REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave Blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	7 Oct. 96	Final report 11 Apr 96 - 11 Oct 96

4. TITLE AND SUBTITLE
Development of a User Friendly Wave Simulator for Military Applications

5. FUNDING NUMBERS
DACA39-96-C-0027

6. AUTHOR(S)
Amarendra Sinha and Dilruba Sultana

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
CMS Consulting
106 Abbot Avenue
Worthington, OH 43085

8. PERFORMING ORGANIZATION REPORT NUMBER
96-01

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U. S. Army Corp of Engineers Waterways Experiment Station
Coastal Engineering Research Center
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; SBIR report, distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*) The objective of the Phase I SBIR project was to demonstrate the feasibility of using Boussinesq type models to simulate time-dependent three-dimensional near-shore ocean waves intended to be used as a military planning tool. The main issues are the accuracy and the domain of applicability of the model, and the adequacy of current computers to simulate these models for a sufficiently large domain. We used Nwogu's form of Boussinesq model for the Phase I work. Following Wei and Kirby, we developed codes to simulate one and two-dimensional equations. We tested the one-dimensional code with an approximate analytical solitary wave solution and several monochromatic waves on flat or uniform slope bathymetry. We used the two-dimensional code to simulate one of the cases of the CERC elliptical shoal experiment of Vincent and Briggs. We compared the experimental and numerical time-series at several gage locations with good agreement between the two. We also made some sample runs on field scale problems with uniform slope bathymetry. We analyzed the performance of the code on several computers. These results show the feasibility of this approach.

14. SUBJECT TERMS	15. NUMBER OF PAGES
Amphibious landing operations Non-linear water waves, Boussinesq models, Numerical simulation	30 16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Notation:

We employ a orthogonal Cartesian coordinate system with: y-axis along the shoreline, x-axis pointing towards the shore and z-axis pointing upward. The wave propagation is assumed to be towards positive x-axis. As there are conflicting conventions, we frequently include explicit definition of dimensionality of equations and models: one-dimensional equation or code refers to one spatial dimension i.e. the model variables are functions of x and t only, with similar meaning for two-dimensional versions. Note that the wave surface predicted by a two-dimensional model is three-dimensional i.e. it includes the height information.

Introduction:

The omnipresence of waves has led to many experimental and theoretical studies to predict and simulate the water waves and their effects under various conditions as diverse as an open sea in a hurricane, the surf on a beach, waves in ports and harbors, waves in canals etc. This is a problem with enormous scope with many classes of physical phenomena. Therefore, the scope of this SBIR project is restricted to long waves in shallow water appropriate for applications to wind generated near-shore waves. All modern models of this type start with the governing equations for fluid flows i.e. the Navier-Stokes equations. It is possible to include many physical effects in these models. The SBIR solicitation identified three classes of models for this problem: models based on Navier-Stokes equation such as Volume Of Fluid (VOF) model, the Green-Naghdi type models and Boussinesq type models. We use these terms to denote classes of models rather than a single model. We also note that all of these models are non-linear. Until recently, most of the work was based on linear approximations. Of course, the linear approximations limit the range of phenomena that can be modeled. The recent advances in computers have allowed the study of many of the effects of non-linearity. This in turn has led to a flurry of activity and advances in non-linear science.

As noted in the SBIR solicitation, there are many potential applications of this simulation, both military and non-military. An immediate application of the simulation is to the planning of military exercises and operations such as amphibious landings. This simulation is very general in nature, and can be used in most applications involving coastal structures and processes.

Selection of a Model:

We begin with a discussion of the selection of a model. As anticipated in our Phase I proposal, we propose a solution based on the Boussinesq type models The VOF type models use the full three-dimensional form of the Navier-Stokes equation. Both Green-

Naghdi and Boussinesq models are specifically adapted for the water wave problem and reduce the dimensionality using approximation on the z-dependence. We now summarize the reasons behind our selection of the model.

- While the VOF type models based on full Navier-Stokes equation are capable of providing the most detailed and accurate information, they are also most resource intensive. This is primarily due to the three-dimensional (x, y, z and t) nature of these equations. For a comparable domain size and resolution, we estimate that the resource (memory and time) requirements of these models are hundreds or thousands of times greater than that of a Boussinesq type model. Moreover, due to the added dimension, the resource requirements grow at a faster rate than Boussinesq type models as the resolution or the domain size grows. We estimate that the technology required to adequately simulate the wave climatology of a region such as Camp LeJeune is many years away.
- The Green-Naghdi[7] model was derived using the theory of fluid sheets. Green and Naghdi[7] did not use a perturbation approach and thereby avoid any explicit smallness (e.g. wave height) assumption. However, they imposed (in our opinion, unacceptable) restrictions on velocity components: the vertical component is linear in z, and the horizontal components are independent of z. Another criticism of the Green-Naghdi model is that it fails to preserve vorticity. An attractive improvement was derived by Demirbilek and Webster[4,5] by allowing general polynomial dependence of velocity components on z. They presented the equations and a numerical code, GNWAVE, for one-dimensional version of Green-Naghdi Level II theory. This code was provided to us and indeed appears to work under wider circumstances than our one-dimensional Boussinesq code. However, even in the one-dimensional case, these equations are very complicated and their derivation requires involved algebraic manipulations. We estimate, based on our earlier experience with symbolic algebra systems, that the derivation of the equations of a two-dimensional Level II Green-Naghdi model will be extremely difficult (possibly unmanageable) even with symbolic algebra packages like MACSYMA, Mathematica or Maple. Furthermore, the complexity of the equations will impose significant precision requirement on the numerical methods. These facts combine to substantially reduce the advantage of lower dimension over VOF type models. Due to these reasons, there appears to be very little work in this area, in particular, there is essentially no comparisons with experimental data. We estimate that the development of such a model is significantly outside the scope of a SBIR project.
- The first modern Boussinesq type model was derived by Peregrine[18] using depth-averaged velocities and perturbation expansions. The approximations are valid for long waves in shallow water and the assumption of weak non-linearity limits the wave height and steepness. In particular, this model cannot be used in deep waters or in the break zone. For surface water waves, this method provides great deal of simplification with relatively little loss of accuracy. By its very nature, Boussinesq models provide limited information on the velocity components. Two closely related non-linear equations: the Kadomtsev-Petviashvili (KP) and the Korteweg-de Vries (KdV) equations have enormous theoretical interest as they are examples of

Completely Integrable Hamiltonian systems. Due to these reasons, this has been and continues to be an area of active research. As can be expected with this degree of research, many[3, 12, 15, 16, 17, 19, 23] improvements of Peregrine's original work has been proposed. The modifications proposed by Nwogu[15] improves the dispersion relation and extends the domain of validity to very close to the deep water limit where Stokes expansions can be used. Nwogu's equations uses the velocity at fixed distance from the still water level rather than the depth averaged values. Our Phase I work is based on Nwogu's version of the Boussinesq model. We use the numerical methods proposed by Wei and Kirby[22]. Several[3, 12, 15, 19, 23] improvements or enhancements of Nwogu's model have been proposed. We selected Nwogu's form of the model for the Phase I work because of it's general acceptance as a base model and it's relative simplicity. We intend to include the fully non-linear version(s) in Phase II. Both Nwogu[16], and Wei and Kirby[22] showed good agreement between experimental data and the model prediction. We performed comparisons of the predicted time-series data with some data from the CERC elliptical shoal experiment of Vincent and Briggs[21], with very good agreement. All this research provides a very strong validation for these models. In fact, one of the reasons of our selection of Boussinesq type model is the availability of research results, without which the progress made during Phase I would have been impossible and work proposed in Phase II would not be possible either. We emphasize that the proposed work is neither a duplication nor a substitute of the academic research that is being performed at various Universities and Institutes. Lastly, we have studied the resource requirements of our code. Based on all of these works, our conclusion is: *for the present and the near future, Boussinesq models provide the best option in providing reasonably accurate predictions on field-scale problems with the available resources.* Of course, eventually (in many years) models like VOF using full Navier-Stokes equations will be manageable for field-scale problems and will be the model of choice at that point.

Numerical Codes:

During Phase I, we developed prototype codes to numerically solve one and two-dimensional time-dependent Boussinesq models. We consider a section of the ocean in the computational domain $0 \le x \le a$ and $0 \le y \le b$, and the initial state is assumed to be quiescent. We wish to predict time-dependent three-dimensional form of the wave in this domain that is generated by a wave-train incident along the boundary at $x = 0$. Of course, the waves depend on the incident wave and the bathymetry data. The computational domain, the incident wave and the bathymetry data must fall within limitations of the model: long waves in shallow water. The boundaries at $x = a$, $y = 0$ and $y = b$ are assumed to be transparent to the wave i.e. we assume "open boundary condition" along these edges.

