Y_t = g_t \star X_t$$

where g_t is the M-ary impulse response of the single input multiple output (SIMO) linear filter and X_t is a scalar WSS process, with zero mean and spectral matrix (of size 1) $\gamma_x(f)$. It is easy to show that the spectral matrix of Y_t writes

$$\Sigma_y(f) = \gamma_x(f) G(f) G^H(f)$$

where $G(f)$ is the discrete time Fourier transform of g_t. We notice that $\Sigma_y(f)$ is a matrix of rank 1. A particular case is a pure delay propagation filter as for example in the planar propagation where the vector entry of $G(f)$ writes:

$$G_m(f) = e^{-2j\pi f r_m^T \theta}$$

where r_m is the 3D location of the sensor m and θ the wave number of the planar wave. It follows that the coherence matrix of the M-ary observation has the following entry expression

$$\begin{aligned} C_{y,m\ell}(f) &= e^{-2j\pi f (r_m - r_\ell)^T \theta} \qquad (9.1) \\ &= \int_{\mathbb{R}^3} e^{-2j\pi f (r_m - r_\ell)^T u} \delta_\theta(u) du \end{aligned}$$

where du is the infinitesimal Lebesgue measure in $\mathbb{R}^3$ and δ_θ the Dirac's distribution located in θ. The expression (9.1) can be seen as an expectation of $e^{-2j\pi f (r_m - r_\ell)^T \theta}$ assuming that θ is deterministic.

Coherence model

In a pioneer work Mack and Flinn [4] propose to model the loss of coherence by considering that the azimuth, the elevation and velocity are uncertain. More specifically, authors in [6] assume that the 3D vector $\mu = (a, e, c)$ where a denotes the azimuth, e the elevation and c the sound velocity writes:

$$\mu = \mu_0 + \nu \tag{9.2}$$

where $\mu_0 = (a_0, e_0, c_0)$ is a deterministic value and ν a zero-mean Gaussian random vector of dimension 3, whose covariance is denoted Σ_μ. We know that μ is related to the 3D wavenumber vector θ by the one-to-one mapping

$$f : \mu = \begin{cases} a = \arg(\theta_2 + j\theta_1) & a \in (0, 2\pi) \\ e = \arg\sin(c\theta_3) & e \in (-\pi/2, \pi/2) \\ c = (\theta_1^2 + \theta_2^2 + \theta_3^2)^{1/2} & c \in \mathbb{R}^+ \end{cases} \longleftrightarrow$$

$$\theta = \begin{cases} \theta_1 = -c^{-1}\sin(a)\cos(e) \\ \theta_2 = c^{-1}\cos(a)\cos(e) \\ \theta_3 = c^{-1}\sin(e) \end{cases} \in \mathbb{R}^3$$

Using a first order Taylor's expansion, it follows that $\theta \approx f(\mu_0) + J(\mu_0)(\mu - \mu_0)$ where the Jacobian writes

$$J(\mu_0) =$$
$$\begin{bmatrix} -c_0^{-1}\cos(a_0)\cos(e_0) & c_0^{-1}\sin(a_0)\sin(e_0) & c_0^{-2}\sin(a_0)\cos(e_0) \\ -c_0^{-1}\sin(a_0)\cos(e_0) & -c_0^{-1}\cos(a_0)\sin(e_0) & -c_0^{-2}\cos(a_0)\cos(e_0) \\ 0 & c_0^{-1}\cos(e_0) & -c_0^{-2}\sin(e_0) \end{bmatrix}$$

Therefore from (9.2) we can write $\theta \approx \theta_0 + \varepsilon$ where $\varepsilon = J(\mu_0)\nu$ appears as a Gaussian, zero-mean random vector of dimension 3 whose the covariance is

$$\Sigma_\theta \approx J(\mu_0)\Sigma_\mu J'(\mu_0) \tag{9.3}$$

Based on the expression (9.1), the coherence matrix entry of the model with random wavefront writes:

$$C_{m,\ell}(f) \quad = \quad e^{-2j\pi f(r_m - r_\ell)^T\theta_0}\Phi_\epsilon(2\pi f(r_m - r_\ell))$$

where $\Phi_\epsilon : u \in \mathbb{R}^3 \mapsto \mathbb{E}\left[e^{ju^H\varepsilon}\right] \in \mathbb{C}$ is the characteristic function of ε. Since ε is Gaussian distributed with zero-mean and covariance matrix Σ_θ, hence we have

$$C_{m,\ell}(f) = \underbrace{e^{-2j\pi f(r_m - r_\ell)^T\theta_0}}_{\text{pure delay}}\underbrace{e^{-2\pi^2 f^2(r_m - r_\ell)^T\Sigma_\theta(r_m - r_\ell)}}_{\text{LOC}} \tag{9.4}$$

The first term corresponds to the pure delay which is associated to the deterministic part and whose the modulus is 1. The second term corresponds to the LOC associated to the stochastic part. Finally the log-MSC writes:

$$\log \rho_{m,\ell}(f) = -2\pi^2 f^2 (r_m - r_\ell)^T \Sigma_\theta (r_m - r_\ell) \tag{9.5}$$

Let us notice that the log-MSC depends on the relative direction of the vector $(r_m - r_\ell)$ w.r.t. the eigenvectors of Σ_θ.

We know that μ is related to the slowness vector k by the one-to-one mapping

$$f : \mu = \begin{cases} a = \arg(k_2 + jk_1) & a \in (0, 2\pi) \\ e = \arg\sin(ck_3) & e \in (-\pi/2, \pi/2) \\ v = (k_1^2 + k_2^2 + k_3^2)^{1/2} & v \in \mathbb{R}^+ \end{cases} \iff$$

$$k = \begin{cases} k_1 = -v^{-1} \sin(a) \cos(e) \\ k_2 = v^{-1} \cos(a) \cos(e) \\ k_3 = v^{-1} \sin(e) \end{cases} \in \mathbb{R}^3$$

Using a first order Taylor's expansion, it follows that $k \approx f(\mu_0) + J(\mu_0)(\mu - \mu_0)$ where the Jacobian writes

$$\begin{aligned} J(\mu_0) &= v_0^{-1} \begin{bmatrix} -\cos(a_0)\cos(e_0) & \sin(a_0)\sin(e_0) & v_0^{-1}\sin(a_0)\cos(e_0) \\ -\sin(a_0)\cos(e_0) & -\cos(a_0)\sin(e_0) & -v_0^{-1}\cos(a_0)\cos(e_0) \\ 0 & \cos(e_0) & -v_0^{-1}\sin(e_0) \end{bmatrix} \\ &= v_0^{-1} \times \tilde{J}(\mu_0) \end{aligned}$$

Therefore from (9.2) we derive $k \approx k_0 + \varepsilon$ where $\varepsilon = J(\mu_0)\,\nu$ appears as a zero-mean Gaussian random vector whose the covariance is:

$$\Sigma_k \approx v_0^{-2} \underbrace{\tilde{J}(\mu_0)\Sigma_\mu \tilde{J}^T(\mu_0)}_{\tilde{\Sigma}_k} \tag{9.6}$$

Hence the coherence matrix entry of the model with random wavefront writes:

$$C_{m,\ell}(f) \;=\; e^{-2j\pi f (r_m - r_\ell)^T k_0} \Phi_\epsilon(2\pi f (r_m - r_\ell))$$

where $\Phi_\epsilon : u \in \mathbb{R}^3 \mapsto \mathbb{E}\left[e^{ju^T\varepsilon}\right] \in \mathbb{C}$ is the characteristic function of ε. Since ε is Gaussian distributed with zero-mean and covariance matrix Σ_θ, hence we have

$$C_{m,\ell}(f) = \underbrace{e^{-2j\pi f (r_m - r_\ell)^T k_0}}_{\text{pure delay}} \underbrace{e^{-2\pi^2 (f/v_0)^2 (r_m - r_\ell)^T \tilde{\Sigma}_k (r_m - r_\ell)}}_{\text{LOC}} \tag{9.7}$$

Chapter 10

Remark on deterministic representation

Remark on a deterministic approach

We have seen that, in the absence of noises, we can use the DFT of each trajectory and perform the ratio without any averaging. But we have also to express a deterministic property which can be the equivalent of the MSC condtion. For that we can work frequency by frequency and therefore omit the dependence in frequency. We denote $X_{u,1}, \dots X_{u,M}$ the DFT of the signal of the SUT and $X_{r,1}, \dots X_{r,M}$ the DFT of the signal of the SREF.

Now to define in the determistic case a notion equivalent to the MSC, we can say that the values of the ratio must be almost constant along the M windows. That writes: it exists 2 coefficients not depending on m, verifying $\alpha_u^2 + \alpha_r^2 = 1$ and s.t.

$$\alpha_u X_{u,m} + \alpha_r X_{r,m} \approx 0 \tag{10.1}$$

With obvious matricial notations, we have:

$$X\alpha = \epsilon$$

$$X = \begin{bmatrix} X_{u,1} & X_{r,1} \\ \vdots & \\ X_{u,M} & X_{r,M} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \alpha_u \\ \vdots \\ \alpha_r \end{bmatrix}$$

The minimization of the norm of ϵ under the constraint $\|\alpha\|^2 = 1$ leads to take for α the eigenvector associated to the lowest eigenvalue of denoted $\lambda_{\min}$. The matrix $M^{-1}X^T X$ writes:

$$\Gamma = \frac{1}{M} \begin{bmatrix} \sum_{m=1}^M |X_{u,m}|^2 & \sum_{m=1}^M X_{u,m} X_{r,m}^* \\ \sum_{m=1}^M X_{u,m}^* X_{r,m} & \sum_{m=1}^M |X_{r,m}|^2 \end{bmatrix}$$

The minimum value is then given by:

$$\epsilon_{\text{min}} = \lambda_{\text{min}}$$

Lower λ_{min} better the approximation (10.1). On the other hand, lower λ_{min} lower the determinant of Γ which writes:

$$\begin{aligned} \Delta &= M^{-2} \sum_{m=1}^{M} |X_{u,m}|^2 \sum_{m=1}^{M} |X_{r,m}|^2 - M^{-2} \left| \sum_{m=1}^{M} X_{u,m} X_{r,m}^* \right|^2 \\ &= M^{-2} \sum_{m=1}^{M} |X_{u,m}|^2 \sum_{m=1}^{M} |X_{r,m}|^2 \, (1 - \text{MSC}) \end{aligned}$$

In conclusion nearer the MSC to 1, better the approximation (10.1).

