Copy No._____

Defence Research and Development Canada
Recherche et développement pour la défense Canada

DEFENCE R&D DÉFENSE

Channel Characterization Modelling

C. Calnan
xwave

xwave
36 Solutions Drive
Halifax, NS B3S 1N2

Project Manager: J. Theriault, 902-426-3100 ext 376

Contract Number: W7707-02-1933

Contract Scientific Authority: J. Theriault, 902-426-3100 ext 376

xwave Contract Number: 1004920 CCMOD

Prepared for:
US Office of Naval Research
800 N Quincy Street
BCT1, Suite 1113
Arlington, Virginia 22217-5660, USA
Contract No. N00014-03-C-0147

Defence R&D Canada – Atlantic

Contract Report
DRDC Atlantic CR 2004-247
January 2005

Canada

This page intentionally left blank.

Channel Characterization Modelling

C. Calnan
xwave

xwave
36 Solutions Drive
Halifax, N.S.
B3S 1N2

Project Manager: J. Theriault 902-426-3100 Ext 376

Contract Number: W7707-02-1933

Contract Scientific Authority: J. Theriault 902-426-3100 Ext 376

xwave Contract Number: 1004920 CCMOD

Prepared for:
US Office of Naval Research
800 N Quincy St.
Arlington, VA 22217-5660
Contract Number N00014-03-C-0147

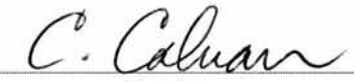
Defence R&D Canada – Atlantic

Contract Report

DRDC Atlantic CR 2004-247

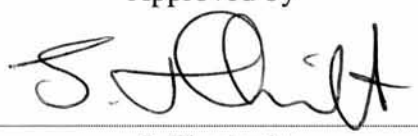
January 2005

Author



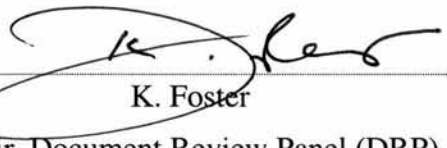
C. Calnan

Approved by



J. Theriault
Contract Scientific Authority

Approved for release by



K. Foster
Chair, Document Review Panel (DRP)

Terms of release: The scientific or technical validity of this Contract Report is entirely the responsibility of the contractor and the contents do not necessarily have the approval or endorsement of Defence R&D Canada.

Abstract

Before underwater transducers and hydrophone arrays are deployed for use it is necessary that researchers have an idea of how successful a particular sound waveform would be in transmitting information to the receivers. Currently rough calculations are used to predict how a specific waveform would be altered by its environment during its travels through the water.

There is a need for a program or programs that would take a specific environment's parameters, as modelled by programs like the US NUWC's **GSM**, and more accurately indicate the suitability of a waveform's parameters as well as other transmission and receiver parameters. This will be an aid to determining optimal combinations of these parameters. Sea trials can then be set up using these sets of preferred parameter combinations with the expectation of obtaining high quality results.

Résumé

Avant de déployer et d'utiliser des transducteurs et des réseaux d'hydrophones sous-marins, il faut que les chercheurs aient une idée de l'efficacité avec laquelle une forme d'onde sonore spécifique permettrait d'acheminer de l'information vers les récepteurs. Actuellement, des calculs approximatifs sont utilisés afin de prévoir comment une forme d'onde spécifique serait modifiée par le milieu sous-marin dans lequel elle se propage.

Il faudrait élaborer un ou plusieurs programmes, qui tiendraient compte des paramètres d'un milieu particulier générés par des programmes comme le GSM (*Generic Sonar Model*) du NUWC (*Naval Undersea Warfare Center* des États-Unis), et qui indiqueraient avec une plus grande précision l'adéquation des paramètres d'une forme d'onde ainsi que des autres paramètres d'émission et de réception. Il serait ainsi plus facile de déterminer les combinaisons optimales de ces paramètres. On pourrait ensuite préparer des essais en mer avec ces combinaisons de paramètres et s'attendre à obtenir des résultats positifs et concluants.

This page intentionally left blank.

Executive Summary

Introduction
Before underwater transducers and hydrophone arrays are deployed for use it is necessary that researchers have an idea of how successful a particular sound waveform would be in transmitting information to the receivers. Currently rough calculations are used to predict how a specific waveform would be altered by its environment during its travels through the water.

There is a need for a program or programs that would take a specific environment's parameters, as modelled by programs like the US NUWC's **GSM**, and more accurately indicate the suitability of a waveform's parameters as well as other transmission and receiver parameters. This will be an aid to determining optimal combinations of these parameters. Sea trials can then be set up using these sets of preferred parameter combinations with the expectation of obtaining high quality results.

Results

This contract report includes a user's and programmer's description of the channel propagation model, WATTCH (**Wa**veform **T**ransmission **T**hrough a **Ch**annel).

Significance

The WATTCH model produces a simulated time series given a range-independent acoustic environment and a given arbitrary broadband input waveform. The output of WATTCH can be used to stimulate signal-processing techniques and investigate the impact of the acoustic environment on the ability to coherently process the received signals. With this capability, coherence measures can be simulated before attempting to make field-trial measurements. By doing so, the use of field trial time, whether for ASW or underwater communications, can be optimized.

Future plans
A desirable extension of this modelling capability is to include the ability to model the effects in a range-dependent acoustic environment. It is conceivable to replace the WATTCH input from GSM with similar input generated by a range-dependant acoustic model.

