

paPAM  
User Manual for version 0.872

Developed by:  
Sophie L. Nedelec, James Campbell, Andrew N. Radford,  
Stephen D. Simpson, Nathan D. Merchant

February 19, 2016



# Contents

<b>1</b>	<b>Setting up paPAM</b>	<b>3</b>
1.1	MATLAB . . . . .	3
1.2	MATLAB Compiler Runtime . . . . .	3
1.2.1	Using MCR on OSX and Linux . . . . .	4
<b>2</b>	<b>Using paPAM</b>	<b>6</b>
2.1	Receiver Calibration files . . . . .	6
2.2	User Interface . . . . .	8
2.2.1	Input . . . . .	9
2.2.2	Recorder Calibration . . . . .	11
2.2.3	Analysis Options . . . . .	11
2.2.4	Window Settings . . . . .	15
2.2.5	Time Stamp . . . . .	16
2.2.6	Batch Processing Options . . . . .	16
2.2.7	Settings Profile . . . . .	17
2.2.8	Execute . . . . .	17
2.2.9	Figure Options . . . . .	18
2.2.10	Tools . . . . .	20
2.3	Bandpass Filters . . . . .	20
<b>3</b>	<b>Analysis Types</b>	<b>22</b>
3.1	PSD . . . . .	22
3.2	Impulse . . . . .	25
3.3	Broadband . . . . .	27
<b>4</b>	<b>Advanced features</b>	<b>30</b>
4.1	Memory Issues . . . . .	30

<b>5</b>	<b>paPAM Calculator</b>	<b>32</b>
5.1	Intro and Setup . . . . .	32
5.2	Interface . . . . .	32
5.3	Equations . . . . .	34

# Chapter 1

## Setting up paPAM

paPAM is written in MATLAB, therefore you need MATLAB to execute the program. If you do not have a working version of MATLAB on your computer you can download MATLAB compiler runtime, a freely available program distributed by Mathworks for executing Matlab scripts.

### 1.1 MATLAB

paPAM was built and tested on MATLAB version **R2013a**. While we've tried to make paPAM compatible with older versions of MATLAB, due to the different libraries of functions used across versions, some compatibility issues may arise when using versions other than R2013a. If you cannot get paPAM working on your version of MATLAB, you can download and use MATLAB compiler runtime instead.

To run paPAM with a full version of MATLAB, all you need to do is open the folder `paPAM`, then run the file:

```
PM_Analysis_GUI_Windows.m
```

### 1.2 MATLAB Compiler Runtime

MATLAB compiler runtime (MCR) is a free version of MATLAB available on the Mathworks website. To install MCR, go to the Mathworks website and download the file corresponding with the version of MATLAB paPAM was compiled in:

[mathworks.com/products/compiler/mcr/](http://mathworks.com/products/compiler/mcr/)

As this version of paPAM was compiled in **R2013a (8.1) 32-bit** on windows, this is the version of MCR which you will have to download and install. For Mac and Linux users, you'll have to download the MCR version **R2013a (8.1) 64-bit**

After MCR is installed, it will not be shown in your **All Programs** menu (or in your **Applications** folder for Mac users). To run the MCR, all you have to do is double click the **paPAM.exe** file provided to you and MCR will automatically open the program.

### 1.2.1 Using MCR on OSX and Linux

Using MCR on UNIX computers (Mac and Linux operating systems) requires an extra step. The steps are the same for both Linux and Mac operating systems. To run a MCR file, you have to change some environment variables within the operating system. A small script will be included with the **paPAM.app** file (paPAM on Linux) which will do this for you. To run paPAM on Mac OSX, follow these steps:

1. Find the location of the MCR installation on your hard drive. By default, MCR is installed in the folder:

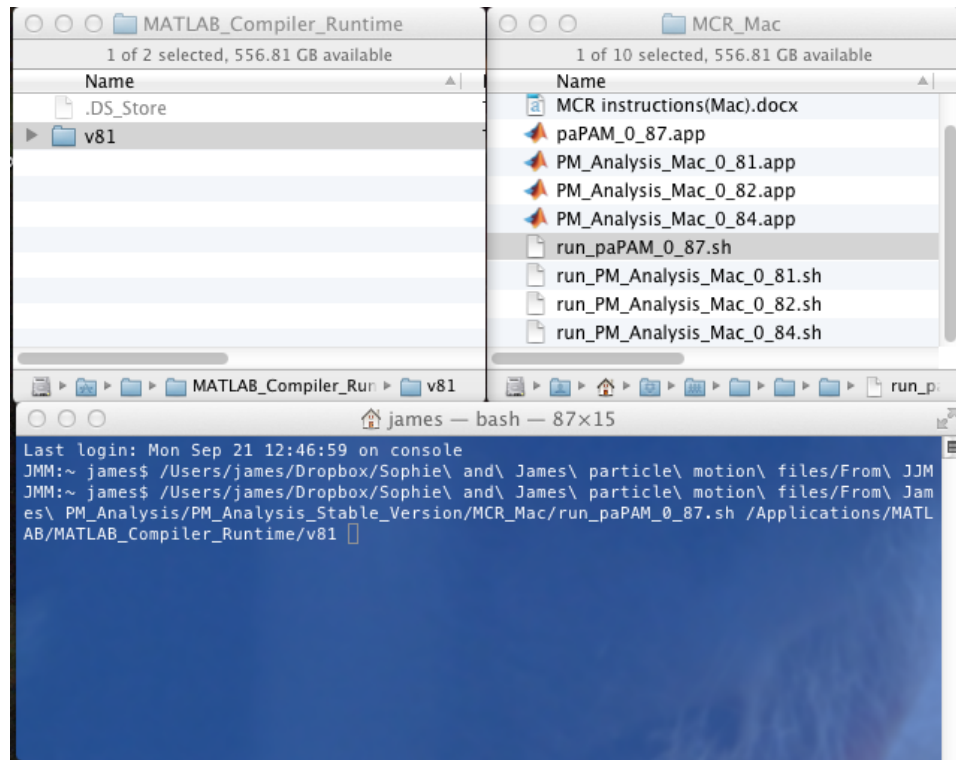
```
/Applications/MATLAB/MATLAB_Compiler_Runtime/v81
```

Once you have located the MCR installation, keep it open in the finder.

2. Open the terminal (**Terminal.app**) and change your working directory to where ever you choose. The working directory will be where all your temporary files are stored during the analysis.

Example: `cd ~/Desktop`

3. Now open a new finder window and find the MCR paPAM files. You should now have two finder windows open in addition to the terminal.



