Naval Surface Warfare Center
Carderock Division
West Bethesda, MD 20817-5700

NSWCCD-65-TR–2005/17 April 2006

Survivability, Structures, and Materials Department

Technical Report

High-Volume Data Analysis Suite (HVDAS)

by

Gregory Alan Hildstrom

Hildstrom Engineering
Subcontract/D.O. DDL 04-D-4032-010
DDL Omni Engineering
Prime Contract N00178-04-D-4032, D.O. FD02

NSWCCD-65-TR–2005/17 High-Volume Data Analysis Suite (HVDAS)

DEPARTMENT OF THE NAVY

NAVAL SURFACE WARFARE CENTER, CARDEROCK DIVISION
9500 MACARTHUR BOULEVARD
WEST BETHESDA MD 20817-5700

9110
Ser 65-83
13 Apr 06

From: Commander, Naval Surface Warfare Center, Carderock Division
To: Commander, Naval Sea Systems Command (SEA 05H, PMS 500)
Chief of Naval Research (ONR 334)

Subj: HIGH VOLUME DATA ANALYSIS SUITE (HVDAS)

Ref: (a) Program Element 0603564N, Ship Preliminary Design and Feasibility Studies
(b) Program Element 0604300N, DD(X) Total Ship Systems Engineering
(c) Program Element 0603581N, Littoral Combat Ship (LCS)

Encl: (1) NSWCCD-65-TR–2005/17, *High-Volume Data Analysis Suite (HVDAS)*

1. References (a), (b) and (c) provided direction and funding by which the Naval Surface Warfare Center, Carderock Division (NSWCCD) conducted several model and full-scale studies of hydrodynamic loads experienced by various types of surface ships. As a part of this work, NSWCCD developed a computer-based system to process, manage and analyze the data from these studies. HVDAS is a collection of data analysis software algorithms designed to run on a single Linux workstation or in parallel on Linux clusters. Enclosure (1) describes this system and provides procedures for its use.

3. Comments or questions may be referred to Mr. William H. Hay, Code 653; telephone (301) 227-1821; e-mail, William.H.Hay@navy.mil.

Will H Hay Jr

E. A. RASMUSSEN
By direction

Copy to:

CNR ARLINGTON VA [ONR 334]

NAVSURFWARCEN CARDEROCKDIV
BETHESDA MD [Codes 3442 (TIC), 65, 651, 652, 653, 653 (Hildstrom (5 copies)), 654, 655]

Naval Surface Warfare Center
Carderock Division
West Bethesda, MD 20817-5700

NSWCCD-65-TR–2005/17 April 2006

Survivability, Structures, and Materials Department

Technical Report

Gregory Alan Hildstrom

Hildstrom Engineering
Subcontract/D.O. DDL 04-D-4032-010
DDL Omni Engineering
Prime Contract N00178-04-D-4032, D.O. FD02

Enclosure (1)

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*
1-Apr-2006

2. REPORT TYPE
Final

3. DATES COVERED *(From - To)*
-

4. TITLE AND SUBTITLE
High-Volume Data Analysis Suite (HVDAS)

5a. CONTRACT NUMBER
N00178-04-D-4032, D.O. FD02

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
0603564N

6. AUTHOR(S)
Gregory Alan Hildstrom

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) AND ADDRESS(ES)

Naval Surface Warfare Center
Carderock Division
9500 Macarthur Boulevard
West Bethesda, MD 20817-5700

8. PERFORMING ORGANIZATION REPORT NUMBER

NSWCCD-65-TR–2005/17

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Attn SEA 05H and PMS 500
Commander
Naval Sea Systems Command
1333 Isaac Hull Ave. SE
Washington Navy Yard, DC 20376

Attn ONR 334
Chief of Naval Research
Ballston Centre Tower One
800 North Quincy Street
Arlington, VA 22217-5660

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
Hildstrom Engineering, Subcontract/D.O. DDL 04-D-4032-010
DDL Omni Engineering, Prime Contract N00178-04-D-4032, D.O. FD02

14. ABSTRACT

This document describes the design, implementation, performance, and use of the High-Volume Data Analysis Suite (HVDAS), which is a collection of parallelized statistical analysis software algorithms for Linux workstations and Linux clusters. HVDAS is designed to reduce and perform statistical analysis on any sampled raw time-history data. This document is targeted toward anyone who wants to know how HVDAS works, what HVDAS does, or how to use HVDAS. This document has been written for HVDAS version 2005-05, which was developed for the Naval Surface Warfare Center Carderock Division (NSWCCD), Code 653. Our main design goals are to increase the speed and quality of our data analysis. Speed improvements come from using custom compiled C++ code for time history analysis and the ability to run that code on one or more processors. Quality improvements come from batch processing, which eliminates most of the human errors from the data reduction process. Faster and higher quality data reduction lead to a more efficient overall data analysis procedure.

15. SUBJECT TERMS
data analysis, LINUX

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Mr. Gregory A. Hildstrom
a. REPORT UNCLASSIFIED	**b. ABSTRACT** UNCLASSIFIED	**c. THIS PAGE** UNCLASSIFIED	SAR	117	**19b. TELEPHONE NUMBER** *(include area code)* (301)-227-3967

Contents

List of Figures

List of Tables

2 Administrative Information

The High Volume Data Analysis Suite (HVDAS) has been used to speed data analyses of the DD-21, HSS, and DDX model tests as well as data analyses of the full scale sea trials of the Incat HSV-X1, RV *Triton*, Incat HSV2, and Fred Olsen. HVDAS is not a specific deliverable; it is a tool developed from project funds to increase the speed and quality of data analyses. Funding for this project has come from a variety of model test and full scale sea trials programs.

2.1 DD-21 Hull Form Hydrodynamic Loads Model Testing

This analysis was performed by the Structures and Composites Division (Code 65) of the Survivability, Structures and Materials Department at the Naval Surface Warfare Center, Carderock Division (NSWCCD). The work was sponsored by Naval Sea Systems Command, SEA 05H and the DD-21 program office, PMS 500, under Program Element 0603564N, Work Unit number 1-5500-114. Supplemental funding was provided by the Office of Naval Research under work unit number 1-5500-159. This analysis was funded in FY00 under Program Element 0604300N, Work Unit number 1-6530-456.

2.2 High Speed Sea Lift (HSS) Hydrodynamic Loads Model Testing

Hydrodynamic loads testing was completed on a segmented high-speed sea lift concept (HSS) Model 5594, scaled ratio 1:45, containing six shell sections connected using a calibrated back spline with instrumentation to measure primary and secondary hull girder loading. Similar to a trimaran, the extremely long slender center hull is stabilized by relatively short side hulls attached at an extreme aft location aligning all three hulls at the transom. This was part of a Navy program evaluating the structural design of a lightweight high-speed hull form as part of the high-speed sea lift program; Program Element 0603564N. Irregular wave experiments were completed to evaluate primary and secondary structural loads and seakeeping performance. Model testing was performed at the NSWC Harold Saunders Maneuvering and Seakeeping (MASK) facility in June 2002 and ended with irregular wave testing in July 2002 on carriage two of the David Taylor Model Basin.

2.3 DDX Model Testing

The DDX is a segmented tumble-home next-generation destroyer model, number 5637 at scale ratio of 1:30, containing five shell sections. It has been built and instrumented to measure both primary and secondary hull girder loadings. The shell sections are connected with a continuous aluminum beam, or back spline, instrumented with strain gages at each segment cut to measure vertical, lateral and torsional bending moments, as well as vertical and lateral shear forces. Secondary, local wave impact loads, are measured with pressure panels mounted on the bow, side shell, forward AGS, foredeck, transom and on the front faceted faces of the deckhouse. Pressure gages are also installed for measurements of quasi-static hull bottom pressures and transient wave impact pressures. The work was sponsored in FY04 by Northrop Grumman Ship Systems under Naval Sea Systems Command contract No. N00024-02-C-2302, under Work Unit number 03-1-5400-413. Program management was provided by Scott Black NSWCCD, Code 28, and Allen Engle NSWCCD, Code 651.

2.4 HSV-X1 Seakeeping and Structural Full Scale Trials

The USNS Joint Venture HSV-X1 is an aluminum catamaran built by Incat. The HSV is produced commercially as a high-speed ferry. Sea trials were carried out in 2001 and 2002 and involved trans-Atlantic crossings, loading and unloading operations, helicopter landing and take off, and high-speed combat support simulations. Funding was provided by two separate divisions of the Office of Naval Research. FY01 - 02: Expeditionary Logistics Future Naval Capabilities (FNC) High Speed Shuttle Product Line, Program Element 060XXXXN, Work Unit number 01-2820-040. FY02 - 03: Swampworks, Program Element 0603758N, Work Unit number 02-5500-195.

2.5 RV *Triton* Seakeeping and Structural Full Scale Trials

The RV *Triton* is a steel trimaran demonstrator research vessel built by Vosper Thornycroft in the UK. The sea trials were conducted in 2001 and 2002 and involved trans-Atlantic crossings, speed and powering maneuvers, seakeeping sea trials, and structural sea trials. Funding was provided by a memorandum of understanding (MOU) contract between the US and UK. The MOU contract number is N0002499MD11140. The data reduction and analysis began in June 2002 and continued through September 2003. As of 2003, this was the largest full-scale sea trials data analysis project ever attempted. The sea trials team recorded 50 star patterns, which consist of 10 different headings, in varying sea conditions. The largest peak wave amplitude experienced during the trials was 9.7 meters or 32 feet.

2.6 HSV2 Seakeeping and Structural Full Scale Trials

The HSV2 is a high-speed aluminum catamaran ferry which has been retrofitted for use as a Navy ship. It is similar in design to the HSV-X1, but smaller, lighter, and a newer design. The work was sponsored by NAVSEA 05D and performed pursuant to NSWCCD work for private parties agreement WPP-23-0376, dated 18 Nov 03. The NAVSEA Ship Design Manager was LCDR Quinten M. King. The Project Manager from NSWCCD was Carl Casamasina, Code 2300. Work was performed by code 65 from June 2004 October 2004 under Work Unit 04-1-2300-225.

2.7 Benchijigua Express Full Scale Trials

The ship Fred Olsen SA Benchijigua Express (AUSTAL Hull 260) is a trimaran which has been evaluated for the U.S. Navy's Littoral Combat Ship (LCS) Program. The work was sponsored by the NAVSEA PMS 501 under Program Element 0603581N, and performed by the Structures and Composites Division (Code 65); Survivability, Structures, and Materials Department (Code 60) of the Naval Surface Warfare Center, Carderock Division (NSWCCD). Funding was provided under NAVSEA Work Request N0002405WX11443.

3 Acknowledgments

I have received a great deal of help understanding the statistical methods upon which HVDAS is being built. I wish to thank the following individuals who have provided valuable insights and assistance to my understanding of general test data analysis procedures, associated data acquisition hardware and software, and the development of this code and documentation.

- Thomas Brady, Code 653, NSWCCD, data analysis;
- Robert H. Colegrove, Code 65, NSWCCD, editing;
- Daniel Gaies, Code 653, NSWCCD, data analysis, testing, HVDAS Data Extractor;
- J. Matthew Grassman, Code 651, NSWCCD, editing;
- William Hay, Code 653, NSWCCD, data analysis, editing;
- Paul Howden, Code 653, NSWCCD, editing;
- Richard Lewis, Code 653, NSWCCD, data analysis;
- Michael McDonald, Code 653, NSWCCD, coding, editing, algorithms, and testing; and
- Jesus Rosario, Code 653, NSWCCD, data acquisition.

4 Introduction

4.1 How To Use This Document

- Read this document in its entirety before performing any of the procedures.
- Check, check, and recheck modifications or deletions of data and configuration files.
- Back up your data before doing anything mentioned in this document.
- Run HVDAS.
- Back up your results before modifying or deleting data or configuration files.

4.2 Assumptions Made in This Document

The user has knowledge of and experience in:

- Sampled time-history data
- Time domain statistical analysis
- Frequency domain statistical analysis
- Linear transfer functions
- Discrete amplitude statistical analysis
- Weibull distribution statistics
- Binary and ASCII multiple-channel data files
- Format and location of the data files to be analyzed
- Finite impulse response filtering
- Linux operating system and a Unix shell
- Relational databases
- Structured query language (SQL)

The user's analysis computer and user account have:

- The Local Area Multicomputer (LAM) implementation of Message Passing Interface (MPI) installed
- The core or core plus version of the Vector Signal Image Processing Library (VSIPL) installed
- A relational database like Postgre SQL installed
- An active database accessible to the user
- UNIX ODBC libraries and drivers installed
- Appropriate permissions to run MPI programs
- Appropriate permissions to access the data files to be analyzed

4.3 What Does HVDAS Do?

HVDAS is a collection of data analysis software algorithms designed to run on a single Linux workstation or in parallel on Linux clusters. HVDAS consists of the hvdalibrary and hvdas executable. The hvdas executable has command line flags for MMM, PSD, RAO, and Weibull analysis. Each of these analysis algorithms is designed to run as an automated process to analyze sampled time-history data in bulk. HVDAS requires LAM-MPI, VSIPL, and UnixODBC to compile from source code. Information about obtaining and installing these software packages can be found at `http://www.lam-mpi.org`, `http://www.vsipl.org`, and `http://www.unixodbc.org`.

HVDALibrary contains most of the code that is common to the different analysis algorithms. HVDALibrary also contains most of the complex reusable analysis code used by each of the different analysis algorithms.

The module "hvdas -mmm" computes basic time domain statistics including mean, max, min, max-mean, min-mean, absolute range, maximum peak-to-peak amplitude, standard deviation, and kurtosis. MMM analysis can also convert data file formats and compute virtual channel data files.

The module "hvdas -psd" computes the power spectral density[1, 2] of selected data channels, using the Finite Fourier Transform Method Segment Averaging Method (PSDFFTMSAM[1]).

The module “hvdas -rao” computes the square root response amplitude operator, which is computed by $\sqrt{RAO} = \sqrt{\frac{PSDFFTMSAM(C_1)}{PSDFFTMSAM(C_2)}}$, which is simply the frequency based square root of the ratio of two power spectral densities. When analyzing surface ship structures, the RAO can calculate and provide ship response per unit of input wave height.

The module “hvdas -weibull” computationally estimates Weibull[16] distribution statistics on selected data channel amplitudes. You can choose the method of Weibull computation of either 2-parameter linear regression and 2-parameter moment method or 3-parameter linear regression and 3-parameter moment method. These statistics are useful for calculating maximum lifetime loads.

The module “hvdas -convertdata” converts, scales, filters, or computes virtual channels and saves the output as tab-delimited ASCII text.

HVDAS can apply scale factors, subtract initial condition zero values, smooth, finite impulse response[7] (FIR) filter, select data files, select channels, name channels, name conditions, select output locations, and read multiple data file formats. HVDAS can store output statistics in ASCII text files or in a relational database. The algorithms in HVDAS are not in any way restricted to ship structure data.

5 What Does “hvdas -mmm” Do?

The module “hvdas -mmm” computes mean maximum minimum (MMM) time domain statistics and is designed to reduce raw test data into a manageable spreadsheet of time domain statistical information. The user can conFigure the program to handle numerous filter types, data types, file formats, and test configurations. This software will compute the mean, max, min, $(maximum - mean)$, $(minimum - mean)$, range, maximum peak-to-peak, standard deviation, kurtosis, mean crossings, and $\frac{(maximum-mean)}{(standarddeviation)}$. These statistics are calculated for each of five possible filter states on selected channels, files, and conditions. The user has the option of subtracting an initial zero condition value from every channel of data before analysis.

The results are also summarized by filter state and condition. The summary contains weighted averages of statistics that do not require a discrete maximum or minimum value.

5.1 Mean

The sample mean[2] is calculated by $\overline{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$. The sample mean $\overline{x}$ is an unbiased estimator of population mean.

5.2 Max

The max is the maximum value that occurs in the time history.

5.3 Min

The min is the minimum value that occurs in the time history.

5.4 Maximum-Mean

The maximum-mean is computed by subtracting the mean of the time history from the maximum value in the time history.

5.5 Minimum-Mean

The minimum-mean is computed by subtracting the mean of the time history from the minimum value in the time history.

5.6 Range

Range refers to the absolute range of the time history data. If the maximum value occurs after the minimum value in the time history, the range is computed by subtracting the minimum value of the time history from the maximum value of the time history. If the minimum value occurs after the maximum value, the range is computed by subtracting the maximum value of the time history from the minimum value of the time history. This can be useful when analyzing non-sinusoidal data, where a positive range value could indicate generally increasing data over time, while a negative range could indicate generally decreasing data over time. For sinusoidal data, the sign of the range indicates whether the minimum or the maximum value occurred first in the time history, which may or may not be useful.

5.7 Maximum Peak-to-Peak

Maximum peak-to-peak is the amplitude of the largest half cycle oscillation in the time history about the mean of the time history.

5.8 Variance

The sample variance[2] is calculated by $\sigma^2 = Var(x_1 \ldots x_N) = \frac{1}{N-1}\sum_{j=1}^{N}(x_j - \bar{x})^2$. The summation is divided by $N-1$. That makes the sample variance an unbiased estimator for the population variance since the mean of the population is estimated from the sample itself.

5.9 Standard Deviation

The standard deviation[2] is calculated by $\sigma = \sqrt{\sigma^2}$. Taking the square root of the unbiased sample variance yields an unbiased estimator, the standard deviation. Figure 1 shows max-mean as a function of standard deviation for strain due to vertical bending on the RV *Triton*.

5.10 Kurtosis

The kurtosis[4, 5] is calculated by $Kurt(x_1 \ldots x_n) = \frac{1}{N}\sum_{j=1}^{N}\left[\frac{x_j - \bar{x}}{\sigma}\right]^4$. This method is also known as the Pearson kurtosis, which defines the kurtosis of a normal distribution as 3. Division by N is required because all two parameter distributions

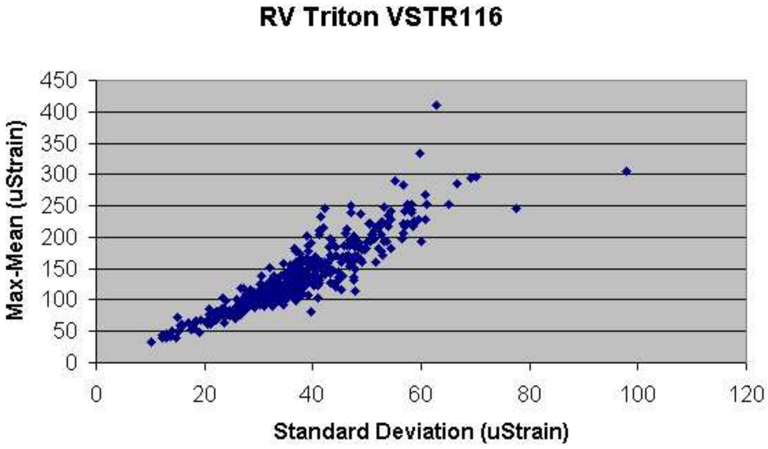


Figure 1: RV *Triton* MMM Standard Deviation Trend

are completely defined by the mean and the variance, which have already taken the degrees of freedom into account by dividing by N and $N-1$ respectively.

The excess, defined as $Excess(x_1 \ldots x_n) = \frac{1}{N}\sum_{j=1}^{N}\left[\frac{x_j-\bar{x}}{\sigma}\right]^4 - 3$, is not the kurtosis. The excess, or Fisher kurtosis, defines the kurtosis of a normal distribution as 0, which is not the intended calculation and is not calculated by HVDAS.

5.11 Mean Crossings

Mean crossings is the number of times the time history data oscillates about its mean.

5.12 (Maximum-Mean)/(Standard Deviation)

This value is computed by subtracting the mean from the maximum value in the time history and then dividing by the standard deviation of the time history. This value is a good check to quantify the severity of slamming, noise, or drop-outs in the data. This value defines how many standard deviations the maximum value is from the mean.

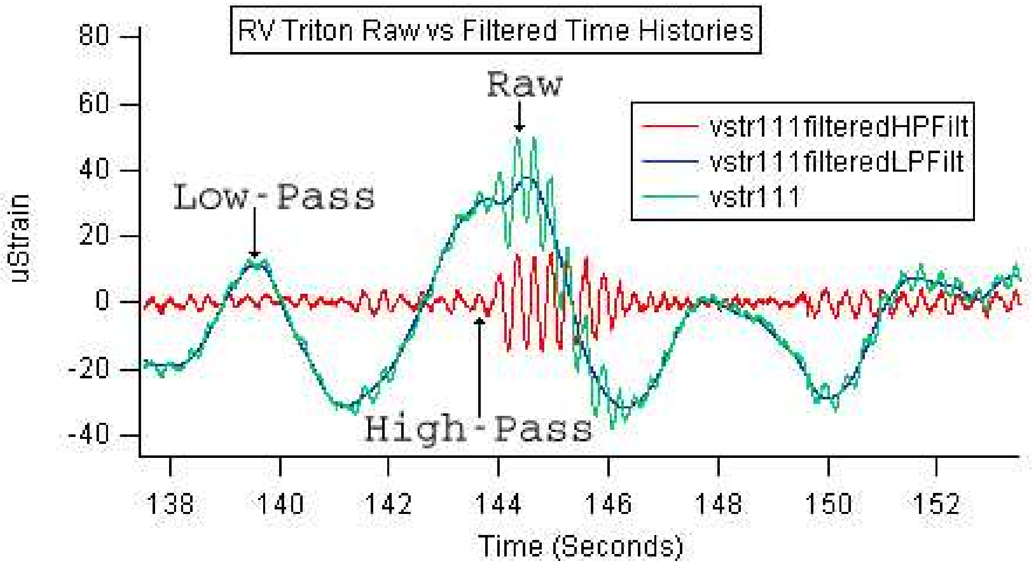


Figure 2: RV *Triton* Filtered Time History Comparison

5.13 Filtering, Smoothing, and Zero Subtraction

Figure 2 shows raw, low-pass Finite Impulse Response (FIR), and high-pass FIR filtered versions of a time history, which were taken from RV *Triton* sea trials. The filters were designed using Wave Metrics IGOR Pro IFDL and the time histories were filtered with HVDAS. HVDAS can calculate time domain statistics (MMM) on each of the different filtered or smoothed versions of each time history at the same time. If zero subtraction values are non-zero, those values will be subtracted from each time history before any analysis is performed.

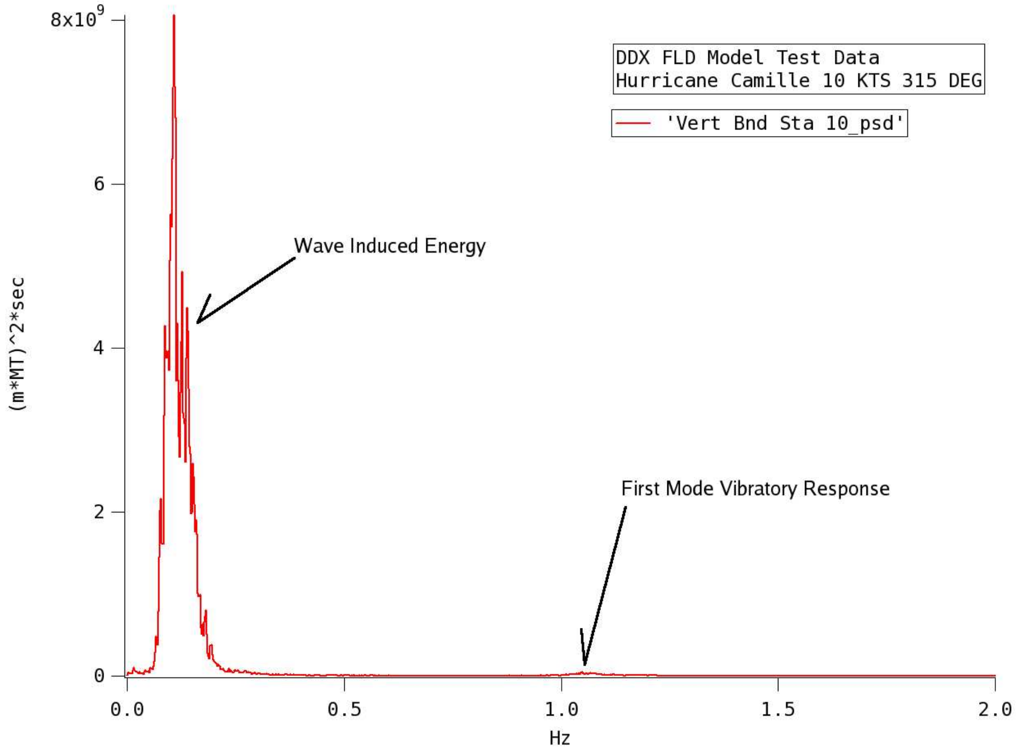


Figure 3: DDX FLD PSD 0 to 2 Hz

6 What Does "hvdas -psd" Do?

The module "hvdas -psd" computes power spectral density (PSD) frequency domain statistics and is designed to reduce raw test data into a manageable spreadsheet of frequency domain information. The user can conFigure the program to handle numerous filter types, data types, file formats, and test configurations. These statistics are calculated for either raw unfiltered or low-pass filtered states on selected channels, files, and conditions.

