**Scientific and Technical Report**

**CDRL A002**

**Fast computation of reverberation using Gaussian beam reflections**

**Prepared for:**

**NAWCTSD**

**ATTENTION: JACOB BOURQUE**

**12350 RESEARCH PARKWAY CODE 4.6**

**ORLANDO FL 32826-3224**

**April 14, 2015**

**Prime Contract Number: N61340-13-C-0018**

**Purchase Order Number: 1300350205**

**Prepared by:**

**Sean Reilly, David Thibaudeau, Ted Burns**

**The AEgis Technologies Group, Inc.**

**410 Jan Davis Drive, NW**

**Huntsville, AL 35806**

**Distribution Statement A - Approved for public release; distribution is unlimited,   
as submitted under NAWCTSD Public Release Authorization Tracking Number 14-ORL102001**

Table of Contents

[1 Introduction 1](#_Toc401566025)

[2 Isochrones, the classic reverberation algorithm 2](#_Toc401566026)

[3 Improved reverberation algorithm 7](#_Toc401566027)

[3.1 Derivation overview 7](#_Toc401566028)

[3.2 Intensity projection onto interface 9](#_Toc401566029)

[3.3 Total energy integration 11](#_Toc401566030)

[3.4 Duration of reverberation contributions 13](#_Toc401566031)

[3.5 Computer implementation 15](#_Toc401566032)

[3.6 Volume reverberation 16](#_Toc401566033)

[4 Reverberation test scenario 16](#_Toc401566034)

[4.1 Classic algorithm solution 20](#_Toc401566035)

[4.2 Eigenverb prototype and testing 22](#_Toc401566036)

[5 Conclusion 25](#_Toc401566037)

[6 References 26](#_Toc401566038)

[A. Appendix A – classic\_reverb.m A-1](#_Toc401566039)

[B. Appendix B – eigenverb\_demo.m B-1](#_Toc401566040)

[C. Appendix C – CASS script C-1](#_Toc401566041)

# ****Introduction****

**The accurate modeling of reverberation is a key element of providing sonar technicians with training in littoral environments. In these environments, reverberation is the dominant feature in modeling signal excess, which determines the ranges for target detection. Current reverberation models are computationally intense, especially for multi-static systems where the number of source/receiver pairs is large. Many training system development programs must increase the cost of their hardware, decrease the fidelity of their model, or both, to address these computational speed problems. This paper presents a new model that seeks to significantly improve the speed of the reverberation** calculation **without significantly decreasing the accuracy.**



Figure 1 – Reverberation envelope and signal

**As illustrated in** Figure 1**, training systems typically compute reverberation as two components. The reverberation envelope defines the reverberation strength as a function of received time, and it depends on the sensor, the transmitted pulse, and the environment. The reverberation signal is created by using this envelope to scale a random process with zero mean and constant variance. This random process need not be Gaussian. This paper focuses solely on the generation of the reverberation envelope.**

**The classic reverberation calculation** [1] **is slow because:**

1. **Eigenrays to each interface must be calculated at a resolution fine enough to support the inversion of the** two-way travel time vs. range for each azimuth and combination of path types.
2. All of the **eigenray** terms needed to compute reverberation must be transformed from functions of range into functions of two-way travel time. This process must be repeated for each azimuth and combination of path types.

**This paper defines the fundamental techniques used by a new model called** Eigenverb**. Instead of populating each interface with a collection of eigenray targets,** Eigenverb **extracts reverberation data as a side-effect of Gaussian beam reflections during the calculation of transmission loss; this increases the speed of ensonifying each interface. Information from the source’s ensonification of the interface is re-used by all receivers; this improves the scalability for multi-static systems.**

Eigenverb **relies on the Wavefront Queue 3‑D (WaveQ3D) model for the calculation of eigenray and reverberation components. WaveQ3D is a research effort to create fast and accurate acoustic transmission loss (TL) eigenrays, in littoral environments, for active sonar simulation/stimulation systems. WaveQ3D is unique among Gaussian beam models in that it solves the eikonal equation in spherical Earth coordinates. The premise of this approach is that, when scenario geometries are constantly evolving, it is more computationally efficient to perform acoustic transmission loss (TL) in the latitude, longitude, altitude coordinates of the underlying environmental databases, than it is to convert the 3‑D environments into a series of Nx2‑D Cartesian slices. This approach also has the benefit of supporting out-of-plane, 3‑D effects.**

**This paper starts with a derivation of the classic reverberation calculation**, **then a detailed derivation of the Eigenverb model is presented. Finally, a simple monostatic reverberation test scenario is used to compute the reverberation envelope and compare the results, using both the classic and** Eigenverb **reverberation algorithms. Comparisons to multi-static examples will be conducted in a future study.**

# ****Isochrones, the classic reverberation algorithm****

**In this section, we derive the classic method** [1] **of computing the bistatic interface reverberation using isochrones, scattering locations with a constant two way travel time between the source, interface, and receiver. We can write the bi-static reverberation from an interface** [2]**, illustrated in** Figure 2**, in the form**

|  |  |  |
| --- | --- | --- |
|  |  | () |

**where**

= received intensity;

= source level intensity;

= transmission loss from the source to the scattering patch, includes   
 source beam pattern, in-water absorption, interface reflection loss,   
 and spreading loss.

= transmission loss from the scattering patch to the receiver, includes   
 receiver beam pattern, in-water absorption, interface reflection loss,   
 and spreading loss.

= interface scattering strength (strength/m2);

= grazing angles at the scattering patch, along source and receiver paths;

= bearings at the scattering patch, along source and receiver paths; and

= ensonified area for this combination of source and receiver paths (m2).

The transmission loss and angle terms are functions of the scenario geometry and the eigenrays for each path. The source level and beam patterns are functions of the sensor characteristics. The scattering strength is a physical property of the interface; note that many scattering strength models have limited or no support for the arguments. The ensonified area is the only term that remains to be computed.

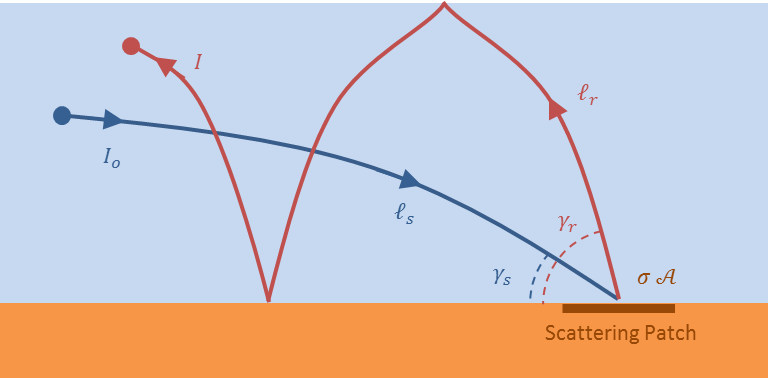


Figure 2 – ****Bi-static reverberation geometry (side view)****

Isochrones, scattering locations with a constant two way travel time between the source, interface, and receiver, are used to compute ensonified area in **the classic reverberation algorithm. As the wavefront sweeps out across the bottom (**Figure 3**), three different parts of the bottom are ensonified at three different travel times, labeled , , and . If each echo returns along the same path, the received signals (illustrated in** Figure 4**) arrive at time , , and . The duration T of the transmitted pulse causes the echoes from different scattering patches to overlap and the boosts the overall intensity. The peak return , in** Figure 4**, is the sum of contributions from all three scattering patches.**