Calnan, C. 2005. *Channel Characterization Modelling* DRDC Atlantic CR 2004-247. Defence R&D Canada – Atlantic.

Sommaire

Introduction
Avant de déployer et d'utiliser des transducteurs et des réseaux d'hydrophones sous-marins, il faut que les chercheurs aient une idée de l'efficacité avec laquelle une forme d'onde sonore spécifique permettrait d'acheminer de l'information vers les récepteurs. Actuellement, des calculs approximatifs sont utilisés afin de prévoir comment une forme d'onde spécifique serait modifiée par le milieu sous-marin dans lequel elle se propage.

Il faudrait élaborer un ou plusieurs programmes, qui tiendraient compte des paramètres d'un milieu particulier générés par des programmes comme le GSM (*Generic Sonar Model*) du NUWC (*Naval Undersea Warfare Center* des États-Unis), et qui indiqueraient avec une plus grande précision l'adéquation des paramètres d'une forme d'onde ainsi que des autres paramètres d'émission et de réception. Il serait ainsi plus facile de déterminer les combinaisons optimales de ces paramètres. On pourrait ensuite préparer des essais en mer avec ces combinaisons de paramètres et s'attendre à obtenir des résultats positifs et concluants.

Résultats
Le présent rapport, découlant d'un contrat, comprend des descriptions utilisateur et programmeur du modèle de propagation dans un canal, WATTCH (***Wa****veform* ***T****ransmission* ***T****hrough a* ***Ch****annel*, propagation des formes d'onde dans un canal).

Portée
Le modèle WATTCH produit une série chronologique simulée en fonction d'un milieu acoustique indépendant de la distance et en fonction d'une forme d'onde d'entrée large bande arbitraire. Les données de sortie du modèle WATTCH peuvent être utilisées pour simuler des techniques de traitement de signaux et analyser les effets du milieu acoustique sur la capacité de traiter de façon cohérente les signaux reçus. Il est ainsi possible de simuler les valeurs de cohérence avant d'effectuer des mesures dans le cadre d'essais sur le terrain. De cette façon, le temps consacré aux essais sur le terrain, que ce soit pour les communications sous-marines ou pour la guerre anti-sous-marine (GASM), peut être utilisé de façon optimale.

Recherches futures
On désire améliorer cette capacité de modéliser en ajoutant la modélisation des effets dans un milieu acoustique dépendant de la distance. On peut envisager de remplacer les données d'entrée du modèle WATTCH qui proviennent du GSM par des données semblables générées par un modèle acoustique dépendant de la distance.

Calnan, C. 2005. *Channel Characterization Modelling* (Modélisation pour l'établissement des caractéristiques de canal). RDDC Atlantique CR 2004-247. R & D pour la défense Canada – Atlantique.

Table of Contents

List of Figures

Acknowledgements

The author wishes to thank the following for their assistance:

Dave Chapman for settling some disquieting questions about negative frequencies and phase shifts.

Sean Pecknold for producing the waveform and eigenray files used during program development as well as information on GSM and its output files.

This work was funded by the US Navy, Office of Naval Research, Contract N00017-03-C-0147.

1. Introduction

When acoustic testing is being performed sound waves are emitted under water and are received by one or more hydrophones. There are tools that can be used to model two parts of this process. Programs such as **TPULSE** can model the shape of the transmitted sound waves as a waveform file and programs like **GSM** can model what happens to sound waves transmitted through a specific underwater environment.

However there was no simple way to combine the two data sets and so determine what a given waveform would do when propagated through the specified environment. Rough calculations could be made by hand or using spreadsheet programs, but these were generally rough indeed.

The purpose of performing the waveform/eigenray calculations is to enable researchers to design waveforms that would propagate through a particular underwater environment and arrive at a series of receivers in a way that would maximize the quality of the recorded information. Operationally, **GSM** would be used to model the environment and then a number of waveforms would be generated and tested until one would be found to maximize results. The requirement, then, is for a program or programs that can read a waveform, combine it with an eigenray view of a physical environment, and determine what the receivers would record.

This contract was issued in order to bridge the gap between the data sets and perform the needed modelling. Two programs were to be written in order to model the problem. The first program would be a frequency-independent version that would simply use time of flight and amplitude data from eigenray files to scale the waveform data and shift it temporally for a series of receivers. This would be done for all eigenrays, and any constructive and destructive interference between multiple eigenrays arriving at the same receiver nearly simultaneously would be due to the differences in the times at which the waveforms reached the receivers.

The second program would be more complex. It would be frequency and phase dependent, in addition to the parameters used for the first program. This program would perform an FFT on the waveform and, using eigenray data interpolated for the FFT frequencies as well as FFT magnitudes and phases, integrate these data to calculate the intensities at the receivers.

All code written for this project was produced in IDL and is compliant with the standards for the DRDC Atlantic IDL Library in force at the time they were written, as specified in *DRDC IDL Library Modernization Project* [Calnan, 2002].

All IDL program files written for this project contain only one main program or function, and the name of the program or function is the same as that of the file containing it. The only exception is that all filenames are in lower case letters only, whereas program and function names are given in capital letters. As two examples:

the program **WATTCH_FI** resides in the file **wattch_fi.pro**, and the function CONVERT_HDR resides in the file **convert_hdr.pro**.

