Spectral Toolkit: User Guide

Wiehan Agenbag
wiehan.a@gmail.com

January 29, 2014

1 Obtaining the software

1.1 Precompiled package

The recommended way to obtain the software created in this project is to download the precompiled all-in-one package that has been created for your platform. A link to where they may be obtained will be posted on the project's GitHub page at `https://github.com/wiehan-a/spectral-toolkit`.

1.2 Source code

The source code for this project may be obtained from its GitHub repository at `https://github.com/wiehan-a/spectral-toolkit`. In order to run the program on your computer, the following software needs to be installed:

Software package	**Version**
Python	2.7
NumPy	$\geq$ 1.7
PySide	$\geq$ 1.2.1
Cython	$\geq$ 0.19.1
FFTW	3.3
Matplotlib	$\geq$ 1.2

Table 1: Software required to run from source

Note also that a C compiler needs to be available on the system—preferably the one that was used to compile the Python distribution installed on your system, or one that is link compatible with it. For Windows users, this will likely be Visual Studio 2008 (MSC v.1500). The compiler version may be checked by running `python` from the command-line, and noting the string between square brackets on the first line of the interpreter prompt.

Before the program can be used for the first time, some of the extension modules need to be compiled. In the command-line, navigate to the folder containing the extracted source code, where files such as `main.py` and `setup.py` and run the command `python setup.py build_ext --inplace`.

On Linux and Mac OS X, it should be enough to have the fftw3 package installed on the system. Windows users would need to go to `http://www.fftw.org/install/windows.html` and follow the instructions for "Precompiled FFTW 3.3.3 Windows DLLs", and then copy the `*.dll` and the generated `*.exp` and `*.lib` files into the project folder hosting the `setup.py` file.

If the previous steps have been completed successfully, the application should now be launchable by double clicking on the `main.py` file or by running `python main.py` from the command-line.

2 Using the Graphical User Interface

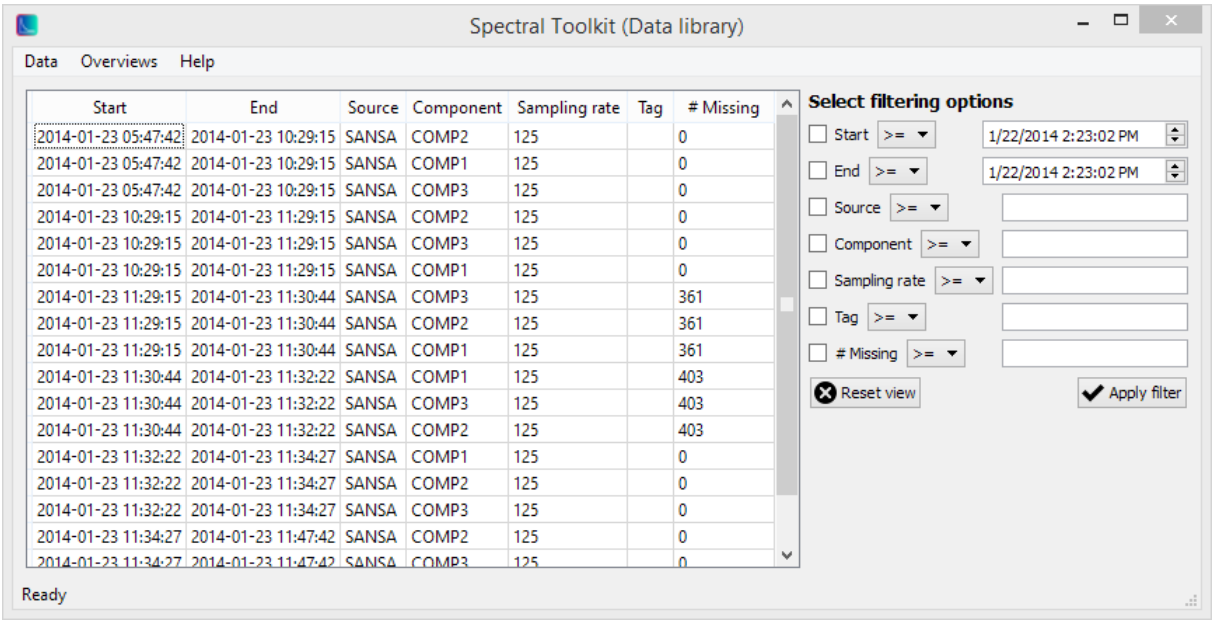

Figure 1: Main application window (Library view)

Figure 1 shows the window that should be visible directly after launching the application. The majority of the the window is taken up by the *record table*, which hosts every data record that has been downloaded through the toolkit. Of course, if this is your first time running the application, the record table will be empty.

