Modeling Pulse Propagation and Scattering in a Dispersive Medium Using the Cray MTA-2

Dr. Guy Norton, *Naval Research Laboratory (Code 7181)*, **Wendell Anderson**, *NRL (Code 5593)*, **Dr. Jorge C. Novarini**, *Planning Systems Inc.* **Dr. Robert Rosenberg** *NRL (Code 5593)* and **Dr. Marco Lanzagorta**, *NCI Information Systems*

ABSTRACT *Accurate modeling of pulse propagation and scattering in a dispersive medium requires the inclusion of attenuation and its causal companion, dispersion. In this work a fourth order in time and space 2-D Finite Difference Time Domain (FDTD) scheme is used to solve the modified linear wave equation with a convolutional propagation operator to incorporate attenuation and dispersion. The MTA-2 with its multithreaded architecture and large shared memory provides an ideal platform for solving the equation, especially in the case when only a very small part of the medium is dispersive.*

KEYWORDS: MTA-2, wave equation, FDTD, propagation, dispersion, bubble plumes

1. Introduction

The presence of bubble plumes underneath the sea surface as the result of breaking waves changes the medium from weakly dispersive to highly dispersive. Accurate modeling of pulse propagation and scattering in such a medium therefore requires the inclusion of attenuation and its causal companion, dispersion. The plumes are also a source of additional scattering, usually only amenable to approximate solutions upon invoking strong approximations and neglecting attenuation and dispersion. The ability to incorporate attenuation and dispersion directly in the time domain has until recently received little attention.

For acoustic propagation in a linear medium, Szabo[1] introduced the concept of a convolutional propagation operator that plays the role of a causal propagation factor in the time domain. Waters *et al.* [2] showed that Szabo's operator could be used for a broader class of media, provided the attenuation possesses a Fourier transform in a distribution sense. Norton and Novarini [3] clarified the use of the operator and demonstrated its validity by solving the inhomogeneous wave equation including the causal convolutional propagation operator via a finite-difference time-domain (FDTD) scheme. It was shown that the inclusion of the operator in modelling propagation in a homogeneous medium correctly carries the information on attenuation and dispersion into the time domain. As a measure of the effectiveness of using the local operators to model a spatially varying medium[4], the convolutional propagation operator was then used to model 2-D pulse propagation in the presence of an interface between two dispersive media via FDTD. The inclusion of the local convolutional propagation operator correctly carries information on attenuation and dispersion into the time domain for both the backscattered and transmitted fields.

In this work a fourth order (in time and space) 2-D Time Domain Finite Difference scheme, which includes attenuation and dispersion via the convolution operator, is used to model backscattering of broadband signals from a composite sub-surface bubble cloud composed of two different plumes beneath a random rough sea-surface. This is a complex problem for which exact analytical solutions are not available.

2 The Convolutional Propagation Operator

Assuming that propagation occurs through an isotropic lossy linear medium, the propagation is governed by a modified wave equation of the form

$$\nabla^2 p(r,t) - \frac{1}{c_0^2}\frac{\partial^2 p(r,t)}{\partial t^2} - \frac{1}{c_0} L_\gamma(t) * p(r,t) = \delta(r - r_s)s(t)$$

where c_0 is a reference velocity (usually the thermodynamic sound speed in the medium is assumed lossless), *s(t)* is the source signature at location r_s, and $L_\gamma(t)$ is the causal convolutional operator, which controls the attenuation and dispersion and plays the role of a generalised dissipative term in the time domain. In the framework of generalised functions, assuming the pressure field is a distribution, this operator is defined as $L_\gamma(t) = \Gamma(t) * \delta^{(1)}(t)$.