**In the real world, the ensonification of the bottom is continuous, but the principle is similar. As illustrated in** Figure 5**, the ensonified area for the intensity at time t is the surface between all interface points with a two way travel time from to . In a range dependent environment, the propagation conditions may also be a function of the azimuthal direction**

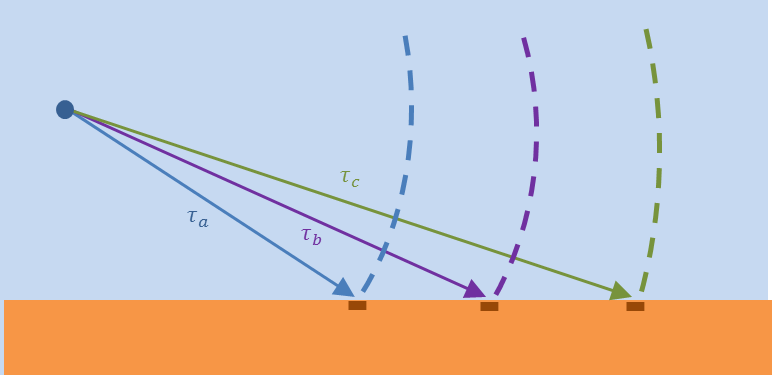


Figure 3 – ****Wavefront sweeps out across the bottom****

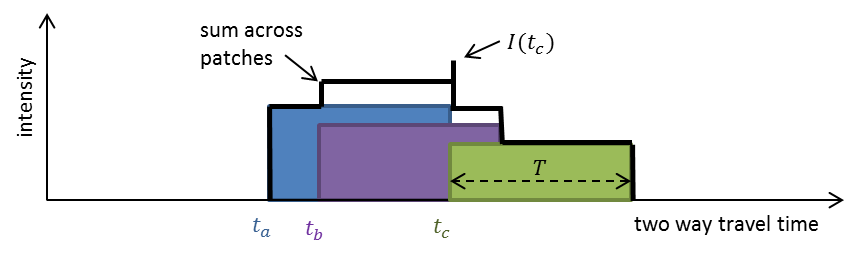


Figure 4 – Time ****overlap between ensonified patches****

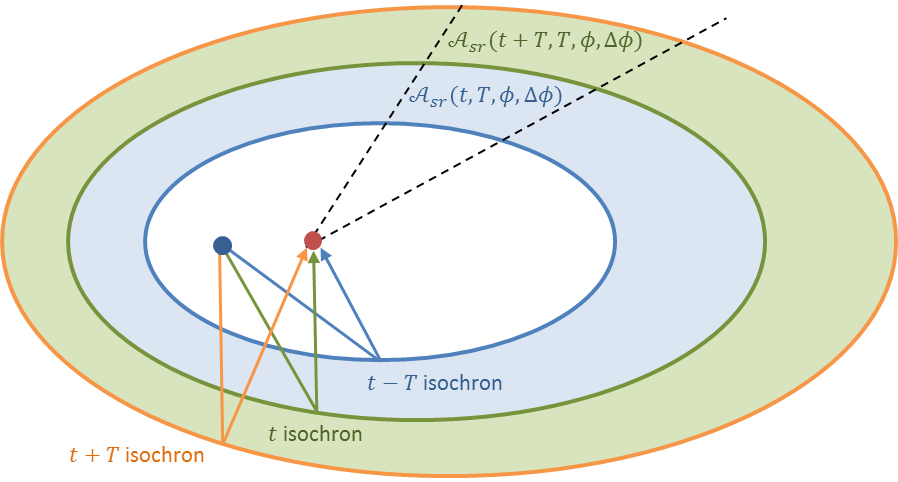


Figure 5 – Isochrone geometry (top-down view)

**The classic reverberation algorithm divides the environment into azimuthal sectors of width and computes the reverberation contribution separately for each sector. A**coustic targets are laid on each interface, along the center of each sector, at a gridded set of ranges. Multipath transmission loss eigenrays are then calculated for each of these acoustic targets. The eigenrays consist of a set of transmission losses, one-way travel times, launch angles, and arrival angles for each target and path type.



Figure 6 – Example of one way travel times as function of range and path type

For each combination of path type and azimuthal sector, the travel time (see Figure 6) is interpolated to create a set of isochrone ranges as a function of two way travel time and path combination (see Figure 7). This inversion of the range/time relationship allows other eigenray products (such as transmission loss, launch angle, and arrival angle) to be interpolated within each isochrone, for each combination of source and receiver path type.



Figure 7 – Example of isochrone range as function of two way travel time and path combination

The method of isochrones allows the ensonified area in each sector to be computed as

|  |  |  |
| --- | --- | --- |
|  |  | () |

**where**

= isochrone range, as a function of two way travel time, for each   
 source path type, receiver path type, and azimuthal sector.

= width of each azimuthal sector,

= ensonified area, as a function of two way travel time and pulse length,  
 for each source path type, receiver path type, and azimuthal sector.

Eqn. (1) is then used to compute the reverberation intensity as a function of two-way travel time and pulse length, for each source path type, receiver path type, and azimuthal sector. Finally, intensity contributions are added incoherently to form the reverberation envelope.

The range between acoustic targets in each sector must be close enough together to create a smooth estimate of isochrone ranges as a function of two-way travel time. The interpolation part of this process also requires the model to group eigenrays into source and receiver path types. In shallow water environments, where paths with large numbers of bottom bounces dominate propagation, these two requirements conspire to increase the density of acoustic targets and number of path types over which isochrones must be computed. This has a negative impact on the computational speed of the **classic reverberation** algorithm.

# ****Improved reverberation algorithm****

Like the isochrone model, the new reverberation algorithm, known as Eigenverb, assumes that the total reverberation envelope is the sum of contributions from all points on the interface. However, instead of calculating transmission loss to explicit acoustic targets on the interface, Eigenverb computes a Gaussian projection onto the interface each time that a reflection occurs, and sums reverberation contributions over all reflections.

## Derivation overview

The derivation of the Eigenverb model starts with the bistatic scenario illustrated in Figure 8. This figure illustrates a situation in which a reflection in the path from the source to the interface occurs near a reflection in the path from the receiver to the interface. Energy from the source reflection is scattered into the receiver reflection in the area where Gaussian intensities overlap.