Besides common English typographic conventions, the following conventions are used in this document:

- **bold text** is used for filenames (e.g. **test.pro** or **/local/files/test.pro**)
- **Bold Arial text** is used to indicate program names.
- Arial text is used to indicate function and subroutine names.
- *italic Arial text* is used for variable names
- `Courier text` is used for text to be typed on the keyboard, code or file lines, etc. (e.g. enter "`idl`")

2. Program Input and Output

Two programs were written for this project to perform channel characterization modelling runs. These programs, **WATTCH_FI** (**Wa**veform **T**ransmission **T**hrough a **Ch**annel – **F**requency **I**ndependent) and **WATTCH** (**Wa**veform **T**ransmission **T**hrough a **Ch**annel), are discussed in Sections 4 and 5 respectively, but both use the same general type of input. The only input files needed by these programs are an eigenray file and a waveform file.

The eigenray file is one that is produced by programs such as **GSM**. In the development of the code for this project the eigenray files used were produced by **GSM**, version F, Generation 4, referred to in this document as **GSM-F-4**.

The second file is a time series file containing a waveform to be modelled. This file must be in DRDC's DAT or DAT32 format. The file used for program development was created by the program **TPULSE** and consisted of 1 second of a 1 kHz pulse sampled at 4096 Hz.

When the programs are running the user must supply two other items:

- the name of the output file, and
- a calibration coefficient that will be used to convert the waveform file's data from volts to pressure.

The modelling is only performed on the data after the conversion is performed, which is done via the method:

$$\text{Pressure} = \text{voltage} \times 10^{(\text{calibration coefficient} \times 0.1)}$$

The results of the models' runs are put into a DRDC DAT32 format time series file with pressure values stored as double precision real numbers. The data file will have one channel containing the original waveform plus one for each range of the eigenray file. The sampling rate of the output file will be the same as that of the input waveform file.

Other programming tools available at DRDC may then be used to further process the output of **WATTCH_FI** and **WATTCH**, but two deserve special mention. The IDL program **VIEW_DAT** reads DAT and DAT32 format files and allows users to plot to the screen the contents of these files as they are recorded. The user may choose which channels to plot and a time window for the selected channels. A program option allows the user to produce a PostScript file version of the screen plots. **VIEW_DAT** is run without input parameters; it interactively obtains all the information it needs to produce the plots.

The function READ_DAT_DAT32 reads a DAT or DAT32 format file and returns the file's contents in two structures, one containing the header information and another containing the data. This function has a number of calling parameters, but its code has a header that explains what the parameters are and how to use them.

WATTCH_FI and **WATTCH** both call this function, so their code may be examined to see how to use the function.

3. List of IDL Files

The programs that comprise **WATTCH_FI** and **WATTCH** are, or will soon be, available from the DRDC Atlantic IDL Library. The main files for the programs are **wattch_fi.pro** and **wattch.pro**, and by way of their "Download All" option in the IDL Library, all the files required to use them will be downloaded. The majority of the functions listed below are used by both programs.

The list below alphabetically names all the files required for both **WATTCH_FI** and **WATTCH** and lists them as programs (main routines) and functions. A number of these routines were written for this project, but the majority had been written earlier for other programs. These older routines are flagged with the notation " - Pre-existing routine".

angdiffr.pro - Function that finds the difference between two angles in radians. The result is always a positive value between 0 and π since that is what was required.

ba2str.pro - Function that converts a byte array to a character string for printout purposes. In doing the conversion any NULL characters in the array are replaced by blanks. – Pre-existing routine.

chk_hbs.pro - Function that checks a DAT or DAT32 file header and determines if the file's data require byte-swapping. – Pre-existing routine.

convert_hdr.pro - Function that converts a DRDC DAT file header to DAT32 format.

freq_arr.pro - Function that produces an array of frequency values based on information in an eigenray file's header. – Pre-existing routine.

get_header.pro - Function that is the overall routine to read DAT and DAT32 file headers. – Pre-existing routine.

gsmf.pro - Program that reads a **GSM** output file and returns its contents within a structure. – Pre-existing routine.

lin_arr.pro - Function that produces a linearly spaced array of values, used to produce an array of range distances based on information in an eigenray file's header. – Pre-existing routine.

make_dat_hdr.pro - Function that creates an appropriate header for the data in a DAT or DAT32 file. – Pre-existing routine.

read_dat_dat32.pro - Function that oversees reading in DAT and DAT32 files. – Pre-existing routine.

read_head.pro - Function that reads the header of a DAT or DAT32 file. – Pre-existing routine.

reality_check.pro	-	Function that checks if a string contains a real number. – Pre-existing routine.
sort_eig_twin.pro	-	Function that sorts eigenray twins into runs of eigenrays across frequencies. (“Twins” and “runs” are described in Sections 5.1 and 5.2.)
validate_hdr.pro	-	Function that checks the data read from a DAT or DAT32 file header to see if the values are internally consistent. – Pre-existing routine.
wattch_fi.pro	-	**Program** that performs channel characterization modelling on frequency-independent eigenray data using time of flight and amplitude to produce results.
wattch.pro	-	**Program** that performs channel characterization modelling on multi-frequency eigenray data using time of flight, amplitude, frequency, and phase data to produce results.
wattch_plot.pro	-	Function that plots the magnitude and frequency of the waveform data after it has been FFT’d on the same plot as a display of eigenray frequencies.

4. WATTCH_FI

WATTCH_FI was the first program written to perform channel characterization modelling. It is a frequency independent program that only uses the time of flight and amplitude data from eigenray files. These eigenray files must be created with only one frequency.

WATTCH_FI was written to allow for subsequent development and testing of the WATTCH model, which simulates the transmission of broadband waveforms.

