

13th March 2019

Noise Barrier Test Rig – Capability report

1 Background

The NZ Transport Agency commissioned Altissimo Consulting to construct a noise barrier test rig able to perform sound insulation and reflection testing using the methodologies in EN 1793-5 and EN 1793-6.

A separate sound insulation rig was previously constructed by the University of Canterbury^{1,2}, but essential components were not dedicated to that rig and it has no longer been available as an operational system for ongoing use by the Transport Agency. Also, as components in the University of Canterbury system were not selected and optimised for this specific application, inefficiencies arose such as with power requirements. Therefore, a new dedicated rig has now been produced for the Transport Agency.

This report describes the system, components, operation, and software now forming the NZ Transport Agency's new dedicated rig. This barrier test rig is at a state where it can be used for field measurements. To further improve and refine the system, the following future developments are recommended:





- Development of a graphical user interface for the testing software
- Field testing on a suitable local noise barrier or equivalent system
- Assembly of casing and tidy up of cabling/system
- Purchase of a dedicated analogue input module

¹ <https://nzta.govt.nz/assets/Highways-Information-Portal/Technical-disciplines/Noise-and-vibration/Research-and-information/Other-research/Measurement-of-the-Acoustic-Performance-of-Traffic-Noise-Barriers-on-Auckland-Motorways-Using-prEN1793-6.pdf>

² <https://nzta.govt.nz/assets/Highways-Information-Portal/Technical-disciplines/Noise-and-vibration/Barrier/Case-studies/NZTA-Noise-Barrier-Case-Study-11-performance-testing-v1.0.pdf>

2 Equipment list

The following equipment has been purchased or developed for the noise barrier test rig. All this equipment is stored in the environmental store at the Transport Agency's CAPTIF Road Research Facility in Christchurch.

Item	Number	Details/Serial Number(s)	Image
Microphone frame	1	n/a	
Tripod	2	Accurate Instruments	
Driver and enclosure	1	JBL SRX712M LF Driver (2262HPL)	
Microphone	3	Aco Type 4152 10667, 10668, 10663	

Microphone cables	3	Ako Pacific 3m cables – BNC to BNC	
National instruments chassis	1	NI cDAQ-9185 1D34SCD	
National instruments 2 channel analogue output	1	NI 9260 with BNC 1CD4DD1	
Pioneer 900W amplifier	1	PFID012218ES	
Dell Laptop running windows XP licence	1	HD1615466	
Ethernet cable	1	20m cat 5	-
Microphone wind shield ø 90mm	3	n/a	

National
Instruments 4
channel input
card

1

NI-9234

**Note: This is currently
borrowed from CPX trailer**



3 File list

The following files used to develop/build and operate the system have been supplied and are saved in InfoHub at:

<https://infohub.nzta.govt.nz/otcs/cs.dll?func=ll&objId=32751507&objAction=browse&viewType=1>

File name	File type	Description
noiseBarrierCAD.zip	.zip archive containing STL and step files	CAD files in .zip archive for 3D printing: <ul style="list-style-type: none"> • Bottom brackets for frame • Frame corners • Microphone holders • Frame clips
noiseBarrierCode.zip	.zip archive	Python scripts for running measurement equipment – Packaged in .zip archive. Also available from bitbucket.com as Git repository

3.1 Python scripts

The following Python scripts are contained within the noiseBarrierCode.zip file

File name	Description
calculateImpulseResponse.py	Calculates and plots impulse response through measured wall
calculateSI.py	Calculates and plots sound insulation of wall
EN_1793_funcs.py	Set of scripted tools for performing measurements using EN1793
octave_filters.py	Script for generating octave and 1/3 rd octave filters
performMeasurements.py	Connects to National Instruments equipment Generates signal Records and stores measurement data
removeDirect.py	Removes direct component of reflected signal

4 Measurement methodology

The barrier test rig allows the testing to the following standards:

- EN 1793-5 describes a method for measuring the sound reflections from a noise barrier in situ
- EN 1793-6 describes a method for measuring the direct sound transmission loss in situ

A simplified overview of the measurement barrier transmission loss is presented below:

1. A measurement of the free-field impulse response of the driver and microphones is made by generating a known input signal and performing a deconvolution of the input and output signals.
2. The sound source and microphone array is placed on either side of the barrier to be tested. An impulse signal is generated, the microphones record the systems response. The impulse response of this system is then calculated
3. The free-field impulse is subtracted from this measured impulse response.
4. The difference between the free-field and the barrier impulse response is calculated.