2.1 Filtering the record table

On the right hand side is the filtering pane. This is useful when you have amassed a large data library, and want to quickly find a record without scanning through the entire table. In order to activate the filtering of the record table

1. check the boxes next to the fields you want to filter by,

2. choose the operand from the drop down list,
3. enter the *value* to compare records' fields against, and
4. click the "Apply filter" button.

2.2 Using the downloader

Before we can do anything useful with the toolkit, we need to obtain some SQUID signals. This may be done by using the downloader tool (Figure 2) included in this program, which may be launched by clicking on the "Download data" menu item located in the "Data" menu.

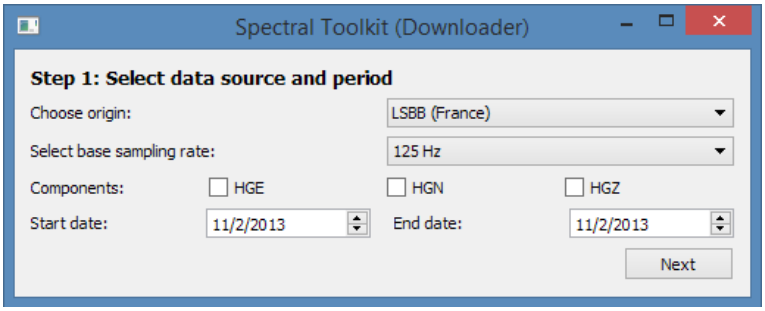

Figure 2: Downloader window—data selection

This window should be fairly self-explanatory. Depending on the source, and its available options, you may configure:

1. the data source (currently only LSBB and SANSA),
2. the data sampling rate (LSBB only),
3. the components you would like to download (LSBB only), and
4. the start and end times of the signal to be downloaded.

Once you have made your selection, proceed by clicking on the "Next" button, which should bring up a window like the one in Figure 3. The expected download size is displayed here, which can be used as a sanity check that the date range selected is correct. This is also a good point to verify that the storage requirements for the intended download can be met on the your secondary storage device.

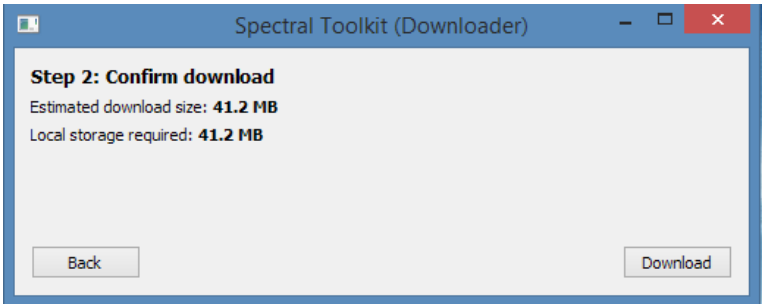

Figure 3: Downloader window—download progress

Once you have confirmed that the download size is not too large, you may initiate the downloading process by clicking "Download". A window such as Figure 4 will report the download progress.

