Audio Engineering Society

# Convention e-Brief

Presented at the 131st Convention
2011 October 20–23 New York, NY, USA



# Statistical Analysis of Electro-Acoustic Measurements Sets Using Scilab

Daniele Ponteggia[1]

[1] Audiomatica Srl, Firenze, 50136, ITALY
dp@audiomatica.com

## ABSTRACT

The production management of electro-acoustic systems require the statistical analysis of measurements data. The analysis process should be sufficiently flexible to match the needs of the production process and the number of measured samples should be large enough to ensure the accuracy in statistical terms. Using an open source numerical computation software (Scilab) it is possible to create statistical analysis procedures in a simple and cost effective way. Scilab syntax is simple enough to be acquired within a fairly short time, while data analysis capabilities are very advanced. In this work some sample applications are shown, with minimal code edit the provided examples can be adapted to several real world cases.

## 1. INTRODUCTION

The efficiency and quality of a manufacturing process can be kept under control through measurements on production items. Every production process is unique and reflects the history and philosophy of a company. This asks for flexible analysis tools tailored to the specific production needs. This is a vast and challenging topic, but we will not go here into the details of measurement techniques and Quality Control (QC) practices. Instead, we would like to show some practical examples of statistical analysis on sets of measurements that can be useful in QC management. The statistical analysis is usually done on a sufficiently large, i.e. statistically relevant, number of samples. Since we are here dealing with electro-acoustic devices, this means a sufficiently large set of electrical or acoustical measurements, as an example frequency responses, to be analyzed. While it is possible to include such kind of tools into a measurement and QC software [1], sometimes this approach may be not sufficiently flexible. We found out that it is possible to create powerful analysis tools by complementing a reliable and programmable QC system with a general purpose numerical computation software.

## 2. SCILAB AS PROGRAMMABLE ANALYSIS TOOL

Scilab is an open source numerical computation software. Scilab syntax is simple [2] and similar to the industry standard Matlab software [3]. The learning curve of the software is not steep and plenty of examples are available through Internet and on-line documentation. Scilab is very powerful in the analysis of measurements and sets of measurements: data is

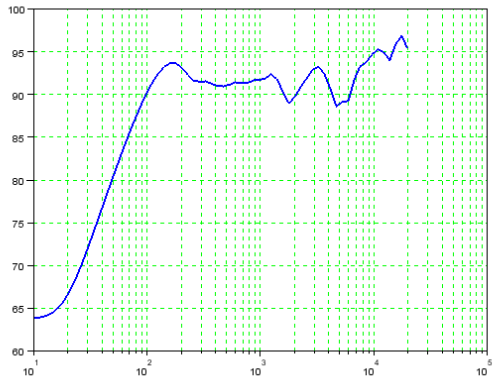

Figure 1 - Frequency response plot from Scilab

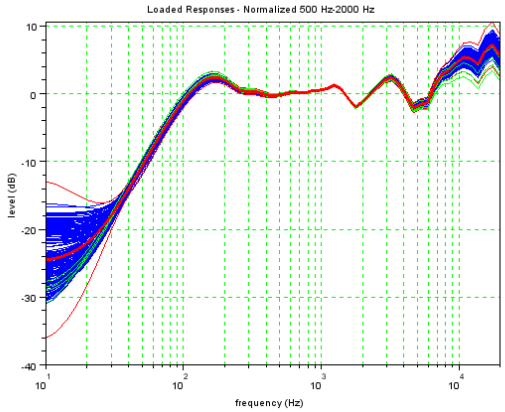


Figure 2 - Loaded responses: (red bold) Average; (red thin) Avg. +/- 3 std. deviation, (green) Outliers

stored and manipulated in vector/matrices, statistical and processing functions are already present as high-level commands and a vast plotting library is available.

### 2.1 Import and plotting CLIO files in Scilab

Here we show the usage of Scilab together with the CLIO measurement system. Files created by the CLIO system are either in binary or text format. In the following examples we will use frequency response CLIO .txt files. Scilab is able to load also CLIO binary files [4]. The same considerations apply to other measurement packages, as far as they publish data formats. A frequency response CLIO .txt file is a simple three column text with frequency, level and phase:

```
Freq [Hz]       dBSPL           Phase [Deg]
10.00           63.86           118.70
11.28           63.91           113.82
…
17555.30        96.84           42.62
19803.29        95.37           70.02
```

Import from formatted text files with Scilab is very simple using the fscanfMat function:

```
-->Mread=fscanfMat('clioresponse.txt');
```

The command loads into the `Mread` matrix the three columns of data from the .txt file. Frequency, level and phase are stored in the matrix columns and are accessible as vectors: `Mread(:,1)`, `Mread(:,2)`, `Mread(:,3)`. Plotting is also very simple. Here is the basic sequence of commands to create the frequency response log-lin graphic in figure 1:

```
-->plot2d(Mread(:,1),Mread(:,2),2);
-->a=gca();
-->a.log_flags="lnn";
-->a.grid=[3,3];
-->a.box="on";
```

## 3. GOLDEN SAMPLE

The search for the golden sample is a critical issue in the quality control process. The golden sample can be defined in product testing as [5]: *"[...] a sample that has all test results in the middle of the nominal range"*.