The function $\Gamma(t)$ is the kernel of the operator, and represents a causal time-domain propagation factor that accounts for causal attenuation. That is, it also governs the dispersion in the system in order to insure causality. The wave equation can be rewritten as

$$\nabla^2 p(r,t) - \frac{1}{c_0^2}\frac{\partial^2 p(r,t)}{\partial t^2} - \frac{1}{c_0}\frac{\partial\left(\Gamma(t) * p(r,t)\right)}{\partial t} = \delta(r - r_s)s(t)$$

Szabo [1] defined the causal time-domain propagation factor $\Gamma(\tau)$ as the Fourier transform of the dispersive component of the complex propagation factor and showed it is given by

$$\Gamma(\tau) = -2\, 1_+(\tau) FT^{-1}\{\alpha(\omega)\}$$

where $\alpha(\omega)$ is the attenuation as a function of frequency for the medium and $1_+(\tau)$ represents the step function.

3. NUMERICAL MODELLING

3.1 Finite-Difference Scheme

The solution of the wave equation was originally expressed in terms of finite-differences by using the classical explicit second-order scheme in time and fourth-order in space. However, to make the algorithm uniformly fourth-order accurate, the second partial of the field with respect to time had to be extended to fourth-order. The usual fourth-order finite-difference representation of the second partial derivative can lead to unconditionally unstable schemes. A technique presented by Cohen [5] that is based on the "modified equation approach" was used to obtain fourth-order accuracy in time. This technique while improving the accuracy in time preserves the simplicity of the second-order accurate time-step scheme. Absorbing Boundary Conditions (ABCs) were imposed at the end of the numerical grid and at the corners. A technique named the Complementary Operators Method (COM) was employed [6]. The COM method is a differential equation-based ABCs. This differs from the other common approach of terminating the grid with the use of an absorbing material. An example of this type of boundary condition is the Perfectly Matched Layer (PML) method originally proposed by Berenger [7].

It is beyond the scope of this paper to present the derivation of either the fourth-order accurate time derivative algorithm or the implementation of the COM operators. The interested reader should refer to the given references. However we do present the equations that result from the application of the FDTD method to the solution of the linear wave equation. The calculation is divided in two parts, the first from the initial two terms of the equation that is calculated at every point of the grid, and another part from the third term that is present only for grid points

that are dispersive. From the first two terms we have

$$p_{i,j}(t+1)=2.0*p_{i,j}(t)-p_{i,j}(t-1) +(c_{i,j}*dt*dt/(12*dx*dz)) * (60*p_{i,j}(t)+ 16*(p_{i+1,j}(t)+p_{i,j+1}(t)+p_{i-1,j}(t)+p_{i,j-1}(t)) -(p_{i+2,j}(t)+p_{i,j+2}(t)+p_{i-2,j}(t)+p_{i,j-2}(t))) +(c_{ij,}*dt*dt*c_{i,j}*dt*dt/(12*dx*dx*dz*dz)* (20*p_{i,j}(t) -8*(p_{i+1,j}(t)+p_{i-1,j}(t)+p_{i,j+1}(t)+p_{i,j-1}(t)) +2*(p_{i+1,j+1}(t)+p_{i-1,j+1}(t)+p_{i+1,j-1}(t)+p_{i-1,j+1}(t)) +(p_{i+2,j}(t) +p_{i-2,j}(t)+p_{i,j+2}(t)+p_{i,j-2}(t)))$$

Discretizing the third term leads to

$$cp_{i,j}(t+1)=dt*sum(cpo0(k)*u_{i,j}(t+1-k))$$

$$p_{i,j}(t+1)=p_{i,j}(t+1)+2.0*c_{i,j}*dt*dt*(25.0*cp_{i,j}(t+1)-48.0*cp_{i,j}(t)+36.0*cp_{i,j}(t-1)-16.0*cp_{i,j}(t-2)+3.0*cp_{i,j}(t-3))$$