4. Drag and drop the file `run_paPAM.sh` into the terminal window, followed by the folder `v81`. By dragging and dropping items into the terminal, you'll be automatically copying their locations to the terminal window. Your terminal command should now look something like the image above.
5. Press **Enter**, and the terminal will execute the command and open paPAM in MCR.

**Note:** The file `paPAM.app` must be in the same folder as `run_paPAM.sh` for this to work.

## Chapter 2

# Using paPAM

### 2.1 Receiver Calibration files

paPAM allows users to analyze vector and hydrophone data that has been recorded either simultaneously or separately. Receivers and recorders must both be calibrated. Here we describe how to use manufacturer-provided receiver sensitivities for vector sensors and hydrophones in paPAM. Vector sensors typically provide voltage as an output which then has to be converted into the relevant velocity or acceleration units. Hydrophones also typically provide voltage as an output which then has to be converted into the relevant pressure units. To convert voltage into useful acoustic metrics, we need to define the receiver sensitivity values for your specific device(s). Receiver sensitivity values used by paPAM should be formatted in the following units:

1. Pressure:  $dB ( re 1 V/\mu Pa)$
2. Velocity:  $dB ( re 1 V/(m/s))$

paPAM calibrates your data by transforming the raw voltage data into the frequency domain then applying the relevant calibration values to each frequency. paPAM then converts the calibrated frequency domain data back into the time domain for analysis.

To be read by paPAM, the receiver sensitivity values should be arranged into an 8 column `.csv` file, where rows [1 3 5 7] hold the frequency ranges and rows [2 4 6 8] hold the respective receiver sensitivity values in dB units.

From the left to right, your values should be arranged in the .csv file by:  
[x accelerometer, y accelerometer, z accelerometer, Hydrophone]  
Here's an example of what the first few lines of a calibration file might look like:

ExampleCalibrationFile.csv							
100	26.2	100	26.1	100	26.1	100	-172.2
110	26.2	110	26.1	110	26.1	1000	-172.2
120	26.3	120	26.1	120	26.2		
130	26.3	130	26.2	130	26.1		

As shown in the example file, each channel in the calibration file does not need to have the same number of rows. For example, hydrophones typically have a flat calibration range across all frequencies. For a flat response hydrophone, you only need to enter the calibration information for the first and last frequencies within its calibration range. It does not matter what the frequency resolution of your file is, as values for *in-between* frequencies will be estimated by paPAM via interpolation. Thus, the discrete calibration values you provide to paPAM will be automatically converted to the appropriate resolution for the data being analyzed.

paPAM interpolates your calibration information with the following matlab code:

```

1 %Interpolate the first 2 columns of the calibration file
2 %to the same frequency resolution as the sound file
3 %for calibration in the frequency domain
4
5 %cConstants = calibration file
6 %convert from dB to linear units (V/(m/s))
7 linear_cConstants=10.^(cConstants(:,2)./20);
8
9 %f0s = vector containing target frequencies
10 %maxCalibration = maximum calibration frequency of device
11
12 %interpolate linear calibration data
13 interpCalib = spline(cConstants(:,1),linear_cConstants,...
14 f0s(1:find(f0s>=maxCalibrationVal,1,'first')));
15
16 %interpCalib = interpolated calibration information

```

Finally, there are a few general rules which the calibration file must obey to be successfully read by paPAM:



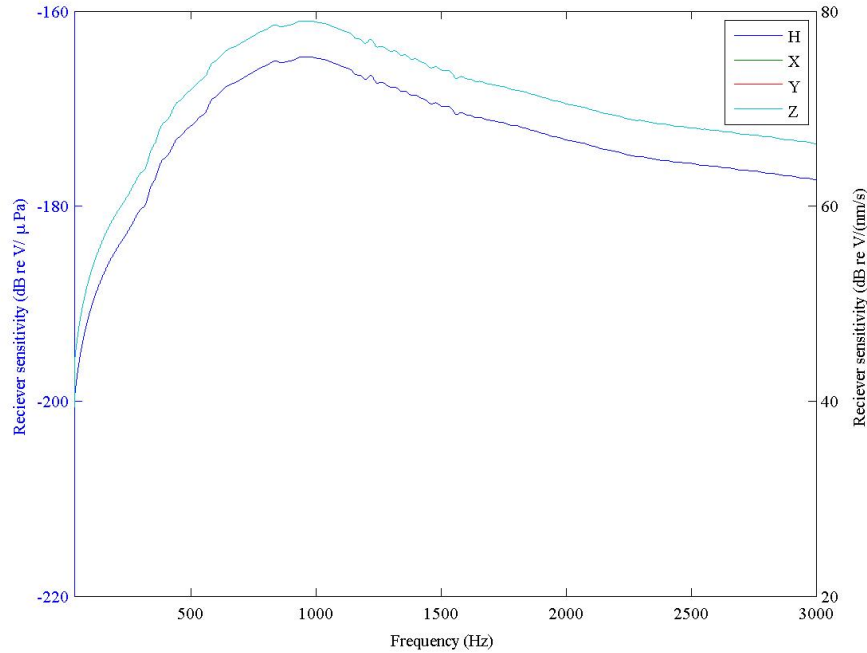
1. All frequencies must be greater than 0:

$$f_n > 0$$

2. There can be no headers or text in the file, as it must contain only numeric data to be correctly imported (trailing blank spaces will be ignored).
3. Rows must contain the frequencies in a monotonically increasing order to properly interpolate the frequency range.