The results are also summarized by filter state and condition. The summary contains weighted averages of spectral ordinates.

Figure 3 shows an example PSD from the DDX FLD model test data. The PSD has been plotted from 0 to 2 Hz, which encompasses wave-induced energy and first mode vibratory response, but the first mode energy is so small it is not very visible on the initial y-axis scale. Figure 4 shows the same plot with a different

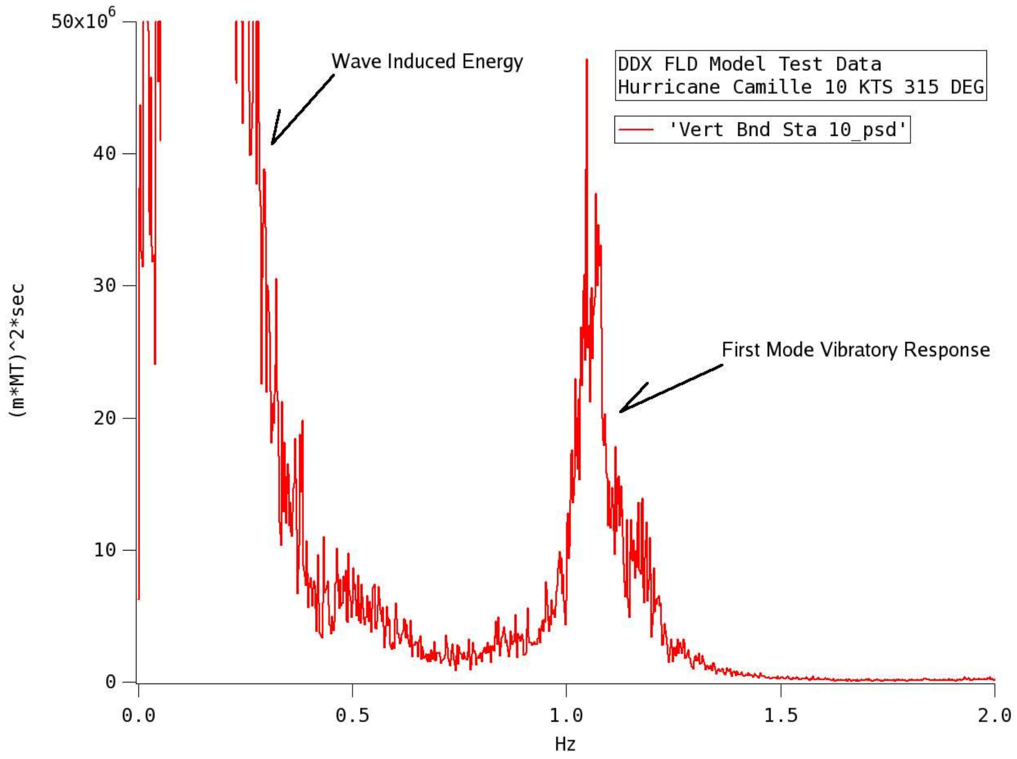


Figure 4: DDX FLD PSD 0 to 2 Hz Amplified

y-axis scale. The first mode energy is clearly visible.

Figure 5 shows the same PSD plotted from 0 to 0.5 Hz, which shows the wave-induced energy more clearly.

Figure 6 shows the same PSD plotted from 0.8 to 1.4 Hz, which shows the first mode vibratory response to the wave energy. The peak first mode frequency of 1.05 Hz is clearly visible; this is the natural frequency of the DDX FLD Model in the vertical longitudinal plane.

6.1 Power Spectral Density Finite Fourier Transform Method Segment Averaging Method

This software computes the power spectral density[1, 6, 2], which has user selectable resolution and frequency content.

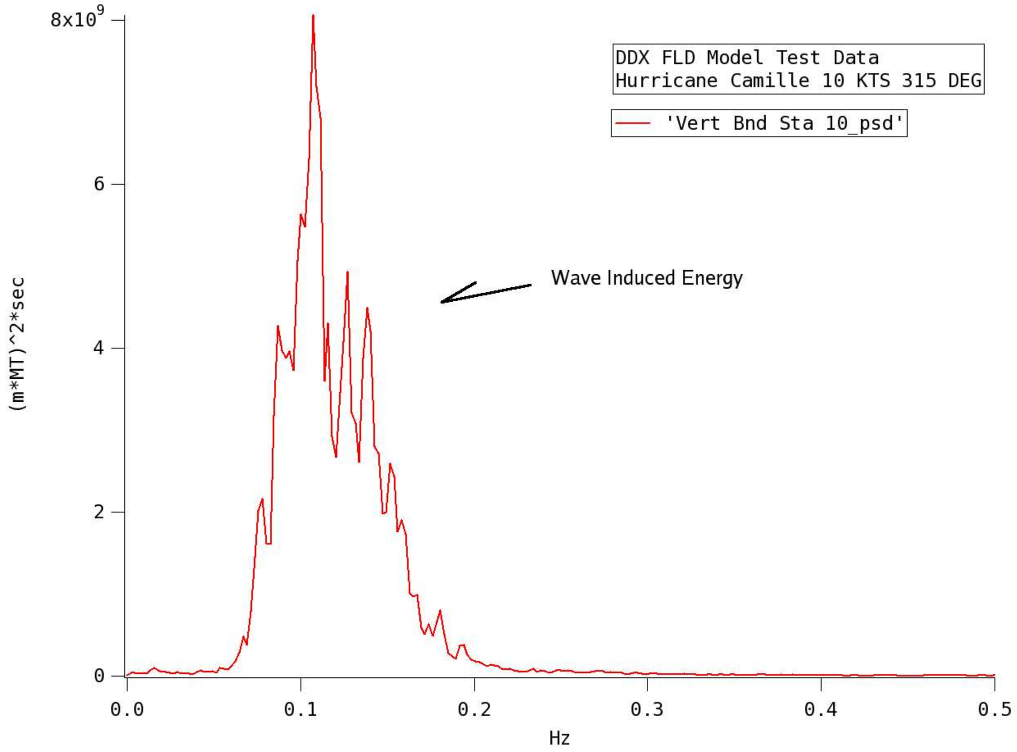


Figure 5: DDX FLD PSD 0 to 0.5 Hz

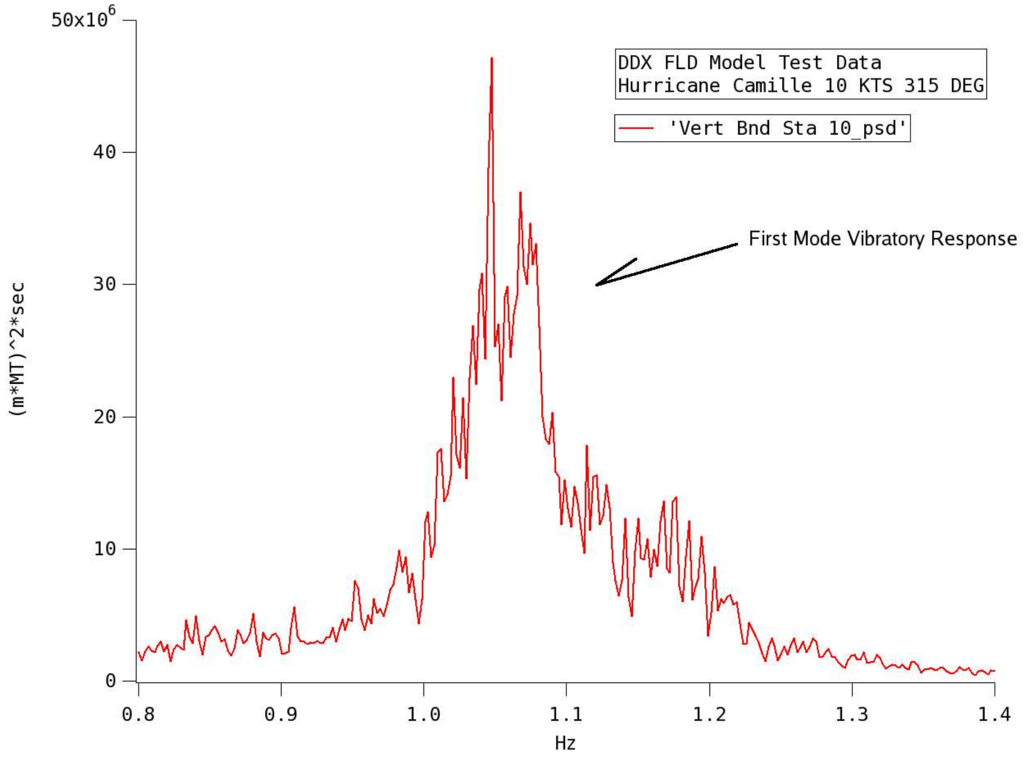


Figure 6: DDX FLD PSD 0.8 to 1.4 Hz

The power spectra in this analysis use the power spectral density finite Fourier transform segment averaging method (PSDFFTMSAM[1]) with a 50% overlap.

The PSDFFTMSAM computes the power spectral density using the finite Fourier transform method (PSDFFTM) on a number of segments of the original data channel, which have the mean of each segment subtracted. The local segment mean subtraction prevents extremely low frequencies or drift from aliasing to DC in the output spectrum. The output PSD will be $\frac{SegmentLength}{2}+1$ points long. Each segment PSD is summed and divided by the total number of segments in the input data channel, which includes the overlap. This summation and division results in an average PSD for the data run.

The computed PSDFFTMSAM uses I-Skipping (ISkip) to achieve greater resolution. The amount of ISkip and the PSD segment length are computed automatically based on the original sample rate (OSR) of the data, highest frequency of interest (HFI) of the output PSD, amount of oversampling (OS) of the HFI, and the target delta frequency (TDF) of the output PSD, which are all selectable by the user. ISkip is defined as keeping 1 out of ISkip points. For example, an ISkip of 8 keeps 1 data point and skips 7 data points. An ISkip of 1 keeps every point. The $ISkip = \frac{OSR}{HFI*OS}$. The new sample rate (NSR) of the ISkip-ed time history is computed by $NSR = \frac{OSR}{ISkip}$. The $SegmentLength = \frac{NSR}{TDF}$. The ISkip process starts in the time history at each possible ISkip start point, from 0 to ISkip-1, and concatenates the different I-Skipped time history versions together to retain as much data as possible; this is illustrated in Figure 7.

If any single data channel contains too few data points to compute the PSD with the desired resolution and frequency content, that channel will be repeated and concatenated with itself until enough points exist to perform the PSD computations. This method assumes each short data run would continue with identical frequency content until enough data exist to compute the desired PSD. Any discontinuities exist as frequencies above the Nyquist frequency, which is half of the sample rate, and will not show up in the output PSD.

The Finite Fourier Transform Method (FFTM) of computing the PSD assumes that $h(t) = RealInputDataVector$ and that $H(f) = FourierTransform(h(t))$. The PSDFFTM is computed by $PSDFFTM(f) = 2* \mid H(f) \mid^2$. The Fourier Transform (FT) is performed on the original data vector segment. The real and imaginary parts of the FT are used to create a real magnitude squared FT. The result is multiplied by two to account for the one-sided spectrum, then divided by the input data vector size and the sample rate to scale the power spectrum. This last step needs to be performed because

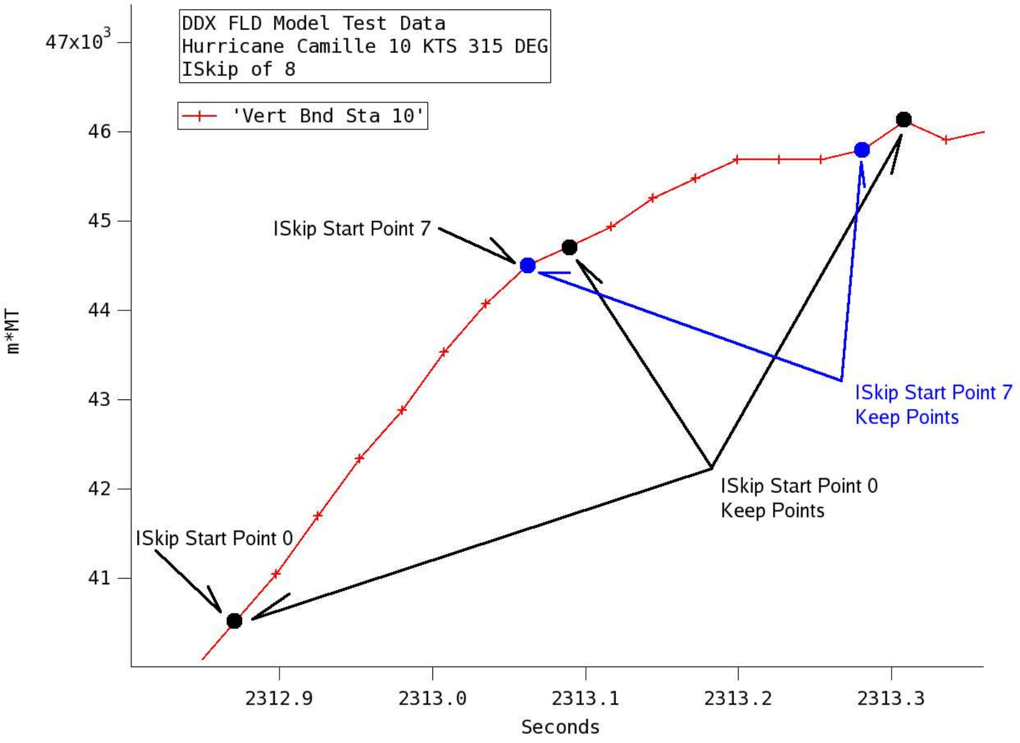


Figure 7: DDX FLD ISkip Example

of the FT implementation used. The area under the final PSD is the variance of the original time history. The corresponding frequency of each PSD magnitude squared is calculated using the number of PSD points, the number of the current point, and the Nyquist frequency, which is half of the new sample rate.

The smoothness of the PSD is inversely proportional to the resolution of the PSD when the number of encounters in the area of interest is too small. This is a side effect of SegmentLength, which ends up being the number of points used for the FFT. For high resolution smooth PSDs, 200 or 300 encounters may be needed to produce a nice averaging effect in the area of interest. This is easier to achieve in higher frequencies than lower frequencies because of recording time and encounters per second. High resolution and averaging of 20-second waves would require 100 minutes of data in a given condition for 300 encounters.

7 What Does “hvdas -rao” Do?

The module “hvdas -rao” computes square root response amplitude operator (RAO) and is designed to reduce raw test data into a manageable spreadsheet of frequency domain information. The user can conFigure the program to handle numerous filter types, data types, file formats, and test configurations. RAOs are calculated for either raw or low-pass filter states on selected channels, files, and conditions.

The results are also summarized by filter state and condition. The summary contains weighted averages of RAO ordinates based on individual run length.

Figure 8 shows a vertical bending strain RAO from the RV *Triton* sea trials in the Sea State 4, 10-knots, head seas condition. You can see that, at 0.3 Hz, the strain, due to vertical longitudinal bending, of the RV *Triton* has the largest response to input wave height. The max valid RAO encounter frequency for the RV *Triton* is probably 0.7 to 0.8 Hz, based on full speed head seas linear wave-induced energy. The true maximum linear response to wave energy occurs close to 0.3 Hz encounter. After 0.8 Hz the response is nonlinear vibration and the RAO transfer function is not useful. Because of this behavior, it is necessary to also look at structural and wave height PSDs separately to determine valid RAO frequency ranges. The RAO is intended to characterize structural response over the frequency range containing significant wave energy only.

7.1 Response Amplitude Operation Computation

This software will compute the square root RAO using two PSDs, which have user-selectable resolution and frequency content.

The power spectra computed in this analysis utilize the PSDFFTMSAM with a 50% overlap, as described in the PSD analysis section.

An RAO is simply the frequency-by-frequency ratio of a spectrum to an excitation spectrum, $\left(\sqrt{RAO}\right)^2 = \frac{PSDFFTMSAM(DataChannel)}{PSDFFTMSAM(DivisorChannel)}$, where the RAO units depend on the units of the input data channels. In ship structures, the $\left(\sqrt{RAO}\right)^2 = \frac{PSDFFTMSAM(ResponseChannel)}{PSDFFTMSAM(WaveHeight)}$, which yields results in terms of ship response per unit of wave height, but the output transfer function depends on the units of your data channels. Users can use the optional conversion from encounter to wave frequency, which is based on the current condition’s speed and heading, to convert RAOs from encounter (observed) frequency to wave frequency.

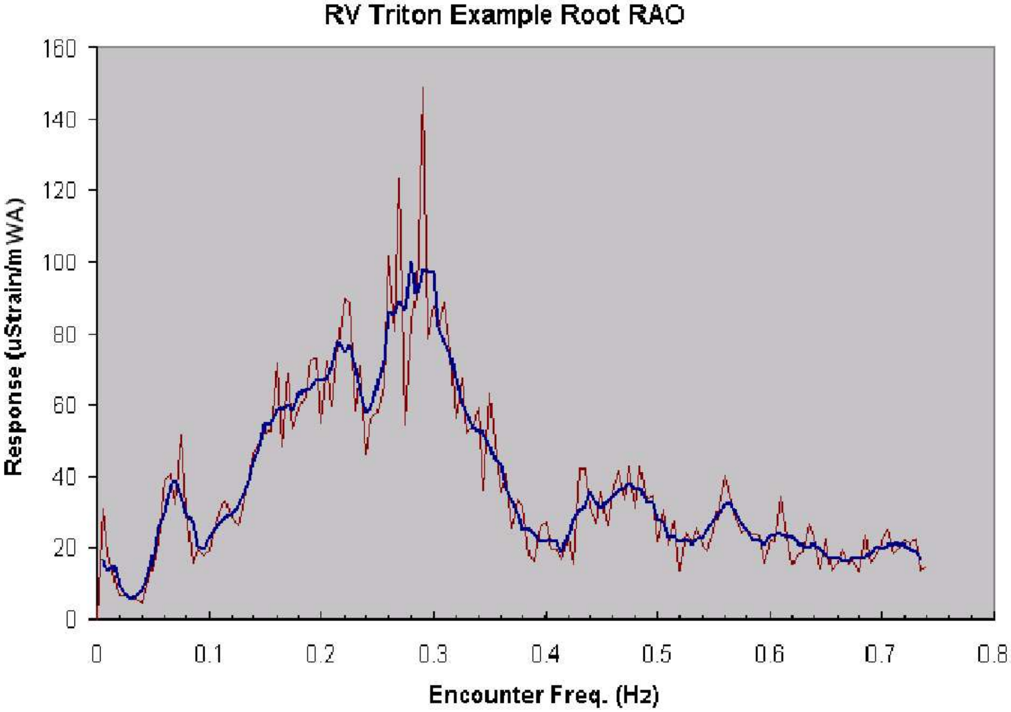


Figure 8: RV *Triton* RAO Example 0 to 0.8 Hz

7.2 Encounter to Wave Frequency Conversion

Encounter frequency spectra show frequencies as they were observed or encountered on the ship. Converting to wave frequency adjusts spectra encounter frequencies to give the frequencies as would be seen by a stationary wave buoy, which essentially removes speed and heading effects of the ship movement.

The optional conversion from encounter to wave frequency[17, 18] is computed using $w_{wave} = \frac{-1+\sqrt{1+4\left(\frac{Vcos(\Theta)}{g}\right)w_e}}{2\left(\frac{Vcos(\Theta)}{g}\right)}$, where $g = 32.18$ ft/sec/sec, $V =$ speed in ft/sec ($6076ft = 1NauticalMile$), $\Theta =$heading in degrees ($HeadSeas = 0$ degrees), $w_e = f_e 2\pi$, and $f_w = \frac{w}{2\pi}$. Head seas means that the ship is heading straight into the waves. This equation works for all runs with headings from 0 to < 90 degrees and from > 270 degrees to 360 degrees. If the heading is beam seas, 90 or 270 degrees, the wave frequency is equal to the encounter frequency. The conversion in following and stern quartering seas, > 90 degrees to < 270 degrees, is not implemented because the solution can be discontinuous and indeterminate.

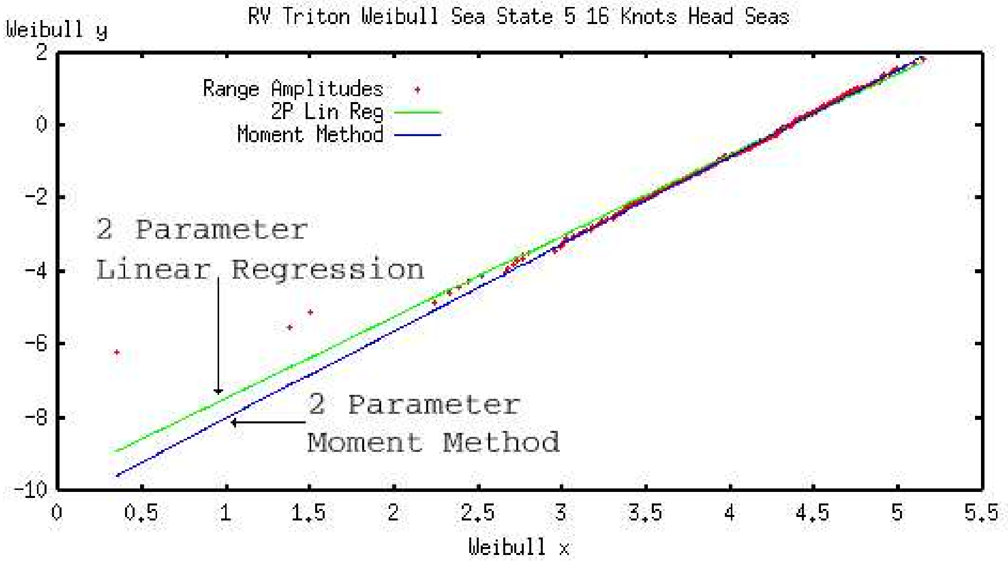


Figure 9: RV *Triton* Range Amplitudes Weibull

8 What Does “hvdas -weibull” Do?

The module “hvdas -weibull” computes Weibull[6] distribution statistics on a set of data amplitudes and is designed to reduce raw test data into a manageable spreadsheet of linearized Weibull distribution parameters. These statistics are calculated for either raw, low-pass, or high-pass filter states on selected channels, files, and conditions. The results are computed on the entire set of amplitudes that exist in a single test condition, which yields one output file per condition for each variant of analysis. Weibull analysis linearizes the data amplitudes[6], in Weibull space, and computes trend lines for lifetime estimation[6].

Figure 9 shows linearized Weibull amplitudes and two corresponding solution methods: 2-parameter linear regression and 2-parameter moment method.

8.1 Solution Methods

8.1.1 Weibull Linearizing

The selected data amplitudes can be linearized into a $y = m * x + b$ format[6]. The Weibull Y-axis data can be represented by $\ln\left(\ln\left(\frac{1}{1-P(x)}\right)\right) = \beta \ln\left(x - x_0\right) -$

$\beta\ln(\eta - x_0)$, where $P(x) = 1 - e^{-\left(\frac{x}{\eta}\right)^{\beta}}$. For 2-parameter distributions $x_0 = 0$. A linearized Weibull plot can be achieved by plotting the Weibull Y-axis data against $\ln(x)$ on the X-axis. If data amplitudes are Weibull in nature, this will yield a straight line in slope-intercept form where β is the slope and $(-\beta\ln(\eta))$ is the Y-intercept.

8.1.2 2-Parameter Linear Regression Method

1. Find data amplitudes.
2. Rank amplitudes in ascending order.
3. Linearize the ordered amplitudes in Weibull space.
4. Perform a linear regression.
5. Determine the correlation coefficient.