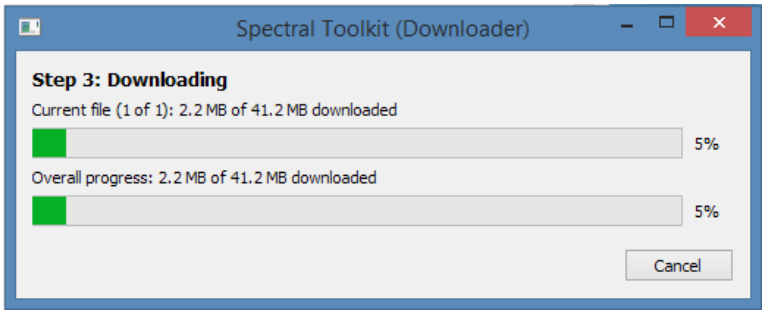

Figure 4: Downloader window—download confirmation

2.3 Importing local data (SANSA)

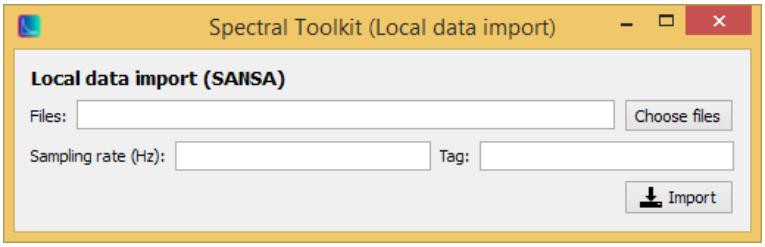

Figure 5: Local data import dialog

The toolkit also allows you to import the raw data produced locally by the SANSA SQUID data acquisition system. This can be done by selecting the "Import local data" option in the "Data" menu. A dialog such as the one in Figure 5 will be shown, which lets you choose the files to be imported, as well as the associated sampling rate of the data. The dialog also lets you set a "tag", which will be displayed in the record table, and which may be useful later to quickly locate the imported data.

2.4 Time/frequency domain overviews (SANSA)

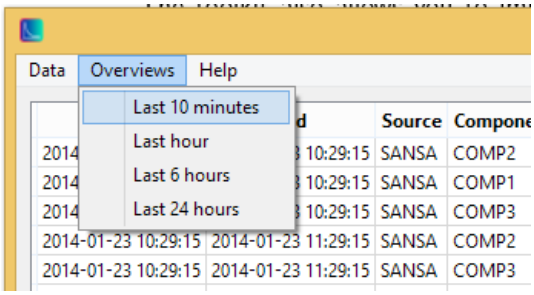

Figure 6: Overviews menu

The "Overviews" menu (see Figure 6) contains a few pre-programmed options to download the most recent data (last 10 minutes, last hour, last 6 hours and last day) and then display the time domain and frequency domain representations of those intervals automatically.

2.5 Interacting with the data records

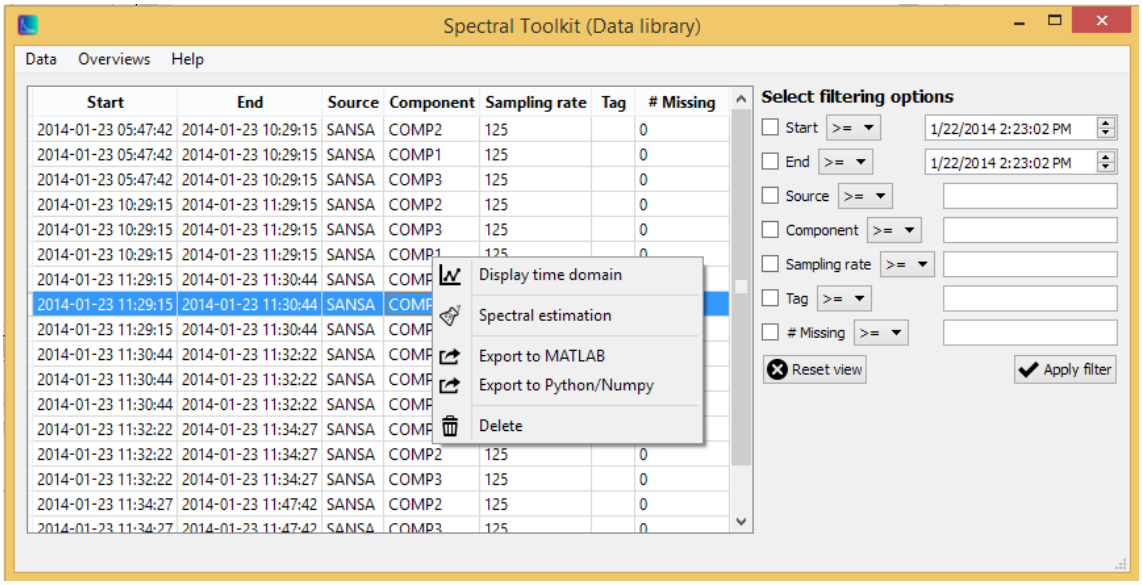

Figure 7: Record selection context menu

The first step in this process is selecting a set of records from the record table in the main application window. The standard selection modifiers such as Control+Click (adding a row to the selection) and Shift+Click (adding a range of rows) are supported. The software expects the records to be contiguous in time, and if they aren't, it will show an error box when you try to do anything with them.