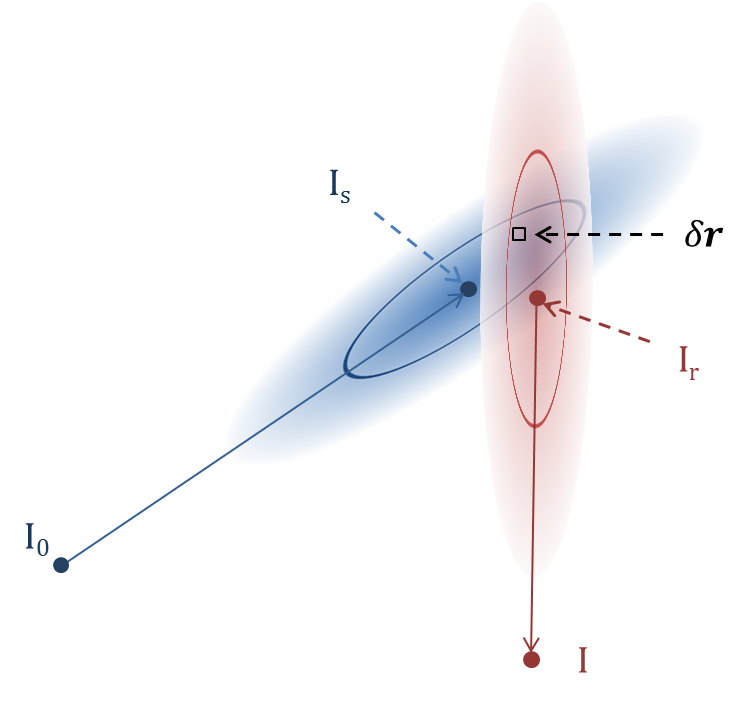


Figure 8 – Geometry of overlap between source and receiver Gaussians (top-down view)

The total energy that arrives at the receiver, for each combination of source and receiver reflection, is computed by estimating the scattering intensity for infinitesimal parts of the ensonified area, and then integrating those contributions across all times and locations.

|  |  |  |
| --- | --- | --- |
|  |  | () |

**where**

s = identification for an interface reflection from the source;

r = identification for an interface reflection from the receiver;

= location on the interface;

= travel time from source, to location on the interface, and then to receiver;

= intensity scattered from source reflection “s” into receiver reflection “r”;

= total energy received for this combination of source and receiver reflections.

The scattered intensity for each infinitesimal area in Figure 8 is the product the intensity of the ray along the source path, the intensity scattered from the area, the intensity of the ray along the receiver path, and the initial intensity at the source.

|  |  |  |
| --- | --- | --- |
|  |  | () |

**where**

= source level intensity;

= one-way travel time from source to interface;

= one-way transmission loss from source to interface;

= one-way travel time from interface to receiver;

= one-way transmission loss from interface to receiver;

= scattering strength per unit area, for this combination of paths;

= size of the scattering area;

= Gaussian profile in, vector form;

= location Gaussian’s peak intensity on the interface; and

= covariance matrix of the Gaussian projection on the interface.

If the transmission loss, scattering strength, and travel time vary slowly across the ensonified area, we can estimate the total energy received for this combination of source and receiver reflections as

|  |  |  |
| --- | --- | --- |
|  |  | () |

**where**

= duration of the transmitted pulse;

= average transmission loss from source to interface for this reflection;

= average transmission loss from interface to receiver for this reflection;

= average scattering strength per unit area, for this combination of paths;

= location of source and receiver reflections;

= covariance matrix for source and receiver Gaussian projections; and

= total energy received for this combination of source and receiver reflections.

The format of Eqn. (8) is very useful because the integral has an analytic solution. The total energy at the receiver can then be converted into an intensity profile using a Gaussian window

|  |  |  |
| --- | --- | --- |
|  |  | () |

**where**

= time of arrival in the reverberation output;

= sum of travel times between source, interface, and receiver;

= duration for this contribution; and

Note that the extra term in the numerator is there to time align the peak of each contribution to the end of the interval in two-way travel time, which matches the alignment of the classic algorithm. The factor of and normalize the total power over the duration of the window and the intensity to the total surface area one meter from the receiver, respectively.

In the improved reverberation algorithm, the data used from a single contribution is known as an eigenverb. The name is taken from the fact that eigenverbs provide discreet components of the total reverberation in the same way that eigenrays provide discreet components of the total transmission loss. Each eigenverb consists of

= D/E and AZ departure angles from the source;

= D/E and AZ arrival angles at the receiver;

= total energy received for this combination of source and receiver reflections;

= travel times between source, interface, and receiver;

= duration for this contribution; and

If the effect of the source and receiver beam patterns are already incorporated into the transmission loss, then the sum of these contributions over all paths provides the total reverberation envelope

|  |  |  |
| --- | --- | --- |
|  |  | () |

**where**  is the total reverberation envelope for this interface as a function of time of arrival. Note that unlike the classic algorithm, Eqn. (7) has the ability to sum contributions without bundling the rays into path type groups. It also has no need to invert the range/time relationship of the eigenrays. For each receiver path “r”, the summation over source paths “s” takes the place of the interpolation used during range/time inversion in the classic algorithm.

The sections that follow provide a detailed derivation of the , , and expressions used to model using Eqns. (5), (6), and (7).

## Intensity projection onto interface

Figure 9 illustrates a scenario in which the curvature of the wavefront and interface are small at the point of reflection. With these curvature assumptions, we can project the source ray’s Gaussian intensity profile onto the interface

|  |  |  |
| --- | --- | --- |
|  |  | () |
|  |  | () |

where

= source ray’s Gaussian beam width in the D/E and AZ directions;

= length and width of Gaussian projection onto interface; and

= incident grazing angle at the point of collision.

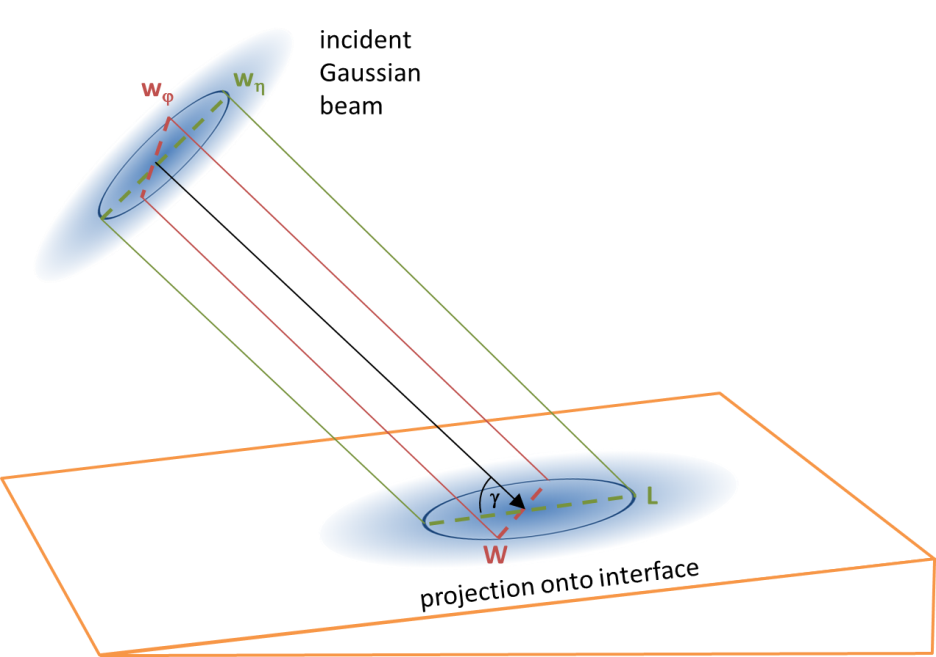


Figure 9 – Gaussian intensity projection onto interface

If we express the projected Gaussian in matrix form, the intensity profile of the area around the point of reflection “s”, becomes

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

**where**

= source level intensity;

= location on the interface, in vector form;

= transmission loss from the source to interface for this reflection;

= Gaussian profile in vector form;

= location vector for the source ray’s reflection;

= covariance matrix of the Gaussian projection onto interface;

= normalized Gaussian profile, in vector form; and

= intensity for this reflection on interface.