8.1.3 2-Parameter Moment Method Iterative Calculation

The moment method[6] attempts to determine the line that best represents the Weibull data based completely on the mean and variance of the amplitudes. The shape, or slope, of the Weibull distribution can be completely defined by the mean and variance of the amplitudes; as shown here $\frac{\sigma_x^2}{\mu_x^2} = \frac{\Gamma\left(1+\frac{2}{\beta}\right)}{\left(\Gamma\left(1+\frac{1}{\beta}\right)\right)^2} - 1$. Iterating through this equation will yield β, the moment method shape parameter, which can be used along with the characteristic value from the population to determine a moment method line, which should better represent the higher magnitude Weibull amplitudes. The characteristic value can be determined by $\mu_x = \eta\left(\Gamma\left(1+\frac{1}{\beta}\right)\right)$. Provided that the data are known to be Weibull, the moment method is a bias-free method of estimating the Weibull parameters.

8.1.4 2-Parameter Moment Method

1. Find data amplitudes.
2. Calculate the sample mean of the amplitudes (μ_x).
3. Calculate the sample variance of the amplitudes (σ_x^2).

4. Iteratively solve for beta, the shape parameter.
5. Calculate the moment method parameters.

8.1.5 3-Parameter Linear Regression Method

1. Find data amplitudes.
2. Iterate x_0 from 0 to the smallest amplitude to find the best x_0.
3. For each x_0 subtract x_0 from the amplitudes, perform a 2-parameter linear regression, and determine the highest correlation coefficient.
4. Choose the x_0 that produced the highest correlation.
5. Subtract x_0 from the amplitudes.
6. Compute a 2-parameter linear regression.
7. Add x_0 to the characteristic value and report the parameters.

8.1.6 3-Parameter Moment Method Iterative Calculation

The moment method[6, 19] attempts to determine the line that best represents the Weibull data based completely on the mean, variance, and estimated characteristic value of the amplitudes. The general three-parameter Weibull distribution is $P = 1 - e^{-(\frac{x-x_0}{\eta-x_0})^{\beta}}$, where β is the Weibull slope or shape parameter, x_0 is the threshold value of the data, η is the characteristic value of the distribution of data, P is the cumulative probability, and x is the Weibull distributed data.

The mean and variance of Weibull data can be expressed as: $\mu_x = (\eta_x - x_0)\Gamma(1+\frac{1}{\beta})+x_0$ and $\sigma_x^2 = (\eta_x - x_0)^2\left[\Gamma(1+\frac{2}{\beta}) - (\Gamma(1+\frac{1}{\beta}))^2\right]$. The characteristic value can be estimated by choosing the value in the distribution with a cumulative probability as close as possible to 0.632. The threshold x_0 can be represented by two equations: $x_0^1 = \eta_x - \frac{\sigma_x}{\sqrt{\Gamma(1+\frac{2}{\beta}) - \left[\Gamma(1+\frac{1}{\beta})\right]^2}}$ and $x_0^2 = \mu_x - \frac{\sigma_x \Gamma(1+\frac{1}{\beta})}{\sqrt{\Gamma(1+\frac{2}{\beta}) - \left[\Gamma(1+\frac{1}{\beta})\right]^2}}$. Solving for β and x_0, by iterating on a range of β and minimizing the difference between these two equations, will converge on a single (β, x_0) pair.

8.1.7 3-Parameter Moment Method[19]

1. Find data amplitudes.
2. Calculate the mean of the amplitudes (μ_x).
3. Calculate the variance of the amplitudes (σ_x^2).
4. Estimate the characteristic value from the amplitudes (η_x).
5. Iteratively solve for the best (β, x_0) pair using the estimated characteristic value from the amplitudes and the two x_0 equations.
6. If (0 < x_0 < Smallest Amplitude*0.99) then the (β, x_0) pair is valid; calculate the moment method line, and the calculation is finished.
7. Else, if (x_0<0), set $x_0$to 0.
8. Else, if (x_0>Smallest Amplitude*0.99), set x_0 to Smallest Amplitude*0.99.
9. Iteratively solve for β using $x_0^2 = \mu_x - \frac{\sigma_x \Gamma(1+\frac{1}{\beta})}{\sqrt{\Gamma(1+\frac{2}{\beta}) - \left[\Gamma(1+\frac{1}{\beta})\right]^2}}$.
10. Solve for a new η_x using $x_0^1 = \eta_x - \frac{\sigma_x}{\sqrt{\Gamma(1+\frac{2}{\beta}) - \left[\Gamma(1+\frac{1}{\beta})\right]^2}}$.
11. Calculate the moment method parameters.

8.2 Amplitude Selection

The amplitude type is selected by the AMPLITUDE_TYPE variable in the Weibull configuration file, which is described in a later section.

8.2.1 Maximum Amplitudes

This method of amplitude selection uses peaks which occur above the mean of the time history in the Weibull calculation. This yields one amplitude per wave encounter. The AMPLITUDE_TYPE for this is 0.

8.2.2 Minimum Amplitudes

This method of amplitude selection uses troughs which occur below the mean of the time history in the Weibull calculation. This yields one amplitude per wave encounter. The AMPLITUDE_TYPE for this is 1.

8.2.3 Maximum - Mean Amplitudes

This method of amplitude selection uses all peaks - mean of the time history in the Weibull calculation. These amplitudes are determined by subtracting the mean of the time history from the maximum amplitudes. This yields one amplitude per wave encounter. The AMPLITUDE_TYPE for this is 2.

8.2.4 Minimum - Mean Amplitudes

This method of amplitude selection uses all mean - troughs of the time history in the Weibull calculation. These amplitudes are determined by subtracting the mean of the time history from the minimum amplitudes. This yields one amplitude per wave encounter. The AMPLITUDE_TYPE for this is 3.

8.2.5 Peak-to-Peak Range Amplitudes

This method of amplitude selection uses all local half-cycle peak-to-peak ranges (maximums - minimums) about the mean of the time history in the Weibull calculation. These amplitudes are determined by looking for oscillations about the mean value of the time history. The peak-to-peak amplitudes have double the number of amplitudes as maximums or minimums because every half-cycle amplitude is counted, which means all peak-to-trough amplitudes and trough-to-peak amplitudes. This ensures that the Weibull distribution data contains the largest half-cycle event in the time history. One must be careful when using these amplitudes however. The number of peak-to-peak amplitudes is twice the number of wave encounters. The AMPLITUDE_TYPE for this is 4.

8.2.6 Raw Amplitudes

This method of amplitude selection treats the entire input data channel as a set of amplitudes. This means that you can use another program to generate or adjust the amplitudes and then analyze them in HVDAS. This allows another program, such as IGOR or Excel, to perform the amplitude selection. These amplitudes can then be run through HVDAS to have Weibull analysis performed on them. The AMPLITUDE_TYPE for this is 5.

8.2.7 Whipping Amplitudes

This method of amplitude selection looks for a series of consecutive high-pass filtered amplitudes that exceed a threshold. The AMPLITUDE_TYPE for this is 6-8.

8.2.8 Development Amplitudes

This method of amplitude selection is a placeholder in the HVDAS software for amplitude selection development. The AMPLITUDE_TYPE for this is 9-11.

8.2.9 Max/Min Ratio Amplitudes

This method of amplitude selection uses all peaks - mean and troughs - mean of the time history in the Weibull calculation. These amplitudes are determined by subtracting the mean of the time history from the maximum and minimum amplitudes and dividing $\frac{maximum-mean}{minimum-mean}$. This yields one amplitude per wave encounter. The AMPLITUDE_TYPE for this is 12.

8.2.10 LGH Impact Amplitudes

This method of amplitude selection looks for an event based on standard deviation exceedance in a very short time period. Impacts are usually very short steep impulse-like events. The user can select three slope standard deviation multipliers for each channel in the Global Configuration file. If that value is exceeded consecutively from one scan to the next, it is an event and a 1-second window is searched for the amplitude and local mean. The detected amplitudes are reported as max - local mean to account for small drift behavior. The local mean comes from a 50ms window 50ms before the impact. An optional increasing requirement can be specified in addition to the standard deviation multiplier exceedance. The AMPLITUDE_TYPE for this is 16. Figure 10 shows a run from the DDX FLD model test. Figure 11 shows an impact event closer and illustrates the source of the preceding local mean calculation. Figure 12 shows a close-up of the event and the scans that satisfy the standard deviation requirements.

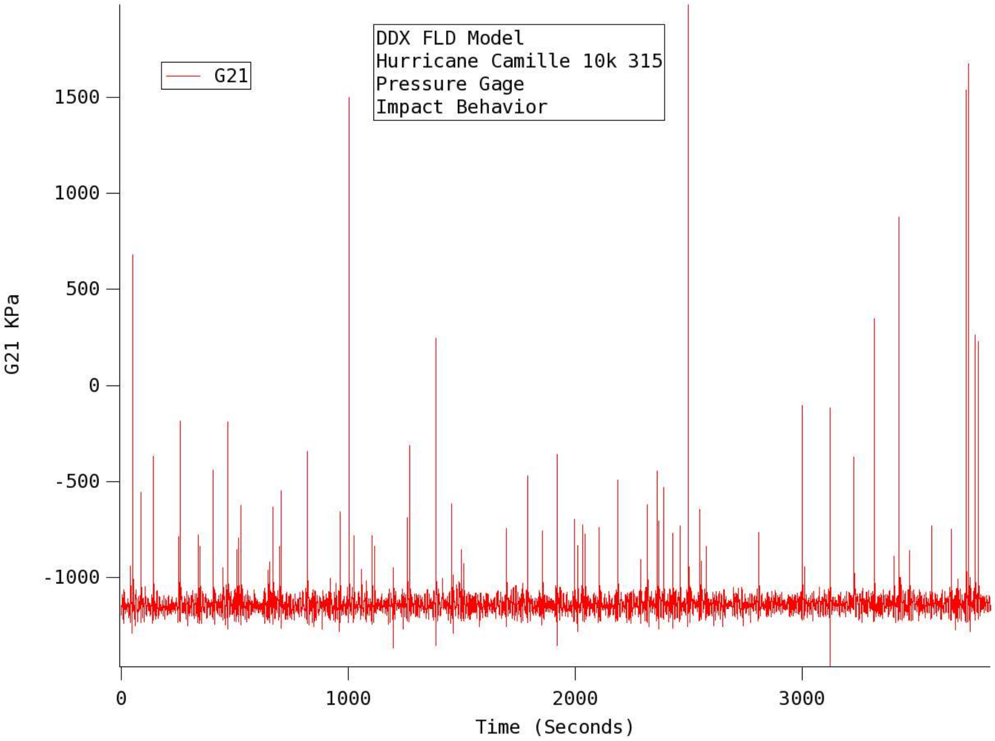


Figure 10: Pressure Gage Impact Example 1

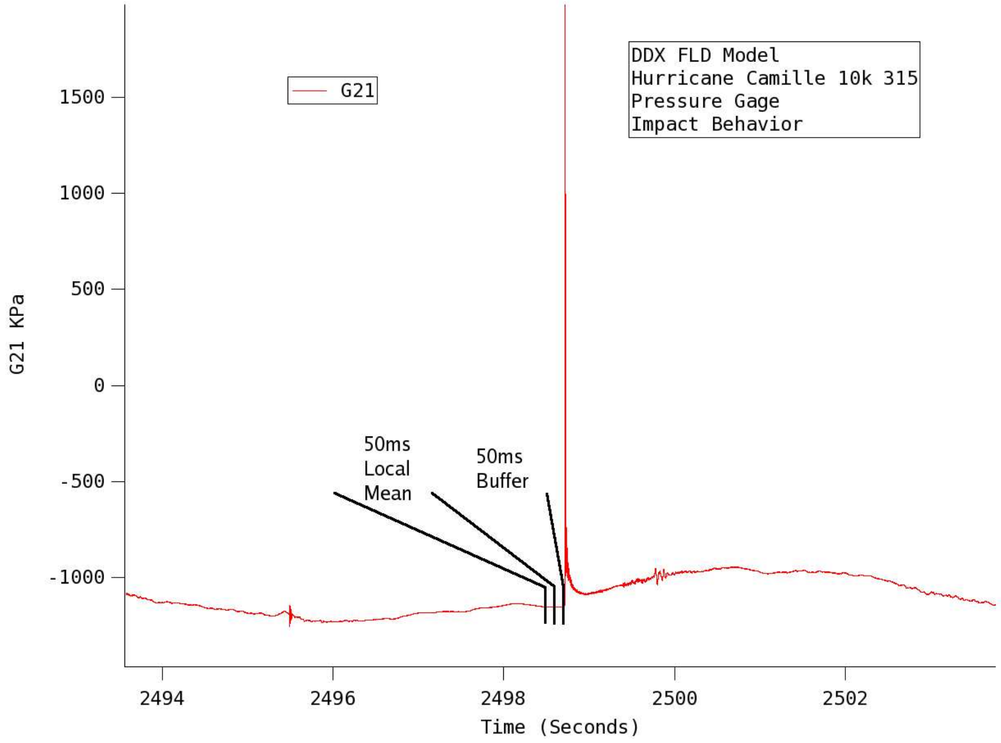


Figure 11: Pressure Gage Impact Example 2

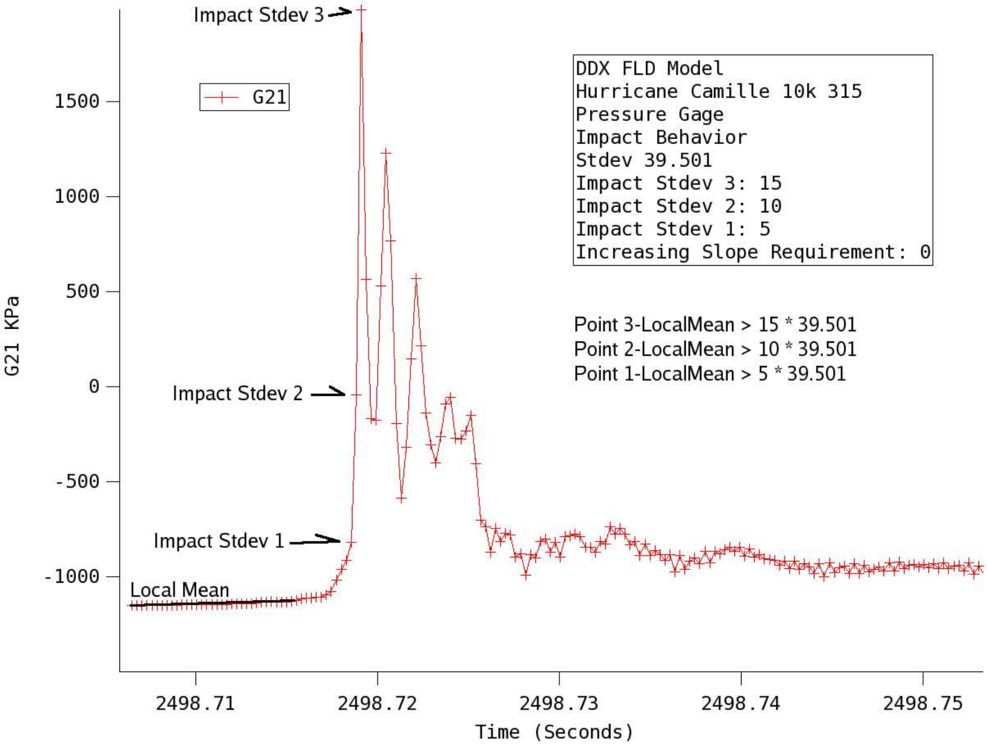


Figure 12: Pressure Gage Impact Example 3

8.2.11 Combined Whipping Amplitudes

This method of amplitude selection looks for football-shaped series of consecutive high-pass filtered amplitudes. The reported amplitudes occur in the raw combined time history during the time window of the high pass event. The AMPLITUDE_TYPE for this is 13-15.

8.3 Filter Choices

This software has the ability to separate frequency components of a time domain signal for further Weibull analysis.

8.3.1 Low Pass

The low-pass finite impulse response (FIR) filter is usually used to remove high-frequency energy from the time domain signal. It can be used to yield the quasi-static ordinary wave-induced portion of the signal by removing higher frequency vibration, slamming, and whipping cycles. The amplitude selection steps are as follows.

1. Read in the raw data.
2. Low-pass FIR filter the raw data.
3. Find all mean crossings and relative maximums and minimums in the filtered time history.
4. Use the local maximums and minimums to generate local ranges, also called peak-to-peak values.
5. Select the proper amplitude set and perform the Weibull analysis.

8.3.2 High Pass

The high-pass FIR filter will remove low-frequency energy from the signal. It can be used to yield the vibration, slamming, and whipping portion of the signal by removing low-frequency ordinary wave-induced energy. The amplitude selection steps are as follows.

1. Read in the raw data.

2. Low-pass FIR filter the raw data.
3. Find all mean crossings in the low-pass FIR data to identify wave events experienced by the ship.
4. High-pass FIR filter the raw data.
5. Find relative maximums and minimums in the high-pass filtered time history using the mean crossing locations.
6. Use the local maximums and minimums to generate local ranges, also called peak-to-peak values.
7. Select the proper amplitude set and perform the Weibull analysis.

8.3.3 Raw

Raw is the absence of a filter and will not remove any energy from the signal. We usually use it to yield the wave+whipping portion of the signal. The amplitude selection steps are as follows.

1. Read in the raw data.
2. Low-pass FIR filter the raw data to identify wave encounters.
3. Find all mean crossings in the low-pass FIR data to identify wave events experienced by the ship.
4. Find relative maximums and minimums in the raw time history using the mean crossing locations.
5. Use the local maximums and minimums to generate local ranges, also called peak-to-peak values.
6. Select the proper amplitude set and perform the Weibull analysis.

8.3.4 Already Filtered

The already filtered option will allow a simple result in the Weibull calculation. The data is assumed to be already filtered. This removes the digital FIR filtering step from the amplitude selection process.

1. Read in the raw already filtered data.
2. Find all mean crossings and relative maximums and minimums in the already filtered time history.
3. Use the local maximums and minimums to generate local ranges, also called peak-to-peak values.
4. Select the proper amplitude set and perform the Weibull analysis.

8.4 Prediction

The main goal of Weibull analysis is to statistically characterize the data and to make predictions. Finite amounts of test data, that are Weibull in nature, can be analyzed with this software and yield response predictions based on estimates of many years or even a ship's entire lifetime.

There are two main types of predictions that we use Weibull analysis for: fatigue life and lifetime maximum value. For fatigue life prediction, a two- or three-parameter linear regression is usually sufficient, because it is the line that best matches the majority of amplitudes that occurred during the test. The two or three parameter moment method line usually does a better job of estimating maximum lifetime values.

For ship structures and wave data, wave induced behavior usually has a linear regression Weibull slope of around 2, corresponding to a Rayleigh distribution. Slamming and vibration behavior usually has a linear regression Weibull slope of around 1. Normal distribution Gaussian behavior, when treated as a Weibull distribution, usually has a linear regression Weibull slope of around 3.5.

8.4.1 Parent Weibull Distribution Definition

The parent Weibull distribution definition is $P\left(x\right)=1-e^{-\left(\frac{x-x_0}{\eta-x_0}\right)^{\beta}}$. For real data, P is estimated by $\frac{m}{n+1}$, where n is the actual number of real data amplitudes from the trial or model test. For estimated lifetime data, P is estimated by $\frac{M}{N+1}$, where N is the estimated number of data points that would be generated in a lifetime exposure to a given single condition. P is the most probable value at N for an entire lifetime of exposure. The expected lifetime value $x=\left(\eta-x_0\right)\left[-ln\left(1-\frac{M}{N+1}\right)\right]^{\frac{1}{\beta}}+x_0$.

8.4.2 Extreme Value Distribution of the Ship Lifetime Max

The extreme value distribution of the ship lifetime max for the Nth point is $F\left(N\right) = \left[1 - e^{-\left(\frac{x-x_0}{\eta-x_0}\right)^{\beta}}\right]^{N}$. For multiple ship lifetimes, like a class of ships, $P(N)$ would have a distribution of its own. To find the extreme value of P that is exceeded in only 1 of 10 lifetimes, we would set $F\left(N\right) = 0.9$ and solve for x. The extreme value $x = \left(\eta - x_0\right)\left[-ln\left(1 - F^{\frac{1}{N}}\right)\right]^{\frac{1}{\beta}} + x_0$.

8.5 Weibull Example Problem

Assume that we have determined the shape parameter β=2 and the characteristic value η=10 from a moment method or linear regression Weibull analysis of some time history data. Now assume that a lifetime exposure to the condition being analyzed would produce a population of 500 points. To find the estimate of the "most probable extreme value" of this 500th point, solve for x in the following equation. $ln\left(ln\left(\frac{1}{1-\frac{500}{501}}\right)\right) = 2ln\left(x\right) - 2ln\left(10\right)$, $1.8272 = 2ln\left(x\right) - 4.6052$, which yields $x = 24.93$. To find the value in the distribution associated with the 500th point that is exceeded only 1 in 10 times; $0.9^{\frac{1}{500}} = 1 - e^{-\left(\frac{x}{10}\right)^2}$, $e^{-\left(\frac{x}{10}\right)^2} = 0.000211$, which yields $x = 29.095$. However, if you plug 24.93 into the extreme value equation and solve for $F\left(N\right)$, you will find that the most probable value of 24.93 has a 63.23% chance of being exceeded. It is likely that the designer will want a much smaller chance of the predicted extreme value being exceeded. For example, typical values selected for $F\left(N\right)$ might be 0.9, 0.95, 0.99, or 0.999. The choice of this value should be representative of the size of the population and the consequence of exceeding the maximum value.

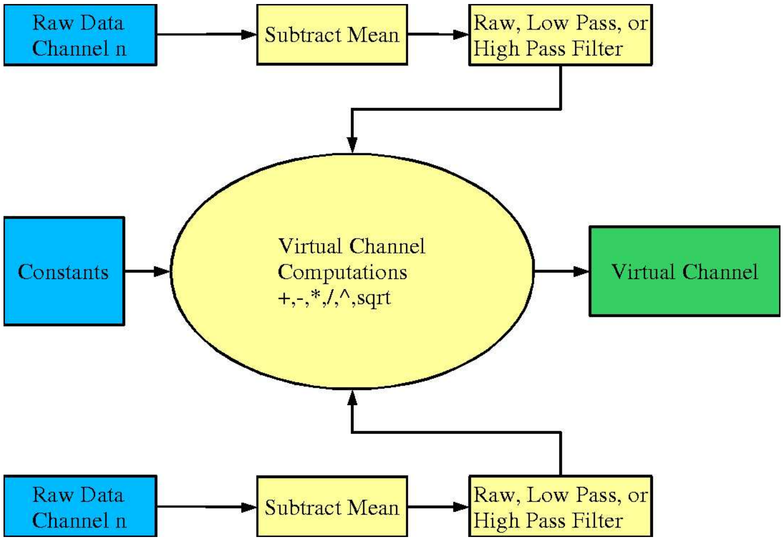


Figure 13: Virtual Channel Block Diagram

9 What Does “hvdas -convertdata” Do?

The module “hvdas -convertdata” can use the MMM configuration settings to modify and convert the original input data. Data conversion can be used to filter, smooth, scale, subtract initial condition zero values, or compute virtual channels from the original input data. Additionally, this operation can be performed on binary or ASCII input data. The output will be in ASCII tab-delimited text format.

9.1 Virtual Channels

Figure 13 shows a block diagram describing the procedure used to create virtual channels.

10 FIR Filter Design

HVDAS does not have any built-in limitations when it comes to finite impulse response (FIR) filters. HVDAS imports its filters as lists of filter coefficients, which need to be generated by some other program, such as Wave Metrics IGOR Pro IFDL. A higher number of filter coefficients is not necessarily better. Filters with high numbers of coefficients generally have a larger impulse response, which means that the output exhibits a "ringing" filtering impulse-like events and consequentially produce an impulse response, which could be larger than the data being analyzed. The number of filter coefficients also places requirements on the size of data file that HVDAS can filter. HVDAS requires data files to have more scans than the number of filter coefficients. It is mathematically impossible to apply a 1500 coefficient FIR filter to a data file which contains only 500 or 1000 samples.

For most ship structures, there is a significant difference between the direct wave-induced frequency response and the first-mode vibratory frequency response. This frequency gap can be advantageous when designing FIR filters. The example, shown in Figure 14, came from RV *Triton*, which is a full-scale trimaran demonstrator ship. The channel chosen is VSTR116, which is the midship keel gage measuring axial strain due to vertical longitudinal bending. Figure 14 shows PSDs of raw, low-pass FIR filtered, and high-pass FIR filtered data in Wave Metrics IGOR Pro. The crossover frequency of 1.55 Hz was chosen because it is about half way between the wave-induced energy and the first mode vibratory response. The crossover frequency needs some frequency space on either side of it because FIR filters are not perfect and do not have immediate or infinite attenuation. Figure 15 shows a close up of the crossover region. Figure 16 shows the raw, low-pass FIR filtered, and high-pass FIR filtered time histories.