As noted earlier we use Nwogu's equations[15] for the Phase I work. This model extends the validity on the earlier versions of Boussinesq models and is simpler than some other

versions. Thus, this model provides a convenient starting point. In fact, the model prediction agrees quite well with several experimental situations. We use the numerical method proposed by Wei and Kirby[22]. This is a finite difference method with a fourth order predictor-corrector time stepping scheme. We follow this method very closely*. This method is described in detail in Appendix A, and we refer the reader to the paper by Wei and Kirby[22] and the report version of the paper for further discussion. Some of the detail (e.g. a flow diagram) in the report version was omitted from the journal publication. As is well known, the implementation of the open or absorbing boundary condition poses several difficulties. Lack of time has forced us to take some shortcuts in this respect. There are several problems in implementing the simplest form of the radiation boundary condition $\eta_t + c\cos\theta\eta_x = 0$: the celerity c is ill-defined in this dispersive situation and in the two-dimensional case the direction is not known *a priori*. We therefore used a simplified version $\eta_t + c\eta_x = 0$ both in one and two-dimensional cases along the $x = a$ boundary with some choice of the celerity c. For the $y = 0$ and $y = b$ boundaries in the two-dimensional version, we have used an even more primitive condition that y-derivatives vanish along these boundaries. The conditions imposed on the y-boundaries are appropriate if there are no incident waves at these boundaries, a condition that is at least approximately satisfied in all our examples. Our code has instability caused by the waves trapped by reflection at the boundaries. This limits the time interval that can be used in our simulations. We also observe that these instabilities are present in the GNWAVE code as well. The expected solution (see Wei and Kirby[22], Nwogu[15, 16], Abbott, McCowan and Warren[1], Engquist and Majda[6], Israelei and Orszag[11], Larsen and Dancy[13], Longuet-Higgins and Cokelet[14]) is: (i) use a higher order radiation condition and (ii) implement absorption layers at these boundaries. We have started the implementation of the higher order radiation condition (see equation (42) of Wei and Kirby[22]) for the two-dimensional code, however it remains to be debugged. It should be noted here that another commonly used boundary condition, the reflection boundary condition, appropriate for a canal, a gate or a wall etc., is simpler to implement and works quite well.

We have implemented a simple platform-independent animation scheme based on the public domain graphics packages Plplot or Pgplot. Similar animation was added to the GNWAVE as well. These codes provide near real time solution and visualization of the time-dependent wave in the one-dimensional case even in a Pentium based PC. The "on the fly" animation is very slow (in a PC) for the two-dimensional case with a reasonable resolution. As soon as the basic debugging was completed, we emphasized the case studies rather than any improvements of the code. In particular, we did not have time to implement the absorption layers or even the pre-factorization of the matrices in the two dimensional case. We were able to perform only a small number of case studies in this short period of time. However, there are many more case studies in the literature.

* After we completed the one-dimensional version of the code, the one-dimensional code developed by Wei and Kirby was publicly available. Due to lack of time, we have not yet been able to study the code extensively. However, we noticed a few minor differences in the implementation e.g. the boundary.

The reflections due to the simplified treatment of boundaries leads to trapped waves and instabilities. The trapped waves and the instabilities are visible (this applies to GNWAVE as well) in the animation when the execution continues significantly past the arrival of the wave-front at the near-shore boundary. This instability makes it difficult to perform long-time simulations necessary for accurate statistical analysis of waves with complex spectrum. Thus, we have not attempted any tests with incident waves with several spectral components. We have developed some prototype post-processing components for the visualization of snapshot of the waves and the statistical analysis of the time-series data. We now describe some simulations performed with these codes.

Remarks on the code:

The code is written in FORTRAN. We use several common extensions to the Fortran 77 standard, these extensions are included in the Fortran 90 standard. We have tested the code with Microsoft Fortran Powerstation compiler, f2c/fcc or g77 under Linux, Sun Fortran, Cray Fortran. The extensions are used to improve the readability, but may require modifications in certain computer-compiler combination. These extension include: (i) use of lower case letters, (ii) variable names may include the underscore character or longer than six characters, (iii) the do loops are terminated with `end do`, (iv) a `NAMELIST` group is used for input (v) the two dimensional code uses `REAL*8`. The code should use at least 8-byte real numbers, double precision on the PC, Sun, etc., and single precision on the Cray etc.

We practise meaningful variables names, and avoid the use one or two character variable names except for loop index etc. We use `eta` for the water displacement, `velx` and `vely` for the two components of the velocity. We use a prefix `wk_` for the variables used in Wei and Kirby[22], e.g. `wk_b1` refers to b_1 in equation (11) in Appendix A or `wk_u` refers to U defined in equation (4) of Appendix A. We generally avoid global variables and use of many common blocks. We use one common block for the model parameters and constants, defined in separate file and "included" in the code. Another common block is used for the input wave parameters. All arrays that depend on the resolution are declared in the main program. Any work space needed by a subroutine is passed to it as an argument.

We use some graphics based on the public domain packages Pgplot or Plplot. We include dummy (no output is generated) graphics routines if graphics libraries are not available or graphics is not desired. We use a few Linpack routines. Optimized Linpack routines are available in many machines. Again we include sources in case the Linpack libraries are not available.

The input is obtained from a `NAMELIST` group in a file, the variables in this grout are described in sample input files and "readme" files. The output consists of the displayed graphics (if any), time series data at selected locations, and snapshots of the solution at

particular instants of time. The code also displays the CPU time information for the run if a working version of the function `second` is available for the system

The code essentially follows the flow chart given in the report version of Wei and Kirby. We begin by defining a default set of values of the parameters. The input quantities are then read in. The `NAMELIST` read does not require all variables to be specified. If some variable is not included in the input, it retains the default value. We then call the subroutine `initialize` which (i) sets up the initial conditions, (ii) computes the bottom profile and its derivatives, (iii) wavelengths of the incident waves etc. We then initialize the graphics system and the work-space variables. Next the execution enters the main time loop which repeatedly call the subroutine `timestep` and updates the current time until the final time is reached. At some time steps we may perform some graphics operations; or write the time series or snapshot data. The subroutine `timestep` implements the predictor-corrector scheme described in Appendix A for one time step. We also use a filter to control the high frequency oscillations. A snapshot data is saved at the end of the run and the graphics system is closed.

One-dimensional Boussinesq code:

First simulation performed was the propagation of a solitary wave over a flat bottom. Wei and Kirby[22] derived an approximate analytical solution:

$$u = A\,\mathrm{sech}^2[B(x - Ct)]$$

$$\eta = A_1\,\mathrm{sech}^2[B(x - Ct)] + A_2\,\mathrm{sech}^4[B(x - Ct)]$$

to Nwogu's equations. Figures 1a and 1b show a comparison of numerical and the analytical solutions. Figure 1b is the magnified version of a portion of figure 1a. The close agreement between the two provides a strong validation of the code.

Next we consider an example by Nwogu: the propagation of a monochromatic wave with period $T = 0.85\,\mathrm{s}$ and height $H = 0.05\,\mathrm{m}$ propagating in a channel of uniform depth of 0.56 m. The results are shown in figure 2. Since, the wavelength is approximately twice the depth, this case demonstrates the extended validity of the model. Figure 2 shows both the computed solution and the incident wave. The approximate agreement between the two comes from the fact that the incident wave is a solution of the linearized equation, and small height of the waves. Increasing the wave-height shows the effects of non-linearity.

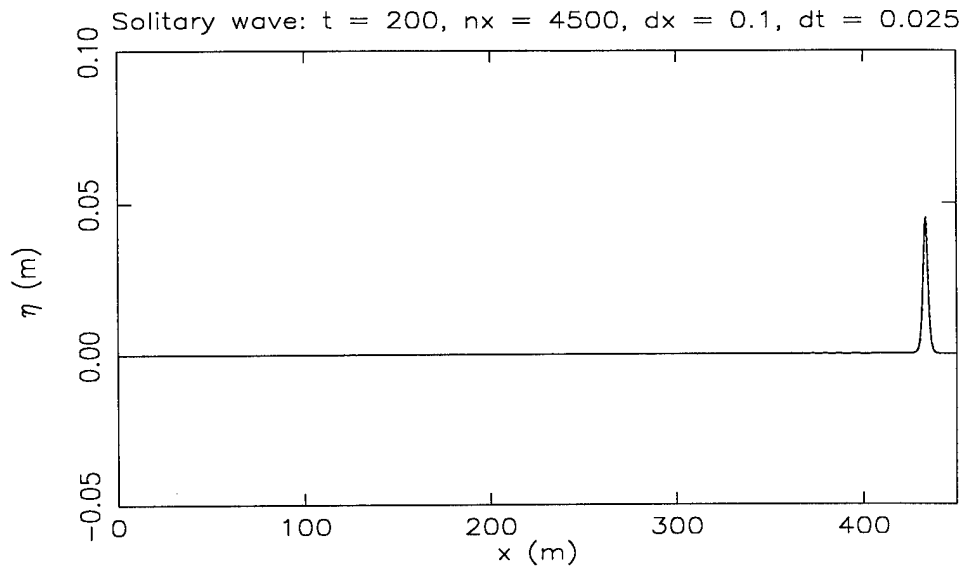

Figure 1a. Comparison of numerical (solid) and analytical (dash) solitary wave solution.