$$f_{row,column} \leq f_{row+1,column}$$

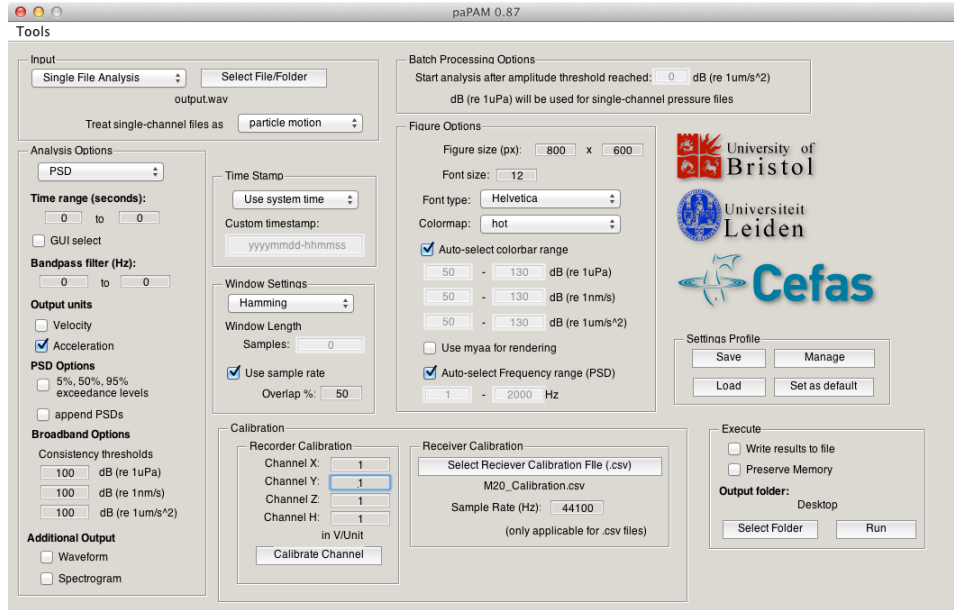
During analysis, the interpolated calibration values are printed in a figure so you can verify the results of paPAM's interpolation.



## 2.2 User Interface

The functionality of paPAM is based around a central graphical user interface (GUI) which houses all the options you need to conduct your analysis.

Below is a brief summary of what the different sections of the GUI are used for:



## 2.2.1 Input

Here you can select which file or folder you'd like to analyze. Input files can be in the following formats:

- **wav & mp3:** Channels [1 2 3 4] in the audio file represent channels [X Y Z Hydrophone] of a vector sensor.
- **csv:** Columns [1 2 3 4] represent the [x y z hydrophone] channels, respectively. If the columns have different lengths, the file will be truncated to the shortest column length to make sure there are an equal number of samples in each channel.

Each file type can hold up to 4 channels, so the x, y, z, and hydrophone data can all be stored in a single file for a given recording session. If your data is currently stored as single channel wav files, see section 2.2.10 for instructions on how to convert them to multi-channel wav files for easier storage and processing, or you can use the freely available program Audacity to make multi-channel audio files.

## Calibration Options

The `Select Receiver Calibration File` selector is where you specify the location of the receiver calibration file (see 2.1). For `wav` and `mp3` files, the sample rate will be extracted from the file automatically by paPAM. For `csv` files, you'll have to manually specify the sample rate you used during data logging in the `Sample Rate (Hz)` field.

## Single channel files

For processing single channel `wav` files, you can specify if the first (and only) channel is either particle motion data or pressure data via the option `Treat single-channel files as`.

## Single or Batch Analysis

For situations where you want to run the exact same analysis on multiple files, you can specify a `Batch analysis`. paPAM will analyze all files located in the specified folder, as opposed to a `single file analysis` where only the specified file will be analyzed. When executing a `batch analysis`, all the options you define for the first file for analysis will be repeated on each proceeding file. More information about batch processing can be seen in section 2.2.6.

## Subsampling large files

While the default MATLAB commands for reading audio or `csv` files are restricted by the amount of memory on your computer, paPAM employs modified versions of these respective importing functions so that your raw data is stored on your harddrive instead of memory. This means that paPAM bypasses the typical size restrictions in file imports, as you are restricted by the amount of free space on your hard drive instead. While you can select files of extremely large sizes, the regions you define within those files for analysis are still restricted by your RAM.

This feature is particularly useful if you want to analyze many small subsections of a large recording which cannot fit into memory. Instead of cutting the audio file into multiple, smaller files, you can load the entire file into paPAM, then add an extra analysis for each subsection of the recording you want to analyze.

For example: On a computer with 2GB of free memory, paPAM will allow you to select `wav` files of up to 4GB (the size limitation of the `wav` file

format). However, when selecting the time range for each analysis sample within that file, these time ranges must select portions of the file which are less than 2GB in size. This feature allows you to skip the step typically required by MATLAB analysis where you have to trim your large wav file in an external program before reading it with MATLAB.

### 2.2.2 Recorder Calibration

In the event that your recorder adds or reduces the gain on your vector sensor during data logging, you can correct for this effect by using the **Channel** fields in the **recorder calibration** section. As an example, for a gain of 10dB applied to your vector sensor measurements during logging (which you might do to increase the resolution of your data and reduce the effects of electrical noise), you would enter a value of 3.16 in the appropriate **channel** field.

$$1 \cdot 10^{\frac{10dB}{20}} = 3.16 \text{ V/unit}$$

paPAM can also automatically calculate the recorder calibration ratios (i.e. recorder gain) from recorded pure tones with the following steps:

1. Record a sine wave (pure tone) of a known voltage directly into your recorder (i.e without the sensor) at the same recording level you used during data collection. For this you'll need an oscilloscope to measure the voltage.
  - You can create a pure tone playback file within paPAM under **Tools > Create Calibration Tone**.
2. Then in paPAM, click **calibrate channel** in the **Recorder Calibration** section, enter your known peak-to-peak voltage (**Reference Voltage**) which you previously measured with your oscilloscope (vpp, in Volts), and select the channel you wish to calibrate.
3. Finally, you can graphically select the start and end points of the calibration file. The maximum peak-to-peak voltage will be calculated from within this region. After selection, the **Recorder Calibration** field for your selected channel will be automatically updated to the calculated calibration value.

### 2.2.3 Analysis Options

The analysis options is where you can specify what types of analyses you will be using, in addition to the frequency time ranges for each analysis.

## Define an analyses type

After selecting a file for analysis, you need to specify what type of analysis you'd like to conduct. Three analysis types are included, each calculating a variety of metrics which are commonly used in underwater acoustic analyses. For information on what metrics are calculated for each type, and how they are calculated, see section 3.

## Define an analyses section

After selecting an analysis type, if you would like to analyze a subsection of a file, as opposed to the whole file, this can be specified here (the default is that the full length and all calibrated frequencies will be analyzed).

- **Time range (seconds):** The selected analysis type will be conducted between these start and end times. If both fields are set to 0, the entire file will be analyzed.
- **Bandpass filter (Hz):** A bandpass filter between this range will be applied to the section before analysis. See section 2.3 for more information about how bandpass filters are applied in paPAM.

For any given file, multiple analysis types can be run over multiple analysis sections in a single execution. For more information on how to add multiple analysis types and sections to a file, see section 2.2.8.

## Output units

The results and figures of your analysis will be displayed in the following units:

- **pressure** (*dB re 1  $\mu Pa$* )
- **velocity** (*dB re 1  $nm/s$* )
- **acceleration** (*dB re 1  $\mu m/s^2$* )

For particle motion channels, you can specify if the output is displayed in either velocity, acceleration, or both using the **Velocity** and **Acceleration** check boxes.