The crossover region can be a single frequency step or a wide range of frequencies. Different data may not even need the filters to cross over in the frequency domain.

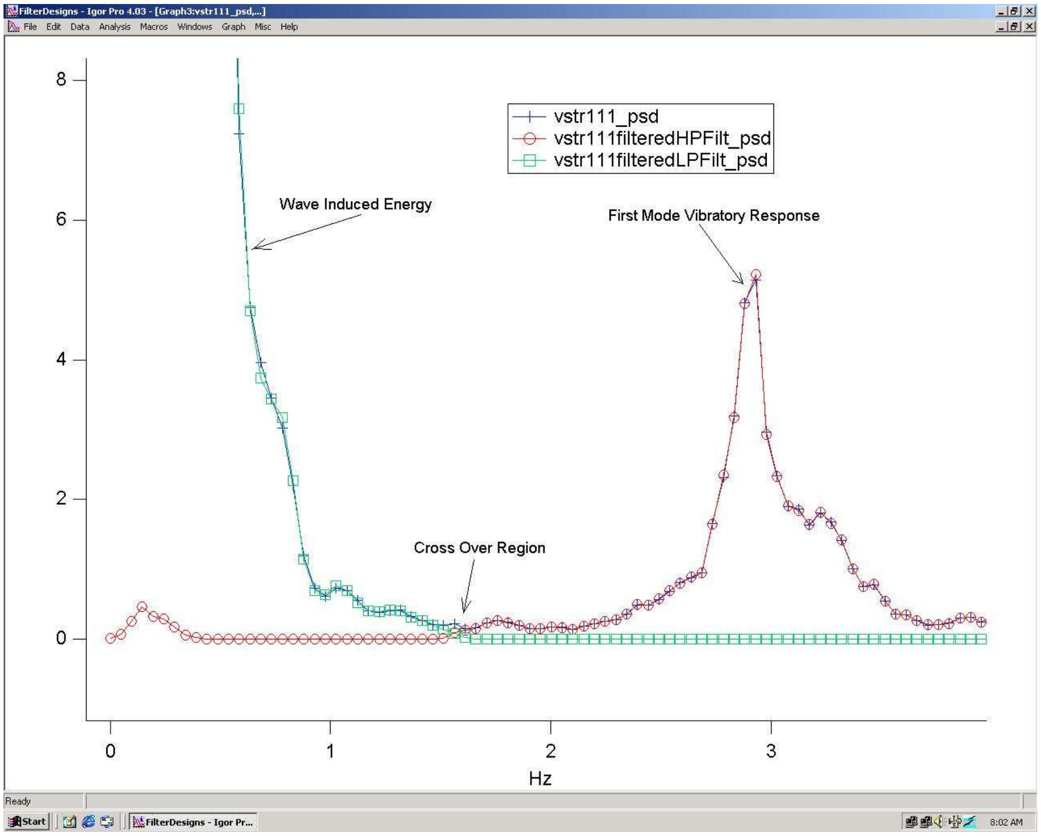


Figure 14: Crossover Region Filter Design Work

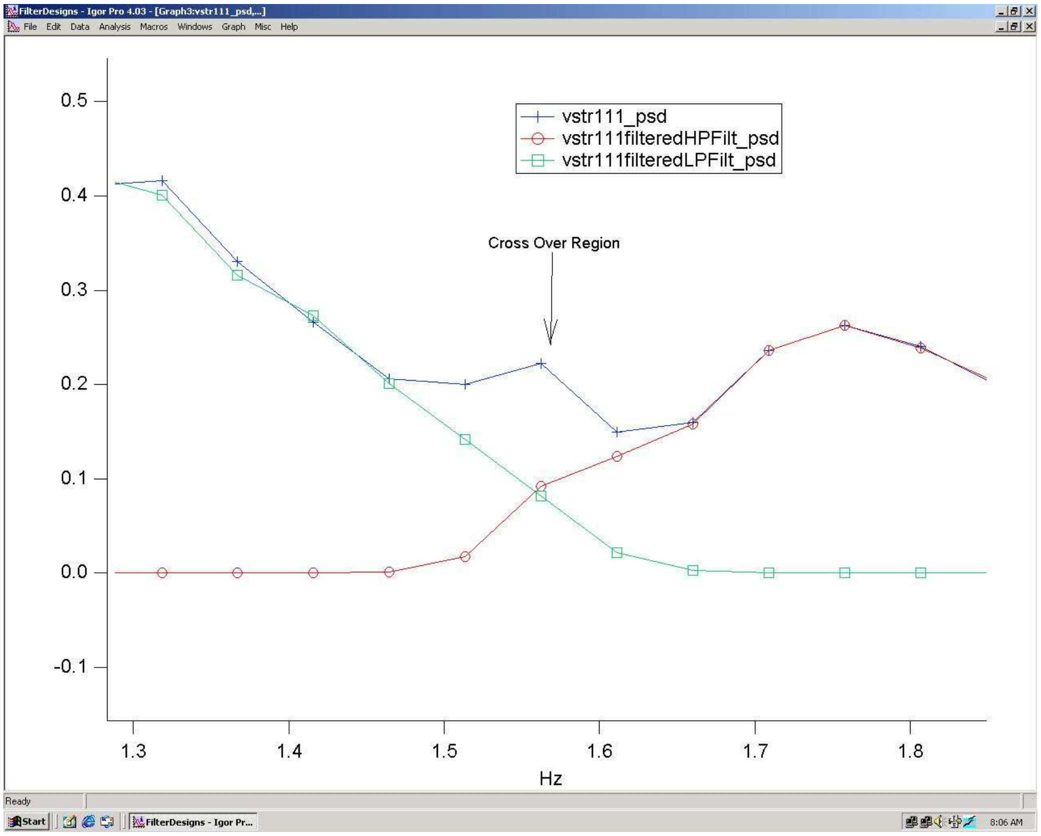


Figure 15: Crossover Region Close Up

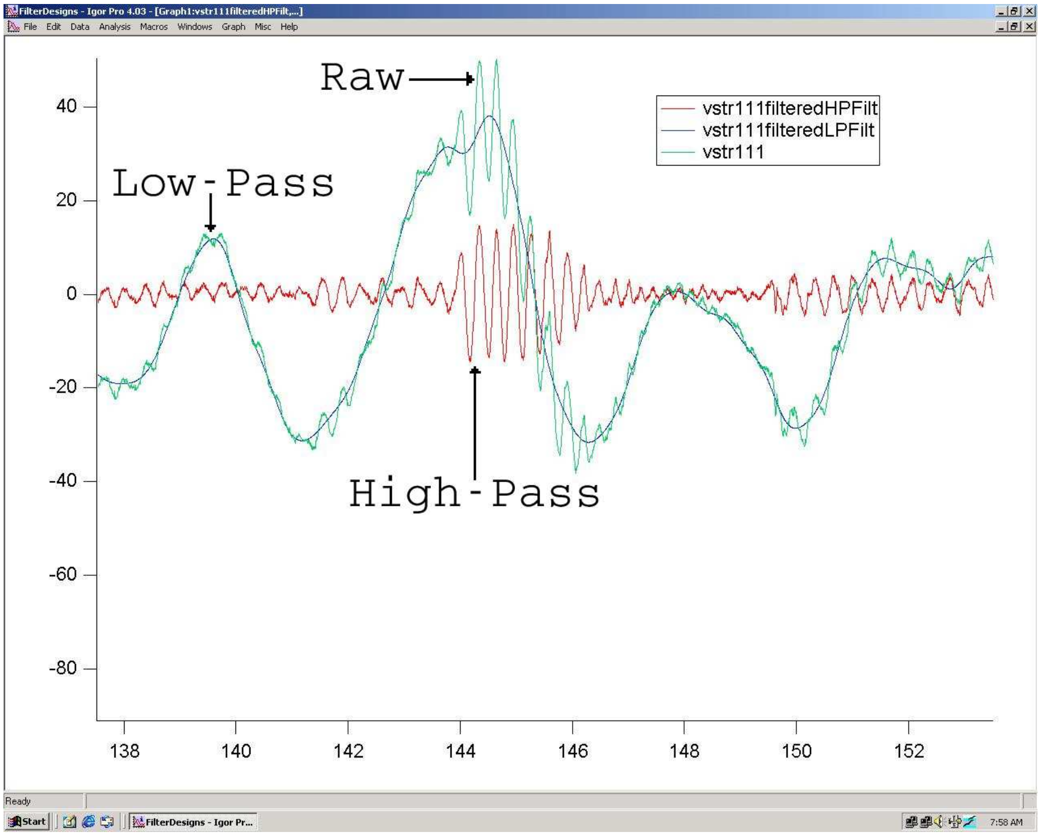

FilterDesigns - Igor Pro 4.03 - [Graph1:vstr111filteredHPFilt,...]
File Edit Data Analysis Macros Windows Graph Misc Help
Raw
Low-Pass
High-Pass
vstr111filteredHPFilt
vstr111filteredLPFilt
vstr111
40
20
0
-20
-40
-60
-80
138
140
142
144
146
148
150
152
Ready
Start
FilterDesigns - Igor Pr...
7:58 AM

Figure 16: Raw, Low-Pass FIR, and High-Pass FIR Time Histories

11 Dropout Detection and Correction

HVDAS detects and optionally corrects impulsive dropout behavior in the data as shown in Figure 17. Dropouts can be caused by loose ground wires, sudden changes in excitation voltage, errors in the analog to digital conversion, bit errors during data storage, and bit errors during data transmission.

HVDAS looks for several different types of dropouts: dropouts to exactly 0.0, maximum of 0.0, minimum of 0.0, single scan slope of greater than 12 sigma, and single scan slope of greater than 3 sigma followed directly by -3 sigma.

A single scan slope of greater than 12 sigma is illustrated by Figure 18. A single point dropout of at least 3 sigma followed by -3 sigma is illustrated in Figure 19.

If dropout correction is requested, the dropout will be replaced with the previous data point and the number of dropouts reported will be negative. If dropout correction is not requested, the number of dropouts reported will be positive.

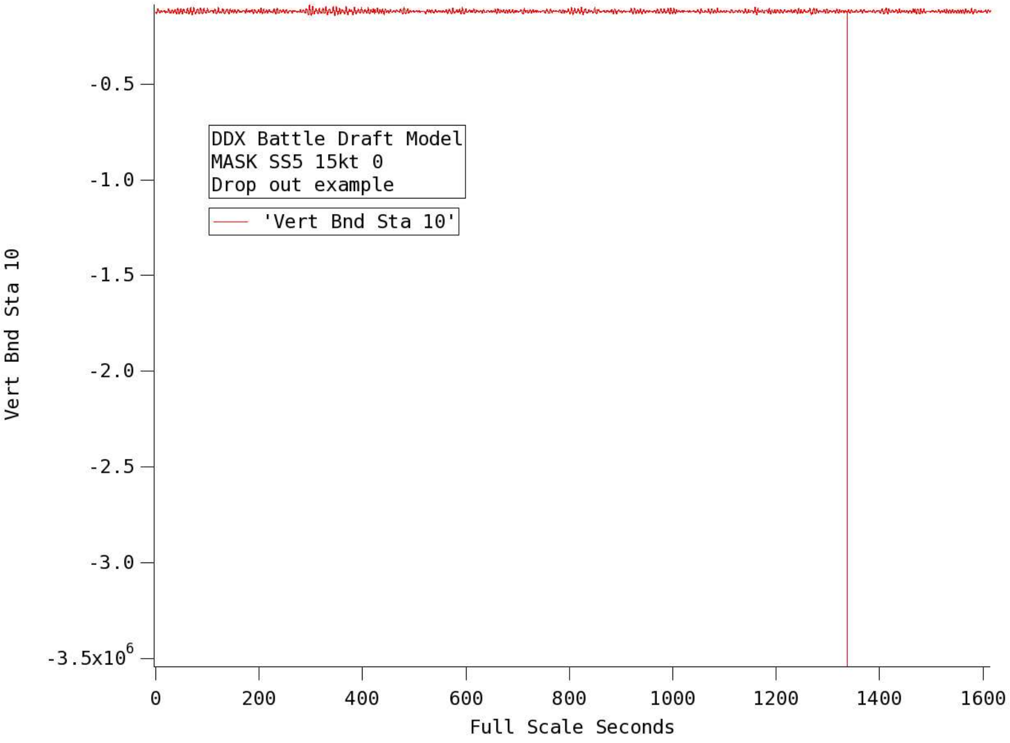


Figure 17: Dropout Example 1

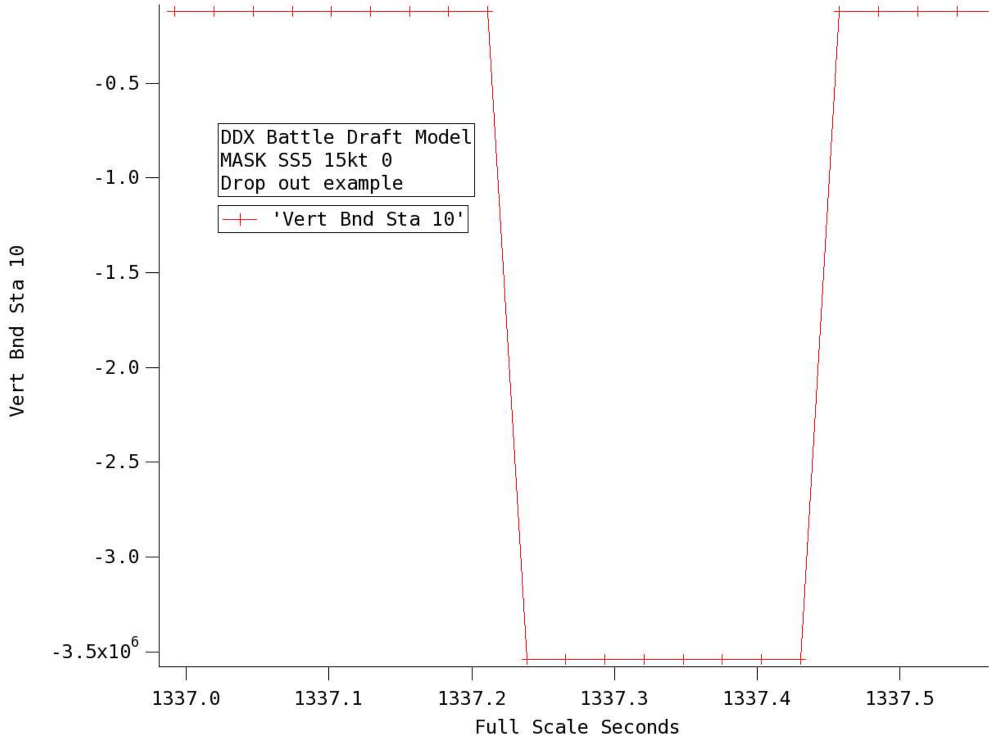


Figure 18: Dropout Example 2

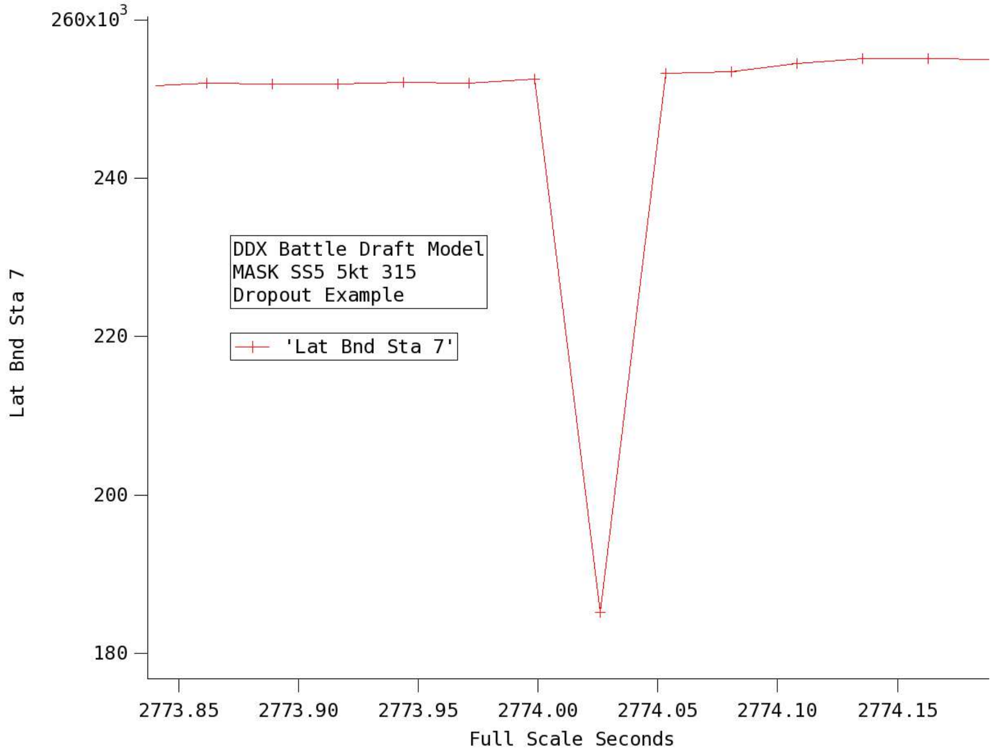


Figure 19: Dropout Example 3

12 Requirements and Limitations

- All channel numbers used by this software are numbered starting from zero for referencing purposes. The first data channel will be referred to as channel '0' and the last data channel will be the number of total channels minus one. For example: if your data files have 54 total channels, you will need to refer to each individual channel as 0 through 53.

- Each block of data (the folders/files to be run with the same configuration files) must have the same channel numbering scheme throughout. For example, if you have chosen to analyze channel '3', then channel '3' must refer to the same channel throughout the data block for consistent results.

- The only data types verified are binary, 4-byte, single-precision, high-byte-first interleaved data; 2-byte, unsigned-integer, low-byte-first interleaved data; and ASCII, tab-delimited, text data. 4-byte, single-precision, low-byte-first data files are also supported, but not heavily tested. Future versions will better accommodate multiple file formats.

- Data files in a single condition directory must have the same data format, number of channels, naming scheme, scale factors, and zero values.

- This software assumes that your data files follow the numeric naming conventions: FILE-PREFIX-####-FILE-EXTENSION or FILE-PREFIX-#-FILE-EXTENSION.

- It is a good idea to stay away from spaces in file names. Not all systems handle spaces in file names the same way, which makes it difficult to code for compatibility across all systems. Instead of using a space in a file name or condition label, replace it with an underscore "_".

- This software assumes that input data files have at least 4-character file names.

- You must have LAM MPI installed to compile the source code.

- You must have VSIPL core or core plus installed to compile the source code.

- You must have UNIX ODBC installed to compile the source code.

- This software will not do everything for you. You must spend time getting familiar with your data and its layout. You will be required to make intelligent decisions about analysis options which can affect your results. A good way to do this is by using Wave Metrics IGOR Pro to look at and analyze small amounts of data. Once you know what you are doing, run a small batch of analysis, check the results to make sure you are getting what you want, and then run the entire batch of automated analysis.

- Time selection support has been tested with 1 or more files. Time selection start and end times must be input correctly. Times that exceed the end of your data files may not be properly detected and may cause the program to segfault.

- HVDAS will currently only scale up to the number of channels in your data files. If you try to analyze a set of data files that contain 100 channels each on a cluster with 128 processors, HVDAS will only use 101 processors; 100 client processors and 1 server processor. This will be changed in the future to support up to (# channels * # data files + 1) processors.

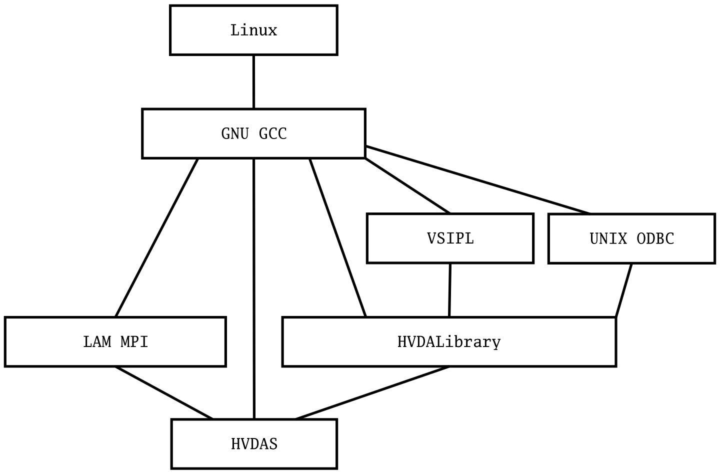


Figure 20: HVDAS General Software Overview

Table 1: Code and Executable Size Breakdown

Program/Library	Lines of Code	Executable/Compiled Size (KBytes)
hvdalibrary	23012	665
hvdas	11441	1480

13 Software Organization

13.1 General Software Overview

HVDAS is about 35,000 lines of original C++ code, including whitespace and comments, as of HVDAS version 2005-03-25. Printing HVDAS would require about 670 printed pages, assuming an average of 52 text lines per printed page on ISO A4 paper. This line count does not include any of the linked code in VSIPL, LAM-MPI, or UnixODBC, which were developed by other people. Figure 20 shows the relationship between the operating system, GNU GCC compilers, and HVDAS. Table 1 shows the breakdown of code and executable size among the different components of HVDAS.

13.2 How Does HVDAS Analyze Data Files So Fast?

HVDAS uses a coarse parallelization to achieve its speed. At NSWCCD, Code 65, there are typically a large number of files and data channels to calculate statistics from. In the early design phases of HVDAS, it was determined that speeding up bulk data analysis was more important than speeding up the analysis of a single channel. This led to the use of a single data channel, or time history, as the smallest piece of data to analyze. Breaking up the work by channel is a logical and highly parallel analysis method, but it does not help analyze a single data channel any faster than our uniprocessor analysis code.

Figure 21 shows the work flow distribution which allows HVDAS to perform and scale so well.

13.3 How Do The HVDAS Programs Analyze Data Files In Parallel?

13.3.1 Server Algorithm Overview

Figure 22 shows how the server portion of HVDAS loops, breaks up the calculations, distributes work to the client nodes, and receives results from the client nodes.

13.3.2 Client Algorithm Overview

Figure 23 shows how the client portion of HVDAS loops, receives work requests from the server node, performs calculations, and sends results back to the server node.

13.4 Data Distribution

HVDAS does not explicitly send any actual data channels to the client nodes using the Message Passing Interface (MPI). HVDAS sends only the information that the clients need to open up the data files themselves. HVDAS is organized this way to minimize the amount of bandwidth used by server to client communication. This distributed data importing allows the data to be hosted in a wide variety of places.