Part III

Library

Chapter 11

Programs

1. RMSE as function of coherence

```
%===============================================================
% estimHanalysis.m
%===============================================================
% Determine the level of MSC useful to reach a given level of
% accuracy for the ratio HUT/HREF.
% Simulation is conducted on synthetical or semi-synthetical
% signals.
%===============================================================
%
% used function: fbankanalysis.m
% with no filterbank
%
% We draw randomly and use the H estimate
% Models of signals
% xREF = BGfield + sqrt(sigma2REF)*wREF;
% xUT  = BGfield + sqrt(sigma2UT)*wUT;
% yREF = filter(bREF,aREF,xREF);
% yUT  = filter(bUT,aUT,xUT);
%
% Results have to be compared to these
% of program STATSONH11H12.m, plotted with allprintstatsonHest.m
%===============================================================
% N is divided in segments of length equal to Tfft_sec*Fs_Hz
% with Tfft_sec=300.
% To form the data block on which the spectrum is performed,
% we use M consecutive segments.
% The simulation is iterated in such a way the duration of the
% total observation is 5 hours. This value is obvioulsy not crucial.
% It is just to perform the RMSE on enough simulations.
% The RMSE is computed by averaging on the full frequency band.
%
%===============================================================
clear
addpath  ../../../ZZtoolbox/
synth            = 1;
Fs_Hz   = 20;
FFT_sec = 300;
Lfft    = fix(FFT_sec*Fs_Hz);
duration_hour    = 5;
N           = duration_hour*3600*Fs_Hz;
frqs           = (0:Lfft-1)*Fs_Hz/Lfft;
selectbandave_Hz = 10;
idbband          = find(frqs<selectbandave_Hz,1,'last');
bandave          = 1:idbband;


%======== filter REF ==========
FREF_Hz      = 0;
rhoREF       = 0.7;
costhetaREF  = (1+rhoREF*rhoREF)*cos(2*pi*FREF_Hz/Fs_Hz)/2/rhoREF;
bREF         = 0.5*[1+rhoREF*rhoREF;-4*rhoREF*costhetaREF;1+rhoREF*rhoREF];
aREF         = [1 -2*rhoREF*costhetaREF rhoREF*rhoREF]';
HREF         = fft(bREF,Lfft) ./ fft(aREF,Lfft);
HREF         = HREF(:);
%======== filter UT ==========
FUT_Hz       = 0;
rhoUT        = 0.7;
costhetaUT   = (1+rhoUT*rhoUT)*cos(2*pi*FUT_Hz/Fs_Hz)/2/rhoUT;
bUT          = 0.5*[1+rhoUT*rhoUT; -4*rhoUT*costhetaUT; 1+rhoUT*rhoUT];
aUT          = [1 -2*rhoUT*costhetaUT rhoUT*rhoUT]';
HUT          = fft(bUT,Lfft) ./ fft(aUT,Lfft);
HUT          = HUT(:);

% randn('seed',0)
if synth
    % for synthetical signals
    % the BGfield is white
    randnN = randn(N,1);
    randnN = randnN/std(randnN);
else
    % the BGfield is drawn
    % from the database
    directorydata = '../../../../DATA_IS/I26/';
    % directorydata = './';
    filesmat = dir([directorydata '*.mat']);
    nbmats   = length(filesmat);
    ifile    = 2;
    commandload = sprintf('load %s%s',directorydata,filesmat(ifile).name);
    eval(commandload)
    id1=fix(rand*1e6);
    data_pa = records{2}.data(id1+(1:N));
    data_pa = data_pa-mean(data_pa);
    randnN = data_pa/std(data_pa);
end
% theoretically gamma has no effect
gamma          = 5;
BGfield        = sqrt(gamma)*randnN;

overlapFFT     = 0.5;
overlapAVE     = 0;

sqrtLfft       = sqrt(Lfft);
frqs           = (0:Lfft-1)*Fs_Hz/Lfft;

listM          = [5 10];
LM             = length(listM);
listMSC        = 0.8:0.02:0.99;
LMSC           = length(listMSC);
% root mean square error on the ratio SUU/abs(SUR)
RMSEUUUR       = zeros(LMSC,LM);
RMSEAB2        = zeros(LMSC,LM);

%=============================
% Because airinlet is given the MSC implies
% the level of white noise (not true for non white)
%=============================
for iMSC=1:LMSC
    MSCtrue   = listMSC(iMSC);
    %=============================
    % airinlet induces lower noise on UT than on REF
    % airinlet value is of order of magnitude of the
    % number of airinlets, typically a few tens
    % but has little consequence because we fix the MSC
    % by taking into  account this number.
    %=============================
    airinlet    = 48;
    sigma2UT    = max(roots([airinlet ...
        (airinlet+1)*gamma ...
        gamma*gamma*(1-1  ./ MSCtrue)]));
    sigma2REF   = airinlet*sigma2UT;
    % gamma*gamma/(gamma+sigma2UT)/(gamma+sigma2REF)

    wREF  = randn(N,1);
    wUT   = randn(N,1);

    xREF  = BGfield  + sqrt(sigma2REF)*wREF;
    xUT   = BGfield  + sqrt(sigma2UT)*wUT;

    yREF    = filter(bREF,aREF,xREF);
    yUT     = filter(bUT,aUT,xUT);
    % U = 1, REF =2
    signals = [yUT, yREF];

    for im=1:LM
        M  = listM(im);
        P=1;
        filtercharacteristics(P).designname         = ’’;
        filtercharacteristics(P).Norder             = 0;
        filtercharacteristics(P).Wlow_Hz            = 0.0001;
        filtercharacteristics(P).Whigh_Hz           = Fs_Hz;
        filtercharacteristics(P).SCPperiod_sec      = FFT_sec*M;
        filtercharacteristics(P).windowshape        = ’hann’;
        filtercharacteristics(P).overlapDFT         = 0.5;
        filtercharacteristics(P).overlapSCP         = 0;
        filtercharacteristics(P).ratioDFT2SCP       = M;

        [SUTs, filteredsignals, allfrqsFFT_Hz, alltimes_sec, filterbank] = ...
            fbankanalysis(signals, ...
            filtercharacteristics, ...
            Fs_Hz,...
            0.5);

        nbruns = size(SUTs.estimRsup.tabmod,2);
        rmse_integrated = zeros(nbruns,1);
        for irun=1:nbruns
            HestUUUR_irun = SUTs.estimRsup.tabmod(:,irun) .* abs(HREF);
            mseHestUUUR_irun = (HestUUUR_irun(bandave)-abs(HUT(bandave))) .^2;
            mseHestUUUR_irun_rel = mseHestUUUR_irun ./ abs(HUT(bandave));
            rmse_integrated(irun)=sqrt(nanmean(mseHestUUUR_irun));
        end
        RMSEUUUR(iMSC,im) = nanmean(rmse_integrated);
    end
end
%%
figure(4)
leg = cell(LM,1);
for it=1:LM
    leg{it}=sprintf(’M = %3i’,listM(it));
end
plot(listMSC,RMSEUUUR,’x-’)
% hold on
% plot(listMSC,RMSEAB2,’o-’)
set(gca,’fontsize’,12,’fontname’,’times’)
grid on
hl=legend(leg);
set(hl,’fonts’,12,’fontn’,’times’)
xlabel(’coherence’,’fontsize’,12,’fontname’,’times’)
ylabel(’RMSE on SUT’,’fontsize’,12,’fontname’,’times’)
hold on
plot([0.8 1],0.05*ones(2,1),’--’)
hold off
set(gca,’ylim’,[0 0.2])
%===========================================
%%
HorizontalSize = 12;
VerticalSize   = 8;
set(gcf,’units’,’centimeters’);
set(gcf,’paperunits’,’centimeters’);
set(gcf,’PaperType’,’a4’);
set(gcf,’position’,[0 5 HorizontalSize VerticalSize]);
set(gcf,’paperposition’,[0 0 HorizontalSize VerticalSize]);

set(gcf,’color’, [1,1,0.92]);%0.7*ones(3,1))
set(gcf, ’InvertHardCopy’, ’off’);

% figure(4); print -depsc -loose ../../textes/6distConjointHMSC/figures/allHest
% print -depsc -loose ../../textes/6distConjointHMSC/figures/allHest
% !epstopdf ../../textes/6distConjointHMSC/figures/allHest.eps
% !rm ../../textes/6distConjointHMSC/figures/allHest.eps

return
%%
% save signals signals Fs_Hz
% return
figure(4)
subplot(411)
semilogx(frqs,10*log10(sRRf))
set(gca,’xlim’,[0 6])
set(gca,’ylim’,[-20 20])
ylabel(’dB’)
grid on
%==
subplot(412)
semilogx(frqs,10*log10(sUUf))
set(gca,’xlim’,[0 6])
set(gca,’ylim’,[-20 20])
ylabel(’dB’)
grid on
%==
subplot(413)
semilogx(frqs,20*log10(mean(HestUUUR,2)))
hold on
semilogx(frqs,20*log10(abs(HUT)),'r')
hold off
set(gca,'ylim',20*log10(abs(HUT(1)))+[-15,5])
% set(gca,'ylim',[-20 40])
set(gca,'xlim',[0 6])
ylabel('dB')
grid on
%==
subplot(414)
semilogx(frqs,MSC)
set(gca,'xlim',[0 6])
grid on


%
HorizontalSize = 25;
VerticalSize   = 18;
set(gcf,'units','centimeters');
set(gcf,'paperunits','centimeters');
set(gcf,'PaperType','a4');
% set(gcf,'position',[0 5 HorizontalSize VerticalSize]);
set(gcf,'paperposition',[0 0 HorizontalSize VerticalSize]);
set(gca,'fontn','times','fonts',10)

set(gcf,'color', [1,1,0.92]);
set(gcf, 'InvertHardCopy', 'off');
```