## **PSD Options**

The options listed in this section are only applicable to power spectral density (PSD) analysis. Checking `5%`, `50%`, `95%` `exceedance levels` will output these metrics to both your figures and results file. This is a useful metric for examining the variability over time of your analysis sample.

The `append PSDs` option is only recommended if you are encountering `Out of memory` errors during your analysis. See 4.1 for more information on how to use this option.

For more information regarding the different analysis options and how they are computed, see section 3.

## **Broadband Options**

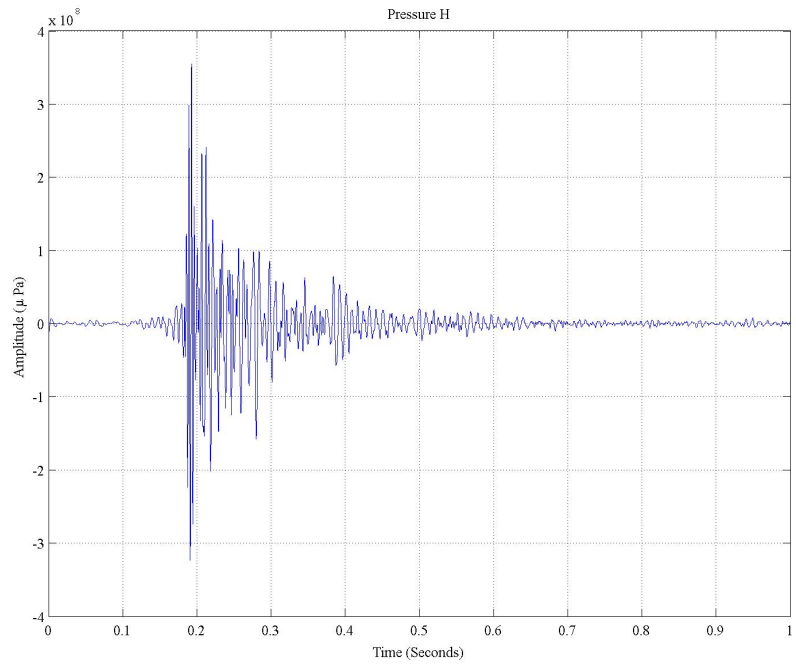
Here you can specify the amplitude thresholds for pressure, velocity, and acceleration during broadband analysis. These thresholds will be used in the consistency analysis (i.e. the percent of time that the amplitude is beyond the given threshold).

For more information regarding the different analysis options and how they are computed, see section 3.

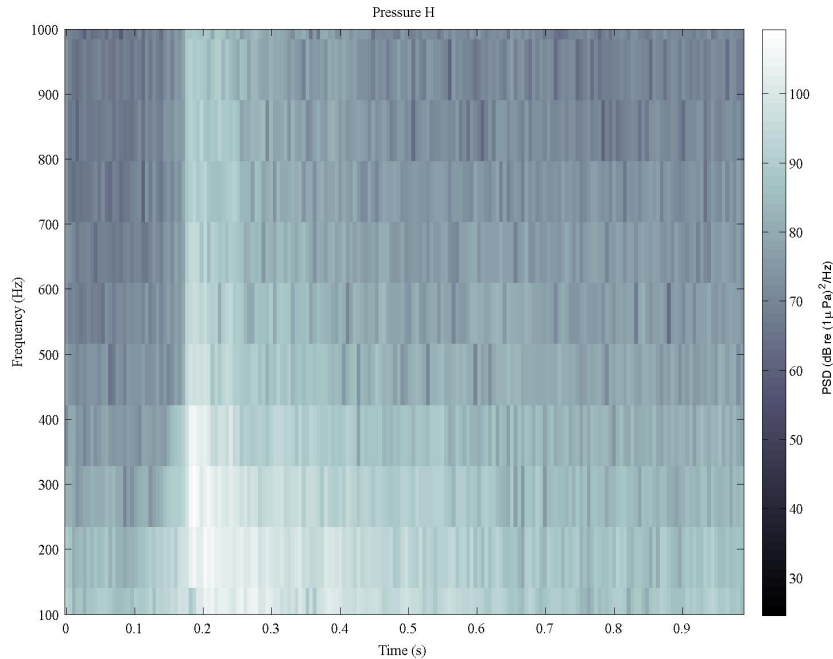
## **Additional Output**

You can select some additional figures for output which are useful for proofing or publishing:

- **Waveform**



- Spectrogram



You can adjust the appearance of these figures using the **Figure Options**.

## 2.2.4 Window Settings

### Window Length

To compute a PSD, we divide the recording into sections of a defined length (*windows*) before conducting a discrete Fourier transform (DFT) on each window. The number of samples in a window determines the temporal and frequency resolution of the analysis. By default, the **Window length** is set to your sampling frequency ( $F_s$ ). This results in a frequency resolution of 1 Hz and a temporal resolution of 1 s. Selecting a higher number than the  $F_s$  will give a lower temporal resolution and a higher frequency resolution, while numbers lower than the  $F_s$  will give a higher temporal resolution and a lower frequency resolution. Thus, the **Window Length** will affect the temporal and frequency resolution of your resulting PSD's.

In addition, your window length must be smaller than the total number of samples in your selected time range for the analysis. If you violate this restriction, a warning message will appear during your analysis.



## Window Type

We apply a *windowing filter* to each window to prevent *spectral leakage* (an artifact of the computation process which reduces the accuracy of the resulting PSD). A windowing filter attenuates the energy in the samples near the beginning and end of the window (this is often in the shape of a bell curve although there are many different types). The windowing filter can be selected via the **Window type** selection box. *Hanning* is selected by default, but you may also select *Hann* (Hanning and Hann windows are very similar and both are fine for PSDs).

## Overlap

To compensate for the energy loss resulting from the *windowing filter*, we allow windows to overlap. By default, an overlap of 50% is used. 50% is sufficient to avoid losing energy due to windowing. Overlapping by more than this will increase the amount of data produced and have the effect of *smoothing* the spectrogram.

### 2.2.5 Time Stamp

When you begin your analysis, by default paPAM will note the current time on your computer and this will be printed in the names of all your resulting figures and results files. All the files printed during a given execution of paPAM will share the same time stamp. This allows you to match up the figures and results of each analysis execution for proofing. If you want to set a custom time stamp (for example, if your system clock is not working correctly), you can specify a custom time stamp here.