Every manufacturing process can have a different strategy for this search, here we will show a possible solution using the aforementioned tools. We deal here with a batch of microphones, and we try to find the golden sample among this batch by searching the driver which is nearest the average frequency response (magnitude only) in a specified range.

First we collected the response of every microphone of the batch by placing it in front of a reference loudspeaker driver, taking great care of the repeatability of the positioning against the transducer. Since every microphone capsule can have a different sensitivity, we calculate the sensitivity of each microphone in a given frequency range (500 Hz – 2 kHz) and then apply a correction in order to align the frequency response to a reference value. Then we calculate the average value and standard deviation of the frequency response. Our research policy requires, at this point, to discard items whose response exceed the average response plus/minus three times the standard deviation. We end up with a new reduced set of microphones where the items with a

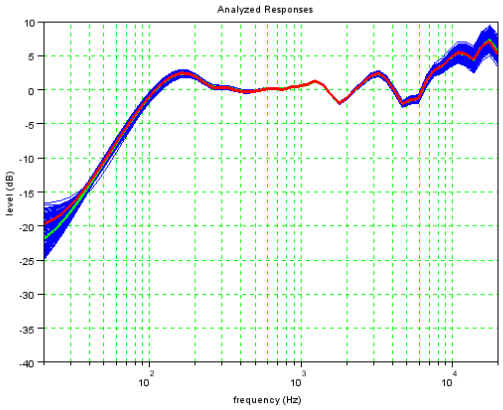


Figure 3 - (red) Golden sample response, (cyan) average response, (blue) batch responses