3.2 The Environment and Experimental Configuration

The scattering from bubble plumes under the sea surface, with due regard to the irregular shapes involved and the presence of attenuation and sound speed dispersion, is a problem for which no analytical solutions are available. Accurate controlled laboratory experiments are virtually impossible due to the difficulties in scaling the process. Therefore, numerical modelling arises as the best candidate to approach the problem. In this work backscattering from a bubble cloud in the ocean, composed of two different bubble plumes is considered. The broadband source signal is a doublet, whose spectrum peaks at 3.5kHz. (The signal is 20dB down at 40Hz and 12.5kHz). Figure 1 sketches the basic geometry of the simulated experiment. The upper medium is assumed to be air (sound speed is 330m/s and density 1x10-3 kg/m3). The sound speed in the water is taken to be 1500m/s. Modeling is carried out in 2-D. The source and receiver are co-located.

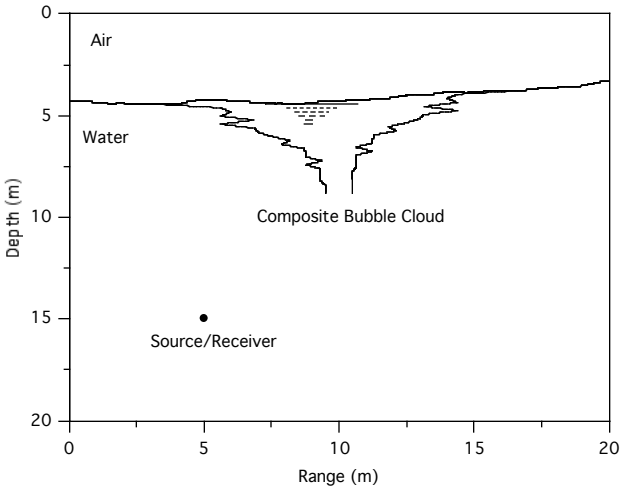

Figure 1: Geometry for the numerical experiments.

The environment contains a composite bubble cloud attached to a pressure release rough surface. Bubble plumes can be classified by their stage of development. The classification proposed by Monahan [8] as parameterised by Novarini et al. [9] is adopted. In this work both the so-called beta and gamma plumes are included (void fraction between 10^{-4} and 10^{-3} for the beta plume and 10^{-7} to 10^{-6} for the gamma plume). The bubble density within the plume as well as the horizontal area of the plume, (in this case, the equivalent horizontal length) is assumed to decay exponentially with depth. The beta plume subtends a length of 3.4m at the top and 0.2m at the bottom (1.2m below the air/water interface). The gamma plume spans 13.75m at the top and 1m at 4.4m below the air/water interface. The source and receiver are located at x = 5m, z = 15m. Each plume is discretized in layers of thickness dz = 0.2m. Random fluctuations in the bubble density, equivalent length vs. depth and e-folding depth are imposed. Assuming an 11m/s wind speed the attenuation vs. frequency is calculated within each layer. The concept of an effective medium is utilised, including thermal and viscous effects [9,10]. The attenuation (and the associated causal phase velocity) are calculated over 8192 points, at df = 40Hz. From the attenuation, the corresponding causal time-domain propagation factor is then generated at time steps dt = 3.05×10^{-6} sec. The attenuation in all layers is strongly frequency dependent. At 0.2m below the surface, for the beta plume it is

0.04 dB/l at 40Hz; 26.6 dB/l at 3.5kHz and 24.8 dB/l at 12.5kHz). The dispersion within the bandwidth of the source is also strong, for the beta plume, at 0.2m below the surface, the sound speed drops to 581m/s at 40Hz, 391m/s at 3.5kHz and 934m/s at 12.5kHz.