The program's operation is as follows:

- It asks the user to select an input eigenray file and an input waveform file via a file-selection GUI, and then asks the user to type in the name of an output file.
- It ensures that the purported eigenray file is indeed an eigenray file, that there is at least one range index, and that there is only one frequency. If any of these conditions are not true then appropriate error messages are written out and an error flag is set.
- If the purported waveform file is not a DRDC DAT or DAT32 format file or if there is more than one channel of data then appropriate error messages are written out and an error flag is set.
- If the named output file can't be opened then an appropriate error message is written out and an error flag is set.
- If the error flag was set then the program exits.
- The waveform data are examined and any trailing zeros, which indicate no data, are removed from the end of the array.
- The user is asked for a calibration value.
- The data read from the waveform file are converted to volts and then to pressure using the data-to-volts conversion factor built into the waveform file and the calibration value just entered.
- The timestep size of the waveform data is taken as the timestep size for the output file.
- The total number of timesteps that will be needed for the output file is calculated by adding the number of timesteps contained in the eigenray file's maximum time of flight to the number of samples in the waveform data.
- The total number of channels that will be needed for the output file is set to be the number of ranges in the eigenray file plus one
- A two-dimensional output data array is created with a size that is total number of timesteps by the total number of channels. All elements of the array are initially set to zero.
- If the waveform data file was in DAT format, then its header is converted to DAT32 format.
- The appropriate information for the output file's requirements are written to the DAT32 file header.

- The input waveform data are put into the output data array's first channel starting at the first time element, i.e. at a time of 0.
- A loop is run once for every eigenray in the input eigenray file.
  - The range index for the eigenray is noted.
  - The time of flight for the eigenray is noted and converted to the nearest timestep by multiplying the time by the waveform's sampling frequency. This produces a time index for the output data array to be used for the first value of the scaled waveform data.
  - The pressure values of the waveform's data are multiplied by the eigenray's amplitude, and the resulting scaled waveform data are added to the contents of the output data array for the appropriate range index starting at the position index just obtained.
- After all the eigenrays' data have been used to produce scaled waveform data that have been added to existing values in the output array, the DAT32 format file header and the output data array are written to the output file. Because of the format and contents of the output file, it is a DRDC DAT32 format file.

As a program check, the original waveform data were converted "by hand" using the eigenray file's parameters for several ranges. The results indicated that the program was working properly.

5. WATTCH

WATTCH was the second program written to perform channel characterization modelling. Unlike **WATTCH_FI** this program has both frequency and phase dependencies.

Because this program is more complex than **WATTCH_FI** its operation is described in this section in two ways: Section 5.1 describes the overall operation in descriptive terms, while Section 5.2 gives the operational description, as was done for **WATTCH_FI** in Section 4.

5.1 Program's General Operational Description

The program reads eigenray and waveform data files. After the waveform data are converted to pressure, an FFT is performed on the data and the FFT results are used to produce magnitudes and phases. The frequencies associated with each magnitude and phase are calculated.

Figure 1 is a plot of the positive frequency magnitudes resulting from an FFT analysis of a 1 second 1 kHz waveform sampled at 4096 Hz. Overlaid dotted lines indicate the frequencies for which eigenrays were calculated.

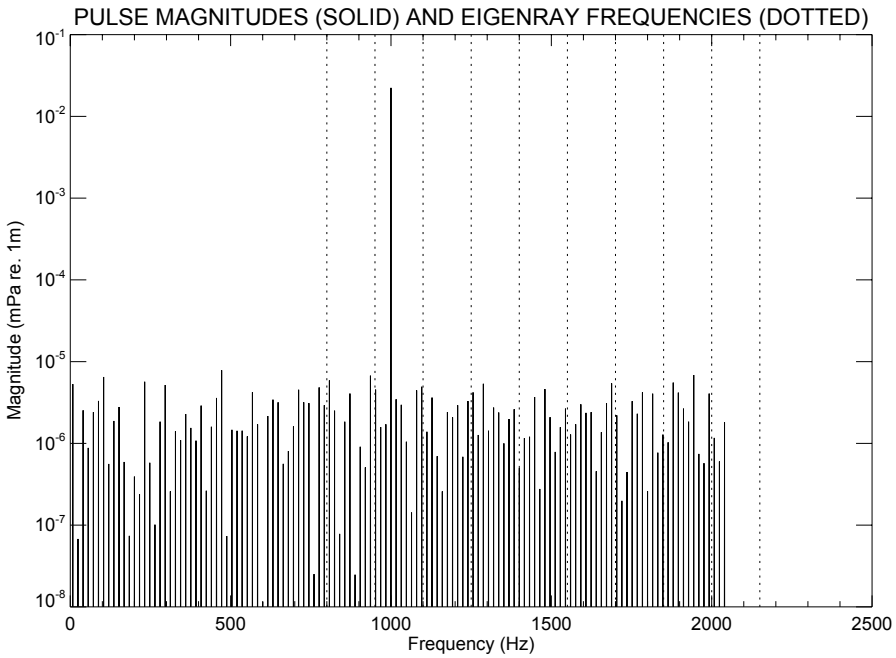


Figure 1. *Waveform spectrum and eigenray frequencies.*

The eigenray data are examined one range at a time. For each range the data are sorted by frequency and corresponding eigenrays are found for each range's frequencies by way of an eigenray "type identifier." This produces a run of eigenray data that crosses a number of frequencies. (There may always be one value for each frequency in the run, but it is not known if this always happens. This may be dependent on the program that creates the eigenray files.)