2. Test of coherence

```
%================================================
% SigniLevelTestCoherence.m
%================================================
% Performances of test on the hypothesis
% H0 = { MCS > cintest} with M d.o.f
% The significance level is denoted signifiancelevel
% Used function: cumulFunctionMSC.m
%================================================
clear
addpath  ../../../ZZtoolbox/
signifiancelevel = 0.9;
cintest = 0.97; NE=100;
E = linspace((0.9),(0.999),NE)';
allM = 5:2:10;
LM = length(allM);
P = zeros(NE,LM);
thres = zeros(LM,1);
leg=[];
for id=1:LM
    N = allM(id);
    P(:,id) = cumulFunctionMSC(E,cintest,N);
    thres(id) = invcumulFunctionMSC(signifiancelevel,cintest,N);
    leg = [leg;{sprintf('M = %2i, th = %4.3f',N,thres(id))}];
end
plot(E,P,'.-')
hold on
plot([0, 1],[signifiancelevel * [1 1]])
ht = plot(thres,signifiancelevel*ones(LM,1),'o');
set(ht,'markerfacec','b')
hold off
grid
set(gca,'ylim',[0.8 1])
set(gca,'xlim',[0.97 1])

ht=legend(leg);
set(ht,'fontsize',12)
ht=title(sprintf('H_0 = %s MSC>%4.2f %s at level %4.2f','\{',cintest,'\}',signifiancelevel));
set(ht,'fontname','times','fontsize',12)

HorizontalSize = 12;
VerticalSize   = 8;
set(gcf,'units','centimeters');
set(gcf,'paperunits','centimeters');
set(gcf,'PaperType','a4');
set(gcf,'position',[0 5 HorizontalSize VerticalSize]);
set(gcf,'paperposition',[0 0 HorizontalSize VerticalSize]);
set(gca,'fontname','times','fontsize',12)

set(gcf,'color', [1,1,0.92]);
set(gcf, 'InvertHardCopy', 'off');
%

% figure(4); print -depsc -loose  ../../textes/6distConjointHMSC/figures/MSCtestthreshold

% figure(4)
% print -dpdf -loose ../../textes/6distConjointHMSC/figures/MSCtestthreshold
% print -depsc -loose ../../textes/6distConjointHMSC/figures/allHest
% !epstopdf ../../textes/6distConjointHMSC/figures/allHest.eps
% !rm ../../textes/6distConjointHMSC/figures/allHest.eps
```

3. Statistical distribution of the ratios

```
%=======================================================================
%   CIHestimate.m
%=======================================================================
% We compar theoretical and simulated distributions
% the thoeretical values are performed in the function
%                 statsRatiosHbis.m (see toolbox)
%=======================================================================
% statistics by drawing vectors under a given spectral matrix GAMMA
% WARNING: that means that the frequency value is fixed and hence can be
% omitted.
%===
% Comparison of Monte-Carlo simulation results and
% theoretical values for the two ratios used to estimate
% HU from HR. The results do depend mainly on the
% true ratio denoted
%
%       HUonHR_k = HU_k / HR_k
%
% which is a complex number.
% To obtain consistent estimates we mix 2M+1 values, either
% by smoothing the periodogram or by averaging several periodograms
% (Welch's approach).
% The statitstics of the modulus of HUonHR_k are given in the report.
% For the phase we used an approximative formula based on delta-method
% and gaussian shape.
%
% Reference =1, Under test=2
%       REF UxR
%       UxR UT
%=======================================================================
%
clear
addpath  ../../../ZZtoolbox/

%====================================================
% R=1, U=2
% we loock for statistics of ratioUUonUR, ratioURonRR,
% then HU = HR x ratioUUonUR OR HU = HR x ratioURonRR
%====================================================

% global factor for the spectral matrices(no effect)
gammafactor            = 5;
% true HUonHR
absHUonHR              = 1;
phase_HUminusHR_degree = 20;
argHUonHR_rad          = phase_HUminusHR_degree*pi/180;
HRonHU                 = absHUonHR*exp(1j*argHUonHR_rad);
twoMminus1              = 9;
% noise on the sensor under test is currently lower
% than noise on the reference sensor, ie
%      sigmaR = g x sigmaU, with g about 10
% C =1/(1+sigmaU2)(1+sigmaR2)
%
% (1+sigmaU2)(1+g^2 sigmaU2)=1/C
% [1-1/C 1+g^2 g^2]
%%
MSC_theo               = 0.9;
noiseratioEff          = sqrt(1);
sigmaU2                = roots([noiseratioEff^2  1+noiseratioEff^2 1-1/MSC_theo]);
sigmaU2                = max(sigmaU2);
sigmaU                 = sqrt(sigmaU2);
sigmaR                 = sigmaU*noiseratioEff;
sigmaR2                = sigmaR ^2;
%%
%=========================
%====== spectralmatrix
spectralmatrix         = gammafactor * ...
    [(1+sigmaR2)/absHUonHR  exp(-1j*argHUonHR_rad); ...
    exp(1j*argHUonHR_rad) (1+sigmaU2)*absHUonHR];
%=========================
% for histograms
nbbins = 500;

absHR2 = gammafactor/absHUonHR;
HR     = sqrt(absHR2);
% Monte-Carlo simulations
Lruns  = 1000000;
sqrtG  = sqrtm(spectralmatrix);
SUU_MC = zeros(Lruns,1);
SRR_MC = zeros(Lruns,1);
SUR_MC = zeros(Lruns,1);
```

```
for in = 1:twoMminus1
    W  = (randn(Lruns,2)+1j*randn(Lruns,2))/sqrt(2);
    X           = W * sqrtG;
    % REF is on RR=1, and UT is on UU=2
    SRR_MC      = SRR_MC + (X(:,1) .* conj(X(:,1)));
    SUU_MC      = SUU_MC + (X(:,2) .* conj(X(:,2)));
    SUR_MC      = SUR_MC + (X(:,1) .* conj(X(:,2)));
end
SUU_MC           = SUU_MC/twoMminus1;
SRR_MC           = SRR_MC/twoMminus1;
SUR_MC           = SUR_MC/twoMminus1;
Rsup_MC          = SUU_MC ./ SUR_MC;
absSUR_MC        = abs(SUR_MC);
realRsup_MC      = real(Rsup_MC);
imagRsup_MC      = imag(Rsup_MC);
phaseRsup        = -atan2(imagRsup_MC,realRsup_MC);
phaseRsup_degree = phaseRsup*180/pi;
%============= compare mean/median ================
medianrealRsup         = nanmedian(realRsup_MC);
medianimagRsup         = nanmedian(imagRsup_MC);
absmedianRsup          = sqrt(medianrealRsup .^2 + ...
    medianimagRsup .^2);
phaseRmediansup        = atan2(medianimagRsup,medianrealRsup);
phaseRmediansup_degree = phaseRmediansup*180/pi;

meanrealRsup         = nanmean(realRsup_MC);
meanimagRsup         = nanmean(imagRsup_MC);
absmeanRsup          = sqrt(meanrealRsup .^2 + ...
    meanimagRsup .^2);
phaseRmeansup        = atan2(meanimagRsup,meanrealRsup);
phaseRmeansup_degree = phaseRmeansup*180/pi;
%====================================================

hatsigmau   = sigmaU*(1+randn(Lruns,1)/sqrt(twoMminus1));
hatsigmar   = sigmaR*(1+randn(Lruns,1)/sqrt(twoMminus1));

hatsigmau2  = hatsigmau .^2;
hatsigmar2  = hatsigmar .^2;

AK               = (SUU_MC .* absHR2) ./ (absSUR_MC .^2);
hatgammaSOI      = 0.5*(1+sqrt(1+4*hatsigmau2 .* AK)) ./ AK;
hatabslambda_MC  = absSUR_MC ./ (hatgammaSOI .* absHR2);

%==================================================
absRinf_MC = absSUR_MC ./ SRR_MC;
absRsup_MC = abs(Rsup_MC);
absRmid_MC = sqrt(SUU_MC ./ SRR_MC);
%==================================================
noiseratioEff2=noiseratioEff^2;
if noiseratioEff>1
    Hgknown_MC = (((noiseratioEff2-1)/(2*noiseratioEff2)) ...
        .*(absRsup_MC)) .*...
        (1+sqrt(1+4*noiseratioEff2/((noiseratioEff2-1)^2)*...
        absRinf_MC ./ absRsup_MC));
end

MSC_MC = real((absSUR_MC .^2) ./ (SUU_MC .*SRR_MC));

%========================
% histograms
[hRinf,binRinf] = hist(absRinf_MC,nbbins);
pdfRinf_MC      = hRinf /Lruns/(binRinf(2)-binRinf(1));

[hRsup,binRsup] = hist(absRsup_MC,nbbins);
pdfRsup_MC      = hRsup /Lruns/(binRsup(2)-binRsup(1));

[hRmid,binRmid] = hist(absRmid_MC,nbbins);
pdfRmid_MC      = hRmid /Lruns/(binRmid(2)-binRmid(1));

[harg,binarg]  = hist(phaseRsup_degree,nbbins);
pdfarg_MC      = harg /Lruns/(binarg(2)-binarg(1));

if noiseratioEff>1
    [hgknown,bingknown] = hist(Hgknown_MC,nbbins);
    pdfgknown_MC        = hgknown /Lruns/(bingknown(2)-bingknown(1));
end

mean2expH       = -(absRinf_MC - absRsup_MC);
[hmean,binmean]  = hist(mean2expH,nbbins);
pdfmean_MC       = hmean /Lruns/(binmean(2)-binmean(1));

[hmym,binmym]    = hist(hatabslambda_MC,100);
```

```
pdfmym_MC        = hmym /Lruns/(binmym(2)-binmym(1));

%============ theoretical expressions
allT.TUUonUR     = binRsup;
allT.TURonRR     = binRinf;
allT.MSC         = 0;
allT.phase       = binarg;
[hatpdfGsup, hatpdfGinf, hatMSC, hatPhase] = ...
    statsRatiosHbis(allT,spectralmatrix,twoMminus1,0.05);

%%
stdapMLEdeltamethod                 = ...
    sqrt(absHUonHR * absHUonHR * (1-MSC_theo)/sqrt(MSC_theo)/(2*twoMminus1));
hatpdfappMLE                        = ...
    (1/sqrt(2*pi)/stdapMLEdeltamethod)* ...
    exp(-(binmym-1) .^2/(2 * stdapMLEdeltamethod^2));
%=========== conditional expectation
% E(Rsup|C\in(c1,c2))
LLlistind = 200;
listind   = linspace(0.1,1,LLlistind);
Econd     = zeros(LLlistind-1,1);
for ib=2:LLlistind
    condindex   = and(MSC_MC>listind(ib-1), MSC_MC<listind(ib));
    Econd(ib-1) = nanmean(absRsup_MC(condindex));
end
%%
if 1
    figure(1)
    clf
    subplot(2,1,1)
    bar(binRsup,pdfRsup_MC)
    hold on
    plot(binRsup,hatpdfGsup.pdf,'r');%,'linew',2)
    grid on
    hold off

    %== title
    title(sprintf('sensor gain ratio = %5.1f, true MSC = % 4.2f,\nnoise ratio = % 4.2f, M = %i',...
        absHUonHR, MSC_theo, noiseratioEff, (twoMminus1+1)/2),'fontsize',12)

    txtsup = '$R_{\sup}$';
    text('string',txtsup,'interpreter','latex','pos',[1.2 5],'fontsize',16)
    set(gca,'fontname','times','fontsize',12)
    set(gca,'xlim',absHUonHR + [-0.3 0.3])
    set(gca,'ylim',[0 6])
    %===========================================

    subplot(2,1,2)
    bar(binRinf,pdfRinf_MC)
    hold on
    plot(binRinf,hatpdfGinf.pdf,'r','linew',2)
    hold off
    grid on
    txtinf = '$R_{\inf}$';
    text('string',txtinf,'interpreter','latex','pos',[1.2 5],'fontsize',16)
    set(gca,'fontsize',12,'fontname','times')
    set(gca,'xlim',absHUonHR + [-0.3 0.3])
    set(gca,'ylim',[0 6])

    %     %===========================================
    %
    %     subplot(3,1,3)
    %     %     bar(bingknown,pdfgknown_MC)
    %     %     set(gca,'xlim',[0.2 1.8])
    %     bar(binarg,pdfarg_MC)
    %     hold on
    %     plot(binarg,hatPhase,'r','linew',2)
    %     hold off
    %     grid on
    %     % text('string',txt,'interpreter','latex','pos',[0.75,3],'fontsize',16)
    %     set(gca,'fontname','times','fontsize',12)
    %===========================================
    HorizontalSize = 12;
    VerticalSize   = 8;
    set(gcf,'units','centimeters');
    set(gcf,'paperunits','centimeters');
    set(gcf,'PaperType','a4');
    set(gcf,'position',[0 5 HorizontalSize VerticalSize]);
    set(gcf,'paperposition',[0 0 HorizontalSize VerticalSize]);

    set(gcf,'color', [1,1,0.92]);%0.7*ones(3,1))
    set(gcf, 'InvertHardCopy', 'off');
end
```

```
% figure(1); print -depsc -loose ../../textes/6distConjointHMSC/figures/theoreticaldistribratios.eps

%=========================================
%%
if 0
    threshold1            = MSC_theo;
    threshold2            = 0.99;
    figure(2)
    clf
    subplot(121)
    plot(MSC_MC(1:1000:Lruns),(absRsup_MC(1:1000:Lruns)),'.')
    set(gca,'ylim',[0 2.5])
    set(gca,'xlim',[0 1])
    grid on
    hold on
    plot([0,1],absHUonHR*ones(2,1),'k','linew',2)
    hold off
    set(gca,'fontn','times','fonts',18)
    xlabel('MSC','fontsi',18)
    ylabel('Rsup','fontsi',18)

    % subplot(223)
    % plot(listind(2:LLlistind)-0.5* (listind(2)-listind(1)),...
    %     20*log10(Econd))
    % grid on
    % set(gca,'fontn','times','fonts',10)
    % ylabel('Conditional expectation - dB')

    text00 = sprintf('Simulation based on spectral matrix');
    text0 = sprintf('True ratio HUT/HREF = %i,\nTrue MSC = % 4.2f\nNoise power ratio = %i (%4.1f dB)\nSmoothing window number = %i\nLruns = %3.1f mega',  .
        absHUonHR,MSC_theo, noiseratioEff^2, 20*log10(noiseratioEff), twoMminus1, Lruns/1e6);
    text1 = sprintf('E(Rsup|MSC>%4.2f) = %4.2f', threshold1, ...
        nanmean(absRsup_MC(MSC_MC>threshold1)));
    text2 = sprintf('E(Rsup|MSC>%4.2f) = %4.2f (%i)', threshold2, ...
        nanmean(absRsup_MC(MSC_MC>threshold2)), sum(MSC_MC>threshold2));
    text3 = sprintf('E(Rsup) = %4.2f', nanmean(absRsup_MC));
    text4 = sprintf('median(Rsup) = %4.2f', nanmedian(absRsup_MC));
    txts = sprintf('%s\n%s\n%s\n%s\n%s\n%s',text00,text0,text1,text2,text3,text4);

    subplot(122)
    set(gca,'ylim',[0 0.2])
    set(gca,'xtick',[],'ytick',[],'box','off')
    set(gca,'color', [1,1,0.92])
    set(gca,'xcolor', [1,1,0.92])
    set(gca,'ycolor', [1,1,0.92])
    text(0,0.1,txts,'fontn','times','fontsi',18)
    hpos = get(gca,'position');
    set(gca,'position',[0.52 hpos(2:4)])
end
```