Applying a rotation to the length and width terms tilts the Gaussian on the interface [3]

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

where

= tilt angle by which length axis is rotated clockwise from true north;

= non-rotated covariance matrix; and

= rotation matrix.

Double angle identities reduce Eqn. (14) into a form that highlights the behavior when or

|  |  |  |
| --- | --- | --- |
|  |  | () |

Although this derivation has focused on the source to interface path, the expressions for in the path to the receiver are identical, except for the fact that the “s” subscript is replaced by “r”.

## Total energy integration

Eqn. (5) requires evaluation of an integral of the product of Gaussians. The product of two multivariant normalized Gaussians is given by [4]

|  |  |  |
| --- | --- | --- |
|  |  | () |

where

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

The product of two normalized Gaussians, is two different normalized Gaussians. The term gives the result a Gaussian shape in ‑space. The term acts as a scaling factor on the peak of the product. Using this identity, Eqn. (5) can be written in the form

|  |  |  |
| --- | --- | --- |
|  |  | () |

As does not depend on it can be brought outside of the integral in Eqn. (19). The remaining term inside the integral is a normalized Gaussian, which evaluates to one. Making these substitutions reduces Eqn. (5) to a form that does not depend on the location on the interface,

|  |  |  |
| --- | --- | --- |
|  |  | () |

We consider defining each source area relative to the current receiver area

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

where

= length and width of the receiver area;

= length and width of the source area;

= location of the source area relative to receiver area; and

= clockwise tilt angle of the source area relative to receiver area.

The sum of the covariance matrices is

|  |  |  |
| --- | --- | --- |
|  |  | (25) |

After some simplification, the determinant and the inverse of Eqn. (25) are

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | |  | () | |
|  | |  |  | |
|  |  | | | () |

The analytic solution for energy of the bistatic reverberation contribution for a single source/receiver reflection combination becomes

|  |  |  |
| --- | --- | --- |
|  |  | () |
|  |  |  |

where the term in the outer {} brackets is the scalar sum of all three lines and not a vector. Note that when **, Eqn.** (28) **can be simplified to**

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

where and are the horizontal range to the center of each patch, and the scattering strength dependence on grazing angle has been made explicit.

## Duration of reverberation contributions

The duration of the signal from each ensonified patch can be approximated by treating the echo as the convolution of a Gaussian pulse, with a Gaussian target response.

|  |  |  |
| --- | --- | --- |
|  |  | () |

where

= duration of the transmitted pulse;

= length of the ensonified patch along the line of sight of the receiver;

= grazing angle for the ensonified patch; and

= speed of sound at the center of the ensonified patch; and

= duration for this contribution.

Note that the factor of 2 in the term is the result of the fact that duration is measured in terms of two-way travel time, while the other terms are expressed in terms of one way time.

If we estimate the size of the ensonified patch from the extent of in Eqn. (16), along the length of the receiver reflection and solve for , Eqn. (32) becomes

|  |  |  |
| --- | --- | --- |
|  |  | () |

where

= variance of the Gaussian overlap along the path to the receiver;

= grazing angle for the path to the receiver; and

= speed of sound at the receiver path point of reflection;

The length of along the receiver path is given by Eqn. (17).

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

The determinant is then

|  |  |
| --- | --- |
|  | (37) |

Factoring a gives us

|  |  |  |
| --- | --- | --- |
|  |  | (38) |
|  | . | () |
|  |  |  |

We then have the complete formulae for the covariance matrix for the overlap of Gaussians

|  |  |  |
| --- | --- | --- |
|  |  | () |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  | () |

Note that f**or a monostatic scenario , and Eqn.** (41) **can be simplified to**

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |
|  |  |  |

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

## Computerimplementation

Calculation of the reverberation envelope using the Eigenverbmodel requires the following steps:

* Use WaveQ3D to propagate a wavefront from the source, and record its collisions with the interfaces to create values for and the locations of these source areas.
* Propagate a wavefront from each receiver to create values for and the locations of these receiver areas.
* Compare each receiver area to every source area and calculate a value for using Eqn. (28). Note that, in monostatic scenarios, source areas can be re-used to represent receiver areas without the need to propagate an additional wavefront. However, the process of comparing each source area to every other source area is still required.
* Create a reverberation intensity contribution as a function of two-way travel time, using Eqns. (6), (28), and (32).
* Combine the effects from all parts of the interface by computing the incoherent sum over all contributions.

The calculation of the reverberation envelope with the new model is still a complicated computation. However, unlike the classic approach, it reuses the data from interface reflections instead of calculating transmission loss to explicit acoustic targets on the interface. We believe that this approach will create a significant computation savings and improved execution speeds.

## Volume reverberation

Although this approach focuses on interface scattering, it also supports volume reverberation from deep scattering layers. It requires a change in WaveQ3D that allows it to detect collisions with the scattering layer without generating an actual reflection. Once a collision is detected, the process uses the same equations as those for interface scattering, with the scattering strength replaced by

|  |  |  |
| --- | --- | --- |
|  |  | () |

where

= volume scattering for this layer;

= thickness of the volume scattering layer;

= grazing angles for source and receiver reflections; and

= equivalent interface scattering strength for use in Eqn. (28).

This volume reverberation approach supports databases that express volume scattering strength in terms of one or more “deep scattering layers” that vary in depth, thickness, and strength as a function of both location and time of day.

# ****Reverberation test scenario****

To test the accuracy of our approach, we start with test scenario (Figure 10) with a monostatic sensor, a flat bottom, and a constant sound speed. This environment has simple analytic solutions for all of the eigenrays components. The symbols used in this scenario are

= water depth;

= horizontal range from source to scattering patch on bottom;

= speed of sound in water;

= depression/elevation angle at source (D/E);

= azimuthal bearing angle at source (AZ);

= number of segments in each path (where n=1, 3, 5, …);

= total length for each path;

= one-way travel time from source to bottom;

= one-way transmission loss time from source to bottom; and

= incident grazing angle at the bottom.