For the gamma plume the effect is weaker, but still noticeable. The attenuation at 0.2m below the surface is $4x10^{-3}$ dB/l at 40Hz; $4.7x10^{-2}$ dB/l at 3.5kHz and 0.1dB/l at 12.5kHz. Within each plume the sound speed increases with depth. At 3.5kHz, within the beta plume, the sound speed goes from 390m/s at the surface to 584m/s at the bottom of the plume. For the gamma plume it goes from 1488.8m/s at the surface to 1499.5m/s at the bottom. These values should be compared with those corresponding to a bubble-free medium, where the sound speed is frequency-independent (1500m/s) and the attenuation due to viscosity and molecular relaxation ranges from 1.510^{-5} dB/l at 40Hz; $6.4x10^{-5}$ dB/l at 3.5kHz and $1.2x10^{-4}$ dB/l at 12.5kHz. Also, in the bubble free case, the sound speed stays essentially constant within the thickness of the cloud.

The computational domain is 2000 horizontal grid points and 2000 vertical grid points. The dx equals the dz, which is 0.01m. The total number of solution grid points is $4x10^6$ and the number of grid points that the composite bubble cloud occupies is 188567. Each solution point within the composite plume will have an operator assigned to it (23 different operators in all) and the number of convolution points is taken to be 1000.

4. Code Optimization

The program to perform the calculation of the propagation of a signal was a parallel OpenMP Fortran 77 program. The original goal was to produce correct results with very little consideration in optimising the code with respect to performance. After an initialisation phase, the program enters a loop that updates on each pass the propagated values of the signal for the next time step. Most of the calculation time inside this loop is spent in a doubly nested `do` loop that updates the values at each of the grid points of the computational domain. The code was compiled to run in parallel by adding MTA specific derivatives to run the passes through the double nested do loops in parallel. These directives included one to indicate that updates to any grid point in the computational domain depended only on the values at the grid points calculated at earlier time steps.

The original optimisations included a test that was performed at each grid point to determine if the source had time to propagate to this point (and perform the update calculations only if it had) and code to ensure that the summation in the calculation of the causal effect is performed only over non-zero terms. Table 1 illustrates how these tests reduce the processing times of the earlier time steps when running on 40 processors of the MTA.

Steps	times (secs)
1-200	2.06
201-400	6.75
401-600	38.52
601-800	88.08
801-1000	138.13
1001-1200	187.60
1201-1400	236.04
1401-1600	257.76
1601-1800	256.67
1801-2000	256.35

Table 1. Time for every 200 time steps for original code on 40 processors.

Much of the running time of the "original" code was spent in the evaluation of the causal integral. An examination of the code revealed that this calculation contained a call to the Fortran `mod` function for each point in the sum.

```
cp(i,j,k)=0.
do n=1,ntau
  kount=mod(nloc+n-1,ncpo1)
  indx=nloc+kount+multi-1
  if(indx.gt.ncpo1)indx=indxncpo1
  cp(i,j,k)=cp(i,j,k)+
     cpo0(i,j,n)*p(i,j,indx+n-1)
end do
```

The mod function is needed because for each i and j the p array is stored in a circular buffer and the program needs to determine if the sum is taken over a set that crosses the end of the buffer. By breaking the sum into two separate parts when the sum crosses the buffer boundary (one to the end of the buffer and the second starting at the beginning of the circular buffer) the mod function can be moved outside the do loop.

```
multi=mod(k,ncpo1)
if(multi.eq.0)multi=ncpo1
nlocm1=ncpo1-multi
if(nlocm1.eq.0)nlocm1=ncpo1
kount=mod(nloc,ncpo1)
indx=nloc+kount+multi-1
if(indx.gt.ncpo1)indx=indx-ncpo1
cp(i,j,k)=0.
do n=1, ncpo1-indx+2
 m= indx-ncpo0+n-1
 cp(i,j,k)=cp(i,j,k)+
     cpo0(i,j,n)*p(i,j,m)
end do
do n= ncpo1-indx+2, ntau
 m= indx-ncpo0+n-1
 cp(i,j,k)=cp(i,j,k)+
     cpo0(i,j,n)*p(i,j,m)
end do
```

Incorporating this change into the program (the "improved" version) reduced the time for running 2000 time steps from 1468 seconds to 391 seconds a factor of 3.75 times improvement.