A simplified overview of the measurement barrier reflections is presented below:

1. A measurement of the free-field impulse response of the driver and microphones is made by generating a known input signal and performing a deconvolution of the input and output signals.
2. The test rig is placed in front of a barrier and an impulse signal is generated, the microphones record the systems response. The impulse response of this system is then calculated
3. The free-field impulse is subtracted from this measured impulse response. An automated evaluation is performed to perform time and amplitude matching between these signals.
4. The impulse response at each location with the direct (free-field) component removed is used to evaluate the barrier reflections.

This report outlines the operation of the noise barrier test rig. The system is currently able to measure the sound insulation of a wall or barrier, and the reflections from a barrier or wall. An image of the physical system is shown below.



Figure 1 Array overview

5 Physical system

5.1 Microphone array

The frame for the microphone array is designed to allow three microphones to be securely held and moved in a repeatable manner to the nine measurement locations. All metallic components are made of lightweight aluminium and the joints between the aluminium components are 3d printed engineering plastic. The frame can be easily dismantled by removing the cap screws in each joint, as shown below.

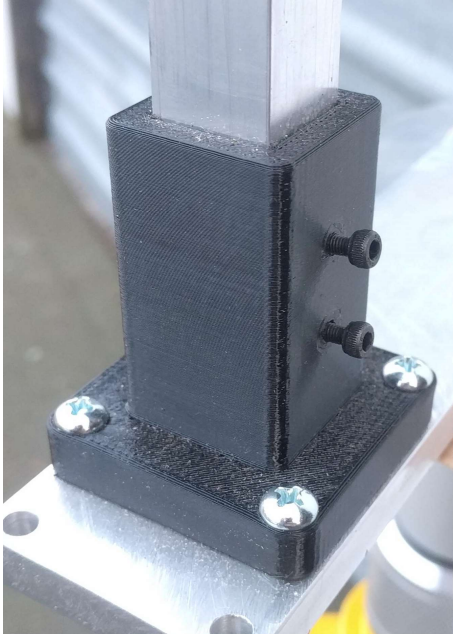


Figure 2 Microphone array – cap screws for dismantling

The horizontal tube can be disconnected from the plastic clips and moved between three standard locations. The three figures below show the frame for the microphone array in the three standard arrangements (without the microphones fitted).

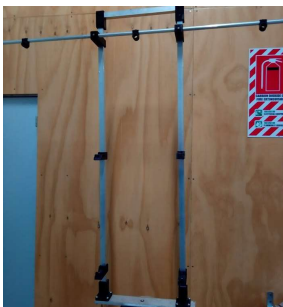


Figure 3 Top location

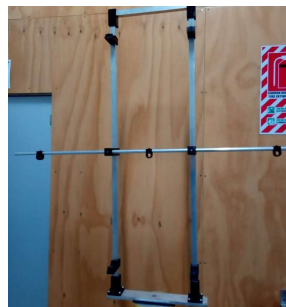


Figure 4 Middle location



Figure 5 Bottom location

Each microphone is held by a lightweight 3d printed plastic microphone holder. These holders were based on the original system used by the University of Canterbury but are made from plastic rather than aluminium.



Figure 6 Microphone holder

The microphone can rapidly be moved between the bottom, middle, and top locations using 3d printed clips. These clips are permanently mounted to the aluminium frame – reducing the likelihood of misalignment during measurements. The clips are shown below:



Figure 7 Microphone array – clips for shifting microphone array

The whole design was modelled using mechanical CAD software. The files are available for reproduction of the system if required. In addition, all the 3d printed components have been exported as compatible file formats, allowing rapid reproduction of the components.

5.2 Microphones

The selected microphones are ACO-Pacific Type 7146. These are class 1, preamplified, integrated microphones. The same microphones are utilised on the Transport Agency's CPX road surface noise measurement trailer, allowing flexibility in the use of equipment. These microphones are suitable for use with the existing National Instruments analogue input equipment on the CPX trailer.

Three microphones are used to reduce the cost of microphones and analogue-to-digital acquisition equipment. The UoC system utilised 9 microphones, but it has been found that the reduction to 3 microphones results in only a minor increase in testing time (A measurement, including setup, takes approximately 30 minutes. Moving the microphones and repeating the measurement takes approximately 5 minutes).

5.3 Speaker

A test speaker enclosure was developed using a readily available Pioneer 12-inch subwoofer which was subsequently replaced with a full-range JBL 12-inch driver. This speaker has a frequency range of 20 – 10kHz, which achieves the EN 1793-5/6 requirements. The enclosure with the JBL driver meets the requirements from EN 1793-5/6. The final driver and enclosure is shown in the figure below:



Figure 8 JBL driver in housing

5.4 Amplifier

The barrier test rig uses a Pioneer 900W full range, bridgeable amplifier. This amplifier is shown below:



Figure 9 Pioneer 900W Bridgeable Amplifier (12V)

This Pioneer amplifier can drive the JBL speaker effectively between 20Hz and 10kHz. The amplifier can be operated utilising a 12V sealed lead acid battery. I

5.5 Data acquisition and signal generation

Data input and output is achieved using National Instruments (NI) equipment. A dedicated 4-slot NI cDAQ-9185 chassis and an NI-9260 analogue output card has been purchased for this test rig. Currently the system is running an NI-9234 4 channel analogue input, borrowed from the CPX trailer.