The data can be easily placed on a RAID-5 archive machine and shared with the client nodes via Microsoft Windows Network File Sharing (SAMBA) or UNIX Network File System (NFS), but that is just one possibility. The HVDAS method

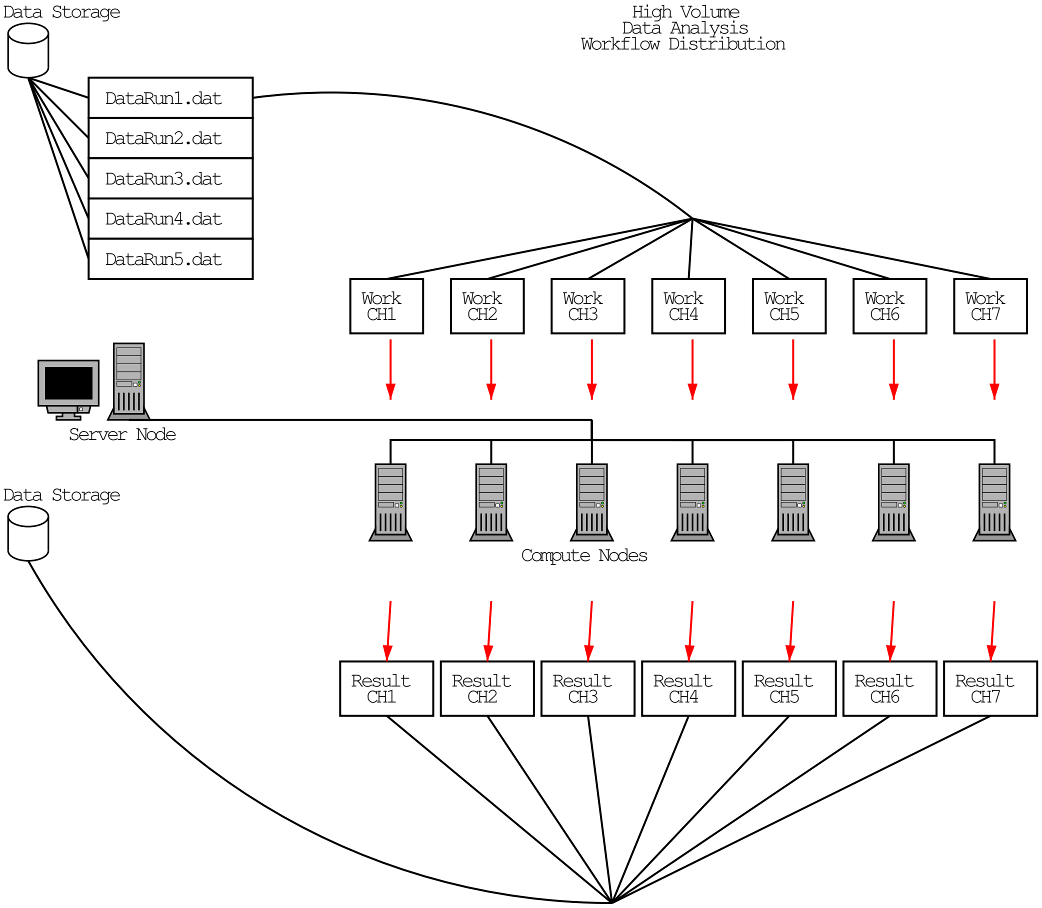


Figure 21: HVDAS Work Flow Distribution

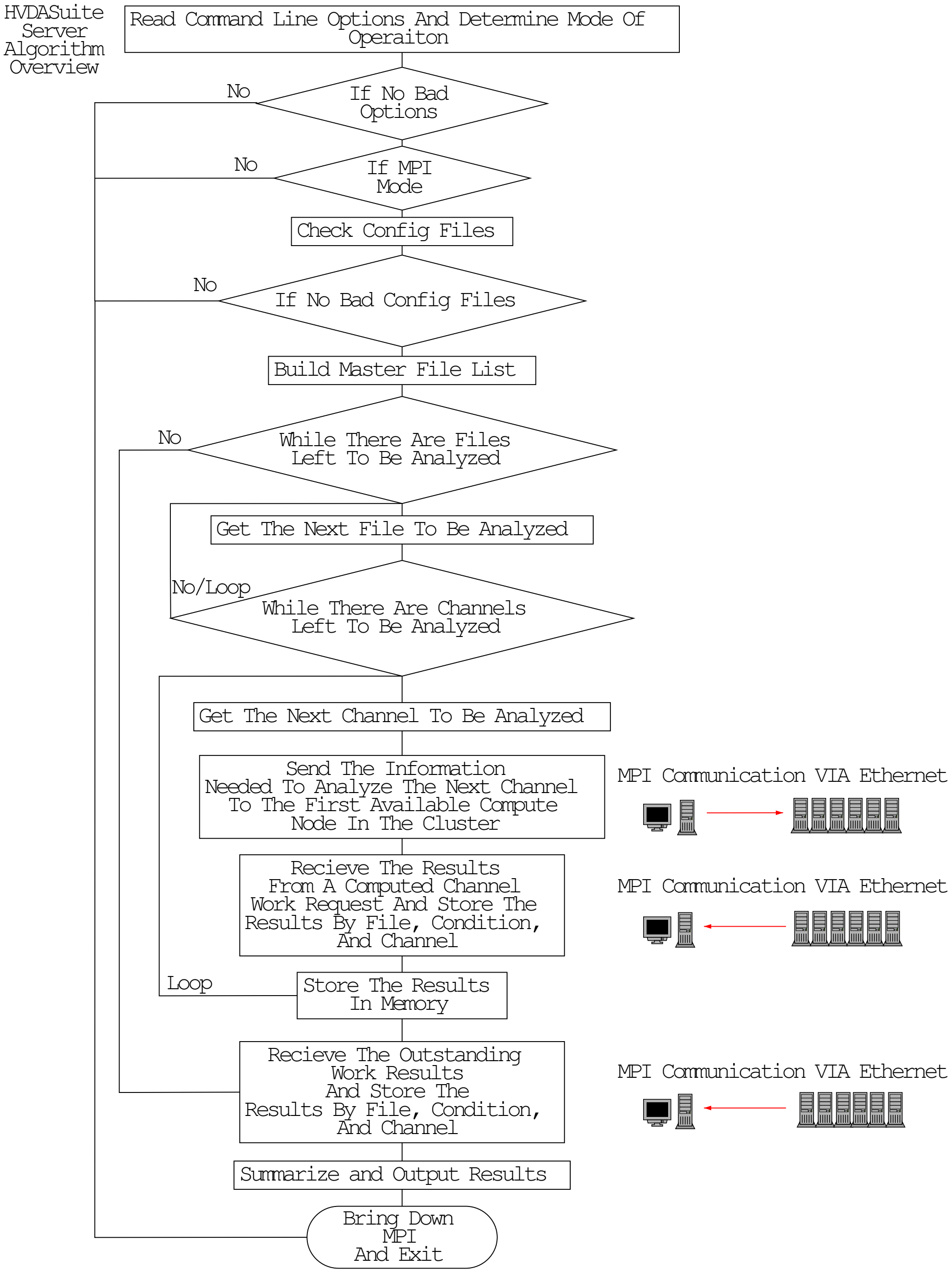

HVDASuite Server Algorithm Overview
Read Command Line Options And Determine Mode Of Operaiton
If No Bad Options
No
If MPI Mode
No
Check Config Files
If No Bad Config Files
No
Build Master File List
While There Are Files Left To Be Analyzed
No
Get The Next File To Be Analyzed
While There Are Channels Left To Be Analyzed
No/Loop
Get The Next Channel To Be Analyzed
Send The Information Needed To Analyze The Next Channel To The First Available Compute Node In The Cluster
MPI Communication VIA Ethernet
Recieve The Results From A Computed Channel Work Request And Store The Results By File, Condition, And Channel
MPI Communication VIA Ethernet
Store The Results In Memory
Loop
Recieve The Outstanding Work Results And Store The Results By File, Condition, And Channel
MPI Communication VIA Ethernet
Summarize and Output Results
Bring Down MPI And Exit

Figure 22: HVDAS Server Algorithm Overview

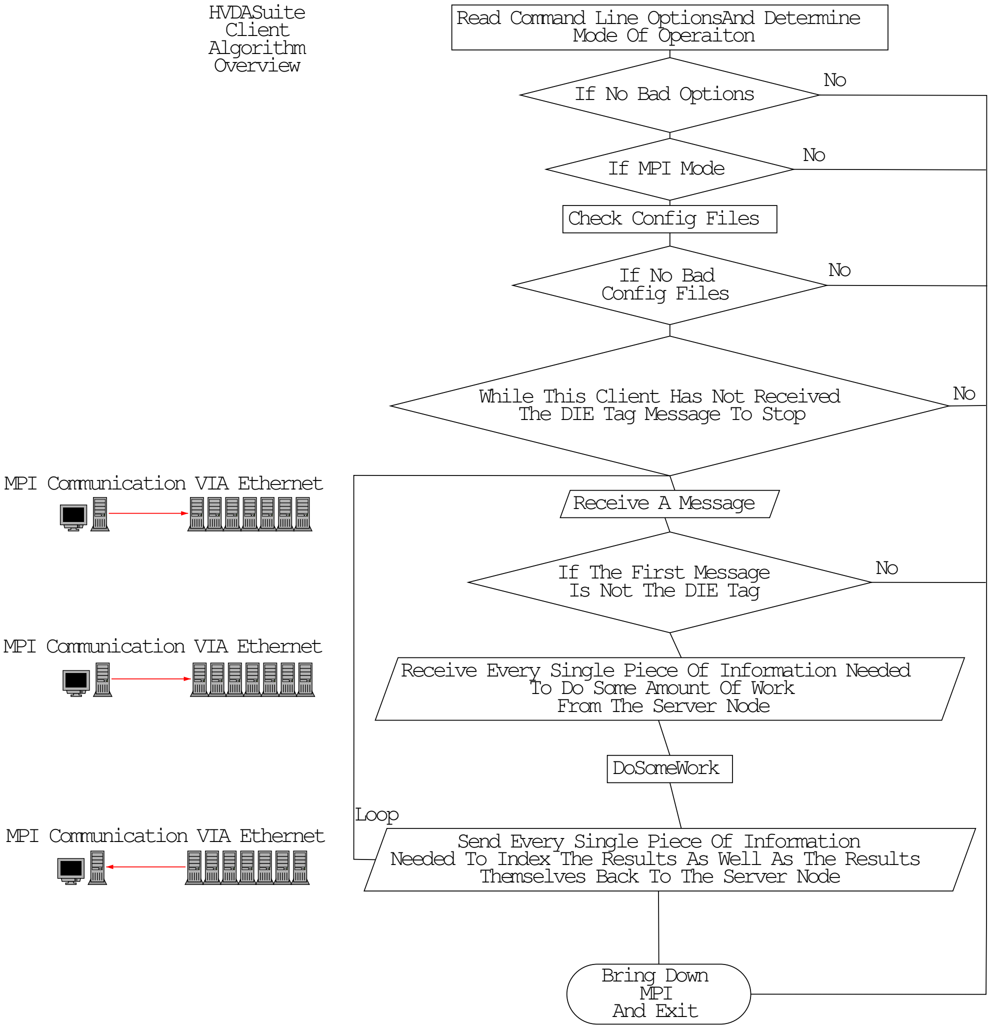

HVDASuite
Client
Algorithm
Overview
Read Command Line OptionsAnd Determine
Mode Of Operaiton
If No Bad Options
No
If MPI Mode
No
Check Config Files
If No Bad
Config Files
No
While This Client Has Not Received
The DIE Tag Message To Stop
No
MPI Communication VIA Ethernet
Receive A Message
If The First Message
Is Not The DIE Tag
No
MPI Communication VIA Ethernet
Receive Every Single Piece Of Information Needed
To Do Some Amount Of Work
From The Server Node
DoSomeWork
Loop
MPI Communication VIA Ethernet
Send Every Single Piece Of Information
Needed To Index The Results As Well As The Results
Themselves Back To The Server Node
Bring Down
MPI
And Exit

Figure 23: HVDAS Client Algorithm Overview

of data import allows the data to be hosted on any cluster-accessible file system. Data can even be hosted on each client node; as long as every other client node has network access to it.

One of the most beneficial results of this design is the ability to use distributed, or parallel, file systems. This allows the cluster nodes, or additional data nodes, to stripe and store data, which will maximize storage space, minimize network bottlenecks, and maximize input output speed. This design also allows Linux to handle caching at the file system level for each node.

HVDAS has been successfully tested with SAMBA, NFS, Parallel Virtual File System (PVFS), and all combinations of the three.

13.5 Bandwidth Utilization

Minimized network communication maximizes performance and expandability. Network saturation can kill the performance benefits of using Linux clusters. HVDAS relies heavily on Linux and UNIX's ability to cache almost everything without any effort from an application. When a client node reads in a single channel from a data file, the operating system, Linux, will try to cache the entire data file in memory regardless of the source of the data file. This means that data files that are read in from networked file systems will be cached in memory by the client node's operating system. If that client's next work request is a different channel in the same data file, which will happen quite often if the number of nodes in the cluster is less than the number of channels in the data file, the operating system will read the data from cache instead of reading it again over the network. This behavior does not require any special programming, which is another reason to not send the data channels using MPI.

13.6 Storage Minimization

All of HVDAS' results can be optionally stored as ASCII tab-delimited text files. Ensuring that all of the results files, with a significant amount of data in them, are not padded with any kind of white space or extra characters reduces file size. HVDAS can optionally skip individual file statistics output or condition summary output to further cut down on results storage space. HVDAS can also optionally store all of the results into a relational database. Relational databases have been heavily developed for storage space efficiency.

13.7 Scalability

HVDAS can scale from one to thousands of nodes. It is possible to run one server and one client portion of HVDAS on a single processor. It is possible to run one server and two client portions of HVDAS on a dual processor computer to use both processors at the same time. It is also possible to run HVDAS on a cluster of single and/or Symmetric Multi-Processing (SMP) nodes and use every processor in the cluster. HVDAS will currently only scale up to the number of channels in your data files. If you try to analyze a set of data files, that contain 100 channels each, on a cluster with 128 processors, HVDAS will only use 101 processors; 100 client processors and 1 server processor. This will be changed in the future to support up to (# channels * # data files + 1) processors.

13.8 Performance Maximization

HVDAS is quite fast even on a single processor computer. Huge linear performance increases come as a result of adding nodes and processors. Network speed, processor speed, memory speed, system bus speed, and input output storage subsystem speed also greatly impact performance. The key to maximizing the parallel calculation performance increase is to make sure that the nodes spend more time calculating than they do communicating or waiting.

13.9 Parallel Benefits

Figure 24 shows the benefits of using HVDAS on a Linux cluster. It is evident that the compute time roughly halves when the number of nodes doubles. The parallel calculation time is nearly $CalcTime_n = CalcTime_1 * \frac{1}{n}$, where n is the number of processors. This behavior will continue until the communication network or some other part of the Linux cluster becomes saturated. This performance increase behavior makes analysis time prediction very easy. Using this behavior, it is possible to predict the analysis time for different data sets or additional processors.

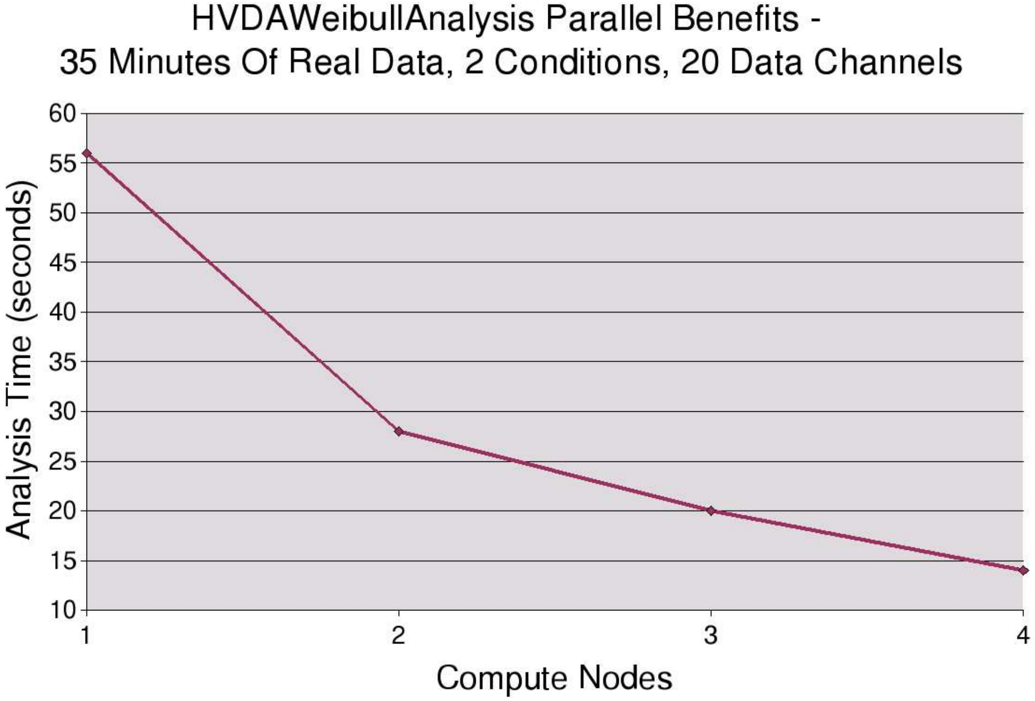


Figure 24: HVDAS Parallel Benefits

14 Data Formats

This software supports four data formats: ASCII columns, 4-byte binary interleaved with high-byte-first byte ordering, 4-byte binary interleaved with low-byte-first byte ordering, and 2-byte unsigned integer interleaved with low-byte-first byte ordering. The data format is a required input for the program to function properly. If the data are in any other format, they must be converted to one of the acceptable formats prior to being run or additional code must be added to HVDAS.

Using the wrong format may not cause a fatal error but would cause the output to be completely meaningless. The headers in all data files are ignored.

If you have a header that exists in your data on a per scan basis, you should try to read the scan header in as one or more dummy channels instead of converting data formats or writing another input plug-in.

14.1 ASCII Columns

In the analysis software, the headers of the ASCII files are skipped. The data are assumed to be in vertical, tab-delimited columns.

14.2 Binary Interleaved High Byte First

- IEEE 32-bit (4-byte) single precision floating point numbers,
- Big endian (high-byte-first) byte ordering,
- Stored using interleaved data points.

14.3 Binary Interleaved Low Byte First

- IEEE 32-bit (4-byte) single precision floating point numbers,
- Little endian (low-byte-first) byte ordering,
- Stored using interleaved data points.

14.4 Binary Interleaved Unsigned Short

- IEEE 16-bit (2-byte) unsigned integers,
- Little endian (low-byte-first) byte ordering,
- Stored using interleaved data points.
- This data type is particularly useful when reading in data stored as integer scans or integer counts.

15 Input

Input to the HVDAS programs can be highly varied, but there are a few guidelines to follow. Keep it as simple as possible. Analyze similar data at the same time. Keep these things in mind when collecting data that you plan on analyzing with HVDAS.

15.1 When HVDAS is Simple to Set Up

Analyze test data, which are made up of multiple test conditions, as one data set. The simplest input directory structure is organized as /Trial0/Condition0-N/Run0-N, where Runs 0-N are numbered sequentially and follow the convention File Prefix<0-N>File Extension. This is the easiest possible input structure to analyze because HVDAS was built around this input structure. Everything will be easy to set up if the number of channels, the channel order, the scale factors, the sample rate, and the data file format remain constant.

15.2 When HVDAS is Complicated to Set Up

If the input data set is not set up in the ideal structure, all is not lost, but you should be more familiar with executing the code to avoid running into problems. Simple UNIX scripts can be written to rename files and folders in bulk, which will make the user's job less painful. If the data exist in multiple file formats or sample rates, you should consider running the analysis in two phases or converting all of the data to a common file format. Configuration will be more difficult to set up if the number of channels, channel order, scale factors, sample, or data file format vary.

15.3 Example Input Structure

This is just one possible file path structure. This example assumes that each condition folder contains binary data files named datarun_0.binary through datarun_3.binary, but each condition does not have to be identical. Example output and configuration files are detailed in their respective sections.

```
/server/test/condition1/
/server/test/condition2/
/server/test/condition3/
/server/test/HVDAGlobalConfiguration.conf
/server/test/HVDAMMMConfiguration.conf
```

/server/test/HVDAPSDConfiguration.conf
/server/test/HVDARAOConfiguration.conf
/server/test/HVDAWeibullConfiguration.conf

16 Text File Output

Text file output from every program in HVDAS is visible with Excel, Star Office, Open Office, Word Pad, Text Pad, nedit, and gnuplot. The results are saved as generic ASCII, tab-delimited, text files. All header information in HVDAS output files have a preceding “#”, which makes them compatible with gnuplot.

Output is stored in a directory structure similar to the input data structure. The base results folder, named by the user in a ’HVDA*Configuration.conf’ file, will contain all of the output. For basic MMM analysis, the results will be in the sub-folder named ’HVDAMMMResults’. Modified and converted input data will be stored in the sub-folder named ’HVDAConvertedOutputData’. For PSD analysis, the results will be in the sub-folder named ’HVDAPSDResults’. For RAO analysis, the results will be in the sub-folder named ’HVDARAOResults’. For Weibull analysis, the results will be in the sub-folder named ’HVDAWeibullResults’. At the next level the results are broken down into their respective conditions using the condition labels defined in the ’HVDAGlobalConfiguration.conf’ file. These folders contain the actual condition results, sorted by filter type, original data file name and index.

16.1 MMM Analysis Output

Each row in the MMM output represents a channel of analyzed data and contains the channel number, channel name, mean, max, min, $(maximum - mean)$, $(minimum - mean)$, range, maximum peak-to-peak, standard deviation, kurtosis, mean crossings, and the ratio $\frac{(maximum-mean)}{(standard deviation)}$. Other pertinent information is in the header at the top of the output file. Each condition, or folder, is summarized on a channel-by-channel basis. The summary gives values for each channel for each statistical value: mean, $(maximum - mean)$, etc.

For each condition, the program will create one results file for each input file for each of four possible filter states. In addition, one summary file is created for each filter state in each condition and is saved in the results folder for that condition. As with individual files, pertinent information is written at the top of the summary file.

Output is stored in a directory structure similar to the input data structure. The base results folder, named by the user in the ’HVDAMMMConfiguration.conf’ file, will contain all of the output. For basic MMM analysis the results will be in the sub-folder named ’HVDAMMMResults’. Modified and converted input data will be stored in the sub-folder named 'HVDAConvertedOutputData'. At the next level the results are broken down into their respective conditions using the condition labels defined in the 'HVDAGlobalConfiguration.conf' file. These folders contain the actual condition results, sorted by filter type, original data file name and index.

16.1.1 Example Output Structure

This is just one possible output structure. Your actual output structure will depend on your choices for base folder name, condition labels, desired filters, and the number of files analyzed. The individual and summary files are shown below.

```
/server/test/HVDAMMMResults/cond1/HVDAMMM_Raw_cond1_data_0.txt
/server/test/HVDAMMMResults/cond1/HVDAMMM_Raw_cond1_data_1.txt
/server/test/HVDAMMMResults/cond1/HVDAMMM_Raw_cond1_data_2.txt
/server/test/HVDAMMMResults/cond1/HVDAMMM_Raw_cond1_data_3.txt
/server/test/HVDAMMMResults/cond1/HVDAMMM_Raw_cond1_Summary.txt
/server/test/HVDAMMMResults/cond2/HVDAMMM_Raw_cond2_data_0.txt
/server/test/HVDAMMMResults/cond2/HVDAMMM_Raw_cond2_data_1.txt
/server/test/HVDAMMMResults/cond2/HVDAMMM_Raw_cond2_data_2.txt
/server/test/HVDAMMMResults/cond2/HVDAMMM_Raw_cond2_data_3.txt
/server/test/HVDAMMMResults/cond2/HVDAMMM_Raw_cond2_Summary.txt
/server/test/HVDAMMMResults/cond3/HVDAMMM_Raw_cond3_data_0.txt
/server/test/HVDAMMMResults/cond3/HVDAMMM_Raw_cond3_data_1.txt
/server/test/HVDAMMMResults/cond3/HVDAMMM_Raw_cond3_data_2.txt
/server/test/HVDAMMMResults/cond3/HVDAMMM_Raw_cond3_data_3.txt
/server/test/HVDAMMMResults/cond3/HVDAMMM_Raw_cond3_Summary.txt
```

16.2 PSD Analysis Output

The first, left-most column contains the frequencies that correspond row by row with the PSD magnitudes. Each remaining column represents a channel of analyzed data and contains the channel name and PSD, which was computed using the PSDFFTMSAM. Other pertinent information is in the header at the top of the output file, including the parameters used to calculate the PSD I-Skip and segment length. Each condition, or folder, is summarized on a channel-by-channel basis. The summary gives values for each frequency's PSD magnitude for each channel.

For each condition, the program will create one results file for each input file for either raw or low-pass filter states. In addition, one summary file is created for either the raw or low-pass filter state in each condition and is saved in the results folder for that condition. As with individual files, pertinent information is written at the top of the summary file.

Output is stored in a directory structure similar to the input data structure. The base results folder, named by the user in the 'HVDAPSDConfiguration.conf' file, will contain all of the output. For this basic analysis, the results will be in the sub-folder named 'HVDAPSDResults'. At the next level, the results are broken down into their respective conditions using the condition labels defined in the 'HVDAGlobalConfiguration.conf' file. These folders contain the actual condition results, sorted by filter type, original data file name and index.

16.2.1 Example Output Structure

This is just one possible output structure. Your actual output structure will depend on your choices for base folder name, condition labels, desired filters, and the number of files analyzed. The individual and summary files are shown below.