The MTA spreads the work of the nested do loops across processors by assigning sets of i,j values to individual threads. This would be very efficient if each block had the same amount of work. However for this application, the amount of work performed at a dispersive grid point is over fifty times more than at a regular grid point. Once the signal has propagated into a sufficient portion of the grid a load imbalance of the assignment of work to processor occurs. In order to better balance the load on the MTA the $MTA LOOP FUTURE directive is inserted before the do loops over the x and z dimensions of the grid. The MTA then treats the nested do as a single do loop of size 4 million and starts assigning a single pass through the loop to a single thread until all threads have work. When a thread completes, then the next pass through the loop is immediately assigned to that thread. This continues until all passes through the loop are completed. The use of the FUTURE directives (in the "future" version of the program) reduced the processing time for the two thousand time steps from 391 seconds to 122 seconds or 3.3 times faster.

An alternate way to balance the load is to split the nested do loops over the x and z dimensions of the grid into two separate loops the first loop a double nested do that performs the non-dispersive part of the calculation over all of the points and the second a loop only over the dispersive grid points to calculate the contribution from dispersion. In each case, the body of the loop contain the same amount of work per pass and the extra work required by the FUTURES approach. This method ("split" program) of load balancing reduced the running time from the 131 seconds of the future method to 95 seconds an improvement of 1.3.

The final program "new kernel" wa developed by using the CRAY canal utility to analyze the non-dispersive FDTD update. After trying several different ways of rewriting the doubly nested do loop that performs the non-dispersive update a better representation was found namely.

$$p_{i,j}(t+1)=e1_{i,j}*p_{j,j}(t) + e2_{i,j}*(p_{i+1,j}(t)+p_{i,j+1}(t)+p_{i-1,j}(t)+p_{i,j-1}(t)) + e3_{i,j}*(p_{i+2,j}(t)+p_{i-2,j}(t)+p_{i,j+2}(t)+p_{i,j-2}(t)) + e4_{i,j}*(p_{i+1,j+1}(t)+p_{i-1,j+1}(t))+p_{i+1,j-1}(t)+p_{i-1,j-1}(t))$$

where e1, e2, e3, and e4 are independent of the time step and need be computed only once. This reduced the cost of evaluating the update on a single grid point from 44 instructions requiring 33 memory accesses and 40 floating point operations to 21 instructions requiring 19 memory accesses and 19 floating point operations. This cut the time to perform the non-dispersive update by over 50% and the overall time from 95 seconds to 83 seconds.

The table below gives a summary of running the codes for 2000 steps for 4 and 40 processors

Code	4 proc	40 proc	Speedup
original	7042	1468	4.7
improved	2047	391	5.2
future	889	131	6.7
split	892	95	9.3
new kernel	784	83	9.4

Table 2. Time for 2000 time steps for 4 and 40 processors.

Further analysis of the running of the code on 40 processors was made by using `mtatop` (Cray's version of the UNIX `top` command) to measure the CPU utilization (in %), memory transfer rate, (in 64-bt words) and floating point (adds and multiplies) operation rate for each of the codes.

Code	Util %	GMops	GFlops
original	43	0.4	0.2
improved	46	0.6	0.4
future	90	0.9	0.7
split	96	0.9	0.7
new kernel	89	0.9	0.7

Table 3. CPU utilization, memory access rate, and floating point rates on 4 processors

Code	Util %	GMops	GFlops
original	25	0.7	0.5
improved	28	1.9	1.9
future	84	6.6	5.3
split	95	6.8	8.3
new kernel	89	7.5	7.1

Table 4. CPU utilization, memory access rate, and floating point rates on 40 processors.