6 Software

6.1 National Instruments python interface

The calculation method requires several standard signal processing methods signal processing (Fourier transforms, inverse Fourier transforms, deconvolution, etc.). The software for the barrier test rig has been implemented in the software package Python. Python and the required National Instruments (NI) drivers are open source, allowing the software to be developed and operated on a computer without LabView, which requires a specific licence.

Python 3.6 was utilised and was run using the Anaconda Stack, which is designed to allow effective version control and manage the installation of Python modules. To interact with the National Instruments equipment, the NI-MAX application must be installed, along with the associated DAQ drivers, this software is freely available from National Instruments.

The "nidaqmx" Python library allows effective control of the underlying National Instruments drivers. This library can be freely installed via the Anaconda Stack. Implementation of signal generation or data acquisition is simple and is achieved via the use of task objects.

The sources for the main software packages and modules are presented in the following table:

Package	Source	Description
Anaconda Stack	https://docs.anaconda.com/anaconda/install/	Package and version manager
NI-MAX	http://www.ni.com/product-documentation/53366/en/	NI interface and driver installer
nidaqmx	https://anaconda.org/conda-forge/nidaqmx-python	Python library for interacting with NI equipment
matplotlib	https://anaconda.org/conda-forge/matplotlib	Python plotting tools
json	Inbuilt in standard python distribution	Package for managing JSON files
scipy	Inbuilt in standard python distribution. https://anaconda.org/anaconda/scipy	Python signal processing and data analysis tools
numpy	Inbuilt in standard python distribution. https://anaconda.org/anaconda/numpy	Tools for interacting with python arrays
pickle	Inbuilt in standard python distribution	Tools for interacting with a range of external file types

The Anaconda Stack and NI-MAX software can be installed using the default settings and following the automated installation interface. During the Anaconda Stack installation ensure Visual Studio Code is installed. To install the additional Python libraries, utilise Anaconda Navigator Environments tool:

<https://docs.anaconda.com/anaconda/navigator/tutorials/manage-packages/>

6.2 Running software

To run the software after the packages listed above have been installed:

- Either:
 - Extract .zip file with Python files in it to local drive
 - Execute Git clone

"git clone <https://RobWAcoustics@bitbucket.org/altissimo/noisebarriertestrig.git>"

- Open Anaconda Navigator
- Open either Spyder or Visual Studio Code
- Connect the NI-Chassis via ethernet
- Start-up NI-MAX – this should automatically detect the chassis and any attached cards.
- To perform measurements run *"performMeasurements.py"*
- To process data use *"calculateImpulseResponse.py"*, *"calculateSI.py"*, and *"removeDirect.py"*

6.3 Time and amplitude correction

The time and amplitude correction between the free-field and measured impulse response is achieved by shifting the free-field response by small (1/50 of the sampling time) steps and evaluating the least squares difference. The small shift is achieved by Fourier transforming the free-field impulse response, phase shifting this by the required amount, and performing an inverse Fourier transform of the signal.

This inverse Fourier transform is a key requirement for implementing the ESS signal system. The results of a signal being passed through an FFT followed by an inverse FFT is shown below. This process has been automated and allows rapid alignment of two signals.

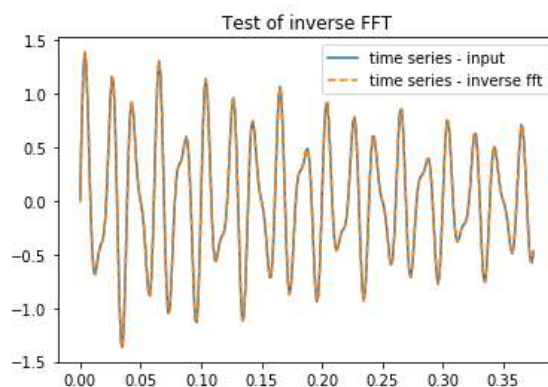


Figure 10 Inverse FFT test case

6.4 Signal generation

Signal generation is achieved simply via a short Python script. Both Maximal Length Sequence (MLS) and an Exponential Swept Sine (ESS) signal generation script have been written. The following Python commands allow for the generation of these signals:

Signal type	Typical python command
MLS	<p>The following code snippet yields an MLS sequence:</p> <pre>mls = scipy.signal.max_len_seq(number_of_tiles)[0] - 0.5</pre> <p>This is repeated and stitched together using the following:</p> <pre>mls_tiled = numpy.tile(mls, number_of_tiles)</pre>
ESS	<p>The scipy "chirp" command can be used to generate exponential swept sine waves:</p> <pre>ess_sweep = scipy.signal.chirp(t=time_vector, f0=start_frequency, t1=length_of_ess, f1=stop_frequency, method="logarithmic")</pre>

These commands and the associated signal conditioning have been incorporated into the *performMeasurements.py* script.