### 2.2.6 Batch Processing Options

If you selected a specific time range for your analysis (as opposed to analyzing the entire file), the times selected for the beginning and end of each analysis sample in the first file will be carried over to all the remaining files during batch processing, thus it is important that all your files have synchronized starting points. This is particularly important in impulse analyses, as a difference in a few milliseconds between your files may cause errors in the analysis. To avoid the tedious task of manually clipping all your files, you can use the **Start analysis after amplitude threshold reached**. Here's an example of how you'd use this feature:

1. If you are doing an impulse analysis with repeated strikes at constant intervals (10 seconds between strikes for example), you can use the occurrence of the first recorded strike to synchronize your files.
2. If the impulse sounds reach an amplitude of  $130\text{ dB}$  (*re*  $1\mu\text{m/s}$ ) at your vector sensor, you can set your threshold value as  $120\text{ dB}$  (*re*  $1\mu\text{m/s}$ ). This will result in the beginning portion of the file, everything before the occurrence of the first strike, being removed from your analysis.
3. Now you can define your **time range** as 9.5–10.5 seconds. All files will be synchronized so that 0 seconds in each file is set to the occurrence of the first impulse sound, and your impulse analysis will encompass the 1 second period when the next strike occurs.

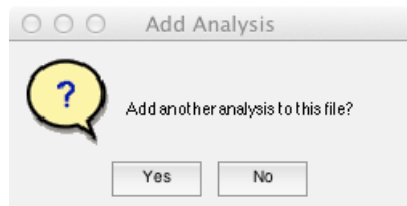
### 2.2.7 Settings Profile

As it takes some time to configure all your analysis settings, you can save your current settings to a profile for quick access later on. Press **Save** to save all your currently selected settings to a profile, and **Load** to load your previously saved settings. By pressing **Set as default**, your current settings will be automatically loaded the next time you start paPAM.

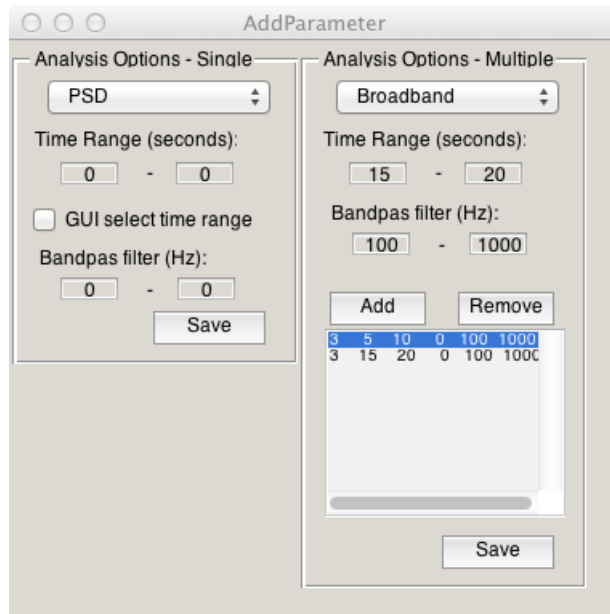
### 2.2.8 Execute

When you're ready to run your analysis, press **Select Folder** to set the location for your results and figures then press **Run** to start the analysis. If **Write results to file** is not selected, you results will only be displayed in the MATLAB command window.

After the first analysis sample is processed (i.e. the first time range you defined), you will be asked if you want to process additional samples within the file:



If you select yes, the following window will open where you can define additional samples to be analyzed:



To add a single additional sample, enter your parameters in the **Analysis Options - Single** and press **Save**. In addition, you can add a series of additional samples to be analyzed in the **Analysis Options - Multiple** frame. Use the **Add** and **Remove** buttons to populate the list box, and then press **Save** to execute the analyses.

When you analyze multiple samples within a file, the loading and calibration of the selected file will only occur once so the analyses will execute much faster as compared to defining all the samples and executing the analysis one-at-a-time from the main GUI. For batch processing, all the samples you define in the first file will be analyzed on all the files found within the selected folder.

For information about the **Preserve Memory** option, see 4.1

### 2.2.9 Figure Options

Here you can define the appearance of your figures which are generated during analysis.

#### Use myaa for rendering

myaa is a freely available script downloaded from the Mathworks file exchange:

<http://www.mathworks.com/matlabcentral/fileexchange/20979-myaa-my-anti-alias-for-matlab>

myaa was written by Anders Brun and can be used and distributed freely so long as the following copywrite notice is presented along with the use of myaa:

Copyright (c) 2009, Anders Brun All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

\* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer. \* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Matlabs default renderer can produce saved images which look quite different from the figures they came from. If you are having problems with the appearance of your saved images, you can try using `myaa` by selecting the `Use myaa for rendering`. `myaa` does wysiwyg renderings and also applies its own anti-aliasing method to smooth out the resulting images. This can be useful option if you are trying to make publishable, high quality figures.

Alternatively, if you want more control over the appearance of your figures, you can reopen the saved `.fig` files in matlab. All the original data

is stored within the `.fig` files, so you can rerender any figures you already saved through this method.

### 2.2.10 Tools

The `Tools` menu at the top of the GUI contains some additional features.

#### Creating multi-channel `.wav` files

For easier storage and processing of your data, you can use the `Create multi-channel wav files` tool to convert single-channel `wav` files into multi-channel files. If you have large amounts of data that you need to convert, you can use the `Batch` option to let paPAM scan a folder and automatically arrange `wav` files with similar names into multi-channel files.

To use the `Batch` feature, you need to specify the naming convention used by the `wav` files in the `Parsing mask` field. In the `Batch` GUI, there are a variety of examples and instructions on how to enter a correct `Parsing mask`.

## 2.3 Bandpass Filters

During any analysis, a bandpass filter is applied to your data up to two times:

1. **Calibration Range:** While paPAM is converting your raw data into calibrated units, a band pass filter is applied around the minimum and maximum defined calibration ranges. This is to ensure that any sounds occurring outside your calibrated range do not influence your analysis.
2. **Analysis Range:** In the `Analysis options`, the user can specify a bandpass range to be applied. This bandpass filter is applied in addition to the bandpass filter already applied to the calibration range. If the user selected 0 Hz for their `low pass` range (maximum frequency), no bandpass will be used here.