/server/test/HVDAPSDResults/cond1/HVDAPSD_Raw_cond1_data_0.txt
/server/test/HVDAPSDResults/cond1/HVDAPSD_Raw_cond1_data_1.txt
/server/test/HVDAPSDResults/cond1/HVDAPSD_Raw_cond1_data_2.txt
/server/test/HVDAPSDResults/cond1/HVDAPSD_Raw_cond1_data_3.txt
/server/test/HVDAPSDResults/cond1/HVDAPSD_Raw_cond1_Summary.txt
/server/test/HVDAPSDResults/cond2/HVDAPSD_Raw_cond2_data_0.txt
/server/test/HVDAPSDResults/cond2/HVDAPSD_Raw_cond2_data_1.txt
/server/test/HVDAPSDResults/cond2/HVDAPSD_Raw_cond2_data_2.txt
/server/test/HVDAPSDResults/cond2/HVDAPSD_Raw_cond2_data_3.txt
/server/test/HVDAPSDResults/cond2/HVDAPSD_Raw_cond2_Summary.txt
/server/test/HVDAPSDResults/cond3/HVDAPSD_Raw_cond3_data_0.txt
/server/test/HVDAPSDResults/cond3/HVDAPSD_Raw_cond3_data_1.txt
/server/test/HVDAPSDResults/cond3/HVDAPSD_Raw_cond3_data_2.txt
/server/test/HVDAPSDResults/cond3/HVDAPSD_Raw_cond3_data_3.txt
/server/test/HVDAPSDResults/cond3/HVDAPSD_Raw_cond3_Summary.txt

16.3 RAO Analysis Output

The first, left-most column contains the frequencies that correspond row by row with the root RAO magnitudes. Each remaining column represents a channel of analyzed data and contains the channel name and square root of the RAO magnitude, which was computed using $\sqrt{RAO} = \sqrt{\frac{PSDFFTMSAM(DataChannel)}{PSDFFTMSAM(DivisorChannel)}}$. Other pertinent information is in the header at the top of the output file, including the parameters used to calculate the PSD I-Skip and segment length. Each condition, or folder, is summarized on a channel-by-channel basis. The summary gives values for each frequency's PSD magnitude for each channel.

For each condition, the program will create one results file for each input file for either raw or low-pass filter states. In addition, one summary file is created for either the raw or low-pass filter state in each condition and is saved in the results folder for that condition. As with individual files, pertinent information is written at the top of the summary file.

Output is stored in a directory structure similar to the input data structure. The base results folder, named by the user in the 'HVDARAOConfiguration.conf' file, will contain all of the output. For this basic analysis, the results will be in the sub-folder named 'HVDARAOResults'. At the next level, the results are broken down into their respective conditions using the condition labels defined in the 'HVDAGlobalConfiguration.conf' file. These folders contain the actual condition results, sorted by filter type, original data file name and index.

16.3.1 Example Output Structure

This is just one possible output structure. Your actual output structure will depend on your choices for base folder name, condition labels, desired filters, and the number of files analyzed. The individual and summary files are shown below.

```
/server/test/HVDARAOResults/cond1/HVDARAO_Raw_cond1_data_0.txt
/server/test/HVDARAOResults/cond1/HVDARAO_Raw_cond1_data_1.txt
/server/test/HVDARAOResults/cond1/HVDARAO_Raw_cond1_data_2.txt
/server/test/HVDARAOResults/cond1/HVDARAO_Raw_cond1_data_3.txt
/server/test/HVDARAOResults/cond1/HVDARAO_Raw_cond1_Summary.txt
/server/test/HVDARAOResults/cond2/HVDARAO_Raw_cond2_data_0.txt
/server/test/HVDARAOResults/cond2/HVDARAO_Raw_cond2_data_1.txt
/server/test/HVDARAOResults/cond2/HVDARAO_Raw_cond2_data_2.txt
/server/test/HVDARAOResults/cond2/HVDARAO_Raw_cond2_data_3.txt
/server/test/HVDARAOResults/cond2/HVDARAO_Raw_cond2_Summary.txt
/server/test/HVDARAOResults/cond3/HVDARAO_Raw_cond3_data_0.txt
/server/test/HVDARAOResults/cond3/HVDARAO_Raw_cond3_data_1.txt
/server/test/HVDARAOResults/cond3/HVDARAO_Raw_cond3_data_2.txt
/server/test/HVDARAOResults/cond3/HVDARAO_Raw_cond3_data_3.txt
/server/test/HVDARAOResults/cond3/HVDARAO_Raw_cond3_Summary.txt
```

16.4 Weibull Analysis Output

The Weibull output is organized by channels. Each input channel yields up to 5 output columns depending on the calculation performed. The result columns report the actual data amplitudes used, linearized Weibull X data, linearized Weibull Y data, and linear Weibull trend lines.

Other pertinent information is in the header at the top of the output file, including the parameters used to calculate the Weibull lifetime statistics. Each condition, or folder, is summarized on a channel-by-channel basis. For each condition, the program will create one results file for either raw, low-pass, or high-pass filter states.

Output is stored in a directory structure similar to the input data structure. The base results folder, named by the user in the 'HVDAWeibullConfiguration.conf' file, will contain all of the output. For this basic analysis, the results will be in the sub-folder named 'HVDAWeibullResults'. At the next level, the results are broken down into their respective conditions using the condition labels defined in the 'HVDAGlobalConfiguration.conf' file. These folders contain the actual condition results, sorted by filter type, original data file name and index.

16.4.1 Example Output Structure

This is just one possible output structure. Your actual output structure will depend on your choices for base folder name, condition labels, desired filters, and the number of files analyzed. The individual and summary files are shown below.

/server/test/HVDAWeibullResults/cond1/HVDAWeibull_Raw_cond1_Summary.txt
/server/test/HVDAWeibullResults/cond2/HVDAWeibull_Raw_cond2_Summary.txt
/server/test/HVDAWeibullResults/cond3/HVDAWeibull_Raw_cond3_Summary.txt

17 Database Organization and SQL Output

HVDAS can also store all results into a relational database using SQL. The database server used for development is Postgre SQL, but HVDAS should work with any relational database that UNIX ODBC supports. The database organization allows multiple model test and full-scale trial analysis results to be stored together. Figure 25 shows an entity relationship model of the database including table names. Figure 26 shows the table names, column names, and data types.

17.1 Array Storage

Arrays like PSD frequencies, PSD magnitudes, and Weibull amplitudes are stored in the database as ASCII in a text field. The text data type in Postgre SQL seems to be limited to 65536 characters at the time of this writing. Assuming 10 characters per array number and delimiter character, over 5000 numbers could be stored in the array using this method, which is currently plenty for our purposes. The arrays are comma-delimited and exclamation-terminated ASCII text. Here is an example database text array storing numbers from 1 to 10: “1,2,3,4,5,6,7,8,9,10!”.

17.2 psql and Useful Queries

psql is a command line tool provided with Postgre SQL to interact with the database. “\d” will give a list of all tables in the current database; it describes the database. “\d tablename” will describe the named table and list all columns and data types. Below are some useful SQL queries which can be run from withing psql.

- select * from test; - shows all tests with results in the database.
- select count(*) from condition where testname=’sometestname’; - counts the number of conditions for a given test.
- select count(*) from run where testname=’sometestname’; - counts the number of runs for a given test.
- select conditionname,totalruntime from condition where testname=’sometestname’; - shows total run time for all conditions for a given test. This can be useful for identifying conditions without enough data to be statistically useful.

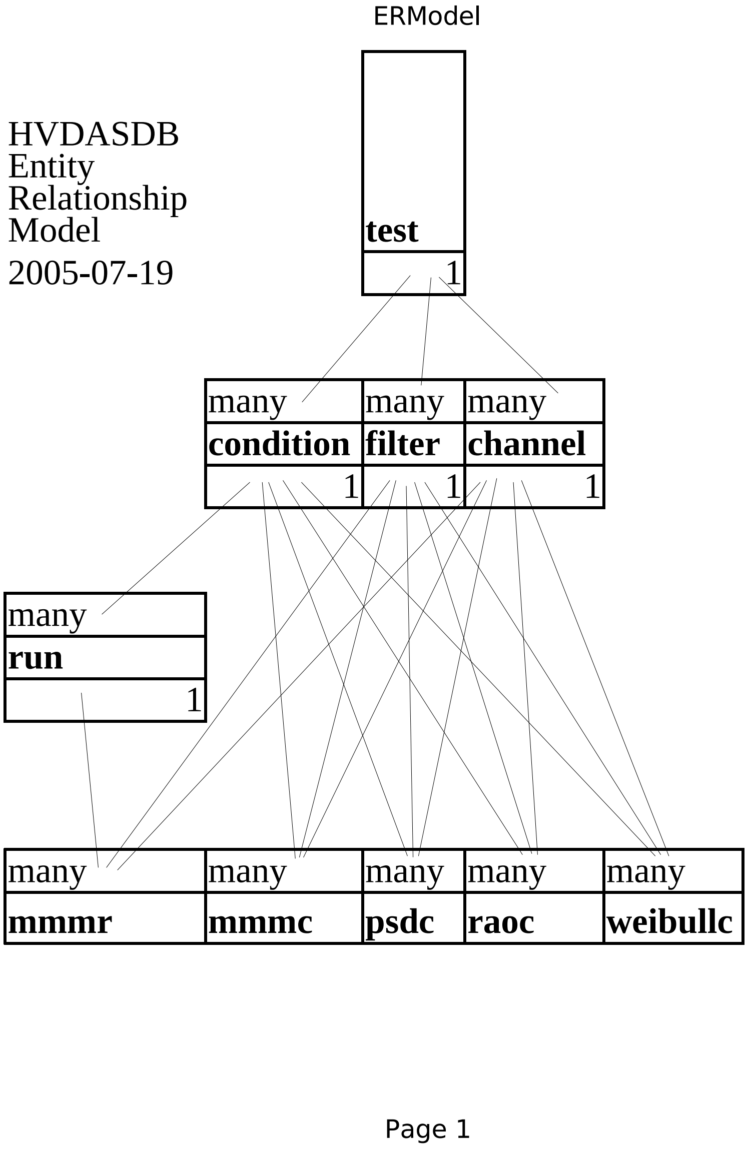
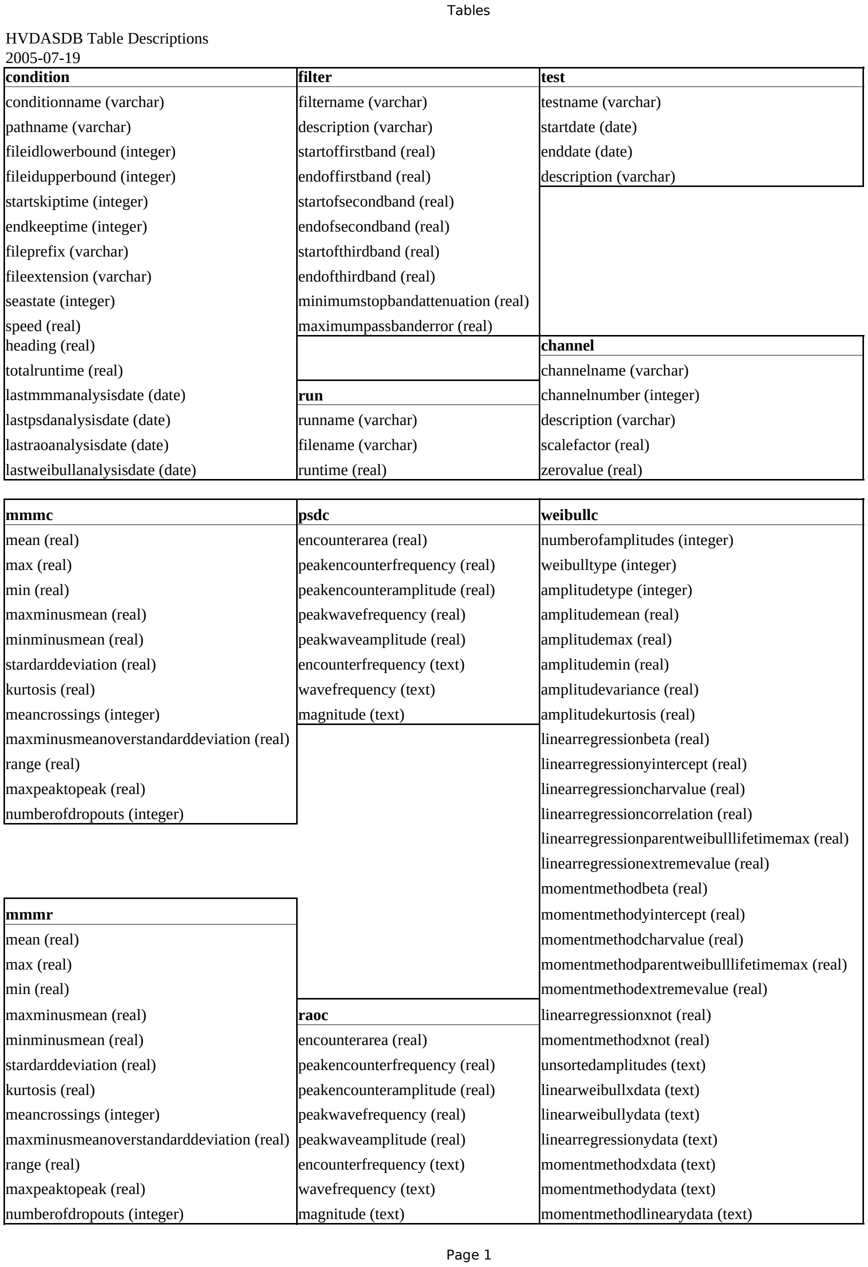


Figure 25: HVDAS Database Entity Relationship Model



Tables

HVDASDB Table Descriptions
2005-07-19

condition
conditionname (varchar)
pathname (varchar)
fileidlowerbound (integer)
fileidupperbound (integer)
startskiptime (integer)
endkeeptime (integer)
fileprefix (varchar)
fileextension (varchar)
seastate (integer)
speed (real)
heading (real)
totalruntime (real)
lastmmmanalysisdate (date)
lastpsdanalysisdate (date)
lastraoanalysisdate (date)
lastweibullanalysisdate (date)

filter
filtername (varchar)
description (varchar)
startoffirstband (real)
endoffirstband (real)
startofsecondband (real)
endofsecondband (real)
startofthirdband (real)
endofthirdband (real)
minimumstopbandattenuation (real)
maximumpassbanderror (real)

test
testname (varchar)
startdate (date)
enddate (date)
description (varchar)

run
runname (varchar)
filename (varchar)
runtime (real)

channel
channelname (varchar)
channelnumber (integer)
description (varchar)
scalefactor (real)
zerovalue (real)

mmmc
mean (real)
max (real)
min (real)
maxminusmean (real)
minminusmean (real)
stardarddeviation (real)
kurtosis (real)
meancrossings (integer)
maxminusmeanoverstandarddeviation (real)
range (real)
maxpeaktopeak (real)
numberofdropouts (integer)

psdc
encounterarea (real)
peakencounterfrequency (real)
peakencounteramplitude (real)
peakwavefrequency (real)
peakwaveamplitude (real)
encounterfrequency (text)
wavefrequency (text)
magnitude (text)

weibullc
numberofamplitudes (integer)
weibulltype (integer)
amplitudetype (integer)
amplitudemean (real)
amplitudemax (real)
amplitudemin (real)
amplitudevariance (real)
amplitudekurtosis (real)
linearregressionbeta (real)
linearregressionyintercept (real)
linearregressioncharvalue (real)
linearregressioncorrelation (real)
linearregressionparentweibulllifetimemax (real)
linearregressionextremevalue (real)
momentmethodbeta (real)
momentmethodyintercept (real)
momentmethodcharvalue (real)
momentmethodparentweibulllifetimemax (real)
momentmethodextremevalue (real)
linearregressionxnot (real)
momentmethodxnot (real)
unsortedamplitudes (text)
linearweibullxdata (text)
linearweibullydata (text)
linearregressionydata (text)
momentmethodxdata (text)
momentmethodydata (text)
momentmethodlinearydata (text)

mmmr
mean (real)
max (real)
min (real)
maxminusmean (real)
minminusmean (real)
stardarddeviation (real)
kurtosis (real)
meancrossings (integer)
maxminusmeanoverstandarddeviation (real)
range (real)
maxpeaktopeak (real)
numberofdropouts (integer)

raoc
encounterarea (real)
peakencounterfrequency (real)
peakencounteramplitude (real)
peakwavefrequency (real)
peakwaveamplitude (real)
encounterfrequency (text)
wavefrequency (text)
magnitude (text)

Page 1

Figure 26: HVDAS Database Table Descriptions

- select runname,conditionname,runtime from run where testname=’sometestname’;
  - shows run time for all runs and conditions for a given test. This can be useful for identifying conditions without enough data to be statistically useful.

- select conditionname,lastmmmanalysisdate from condition where testname=’sometestname’;
  - shows the last date when mmm analysis was performed on the conditions.

17.3 Backup

pg_dump is a program provided with Postgre SQL which can backup an existing production database without shutting it down. The output from pg_dump is on the standard output so it must be redirected to a file or to an application such as gzip. “pg_dump code653 > dbBackup.sql” will backup the database code653 to the ASCII file dbBackup.sql. This backup file contains all of the SQL queries necessary to recreate the entire database from scratch. This backup command can also be used for automatic backup using cron. The example script which follows can be placed in /etc/cron.weekly/ to create a weekly database backup with the date in the filename:

```
#!/bin/bash
su code653
date=‘date +%F‘
pg_dump code653 > "/backuppath/hvdasDBBackup-$date.sql"
gzip /backuppath/*$date*
```

18 Configuration Files and Options

HVDAS has the ability to generate example configuration files by running “hvdas -createexampleconfigs”. The example configuration files are heavily documented. Basic documentation is provided in this document, while more extensive documentation and examples are also provided in the example configuration files. Every HVDAS configuration file is similar to a standard UNIX configuration file. They all end with the extension “.conf” and can have comment lines which start with “#”. If you choose to generate example configuration files, existing configuration files will be overwritten without warning.

18.1 HVDAGlobalConfiguration.conf

The Global Configuration file is a single file containing parameters and settings which define the entire data set. The global configuration values are as follows.

18.1.1 Test Name

This variable names the current batch of analysis for use in SQL database output.
Keyword: TEST_NAME
Example: ddxmodel
Example: fredolsenseatrial

18.1.2 ODBC Data Source Name (DSN)

This variable specifies the DSN which has been set up using UNIX ODBC. This basically tells HVDAS which database server and which database to connect to.
Keyword: ODBC_DATA_SOURCE_NAME
Example: psqldsn

18.1.3 ODBC User Name

This variable specifies the user to use when connecting to the DSN.
Keyword: ODBC_USER_NAME
Example: code653

18.1.4 Data Type

This is a setting which describes the data storage format. A full description of the data types exists in the example configuration files and in the Data Formats section.

Keyword: DATA_TYPE

This setting is spelling and case sensitive, so the options must be input exactly as listed below. Choose only one.

- ASCII_COLUMNS: ASCII data stored in interleaved format (columns are channels and rows are samples). Usually tab delimited.
- BINARY_INTERLEAVED_HIGH_BYTE_FIRST: Binary real data stored in interleaved format with the high byte first.
- BINARY_INTERLEAVED_LOW_BYTE_FIRST: Binary real data stored in interleaved format with the low byte first.
- BINARY_INTERLEAVED_UNSIGNED_SHORT: Binary integer data stored in interleaved format with the low byte first.

18.1.5 Data Size

This is the number of bytes which each data point or value takes up. The default value is fixed at 4 bytes, a standard single precision floating point number. Future versions will allow for higher precision. This setting has no effect for ASCII data.

Keyword: DATA_SIZE

18.1.6 Sample Rate

The number of samples recorded per second per channel during acquisition. This number can be scaled up or down to aid in the conversion from model scale to full scale or vice verse. If you wish to scale the results, remember to change every single value to the desired scale. You will need to convert frequencies, scale factors, and speeds in multiple configuration files to the same scale.

Keyword: SAMPLE_RATE

18.1.7 Total Number of Acquired Channels

This number must include dead and/or dummy channels that were saved during acquisition. These channels may be ignored later.

Keyword: NUMBER_OF_CHANNELS

18.1.8 Header Size

The header size is an integer representing the number of bytes for a binary file header or the number of lines for an ASCII file header.

Keyword: HEADER_SIZE

18.1.9 Drift Compensation Seconds

The drift compensation seconds parameter is used to compensate for non-stationary mean shift behavior that you wish to eliminate from your data after the data has been read in and scaled. A value < 0 will not compensate, a value of 0 will subtract the entire time history's mean, and a value > 0 will subtract the local mean for the desired number of seconds.

Keyword: DRIFT_COMPENSATION_SECONDS

18.1.10 Correct Dropouts

This dropout correction parameter is used to remove dropout behavior from the time histories. This detects and corrects dropouts to exactly 0.0, maxima that are exactly 0.0, and erroneous huge positive or negative amplitudes. If the single scan delta is more than 6 * (standard deviation of the run), the point is flagged as a dropout. The dropouts are replaced with the previous data point. A value of 0 will not correct, a value of 1 will.

Keyword: CORRECT_DROP_OUTS

18.1.11 Total Number of Acquired Channels

This number must include dead and/or dummy channels that were saved during acquisition. These channels may be ignored later.

Keyword: NUMBER_OF_CHANNELS

18.1.12 Header Size

The header size is an integer representing the number of bytes for a binary file header or the number of lines for an ASCII file header.

Keyword: HEADER_SIZE

18.1.13 Test Condition List

This is a list containing the paths to each test condition directory to be analyzed in the entire data block, as well as information describing each test condition. Each entry must be on a separate line and contain every piece of information. These entries are case sensitive. If the entered directory is not found during a self-test, the directory will not be analyzed or an error will occur depending on the problem with the directory path. As with all configuration lists, the test condition list must end with ’END’ on a separate line after the last test condition entry. The user must have the proper permissions to access the data directories and data files before running the program or it will terminate in error. The test condition list is a tab-delimited matrix of information describing all of the test conditions, which lends itself to being edited with a spreadsheet application.

Keyword: TEST_CONDITION_LIST

This list contains the following parameters to describe each test condition.

CONDITION_LABEL
FILE_PREFIX
FILE_EXTENSION
FILE_ID_LOWER_BOUND
FILE_ID_UPPER_BOUND
CONDITION_START_SKIP_TIME
CONDITION_END_KEEP_TIME
SEA_STATE
SPEED_IN_KNOTS
HEADING_IN_DEGREES
CONDITION_PROBABILITY

18.1.14 Lowest and Highest Data File Indexes

The lowest and highest number following the file prefix in the file names of the data files in the condition directory.

Keyword: FILE_ID_LOWER_BOUND: FILE_ID_UPPER_BOUND

18.1.15 Data File Prefix

The part of the file name after the directory path and before the indexing number, include any characters before beginning the number including spaces and special characters.

Keyword: FILE_PREFIX

18.1.16 Data File Extension

The file extension of the data files in the corresponding condition folder.

Keyword: FILE_EXTENSION

18.1.17 Condition Label

The user defined ’name’ for the condition. This has no bearing on the actual analysis, but the program will terminate if two folders have the exact same condition label.

Keyword: CONDITION_LABEL

18.1.18 Condition Probability

The user defined probability for the condition as a percentage of the ship’s lifetime. This value is used in Weibull estimation of maximum lifetime statistics.

Keyword: CONDITION_PROBABILITY

18.1.19 Sea State

This sets the sea state of the data in the current condition. For example, in analyzing ship structure data collected in a Sea State 7, set this value to 7. If you are not analyzing ship structure data, you may enter an identifier which will define the type of test or condition you are analyzing. The combination of sea state, speed, and heading will be used to generate trend tables for all parts of HVDAS. Table 2 shows significant wave height values for different sea states according to the NATO North Atlantic Wave Height tables.

Keyword: SEA_STATE

Table 2: Sea State and Significant Wave Height

Sea State	Low Sig WH (m)	Mean Sig WH (m)	High Sig WH (m)
1	0	0.0488	0.0975
2	0.0975	0.3048	0.4999
3	0.4999	0.8839	1.2497
4	1.2497	1.8898	2.4994
5	2.4994	3.2522	3.9990
6	3.9990	4.9987	5.9985
7	5.9985	7.4981	8.9977
8	8.9977	11.4910	13.9964
9	13.9964	14.0208	

18.1.20 Speed In Knots

This sets the speed of the data in the current condition. For example, in analyzing ship structure data collected at 15 Knots, set this value to 15. Make sure that your speed matches your chosen scale. Do everything in either model or full scale for ship structures. The default value is 0.0. The speed is used by RAO analysis in the conversion from encounter to wave frequency.

Keyword: SPEED_IN_KNOTS

18.1.21 Heading in Degrees

This sets the relative heading of the data in the current condition in degrees. This software assumes the Structures Code 60 convention for relative heading. For example, 0.0 degrees is assumed to mean that the ship is heading straight into the waves, which is known as “head seas”. The default value is 0.0. The heading is used by RAO analysis in the conversion from encounter to wave frequency. The heading is also used to describe the condition in the database and in the text file output. Figure 27 shows a diagram of the Structures and Hydro Departments’ conventions for relative heading.