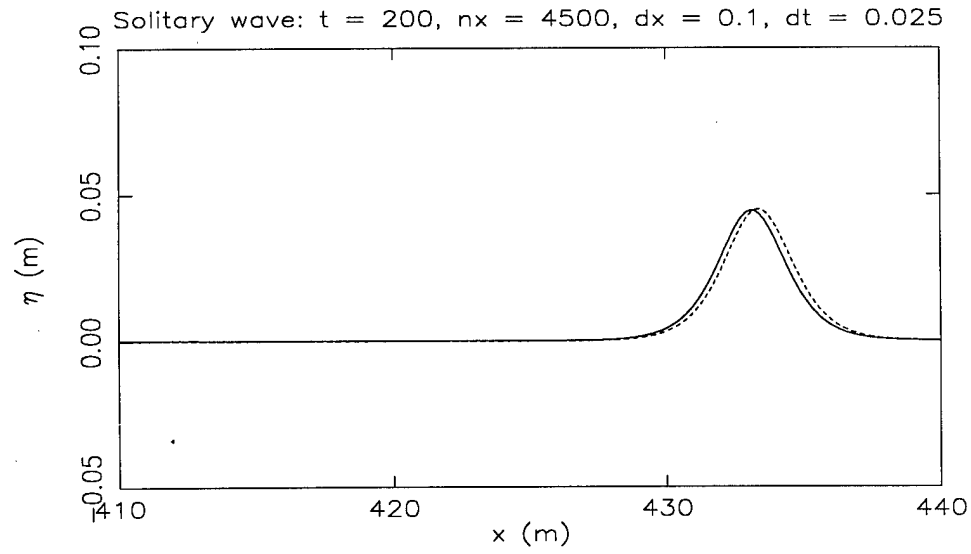

Figure 1b. A detailed comparison of the numerical (solid) and analytical solitary wave solution. Figures 1a and 1b refer to the same solution, i.e. figure 1b is a magnified section of figure 1a. The water depth in this example is 0.45 m.

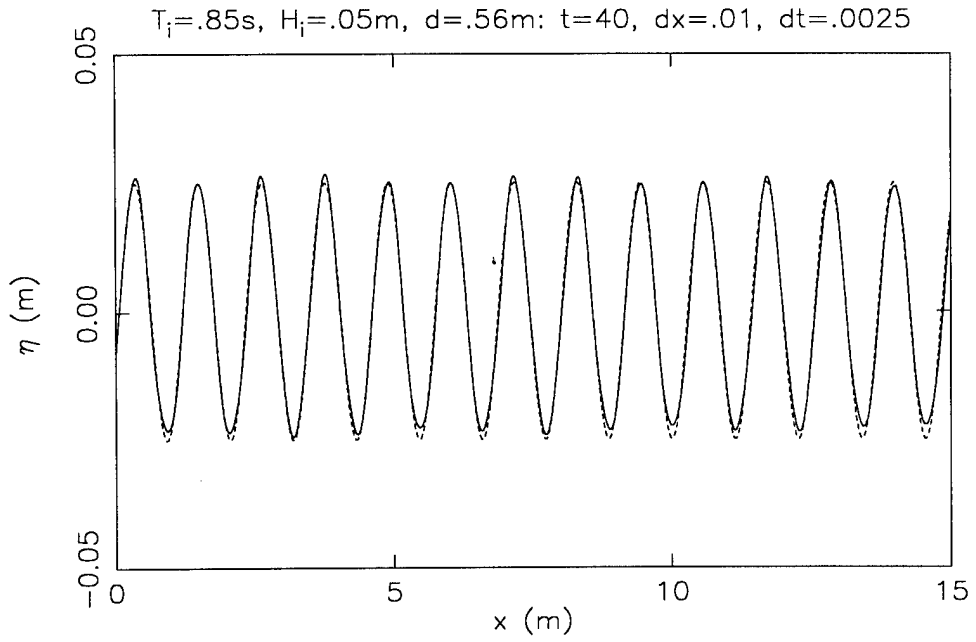

Figure 2. The solid curve depicts the numerical solution of the Boussinesq equation corresponding to the incident wave shown in dash. The incident wave is a solution of the linearized Boussinesq equation, so it is an approximate solution of the Boussinesq equation.

We conducted test runs with bathymetry of the CERC elliptical shoal experiment of Vincent and Briggs[21]. We used both our code and the Green-Naghdi model of Demirbilek and Webster[5]. Both of these codes show the general features of the wave up-to the shoal, but they fail to reproduce the features of the wave past the shoal. This can be expected, since the one-dimensional model does not include refraction and diffraction of the waves.

Two-dimensional Boussinesq code:

When a new model is proposed, it is judged by a variety of techniques. A common practice is to compare the model with the predictions of the linearized equations. While such agreement is necessary, usually it not a sufficient condition for the usefulness of the model. Ultimately, for physical models, the validity must be based on comparison with physical measurements. Indeed, most papers in this area contain some comparison of experimental data and the model predictions. The Boussinesq models agree well with such measurements. In Phase I, we were able to make such a comparison with one case of the CERC elliptical shoal experiment of Vincent and Briggs[21]. Any comparison with this experiment and a time-dependent model has not been reported in the literature. The time-series data from the experiment were provided by Dr. Zeki Demirbilek, Mr. Mike

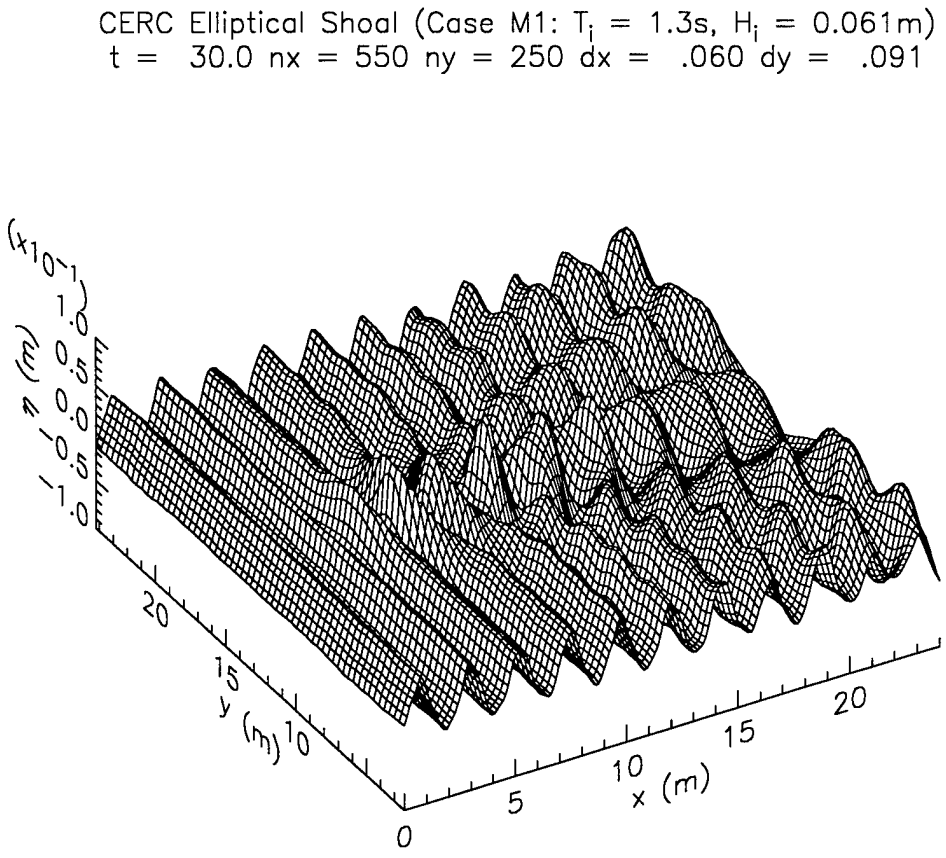

Figure 3. The computed water surface at time 30 sec. for the CERC elliptical shoal bathymetry and a monochromatic incident wave. The z-axis shows the displacement from the still water surface. The shoal center is located at (6.096, 13.72). The effects of refraction and diffraction is clearly visible in the figure.

Briggs and Ms. Deborah Green of CERC. The agreement between the model and the experiment is very good. We propose to perform more extensive comparison of the models with available laboratory experiments and field measurements during Phase II.

The experiment (CERC elliptical shoal experiment) by Vincent and Briggs[21] was conducted at the U. S. Army Engineer Waterways Experiment Station's Coastal Engineering Research Center (CERC). The shoal tests were conducted in CERC's 35 m wide by 29 m long directional spectral wave basin. The bathymetry was uniform with a maximum variation of 9 mm (0.35 in.). The elliptical shoal had a major radius of 3.96 m (13 ft), minor radius of 3.05 m (10 ft), and a maximum height of 30.48 cm (1 ft). All tests were run in 45.72 (1.5 ft) of water depth, so that the distance at the center of the shoal to still water level was 15.24 cm (0.5 ft). The waves were generated with the

directional spectral wave generator (DSWG). The wave measurements were taken at several gage locations on a rectangular grid.

The case considered here is the one labeled as Case ID: Lab M31 or Zeki M1. This case corresponds to a monochromatic input wave with period 1.30 sec. The incident wave-height of 2.56 in (0.065 m) was inferred from the measurements at gage #10 (see figure 5 for the locations of the gages). After analyzing the time-series data at these gage locations with a post-processing component developed during Phase I, we concluded that the time-series could be considered to be consistent with a range of values for the wave-heights, and therefore decided to perform the simulations with a range of values. To date we have performed simulations with only one set of parameters. This simulation has been run with several sets of resolutions (n_x, dx, n_y, dy, dt) and on several computers. The consistency of the results is a necessary step in the validation of the code. We have performed some other consistency checks as well.