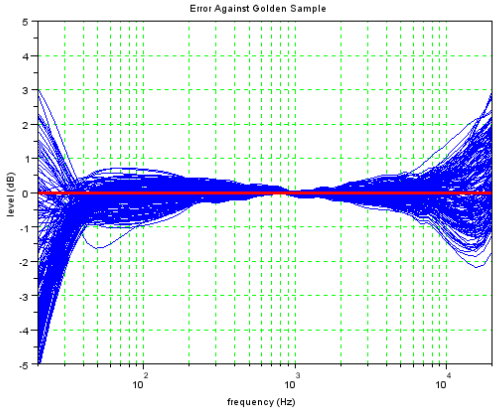


Figure 4 - Error against golden sample

frequency response which deviates too much from the average (outliers) are discarded (Figure 2). At this stage the average frequency response is calculated again on the new set and a research of the item with minimum deviation (in a frequency range 200 Hz – 16 kHz) from the average is carried out. The item found is the golden sample (Figure 3). Figure 4 shows the relative error of each item response against the golden sampe, this plot can be useful to identify limit curves in a QC procedure. Following is the Scilab script that performs the golden sample search. The script will also show on the Scilab console the filename of the golden sample and of the discarded items.

```
//Golden Sample.sce
//
//Load responses in CLIO .txt format from a folder
//and search for the Golden Sample
//
//fming,fmaxg - analysis frequency range
//normsens    - =1 apply sensitivity normalization
//fmins,fmaxs - sensitivity frequency range
//purgeout    - =1 purge outliers
//stdpurge    - std purge multiplier

//data reset
clear;

//Analysis settings
fming=200;
fmaxg=16000;
normsens=1;
fmins=500;
fmaxs=2000;
purgeout=1;
stdpurge=2;

//Load frequency responses from CLIO .txt files
S=dir('*.txt');
filelist=S(2);
for i=1:size(filelist,1) do
    measname(i)=filelist(i);
    Mread=fscanfMat(measname(i));
    measfreq(i,:)=Mread(:,1)';
    measresp(i,:)=Mread(:,2)';
end

//if normsens then apply normalization
if normsens==1 then
    [errval,fmini]=min(abs(measfreq(1,:)-fmins));
    [errval,fmaxi]=min(abs(measfreq(1,:)-fmaxs));
    meassens=mean(measresp(:,fmini:fmaxi),2);
    measresp=measresp-meassens.*.ones(1,size(measresp
,2));
end

//plot frequency response set (after normalization)
f=scf();
drawlater();
for i=1:size(measname,1) do
    plot2d(measfreq(i,:),measresp(i,:),2);
end
a=gca();
a.box="on";
atmax=ceil(max(measresp)/10).*10;
a.data_bounds=[20,atmax-50;20000,atmax];
a.tight_limits="on";
a.log_flags="lnn";
a.grid=[3,3];
if normsens==1 then
    titlestring="Loaded Responses - Normalized "+stri
ng(fmins)+" Hz-"+string(fmaxs)+" Hz";
else
    titlestring="Loaded Responses";
end
title(titlestring);
xlabel("frequency (Hz)");
ylabel("level (dB)");
drawnow();

//compute statistics
measmean=mean(measresp,1);
measdstd=stdev(measresp,1);

//golden sample search frequency range
[errval,fmini]=min(abs(measfreq(1,:)-fming));
[errval,fmaxi]=min(abs(measfreq(1,:)-fmaxg));

//if purgeout then find and purge outliers
drawlater();
measOK=[];
```

```
if purgeout==1 then
    plusstd=measmean+stdpurge.*measdstd;
    minustd=measmean-stdpurge.*measdstd;
    plot2d(measfreq(1,:),plusstd,3);
    plot2d(measfreq(1,:),minustd,3);
    j=1;
    for i=1:size(measname,1) do
        if (sum(measresp(i,fmini:fmaxi)>plusstd(fmini
:fmaxi))+sum(measresp(i,fmini:fmaxi)<minustd(fmini:fm
axi)))==0 then
            measOK(j)=i;
            j=j+1;
        else
            plot2d(measfreq(1,:),measresp(i,:),1);
            disp(measname(i)+" DISCARDED")
        end
    end
else
    measOK=(1:size(measfreq,1))';
end
drawnow();

if measOK==[] then
    disp("Purge excess!!!");
    abort;
end

//plot statistics
plot2d(measfreq(1,:),measmean,3);
e=gce();
e.children(1).thickness=3

//recompute statistics
measmean=mean(measresp(measOK,:),1);
measdstd=stdev(measresp(measOK,:),1);

//golden sample research
//compute difference between average and whole set of
measurements
measerro=measresp(:,:)-ones(size(measresp(:,:),1),1).
*.measmean;
//compute sum of squared errors and find minimum
between valid items (golden)
[errval,igolden]=min(sum(sqrt(measerro(measOK,fmini:f
maxi).^2),2));
igolden=measOK(igolden);
//compute error from golden
golderro=measresp(:,:)-ones(size(measresp(:,:),1),1).
*.measresp(igolden,:);

//show golden
disp(measname(igolden)+" GOLDEN SAMPLE");

//plot frequency response set with golden sample
f=scf();
drawlater();
for i=1:size(measOK,1) do
    plot2d(measfreq(1,:),measresp(measOK(i),:),2);
end
plot2d(measfreq(1,:),measmean,3);
e=gce();
e.children(1).thickness=3
plot2d(measfreq(igolden,:),measresp(igolden,:),5);
e=gce();
e.children(1).thickness=3
a=gca();
a.box="on";
atmax=ceil(max(measresp)/10).*10;
a.data_bounds=[20,atmax-50;20000,atmax];
a.tight_limits="on";
a.log_flags="lnn";
a.grid=[3,3];
title("Analyzed Responses");
xlabel("frequency (Hz)");
ylabel("level (dB)");
drawnow();

//plot error against golden sample
f=scf();
drawlater();
for i=1:size(measOK,1) do
    plot2d(measfreq(1,:),golderro(measOK(i),:),2);
end
plot2d(measfreq(igolden,:),golderro(igolden,:),5);
e=gce();
e.children(1).thickness=3
a=gca();
a.box="on";
atmax=ceil(max(golderro)/5).*5;
a.data_bounds=[20,-atmax;20000,atmax];
a.tight_limits="on";
a.log_flags="lnn";
a.grid=[3,3];
title("Error Against Golden Sample");
xlabel("frequency (Hz)");
ylabel("level (dB)");
drawnow();
```

## 4. CONCLUSIONS

Using Scilab it is possible to easily create flexible post-processing scripts that are able to handle very large sets of measurements. This approach is well suited to deal with typical QC applications that often require statistical analysis tools over large sets of data.

## 5. REFERENCES

[1] Audiomatica, "*CLIO 10 QC Software Extension User's Manual*", http://www.audiomatica.com/download/qcmanual10.pdf

[2] Baudin, M., "*Introduction to Scilab*", Scilab Consortium, 2010, http://www.scilab.org/content/download/1754/19024/file/introscilab.pdf

[3] Rietsch, E. "*An Introduction to Scilab from a Matlab User's Point of View*", 2010, http://wiki.scilab.org/Tutorials?action=AttachFile&do=get&target=Scilab4Matlab.pdf

[4] Ponteggia, D., "*CLIO 10 Sinusoidal File Structure With Import Examples In SCILAB*", http://www.audiomatica.com/download/appnote_001.pdf

[5] http://en.wikipedia.org/wiki/Golden_sample (Accessed August 30, 2011)