It occasionally happens that more than one version of an eigenray type is created for a range/frequency pair. If this happens these "twins" are passed into a subroutine to be sorted into runs. The sorting is done by pairing the eigenrays from different frequencies with those having the closest phase values.

The eigenray amplitude and phase data are interpolated linearly for the waveform frequencies that lie between pairs of eigenray frequencies.

The eigenray amplitude and phase data for the lowest eigenray frequency are assigned to all waveform frequencies smaller than that eigenray frequency. In Figure 1 this means that the data for the minimum eigenray frequency of 800 Hz are used for all waveform frequencies from 0 Hz through 800 Hz. Similarly, if any waveform frequencies are greater than the maximum eigenray frequency, the data associated with the highest eigenray frequency are assigned to those waveform frequencies. The data are simply assigned instead of being extrapolated because if the last two eigenray values at either end of its frequencies are not close to each other in value, extrapolation could produce unrealistic results.

By this point eigenray data have been assigned to or interpolated for all waveform frequencies for a given eigenray run. Part of the eigenrays' type identifier is the time of flight of the eigenray, so these data are all associated with the same time of flight.

With these values a summation can be made over all waveform frequencies via the equation:

$$Sum = \Sigma Amp_W \bullet Amp_I \bullet [\cos(2\pi f_W t + \phi_W + \phi_I) + \cos(-2\pi f_W t - \phi_W - \phi_I)]$$

where:

Amp_W	-	is the array of waveform amplitudes at the waveform frequencies
Amp_I	-	is the array of interpolated or assigned eigenray amplitudes at the waveform frequencies
f_W	-	is the array of waveform frequencies
ϕ_W	-	is the array of waveform phases at the waveform frequencies
ϕ_I	-	is the array of interpolated or assigned eigenray phases at the waveform frequencies
t	-	is the elapsed time of the wave, initially zero and discussed further below

The second cosine term in the equation is required to take into account the fact that the results of an FFT produce values for a range of frequencies from $-X$ Hz to $+X$ Hz. Figure 1 only displays the positive frequency values because magnitudes are

symmetrical about zero Hz; i.e. the magnitude for, say, 1200 Hz is the same as that for -1200 Hz. The phases, however, are the negative of each other for "opposite" frequencies; i.e. if a frequency of 850 Hz has a phase of -27° then the phase for -850 Hz would be +27°.

Sum, as calculated above, produces a time *t* amplitude for the current eigenray run, and this value is added to the output data array in the slot designated for the eigenray's range and time. With an initial value of $t = 0$, the time slot is that associated with the eigenray run's time of flight.

Once the $t = 0$ work is done, *t* is incremented by the time difference between waveform samples and *Sum* is recalculated with all the other values unchanged. The value of *Sum* produced this time is added to the output data array value currently in the next time slot. The time is continuously incremented until one calculation has been made for each sample timestep in the input waveform.

After values for all the timesteps have been calculated, the program moves on to the next eigenray run and begins all over again.

The flight times of the various eigenray runs tend to be clustered in groupings closer together than the duration of the waveform data. Because of this, a particular output data array slot for a specific range and time can have a number of *Sum* values added to its initial value of zero, with each added value being due to a different eigenray run's contribution. This results in the production of summed waves.

If this program is run with all phase values set to zero before calculating *Sum*, this program should have produced practically the same results as **WATTCH_FI**. This was indeed done in testing, and the two programs' results were indistinguishable when plotted.

Runs of **WATTCH** made with the actual phases produced appreciably different results from the zero-phase tests. Whether or not these results were correct was not easily determined. See Section 8 for more on the topic of rigorously testing the program.

5.2 Program's Detailed Operational Description

This section presents the operation of **WATTCH** in a more detailed manner, analogous to that of Section 4. It will be noted that a large part of the early operation is similar to that of **WATTCH_FI**.

The operation of the program is:

- It asks the user to select an input eigenray file and an input waveform file via a file-selection GUI, and asks the user to type in the name of an output file.

- It ensures that the purported eigenray file is indeed an eigenray file and that there is at least one range index. If either of these is not true then appropriate error messages are written out and an error flag is set.
- If the purported waveform file is not a DRDC DAT or DAT32 format file or if there is more than one channel of data then appropriate error messages are written out and an error flag is set.
- If the named output file can't be opened then an appropriate error message is written out and an error flag is set.
- If the error flag was set then the program exits.
- The user is asked for a calibration value.
- The waveform data are examined and any trailing zeros, which indicate no data, are removed from the end of the array.
- If there are less than 512 valid waveform values an error message is written out and the program exits.
- The data read from the waveform file are converted to volts and then to pressure using the data-to-volts conversion factor built into the waveform file and the calibration value just entered.
- An FFT is performed on the waveform pressure data and magnitude and phase are calculated from the results.
- An array of the waveform's FFT positive frequencies is calculated.
- The range of waveform FFT frequencies and the eigenray frequencies are written to the screen and a Figure 1 format plot is drawn on the screen. The user is given the option of producing a PostScript version of the plot.
- The user is given the option of exiting the program at this point, something that may be desired if the FFT indicates something odd with the waveform file or if the eigenray frequencies provide poor coverage of the waveform spectrum.
- The total number of timesteps that will be needed for the output file is calculated by adding the product of the eigenray file's maximum time of flight and the waveforms data's sampling rate to the number of samples in the waveform data.
- The timestep size of the waveform data (i.e. the inverse of its sampling frequency) is used as the timestep size for the output file.
- An array is created containing 2π times the waveform's frequencies, and another is created with the times of the waveform file's samples using zero seconds for the time of the first element.
- New, "empty" arrays are created to hold interpolated eigenray phases and amplitudes, and the output data. The first two are 1-D arrays, but the last is a 2-D array with dimensions of number of total timesteps by number of channels. The total number of channels is the number of eigenray ranges plus 1.
- If the waveform data file was in DAT format, then its header is converted to DAT32 format.
- The appropriate information for the output file's requirements are written to the DAT32 file header.
- The input waveform data are put into the output data array's first channel starting at the first time element, i.e. at a time of 0.
- The eigenray data are sorted by range if they weren't already.
- A loop is run once for every eigenray range.