4. Example of filter bank description

```
%===========================================================
% filtercharacteristics.m
%===========================================================
%         xx.Norder: order of the filter
%         xx.Wlow_Hz:
%         xx.Whigh_Hz:
%         xx.SCPperiod_sec: duration in second
%                 over which SPC is performed,
%                 expected as stationarity time
%         xx.windowshape: window shape for spectral
%                 analysis (ex. 'hann' or 'hamming', ...)
%         xx.overlapDFT: overlap rate for successive
%                 DFT (typically 0.5)
%         xx.overlapSCP: overlap rate for successive
%                 spectral components (typically 0)
%         xx.ratioDFT2SCP: ratio between period_sec and
%                 DFT duration (typical integer value is 5).
%===========================================================
P=1;
filtercharact(P).designname = 'butter';
filtercharact(P).Norder      = 2;
filtercharact(P).Wlow_Hz       = 0.001;
filtercharact(P).Whigh_Hz       = 0.2;
filtercharact(P).SCPperiod_sec  = 1000;
filtercharact(P).windowshape    = 'hann';
filtercharact(P).overlapDFT     = 0.5;
filtercharact(P).overlapSCP     = 0;
filtercharact(P).ratioDFT2SCP   = 5;
%====================
P=P+1;
filtercharact(P).designname = 'butter';
filtercharact(P).Norder      = 2;
filtercharact(P).Wlow_Hz      = 0.2;
filtercharact(P).Whigh_Hz      = 1;
filtercharact(P).SCPperiod_sec      = 400;
filtercharact(P).windowshape      = 'hann';
filtercharact(P).overlapDFT      = 0.5;
filtercharact(P).overlapSCP      = 0;
filtercharact(P).ratioDFT2SCP      = 5;
%====================
P=P+1;
filtercharact(P).designname = 'butter';
filtercharact(P).Norder      = 2;
filtercharact(P).Wlow_Hz      = 1;
filtercharact(P).Whigh_Hz      = 2;
filtercharact(P).SCPperiod_sec      = 100;
filtercharact(P).windowshape      = 'hann';
filtercharact(P).overlapDFT      = 0.5;
filtercharact(P).overlapSCP      = 0;
filtercharact(P).ratioDFT2SCP      = 5;
%====================
P=P+1;
filtercharact(P).designname = 'butter';
filtercharact(P).Norder      = 2;
filtercharact(P).Wlow_Hz      = 2;
filtercharact(P).Whigh_Hz      = 3;
filtercharact(P).SCPperiod_sec      = 50;
filtercharact(P).windowshape      = 'hann';
filtercharact(P).overlapDFT      = 0.5;
filtercharact(P).overlapSCP      = 0;
filtercharact(P).ratioDFT2SCP      = 5;
%====================
P=P+1;
filtercharact(P).designname = 'butter';
filtercharact(P).Norder      = 2;
filtercharact(P).Wlow_Hz      = 3;
filtercharact(P).Whigh_Hz      = 4;
filtercharact(P).SCPperiod_sec       = 25;
filtercharact(P).windowshape      = 'hann';
filtercharact(P).overlapDFT      = 0.5;
filtercharact(P).overlapSCP      = 0;
filtercharact(P).ratioDFT2SCP      = 5;

%======================= END ===========================
```