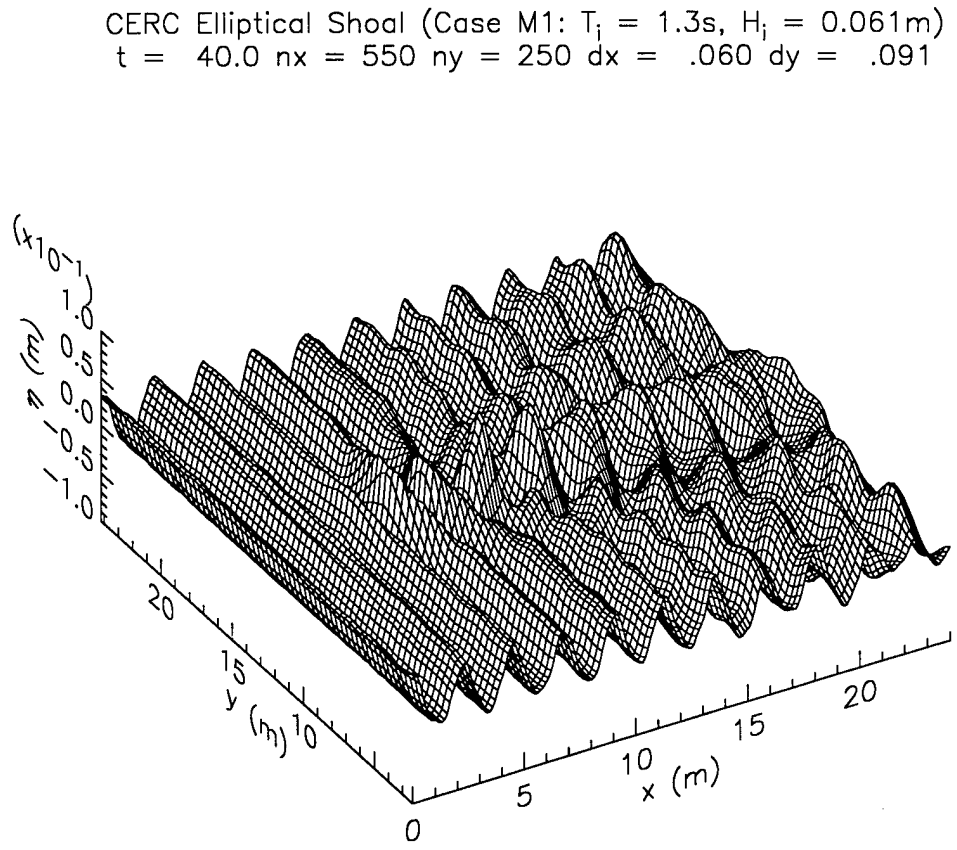

Figure 4. The computed water surface at 40 sec for the CERC elliptical shoal bathymetry. Figure 4, shows the same wave as in figure 3, but at a later time.

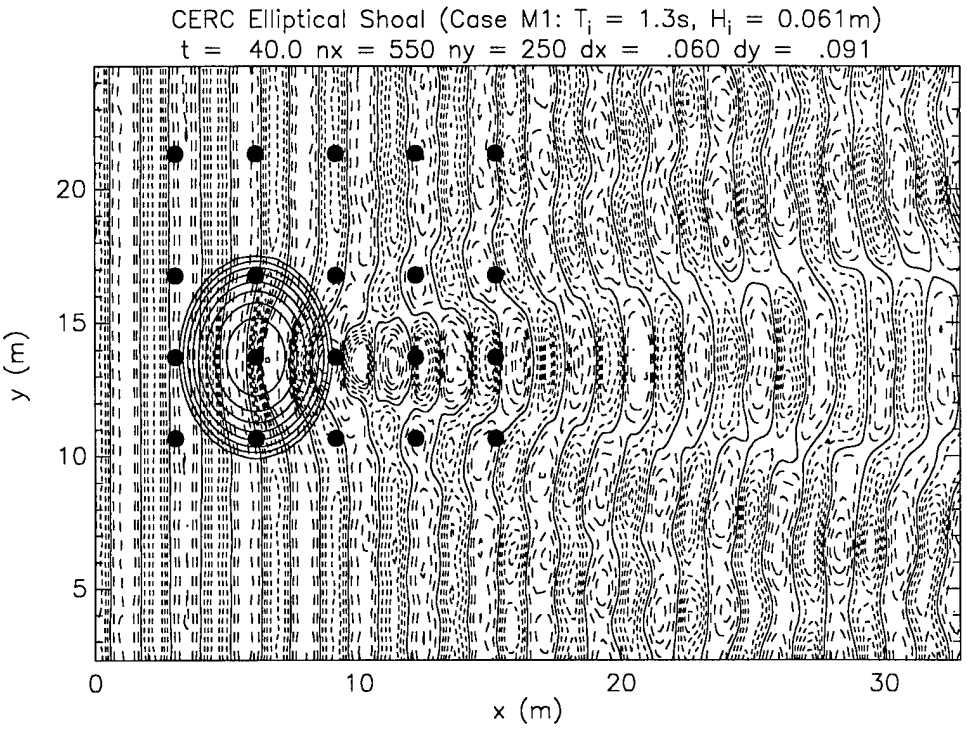

Figure 5. A contour plot of the computed water surface for the CERC elliptic shoal bathymetry at time 40 sec i.e. the surface in figure 4, is shown in this plot. The solid lines correspond to still water level, the long-dash contours correspond to water level below still water i.e. the troughs, and the short dash contours correspond to above still water level i.e. the crests. The contours are shown at 1 cm interval i.e. the contour lines correspond to water displacements of 0, ±1, ±2,..cm. The solid ellipses are contour lines of the shoal. The effects of refraction and diffraction on the shoal are clearly visible. The solid circles denote the gage locations. Starting form the bottom of the figure, in each vertical row, the gages are labeled as Gage #9, 5, 1, 10 respectively. The vertical rows of gages are referred to as Transects 1–5.

In figures 3 and 4, we present the computed three-dimensional wave surface at time 30 and 40 sec. respectively. In figure 5, we present a contour plot of the wave-surface. For reference, we show some contour lines of the shoal and the locations of the gages: Gage #9 corresponds to $y = 10.668\,\text{m}$ (bottom row in the figure), Gage #5 corresponds to $y = 13.716\,\text{m}$ (second row from bottom in the figure), Gage #1 corresponds to $y = 16.764\,\text{m}$ (third row from bottom in the figure) and Gage #10 corresponds to $y = 21.336\,\text{m}$ (top row in the figure). We use the term Transect 1, ··, Transect 5 for labeling down-wave locations of 10, 20, ··, 50 ft respectively (vertical rows in the figure). Thus, for example, Gage 5 Transect 3 refers to the location $x = 9.144\,\text{m}$, $y = 13.716\,\text{m}$. We follow the notation used in the data files provided to us. Please observe that the effects of refraction and diffraction over the mound is clearly visible in figures 3–5. The

simplified boundary condition imposed in our code assumes no waves are incident on the y-boundaries. It is also clear from these figures that this condition is at least approximately satisfied. The effects of reflection on the boundaries is visible in the computed solution at sufficiently later time.

In figures 6–10, we compare the experimental and the computed time-series data at the gage locations. We need to match the phases of the two sets of waves for this comparison. Since, the simulation starts in quiescent state, the comparison is meaningful only after the wave "attains the full wave-height". To this end, we look at the plots of the computed time-series and then match the phases by comparing the experimental time-

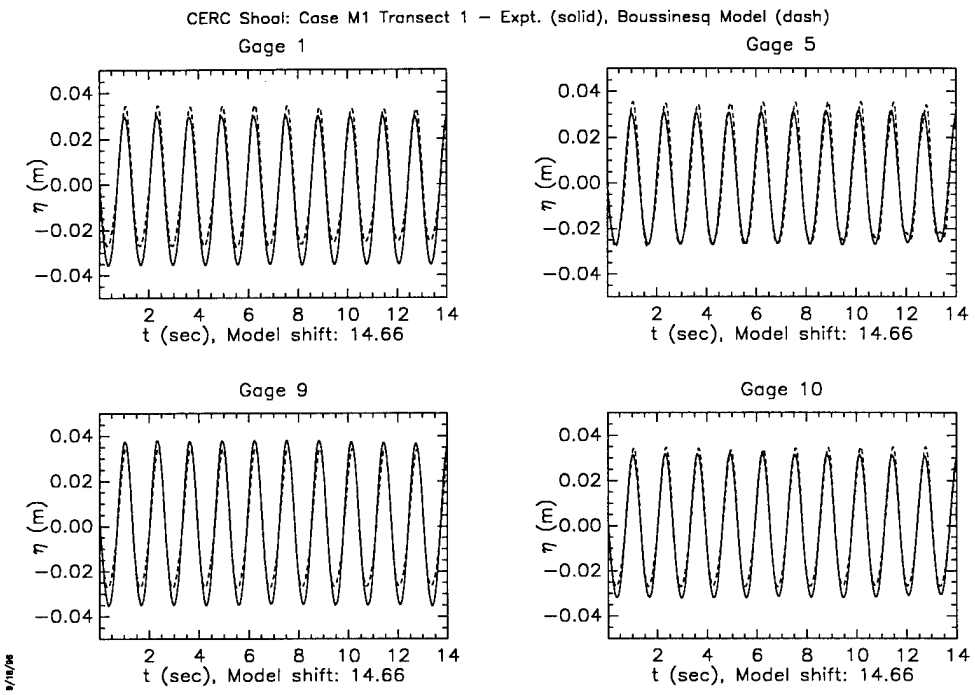

Figure 6. A comparison of the measured time-series and the computed time-series at the gages along transect 1. The measured time-series is plotted as a solid curve. The computed time-series (dash) is shifted by 14.66 sec. The shift in the computed time-series selected by matching the phases of the two curves for Gage #10. This shift can be changed by a period (1.3 sec.) of the waves. The shift is chosen to be large enough so that the computed wave has "achieved full height" or "reached a steady state" for all gages.

series at gage #10, transect 1 with the corresponding computed time-series but starting at different times. Thus we superimpose the measured time-series (solid lines) and the computed time-series (dashed lines) starting at 14.66 sec in figures 6–10. Note that this will be a meaningful comparison if the starting time for the data collection were same for all the gages. In which case this comparison matches not only the individual time-series, but the correlation as well i.e. it provides a comparison of the entire time-dependent wave

surface at these locations. Observe that the agreement in height, shape and phase is very good for gages #1, #5 and #9 for all transects. However, there is significant disagreement in the phases of gage #10 in transects 2, 3 and 5. This is due to the lack of synchronization in the starting time for the data collection in these gages. This conclusion follows from a comparison of the measured gage #10 time-series data at all the transects. The matching of the phases of gage #10 in transect 4 is accidental, the time difference is approximately an integral multiple of the period.