- Start and end indices for the range's eigenray data (read into 1-D arrays) are located. If there are no data for the current range a message is printed and operations move to the next eigenray range.
- An eigenray type identifier (TID) is produced for each eigenray in the range. The TID is made by combining the data unique to an eigenray type: the number of bottom and surface reflections, the source and target angles, and the time of flight.
- A loop is run for each eigenray in the range.
    - If the current eigenray has not been used for calculations yet then processing continues. If the eigenray has been used skip to the next one.
    - A loop runs through the remaining eigenrays in the range, starting at the current one. Any with a TID that matches that of the current eigenray are noted in order to build up eigenray runs. (A run is a series of eigenrays in different frequency ranges that all have the same TID.) As part of the match checks, the number of TID matches per frequency are noted.
    - The total number of runs is set to the maximum number of occurrences of any TID in a frequency.
    - Arrays are created to hold the eigenrays' runs values for amplitude, frequency, and phase. These arrays are 2-D and their size is the number of runs by the number of eigenray frequencies.
    - If the number of runs is 1, the appropriate eigenray data are copied from the eigenray file data structure to the newly created arrays.
    - If the number of runs is greater than 1 then a function is called to sort the eigenrays into runs (based on minimum differences between phases) and identify the data belonging to each run. Then the appropriate eigenray data are copied from the eigenray file data structure to the newly created arrays. Note that if the number of TIDs is not the same for every frequency a flag is set that prevents any of the eigenrays with the current TID from being used. This is done by setting the number of runs to a negative number.
    - A loop is run from 1 to the number of runs.
        - If there is only 1 eigenray frequency in the run then the interpolated eigenray data arrays are filled with the data from that frequency.
        - If there is more than 1 eigenray frequency:
            - The interpolation array elements for all frequencies lower than the minimum eigenray frequency are filled with the data for the minimum eigenray frequency.
            - Linear interpolation is used to calculate eigenray values for all waveform frequencies falling between pairs of eigenray frequencies.
            - Should there be any waveform frequencies higher than the maximum eigenray frequency, then the values associated with the maximum eigenray frequency are assigned to those waveform frequencies.
        - An array (*PRODAMP*) is produced that is the element by element product of the waveform amplitudes at its FFT frequencies and the associated interpolated eigenray amplitudes.

- A loop is run once for each timestep of the input waveform.
  - An array (*TWOPIWF*) is calculated that is 2π times the waveform's frequencies the timestep index times the duration of a timestep (i.e. $2\pi f_W t$, or $\omega_W t$).
  - An array of amplitudes for each waveform frequency is calculated by:

    $$PRODAMP \bullet [\ \cos(TWOPIWF + PULPHS + INTPHS) + \cos(-TWOPIWF - PULPHS - INTPHS)\ ]$$

    where:

    PULPHS - is the array of waveform phases at all waveform FFT frequencies

    INTPHS - is the array of eigenray phases interpolated or assigned for the same frequencies
  - The values in the amplitude array are summed up and added to the value currently in the output data array for the current range and at a time equal to the time of flight plus the timestep number.
- End of the timestep loop.
- End of the number of runs loop.
- The values of the TIDs of all the eigenrays that have been used (or not used, if not all frequencies had the same numbers of TIDs) are reset to "DONE" so they will not be used again.
- End of the loop run for each eigenray in the range.
- End of the loop run for each range.
- The DAT32 format file header and the output data array are written to the output file. Because of the format and contents of the output file, it is a DRDC DAT32 format file.

Notes at the end of Section 5.1 describe the testing done with this program.

6. Running IDL Programs

This section presents some information needed by a user in order to run **WATTCH_FI** and **WATTCH**. This section contains:

Section 6.1 - specifies what must be done in order to satisfy IDL requirements before analyses can be run.

Section 6.2 - gives general information on running the analyses routines and IDL in general.

6.1 IDL Setup Requirements

The programs that comprise **WATTCH_FI** and **WATTCH** are available from the DRDC Atlantic IDL Library. Downloading the main programs from the IDL Library using the "Download All" option should result in the downloading of all the files needed to run the programs.

To run the programs either:

- all required IDL files must be in the directory from which they will be run, or
- the directory (or directories) that the files are in must be in the IDL installation's path.

The files may be placed in a directory that is already in IDL's path but it is recommended that either their home directory be added to the path or a new directory be created for them, both of which would simplify the files' upgrade or removal.

The IDL installation's current path is stored in the environment variable *!PATH*, whose value may be found by entering the command "`print, !PATH`" while in an IDL window.