5. Full process

```
addpath  ../../../ZZtoolbox/
addpath ../5performCoherenceeffect/
addpath ../00gabrielson/

%========= for extracting data from the database =================

channel     = ’(’’BDF’’,’’BDF’’,’’LWS’’,’’LWD’’,’’LKO’’)’;
stations    = ’(’’I26H6’’,’’I26C6’’)’;
yearstart   =  ’2015’;
monthstart  =  ’06’;
daystart    =  ’06’;
HMSstart    = ’00:00:10’;

yearend     =  ’2015’;
monthend    =  ’06’;
dayend      =  ’16’;
HMSend      = ’23:50:10’;

temporary_gparse = ’gparse_temp.par’;
filewfdisc = ’gparse.wfdisc’;

savedirnamefull =’/dvlscratch/SHI/users/charbit/ProjectIMS2015b/DATA_IS/I26/’;

%========= for analyzing data =================
MSCthreshold = 0.97;
filtercharactfilename = ’filtercharacteristics’;


clear
%============= read settings ======
settings

%============= extract data from the database
h_starttime = sprintf(’%s/%s/%s %s’,yearstart,monthstart,daystart, HMSstart);
h_endtime = sprintf(’%s/%s/%s %s’,yearend,monthend,dayend, HMSend);
[~,starttime] = unix([’h2e ’,h_starttime,’ ofmt="%#"’]);
[~,endtime]   = unix([’h2e ’,h_endtime,’ ofmt="%#"’]);
starttime     = str2double(starttime);
endtime       = str2double(endtime);
wlength       = endtime-starttime;
fid=fopen(temporary_gparse,’w’);
%========= Write the query
fprintf(fid, ’%s\n’, ’open data_source=testbed_archive user=charbit password=sqlmomo’);
fprintf(fid, ’%s\n’,[’query wfdisc select * from sel3.wfdisc where sta in ’, ...
    stations, ’and chan in ’, channel,’ and time between ’,num2str(starttime),...
    ’ and ’,num2str(starttime+wlength),’ order by sta,chan,time’]);
fprintf(fid, ’%s\n’,’read waveforms’);
fprintf(fid, ’%s\n’,’write waveforms’);
fclose(fid);

%====================== Gparse run
unix(’setenv ORACLE_HOME /cots/oracle/oracle-10.2;’);
unix(’setenv D_LIBRARY_PATH $ORACLE_HOME/lib:$ORACLE_HOME/lib32;’);
unix(’/ctbto/ims/sm/local/linux/Geotool++/2.3.10/bin/gparse<gparse_temp.par;’);

%====================== Convert to Matlab
filenamesavemat = convertCSStomatlab(filewfdisc,savedirnamefull);
%======================
commandload = sprintf(’load %s’,filenamesavemat);
eval(commandload)

[SUTs, filteredsignals, allfrqsFFT_Hz, alltimes_sec, filterbank] = ...
    analyzeSUT(records,filtercharactfilename, MSCthreshold);
P = length(filterbank);
%====================== END ============================
% to test display a figure
displayafigure
```

```
function filenamesavemat = convertCSStomatlab(filewfdisc,dirname)
fid   = fopen(filewfdisc,'r');
tline = fgetl(fid);
iline = 0;
while ischar(tline)
    iline           = iline+1;
    station{iline}  = strtrim(tline(1:5));
    chan{iline}     = strtrim(tline(6:13));
    stime(iline)    = str2double(tline(14:34));
    wfid(iline)     = str2double(tline(36:43));
    chanid(iline)   = str2double(tline(44:52));
    jdate(iline)    = str2double(tline(53:61));
    etime(iline)    = str2double(tline(62:79));
    nsamp(iline)    = str2double(tline(80:88));
    samprate(iline) = str2double(tline(89:100));
    calib(iline)    = str2double(tline(101:117));
    calper(iline)   = str2double(tline(118:135));
    instype{iline}  = tline(135:136);
    segtype{iline}  = tline(137:144);
    datatype{iline} = tline(144:146);
    clip{iline}     = tline(147:148);
    dir1{iline}     = strtrim(tline(149:209));
    dfile{iline}    = strtrim(tline(210:247));
    foff(iline)     = str2double(tline(248:257));
    commid(iline)   = str2double(tline(258:266));
    lddate{iline}   = tline(267:276);
    tline           = fgetl(fid);
end
fclose(fid);
DY = num2str(jdate(1));
filenamesavemat = sprintf('%ssta%s_Y%s_D%s.mat',dirname,station{1}(5),DY(1:4),DY(5:end));
filenamesavemat
%%
% if (sum(diff(samprate))~=0)
%     disp('   Sampling rate varies between channels!!!')
% %     return
% end
% Offset between asked start time and retrieved start time
%  [~,stime_wfdisc] = eval(['unix(''e2h ',num2str(stime(1)),...
%      ' ofmt="%#" '')']);
% stime_wfdisc = str2double(stime_wfdisc);
% offset_s = (starttime - stime_wfdisc)*samprate(1);
% length_record = (endtime - starttime)*samprate(1);

% ratiorates = samprate / min(samprate);
length_record = fix(etime-stime) .* samprate;
% Read waveforms
fid = fopen('gparse.w','r','b');
signal_temp = fread(fid,'float32');
fclose(fid);
%========= save data
% signal = zeros(max(length_record),length(wfid));
records = cell(length(wfid),1);
offset = 0;
for is = 1 : length(wfid)
    offset = foff(is)/4;
    signal_is = signal_temp(offset + 1:offset+length_record(is));
%     if not(ratiorates(is)==1)
%         signal_is=resample(signal_is,1,ratiorates(is));
%     end
    records{is}.data = signal_is*calib(is);
    records{is}.Fs_Hz = samprate(is);
    records{is}.stime = stime(is);
    records{is}.etime = etime(is);
    records{is}.station = station{is};%(4:5);
    records{is}.channel = chan{is};
%     if (ratiorates(is)==1)
%         signal(1:length_record(is),is) = signal_is;
%     else
%         signal(1:length_record(is),is) = resample(signal_is,1,ratiorates(is));
%     end
end
records
cmdsave = sprintf(' save %s records samprate',filenamesavemat);
eval(cmdsave)
```

```
function [SUTs, filteredsignals, allfrqsFFT_Hz, alltimes_sec, filterbank] = ...
    analyzeSUT(records,filtercharactfilename, MSCthreshold)

idSc = 1;
idSh = 1;
idWS = 1;
idWD = 1;
idT  = 1;
signals     = zeros(34560000,2);
windSpeed   = zeros(34560000,1);
temperature = zeros(34560000,1);
windDir     = zeros(34560000,1);

Lrecords =length(records);
for ir =1:Lrecords
    switch records{ir}.channel
        case 'BDF'
            Fs_Hz = records{ir}.Fs_Hz;
            switch records{ir}.station(4)
                case 'C'
                    LLC = length(records{ir}.data);
                    signalsC = [records{ir}.data];
                    signals(idSc:idSc+LLC-1,2)=signalsC;
                    idSc = idSc+LLC;
                case 'H'
                    LLH = length(records{ir}.data);
                    signalsH = [records{ir}.data];
                    signals(idSh:idSh+LLH-1,1)=signalsH;
                    idSh = idSh+LLH;
            end
        case 'LKO'
            LLT = length(records{ir}.data);
            temperature(idT:idT+LLT-1) = [records{ir}.data];
            idT = idT + LLT;
        case 'LWS'
            LLWS = length(records{ir}.data);
            windSpeed(idWS:idWS+LLWS-1) = [records{ir}.data];
            idWS = idWS + LLWS;
        case 'LWD'
            FsWind_Hz = records{ir}.Fs_Hz;
            LLWD = length(records{ir}.data);
            Fs_wind_Hz = records{ir}.Fs_Hz;
            windDir(idWD:idWD+LLWD-1) = [records{ir}.data];
            idWD = idWD + LLWD;
    end
end

signals = signals(1:idSc-1,:);
windSpeed = windSpeed(1:idWS-1);
windDir = windDir(1:idWD-1);
temperature = temperature(1:idT-1);

Ts_sec = 1/Fs_Hz;
signals_centered=signals-ones(size(signals,1),1)*mean(signals);

%============================================
cmdloadcharact = sprintf('run(''%s'')',filtercharactfilename);
eval(cmdloadcharact);

%============================================
[SUTs, filteredsignals, allfrqsFFT_Hz, alltimes_sec, filterbank] = ...
    fbankanalysis(signals_centered,filtercharact,Fs_Hz,MSCthreshold);

%============================================
%%
figdisplay = 3;
figure(figdisplay)
%============================================
allcolors = [’b.’;’r.’;’m.’;’c.’;’g.’;’k.’;’rx’;’yx’;’mx’;’rx’;’kx’;’c.’;’k.’;’r.’;’c.’;’m.’;’g.’;’b.’;’k.’;’r.’;’c.’;’m.’;’g.’;’k.’];
idipinf = zeros(P,1);
idipsup = zeros(P,1);
for ip=1:P
    idipinf(ip) = SUTs(ip).indexinsidefreqband(1);
    idipsup(ip) = SUTs(ip).indexinsidefreqband(2);
end
for ip=1:P
    fq_ip = SUTs(ip).frqsFFT_Hz(idipinf(ip):idipsup(ip));
    Rsuptab_ip = 20*log10(SUTs(ip).estimRsup.tabmodcst(idipinf(ip):idipsup(ip),:));
    semilogx(fq_ip, Rsuptab_ip,’.’,’color’,[1 1 1]*0.9)
end
for ip=1:P
    fq_ip = SUTs(ip).frqsFFT_Hz(idipinf(ip):idipsup(ip));
    Rsup_ip = 20*log10(SUTs(ip).estimRsup.modcst(idipinf(ip):idipsup(ip)));
    Rinf_ip = 20*log10(SUTs(ip).estimRinf.modcst(idipinf(ip):idipsup(ip)));
    racN_ip = sqrt(SUTs(ip).Nsupthresholdintheband);
    stdRsup_ip = SUTs(ip).estimRsup.stdmodcst(idipinf(ip):idipsup(ip)) ./racN_ip ;
    semilogx(fq_ip,Rsup_ip,...
        ’o’,’color’,allcolors(ip,1),’markerfac’,allcolors(ip,1))
    hold on
    %             semilogx(fq_ip,Rinf_ip,...
    %                 ’o’,’color’,allcolors(ip,1),’markerfac’,’k’)
    semilogx(fq_ip, Rsup_ip+2*stdRsup_ip,’o’,’color’,allcolors(ip,1))
    semilogx(fq_ip, Rsup_ip-2*stdRsup_ip,’o’,’color’,allcolors(ip,1))
    semilogx(ones(2,1)*fq_ip, ...
        [Rsup_ip-2*stdRsup_ip,Rsup_ip+2*stdRsup_ip]’,’color’,allcolors(ip,1),’linew’,1.5)
end

hold off
grid on
set(gca,’xlim’,[1e-3 6])
set(gca,’ylim’,[-2 2])
xlabel(’frequency - Hz’)
ylabel(’gain ratio - dB’)
set(gca,’fontname’,’times’,’fontsize’,16)
%===================
figure(figdisplay)
% subplot(121)
hold off
title(’Infrasonic signals - IS26’,’fontsize’,24)
% subplot(122)
% hold off
HorizontalSize = 14;
VerticalSize   = 7;
set(gcf,’units’,’centimeters’);
set(gcf,’paperunits’,’centimeters’);
set(gcf,’PaperType’,’a3’);
%         set(gcf,’position’,[0 5 HorizontalSize VerticalSize]);
set(gcf,’paperposition’,[0 0 HorizontalSize VerticalSize]);
set(gcf,’color’, [1,1,0.92]);
set(gcf, ’InvertHardCopy’, ’off’);
```