All bandpass filters in paPAM are executed in the same manner. Below is the matlab code used for each bandpass:

```
1 % Define butter filter
2 [b,a] = butter(3,[HighPassFrequency LowPassFrequency]./...
```

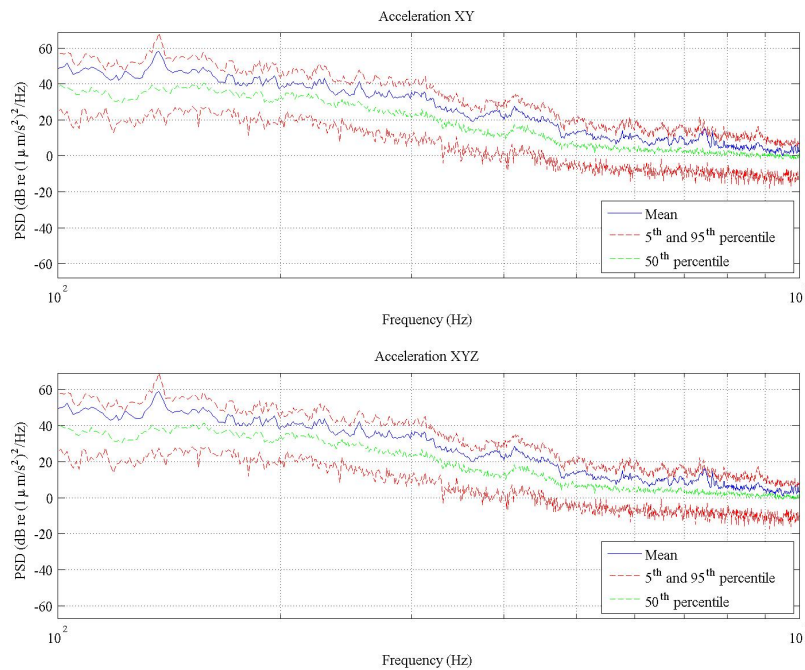
```
3     (SampleRate./2));  
4 % Apply filter to selected data  
5 output = filter(b,a,data);
```

# Chapter 3

## Analysis Types

This section contains detailed information on how each metric is calculated in paPAM. When applicable, both the an example of matlab code (gray boxes), and the mathematical formulas for each analysis are given.

### 3.1 PSD



The PSD analysis will produce PSD figures in addition to `csv` files reporting the dB power levels for each window segment in the analysis. On the main GUI, the options which affect the PSD figures and results are:

- Window Settings
  - Window type:  
In this version of paPAM, only the Hamming window is active and will be selected automatically. Other window types will be implemented in later versions of paPAM
  - Window length
  - Overlap
- 5%, 50%, 95% exceedance levels:  
This option is useful for examining the variance between each window segment of the PSD.

The following MATLAB code is used to generate the PSD's:

```
1 %Chop waveform into segments
2 data2ch1 = buffer(waveformData,nfft,ceil(nfft*olap*1e-2),...
3 'nodelay');
4 %olap = integer between 0 and 100
5
6 %% Define window type and constants
7 switch windowS{1}
8     case 'Hann';
9         window = hann(windowL);
10        windowScalingFactor = 0.54;
11        noisePowerBandwidth = 1.36;
12     case 'Hamming';
13        window = hamming(windowL);
14        windowScalingFactor = 0.5;
15        noisePowerBandwidth = 1.5;
16 end
17
18 %% FFT
19 %Apply scaling factor
20 [~,n] = size(data2ch1);
21 data2ch1 = data2ch1.*repmat(window,1,n)/windowScalingFactor;
22 %fft
23 fft_data_ch1 = abs(fft(data2ch1))./nfft;
24 fft_amp_ch1=fft_data_ch1(1:nfft/2+1,:).^2;
25 % Correction for the noise power bandwidth
26 fft_amp_ch1=2*fft_amp_ch1/noisePowerBandwidth;
```



```

27 % Deriving the frequency units
28 f = (Fs/2*linspace(0,1,nfft/2+1))';
29 % Normalizing for 1 second
30 wintime = nfft/Fs; %time in seconds of window
31 acc1 = fft_amp_ch1*wintime;
32
33 %% Calculate results
34 % Mean
35 Xmean = mean(acc1,2);
36 % Percentiles
37 X5 = prctile(acc1,5,2);
38 Xmedian = prctile(acc1,50,2);
39 X95 = prctile(acc1,95,2);

```

For PSD's calculated across multiple channels, the resulting power values calculated from the above function for each channel per frequency are summed using vector addition:

$$PSD(f)_{xyz} = \sqrt{PSD(f)_x^2 + PSD(f)_y^2 + PSD(f)_z^2}$$

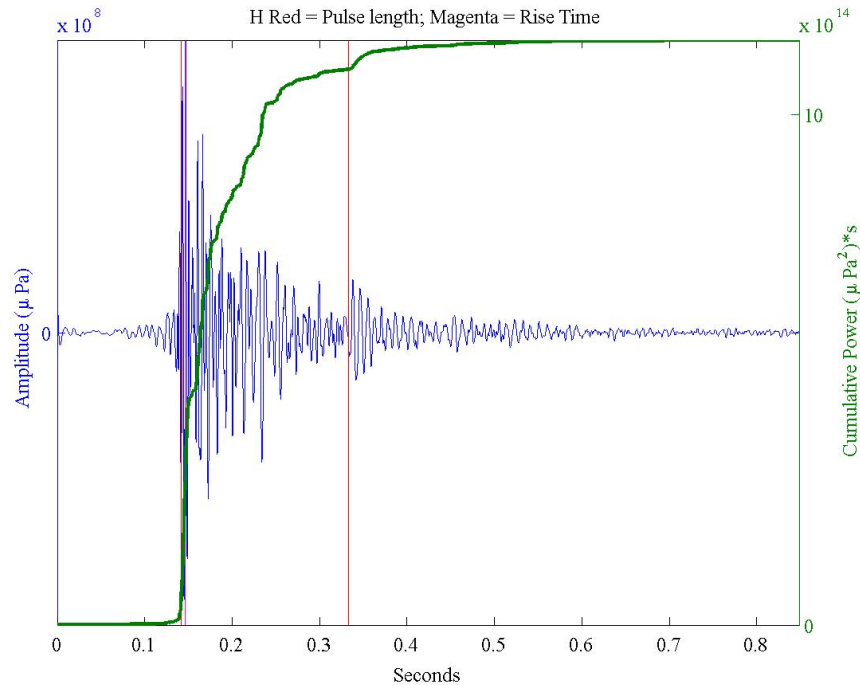
$f$  = Frequency

```

1 %PSD's are vectors containing the calculated
2 %power values for each frequency in F
3 PSDxyz = sqrt(PSDx.^2 + PSDy.^2 + PSDz.^2);
4 plot(F,PSDxyz) % plot resulting PSD;