Once you are happy with your selection, you may launch the context menu (see Figure 7) by right clicking anywhere on the selection. The available options are described in Table 2.

2.6 Performing spectral estimation

The spectral estimation tool is launched by selecting the data records of interest, launching the context menu and choosing the "Spectral estimation" option. The first dialog that is displayed is the one used to set the preprocessing options (Figure 8). Table 3 describes the available preprocessing options.

Once you have configure the desired preprocessing options, you may click on the "Next" button to proceed. The program will now display the "Estimation options" dialog (see Figure 9), while it performs the loading of the signal and the preprocessing actions. Table 4 describes the available options.

Option	Description
Display time domain	Allows you to view a time domain plot of the selected signal. This could be useful to see if there are any flux jumps that need to be fixed.
Discontinuity tool	Lets you do a guided "correction" of a signal containing flux jumps. (Not implemented yet.)
Spectral estimation	Launches the spectral estimation tool.
Export to MATLAB	Allows you to export a MATLAB "*.m" script that makes the time domain signal available for manipulation in MATLAB.
Export to Python/Numpy	The same as above, but for Python/NumPy.
Delete	Delete the selected records from the data library and filesystem.

Table 2: Record table selection context menu options and descriptions

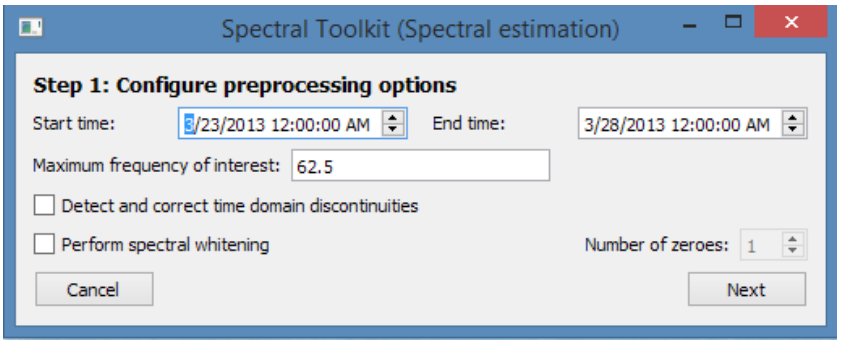

Figure 8: Spectral estimation tool—preprocessing options

Option	Description
Start time & end time	Useful for selecting a sub-interval in the data records to analyse.
Maximum frequency of interest	Lets you to tell the program that you are not interested in frequencies above a certain threshold, so that it may reduce the sampling rate of the signal before analysis is performed. This is strongly advised, especially if you are working with long signals, since a reduced sampling rate drastically lessens the computational and storage burden for the estimation procedure.
Detect and correct time domain discontinuities	Activates the flux jump filter.
Perform spectral whitening	Activates the "Decorrelator" function. This option compresses the dynamic range of the spectrum, which lessens some distortions and allows for crisper peaks, however, the exact amplitudes of the spectrum loses all meaning. Essentially, it "flattens" the spectrum. It is not advised to change the number of zeroes from 1.