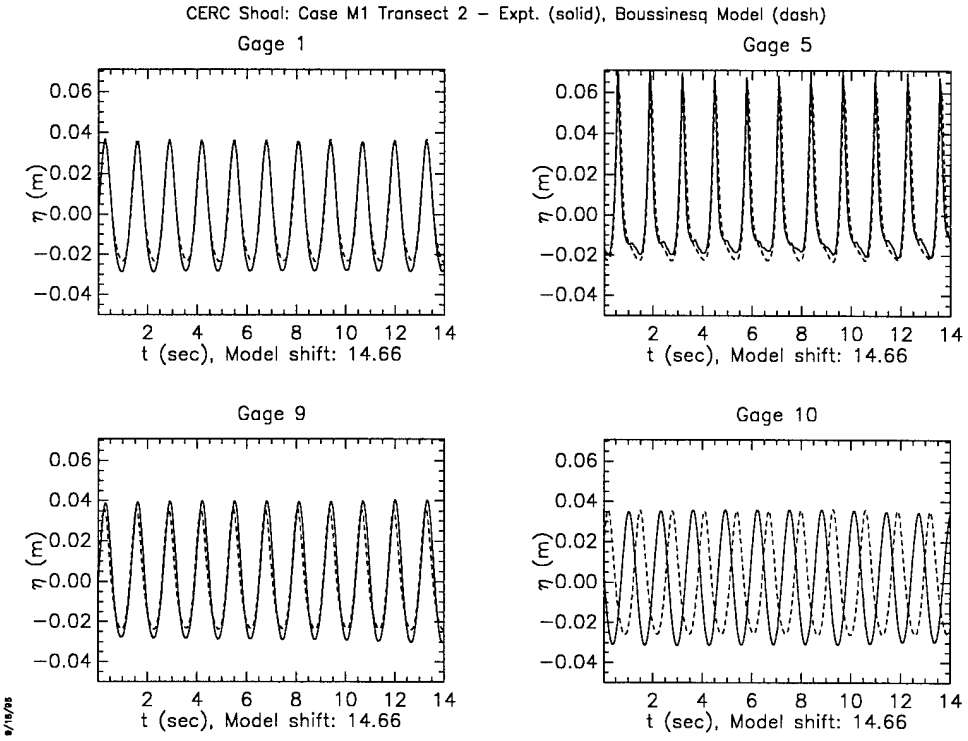

Figure 7. Comparison of measured and computed time-series. The offset used for the computed series is same for all transects and gages. The phase mismatch for gage #10 is due to different starting times for the measured series.

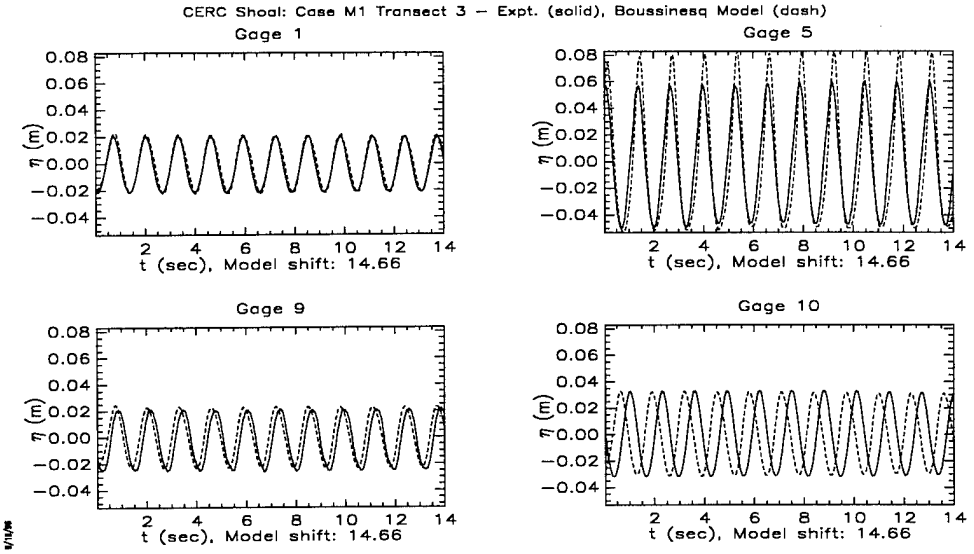

Figure 8. Comparison of measured and computed time-series. The offset used for the computed series is same for all transects and gages. The phase mismatch for gage #10 is due to different starting times for the measured series.

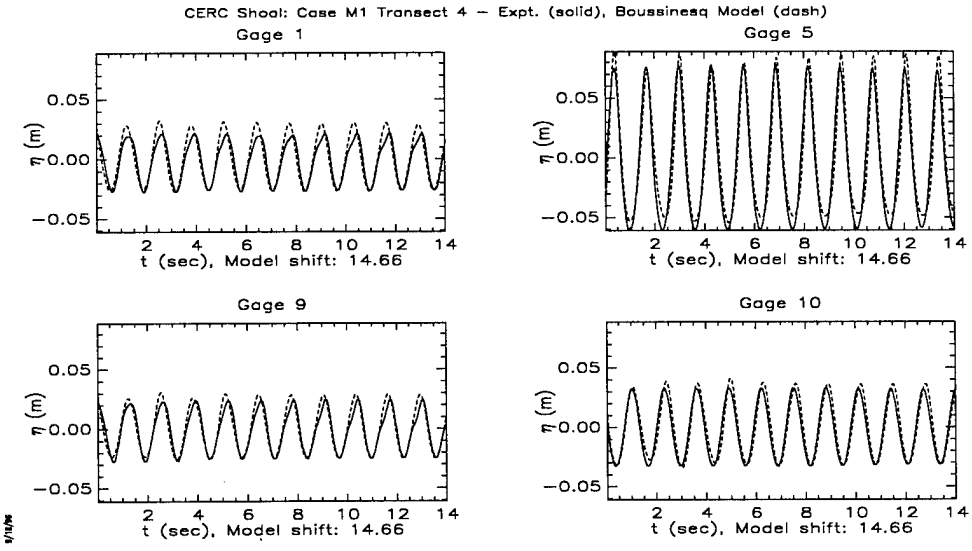

Figure 9. Comparison of measured and computed time-series. The offset used for the computed series is same for all transects and gages. The phases for gage #10 for this transect matches since the difference of starting time between transect 1 and 4 is nearly an integral multiple of the period.

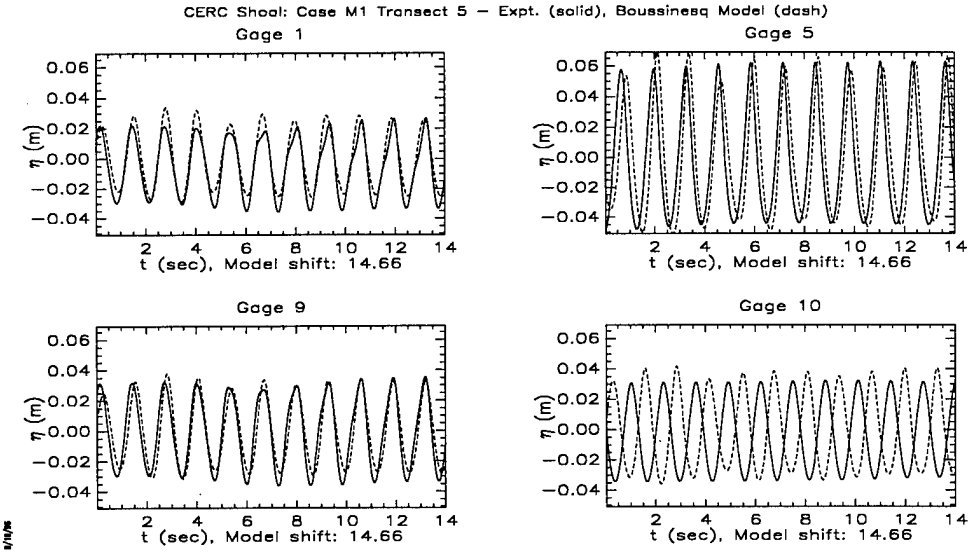

Figure 10. Comparison of measured and computed time-series. The offset used for the computed series is same for all transects and gages. The phase mismatch for gage #10 is due to different starting times for the measured series.