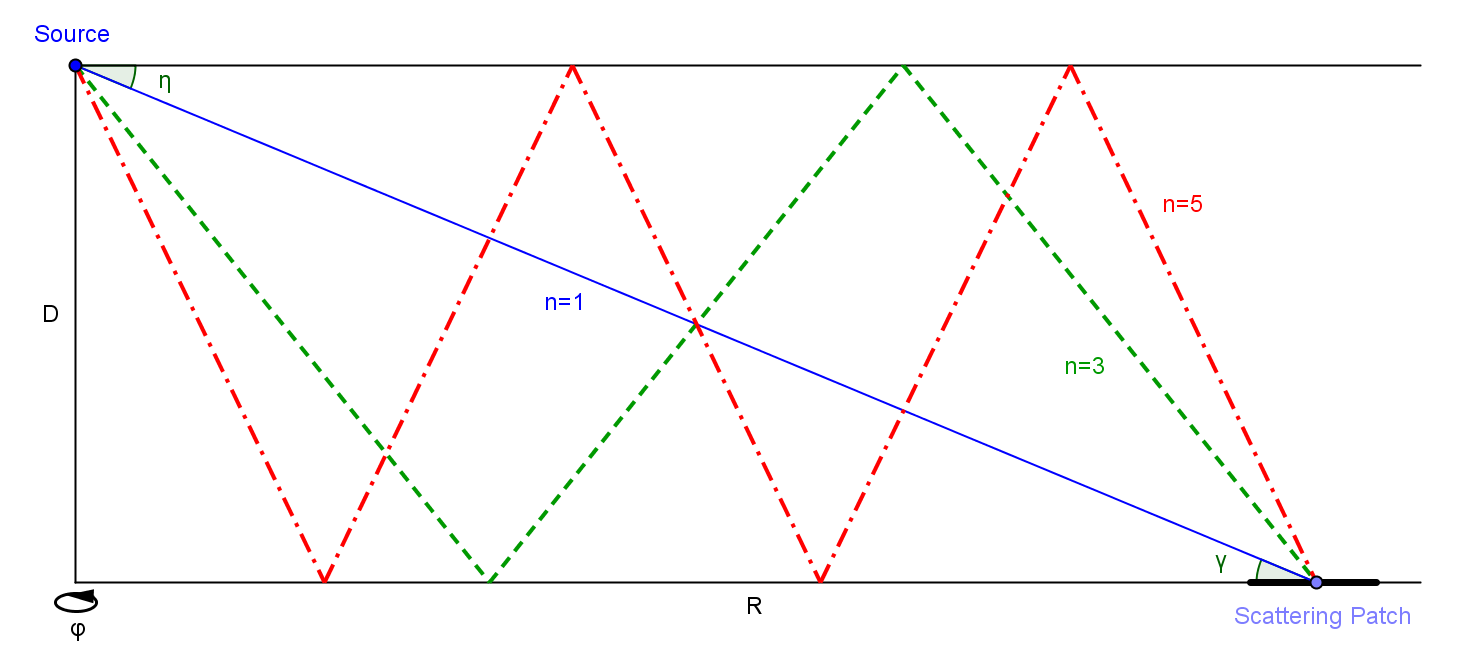
****

Figure 10 – Simplified test environment

In this scenario, the surface acts as perfect reflector and bottom loss is modelled as a function of grazing angle. The source is an omnidirectional, monostatic transmitter and receiver. Each path illustrated in Figure 10 is assumed to consist of a pair of paths, where the second path encounters a surface reflection **at the source location.** Given these definitions, analytic solutions for the eigenrays products as a function of patch range and the number of path segments are:

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

|  |  |  |
| --- | --- | --- |
|  |  | () |

The bottom loss for this scenario, illustrated in Figure 11, is based on a Rayleigh reflection model with typical values for a highly reflective sandy bottom: sound speed ratio of 1.1, a density ratio of 1.9, and an attenuation of 0.8 dB/wavelength. Bottom loss is very low below the critical angle at 24o, where it would be perfectly reflecting if there was no attenuation in the bottom.



Figure 11 – Reflection loss for the sandy bottom

Figures 12-14 illustrate the analytic solutions for one way propagation in this environment, with a water depth of 200 meters, and a sound speed of 1500 m/s. The one-way travel time has a lower bound of R/c and increases as the number of segments increases. The incident grazing angle decreases as a function of range and increases as the number of segments get large. The transmission loss clearly has a bound of 10log(R), and signals get quieter as the number of segments increases. The bottom loss components of transmission loss have the strongest impact at short ranges, where the grazing angle is the largest.



Figure 12 – Analytic solution for one-way travel time



Figure 13 - Analytic solution for incident grazing angle



Figure 14 - Analytic solution for one-way transmission loss

## ****Classic algorithm solution****

**The enclosed program** classic\_reverb.m **(Appendix A) is a Matlab script that computes an analytic solutions for the reverberation envelope from eqn. (5), using the classic reverberation calculation, for the simplified monostatic scenario shown in** Figure 10**.** Figure 15 **illustrates the results of this bottom reverberation calculation for a 250 millisecond pulse with a peak intensity of 200 dB.**

**The colored lines in** Figure 15 **represent the individual reverberation contributions from each unique combination of path types. Because the scenario is monostatic and there is no beam pattern modelled, the individual contributions each have an identical sibling whenever .**

**From 0.267 to 0.533 seconds, the only contributions to the total reverberation are from direct path propagation. In this zone, the total reverberation is 3 dB higher than the direct path because it includes the contribution of a surface reflected path. The ramp-up to the peak level at 0.367 seconds is the result of the convolution with the 250 millisecond transmit pulse.**

**Between 0.533 and 0.800 seconds there is a second smaller peak in the reverberation that represents the onset of four contributions from direct/surface paths. Beyond 0.800 seconds, the reverberation is a combination of many weaker paths. At the 4 second point, the reverberation is approximately 19 dB stronger than would be predicted by only modeling the direct path.**



Figure 15 – Reverberation envelope using classic reverberation calculation



Figure 16 – Ensonified area for classic reverberation algorithm

The ensonified area for the reverberation contributions in Figure 15 are illustrated in Figure 16. In each case, the ensonified areas for the simplified test environment are approximately equal to that for a horizontally propagating wave

|  |  |  |
| --- | --- | --- |
|  |  | () |

Figure 16 displays ensonified area as a ratio relative to , expressed in decibels. When the signal returns along the same path as the transmission, the area ramps-up to the value almost immediately. When the signal returns along a different path, the ramp-up is slightly slower.

## Eigenverb prototype and testing

**The enclosed program** eigenverb\_demo.m **(Appendix B) is a Matlab script that implements the** Eigenverbalgorithm (Section 3.5) **for the monostatic scenario shown in** Figure 10**. Unlike the classic algorithm, which computes** eigenray products (Eqns. (47) through (50)) as a function of the range to the target, **the** eigenverb\_demo.mscript computes eigenrays as a function of depression/elevation launch angle. Source eignerays are re-used to represent receiver paths, and a double loop creates the combination of source/receiver paths required to calculate intensity contributions as a function of time using **Eqns.** (6)**,** (7)**,** (31)**,** (32)**, and** (45)**.**



Figure 17 – Reverberation envelope using Eigenverb model



Figure 18 – Difference between Eigenverb and analytic solution

Figure 17 **illustrates bottom reverberation for a 250 millisecond pulse with a peak intensity of 200 dB. The solid lines represent the results of the Eigenverb model; the dashed lines represent the same analytic solution results shown in** Figure 15**, the classic reverberation model. The colored lines represent the individual reverberation contributions from each unique combination of path types; the black line represents the total reverberation.** Figure 18 **illustrates the numeric difference between Eigenverb and the** analytic solution**.**