Table 3: Preprocessing dialog options and descriptions

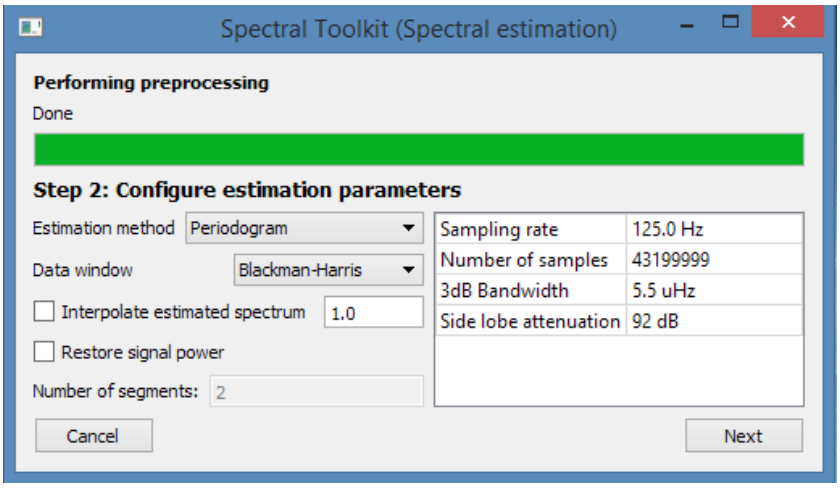

Figure 9: Spectral estimation tool—estimation options

Option	Description
Estimation method	Configures what method the system should use to estimate the spectum. Please refer to section 3 for guidance on what estimation method to select and how to configure it and section 4 for a theoretical overview of the estimation functions.
Data window	Lets you to select the windowing function that should be applied to the data. See section 4.1.
Interpolate estimated spectrum	Activates linear interpolation (zero padding) on the estimated spectrum sequence. The interpolated spectral estimate has the length of the non-interpolated estimate multiplied by the interpolation factor.
Restore signal power	Enforces the condition that the total power contained in the spectral estimate should equate the power in the original signal by scaling the estimated spectrum by the required factor.

Table 4: Spectral estimation dialog options and descriptions

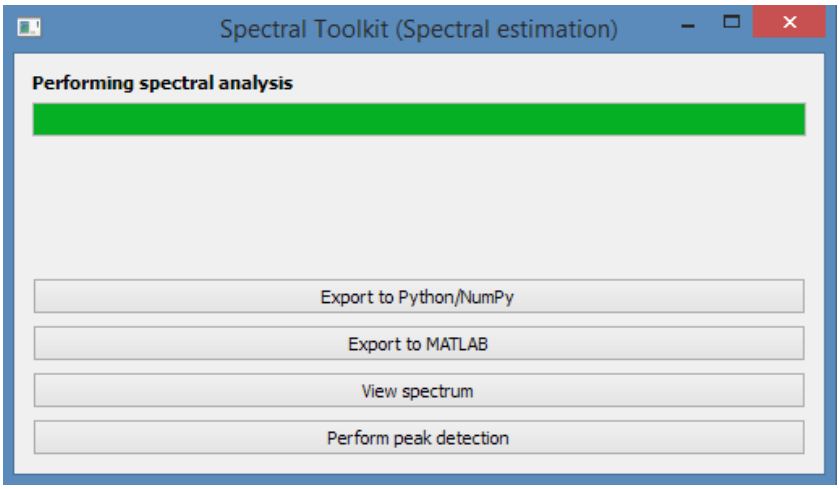

Figure 10: Spectral estimation tool—result dialog

3 Recommended spectral estimation configuration

As a rule, it is not advised to use the Periodogram estimate. It is also not advised to use the Bartlett estimate when the Welch method does roughly the same thing, but better. Lastly, it may be wise to corroborate estimates originating from different techniques, to see how they affect which spectral features are visible.

3.1 Accurate, wide-band long term estimates

Select the Welch estimation method and the Blackman-Harris window. The number of samples will likely be huge, so a segment length of say 10 000–50 000, will provide a high-confidence spectral estimate, while still giving reasonable spectral resolution. Ultimately, you should use your own judgement to make the tradeoff between high confidence (i.e. low variance) and high spectral resolution (longer segment lengths). For amplitude accuracy, select the "Fix signal power" option.

3.2 High-resolution low frequency estimates (short timespans)

It is advised to set the maximum frequency you want in the preprocessing dialog the data (so that you don't waste modeling capacity on high-power features out of band). It is also advised to activate the spectral whitening functionality (unless you really care about amplitude values, though they may not be correct anyways).

None of the estimation techniques implemented in this toolkit beat autoregressive modeling for high frequency resolution when very few signal samples are available. Unfortunately, the model order needs to be well tuned. Until automatic order selection is implemented in a future version, you will need to experiment with this yourself. A good starting point for experimentation is about a third of the number of available samples.

The only other option available here is the Periodogram estimate with a rectangular window (or perhaps a Blackman window, depending), but be aware of its awful estimation performance, and that here it is absolutely crucial to perform the spectral whitening step.