Testing of the MLS and ESS has been performed. The ESS provides a higher signal to noise ratio and will be utilised for all testing and development. The ESS signal has been used in all further work detailed in this report.

During initial testing with the ESS signal it was found that the sudden switch from no excitation to high amplitude, low frequency excitation was causing adverse responses from the speaker. These adverse responses included:

- Clipping of the initial input signal
- Prominent click from the speaker
- Occasional tripping of the amplifier limiting circuits

To reduce these effects a "warm-up" section of signal was included to ensure the ESS excitation was repeatable. In addition, only one ESS should be generated in each measurement run, this allows rapid alignment of the input signal with the measured levels.

A test signal that meets these criteria has been developed, and is described below:

- A 1 second period of silence (except for any speaker hum)
- A 5 second period of MLS (pseudo-random) noise which ramps up linearly to avoid sudden impulses to the speaker
- A 2 second period of silence prior to the start of the ESS signal
- An ESS signal that sweeps between 40Hz to 20kHz over 30 seconds
- A 2 second period of silence

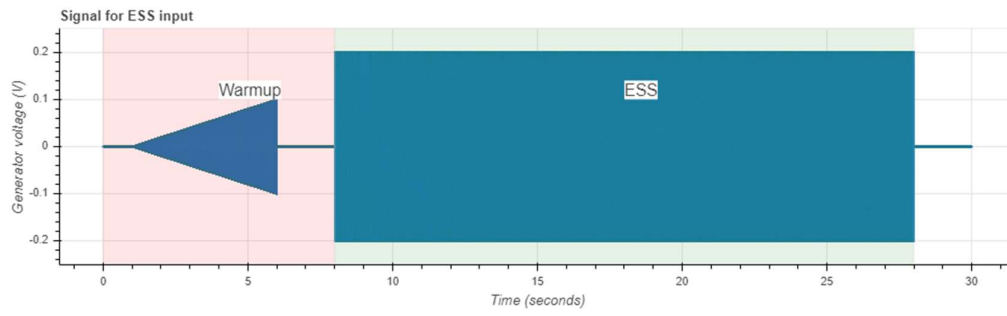


Figure 11 Excitation signal

7 Calculation of sound insulation

The ESS input is repeated at least 5 times to provide adequate signal to noise, this repetition is performed automatically by the *performMeasurements.py* script. The input signal and the recorded microphone signals are stored as a JSON³ file for easy access. A JSON file is a lightweight data interchange format that is human readable, that allows disparate data to be stored in a single file. The JSON format was selected as it allows measurement metadata to be stored in the same file as the measurement data.

The following data is automatically stored by the *performMeasurements.py* script for every measurement:

- Measured levels for microphones 1, 2, and 3
- Time vector for measured levels
- The test signal and associated time vector
- The separated ESS sweep and associated time vector
- Start and stop frequencies for ESS
- Sampling rate
- Number of ESS repetitions

The JSON file format used for storing the measurement data is shown below.

```
{
  "mic0" : <microphone 1 data>,
  "mic1" : <microphone 1 data>,
  "mic2":mic2,
  "test_signal":test_signal.tolist(),
  "test_t":test_t.tolist(),
  "ess_sweep":ess_sweep.tolist(),
  "ess_t":ess_t.tolist(),
  "f0":f0,
  "f1":f1,
  "SR":SR,
  "t_meas":t_meas,
  "n_runs":n_runs
}
```

³ See <https://json.org/> for detailed description of JSON file format

Typical examples of the measured results are shown below.