To evaluate the accuracy of the Eigenverb model relative to known standards, CASS 4.2a was used to compute reverberation levels for the same **monostatic scenario using the GRAB transmission loss model. The run script for CASS is enclosed in Appendix C. To match the number of ray path segments in our analytic solution, the number of bottom reflections is artificially limited to two. Figure 19 illustrates the reverberation envelop for three models. Figure 20 compares the results of the Eigenverb model with the CASS/GRAB model for the simplified test environment.**

**In the times near the peak at 0.517 sec of Figure 19, the Eigenverb model is broader and about 3 dB less intense than the classic reverberation** analytic solution**. This is a side effect of the fact that Eqn.** (6) **assumes that the received signal has a Gaussian form. In a real-world scenario, where all forms of reverberation where modelled, the area from 0.267 to 0.517 seconds would normally be dominated by the 1st fathometer, the specular reflection from the bottom. The times before the 1st fathometer Eigenverb incorrectly has a non-trivial contribution, but we expect this area to be dominated by surface reverberation in training applications. The Eigenverb model closely matches the shape and level of the classic reverberation** analytic solution **immediately following the fathometer. At times greater than 2 seconds, the Eigenverb solutions all appear to have a bias of approximately -0.5 dB. The source of this bias is not well understood at this time.**

**The CASS/GRAB model does not have contributions prior to the arrival of the 1st fathometer, but the level of the peak is about 2 dB higher than the** analytic solution**. The CASS model closely matches the shape and level of the** analytic solution **immediately following the fathometer, but exhibit a bias of about +2.0 dB at times greater than 2 seconds.**



Figure 19 – Comparison of reverberation models



Figure 20 – Difference between reverberation models

# Conclusion

This paper defines new model for computing bistatic reverberation called Eigenverb. This model reuses the data from interface reflections instead of calculating transmission loss to explicit acoustic targets on the interface. Because of this, we believe that Eigenverb will provide sonar training systems with a significant computation savings and improved execution speeds.

To evaluate the accuracy of this approach, we have created a Matlab implementation of this algorithm, in a simplified environment, and compared its results to both an analytic solution and CASS/GRAB. In this simplified environment, Eigenverb accuracy is comparable to CASS/GRAB. In a future effort, a C++ implementation of the Eigenverb will be created to tested the accuracy and speed of this model in a 3-D environment.

# References

|  |  |
| --- | --- |
| [1] | W. J. Powers, "Bistatic Active Signal Model -- an Extension of the Generic Sonar Model," Naval Underwater Systems Center, NUSC TM. 871025, New London, CT, 14 Feb 1987. |
| [2] | R. J. Urick, Principles of underwater sound, 3rd Edition ed., New York: McGraw-Hill, 1983. |
| [3] | D. Lischinski, "Multivariate Normal Distribution," [Online]. Available: http://www.cs.huji.ac.il/~csip/tirgul34.pdf. [Accessed 3 June 2014]. |
| [4] | K. B. Petersen and M. S. Pedersen, "The Matrix Cookbook," 14 November 2008. [Online]. Available: http://martrixcookbook.com. |

1. Appendix A – classic\_reverb.m

%%

% classic\_reverb.m

%

% Compute bottom reverberation using the classic method of collecting

% ensonified areas from isochrons. Uses a Lambert's Law scattering

% strength with a coefficient of -27 dB.

%

clc; clear ; close all

debug = 0 ;

speed = 1500.0 ; % speed of sound in water

bottom\_speed = 1.10 ; % speed of sound in bottom

bottom\_density = 1.9 ; % density of bottom

bottom\_atten = 0.8 ; % attenuation in the bottom

depth = 200.0 ; % water depth in meters

maxRange = 3000 ; % maximum plot range

maxTime = 7.0 ; % maximum reverb time

dphi = 2\*pi ; % angular extent in radians

dtime = 0.001 ; % time step in seconds

T = 250 ; % pulse duration (in samples)

bss = 10^(-27/10); % scattering strength

SL = 200 ; % source level (dB)

path = 1:2:5 ; % number of path segments

[range,path] = meshgrid( 0:1:6000, path ) ;

header = sprintf('depth=%.0f m SL=%d dB T=%.0f ms',depth,SL,T\*dtime\*1000);

% plot the model for bottom loss

angle = 0:90;

bl1 = -20\*log10( abs( ...

reflection( pi/2-angle\*pi/180, bottom\_density, bottom\_speed, 0.0 ))) ;

bl2 = -20\*log10( abs( ...

reflection( pi/2-angle\*pi/180, bottom\_density, bottom\_speed, bottom\_atten ))) ;

if ( debug )

figure ;

plot( angle, bl1, '--', angle, bl2, 'LineWidth',2); grid

xlabel('Grazing Angle (deg)');

ylabel('Bottom Loss (dB)');

legend({'no atten','atten=0.8'},'Location','NorthWest');

title(sprintf('speed ratio = %.2f density ratio = %.2f',bottom\_speed,bottom\_density));

end

% compute the one-way eigenray components

path\_length = path .\* sqrt( depth.^2 + (range./path).^2 ) ;

angle1 = atan2( path .\* depth, range ) ; % launch and grazing angle

time1 = path\_length ./ speed ; % one way trvel time

loss1 = abs(reflection(pi/2-angle1,bottom\_density,bottom\_speed,bottom\_atten)) ...

.^ (path-1) ./ path\_length.^2 ; % one way TL with bottom loss

% note that the squaring of bottom loss cancels the 1/2 term in (n-1)

range = range(1,:) ;

path = path(:,1) ;

if ( debug )

type = { 'n=1'; 'n=3'; 'n=5'; 'r/c' } ;

figure ;

plot( range', time1', range', range'/speed, 'k', 'LineWidth', 2 ) ; grid

xlabel('Range (m)');

ylabel('One Way Travel Time (sec)');

set(gca,'Xlim',[0 maxRange]);

legend(type,'Location','SouthEast');

figure ;

plot( range', angle1'\*180/pi, 'LineWidth', 2 ) ; grid

xlabel('Range (m)');

ylabel('Incident Grazing Angle (deg)');

set(gca,'Xlim',[0 maxRange]);

set(gca,'YLim',[0,90]);

type = { 'n=1'; 'n=3'; 'n=5' } ;

legend(type,'Location','NorthEast');

figure ;

plot( range', 10\*log10(loss1'), range(1,:)', -20\*log10(range(1,:))', 'k', 'LineWidth', 2 ) ; grid

xlabel('Range (m)');

ylabel('One Way Transmission Loss (dB)');

set(gca,'Xlim',[0 maxRange]);

set(gca,'YLim',[-80,-30]);

type{4} = '20logR' ;

legend(type,'Location','NorthEast');

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% use classic algorithm to compute reverberation