3.3 High-resolution low frequency estimates (long timespans)

Depending on the length of the timespan, you may consider sacrificing some spectral resolution for estimate confidence by employing the Welch method with the segment length equal to say a third or half of the available data samples.

4 Theoretical overview of spectral estimation

Throughout this section, $x[n]$ is the time domain signal being studied. Also, $\hat{S}_{XX}$ refers to the estimated power spectral density (PSD), whereas S_{XX} refers to the true PSD.

4.1 Windowing functions

The windowing function chosen during spectral estimation has an important effect on the quality of the spectral representation produced. In particular, the window function controls whether the estimate achieves good spectral resolution (i.e. frequency components that are very close to each other can be distinguished clearly) or good dynamic range (i.e. frequency components that differ significantly in terms of signal power can be distinguished).

In order to achieve good frequency resolution, one wants to apply a windowing function that has a narrow main lobe. Similarly good dynamic range is determined by how much the side-lobes are attenuated. These two criteria are mutually exclusive, so the window choice often represents a compromise.

Window	**First side-lobe**	**3dB Main-lobe bandwidth**	**Optimal overlap**
Rectangular	-13 dB	0.89	0
Blackman	-58 dB	1.68	0.5
Blackman-Harris	-92 dB	1.9	0.661
Flat-Top	-90.2 dB	3.83	0.78

Table 5: Selected window functions and figures of merit

Table 5 gives a summary of the key window properties. Note that the bandwidth is measured in DFT bins.

4.1.1 Rectangular window

The rectangular window is equivalent to simply truncating the infinite time sequence to N samples. Its effect is therefore there by default, though it may be perfectly masked by the use of another window.

4.1.2 Blackman window

The Blackman window $w[n]$ is given by:

$$w[n] = 0.42659 - 0.49656\cos\left(\frac{2\pi n}{N-1}\right) + 0.076849\cos\left(\frac{4\pi n}{N-1}\right) \tag{1}$$

It achieves a good compromise between main-lobe width and side-lobe attenuation and can be used for applications where both are somewhat important.

4.1.3 Blackman-Harris window

The minimum 4-sample Blackman-Harris window $w[n]$ is given by:

$$w[n] = a_0 - a_1\cos\left(\frac{2\pi n}{N-1}\right) + a_2\cos\left(\frac{4\pi n}{N-1}\right) - a_3\cos\left(\frac{6\pi n}{N-1}\right) \tag{2}$$

with $a_0 = 0.35875$, $a_1 = 0.48829$, $a_2 = 0.14128$ and $a_3 = 0.01168$.

The Blackman-Harris gives excellent sidelobe-attenuation.

4.1.4 Flat-Top window

The Flat-Top window used here is given by:

$$w[n] = a_0 - a_1 \cos\left(\frac{2\pi n}{N-1}\right) + a_2 \cos\left(\frac{4\pi n}{N-1}\right) - a_3 \cos\left(\frac{6\pi n}{N-1}\right) + a_4 \cos\left(\frac{8\pi n}{N-1}\right) \quad (3)$$

with $a_0 = 1$, $a_1 = 1.942604$, $a_2 = 1.340318$, $a_3 = 0.440811$ and $a_4 = 0.043097$.

This Flat-Top window gives excellent sidelobe-attenuation and has a flat 0dB attenuation at its spectral center, making it good for amplitude accuracy, but comes at the expense of the fattest main lobe of the windows considered here.

4.2 Periodogram estimate

The periodogram spectral estimate may be computed directly as:

$$\hat{S}_{XX}(F) = \frac{1}{N}|X[k]|^2 \quad (4)$$

where $X[k]$ denotes the Discrete Fourier Transform of $x[n]$.

4.2.1 Estimator mean and variance

It can be shown that

$$\lim_{N\to\infty} E[\hat{S}_{XX}(F)] = S_{XX}(F) \quad (5)$$

That is, the periodogram estimate is asymptotically unbiased.

Similarly, for an arbitrary spectral density process:

$$\text{Variance}[\hat{S}_{XX}(F)] \approx S_{XX}^2(F) \quad (6)$$

Crucially, this variance does not diminish as $N \to \infty$, so that, at best, the spectral estimate does not improve with larger N. This makes the periodogram estimate undesirable as a spectral estimation technique.