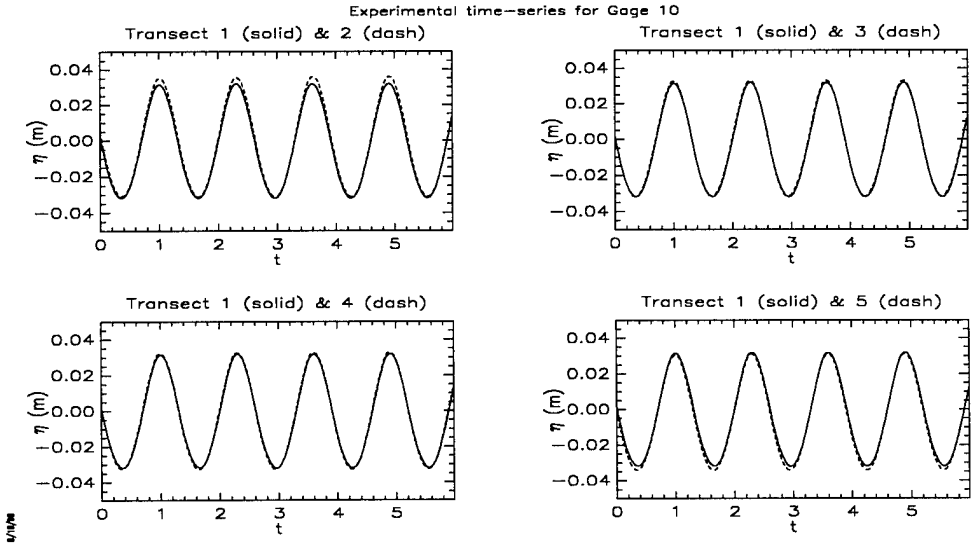

Figure 11 In this figure, we compare the gage #10 measures time-series for transect 1 with that of transects 2–5. None of the time series were shifted.

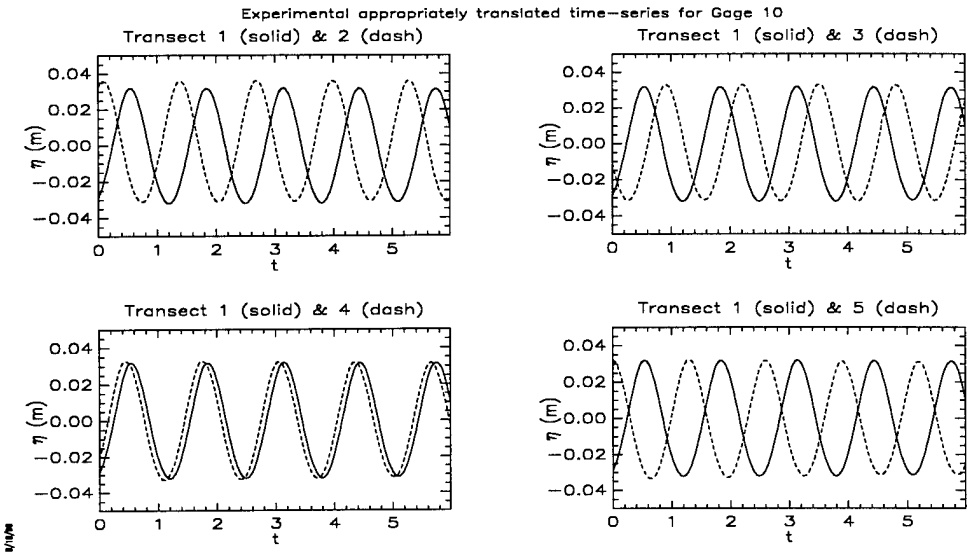

Figure 12 In this figure, we compare the gage #10 measured time-series for transect 1 with that of transects 2–5. The time-series in transects 2–4 were shifted by an amount of that required by the wave to travel from transect 1 to the transect being compared. We use the computed value of the celerity for this purpose.

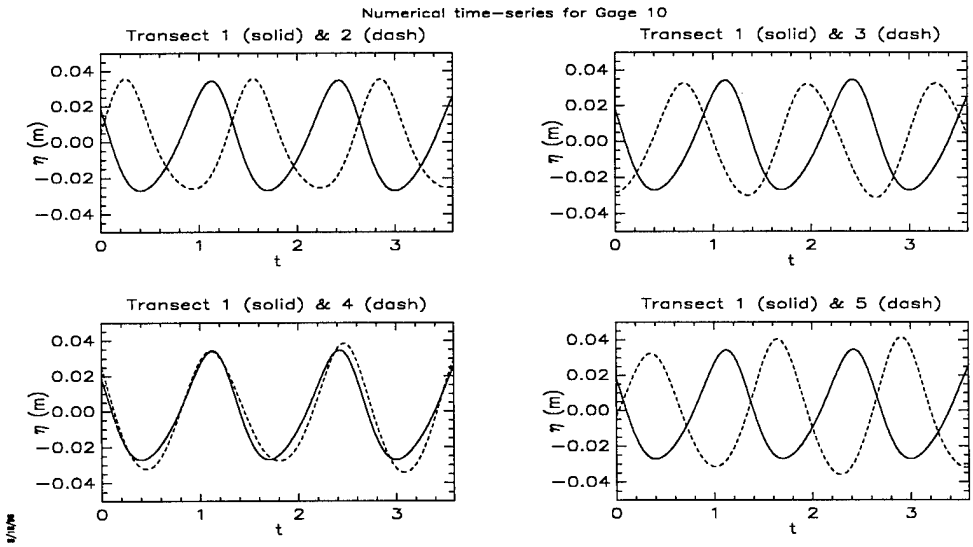

Figure 13 In this figure, we compare the gage #10 numerical time-series for transect 1 with that of transects 2–5. None of the time series were shifted.

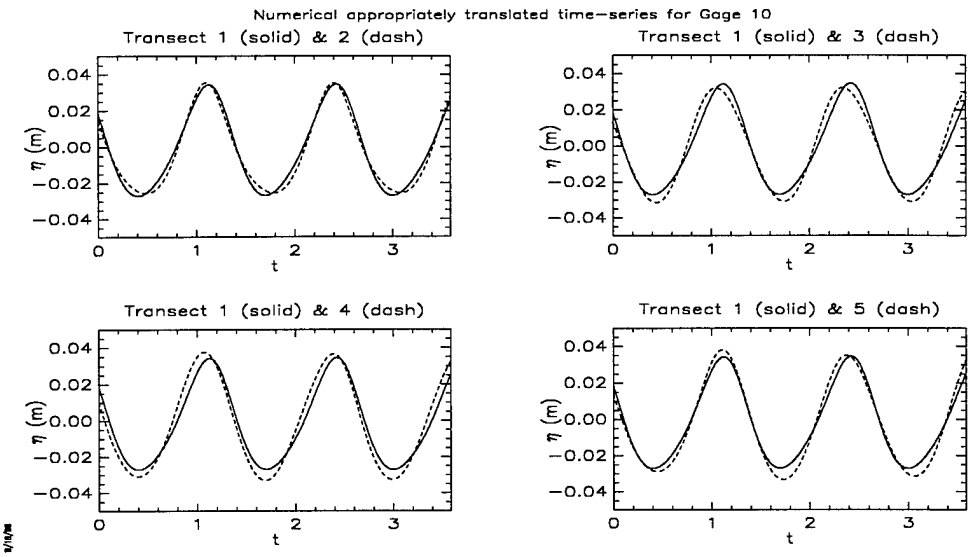

Figure 14 In this figure, we compare the gage #10 computed time-series for transect 1 with that of transects 2–5. The time-series in transects 2–4 were shifted by an amount of that required by the wave to travel from transect 1 to the transect being compared. We use the computed value of the celerity for this purpose.

We compare in figures 11 and 12 the measured time series of gage #10 for different transects in order to determine it the data collection for the gage #10 for different transects started at different instants of time. In figure 11, we compare the time series without any shift, leading to a near perfect match. This match in figure 11 is consistent with a simultaneous start of data collection in gage #10 for different transects only if the separation between the transects is an integral multiple of the wavelength. The experimental wavelength is not known, but the computed wavelength does not satisfy this condition. In figure 12, we compare the time series in transect 1 with the time series in transect 2(3, 4, 5) shifted by an amount of time equal to the time required by the wave to travel from transect 1 to transect 2(3, 4, 5). The curves in figure 12 should match if the data collection started simultaneously. The computed celerity was used in figure 12. The computed celerity should be a good approximation to the physical value. We note that a match could not be obtained by using any nearby value of the celerity. The match between transect 1 and 4 reflects the fact that time required by the wave to travel from transect 1 to transect 4 is approximately four times the period. A similar test was performed on the computed time series with the results shown in figures 13 and 14. As expected, the agreement of the numerical curves are consistent with the simultaneous beginning of the time series. We conclude that data collection in gage #10 at different transects started at the respective downward zero-crossing, rather that at the same instant of time.