% compute the two-way reverberation components as function of time

time2 = 0:dtime:2\*max(time1(:));

range2 = NaN\*ones( length(path)^2, length(time2) ) ;

loss2 = range2 ;

angleI = range2 ;

angleS = range2 ;

area = range2 ;

reverb = 1e-20 \* ones(size(range2)) ;

total\_reverb = 1e-20 \* ones(size(time2)) ;

type = cell( length(path)^2, 1 ) ;

n = 0 ;

for s=1:length(path)

for r=1:length(path)

n = n + 1 ;

t = time1(s,:) + time1(r,:) ;

range2(n,:) = interp1( t, range, time2 ) ;

loss2(n,:) = interp1( t, loss1(s,:) .\* loss1(r,:), time2 ) ;

angleI(n,:) = interp1( t, angle1(s,:), time2 ) ;

angleS(n,:) = interp1( t, angle1(r,:), time2 ) ;

type{n} = sprintf('s%dr%d',path(s),path(r));

% compute ensonified area as A = r dr dphi

m = find( ~isnan(range2(n,:)) ) ;

area(n,m) = range2(n,m) .\* [ diff(range2(n,m)) NaN ] .\* dphi ;

% compute reverb as RL = TL1 + TL2 + 10\*log( BSS \* A )

scattering = bss .\* sin(angleI(n,m)) .\* sin(angleS(n,m));

reverb(n,m) = loss2(n,m) .\* scattering .\* area(n,m) ;

reverb(n,:) = filter(ones(1,T), 1, reverb(n,:) ) ;

total\_reverb(1,m) = total\_reverb(1,m) + 2 \* reverb(n,m) ;

end

end

reverb = SL + 10\*log10(reverb);

total\_reverb = SL + 10\*log10(total\_reverb);

ideal\_area = (speed/2).^2 \* time2 \* dtime \* dphi ;

m = [ 1 2 3 5 6 9 ] ; % unique entries

% m = 1:length(reverb(:,1));

save classic\_reverb time2 reverb total\_reverb type ...

speed depth dphi dtime bss SL

if ( debug )

figure ;

plot( time2, 10\*log10(loss2(m,:)), 'LineWidth', 2 ) ; grid

xlabel('Two Way Travel Time (sec)');

ylabel('Two Way Transmission Loss (dB)');

set(gca,'Xlim',[0 maxTime]);

legend(type{m},'Location','NorthEast');

title(header);

figure ;

plot( time2, 10\*log10(area(m,:))-10\*log10(ones(length(m),1)\*ideal\_area), 'LineWidth', 2 ) ; grid

xlabel('Two Way Travel Time (sec)');

ylabel('Ensonified Area - Ideal (dB)');

set(gca,'Xlim',[0 maxTime]);

set(gca,'Ylim',[-0.25 0]);

legend(type{m},'Location','SouthEast');

title(header);

end

figure ;

plot( time2, reverb(m,:), time2, total\_reverb, 'k', ...

'LineWidth', 2 ) ; grid

xlabel('Two Way Travel Time (sec)');

ylabel('Reverberation Level (dB)');

legend({ type{m} 'total' },'Location','NorthEast');

title(header);

set(gca,'Xlim',[0 maxTime]);

set(gca,'Ylim',[55 135]);

% generate example of reverberation signal

if ( debug )

envelop = 10.^((total\_reverb)/10.0) ;

signal = envelop .\* randn(1,length(envelop)) ;

figure ;

h = plot( time2, signal, time2, envelop, 'r' , time2, -envelop, 'r' ) ; grid

set(h(2),'LineWidth',2);

set(h(3),'LineWidth',2);

set(gca,'Xlim',[0 1.4]);

set(gca,'Ylim',[-3e13 3e13]);

legend({'signal';'envelope'},'Location','NorthEast');

xlabel('Received Time (sec)');

ylabel('Reverberation Signal (uPa)');

title(header);

end

1. Appendix B – eigenverb\_demo.m

%%

% eigenverb\_demo.m

%

% Compute bottom reverberation using the Eigenverb model.

% Parallels the case implemented in the classic\_reverb.m script.

% In the classic algorithm, ranges are evenly gridded. But in this one,

% the ranges are determined by the launch angle of the ray.

% Rays are launched from a tangent spaced ray fan, like they would

% be in most WaveQ3D scenarios.

%

clc; clear ; close all

speed = 1500.0 ; % speed of sound in water

bottom\_speed = 1.10 ; % speed of sound in bottom

bottom\_density = 1.9 ; % density of bottom

bottom\_atten = 0.8 ; % attenuation in the bottom

depth = 200.0 ; % water depth in meters

linear = 0 ; % use linear fan if = 1, tan fan if ~= 1

num\_rays = 91 ; % number of rays to create

maxRange = 3000 ; % maximum plot range

maxTime = 7.0 ; % maximum reverb time

dphi = 2\*pi ; % azimuthal extent in radians

dtime = 0.001 ; % time step in seconds

T0 = 0.25 ; % pulse duration (in seconds)

bss = 10^(-27/10); % scattering strength

SL = 10.^(200/10) ; % source level (linear)

path = 1:2:5 ; % number of path segments

header = sprintf('depth=%.0f m SL=%d dB T=%.0f ms',depth,10\*log10(SL),T0\*1000);

% find the ranges appropriate to each ray in a tangent spaced ray fan

% Using same algorithm as seg\_rayfan.h, but limit to downward facing rays

if ( linear ) % use linear ray spacing

angle1 = (1:-1/(num\_rays-1):0)\*pi/2 ;

else % use tangent ray spacing (like seq\_rayfan.h)

spread = 6.0 ; % controls spacing of rays

first\_ang = atan( -90/spread ) ;

last\_ang = atan( 0/spread ) ;

scale = (last\_ang - first\_ang) / (num\_rays - 1) ;

n=1:num\_rays ;

x = first\_ang + scale \* (n-1) ;

angle1 = -tan(x)\*spread\*pi/180 ; % D/E angle with positive down

end

num\_paths = length(angle1) ;

[angle1,path] = meshgrid( angle1, path ) ;

range = depth .\* cot(angle1) .\* path ;

% compute the one-way eigenray components

% in the same format used for classic\_reverb.m

path\_length = path .\* sqrt( depth.^2 + (range./path).^2 ) ;

time1 = path\_length ./ speed ; % one way travel time

loss1 = abs(reflection(pi/2-angle1,bottom\_density,bottom\_speed,bottom\_atten)) ...

.^ (path-1) ./ path\_length.^2 ; % one way TL with bottom loss

% note that the squaring of bottom loss cancels the 1/2 term in (n-1)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% use Eigenverb algorithm to compute reverberation

% compute beam width in D/E using half distance to prev/next rays

mu\_half = 0.5 \* ( angle1(:,1:end-1) + angle1(:,2:end) ) ;

mu\_diff = -[ 2\*(mu\_half(:,1)-pi/2), diff(mu\_half')', -2\*mu\_half(:,end) ] ;

% compute length and width terms for each ensonified patch

% SL = SL / 2 ;

L = path\_length .\* mu\_diff ./ sin(angle1) ;

L2 = L .\* L ;

W = range .\* dphi ;

W2 = W .\* W ;

% loop over source and receiver path types

path = path(:,1)' ; % un-do meshgrid for path types

time2 = 0:dtime:maxTime ; % two-way travel time

reverb = 1e-20\*ones( length(path)^2, length(time2) ) ; % empty reverb

type = cell( length(path)^2, 1 ) ; % plot legend

n = 0 ; % index number for each contribution

for s=1:length(path)

for r=1:length(path)

n = n + 1 ;

type{n} = sprintf('s%dr%d',path(s),path(r)) ;

disp(type{n});

% loop over the patches in each path type

for sp=2:(num\_paths-1)

for rp=2:(num\_paths-1)

% compute integrated intensity for this path type combo

scatter = bss .\* sin(angle1(s,sp)) .\* sin(angle1(r,rp)) ;

S2 = L2(s,sp) + L2(r,rp) ; % sum of covariances

area = 0.5 \* L(s,sp) \* L(r,rp) \* W(s,sp) \* W(r,rp) ...