Chapter 12

Toolbox

```
function  [SUTs, filteredsignals, allfrqsFFT_Hz, alltimes_sec, filterbank] = ...
    fbankanalysis(sigin, ...
    filtercharacteristics, ...
    Fs_Hz,...
    MSCthreshold)
%=========================================================
% synopsis:
%    SUTs = fbankanalysis...
%          (sigin,filtercharacteristics,Fs_Hz,MSCthreshold)
% DFT for discrete Fourier transform
% SCP for spectral components
%     SCP_k = MEAN_{m=1}^M wDFT_{m,k}
%     wDFT_{m,k} = DFT( window .* signal_m, k, Lfft)
%          where signal_m is the m-th segment with overlap,
%          Lfft the DFT length
%          and k the frequency index associated to the
%          frequency k*Fs_Hz/Lfft
%=========================================================
% Remark:
% The 3 parameters, M, Lfft, and stationarity-duration are
% related by:
%              stationarity-duration ~ M * Lfft / Fs_Hz
%
% The choice of M is related to the accuracy of the estimator
% In the following we focus on the stationarity-duration value,
% depending on the frequency band, and on M:
%    stationarity-duration is defined by xx.SCPperiod_sec
%    M is defined by xx.ratioDFT2SCP (overlap not applied)
%    therefore we derive Lfft, regarding the overlap
%    xx.overlapDFT
%=========================================================
% Inputs:
%    sigin: input signals T x 2
%    filtercharacteristics: structure Px1
%         P: number of frequency bands
%         xx.Norder: order of the filter
%                if xx.Norder=0, no filtering
%         xx.Wlow_Hz: lower bound of the frequency band in Hz
%         xx.Whigh_Hz: upper bound of the frequency band in Hz
%         xx.SCPperiod_sec: duration in second
%                over which SPC is performed,
%                expected as stationarity time
%         xx.windowshape: window shape for spectral
%                analysis (ex. 'hann' or 'hamming', ...)
%         xx.overlapDFT: overlap rate for successive
%                DFT (typically 0.5)
%         xx.overlapSCP: overlap rate for successive
%                spectral components (typically 0)
%         xx.ratioDFT2SCP: ratio between period_sec and
%                DFT duration (typical integer value is 5).
%     Fs_Hz: sampling frequency in Hz
%     MSCthreshold: MSC threshold (advised value > 0.99)
%
%=========================================================
% Rk: the spectral components are performed on successive DFTs with
%     overlapping. If M denotes the ratioFFT2DSP and
%     if overlapFFT is 0.5, the number of DFTs is 9
%=========================================================
% Example of filtercharacteristics structure:
%===========================================================
%         xx.Norder: 4
%         xx.Wlow_Hz: 0.02
%         xx.Whigh_Hz: 0.2
%         xx.SCPperiod_sec: 250
%         xx.windowshape: 'hann'
%         xx.overlapDFT: 0.5
%         xx.overlapSCP: 0
%         xx.ratioDFT2SCP: 5.
%===========================================================
% filtercharact(1).Norder           = 4;
% filtercharact(1).Wlow_Hz          = 0.02;
% filtercharact(1).Whigh_Hz         = 0.2;
% filtercharact(1).SCPperiod_sec    = 500;
% filtercharact(1).windowshape      = 'hann';
% filtercharact(1).overlapDFT       = 0.5;
% filtercharact(1).overlapSCP       = 0;
% filtercharact(1).ratioDFT2SCP     = 5;
%===========================================================
%===========================================================
% Outputs:
%     SUTs: structures P x 1
%         xx.estimRsup: Rsup ratio
%         xx.estimRinf: Rinf ratio
%         xx.allMSCs: allMSCs;
%         xx.Nsupthreshold: count of the number of values over
%                   the threshold
%         xx.Nsupthresholdintheband: counts of the number of values
%                   over the threshold in the filter bandwidth
%         xx.frqsFFT_Hz: cell P x 1, each cell consists of
%                   frequency list in Hz of the DFTs
%         xx.SCP = all spectral components
%         xx.indexinsidefreqband = P x 1, indices of the
%                    frequency bounds of each filter
%                    in the xx.frqsFFT_Hz
%         xx.alltimes_sec: cell  Px 1, each cell consists of
%             yy.FFT: time list in second of the DFTs
%             yy.SD: time list in second of the SCPs
%             yy.signals: time list in second of the signals
%===========================================================
% included functions: estimSCPs, estimSUT, fbank
%===========================================================
%===========================================================
overlapDFT       = filtercharacteristics.overlapDFT;
overlapSCP       = filtercharacteristics.overlapSCP;
ratioDFT2SCP     = filtercharacteristics.ratioDFT2SCP;
P                = length(filtercharacteristics);
allfrqsFFT_Hz    = cell(P,1);
alltimes_sec     = cell(P,1);
SUTs             = struct;
T                = size(sigin,1);
%===== filtering SIGIN ==> SIGOUT
[sigout, filterbank]  = fbank(sigin,filtercharacteristics,Fs_Hz);
filteredsignals       = zeros(T,2,P);
for ifilter = 1:P
    taustationary_sec = filtercharacteristics(ifilter).SCPperiod_sec;
    windowingshape    = filtercharacteristics(ifilter).windowshape;
    signals   = sigout(:,:,ifilter);
    [T,d]     = size(signals);
    signals   = signals-ones(T,1)*mean(signals);
    filteredsignals(:,:,ifilter) = signals;
    Tfft_sec  = taustationary_sec/ratioDFT2SCP;
    Lfft      = fix(Tfft_sec*Fs_Hz);
    %==================== spectral analysis ========================
    [allSDs, time_temp, allfrqsFFT_Hz{ifilter}] = ...
        estimSCP(signals(:,1),signals(:,2),...
        ones(Lfft,1), overlapDFT, ...
        ratioDFT2SCP,overlapSCP, Fs_Hz, windowingshape);
    alltimes_sec{ifilter} = time_temp;
    %================================================================
    [estimRsup, estimRinf, spectralmatrix, allMSCs, Nsupthreshold] ...
          = estimSUT(allSDs, MSCthreshold);
     SUTs(ifilter).estimRsup = estimRsup;
     SUTs(ifilter).estimRinf = estimRinf;
     SUTs(ifilter).allMSCs   = allMSCs;
     SUTs(ifilter).Nsupthreshold = Nsupthreshold;
     SUTs(ifilter).frqsFFT_Hz = allfrqsFFT_Hz{ifilter};
     SUTs(ifilter).SCP = allSDs;
     SUTs(ifilter).indexinsidefreqband = [ ...
         ceil((filtercharacteristics(ifilter).Wlow_Hz/Fs_Hz)*length(SUTs(ifilter).frqsFFT_Hz)); ...
         ceil((filtercharacteristics(ifilter).Whigh_Hz/Fs_Hz)*length(SUTs(ifilter).frqsFFT_Hz))];
     SUTs(ifilter).Nsupthresholdintheband = ...
```

```
         nansum(SUTs(ifilter).allMSCs.indexcst(SUTs(ifilter).indexinsidefreqband(1):...
         SUTs(ifilter).indexinsidefreqband(2),:),2);
end
%=======================================================
function [sigout, filterbank] = fbank(sigin,filtercharact,Fs_Hz)
%=======================================================
P           = length(filtercharact);
[T,d]       = size(sigin);
sigout      = zeros(T,d,P);
filterbank = cell(P,1);
for ifilter = 1:P
    if not(filtercharact(ifilter).Norder==0)
        flow = filtercharact(ifilter).Wlow_Hz/Fs_Hz;
        fhigh = filtercharact(ifilter).Whigh_Hz/Fs_Hz;
        fname = filtercharact(ifilter).designname;
        forder = filtercharact(ifilter).Norder;
        switch fname
            case 'fir1'
            fdesign = sprintf('filnum = %s(%i,[%5.8f,%5.8f]);',...
                fname,forder,2*flow,2*fhigh);
            filden = 1;
            otherwise
            fdesign = sprintf('[filnum,filden] = %s(%i,[%5.8f %5.8f]);',...
                fname,forder,2*flow,2*fhigh);
        end
        eval(fdesign);
        sigout(:,:,ifilter) = filter(filnum,filden,sigin);
        filterbank{ifilter}.num = filnum;
        filterbank{ifilter}.den = filden;

    else
        sigout(:,:,ifilter) = sigin;
        filterbank{ifilter}.num = NaN;
        filterbank{ifilter}.den = NaN;

    end
end
%=======================================================
function [allSDs, time_sec, frqsFFT_Hz] = ...
    estimSCP(xU,xR,GREF,overlapFFT, ...
    NaverageFFTs, overlapSD, Fs_Hz, smoothwindow)
%==========================================================================
% Perform the spectral components of the two signals xU et xR.
%==========================================================================
% The code uses the Welch's approach. The signal is shared into DFT
% windows of which the length is the length of GREF, with the
% overlap rate of OVERLAPFFT. Then the specral components is averaged
% on NaverageFFTs DFT blocks. Therefore each spectral block corresponds
% to a time period reported in TIME_SEC.
%
% Inputs:
%    xU: signal observed on the SUT (T x 1)
%    xR: signal observed on the SREF (T x 1)
%    GREF: frequency response of the SREF (Lfft x 1)
%    overlapFFT: between 0 and 1, overlap on the FFT block
%    NaverageFFTs: number of FFT-length to averaging
%    overlapSD: between 0 and 1, overlap on the averaging
%                 Spectral Density block
%    Fs_Hz: sampling frequency in Hz
%    smoothwindow: character word as 'rect', 'hann', ...
%            [default: rectangular]
% Outputs:
%    allSDs.RR = auto-spectrum of SREF
%    allSDs.UU = auto-spectrum of SUT
%    allSDs.UR = cross-spectrum of (SUT,SREF)
%    allSDs.MSC = Magnitude Square Coherence
%    allSDs.Rsup = SUU/SUR
%    allSDs.Rinf = SUR/SRR
%    allSDs.det = determinant of the spectral matrix
%
%    time_sec.FFT: grid of time for the FFT blocks (in second)
%    time_sec.SD: grid of time for the Spectral Density blocks (in second)
%    time_sec.signals: grid of time for the samplig period (in second)
%    frqsFFT_Hz: grid of frequency for the interval [0, Fs_Hz] (in Hz)
%==========================================================================
%==========================================================================
xU   = xU(:);
xR   = xR(:);
N    = length(xU);
Lfft = length(GREF);
sqrtLfft    = sqrt(Lfft);
shiftSignal = fix((1-overlapFFT)*Lfft);
NblocksFFT  = fix((N-(Lfft-shiftSignal))/shiftSignal);
```

```
allFFTsRR   = zeros(Lfft,NblocksFFT);
allFFTsUU   = zeros(Lfft,NblocksFFT);
if exist('smoothwindow','var')
    switch smoothwindow
        case 'hann'
            Hwin = hann(Lfft,'periodic');
        case 'bartlett'
            Hwin = bartlett(Lfft);
        case 'hamming'
            Hwin = hamming(Lfft);
        case 'rectwin'
            Hwin = rectwin(Lfft);
        case 'blackman'
            Hwin = blackman(Lfft);
    end
else
    Hwin     = hamming(Lfft);
end
%========= normalisation
% not useful if only PSD ratios are considered
Hwin = Hwin *sqrt(Lfft/(Hwin'*Hwin));
for ibF  = 1:NblocksFFT
    ibT  = (ibF-1)*shiftSignal+(1:Lfft);
    xU_i = xU(ibT) .* Hwin;
    xU_i = xU_i-mean(xU_i);
    xR_i = xR(ibT) .* Hwin;
    xR_i = xR_i-mean(xR_i);
    allFFTsUU(:,ibF) = fft(xU_i,Lfft)/sqrtLfft;
    allFFTsRR(:,ibF) = fft(xR_i,Lfft)/sqrtLfft;
end
shiftFFTs = fix((1-overlapSD)*NaverageFFTs);
NSD       = fix((NblocksFFT-(NaverageFFTs-shiftFFTs))/shiftFFTs);
allSDs    = struct;
time_sec  = struct;
for ibB=1:NSD,
    indB1 = (ibB-1)*shiftFFTs+1;