4.3 Bartlett estimate

The periodogram estimate may be modified so $x[n]$ is partitioned into K shorter segments of length L. These shorter periodograms are then averaged together to form a new estimate:

$$\hat{S}_{XX}(F) = \frac{1}{K}\sum_{n=0}^{K-1} \text{Periodogram}\{x[nL : (n+1)L]\} \quad (7)$$

The Bartlett estimate is based on the periodogram estimate, and inherits most of its statistical properties. However, the variance is reduced by a factor of K if the segments are uncorrelated. This requirement may not be fully met, but an improvement in variance is still achieved.

The reduction of estimator variance comes at the price of decreased spectral resolution, since the effective time resolution is increased by a factor of K for each DFT, the effective frequency resolution dimishes by a factor of K.

The key strength of the Bartlett estimate (as well as the Welch procedure described in the next section) is the fact that it emphasises stationary processes in the absolute data window, and deemphasises non-stationary processes. Spurious peaks introduced by noise tend not to persist between subframes so that the averaging over multiple subframes suppresses their power spectral density.

4.4 Welch procedure

In typical application of the periodogram estimate, a window is applied to $x[n]$, either implicitly (in the case of a rectangular window) or explicitly before the DFT is taken to compute $X[k]$. These windows typically taper off at the ends, which emphasises signal content in the middle and deemphasises it at the ends.

In the case of a pure periodogram, this may be used to good effect by including more samples in the sequence x[n] at the beginning and end, allowing one to achieve better spectral resolution (for the given window) with minimal bias introduced by the added samples since they are in deemphasised regions.

When the windowed periodogram is used in the Bartlett estimate, this results in many more sections of $x[n]$ being deemphasised, since data on each side of the subframe boundary is suppressed. This can be overcome be ensuring some overlap between the segments. In the case of maximum overlap, the estimate is given as:

$$\hat{S}_{XX}(F) = \frac{1}{N-L+1} \sum_{n=0}^{N-L} \text{Periodogram}\{x[n:n+L)]\} \tag{8}$$

4.5 AR modeling

Attempts to find a direct decomposition by the Fourier transform of some random signal $x[n]$ onto a set of oscillating complex exponential functions are fundamentally limited in either time or frequency resolution by the Uncertainty Principle. This limitation may be overcome to some extent by imposing some statistical model (or structure) on the process. Spectral estimation then becomes an after-effect of estimating the parameters that allow the model to best fit the data.

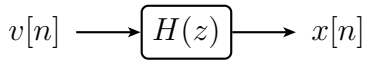

Figure 11: ARMA system diagram

The structure imposed here is that of an autoregressive-moving-average (ARMA) random process, which asserts that the process is the output of some linear filter $H(z)$ driven by a white noise process $v[n]$ (known as the innovations process) with variance σ_v^2, as shown in Figure 11. It is known that

$$S_{XX}(F) = S_{VV}(F) \cdot |H(e^{j2\pi F})|^2 \tag{9}$$

$$= \sigma_v^2 |H(e^{j2\pi F})|^2 \tag{10}$$

Clearly, the shape of $S_{XX}(F)$ is solely determined by $|H(e^{j2\pi F})|^2$, with the innovations variance simply providing a scaling factor. It is further imposed that $H(z)$ is rational and causal, so that it may be written as

$$H(z) = \frac{A(z)}{B(z)} = \frac{\sum_{k=0}^{q} b_k z^{-k}}{1 + \sum_{k=1}^{p} a_k z^{-k}} \qquad \text{ROC: } |z| > r \tag{11}$$

and that all the roots of $A(z)$ (zeroes) and $B(z)$ (poles) lie within the unit circle, so that both $H(z)$ and its inverse $1/H(z)$ are stable. The roots of $A(z)$ can be seen to contribute nulls to the spectrum, whereas the roots of $B(z)$ contribute peaks. If $A(z) = 1$, the process is said to be autoregressive (AR) and if $B(z) = 1$ the process is moving-average (MA).

If the spectrum of the process is dominated by peaks, it is not too disruptive on the estimated spectrum to assume an AR model, especially if one bears in mind that nulls in the spectrum can be approximated by increasing the AR model order p. In this project, the spectra are assumed to be dominated by peaks—something that can easily be verified by looking at the classical estimates of real data.