The value of *!PATH* is set in different ways on different platforms, but briefly:

UNIX	-	*!PATH* is initialised from the environment variable *IDL_PATH* when IDL starts; this variable may be assigned in the files **.cshrc** or **.profile**.
VMS	-	*!PATH* is initialised from the logical name *IDL_PATH* when IDL starts; this variable may be assigned in the file **LOGIN.COM**.
Windows	-	*!PATH* is initialised from the saved *IDL for Windows* preferences data or from the DOS environment variable *IDL_PATH* when IDL starts.
Macintosh	-	*!PATH* is initialised from the saved *IDL for Macintosh* preferences data when IDL starts.

In any event, the value of *!PATH* can be changed while in IDL by the simple expedient of an assignment statement. For example, on a UNIX system a user may enter:

```
!PATH=!PATH+':/local/user/idl-code'
```

More detail is given in greater clarity in the IDL documentation. In the V5.1 manuals the information is found in *Reference Guide (Vol 1: A-M)*, Chapter 3, pp. 19-21, under the "IDL Environment System Variables" subsection "!PATH." Alternatively, the online manual index may be searched for the entry "!PATH."

As well, the IDL setup may have to be run. That is accomplished on most computers by having the following line, or something similar, in the user's **.login** file (or whatever the appropriate shell startup file is named):

```
source /local/rsi/idl_5.6/bin/idl_setup
```

If the analyses are being run on a remote computer, to get the graphics to appear on the screen locally the user must obtain the IP address of the local computer (111.222.33.44, for example) and enter the following command on the remote computer:

```
setenv DISPLAY 111.222.33.44:0.0
```

Note that **WATTCH_FI** and **WATTCH** have only been tested on IDL version 5.6, its development version.

6.2 Running WATTCH_FI and WATTCH

To run the programs either:

- the user must be in the directory with the IDL program files, or
- the directory holding those files must be in the account's path, a point referred to in the previous section.

Assuming that at least one of the above conditions is met, to run the programs the user must:

- At the system's command line prompt enter "`idl`" to start IDL.
- At the IDL prompt enter "`wattch_fi`" or "`wattch`", whichever is desired. The main program and any functions it calls will be compiled as IDL attempts to run them.

IDL is not case sensitive with regard to commands or switches but is with regard to filenames on case sensitive operating systems. When compiling an IDL program on a case sensitive operating system the filename must be entered exactly as it appears in a directory listing. However, since IDL itself is not internally case sensitive, a file named **DooDah.pro** containing the program **DooDah** must be compiled via the command "`.com DooDah`", but may be run by issuing the commands "`doodah`", "`DOODAH`", "`DoOdAh`", etc. This means that to run **WATTCH** a user need not enter "`wattch`" (the main file's case appropriate name) once the program is compiled. As can be imagined, this feature leads itself to great possibilities of confusion for a case-unwary programmer.

7. Other Work Performed

The current project work was done on a computer running Red Hat Linux 8.0. Near the start of the project eigenray and waveform data files that had been produced elsewhere were put on the development computer. The programs that had produced these files (**GSM-F-4** and **TPULSE**, respectively) were run on another Linux box since that computer contained those programs and the development computer didn't.

The Linux version of **GSM-F-4**, however, would crash if it was instructed to sort the eigenrays after producing them. Some time was spent examining the subroutines that performed the sorting, but nothing was found in the code that could explain the problem. Then, for comparison, **GSM-F-4** was run on an HP computer. It was quickly realized that the Linux and HP versions of **GSM-F-4** produced eigenray files of different sizes, despite the fact that they should not have.

This led to a deeper examination of the two eigenray files. Considerable differences were found between the files, although individually the files' contents looked reasonable on a gross scale. A closer examination of the Linux-produced file, however, revealed that it contained erroneous values that were not present in the HP-produced files. At this point it was decided that the Linux version of **GSM-F-4** was not operating properly and was calculating eigenrays incorrectly. Then, when an eigenray sort was attempted, either the contents of these files or the program itself caused the program to crash.

An intensive examination of the program's code was undertaken and ultimately the reason for the errors was discovered. The proper operation of a number of **GSM-F-4's** subroutines requires that a number of their variables are static. That is, these variables are expected to retain their values between calls, with subsequent calls making use of results of earlier processing. As it happened, the FORTRAN compiler being used, g77, by default only treated variables as static if they belonged to `COMMON` areas or were initialized by `DATA` statements. After this discovery was made it was a matter of reading through the g77 documentation until a compiler option was discovered that caused subroutines' variables to always be treated as static variables.

With this option in place the newly compiled code produced eigenray files, both sorted and non-sorted, that were identical to those produced on the HP.

After this problem was solved, properly operating versions of **GSM-F-4** were distributed to a number of Linux users, none of whom have reported any problems with the program.

8. Suggestions for Further Work

A thorough testing of the program **WATTCH** should be undertaken. As is mentioned in the last two paragraphs of Section 5.1, comparison with **WATTCH_FI** output was only performed subjectively, by visually comparing plots of the output data for one test. A more objective comparison of the two programs' output files should be performed. One way would be to perform a value-by-value comparison and noting any differences, perhaps as a percentage of the **WATTCH_FI** values.

Proper testing of **WATTCH** using phase values should be performed. The waveform and eigenray files used as development tools for this program were created more or less randomly and did not easily lend themselves to the determination of what the phase shifted results should be. Proper testing would require the creation of new waveform and eigenray files for which results as produced by **WATTCH** could be calculated "by hand." These two sets of results could then be compared, as would be done for the comparison with **WATTCH_FI** results.