From a previous study[11] for a case where the dispersive region made up only an extremely small part of the region (less than 1%) we had expected that all of the codes would show a high percentage of utilization. This was not the case for the original and improved versions of the codes for this case (where 5% of the region was dispersive). The poor cpu utilization was due to poor load balancing by the default assignment of the nested do's to threads. In the cases where we had finer grain assignment of loops to processors, each pass through the loops required nearly the same amount of work utilization as above 95%

5. Results

Results will be shown of the backscattered field collected at the source location as a function of time. In order to evaluate the impact that the composite plume has on the backscatter field, the backscattered field was generated for various combinations of plume composition. The first solution is when no plume is present with only the rough surface and serves as a benchmark. Next, the beta plume is added to the rough surface and the backscatter field is determined for this case. Then the beta plume is replaced with the gamma plume attached to the rough surface and its backscattered field determined. Finally the backscattered field for the composite bubble cloud, which consists of both the beta, and gamma plume attached to the rough surface is determined.

Figure 2 compares the backscattered field from the rough surface without any plume attached (solid line) to the case where the gamma plume is attached to the rough surface (dotted line). Note that the two results are nearly identical, indicating that the gamma plume even though spatially large has very little effect on the backscatter field. This is because the dispersive effects of this plume are very weak as the frequency dependent sound speed is very close to the non-dispersive sound speed of 1500 m/s. and the frequency dependent attenuation is very weak.

The backscattered field from the beta plume attached to the rough surface was compared to the backscattered field from the composite bubble cloud attached to the rough surface. However since no difference was observed, the comparison is not shown. This result though does support the results observed in Fig. 2 in that the gamma plume has very little effect of the backscattered field.

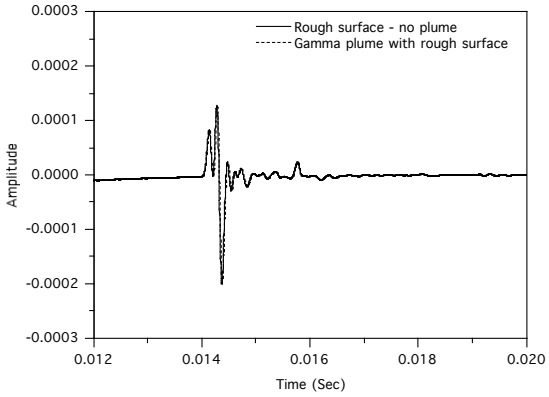

Figure 2: Comparison of the backscattered field from the rough surface without any plumes (solid line) with the backscattered field from the gamma plume attached to the rough surface (dotted line).

Figure 3 compares the backscattered field from the composite bubble cloud attached to the rough surface (dotted line) and the rough surface alone (solid line). There are observable differences for these two cases. This indicates that the beta plume is responsible for the differences observed. Even though the spatial extent of this plume is small, its dispersive nature is such that it has a major impact upon the backscattered field. It should be pointed out however that, although scattering by the beta plume is dominant, if calculations are repeated over an ensemble of surface realisations, as should be the case for a more complete analysis, the surface contribution to the total field would increase. And the backscatter will also be enhanced by the upper refractive effect inside the gamma plume.

Due to the fact that these fields are produced via numerical modeling, one can easily subtract out the effects of the rough surface leaving the effects solely due to the dispersive characteristics of the individual plumes. Thus Fig. 3 compares the backscatter results after the rough surface effects have been subtracted out of the time series leaving only the effects of the plume. First note the difference in amplitude of the backscattered field. It is approximately one third as strong as when the rough surface is present. Additionally note that the backscattered field from the beta plume (solid line) and the composite bubble cloud (dashed line) are nearly identical. This tracks our previous finding that the beta plume is the major contributor to the backscattered field. In addition note that the peak values associated with the beta plume (solid line) and composite bubble cloud (dashed line) is smaller than for the gamma plume (dotted plume). This is in accordance with the dispersive character of the plumes. The beta plume has a larger frequency dependent attenuation associated with it than does the gamma plume.