	Gage #9	Gage #5	Gage #1	Gage #10
Expt T1	0.0727	0.0579	0.0651	0.0634
Code T1	0.0625	0.0630	0.0623	0.0621
Expt T2	0.0718	0.0895	0.0671	0.0666
Code T2	0.0595	0.0838	0.0600	0.0621
Expt T3	0.0475	0.1081	0.0419	0.0644
Code T3	0.0476	0.1367	0.0449	0.0659
Expt T4	0.0495	0.1364	0.0477	0.0651
Code T4	0.0547	0.1393	0.0583	0.0718
Expt T5	0.0689	0.1087	0.0617	0.0653
Code T5	0.0640	0.1130	0.0567	0.0728

Table 1. Comparison of the measured and computes significant wave-heights.

In table 1, we present a comparison of the significant wave-heights at the gages. Note that this calculation pertains to a single gage location and is unaffected by the lack of synchronization of the starting times. In this comparison, one must consider the accuracy of the measured values. We do not have any information for the experimental accuracy, but can hazard a guess that it is at least a few percent. This error estimate is based partly on the comparison between the gage #1 and gage #9 wave-heights which should be same due to the symmetry. The accuracy of the numerics is estimated to be in the range 1–5% for transects 1–4, and slightly higher for transect 5. We therefore conclude that the agreement between the measurement and the simulation is very good. The largest difference (about 26%) occurs for gage #5 and transect 3, for a large wave-height. Thus we anticipate that the agreement will become even better when a fully non-linear version of Boussinesq model is used.

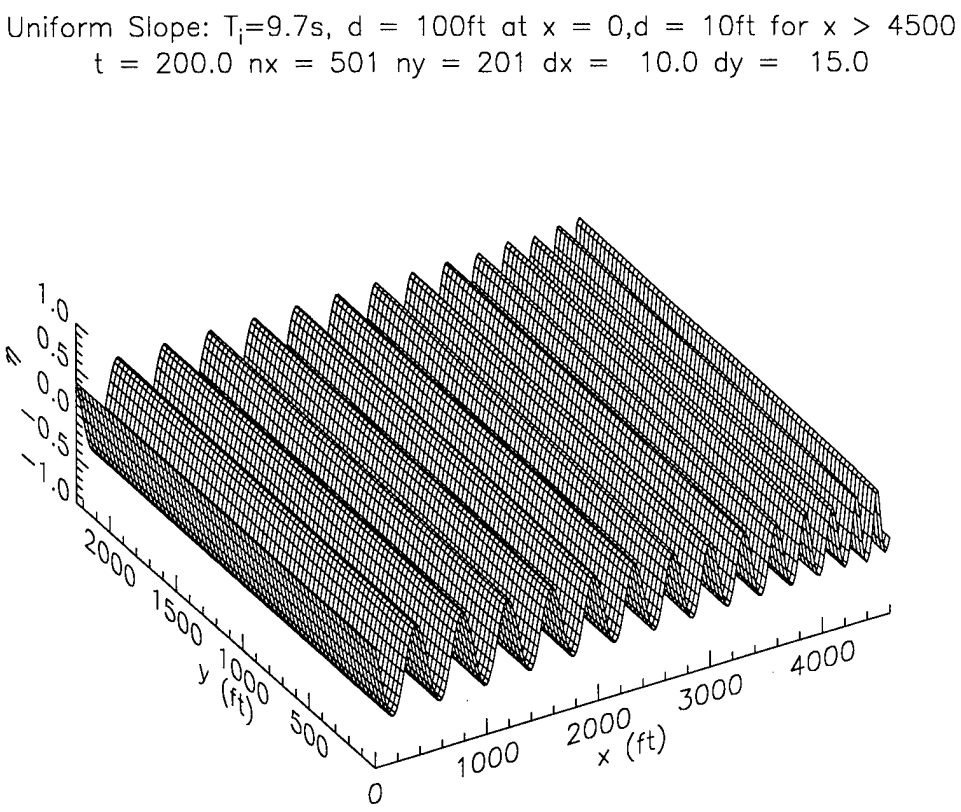

Figure 11. An example near field-scale simulation on a PC. For this example, we consider a beach with a uniform slope of 1/50 along the x-direction. The water depth is taken to be 100 ft at the beginning of the domain. The minimum water depth is 10 ft. The incident wave has period of 9.7 sec and the wave-height is 1 ft.

This last case study is intended to illustrate the feasibility of simulating a field-scale problem. For the purposes of this illustration we consider a wind generated wave with period 9.7 sec. and a beach with a uniform slope of 1/50. Figure 11 shows this computed wave surface at 200 sec. The simulation was performed on a PC and covered an area approximately one mile by one-half mile. The resolution used can be considered to be moderate, this simulation could have been done with a lower resolution, but a more complex incident wave will require higher resolution. As we argue in the next sub-section, much larger scale simulations can be run on more powerful computers currently in the market, and significant improvements are anticipated in Phase II.

The resource requirements:

A crucial aspect of the feasibility is a match between the resource requirement and the availability. By resources we mean computer memory commonly referred to as the RAM and execution or CPU time. The memory requirement depends on the number of grid points NX·NY used in the computation. The current version of the codes are written in FORTRAN and does not use any compiler extension for dynamic memory allocation. Thus the "code size", the memory required by the code, depends on the declared maximum values for these grid points. If the execution is performed with less grid points than the declared maximum, a portion of the code space is not accessed, only the frequently accessed portion of the memory is important to us. Most modern operating systems use virtual memory, typically implemented by using some slower permanent memory, the hard-disk or a device like SSD. In order for the code to execute, the code size must be less than the virtual memory limit. However, if the portion of the memory that needs to be accessed frequently (e.g. it refers to an array-element that is being used) does not fit in the available RAM, a phenomena known as "thrashing" occurs due to the need to constantly access the slower hard-disk. For all practical purposes the code is inoperable if thrashing occurs. Thus the maximum number of grid points that can be used depends on the available core memory (RAM). The time span of the simulation is limited by the available CPU time. The time limitation is frequently unrelated to computer issues, but effectively must be smaller than the mean time between failures from all causes.

Therefore, we conducted some performance analysis of the codes. The memory requirement is linear in the number of grid points NX·NY used in the computation. The time required is approximately linear in NX·NY·NT with NT being the number of time steps. We verified this linear behavior in some examples. Since the time stepping scheme is iterative in nature, the CPU time may be substantially longer in some cases. The tests were performed on several computers, some of the results are shown in Table 2. The two PCs Abbot and Abbot2 are at or slightly below the bottom of the line PCs currently on the market. All the runs described in the earlier sections were performed on Abbot2. The Cray is also outdated. The Sun workstation is a high-end workstation currently on the market. For each computer/compiler combination we used the default compiler options in one set of runs (columns without any compiler flags), and one or more runs with various optimization options for the compiler. The conventional wisdom in the scientific computing community on compiler optimization is that it can change the results. While we are aware that this does happen and one must be careful, it is our opinion that any such changes beyond the cumulative truncation errors is due to some bug in the compiler, bug in the code, poor code design, or the presence of some sensitive dependence. If it is a case of sensitive dependence then it presents serious problem to the numerical methods. In any case, it is prudent to be cautious in this area. For the test cases, the results are independent of the optimization. All calculations were performed

with 8-byte real numbers, single precision on the Cray and double precision on the other machines. All the optimization levels attempt to maximize the execution speed, the flag -Zp on the Cray YMP does not attempt scalar optimization.

	Abbot2		**Abbot**		**Math**		**OSCA**		
	Gateway 2000		Gateway 2000		Sun Microsystems		Cray Research		
	Pentium 120 MHz		Pentium 120 MHz		Ultra 2 200 MHz		Cray YMP 8/128E		
	Processors: 1		Processors: 1		Processors: 2		Processors: 8		
	Memory: 16MB		Memory: 64MB		Memory: 764MB		Memory: 1024MB		
	Windows 95		RedHat Linux 3.0.3		SunOS 5.5.1		UNICOS 9.0.2		
	MS Powerstation 4.0		F2c with gcc 2.7.2		Sun f77 4.0		Cray cf77 6.2		
	Est. Price: $1,800		Est. Price: $2,500		Est. Price: $80,000				
Compiler Flags		/G5 /Ox /Ob2		-m486 -O3 -ffast-math		-fast -xO4 -xparallel -xarch = v8plus		-Zp	-O inline3 -O scalar3 -O task3 -O vector3
Ngwou 2d 100x100x200	196.09	79.10	298.68	137.52	239.53	22.14	35.81	14.31	14.76
Megaflops	5.48	13.58	3.60	7.81	4.49	48.53	30.01	75.09	72.80
Relative Speed	**1.000**	2.479	0.657	1.426	0.819	8.857	5.476	13.703	13.285
Ngwou 1d 4500x800 pre-factored	62.89	45.91	105.74	54.55	68.09	6.15	7.39	7.57	7.41
Megaflops	11.95	16.36	7.10	13.77	11.03	122.16	101.66	99.24	101.39
Relative Speed	**1.000**	1.370	0.595	1.153	0.924	10.226	8.510	8.308	8.487
Ngwou 1d 4500x800 not pre-factored	83.54	66.30	134.13	73.69	82.12	11.18	15.58	15.65	15.71
Megaflops	10.29	12.96	6.41	11.66	10.47	76.87	55.16	54.91	54.70
Relative Speed	**1.000**	1.260	0.623	1.134	1.017	7.472	5.362	5.338	5.318
Whetstone (scalar)	16.69	9.01	13.78	10.99	9.77	2.65	11.19	11.21	4.82
Relative Speed	**1.000**	1.852	1.211	1.519	1.708	6.298	1.492	1.489	3.463
Mflops	5.84	11.34	5.70	7.04	6.99	39.85	2.30	19.10	270.16
Relative Speed	**1.000**	1.942	0.977	1.205	1.196	6.823	0.393	3.271	46.261