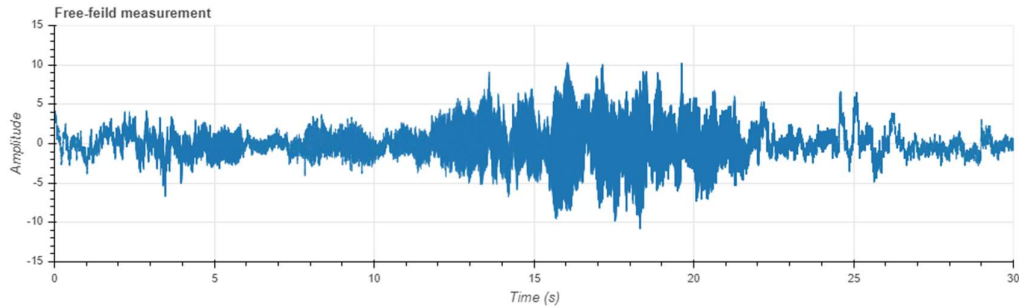


Figure 12 Free-field measurement

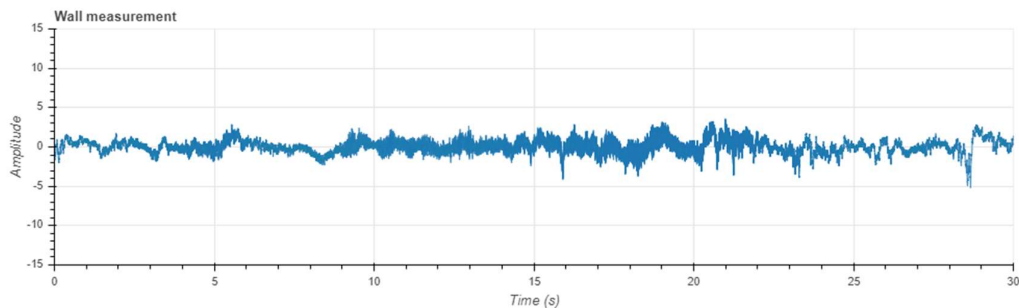


Figure 13 Wall measurement

This data is then post-processed using the *calculateImpulseResponse.py* and *calculateSI.py* scripts. These scripts automate the following process:

- The measured data is automatically aligned automatically by the *calculateImpulseResponse.py* script with the input signal using a correlation function. This removes any time of flight or system delay between the input signal and the measured results. The non-random nature of the input signal means there is only one peak in the correlation (as shown below).

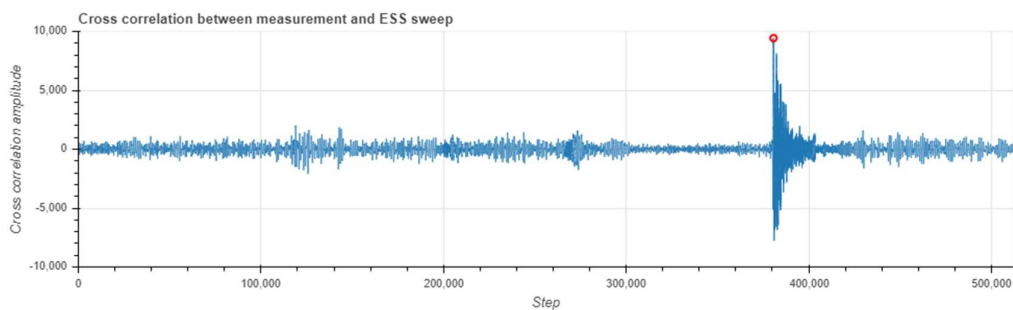


Figure 14 Cross correlation between measured data and ESS

- The measurement data is clipped to extract the response to the ESS, the warmup and cooldown data is discarded. This is performed by the *calculateImpulseResponse.py* script.
- The impulse response between the input and output signal is calculated. This is shifted 10ms to avoid the impulse response wrapping around the zero-time point. An example of the impulse response is shown below:

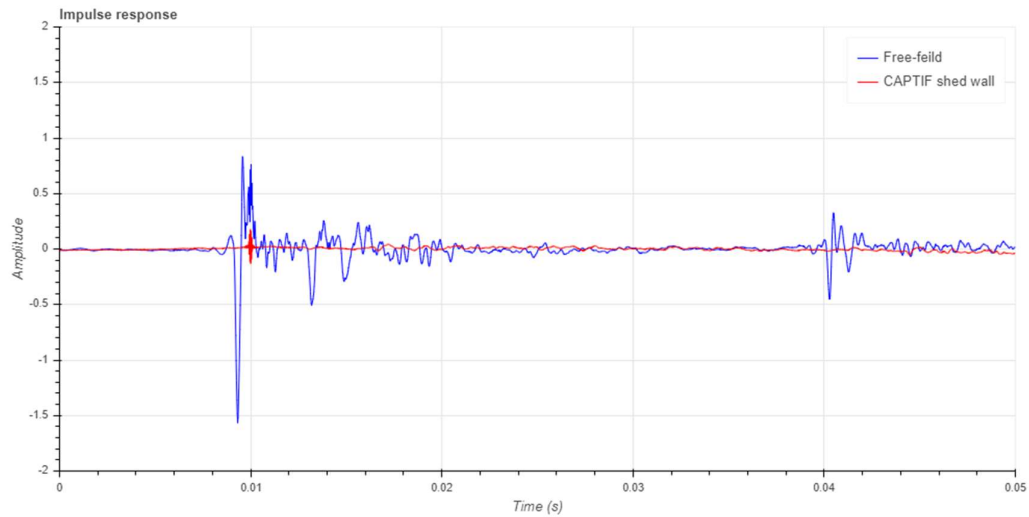


Figure 15 Unwindowed impulse response

- The impulse response is filtered using the Adrienne window. This removes any indirect transmission paths from the impulse response. This is performed automatically by the *calculateSI.py* script.

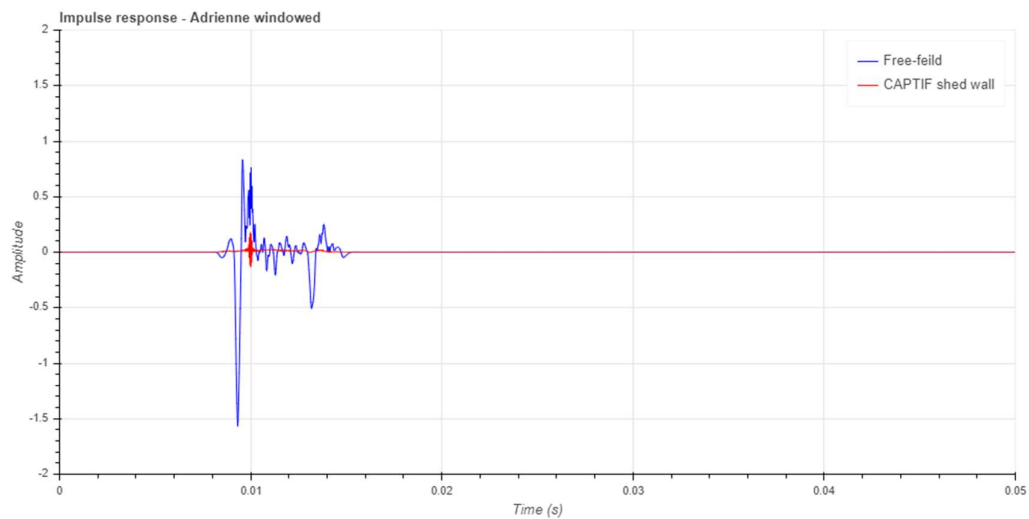


Figure 16 Adrienne windowed impulse response

- The sound insulation is calculated using equation 1 from ISO 1793-6, this calculation is performed automatically by the *calculateSI.py* script. This equation is implemented as described below:
 - The Adrienne windowed free-field and transmitted measurements are Fourier transformed

- The results are integrated across each 1/3rd octave band. This is achieved by binning and summing the FFT lines within each 1/3rd octave band.
 - The transmitted 1/3rd octave band values are divided by the corresponding free-field values.
 - This calculation is summed across all microphone locations and is converted to a log scale.
 - This yields sound insulation values in each 1/3rd octave band.
- The sound insulation from a test at CAPTIF is presented below:

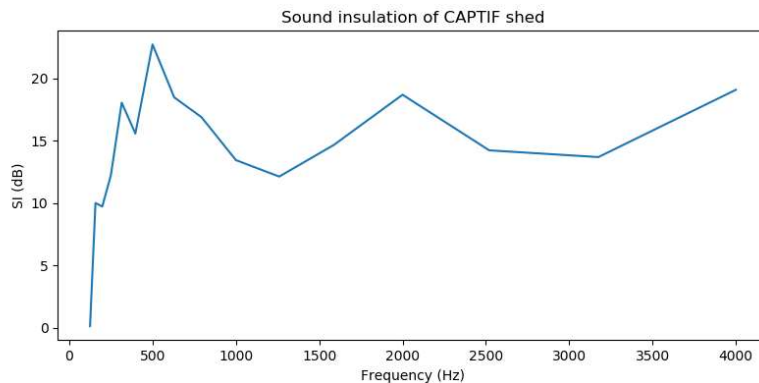


Figure 17 Sound insulation of CAPTIF shed

The low frequency limit of these measurements is defined in Section 4.5.8 of EN 1793-6. This initial testing was based on a wall so does not have a top diffraction component, diffraction was through a doorway approximately 3 m from the measurement rig. The Adrienne window length was left unaltered at 7.9 ms. This yields a low frequency limit of approximately 160 Hz.