A final extension to work on the programs would be to have knowledgeable users run the programs for a large number of real-world cases. These users could then be questioned to determine how the program could be expanded or made more user friendly.

Bibliography

Calnan, C. (2002). *DRDC Atlantic IDL Library Modernization Project* (DRDC CR 2002-187, November 2002)

Initialisms and Acronyms

DRDC	Defence Research and Development Canada
IDL	Interactive Data Language

Distribution List

Internal Distribution

2 DRDC Atlantic LIBRARY FILE COPIES
4 DRDC Atlantic LIBRARY (SPARES)

3 J. Theriault
1 S. Pecknold
1 D. Ellis
1 D. Chapman
1 N. Allen
1 D. Hazen
1 M. LaFrancois
1 M. Hazen
1 B. Roger

External Distribution

1 AUTHOR
3 Dr. Maribel Soto
US Office of Naval Research
800 N Quincy St.
Arlington, VA 22217-5660
1 Dr. Dan Nagle
NUWC, Division Newport
1176 Howell Street
Newport, RI 02841
1 DMCA Americas (copy of distribution letter only)
Ottawa
1 LCdr W. Renauld,
SSO METOC,
MARLANT
CFB Halifax
1 D. McGaughey,
Dept Electrical and Computer Engineering
Royal Military College of Canada
Box 17000, Station Forces
Kingston, Ontario, K7K 7B4
1 NDHQ/ CRAD/ DRDKIM 3

Total 26 copies

This page intentionally left blank.

DOCUMENT CONTROL DATA

(Security classification of title, body of abstract and indexing annotation must be entered when the overall document is classified)

1. ORIGINATOR (the name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Establishment sponsoring a contractor's report, or tasking agency, are entered in section 8.)

Xwave
36 Solutions Drive, Halifax, NS B3S 1N2

2. SECURITY CLASSIFICATION (overall security classification of the document including special warning terms if applicable).

UNCLASSIFIED

3. TITLE (the complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S,C,R or U) in parentheses after the title).

Channel Characterization Modelling

4. AUTHORS (Last name, first name, middle initial. If military, show rank, e.g. Doe, Maj. John E.)

C. Calnan

5. DATE OF PUBLICATION (month and year of publication of document)

January 2005

6a. NO. OF PAGES (total containing information Include Annexes, Appendices, etc).

31 (approx.)

6b. NO. OF REFS (total cited in document)

1

7. DESCRIPTIVE NOTES (the category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered).

CONTRACT REPORT

8. SPONSORING ACTIVITY (the name of the department project office or laboratory sponsoring the research and development. Include address).

Defence Research & Development Canada – Atlantic
PO Box 1012
Dartmouth, NS, Canada B2Y 3Z7

9a. PROJECT OR GRANT NO. (if appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant).

30ai41

9b. CONTRACT NO. (if appropriate, the applicable number under which the document was written).

W7707-02-1933

10a ORIGINATOR'S DOCUMENT NUMBER (the official document number by which the document is identified by the originating activity. This number must be unique to this document.)

1004920 CCMOD

10b OTHER DOCUMENT NOs. (Any other numbers which may be assigned this document either by the originator or by the sponsor.)

DRDC Atlantic CR 2004-247

11. DOCUMENT AVAILABILITY (any limitations on further dissemination of the document, other than those imposed by security classification)

(X) Unlimited distribution
( ) Defence departments and defence contractors; further distribution only as approved
( ) Defence departments and Canadian defence contractors; further distribution only as approved
( ) Government departments and agencies; further distribution only as approved
( ) Defence departments; further distribution only as approved
( ) Other (please specify):

12. DOCUMENT ANNOUNCEMENT (any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected).

13. ABSTRACT (a brief and factual summary of the document. It may also appear elsewhere in the body of the document itself. It is highly desirable that the abstract of classified documents be unclassified. Each paragraph of the abstract shall begin with an indication of the security classification of the information in the paragraph (unless the document itself is unclassified) represented as (S), (C), (R), or (U). It is not necessary to include here abstracts in both official languages unless the text is bilingual).

Before underwater transducers and hydrophone arrays are deployed for use it is necessary that researchers have an idea of how successful a particular sound waveform would be in transmitting information to the receivers. Currently rough calculations are used to predict how a specific waveform would be altered by its environment during its travels through the water.

There is a need for a program or programs that would take a specific environment's parameters, as modelled by programs like the US NUWC's **GSM**, and more accurately indicate the suitability of a waveform's parameters as well as other transmission and receiver parameters. This will be an aid to determining optimal combinations of these parameters. Sea trials can then be set up using these sets of preferred parameter combinations with the expectation of obtaining high quality results.

14. KEYWORDS, DESCRIPTORS or IDENTIFIERS (technically meaningful terms or short phrases that characterize a document and could be helpful in cataloguing the document. They should be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location may also be included. If possible keywords should be selected from a published thesaurus. e.g. Thesaurus of Engineering and Scientific Terms (TEST) and that thesaurus-identified. If it not possible to select indexing terms which are Unclassified, the classification of each should be indicated as with the title).

Underwater Acoustics
IDL software
Data Analysis
Channel characterization
GSM

This page intentionally left blank.

Defence R&D Canada

Canada's leader in defence
and National Security
Science and Technology

R & D pour la défense Canada

Chef de file au Canada en matière
de science et de technologie pour
la défense et la sécurité nationale

www.drdc-rddc.gc.ca