```

## 3.2 Impulse



The impulse analysis gives the following outputs, each of which can be calculated for pressure, velocity, and acceleration:

- **Zero-to-peak**

$$\max|p(t)|$$

$t = \text{time}$

```

1 zeroToPeak = max(abs(p));
2 %where p is a vector containing all the calibrated
3 %pressure values over time (uPa)
4
5 %For multiple channels combined
6 zeroToPeakxyz = max(sqrt(x.^2 + y.^2 + z.^2));
7
8 % where x, y and z are vectors containing the calibrated
9 % particle motion waveforms for each channel
10 % (units are in either nm/s or um/s^2)

```

- **90% energy envelope**

This represents the time interval between the first occurrence of the 5<sup>th</sup> and 95<sup>th</sup> cumulative energy percentile in the analysis region, where the cumulative energy at any given point in time,  $t_n$ , is defined by:

$$\sum_{t=1}^n p(t)^2 dt$$

$p$  = pressure

```
1 dt = 1/sampleRate;
2 cumulativeEnergy = sum(p.^2)*dt;
```

For velocity and acceleration, pressure,  $p$ , is replaced by either  $u$  or  $a$ . When calculating the energy envelope across multiple particle motion channels, the following approach is used:

$$\sum_{t=1}^n \left( u(t)_x^2 + u(t)_y^2 + u(t)_z^2 \right) dt$$

```
1 cumulativeEnergy = sum(x.^2 + y.^2 + z.^2)*dt;
```

- **Single-strike exposure**

The single-strike exposure is defined as the sound exposure within the period encompassing the 90% energy envelope. Exposure is calculated in the same way as cumulative energy:

$$\sum_{t=1}^n p(t)^2 dt$$

- **Crest factor**

Crest factor is defined as the zero-to-peak sound pressure divided by the root-mean-squared sound pressure:

$$\frac{\max |p(t)|}{\sqrt{\frac{1}{n} \sum_{t=1}^n p(t)^2}}$$

```
1 crestFactor = max(p) / (rms(p));
```

For particle motion calculated over multiple channels, the crest factor is defined as:

$$\frac{\max \left( \sqrt{u(t)_x^2 + u(t)_y^2 + u(t)_z^2} \right)}{\sqrt{\left( \frac{1}{n} \sum_{t=1}^n u(t)_x^2 \right) + \left( \frac{1}{n} \sum_{t=1}^n u(t)_y^2 \right) + \left( \frac{1}{n} \sum_{t=1}^n u(t)_z^2 \right)}}$$

```
1 crestFactorXYZ = max(sqrt(x.^2 + y.^2 + z.^2)) / ...
2 sqrt(rms(x).^2 + rms(y).^2 + rms(z).^2);
```

### 3.3 Broadband

This analysis calculates the following metrics for pressure, velocity, and acceleration:

- **Root-mean-square pressure/particle motion**

This is the root-mean-squared sound pressure over the specified time range.

$$\sqrt{\frac{1}{n} \sum_{t=1}^n p(t)^2}$$

```
1 rms(p);
```

For measurements combined over multiple particle motion channels, the root-mean-square velocity or acceleration is calculated separately for each channel, then combined according to vector addition:

$$\sqrt{\left( \frac{1}{n} \sum_{t=1}^n u(t)_x^2 \right) + \left( \frac{1}{n} \sum_{t=1}^n u(t)_y^2 \right) + \left( \frac{1}{n} \sum_{t=1}^n u(t)_z^2 \right)}$$

```
1 sqrt(rms(x).^2 + rms(y).^2 + rms(z).^2);
```

- **Sound Exposure**

Sound exposure is calculated over the specified time range:

$$\sum_{t=1}^n p(t)^2 dt$$

```
1 dt = 1/sampleRate;
2 SoundExposure = sum(p.^2)*dt;
```

As with the root-mean-squared metric, the results for combined particle motion channels are summed together with vector addition:

$$\sqrt{\left(\sum_{t=1}^n u(t)_x^2 dt\right)^2 + \left(\sum_{t=1}^n u(t)_y^2 dt\right)^2 + \left(\sum_{t=1}^n u(t)_z^2 dt\right)^2}$$

```
1 SoundExposureXYZ = sqrt((sum(x.^2)*dt).^2 + ...
2 (sum(y.^2)*dt).^2 + (sum(z.^2)*dt).^2);
```

- **Consistency Analysis**

The consistency analysis calculates the percent of the time in which the amplitude of the signal exceeds a user defined threshold. The following code describes how consistency is calculated over a single channel:

```
1 % data = waveform data
2
3 %Count number of times the
4 % threshold was exceeded in waveform
5 countExceedThreshold = ...
6 sum(data < -1*(10^(dBthreshold/20))) ...
7 + sum(data > (10^(dBthreshold/20)))
8 % Convert to percentage
9 consistency = countExceedThreshold*100/...
10 length(data)
```

and for multiple channels:

```
1 % dataX = X channel waveform data
2 % dataY = Y channel waveform data
3 % dataZ = Z channel waveform data
4
5 data = sqrt(dataX^2 + dataY^2 + dataZ^2)
6
7 %Count number of times the threshold
8 % was exceeded in waveform
9 countExceedThreshold =...
10 sum(data < -1*(10^(dBthreshold/20))) ...
11 + sum(data > (10^(dBthreshold/20)))
12 % Convert to percentage
13 consistency = countExceedThreshold*100/...
14 length(data)
```

## Chapter 4

# Advanced features

### 4.1 Memory Issues

While paPAM saves all your calibrated data to the hard drive, MATLAB requires free RAM to execute the functions which are used in each analysis. On computers with low amounts of memory, an **Out of memory** error may occur when analyzing large segments of data. To workaroud and avoid these errors, you can try the following steps.