    indB2 = indB1+NaverageFFTs-1;
    indB  = fix(indB1):fix(indB2);
    indB  = indB(indB<= NblocksFFT);
    allSDs(ibB).RR  = mean(abs(allFFTsRR(:,indB)) .^ 2,2);
    allSDs(ibB).UU  = mean(abs(allFFTsUU(:,indB)) .^ 2,2);
    allSDs(ibB).UR  = mean(allFFTsRR(:,indB) .* conj(allFFTsUU(:,indB)),2);
    allSDs(ibB).MSC = (abs(allSDs(ibB).UR) .^2) ./...
        (allSDs(ibB).RR .* allSDs(ibB).UU);
    allSDs(ibB).Rsup  = allSDs(ibB).UU ./ allSDs(ibB).UR;
    allSDs(ibB).Rinf  = allSDs(ibB).UR ./ allSDs(ibB).RR;
    allSDs(ibB).det   = (abs(allSDs(ibB).RR) .* abs(allSDs(ibB).UU)) ...
        -(abs(allSDs(ibB).UR) .^2);
end
frqsFFT_Hz = (0:Lfft-1)*Fs_Hz/Lfft;
time_sec.FFT = ((0:NblocksFFT-1)+1/2)*shiftSignal/Fs_Hz;
time_sec.SD =  ((0:NSD-1)+1/2)*shiftFFTs*shiftSignal/Fs_Hz;
time_sec.signals =  ((0:N-1)+1/2)/Fs_Hz;
%==========================================================================
function [Rsup, Rinf, SCPsupeta, MSC, Nsupthreshold] ...
                 = estimSUT(allSDs, MSCThreshold)
%===========================================================
% synopsis:
% estimate the stastitics of interest from the spectral
%     components. Then extract the values over the
%     threshold.
%
% [Rsup, Rinf, SpMatrix, MSC, Nsupthreshold] ...
%    = estimSUT(allSDs, MSCThreshold)
%
%====
% Inputs:
%        allSDs:
%            all spectrals components (provided by the function
%                estimSD.
%        MSCThreshold: MSC threshold, typicall 0.99
%           Rk: the median is not sensible to the outlayers.
%
%====
% Outputs:
%       Rsup: ratio SUU on SUR (see document)
%         structure
%            Rsup.tabmod: table of the modulus of Rsup
%            Rsup.tabmodcst: table of the modulus of Rsup with MSC
%                   above the threshold
%
%            Rsup.tabphase: table of the phase of Rsup
%            Rsup.tabphasecst: table of the phaseof Rsup with MSC
%                   above the threshold
%
%            Rsup.mod: modulus of Rsup estimated on Rsup.tabmod
%            Rsup.modcst: modulus of Rsup estimated on Rsup.tabmod
%            Rsup.stdmodcst: STD estimated on Rsup.tabmod
%                   with MSC above the threshold
%            Rsup.phase: table of the phases estimated on Rsup.tabphase
%                   with MSC above the threshold
%            Rsup.phasecst: phases of Rsup estimated on Rsup.tabphase
%                   with MSC above the threshold;
%            Rsup.stdphasecst: STD estimated on Rsup.tabphase
%                   with MSC above the threshold
%
%       Rinf: ---- idem Rsup
%
%       SCPsupeta: spectral components 3 x Lfft
%           for cells with MSC is over the threshold.
%           For each frequency the spectral components is performed
%           as the mean for all cells over the threshold.
%
%       MSC: structure
%           MSC.tab: table of the MSC
%           MSC.tabcst: table of the MSC with values above the threshold
%           MSC.indexcst: index of the
%               table of the MSC with values above the threshold
%
%       Nsupthreshold: number of values above the threshold
%
%
%===========================================================
%===========================================================

nbblocksAVE = length(allSDs);
Lfft = length(allSDs(1).UU);
indextabMSCsupthreshold = false(Lfft,nbblocksAVE);
indextabMSCsupthreshold([allSDs.MSC]>MSCThreshold) = true;
Nsupthreshold = sum(sum(indextabMSCsupthreshold));

tabUU    = [allSDs.UU];
tabRR    = [allSDs.RR];
tabUR    = [allSDs.UR];
tabMSC   = [allSDs.MSC];

tabMSCwithMSCsupeta = nan(Lfft,nbblocksAVE);
tabMSCwithMSCsupeta(indextabMSCsupthreshold) = ...
    (tabMSC(indextabMSCsupthreshold));

tabUUwithMSCsupeta = nan(Lfft,nbblocksAVE);
tabUUwithMSCsupeta(indextabMSCsupthreshold) = ...
    (tabUU(indextabMSCsupthreshold));

tabRRwithMSCsupeta = nan(Lfft,nbblocksAVE);
tabRRwithMSCsupeta(indextabMSCsupthreshold) = ...
    (tabRR(indextabMSCsupthreshold));
tabURwithMSCsupeta = nan(Lfft,nbblocksAVE);
tabURwithMSCsupeta(indextabMSCsupthreshold) = ...
    (tabUR(indextabMSCsupthreshold));

SCPsupeta      = zeros(3,Lfft);
SCPsupeta(1,:) = nanmean(tabRRwithMSCsupeta,2);
SCPsupeta(2,:) = nanmean(tabUUwithMSCsupeta,2);
SCPsupeta(3,:) = nanmean(tabURwithMSCsupeta,2);

%================================================
%================================================
%================= on Rsup, Rinf======================
tabHUUUR    = [allSDs.Rsup];
tabHURRR    = [allSDs.Rinf];
%================================================
%================================================
%================= on Rsup ======================
%===== real/imaginary part without constraint
tabrealHUUUR = real(tabHUUUR);
tabimagHUUUR = imag(tabHUUUR);
tabmodHUUUR  = sqrt(tabrealHUUUR .^2 + tabimagHUUUR .^2);
tabphaseHUUUR = atan2(tabimagHUUUR,tabrealHUUUR);

%================================================
%================================================
%================= on Rinf ======================
%===== real/imaginary part without constraint
tabrealHURRR = real(tabHURRR);
tabimagHURRR = imag(tabHURRR);
```

```
tabmodHURRR  = sqrt(tabrealHURRR .^2 + tabimagHURRR .^2);
tabphaseHURRR = atan2(tabimagHURRR, tabrealHURRR);

%================================================
%================================================
%================= on Rsup HAT ==================
hatrealHUUUR = nanmean(tabrealHUUUR,2);
hatimagHUUUR = nanmean(tabimagHUUUR,2);
%===== mod/phase part without constraint
hatmodHUUUR   = sqrt(hatrealHUUUR .^2 + hatimagHUUUR .^2);
hatphaseHUUUR = atan2(hatimagHUUUR, hatrealHUUUR);
%================= on Rinf HAT ==================
hatrealHURRR = nanmean(tabrealHURRR,2);
hatimagHURRR = nanmean(tabimagHURRR,2);
%===== mod/phase part without constraint
hatmodHURRR   = sqrt(hatrealHURRR .^2 + hatimagHURRR .^2);
hatphaseHURRR = atan2(hatimagHURRR, hatrealHURRR);

%================================================
%================================================
%================ on Rsup with constraint ======
%===== real part with constraint
tabrealHUUURwithMSCsupeta = nan(Lfft,nbblocksAVE);
tabrealHUUURwithMSCsupeta(indextabMSCsupthreshold) = ...
    (real(tabHUUUR(indextabMSCsupthreshold)));
%===== imaginary part with constraint
tabimagHUUURwithMSCsupeta = nan(Lfft,nbblocksAVE);
tabimagHUUURwithMSCsupeta(indextabMSCsupthreshold) = ...
    (imag(tabHUUUR(indextabMSCsupthreshold)));
%===== module with constraint
tabmodHUUURwithMSCsupeta = sqrt(...
    tabrealHUUURwithMSCsupeta .^2+...
    tabimagHUUURwithMSCsupeta .^2);
tabphaseHUUURwithMSCsupeta = atan2(tabimagHUUURwithMSCsupeta,tabrealHUUURwithMSCsupeta);


%================================================
%================================================
%================ on Rsinf with constraint ======
%===== real part with constraint
tabrealHURRRwithMSCsupeta = nan(Lfft,nbblocksAVE);
tabrealHURRRwithMSCsupeta(indextabMSCsupthreshold) = ...
    (real(tabHURRR(indextabMSCsupthreshold)));
%===== imaginary part with constraint
tabimagHURRRwithMSCsupeta = nan(Lfft,nbblocksAVE);
tabimagHURRRwithMSCsupeta(indextabMSCsupthreshold) = ...
    (imag(tabHURRR(indextabMSCsupthreshold)));
%===== module with constraint
tabmodHURRRwithMSCsupeta = sqrt(...
    tabrealHURRRwithMSCsupeta .^2+...
    tabimagHURRRwithMSCsupeta .^2);
tabphaseHURRRwithMSCsupeta = atan2(tabimagHURRRwithMSCsupeta,tabrealHURRRwithMSCsupeta);

RsuptheowithMSCsupeta = tabmodHUUURwithMSCsupeta .* ...
    tabmodHURRRwithMSCsupeta;

%================================================
%================ if weightingflag = 1 ==========
% a ponderation is applied using the estimated variance
% of the MSC estimate
weightMSCsupeta = (tabMSCwithMSCsupeta .^2) ./  ...
    (1-tabMSCwithMSCsupeta) .* RsuptheowithMSCsupeta;
weightMSCinfeta = 1 ./  ...
    (1-tabMSCwithMSCsupeta) .* RsuptheowithMSCsupeta;
weightingflag = 1;

%================================================
%================================================
%========== on Rsup HAT with constraint =========
%========== and weighting coeffs ================

if weightingflag
    hatrealHUUURwithMSCsupeta = ...
        nansum(tabrealHUUURwithMSCsupeta .* weightMSCsupeta,2) ...
        ./ nansum((weightMSCsupeta),2);

    hatimagHUUURwithMSCsupeta = ...
        nansum(tabimagHUUURwithMSCsupeta .* weightMSCsupeta,2) ...
        ./ nansum((weightMSCsupeta),2);
else
    hatrealHUUURwithMSCsupeta = ...
        nanmean(tabrealHUUURwithMSCsupeta,2);
    hatimagHUUURwithMSCsupeta = ...
        nanmean(tabimagHUUURwithMSCsupeta,2);
end

%===== mod/phase part with constraint
hatmodHUUURwithMSCsupeta = sqrt(...
    hatrealHUUURwithMSCsupeta .^2+...
    hatimagHUUURwithMSCsupeta .^2);

hatphaseHUUURwithMSCsupeta = atan2(...
    hatimagHUUURwithMSCsupeta,...
    hatrealHUUURwithMSCsupeta);

stdmodHUUURwithMSCsupeta   = nanstd(tabmodHUUURwithMSCsupeta,[],2);
stdphaseHUUURwithMSCsupeta = nanstd(tabphaseHUUURwithMSCsupeta,[],2);

%================================================
%================================================
%========== on Rinf HAT with constraint =========
%========== and weighting coeffs ================

if weightingflag
    hatrealHURRRwithMSCsupeta = ...
        nansum(tabrealHURRRwithMSCsupeta .* weightMSCinfeta,2) ...
        ./ nansum((weightMSCinfeta),2);

    hatimagHURRRwithMSCsupeta = ...
        nansum(tabimagHURRRwithMSCsupeta .* weightMSCinfeta,2) ...
        ./ nansum((weightMSCinfeta),2);
else
    hatrealHURRRwithMSCsupeta = ...
        nanmean(tabrealHURRRwithMSCsupeta,2);
    hatimagHURRRwithMSCsupeta = ...
        nanmean(tabimagHURRRwithMSCsupeta,2);
end

%===== mod/phase part with constraint
hatmodHURRRwithMSCsupeta = sqrt(...
    hatrealHURRRwithMSCsupeta .^2+...
    hatimagHURRRwithMSCsupeta .^2);

hatphaseHURRRwithMSCsupeta = atan2(...
    hatimagHURRRwithMSCsupeta,...
    hatrealHURRRwithMSCsupeta);

stdmodHURRRwithMSCsupeta   = nanstd(tabmodHURRRwithMSCsupeta,[],2);
stdphaseHURRRwithMSCsupeta = nanstd(tabphaseHURRRwithMSCsupeta,[],2);



%================ on Rinf with constraint ======
%===== hat's


Rsup.tabmod = tabmodHUUUR;
Rsup.tabmodcst = tabmodHUUURwithMSCsupeta;
Rsup.tabphase = tabphaseHUUUR;
Rsup.tabphasecst = tabphaseHUUURwithMSCsupeta;

Rsup.mod = hatmodHUUUR;
Rsup.phase = hatphaseHUUUR;
Rsup.modcst = hatmodHUUURwithMSCsupeta;
Rsup.stdmodcst = stdmodHUUURwithMSCsupeta;
Rsup.phasecst = hatphaseHUUURwithMSCsupeta;
Rsup.stdphasecst = stdphaseHUUURwithMSCsupeta;

Rinf.tabmod = tabmodHURRR;
Rinf.tabmodcst = tabmodHURRRwithMSCsupeta;
Rinf.tabphase = tabphaseHURRR;
Rinf.tabphasecst = tabphaseHURRRwithMSCsupeta;

Rinf.mod = hatmodHURRR;
Rinf.phase = hatphaseHURRR;
Rinf.modcst = hatmodHURRRwithMSCsupeta;
Rinf.stdmodcst = stdmodHURRRwithMSCsupeta;
Rinf.phasecst = hatphaseHURRRwithMSCsupeta;
Rinf.stdphasecst = stdphaseHURRRwithMSCsupeta;

MSC.tab = tabMSC;
MSC.tabcst = tabMSCwithMSCsupeta;
MSC.indexcst = indextabMSCsupthreshold;
MSC.weightMSC = weightMSCsupeta;
%===========================================================
%==========================================================
```