A tool for calculating the low frequency limit has been included in the software package as the *calculateLowFreqLim.py* script. This tool generates the Adrienne window and calculates the associated frequency response of the Adrienne window. The low frequency limit is given by the end of the first lobe of the frequency response, this is automatically calculated by the *calculateLowFreqLim.py* script.

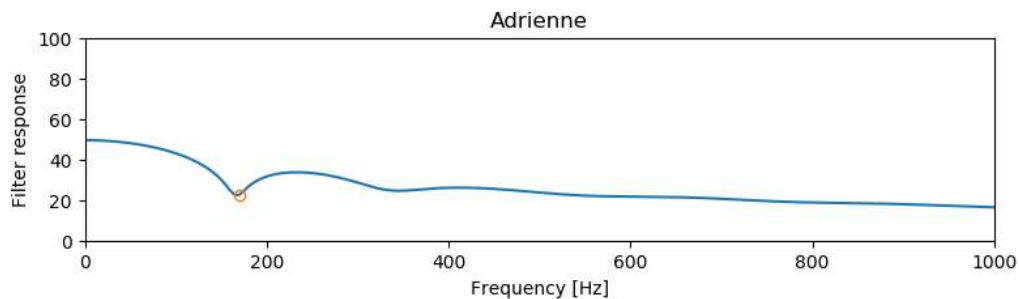


Figure 18 Frequency response of Adrienne window

8 Calculation of sound reflection

The sound reflections are calculated using the same ESS impulse signal and much of the same data processing tools. The direct impulse signal is automatically removed from the reflected signal by subtracting the free-field signal from the measured signal by the *removeDirect.py* script. The direct component of the signal is aligned with the measured signal using the cross-correlation described in Section 4. Fine corrections to the amplitude and alignment between the direct and measured signal are performed using the method described in Section 3.4. Currently these steps are performed manually and have not been included in an automated script.

The raw impulse response data based on reflections off the CAPTIF shed are presented below.

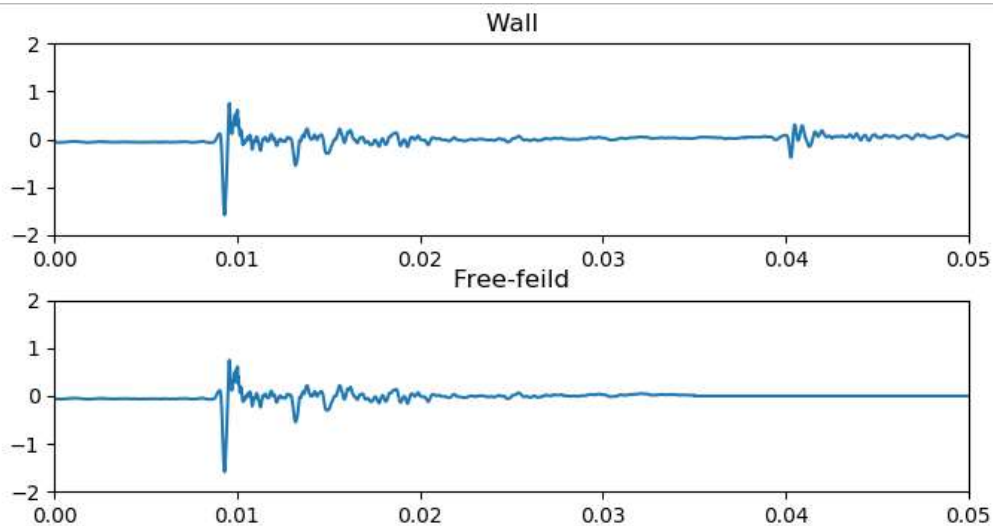


Figure 19 Reflections off the wall of CAPTIF shed

The reflected signal is shown below:

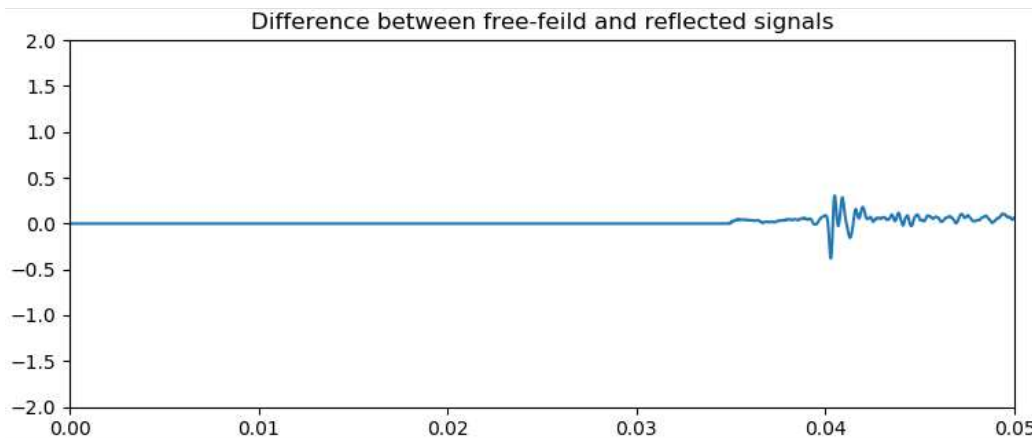


Figure 20 Reflections with free-field response removed

The Adrienne temporal window is applied to the reflected signal. The start of the Adrienne window is placed 0.2 milliseconds before the first peak of the reflected component.

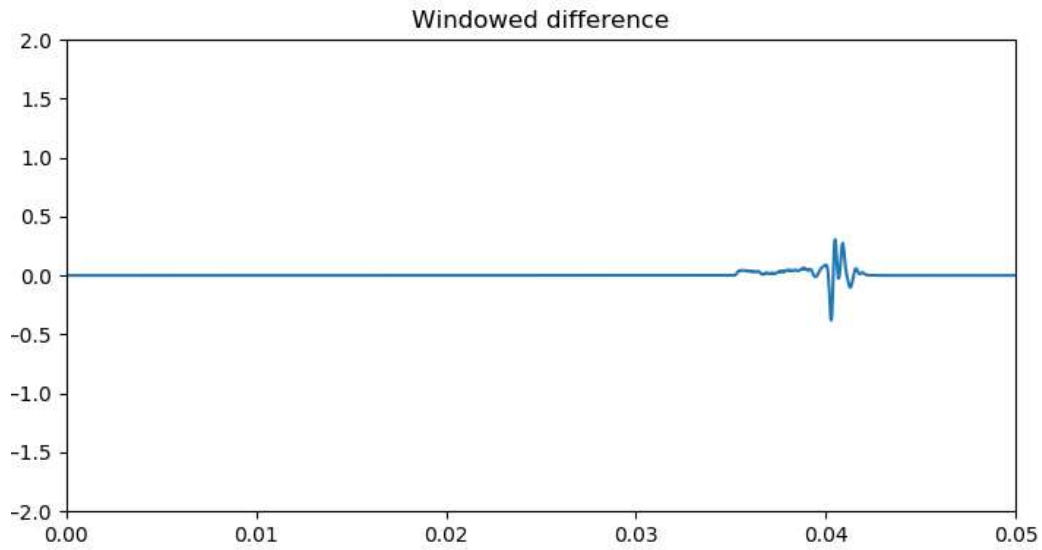


Figure 21 Windowed reflection from CAPTIF Shed wall

The Reflection Index (RI) is calculated using Equation 1 from EN 1793-5:2016, reproduced below:

$$RI_j = \frac{1}{n_j} \sum_{k=1}^{n_j} \left[\frac{\int_{\Delta f_j} |F[h_{r,k}(t) \cdot w_{r,k}(t)]|^2 df}{\int_{\Delta f_j} |F[h_{i,k}(t) \cdot w_{i,k}(t)]|^2 df} \cdot C_{geo,k} \cdot C_{dir,k}(\Delta f_j) \cdot C_{gain,k}(\Delta f_g) \right]$$

The free-field and reflected signals are divided in the frequency domain and are adjusted using three correction factors. The mean of these values across all microphones yields the Reflection Index.

9 Future work

This section presents potential future developments for the barrier test rig. These are not part of the current program of work.

9.1 Further testing of reflection method

Whilst the system can be used to calculate the reflection index further work should be undertaken to streamline the process and ensure repeatable results are yielded.

9.2 Software front-end development

A graphical user interface should be developed to enhance future usability of the software. This front end without allow persons without Python experience to operate the test rig without supervision. The current system requires detailed experience with both National Instruments and Python experience.

9.3 Cases and power supplies

The following cases and housings should be built/procured:

- A case for accommodating the amplifier, NI chassis, batteries, and power supplies
- A case for the tripods and microphone array
- The speaker requires covering and a handle for transport.
- All cabling needs to be bundled and wound onto spools (where required) for rapid deployment

9.4 Input module

The system has been developed using the analogue input module from the CPX trailer. A dedicated input module for the barrier test rig is recommended.

9.5 Field testing

Trial testing should be performed on an existing noise barrier (or similar wall system) to develop an appropriate testing procedures and protocols. The development of an effective, safe, and efficient test methodology should be the key outcome of this testing.

This testing should preferably be performed on a site that is sufficiently separated from the road to avoid traffic management requirements.

Robin Wareing
Principal
rob@altissimo.nz