1. **Preserve Memory.** By selecting the **Preserve Memory** option from the **Execute** section on the main GUI, you can free up large portions of RAM during your analyses. Figures in MATLAB take up considerable amounts of memory, so having multiple figures open at once during the analysis can severely restrict the memory available. By checking the **Preserve Memory** option, paPAM will close each figure after it has been generated and saved to your hard disk, thus freeing up memory for your analysis. This can be especially useful when you want to save Spectrogram images during your analysis, as Spectrogram figures can easily take up over 100MB per figure.
2. **append PSDs.** This option is useful if you want to conduct a PSD analysis over a large region, but paPAM keeps on hitting an **Out of memory** error. As a workaround, you can divide your PSD analysis into several smaller PSD analyses. When you check the **append PSDs** option, paPAM will combine the window segments generated from each PSD analysis into a single PSD result where the percentiles can be calculated. For example:

- Suppose you want to construct a PSD which spans from 0-500 seconds in your `example.wav` file, but when you try to execute the analysis, you receive an `Out of memory` error.
- To work around this limitation, you check the option `append PSDs` on the main GUI, and select the time range 0-100 seconds for your first PSD analysis.
- After the analysis has completed, you will be asked `Would you like to add another analysis to this file?` Select `Yes`, and in the `Analysis Options - Multiple` section, add 4 more PSD analyses which span the ranges 100-200, 200-300, 300-400, and 400-500 seconds. Press `save` and allow the analysis to run to completion. The file `appendedPSD.csv` in your results folder will now contain the results of a PSD analysis which was conducted over the window segments of all 5 PSD analysis you did combined.

Here is some example output from the `append PSDs` feature:

Aa	n = 118	1-240000 240001-480000				Ba	n = 118	1-240000 240
Hz	Mean	5th	50th	95th	Hz	Mean	5th	
0	7.4743	-23.5638	-1.6904	10.8728	0	7.4743	-23.5638	
1	7.4942	-23.4171	-1.6812	10.8797	1	7.4942	-23.4171	
2	7.5538	-22.5298	-1.6539	10.9006	2	7.5538	-22.5298	
3	7.6529	-21.7515	-1.6095	10.9354	3	7.6529	-21.7515	
4	7.7909	-20.9538	-1.4417	10.9847	4	7.7909	-20.9538	
5	7.9674	-20.1331	-1.3863	11.0487	5	7.9674	-20.1331	

- The cell `A1` holds the name of the channel analyzed, where `[A B C D]` represent the `[X Y Z Hydrophone]` channels. Channel calculated for acceleration are represented with a preceding lowercase `a`, `[Xa Ya Za]`, while velocity channels have no preceding `a`, `[X Y Z]`.
- The cell `B1` denotes the number of window segments used in the calculation. This is useful as an error checking tool.
- The cell `C1` holds information about the sample ranges included in each analysis by showing the start and end samples of each PSD included in the analysis. A range of `1-240000 240001-480000` tells that 2 PSD analysis were used in the calculation, and each PSD analysis contained 240000 samples and there is no gap in between the PSD segments.



# Chapter 5

## paPAM Calculator

### 5.1 Intro and Setup

paPAM calculator is a tool you can use to predict the particle velocity and acceleration derived from measured pressure values. By comparing your predicted particle motion values to your measured particle motion values, paPAM calculator can be used as a rough proofing tool to check for major errors in your receiver calibration.

Setting up the paPAM Calculator requires the exact same steps found in section 1. For full versions of MATLAB, you'll want to run the file in the `paPAM.Calculator` folder:

- `PM_Analysis_Modeling.m`

To run paPAM using MCR, you'll use the following files:

- Mac
  - `paPAM_Calculator_mac.app`
  - `run_paPAM_Calculator_mac.sh`
- Windows
  - `paPAM_Calculator_win.exe`

### 5.2 Interface

Like paPAM, the paPAM calculator is built upon a central GUI:

paPAM Calculator 0.95

**Parameters**

Lowest frequency of interest:  Hz

Shallowest depth of interest:  m

Distance to sound source:  m

Source type:

Sediment type:

**Water**

Speed of sound in water:  m/s

Density of water:  kg/m<sup>3</sup>

**Sediment**

Speed of sound in sediment:  m/s

Density of sediment:  kg/m<sup>3</sup>

**Instructions**

You should measure the particle motion in this circumstance, but you can use the equation to the right to predict particle motion. To measure in the far field, you must be at least 15 m from the source. You can use the calculator below to double check your calculations.

**Equation**

$$u = \frac{p}{\rho \cdot c} \cdot \left( 1 + \frac{\lambda^2}{2\pi r} \right)^{1/2}$$


p = Measured pressure (Pa)  
 (rho) = Density of water (kg/m<sup>3</sup>)  
 c = Speed of sound in water (m/s)  
 u = particle velocity (m/s)  
 r = distance from source (m)  
 (lambda) = wavelength (m)

**Particle Motion Calculator**

Measured pressure:  (dB re 1 uPa)

The following values are calculated under the assumption that you are recording a single frequency pure tone (sin wave) of 100Hz with a strong signal-to-noise ratio.

Particle velocity (nm/s) = 65.9152  
 PVL = 36.3797 dB (ref 1nm/s)  
 Particle Acceleration (um/s<sup>2</sup>) = 41415.7401  
 PAL = 92.3433 dB (ref 1um/s<sup>2</sup>)



As you enter values into the text boxes, the calculator will automatically update and provide measurement recommendations based on your parameters. In the fields **Lowest frequency of interest** and **Shallowest depth of interest**, the lowest frequency is the lowest frequency you are interested in analyzing and the shallowest depth refers to the shallowest depth along the direct path between the sound source and the hydrophone.

After you have selected your parameters, you can enter in a *dB* value in the **Particle Motion Calculator** to estimate the particle motion values. Because some of the equations used to predict particle motion are dependent on frequency, all velocity and acceleration numbers presented are assuming that you are measuring a single pure tone sound which has a frequency defined by the **Lowest frequency of interest** you selected.

### 5.3 Equations

The paPAM calculator employs 3 commonly used acoustic equations to predict particle motion. These equations are listed below:

- Near-field particle velocity from pressure:

$$u = \frac{p}{\rho \cdot c} \cdot \left(1 + \frac{\lambda^2}{2\pi r}\right)^{1/2}$$

$u$  = particle velocity ( $m/s$ )

$p$  = acoustic pressure ( $m/s$ )

$\rho$  = density of water ( $kg/m^3$ )

$\lambda$  = wavelength of frequency of interest ( $m$ )

$r$  = distance from source ( $m$ )

$c$  = speed of sound in water ( $m/s$ )

- Far-field particle velocity from pressure:

$$u = \frac{p}{\rho \cdot c}$$

- Cut-off frequency (for open-water environments):

$$f_c = \frac{\pi - \rho_{sed}/\rho_{water}}{2 \cdot \pi \cdot \sin(\arccos(c/c_{sed}))} \cdot \frac{c}{H}$$

$\rho_{sed}$  = density of sediment ( $kg/m^3$ )

$\rho_{wat}$  = density of water ( $kg/m^3$ )

$c_{sed}$  = speed of sound in sediment ( $m/s$ )

$H$  = Depth of water ( $m$ )