Chapter 13

Utilitaries

```
%====================================================
% this program executes GPARSE, with
% gparse_temp.par file (considered as temporary)
%
% to extract the signal informations from
% the database. Results are saved into
%      - gparse.wfdisc
%      - gparse.w
% written in this directory (considered as temporary)
% usually after we run CONVERTCSS2matlab.m
%
%====================================================
clear all
% close all
channel     = '(''BDF'',''BDF'',''LWS'',''LWD'',''LKO'')';
% channel     = '(''SHZ'',''SHZ'')';
% sta         = '(''ZAA0'',''ZAA0B'')';%,''ZAA0B'')';
% sta         = '(''PD31'',''PD32'')';
% sta         = '(''USA0'',''USA1'')';
% sta         = '(''I26H1'',''I26C1'')';
% h_starttime = '2015/05/28 01:00';
% h_endtime   = '2015/05/28 23:00';
for mmonth = 6
    for ddaystart=1:2:29
        if ddaystart<10
            h_starttime = sprintf('2015/0%i/0%i 00:10:00',mmonth,ddaystart);
        else
            h_starttime = sprintf('2015/0%i/%i 00:10:00',mmonth,ddaystart);
        end
        ddayend = ddaystart+1;
        if ddayend<10
            h_endtime = sprintf('2015/0%i/0%i 23:50:00',mmonth,ddayend);
        else
            h_endtime = sprintf('2015/0%i/%i 23:50:00',mmonth,ddayend);
        end
        for ista=1:1
            %     sta = sprintf('(%sI07H%i%s,%sI07L%i%s)','''',ista,'''','''',ista,'''');
            sta = sprintf('(%sI26H%i%s,%sI26C%i%s)','''',ista,'''','''',ista,'''');
            % Time transformation performing
            [~,starttime] = unix(['h2e ',h_starttime,' ofmt="%#"']);
            [~,endtime]   = unix(['h2e ',h_endtime,' ofmt="%#"']);
            starttime     = str2double(starttime);
            endtime       = str2double(endtime);
            wlength       = endtime-starttime;

            % Gparse script edition:
            fid=fopen('gparse_temp.par','w');
            %Write the query
            % fprintf(fid, '%s\n', 'open data_source=testbed_archive user=charbit password=sqlmomo');
            % fprintf(fid, '%s\n', 'open data_source=testbed user=charbit password=sqlmomo');
            fprintf(fid, '%s\n', 'open data_source=testbed_archive user=charbit password=sqlmomo');
            fprintf(fid, '%s\n',['query wfdisc select * from sel3.wfdisc where sta in ', ...
                sta, 'and chan in ', channel,' and time between ',num2str(starttime),...
                ' and ',num2str(starttime+wlength),' order by sta,chan,time']);
            fprintf(fid, '%s\n','read waveforms');
            fprintf(fid, '%s\n','write waveforms');
            fclose(fid);
            % Gparse run:
            unix('setenv ORACLE_HOME /cots/oracle/oracle-10.2;');
            unix('setenv D_LIBRARY_PATH $ORACLE_HOME/lib:$ORACLE_HOME/lib32;');
            unix('/ctbto/ims/sm/local/linux/Geotool++/2.3.10/bin/gparse<gparse_temp.par;');

                %======================
                convertCSS2matlab
                %======================
                %======================
%             end
        end
    end
end
%====================== END ============================
% The program reads gparse.wfdisc and gparse.w
% and save useful data in a .mat format
% The name of the file is
%     station+dates.mat
% it is recorded in the directory
%     /dvlscratch/SHI/users/charbit/ProjectIMS2015b/seismicanalysis/records/
%
%===========================================
function convertCSS2matlab()

ISnumber = ’I26’;
dataOrig = ’DATA_IS’;
dirname =sprintf(’/dvlscratch/SHI/users/charbit/ProjectIMS2015b/%s/%s/’,dataOrig,ISnumber);


% read wfdisc
% clear all
% dirname =’/dvlscratch/SHI/users/charbit/ProjectIMS2015b/DATA_IS/’;
fid   = fopen(’gparse.wfdisc’,’r’);
tline = fgetl(fid);
iline = 0;
while ischar(tline)
    iline           = iline+1;
    station{iline}  = strtrim(tline(1:5));
    chan{iline}     = strtrim(tline(6:13));
    stime(iline)    = str2double(tline(14:34));
    wfid(iline)     = str2double(tline(36:43));
    chanid(iline)   = str2double(tline(44:52));
    jdate(iline)    = str2double(tline(53:61));
    etime(iline)    = str2double(tline(62:79));
    nsamp(iline)    = str2double(tline(80:88));
    samprate(iline) = str2double(tline(89:100));
    calib(iline)    = str2double(tline(101:117));
    calper(iline)   = str2double(tline(118:135));
    instype{iline}  = tline(135:136);
    segtype{iline}  = tline(137:144);
    datatype{iline} = tline(144:146);
    clip{iline}     = tline(147:148);
    dir1{iline}     = strtrim(tline(149:209));
    dfile{iline}    = strtrim(tline(210:247));
    foff(iline)     = str2double(tline(248:257));
    commid(iline)   = str2double(tline(258:266));
    lddate{iline}   = tline(267:276);
    tline           = fgetl(fid);
end
fclose(fid);
DY = num2str(jdate(1));
filenamesavemat = sprintf(’%ssta%s_Y%s_D%s.mat’,dirname,station{1}(5),DY(1:4),DY(5:end));
filenamesavemat
%%
% if (sum(diff(samprate))~=0)
%     disp(’   Sampling rate varies between channels!!!’)
% %     return
% end
% Offset between asked start time and retrieved start time
%  [~,stime_wfdisc] = eval([’unix(’’e2h ’,num2str(stime(1)),...
%      ’ ofmt="%#" ’’)’]);
% stime_wfdisc = str2double(stime_wfdisc);
% offset_s = (starttime - stime_wfdisc)*samprate(1);
% length_record = (endtime - starttime)*samprate(1);

% ratiorates = samprate / min(samprate);
length_record = fix(etime-stime) .* samprate;
% Read waveforms
fid = fopen(’gparse.w’,’r’,’b’);
signal_temp = fread(fid,’float32’);
fclose(fid);
%========= save data
% signal = zeros(max(length_record),length(wfid));
records = cell(length(wfid),1);
offset = 0;
for is = 1 : length(wfid)
    offset = foff(is)/4;
    signal_is = signal_temp(offset + 1:offset+length_record(is));
%     if not(ratiorates(is)==1)
%         signal_is=resample(signal_is,1,ratiorates(is));
%     end
    records{is}.data = signal_is*calib(is);
    records{is}.Fs_Hz = samprate(is);
    records{is}.stime = stime(is);
    records{is}.etime = etime(is);
    records{is}.station = station{is};%(4:5);
    records{is}.channel = chan{is};
```

```
%     if (ratiorates(is)==1)
%         signal(1:length_record(is),is) = signal_is;
%     else
%         signal(1:length_record(is),is) = resample(signal_is,1,ratiorates(is));
%     end
end
records
cmdsave = sprintf(' save %s records samprate',filenamesavemat);
eval(cmdsave)
```

Bibliography

[1] B. Alcoverro and A. Le Pichon. Design and optimization of a noise reduction system for infrasonic measurements using elements with low acoustic impedance. *The Journal of the Acoustical Society of America*, 117(4):1717–1727, 2005.

[2] G. C. Carter, C. H. Knapp, and A.H. Nutall. Estimation of the magnitude-squared coherence function via overlapped fast fourier transform processing. *IEEE trans. on Audio and Electroacoustics*, 21(4):337–344, August 1973.

[3] T. B. Gabrielson. In-situ calibration of infrasound array elements. *The Journal of the Acoustical Society of America*, 130(3):1154–1163, 2011.

[4] H. Mack and E. A. Flinn. Analysis of the spatial coherence of short-period acoustic-gravity waves in the atmosphere. *Geophysical Journal of the Royal Astronomical Society*, 26(1-4):255–269, 1971.

[5] A. Nouvellet, M. Charbit, A. Le Pichon, F. Roueff, and I.Y. Che. Coherence parameters estimation from noisy observations. *ITW, Vienna*, October 2013.

[6] A. Nouvellet, A. Le Pichon, M. Charbit, and F. Roueff. Constraint on speed of sound for tdoa based estimator. Technical Report 4600239674/P6H53, CEA, DAM, DIF, F-91297 Arpajon France, November 2012. Available upon request to the authors.