Keyword: HEADING_IN_DEGREES

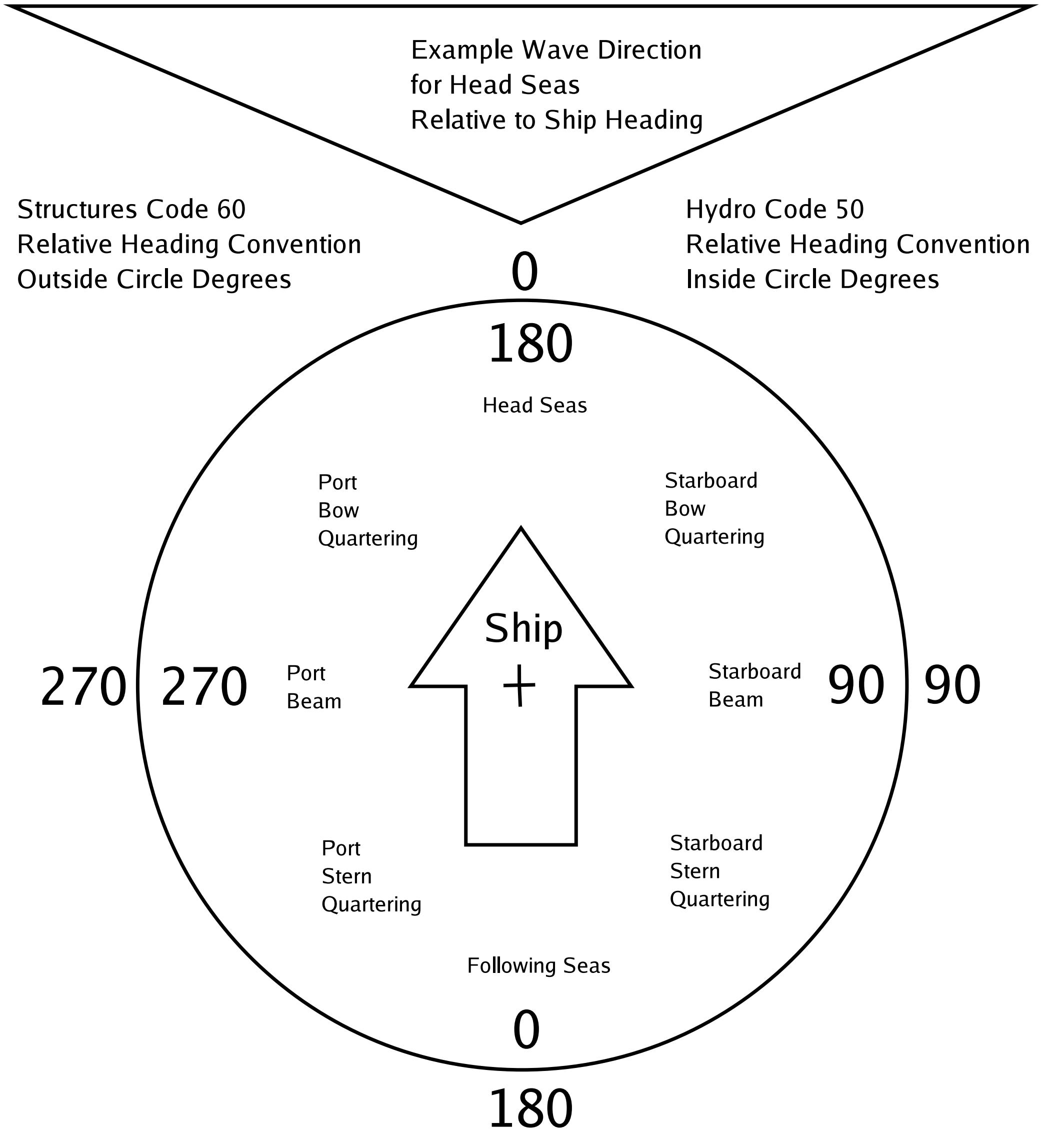


Figure 27: Relative Heading Conventions

18.1.22 Condition Start Skip Time Selection

This field represents the amount of time you would like to skip at the beginning of a condition; in seconds. This feature is very useful if the data files were collected in even intervals. For example, the RV *Triton* data conditions are comprised of data files which were recorded in 5-minute increments. The condition you may want to analyze might require you to skip the first 2 minutes of the first data file and keep the first 4 minutes of the last data file. This setting allows you to do that. Example: 'CONDITION_START_SKIP_TIME:\t60' to skip the first minute of the first data file in the condition. Leave the default value of 0.0 if you would like to analyze the entire condition worth of data. Note: In order to use this feature, you must set FILE_ID_UPPER&LOWER to the first and last files in the current condition respectively.

Key Word: CONDITION_START_SKIP_TIME

18.1.23 Condition End Keep Time Selection

This field represents the amount of time you would like to keep at the end of a condition; in seconds. This feature is very useful if the data files were collected in even intervals. For example, the RV *Triton* data conditions are comprised of data files which were recorded in 5-minute increments. The condition you may want to analyze might require you to skip the first 2 minutes of the first data file and keep the first 4 minutes of the last data file. This setting allows you to do exactly that. Example: 'CONDITION_END_KEEP_TIME:\t240' to keep the first four minutes and truncate the last minute of the last data file in the condition. Leave the default value of 0.0 if you would like to analyze the entire condition worth of data. Note: In order to use this feature, you must set FILE_ID_UPPER&LOWER to the first and last files in the current condition respectively. If you have a condition that contains only one data file, set the FILE_IDs to the same file, then the skip time will be applied followed immediately by the end keep time; in this case the end keep time should be the total time occupied by the condition or 0.0 to use all of it automatically.

Key Word: CONDITION_END_KEEP_TIME

18.1.24 Channel List

The channel list is a tab-delimited list of channel names, scale factors, zero values, and impact slopes. ALL channel names must be listed, even if the channel is empty, dead or will not be analyzed. Only 30 characters will be output to a spreadsheet, so any characters after this 30-character limit will result in an error or be truncated. The list must end with ’END’ on the line after the last channel entry.

Keyword: CHANNEL_LIST

18.1.25 ALL Scale Factors

These scale factors are used when reading the data stored in the data files into memory. If the recorded data are already in the desired units, use scale factors of 1.0. This follows the same restrictions as those for the channel names. For an empty or dead channel put, 1.00 as the scale factor. The list must end with ’END’ on the line after the last scale factor. These numbers are applied to every channel in every data file for the current condition. If no scaling is needed for selected channels, the corresponding scale factors in this file should be set to 1.

Keyword: SCALE_FACTORS

18.1.26 List of Initial Condition Zero Values

This list of coefficients is used to subtract an initial condition zero value from every channel you want to analyze. This subtraction will be performed on every filter state you choose, which means it must be used in conjunction with at least one other filter state.

Keyword: ZERO_VALUES

18.1.27 List of Impact Standard Deviation Exceedance Multipliers

This list of multipliers is used by the impact detection code in Weibull analysis. The impact standard deviation multiplier * standard deviation is compared on a scan-to-scan basis for exceedance.

Keyword: IMPACT_STDEV3-1

18.1.28 List of Impact Increasing Requirement

This flag is used by Weibull analysis to specify if each slope between three successive scans must be increasing in addition to exceeding the impact standard deviation exceedance multipliers.

Keyword: IMPACT_INCREASING_REQUIREMENT

18.1.29 List of Smoothing, Low-Pass and High-Pass Filter Coefficients

Between the filter coefficients keyword and 'END', a list of calculated filter coefficients must be copied, one coefficient per line, for each desired filter. These coefficients must be created using another program like Wave Metrics IGOR Pro IFDL and copied into the configuration file.

Keyword(s):
LOW_PASS_COEFFICIENTS,
HIGH_PASS_COEFFICIENTS,
SMOOTHED_COEFFICIENTS

18.2 HVDAMMMConfiguration.conf

The HVDAMMMConfiguration.conf file contains the parameters and settings specific to the basic MMM statistical analysis. The settings in this file are constant for the entire data block. Therefore, there need be only one copy of this file in the base directory of the analysis software. HVDAMMMConfiguration.conf includes the following parameters.

18.2.1 List of Channels to Analyze

A number in this list refers to a channel in your data file. Channel '0' (zero) refers to the first saved channel. The number of channels minus one references the last saved channel. So, if there are 16 channels, the first channel is 0 and the last is 15. The numbers can be listed in any order as long as the requested channel is less than the total number of channels, greater than or equal to 0, and exists in all data files in all conditions you wish to analyze.

Keyword: CHANNELS_TO_RUN

18.2.2 List of Filter States to Apply

The filter states are referred to using a list of integers.

- 0 - No Filtering, use raw data
- 1 - Smooth-Pass FIR Filter the raw data using the Smoothing Coefficients to remove noise
- 2 - Low-Pass FIR Filter using the Low-Pass Coefficients to remove high-frequency energy

- 3 - High-Pass FIR Filter using the High-Pass Coefficients to remove low-frequency energy
- 4 - Box Car Average smoothing of the data to remove drop-outs and noise
- 5 - Initial Conditions Zero Subtraction subtracts user defined values from each channel for every filter state requested, which means this option must be used in conjunction with at least one other filter state
- 6 - Compute Virtual Channel Expressions from raw input data before converting data files; this option must be used in conjunction with at least one other filter state and can only be used when converting data.

Use one or all of these by putting one integer per line in the configuration file. Each filter type will produce one spreadsheet of output per file and a summary of the filter condition. End the list of filters with 'END'.

Keyword: FILTERS

18.2.3 Box Car Average Window Size

This is used to select the number of data points on each side of the box car. This value can be even or odd, it does not matter.

Keyword: BOX_CAR_WINDOW_SIZE

18.2.4 Virtual Channel Expressions

This list should contain the expressions you want to use to compute virtual channels. These virtual channel expressions use postfix notation to eliminate the need to Figure out order of operations and to eliminate the need for parenthesis. Postfix notation allows the user to completely control the execution of the expression without parenthesis or brackets. It is used because of the simplicity of implementation. Virtual Channels will only be computed when converting the data using –convertdata to output ASCII data files. The newly converted ASCII data files can then be run through the MMM analysis software by changing data file descriptions and running HVDAS in MMM MPI mode just like any other data files.

Virtual Channel Support has not been heavily tested and requires precise expressions. You must follow the expression spec exactly. You cannot perform expressions on constants. Be sure to do a test channel to make sure that your expression is correct. Every complex expression is made up of simple expressions.

Key Word: ’VIRTUAL_CHANNEL_EXPRESSIONS’

```
//////////////////////////////////
//expression format spec from the source code
//operand d, which can be a constant(1, 1.5, 45.987, ...)
//or a channel (c0, c1, c2,...)
//operator r, which can be (+, -, *, /, and ^ for simple expressions)
//or (+, -, *, /, and sqrt) for complex expressions
//space s
//tab t
//comma ,
//semicolon ;
///////////////////////////
//Simple Expression
//d,sd,sd,tr,sr
//or
//Complex Expression composed of Simple Expressions
//d,sd,sd,tr,sr;sd,sd,sd,tr,sr;tEOsr
///////////////////////////
```

Example simple expression $(c0+c1+c2)$ should be input as
’c2, c1, c0,<tab>+, +’

Example simple expression $(c0+2+1.1)$ should be input as
’1.1, 2, c0,<tab>+, +’

Example simple expression $\frac{c2\times5.5}{2.2}$ should be input as
’2.2, 5.5, c2,<tab>*, /’

Example complex expression $(c11+3.3)\times(c9+3.3)$ should be input as
’3.3, c9,<tab>+; 3.3, c11,<tab>+;<tab>EO *’

Example complex expression $\sqrt{\frac{c2\times c1}{c4\times5.5}}$ should be input as
’5.5, c4,<tab>*; c1, c2,<tab>*;<tab>EO /, sqrt’

Example complex expression $\frac{c3}{1.1}+\frac{c0}{2.2}$ should be input as
’2.2, c0,<tab>/; 1.1, c3,<tab>/;<tab>EO +’

Example complex expression $\frac{\frac{\frac{c3+1.1}{c0+1.5}}{c5+2.2}}{c9^{3.3}}$ should be input as
’3.3, c9,<tab>^; 2.2, c5,<tab>+; 1.5, c0,<tab>+; 1.1, c3,<tab>+;<tab>EO /, /, /’

Example complex expression $\frac{Econst}{2}*\left[\frac{c1+c3}{1-Vconst}\pm\frac{\sqrt{2}}{1+Vconst}*\sqrt{(c1-c2)^2+(c2-c3)^2}\right]$ should be input as

’(1-Vconst)value, c1, c3,<tab>+, /; (sqrt(2)/(1+Vconst)^2value, 2, c2, c1,<tab>-, ^, *; (sqrt(2)/(1+Vconst)^2value, 2, c3, c2,<tab>-, ^, *;<tab>EO +, sqrt, +’ and as

’(1-Vconst)value, c1, c3,<tab>+, /; (sqrt(2)/(1+Vconst)^2value, 2, c2, c1,<tab>-, ^, *; (sqrt(2)/(1+Vconst)^2value, 2, c3, c2,<tab>-, ^, *;<tab>EO +, sqrt, -’

18.2.5 Output Path

The output path contains the complete path to the folder where the user wants the results saved. The output path must begin and end with a ’/’. This folder will be appended if it already exists. Any existing sub-folder of the same name as a program-created folder will be overwritten. For this reason, it is recommended that the folder named here be empty prior to running the program. This will allow the program to build the file structure without incident. The user must have the proper permissions to access the output directory before running the program or it will terminate in error.

The user may optionally select ’*’ as the output path when converting data files. This will write output files into the same directories as the input data files. This option is very useful when analyzing converted data because the configuration files to this software will not need much modification.

Keyword: OUTPUT_PATH

18.3 HVDAPSDConfiguration.conf

The HVDAPSDConfiguration.conf file contains the parameters and settings specific to the PSD frequency domain analysis. The settings in this file are constant for the entire data block. Therefore, there need be only one copy of this file in the base directory of the analysis software. HVDAPSDConfiguration.conf includes the following parameters.

18.3.1 List of Channels to Analyze

A number in this list refers to a channel in your data file. Channel ’0’ (zero) refers to the first saved channel. The number of channels minus one references the last saved channel. So, if there are 16 channels, the first channel is 0 and the last is 15. The numbers can be listed in any order as long as the requested channel is less than the total number of channels, greater than or equal to 0, and exists in all data files in all conditions you wish to analyze.

Keyword: CHANNELS_TO_RUN

18.3.2 List of Filters to Apply

The filters are referred to using a list of integers.

- 0 - No Filtering, use raw data
- 1 - Low-Pass Filter using the Low-Pass Coefficients to remove high-frequency energy

Use only one of these two possible filter states at a time by putting only one integer in the configuration file. Each filter type will produce one spreadsheet of output per file and a summary of the filter condition. End the one line filter list with 'END'.

Keyword: FILTERS

18.3.3 Output Path

The output path contains the complete path to the folder where the user wants the results saved. The output path must begin and end with a '/'. This folder will be appended if it already exists. Any existing sub-folder of the same name as a program-created folder will be overwritten. For this reason, it is recommended that the folder named here be empty prior to running the program. This will allow the program to build the file structure without incident. The user must have the proper permissions to access the output directory before running the program or it will terminate in error.

Keyword: OUTPUT_PATH

18.3.4 Highest Frequency of Interest Selection

The highest frequency of interest (HFI) is used to scale the output PSD. The PSD can have frequencies beyond the HFI, but you should be certain of the samples per cycle at the HFI because of the OVER_SAMPLING input. Note: The frequencies referenced here are relative to the SAMPLE_RATE in HVDAGlobalConfiguration.conf. Choose either model- or full-scale units by changing the sample rate, scale factors, and these frequencies.

Keyword: HIGHEST_FREQUENCY_OF_INTEREST

18.3.5 Over Sampling Selection

The over sampling (OS) is used to scale the output PSD. The HFI and OS determine the re-sampling, i-skip, factor needed to give OS samples per cycle at the HFI in the output PSD.

Keyword: OVER_SAMPLING

18.3.6 Target Delta Frequency Selection

The target delta frequency (TDF) is used to compute the number of points needed to calculate the PSD to arrive at the closest possible actual delta frequency to the TDF. You must determine, by hand, a good TDF for your data set. You must decide on the best trade-off between accuracy and resolution. You may want to choose a TDF that will cause a PSD data point to be calculated at a known frequency, such as the frequency when wave length is equal to ship length.

Note: If the HFI, OS, TDF combination requires a large number of data points, some files may not have enough data. In this case, the input data are duplicated until enough data points exist to perform the desired computations. The effect of this solution is minimal on the PSD because the statistics of the input waves are not changed. The resolution you get from a duplicated data PSD is artificial. Its added resolution comes from the assumption that the data run would have continued with identical frequency content for as long as needed. This assumption affects confidence bounds statistics. This is sometimes necessary to get every single PSD from each condition to have the same frequencies, which makes summarizing and reporting much easier for the user.

Keyword: TARGET_DELTA_FREQUENCY

18.4 HVDARAOConfiguration.conf

The HVDARAOConfiguration.conf file contains the parameters and settings specific to the RAO frequency domain analysis. The settings in this file are constant for the entire data block. Therefore, there need be only one copy of this file in the base directory of the analysis software. HVDARAOConfiguration.conf includes the following parameters.

18.4.1 List of Channels to Analyze

A number in this list refers to a channel in your data file. Channel ’0’ (zero) refers to the first saved channel. The number of channels minus one references the last saved channel. So, if there are 16 channels, the first channel is 0 and the last is 15. The numbers can be listed in any order as long as the requested channel is less than the total number of channels, greater than or equal to 0, and exists in all data files in all conditions you wish to analyze.

Keyword: CHANNELS_TO_RUN

18.4.2 List of Filters to Apply

The filters are referred to using a list of integers.

- 0 - No Filtering, use raw data
- 1 - Low-Pass Filter using the Low-Pass Coefficients to remove high-frequency energy

Use only one of these two possible filter states at a time by putting only one integer in the configuration file. Each filter type will produce one spreadsheet of output per file and a summary of the filter condition. End the one line filter list with ’END’.

Keyword: FILTERS

18.4.3 List of RAO Filters to Apply

The filters are referred to using a list of integers.

- 0 - No Filtering, use raw RAO
- 1 - Low-Pass Filter the output RAO using the RAO Pass Coefficients to remove high-frequency oscillations in high-resolution RAOs

Use only one of these two possible filter states at a time by putting only one integer in the configuration file. Each filter type will produce one spreadsheet of output per file and a summary of the filter condition. End the one line filter list with ’END’.

Keyword: RAOFILTERS

18.4.4 List of RAO Pass Coefficients

Between the RAO filter coefficients keyword and ’END’, a list of calculated filter coefficients must be copied, one coefficient per line, for each activated filter. These coefficients must be created using another program like Wave Metrics IGOR Pro IFDL and copied into the configuration file.

Keyword(s): RAO_PASS_COEFFICIENTS

18.4.5 Output Path

The output path contains the complete path to the folder where the user wants the results saved. The output path must begin and end with a ’/’. This folder will be appended if it already exists. Any existing sub-folder of the same name as a program-created folder will be overwritten. For this reason, it is recommended that the folder named here be empty prior to running the program. This will allow the program to build the file structure without incident. The user must have the proper permissions to access the output directory before running the program or it will terminate in error.

Keyword: OUTPUT_PATH

18.4.6 Highest Frequency of Interest Selection

The highest frequency of interest (HFI) is used to scale the output RAO. The RAO can have frequencies beyond the HFI, but you should be certain of the samples per cycle at the HFI because of the OVER_SAMPLING input. Note: The frequencies referenced here are relative to the SAMPLE_RATE in HVDAGlobalConfiguration.conf. Choose either model- or full-scale units by changing the sample rate, scale factors, and these frequencies.

Keyword: HIGHEST_FREQUENCY_OF_INTEREST

18.4.7 Over Sampling Selection

The over sampling (OS) is used to scale the output RAO. The HFI and OS determine the re-sampling, i-skip, factor needed to give OS samples per cycle at the HFI in the output RAO.

Keyword: OVER_SAMPLING

18.4.8 Target Delta Frequency Selection

The target delta frequency (TDF) is used to compute the number of points needed to calculate the PSD to arrive at the closest possible actual delta frequency to the TDF. You must determine, by hand, a good TDF for your data set. You must decide on the best trade-off between accuracy and resolution. You may want to choose a TDF that will cause a PSD data point to be calculated at a known frequency, such as the frequency when wave length is equal to ship length.

Note: If the HFI, OS, TDF combination requires a large number of data points, some files may not have enough data. In this case, the input data are duplicated until enough data points exist to perform the desired computations. The effect of this solution is minimal on the PSD because the statistics of the input wave are not changed. The resolution you get from a duplicated data RAO is artificial. Its added resolution comes from the assumption that the data run would have continued with identical frequency content for as long as needed. This assumption affects confidence bounds statistics. This is sometimes necessary to get every single RAO from each condition to have the same frequencies, which makes summarizing and reporting much easier for the user.

Keyword: TARGET_DELTA_FREQUENCY

18.4.9 Divisor Data Channel Selection

The divisor data channel (DDC) is used to divide every PSD calculated for a given data file. Dividing a PSD by the PSD of the DDC creates a transfer function called an RAO. In ship structures, if you divide the PSD(vertical bending) / PSD(wave height) then the RAO output is in terms of bending response per foot of input wave height. The units of the transfer function RAO depend on the units of the input data and are not limited to ship structure data.

Keyword: DIVISOR_DATA_CHANNEL

18.4.10 Convert from Encounter to Wave Frequency Selection

Enter a ’1’ to convert from encounter frequency to wave frequency or a ’0’ to skip conversion. This enables the use of speed and heading from each condition’s local configuration. A speed of 0.0 with a heading of 0.0 will result in the wave frequency equal to the encounter frequency.

Keyword: CONVERT_FROM_ENCOUNTER_TO_WAVE_FREQUENCY

18.5 HVDAWeibullConfiguration.conf

The HVDAWeibullConfiguration.conf file contains the parameters and settings specific to the Weibull distribution analysis. The settings in this file are constant for the entire data block. Therefore, there need be only one copy of this file in the base directory of the analysis software. HVDAWeibullConfiguration.conf includes the following parameters.

18.5.1 List of Channels to Analyze

A number in this list refers to a channel in your data file. Channel ’0’ (zero) refers to the first saved channel. The number of channels minus one references the last saved channel. So, if there are 16 channels, the first channel is 0 and the last is 15. The numbers can be listed in any order as long as the requested channel is less than the total number of channels, greater than or equal to 0, and exists in all data files in all conditions you wish to analyze.

Keyword: CHANNELS_TO_RUN

18.5.2 List of Filters to Apply

The filters are referred to using a list of integers.

- 0 - No Filtering, use raw data
- 1 - Low-Pass Filter using the Low-Pass Coefficients to remove high-frequency energy
- 2 - High-Pass Filter using the High-Pass Coefficients to remove low-frequency energy
- 3 - Already Filtered data

Use only one of these possible filter states at a time by putting only one integer in the configuration file. Each filter type will produce one spreadsheet of output per file and a summary of the filter condition. End the one line filter list with ’END’.

Keyword: FILTERS

18.5.3 Output Path

The output path contains the complete path to the folder where the user wants the results saved. The output path must begin and end with a '/'. This folder will be appended if it already exists. Any existing sub-folder of the same name as a program-created folder will be overwritten. For this reason it is recommended that the folder named here be empty prior to running the program. This will allow the program to build the file structure without incident. The user must have the proper permissions to access the output directory before running the program or it will terminate in error.

Keyword: OUTPUT_PATH

18.5.4 Weibull Calculation Type

There are currently two different Weibull calculation methods to choose from:

- 1 - 2-Parameter Linear Regression and 2-Parameter Moment Method, or
- 3 - 3-Parameter Linear Regression and 3-Parameter Moment Method.

Keyword: WEIBULL_TYPE

18.5.5 Weibull Amplitude Type

There are many types of amplitudes you may use Weibull analysis with:

- 0 - Maximums,
- 1 - Minimums,
- 2 - Maximums-Mean,
- 3 - Minimums-Mean,
- 4 - Ranges,
- 5 - Raw,
- 6 - Whipping Maxes,
- 7 - Whipping Mins,

- 8 - Whipping Ranges,
- 9 - Development Max-Mean,
- 10 - Development Min-Mean,
- 11 - Development Range,
- 12 - Max/Min Ratio,
- 13 - Combined Whipping Maxes,
- 14 - Combined Whipping Mins,
- 15 - Combined Whipping Ranges,
- 16 - LGH Impact Max-Localmean

The Raw amplitude type allows you to provide your own set of amplitudes to HVDAS.