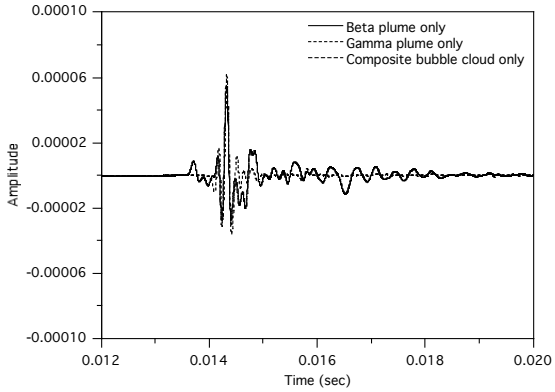

Figure 3: Comparison of the backscattered field with the rough surface effects removed for the beta plume (solid line), gamma plume (dotted line); and the composite bubble cloud (dashed line).

6 Conclusions

Due to the architecture of the MTA implementing the standard FDTD method of solving the propagation equation and including the effects of dispersion was very straightforward. In particular we did not have to worry about laying out the data to be either cache friendly or to be spread out across a set of distributed processors. The fastest implementation of the code (i. e. the one with the smallest wall clock time) operated at 90% utilization of the cpu and used 85% of he available bandwidth of the machine.

Acknowledgments

This work has been supported by the Office of Naval Research, (Program Element No. 61153N) and by a grant of computer time at the DoD High Performance Computing Shared Resource Center (Naval Research Laboratory, Washington, DC). The authors are deeply indebted to Jace Mogill of Cray Inc. for his willingness to share his knowledge and insights into the MTA-2 and its multithreading capabilities.

References

[1] **T. L. Szabo**, "Time domain wave equations for lossy media obeying a frequency power law," *J. Acoust. Soc. Am*, 96,pp. 491-500, 1994.

[2] **K. R. Waters, M. S. Hughes, G. H. Brandenburger, and J. G. Miller**, "On a time-domain representation of the Kramers-Kronig dispersion relations," *J. Acoust. Soc. Am.* 108, pp. 2114-2119, 2000.

[3] **G. V. Norton and J. C. Novarini**, "Including dispersion and attenuation directly in the time domain for wave propagation in isotropic media," *J. Acoust. Soc. Am.* 113, pp. 3024-3031, 2003.

[4] **G. V. Norton and J. C. Novarini**, "Including dispersion and attenuation in time domain modelling of pulse propagation in spatially-varying media," accepted for publication in *J. Comp. Acoust. Soc. Am*

[5] **G. Cohen**, *Higher-Order Numerical Methods for Transient Wave Equations*, Springer, pp. 35-63, 2001.

[6] **J. B. Schneider and O. M. Ramahi**, "The complementary operators method applied to acoustic finite-difference time-domain simulations," *J. Acoust. Soc. Am.*, 104, pp. 686-693, 1998.

[7] **J. P. Berenger**, "A perfectly matched layer for the absorption of electromagnetic waves," *J. Comput. Phys.*, 114, pp. 185-200, 1994.

[8] **E. C. Monahan**, "Occurrence and evolution of acoustically relevant sub-surface bubble plumes and their associated, remote monitorable, surface whitecaps," In *Natural Physical Sources of Underwater Sound*, pp. 503-516, 1993.

[9] **J. C. Novarini, R. S. Keiffer, and G. V. Norton**, "A model for variations in the range and depth dependence of the sound speed and attenuation induced by bubble clouds under wind-driven sea surfaces," *IEEE J. Ocean. Eng.*, 23, pp. 423-438 1998.

[10] **H. Medwin and C. S. Clay**, *Fundamentals of Acoustical Oceanography*, Academic Press, Chap. 8, 1998

[11] **W. Anderson and M. Lanzagorta,** "Effects of Tripling the Memory Bandwidth of the Cray MTA-2", *Cray Users Group,* 2004