/ sqrt( S2 \* ( W2(s,sp) + W2(r,rp)) ) ...

\* exp( -0.5\*( range(s,sp) - range(r,rp) ).^2 / S2 ) ;

Esr = SL \* T0 \* loss1(s,sp) \* loss1(r,rp) \* scatter \* area ;

% only add to result if peak intensity is significant

if ( Esr > 1e2 )

Lp = sqrt( 1 / (1/L2(s,sp)+1/L2(r,rp) ) ) ;

Tarea = Lp \* sin(angle1(r,rp)) / speed ;

Tsr = sqrt( T0\*T0 + Tarea\*Tarea ) / 2 ;

t0 = time1(s,sp) + time1(r,rp) + Tsr ;

contrib = Esr \* exp(-0.5\*((time2-t0)/Tsr).^2) ...

/ ( Tsr \* sqrt(2\*pi) ) ;

reverb(n,:) = reverb(n,:) + contrib ;

end

end

end

end

end

% compute total reverberation in dB

total\_reverb = 10\*log10( 2\*sum(reverb) ) ;

reverb = 10\*log10( reverb ) ;

save eigenverb\_demo time2 reverb total\_reverb type ...

speed depth dphi dtime bss SL

% load classic result and trim to use the same times as Eigenverb

classic = load('classic\_reverb.mat');

n = find( classic.time2 <= time2(end) ) ;

classic.time2 = classic.time2(n) ;

classic.reverb = classic.reverb(:,n) ;

classic.total\_reverb = classic.total\_reverb(n) ;

eigenverb.total\_reverb = total\_reverb ;

eigenverb.reverb = reverb ;

eigenverb.time2 = time2 ;

% plot results side by side

m = [ 1 2 3 5 6 9 ] ; % unique entries

figure;

plot( eigenverb.time2, eigenverb.reverb(m,:), '-', eigenverb.time2, eigenverb.total\_reverb, 'k-', 'LineWidth', 2 ) ; grid

grid

xlabel('Two Way Travel Time (sec)');

ylabel('Reverberation Level (dB)');

legend({ type{m} 'total' },'Location','NorthEast');

set(gca,'Xlim',[0 7]);

set(gca,'Ylim',[-2 2]);

title(header);

set(gca,'Xlim',[0 7]);

set(gca,'Ylim',[55 135]);

hold on

plot( classic.time2, classic.reverb(m,:), '--', classic.time2, classic.total\_reverb, 'k--', 'LineWidth', 2 ) ; grid

hold off

% compute differences and plot them

reverb = eigenverb.reverb(m,:) - classic.reverb(m,:) ;

total\_reverb = eigenverb.total\_reverb - classic.total\_reverb ;

figure;

plot( time2, reverb, time2, total\_reverb, 'k', 'LineWidth', 2 ) ; grid

xlabel('Two Way Travel Time (sec)');

ylabel('Reverberation Difference (dB)');

legend({ type{m} 'total' },'Location','NorthEast');

set(gca,'Xlim',[0 7]);

set(gca,'Ylim',[-4 4]);

title(header);

1. Appendix C – CASS script

OUTPUT FILE = OUTPUTRESET OUTPUT DEVICEFREQUENCY MINIMUM = 3000 HZFREQUENCY MAXIMUM = 3000 HZCOMMENT TABLE Simple environment for comparing reverb to analytic result Bottom = flat 200 m Sound speed = 1500 m/s isovelocity No in-water attenuation Perfect surface reflection Bottom reflection = custom sandy bottom w/o shearEOTRADIUS OF CURVATURE = 99999999 KMBOTTOM DEPTH = 200 MSOUND SPEED TABLE = 1500 M/SVOLUME ATTENUATION MODEL = TABLEVOLUME ATTENUATION TABLE = 0 DB/KMSURFACE REFLECTION COEFFICIENT MODEL = TABLESURFACE REFLECTION COEFFICIENT TABLE = 0.0 DBBOTTOM REFLECTION COEFFICIENT MODEL = RAYLEIGHBOTTOM DENSITY = 1.9 GM/CM3 BOTTOM SEDIMENT ATTENUATION COEFFICI = 0.8 1/WL BOTTOM SOUND SPEED RATIO = 1.1BOTTOM SHEAR WAVE SPEED RATIO = 0.0VERTICAL ANGLE MINIMUM = 0 DEGVERTICAL ANGLE MAXIMUM = 90 DEGVERTICAL ANGLE INCREMENT = 1 DEGFUNCTION SYMBOL = BTM\_RFLFUNCTION UNIT = DBPRINT FUNCTION VS VERTICAL ANGLECOMMENT TABLE Monostatic source/receiver at surface 250 millisec CW pulse at 1000 Hz, 200 dBEOTSOURCE LEVEL MODEL = TABLESOURCE LEVEL TABLE = 200. DBRANGE REFERENCE = 1 MPULSE LENGTH = 0.25 SBANDWIDTH TABLE = 4 HZRECEIVER DEPTH = 0.01 MTRANSMITTER DEPTH = 0.01 MSOURCE DEPTH = 0.01 MVERTICAL ANGLE MINIMUM = -89.9 DEGVERTICAL ANGLE MAXIMUM = +89.9 DEGVERTICAL ANGLE INCREMENT = 0.1 DEG

COMMENT TABLE Eigenrays to bottom for reverberationEOT

RANGE MINIMUM = 0 KMRANGE MAXIMUM = 7 KMRANGE INCREMENT = 0.01 KM

BEARING ANGLE MINIMUM = 0 DEG BEARING ANGLE MAXIMUM = 0 DEG BEARING ANGLE INCREMENT = 360 DEG

MAXIMUM BOTTOM REFLECTIONS = 2TARGET DEPTH = BOTTOMEIGENRAY FILE = BOTEIGCOMPUTE EIGENRAYS

COMMENT TABLE Monostatic bottom reverberation for 7 secs Mackenzie bottom scattering strengthEOT

TIME MINIMUM = 0 STIME MAXIMUM = 7 STIME INCREMENT = 0.01 S

BOTTOM SCATTERING STRENGTH MODEL = MACKENZIEBOTTOM SCATTERING FACTOR = -27 DBACTIVE MODE = MONOSTATICREVERBERATION FILE = REVERBRESET REVERBERATIONEIGENRAY FILE = BOTEIGCOMPUTE BOTTOM REVERBERATION PRINT REVERBERATION VS TIME

END