Keyword: AMPLITUDE_TYPE

18.5.6 Beta Lower Bound

This is the lower search bound when iteratively solving for beta, the Weibull shape parameter, using the moment method calculation.

Keyword: BETA_LOWER_BOUND

18.5.7 Beta Upper Bound

This is the upper search bound when iteratively solving for beta using the moment method calculation.

Keyword: BETA_UPPER_BOUND

18.5.8 Beta Step Size

This is the beta increment size when iteratively solving for beta using the moment method calculation.

Keyword: BETA_STEP_SIZE

18.5.9 Ship Lifetime

This specifies the expected, designed, or desired ship lifetime in years, which is used in maximum lifetime estimation.

Keyword: SHIP_LIFETIME

18.5.10 Extreme F

This specifies the exceedance probability value to use when computing extreme value analysis of the parent Weibull distribution.

Keyword: EXTREME_F

18.5.11 Blanket Amplitude Cutoff

This selects a Weibull threshold which will cut off detected amplitudes that are below the cutoff value for every filter state and amplitude type.

Key Word: BLANKET_AMPLITUDE_CUTOFF

18.5.12 XNot Percent Step Size

This selects a step size used in xnot calculation as a percentage of the smallest detected amplitude that is above the cutoff. This setting is only used during 3-parameter analysis.

Key Word: XNOT_PERCENT_STEP_SIZE

18.5.13 Individual Amplitude Cutoff Lists

These cutoff lists are used to cutoff amplitudes below a threshold value for each amplitude type and filter state. The lists must specify a cutoff value for every channel in the data file even if it is not going to be analyzed. The list must end with END. The complete list is in the example Weibull configuration file.

Key Words: FILTERTYPE_AMPLITUDETYPE_AMPLITUDE_CUTOFFS

19 Using MPI

This parallelized computer software uses Message Passing Interface (MPI) to communicate between the various computers in a cluster of computers. The user must be familiar with a few simple Linux command line calls that enable and maintain the MPI communication interface. This documentation is written for the Local Area Multicomputer (LAM) implementation of MPI. The LAM-MPI communication software is what allows the Linux cluster to perform calculations in parallel.

19.1 lamhosts

’lamhosts’ is a text file containing the node names for all of the nodes available to the MPI software. If a node is not listed in this file, you cannot use it for calculations. The user can append nodes to this file from the user’s home directory by typing "vi lamhosts" from the console or "nedit lamhosts" from an X Windows terminal. This brings up an editor. For ’vi’ usage information, type "man vi" at the command line. The order of the node names in this file will determine the node’s number in the ’mpirun’ call. The server node must be listed first or the program will fail. The ’lamhosts’ file can be found or created in the user’s home directory and usually consists of a list of IP addresses or hostnames.

19.2 recon

The ’recon’ command checks the connections to the nodes in the ’lamhosts’ file without actually bringing up the communications interface. The ’recon’ command should be run before enabling the MPI interface. Type "recon -v lamhostsfile".

19.3 lamboot

’lamboot’ is the command that actually enables MPI communication between computers. The number assigned to each node can be determined by looking at the output from this command. Type "lamboot -v lamhostsfile" when the ’lamhosts’ file is correct and the ’recon’ command has completed successfully.

19.4 mpirun

’mpirun’ is the command that starts a program which uses MPI in parallel. Precede the program name with "mpirun -v" followed by the first and last nodes in the ’lamhosts’ file (for 4 nodes use n0-3, this uses nodes 0,1,2,3 in the ’lamhosts’ file). The ’mpirun’ command can use any node listed in the ’lamhosts’ file. It is not required to use all the nodes listed in ’lamhosts’. A sample call to MPI using 5 nodes would be "mpirun -v n0-4 program program_args". The ’mpirun’ command should be used in the folder containing the program or configuration files to be run. Another sample call to MPI using all available nodes would be “mpirun -v C program program_args"; the “C” flag tells mpirun to use all available processors.

19.5 lamclean

’lamclean’ resets MPI communication and stops any runaway processes that are trying to use MPI. This should be done between large data runs or after the program has terminated in error. This can be done from any location assuming that the ’lamboot’ command has already been used and MPI communication is still active. Type "lamclean -v" at the command prompt.

19.6 wipe or lamhalt

’wipe’ or ’lamhalt’ stops MPI communication when the program is complete and no further analysis is desired. Type "wipe -v lamhostsfile" or “lamhalt -v lamhostsfile” at the command prompt.

20 Using HVDAS

20.1 General Setup

1. Make sure all of your data are backed up before trying to setup anything.

2. Make sure all of your data are backed up before trying to analyze anything.

3. Triple check your backups before moving on. You have been warned.

4. Use Wave Metrics IGOR Pro Filter Design Laboratory, or similar software package, to design appropriate finite impulse response (FIR) filters for your specific data. You may create up to 3 different FIR filters. You may create a smoothing-pass filter to remove noise, a low-pass filter to remove high-frequency energy, and a high-pass filter to remove low-frequency energy. You may also choose to box car average your data to remove drop-outs. Look at your data in a package like IGOR to make sure that your settings and filters make sense with your current data set and desired results.

5. Save the FIR filter coefficients which describe your filters in ASCII text files. These coefficients will be pasted into the proper configuration files later.

6. Locate the file ’HVDASuite-YYYY-MM-DD.tar.gz’ or ’HVDASuite.tar.gz’. An ’_i386’ in the file name indicates that this binary executable program was compiled and built for Intel i386-based computer architecture. Copy this file to a globally accessible working directory like ’/Server/Analysis/’. Change the working directory to be that directory. Type, "tar xvzf HVDA*.tar.gz", to unpack the archive file. This will create a sub-folder, called ’HVDA-Suite’, which will contain the executable program ’hvdas’ and documentation.

7. Type "./" to reference the current directory in most Unix operating systems. You will need to add the ’./’ before ’hvdas’ if you only unpacked the archive and did not install the executable code into a system-wide accessible location.

8. Type "hvdas –help" or “hvdas -help” for usage information.

9. Make sure the data on the analysis computer are stored in a globally accessible location if you are going to run the analysis on a cluster. It is generally a

good idea to put all of the data conditions in a small number of well-named locations. See section ’Input’ for an example data layout. The best approach is to keep it simple. Each separate condition must be in a separate sub-folder with a unique name. The simplest layout is to have one main data directory which contains one sub-folder for each of your separate conditions.

10. Create the example configuration files ’HVDAGlobalConfiguration.conf’, ’HVDAMMMConfiguration.conf’, ’HVDAPSDConfiguration.conf’, ’HVDARAOConfiguration.conf’, and ’HVDAWeibullConfiguration.conf’ by typing “hvdas -createexampleconfigs”.

11. Modify the ’HVDAGlobalConfiguration.conf’, ’HVDAMMMConfiguration.conf’, ’HVDAPSDConfiguration.conf’, ’HVDARAOConfiguration.conf’, and ’HVDAWeibullConfiguration.conf’ configuration files. The example configuration files are well documented.

12. Add an entry to ’HVDAGlobalConfiguration.conf’ in TEST_CONDITION_LIST for test condition folder. You will need to change each entry to reflect the data contained in each condition directory. Make sure you change CONDITION_LABEL. Each condition label must be unique or the results will be invalid. A well-designed template will not need to be modified much from condition to condition if the data is similar.

13. Be sure to paste all of the FIR filter coefficients and initial condition zero values into the ’HVDAGlobalConfiguration.conf’ file. Also, be sure to get all of the data and header formatting information, channel names, and scale factors correct or the results will be wrong.

20.2 Command Line Options

HVDAS has several command line options. These options are very useful for scripting large sets of data to be run over an extended period of time. The options and flags can be input in any order.

20.2.1 –help or -help

’hvdas –help’ or ’hvdas -help’ displays usage information and command line options to the standard output.

20.2.2 -convertdata

'hvdas -convertdata' specifies that the software should convert and modify data to ASCII. This will output converted or virtual channels to new ASCII files. Old files will be overwritten without additional warning.

20.2.3 -createexampleconfigs

'hvdas -createexampleconfigs' specifies that the software should create the example configuration files necessary to run all of the desired analysis programs. It will create HVDAGlobalConfiguration.conf, HVDAMMMConfiguration.conf, HVDAPSDConfiguration.conf, HVDARAOConfiguration.conf, and HVDAWeibull-Configuration.conf. Old configuration files will be overwritten if you type 'yes' after the warning.

20.2.4 -createhvdastablesindb

'hvdas -createexampleconfigs' specifies that the software should create the necessary tables in the database. This will try to create test, channel, filter, condition, run, mmmr, mmmc, psdc, raoc, and weibullc tables in the database.

20.2.5 -drophvdastablesfromdb

'hvdas -drophvdastablesfromdb' specifies that hvdas should drop all tables, results, and everything from the database. This option is highly destructive and will delete all calculated results after a warning and confirmation. This is useful if you need to delete everything HVDAS has put into the database.

20.2.6 -deleteentiretestfromdb

'hvdas -deleteentiretestfromdb' will delete all results and table entries for the test specified in the global configuration. This option does not drop tables from the database; it deletes results for the desired test only. This is useful if there is a major problem with the current test analysis and it must be reanalyzed. This option is highly destructive and will delete some results after a warning and confirmation.

20.2.7 -setupdbfromglobal

’hvdas -setupdbfromglobal’ will insert and update the current database to include information in the global configuration file. This option populates existing HVDAS tables to include information about the current test analysis from the global configuration file. If conditions or channels are renamed, added, or changed in any way this option must be run again.

20.2.8 -mpi

’hvdas -mpi’ specifies that the software should run in MPI mode. This tells the HVDAS program to run analysis in parallel. The most basic way to start HVDAS on a cluster is ’mpirun -v C hvdas -mpi <options>’ to run this program on all available nodes; or ’mpirun -v -np 2 n0 hvdas -mpi <options>’ to run the client and server portions of HVDAS locally, on one singular node with one or more processors. This option must be used with only one of the analysis types listed below at any given time: -mmm, -psd, -rao, or -weibull.

20.2.9 -mmm

’hvdas -mmm’ specifies that HVDAS should perform MMM analysis. This option must be used in conjunction with -mpi.

20.2.10 -psd

’hvdas -psd’ specifies that HVDAS should perform PSD analysis. This option must be used in conjunction with -mpi.

20.2.11 -rao

’hvdas -rao specifies that HVDAS should perform RAO analysis. This option must be used in conjunction with -mpi.

20.2.12 -weibull

’hvdas -weibull specifies that HVDAS should perform Weibull analysis. This option must be used in conjunction with -mpi.

20.2.13 -version

’hvdas -version’ will display version information.

20.2.14 -debugoutput

’-debugoutput’ is a flag which tells HVDAS to output extra textual information to the standard output, for debugging purposes. The default for this option is false.

20.2.15 -hvdaglobalconfigurationname

’-hvdaglobalconfigurationname’ lets you specify the full path name of the global configuration file. The default is “./HVDAGlobalConfiguration.conf”.

20.2.16 -hvdammmconfigurationname

’-hvdammmconfigurationname’ lets you specify the full path name of the MMM configuration file. The default is “./HVDAMMMConfiguration.conf”.

20.2.17 -hvdapsdconfigurationname

’-hvdapsdconfigurationname’ lets you specify the full path name of the PSD configuration file. The default is “./HVDAPSDConfiguration.conf”.

20.2.18 -hvdaraoconfigurationname

’-hvdaraoconfigurationname’ lets you specify the full path name of the RAO configuration file. The default is “./HVDARAOConfiguration.conf”.

20.2.19 -hvdaweibullconfigurationname

’-hvdaweibullconfigurationname’ lets you specify the full path name of the Weibull configuration file. The default is “./HVDAWeibullConfiguration.conf”.

20.2.20 -logfilename

’-logfilename’ lets you specify the full path name of the log file you would like to output. The default is “./HVDASLog.log”.

20.2.21 -nofilestats

’-nofilestats’ tells HVDAS that it should not output results files for every single input file. This flag will restrict output to condition summary results only. This option is useful if you have a lot of data files in each condition and you do not care about the individual data files. The default is false, which means that HVDAS will output individual file and condition results.

20.2.22 -nosummarystats

’-nosummarystats’ tells HVDAS that it should not output results files for the condition summary. This flag will restrict output to individual file results only. This option is useful if your condition directories have only one data file. The default is false, which means that HVDAS will output individual file and condition results.

20.2.23 -notextfiles

’-notextfiles’ tells HVDAS that it should not output any ASCII text results files. This option would allow the database to contain all calculated results. The default is false, which allows text file output.

20.2.24 -nodatabase

’-nodatabase’ tells HVDAS that it should not output any SQL results to the database. This option would allow the ASCII text files to contain all calculated results. The default is false, which allows database output.

20.2.25 -weibullfilterstate

’-weibullfilterstate’ lets you specify the filter to use for Weibull analysis and will override the setting in the HVDAWeibullConfiguration.conf configuration file.

20.2.26 -weibullamplitudetype

’-weibullamplitudetype’ lets you specify the set of amplitudes to use for Weibull analysis and will override the setting in the HVDAWeibullConfiguration.conf configuration file.

20.3 Cluster Execution

1. Type: "hvdas –help" for usage information.
2. Warning: Old results in the results output path will be overwritten without warning!
3. Triple check your results output path to make sure you are not going to overwrite anything important! You have been warned.
4. Go to the home directory "cd ~".
5. Edit ’lamhosts’ if necessary "vi lamhosts".
6. Recon ’lamhosts’ Type "recon -v lamhosts" to see if the cluster is bootable.
7. If ’recon’ is successful, bring up MPI using "lamboot -v lamhosts".
8. If ’lamboot’ is successful, go to the analysis directory where the HVDAS configuration files are located. For example: Type “cd /server/test/”.
9. If all the configuration files are set up and ’lamboot’ worked, run the program: "mpirun -v C hvdas -mpi <options>", which will actually perform the desired analysis. Adjust the “C” to “n0-5” to use a smaller subset of available nodes. In one example case, 6 computers are in the ’lamhosts’ file, 0 is the server and 1 through 5 are the compute nodes, so specifying “C” is exactly the same as specifying “n0-5”. You may optionally run ’hvdas -convertdata’ to smooth, filter, scale, zero, or reformat your input data files and save them in new ASCII tab-delimited text data files.
10. To perform MMM analysis run “mpirun -v C hvdas -mpi -mmm -logfilename mmmlog.log”.
11. To perform PSD analysis run “mpirun -v C hvdas -mpi -psd -logfilename psdlog.log”.
12. To perform RAO analysis run “mpirun -v C hvdas -mpi -rao -logfilename raolog.log”.
13. To perform Weibull analysis run “mpirun -v C hvdas -mpi -weibull -logfilename weibulllog.log”.
14. If errors are reported, fix and rerun.

15. When program has completed, clean MPI "lamclean -v".
16. The MMM analysis results should be in the '/server/test/HVDAMMMResults/' folder in the location chosen in the 'HVDAMMMConfiguration.conf' file. The converted or modified data files should be in the '/server/test/HVDAConvertedOutputData/' folder in the same location chosen in the 'HVDAMMMConfiguration.conf' file.
17. If another analysis is desired, change the necessary settings and rerun, otherwise shutdown MPI from the home directory using "wipe -v lamhosts".
18. If everything worked properly, you should see messages telling you that everything finished properly. You should see the elapsed time and the location of the results.
19. A log file called 'HVDASLog.log' should have been generated unless the -logfilename option was used. This log file contains debugging information, analysis information, progress information, and error information. It is a good idea to check this file to make sure everything went as planned and to check for errors.
20. Check the results output folder. The 'HVDA*Results/' folder should contain one sub-folder for each input condition. Check these folders for individual output files and condition summaries. The output is standard ASCII tab delimited text, which can be imported into almost any spreadsheet and by any ASCII text editor.
21. It is a good idea to manually check the results for three or more channels, two or more filter states, in three or more conditions to ensure good output.
22. If you obtained good results, back them up.
23. Back up your results. You have been warned.

20.4 Workstation (Single Processor) Execution

1. Type: "hvdas –help" for usage information.
2. Warning: Old results in the results output path will be overwritten without warning!

3. Triple check your results output path to make sure you are not going to overwrite anything important! You have been warned.

4. Go to the home directory "cd ~".

5. Edit ’lamhosts’ if necessary "vi lamhosts". You need at least the single local processor which is going to be used.

6. Recon ’lamhosts’ Type "recon -v lamhosts" to see if the cluster is boot-able.

7. If ’recon’ is successful, bring up MPI using "lamboot -v lamhosts".

8. If ’lamboot’ is successful, go to the analysis directory where the HVDAS configuration files are located. For example: Type “cd /server/test/”.

9. If all the configuration files are set up and ’lamboot’ worked, run the program: "mpirun -v -np 2 n0 hvdas -mpi <options>", which will actually perform the desired analysis. This call will run two copies (one server and one client) of HVDAS locally on node0, which may be the only node listed in the lamhosts file. You may optionally run ’hvdas -convertdata’ to smooth, filter, scale, zero, or reformat your input data files and save them in new ASCII tab-delimited text data files.

10. To perform MMM analysis run “mpirun -v -np 2 n0 hvdas -mpi -mmm -logfilename mmmlog.log”.

11. To perform PSD analysis run “mpirun -v -np 2 n0 hvdas -mpi -psd -logfilename psdlog.log”.

12. To perform RAO analysis run “mpirun -v -np 2 n0 hvdas -mpi -rao -logfilename raolog.log”.

13. To perform Weibull analysis run “mpirun -v -np 2 n0 hvdas -mpi -weibull -logfilename weibulllog.log”.

14. If errors are reported, fix and rerun.

15. When program has completed, clean MPI "lamclean -v".

16. The MMM analysis results should be in the ’/server/test/HVDAMMMResults/’ folder in the location chosen in the ’HVDAMMMConfiguration.conf’ file. The converted or modified data files should be in the

’/server/test/HVDAConvertedOutputData/’ folder in the same location chosen in the ’HVDAMMMConfiguration.conf’ file.

17. If another analysis is desired, change the necessary settings and rerun, otherwise shutdown MPI from the home directory using "wipe -v lamhosts".

18. If everything worked properly, you should see messages telling you that everything finished properly. You should see the elapsed time and the location of the results.

19. A log file called ’HVDASLog.log’ should have been generated unless the -logfilename option was used. This log file contains debugging information, analysis information, progress information, and error information. It is a good idea to check this file to make sure everything went as planned and to check for errors.

20. Check the results output folder. The ’HVDA*Results/’ folder should contain one sub-folder for each input condition. Check these folders for individual output files and condition summaries. The output is standard ASCII tab delimited text, which can be imported into almost any spreadsheet and by any ASCII text editor.

21. It is a good idea to manually check the results for three or more channels, two or more filter states, in three or more conditions to ensure good output.

22. If you obtained good results, back them up.

23. Back up your results. You have been warned.

20.5 Viewing Results

20.5.1 Importing ASCII Text Files Into a Spreadsheet

A great way to look at the result files from HVDAS is with a spreadsheet program such as Microsoft Excel, Sun Microsystems Star Office, or Open Office. Try to open the result file as an ASCII text file or comma separated value (CSV) text file. Most spreadsheet programs will allow you to select some options to describe the text file. Make sure that you select a tab character as the separating or delimiting character if it is not the default.

20.5.2 Printing ASCII Text Files From Linux/UNIX

1. Go to the directory containing the data to be printed.
2. Type: "a2ps –landscape –columns 1 –rows 1 -f 8 -d Output.txt" or open the results in a spreadsheet program and print from there.

- Option ’–landscape’ Sets the orientation of the printed page
- Option ’–columns’ Sets the number of pages to print vertically per-page
- Option ’–rows’ Sets the number of pages to print horizontally per-page
- Option ’-f’ Sets the font size, 8 in this example
- Option ’-d’ Precedes the file name to print
- The file name can include *’s to print multiple files at once.
- The output results files are tab delimited and may be imported into various spreadsheets easily.

20.5.3 Using gnuplot to View Data

1. Go to the directory containing the data to be printed.
2. Type: “gnuplot” to start the gnuplot command interpreter or type: “gnuplot scriptfile.gnuplot” to have gnuplot read commands from a text file.
3. gnuplot can read ASCII text data files if all of the header lines begin with “#” comment characters
4. Command: “plot "Data_mod_0.txt" using 1 title ’ASDSD19S38ch 1’ with lines, "smooth1.txt" using 1 with lines lw 3“ will plot channel 1 from Data_mod_0.txt using lines, and channel 1 from smooth1.txt using lines with line width of 3.
5. Command: “set out "pstest.ps"” will set the output file to “pstest.ps”
6. Command: “set size 1.0, 0.5” will set the size of the output plot
7. Command: “set terminal postscript landscape enhanced "Helvetica" 12” will set the type of output, font, and font size

8. Steps to print out plots on paper:

- set size 1.0, 0.5
- set terminal postscript landscape enhanced "Helvetica" 12
- !rm -f pstest.ps
- set out "pstest.ps"
- plot "Data_mod_0.txt" using 1 title ’ASDSD19S38ch 1’ with lines, "smooth1.txt" using 1 with lines lw 3
- !lpr pstest.ps

References

[1] Hardin, Jay C. “Introduction to Time Series Analysis.” NASA Reference Publication 1145. NASA Publishing, 1986.

[2] Press, William H., Saul A. Teukolsky, William T. Vetterling, Brian P. Flannery. “The Art of Scientific Computing.” Numerical Recipes in C Second Edition. New York: Cambridge University Press, 1992.

[3] IGOR Pro Team. “Technical Note 006.” Wave Metrics IGOR Pro 3.15 Manual. 2000.

[4] Weast, Robert C., Samuel M. Selby. CRC Handbook of Math 4TH Edition. Cleveland: The Chemical Rubber Co., 1970.

[5] Abramowitz, Milton, Irene A. Stegun. Handbook of Mathematical Functions. New York: Dover Publications, 1972.

[6] Lewis, Richard R. “An Assessment of Statistical Techniques and Current Assumptions for Analysis of Hull Girder Bending Response Time Histories.” Survivability, Structures and Materials Directorate Research and Development Report. Bethesda: Naval Surface Warfare Center Carderock Division, 1995.

[7] Schwartz, David A., Randall R. Judd, William J. Harrod, Dwight P. Manley. VSIPL Spec 1.0. Georgia: Georgia Tech Research Corporation, 2000.

[8] Message Passing Interface Forum. MPI-2: Extensions to the Message-Passing Interface. Knoxville: University of Tennessee, 1997.

[9] Ohio Supercomputer Center. MPI Primer / Developing With LAM. Ohio, Ohio State University, 1996.

[10] Radajewski, Jacek, Douglas Eadline. Beowulf HOWTO Version 1.1.1. On-line PDF File. 1998.

[11] Radajewski, Jacek, Douglas Eadline. Beowulf Installation and Administration HOWTO Version 0.1.2. On-line PDF File. 1999.

[12] Bendat, Julius C., Allan G. Piersol. Random Data: Analysis and Measurement Procedures. New York: Wiley-Interscience, 1971.

[13] Society of Naval Architects and Marine Engineers. *Principles of Naval Architecture*. New York: The Society of Naval Architects and Marine Engineers, 1967.

[14] Newland D. E. *An Introduction to Random Vibrations and Spectral Analysis*, Second Edition. New York: Longman Inc., 1984.

[15] Sikora, Jerome P., Alfred Dinsenbacher, Jeffrey E. Beach. "A Method for Estimating Lifetime Loads and Fatigue Lives for Swath and Conventional Monohull Ships." Naval Engineers Journal. May 1983.

[16] Abernethy, Dr. R. B., J. E. Breneman, C. H. Medlin, G. L. Reinman. Weibull Analysis Handbook. Wright-Patterson Air Force Base, Ohio: Air Force Systems Command, 1983.

[17] Sikora, Jerome P. "Cumulative Lifetime Load Criteria For Surface Ships." Ship Structures and Protection Department Departmental Report. Bethesda: Naval Surface Warfare Center Carderock Division, 1989.

[18] Dalzell, John F. "The Ship in a Seaway." NSWCCD Technical Report. Bethesda: Naval Surface Warfare Center Carderock Division, 1970.

[19] Lewis, Richard R. *A Systems-Level Approach for Estimating Wave Impact Design Pressures*. Dissertation for Engineering Management and Systems Engineering, School of Engineering and Applied Science of The George Washington University. Washington: The George Washington University, 2005.

NAVSEA
WARFARE CENTERS
Carderock Division