Table 2. Computer performance

In Table 2, the row labeled Nwogu 2d gives the CPU time in seconds taken by our two-dimensional code with 100 x 100 grid points for 200 time steps. Next row labeled Megaflops (= Number (in Millions) of floating point operations per second) is obtained by using the number of floating point addition, multiplication and reciprocals given in the output of the *hpm* command on the Cray YMP. Next row gives the relative speeds. The Boussinesq code requires the solutions of tridiagonal systems of linear equations during each computational step. The matrix is fixed and, therefore, can be pre-factored. This pre-factoring has not yet been implemented in our two-dimensional code. The next six rows in the table give the CPU times for the one-dimensional code (4500 grid points and 800 time steps) with and without this pre-factorization of the tridiagonal matrix. The next two rows give results of the standard non-vectorizable Whetstone benchmark. The

Mflops for the scalar machines Abbot, Abbot2 and Math gives the "megaflops" for a subroutine call and the average of one addition and a multiplication. For the Cray YMP, the first Mflops number is this scalar number, the second Mflops number is the result of a partially vectorizable loop: a(i) = a(i - 1) + b*c(i), and the third Mflops number is due to a call to the fully vectorizable machine optimized subroutine SAXPY. This third Mflops number should be very close to the peak floating point speed (per CPU) on the Cray. However, due to necessity to solve the tridiagonal system, the second number provides a more realistic assessment of the performance gain expected for the Boussinesq code. Note that pre-factorization improves the execution speed by nearly a factor of two. Note that the degree of improvement due to compiler optimization varies widely with the compiler, in fact there is no improvement on the Cray for the one-dimensional code. There is generally more improvement in the two-dimensional code that the one-dimensional code. The compiler optimization converts the doubly subscripted arrays to singly-subscripted arrays for part of the improvement. The vector architecture of the Cray supercomputers does not provide large gain for this class of problems. With the help of Dr. Zeki Demirbilek we made some test runs on the Cray C90. Finally, this table (possibly in conjunction with other standard benchmarks such as SPEC_fp) can be used to estimate execution times of the Boussinesq codes in most machines.

Resource requirement of a sample field-scale simulation:

We conclude this section with the estimate of the resource requirements of a field-scale simulation. Note that the size of the domain that can be simulated depends on the wavelengths, the domain size increases as the wavelength increases. Thus a better measure of the size of the domain is the "number of waves" contained in the domain. Here we consider the most likely wind wave and a typical beach. Let us consider an idealized beach with uniform slope of 1/50 and a monochromatic wave of period 10 sec. The wavelength of a wind generated wave of period 10 sec is 156 m in deep water. For this wave, water depth larger than 78 m is considered deep. So we take the simulation domain to begin 3750 m off-shore at a water depth of 75 m and end at 150 m off-shore at a water depth of 3 m. The wavelength varies from 156 m at the deep end to 53.1 m at the shallow end. The resolutions $dx = 2$ m, $dy = 3$ m, $dt = 0.05$ s are reasonable, though one can use coarser grid at the deep end, a finer grid may be necessary at the shallow end. With this set of parameters, we have NX = 1804, NY = 1004 and NT = 12,000 for a 3.6 km by 3 km section of the ocean and 10 min simulation. The memory required for this simulation is approximately 675 MB, larger than the maximum in a typical PC, but well within the range for a typical high-end workstation. Using table 2, we estimate the run times to be 72 hours in a Sun Ultra II, 44 hours in a Cray YMP and 24.5 hours in a Cray C90. Note that approximately 10–12 such simulations can run simultaneously in a Cray C90. Based on other benchmarks available on the Internet, we estimate the run times to be 25 hours on a top of the line Digital AlphaServer 8400 5/440 or approximately 12–15 hours on a Cray T90. We conclude that it is feasible to use the Boussinesq type model to

simulate the wave-climateology of sufficiently large (to be of practical military interest) sections of ocean using currently available computing resources.

The resource requirements of the previous paragraph pertain to the Phase I prototype code. We anticipate that the improved numerics in Phase II will significantly reduce the run times. In fact a doubling of the speed is expected from pre-factorization alone. Combined with the expected advances in computers, a speed gain by at least a factor of 5 by the end of Phase II seem to be reasonable. We also expect that the optimization efforts will reduce the memory requirements and increase the speed. Lastly, any success in parallelizing the code can lead to significant speed improvements in the massively parallel computer.

We believe that the Phase I work and the extensive research in this area clearly demonstrate the feasibility of using Boussinesq type models to simulate nonlinear three-dimensional near shore water waves in field scale size domains using current computers.

Acknowledgments:

Amar Sinha would like to acknowledge several helpful discussions with Dr. Zeki Demirbilek. He would also like to thank Ms. Mary Holman, Ms. Sandra Staggs and Mr. Philip Stewart for their help during the project.

References:

1. Abbott, M. B., McCowan, A. D., and Warren, I. R. (1984), *Accuracy of Short Wave Numerical Model*, J. Hydraul. Engng., **110**, 1287–1301.
2. Anderson, C. M., (1995), *USAF Phillips Laboratory SBIR Software Engineering Guide,* USAF Phillips Laboratory, Kirtland AFB, NM.
3. Chen, Y., and Liu, P. L.-F.(1995), *Modified Boussinesq Equations and Associated Parabolic Models for Water Wave Propagation*, J. Fluid Mech., **288**, 351–381.
4. Demirbilek, Z., and Webster, W. C. (1992), *Application of the Green-Naghdi Theory of Fuild Sheets to Shallow-Water Wave Problems*, Report I . Model development, Tech. Rep. CERC-92-11, US Army Waterways Experiment Station, Vicksburg, MS.
5. Demirbilek, Z., and Webster, W. C. (1992), *User's Manual and Examples for GNWAVE* , Final Report, Tech. Rep. CERC-92-13, US Army Waterways Experiment Station, Vicksburg, MS.
6. Engquist, B., and Majda, A. (1977), *Absorbing Boundary Conditions for the Numerical Simulation of Waves*, Math. Comp., **31**, 629–651.
7. Green, A. E., and Naghdi, P. M. (1976), *A Derivation of Equations for Wave Propagation in Water of Variable Depth*, J. Fluid Mech., **78**, 237–246.

8. Grilli, S. T., Skourup, J., and Svendsen, I. A. (1989), *An Efficient Boundary Element Method for Nonlinear Water Waves*, Engineering Analysis with Boundary Elements, **6**, 97–107.
9. Hammack, J., McCallister, D., Scheffner, N., and Segur, H. (1995), *Two-dimensional Periodic Waves in Shallow Water. Part 1. Asymmetric Waves*, J. Fluid Mech., **285**, 95–122.
10. Hammack, J., Scheffner, N., and Segur, H. (1989), *Two-dimensional Periodic Waves in Shallow Water. Part 1. Symmetric Waves*, J. Fluid Mech., **209**, 567–589.
11. Israeli, M. and Orszag, S. A. (1981), *Approximation of Radiation Boundary Conditions*, J. Comp. Phys. **41**, 115–135.
12. Jiang., L, Ren, X., Wang, K.-H. and Jin, K.-R. (1995), *Generalized Boussinesq Model for Periodic Non-linear Shallow-Water Waves*, **23**, 309–323.
13. Larsen, J., and Dancy, H. (1983), *Open Boundaries in Short Wave Simulations - A New Approach*, Coastal Engng. 7, 285–297.
14. Longuet-Higgins, M. S., and Cokelet, E. D. (1976), *The Deformation of Steep Surface Waves on Water I. A Numerical Method of Computation*, Proc. R. Soc. Lond. A., **350**, 1–26.
15. Nwogu, O. (1993), *Alternative Form of Boussinesq Equations for Nearshore Wave Propagation*, J. Waterway, Port, Coast. Ocean Engng., **119**, 618–638.
16. Nwogu, O. (1994) *Nonlinear Evolution of Directional Wave Spectra in Shallow Water*, Coastal Engineering. 1994, 467–481.
17. Ohyama, T., Kioka, W., and Tada, A. (1995), *Applicability of Numerical Models to Nonlinear Dispersive Waves*, Coastal Engng., **24**, 297–313.
18. Peregrine, D. H. (1967), *Long waves on a beach*, J. Fluid Mech., **27**, 815–827.
19. Schaffer, H. A., and Madsen P. A. (1995), *Further Enhancements of Boussinesq-type Equations*, Coastal Engng., **26**, 1–14.
20. Svendsen, I. A. (1974), *Cnoidal Waves Over a Gently Sloping Bottom*, Tech. Univ. Denmark, Lyngby, ISVA, Ser. Pap. 6.
21. Vincent, C. L. and Briggs, M. J. (1988) *Refraction-Diffraction of Irregular Waves over a Mound* **115**, 269–284.
22. Wei, G., and Kirby, J. T. (1995), *Time-Dependent Numerical Code for Extended Boussinesq Equations*, J. Waterway, Port, Coast. Ocean Engng., **121**, 251–261.
23. Wei, G., Kirby, J. T., Grilli, T. S., and Subramanya, R. (1995), *A Fully Nonlinear Boussinesq Model for Surface Waves. Part 1. Highly Nonlinear Unsteady Waves*, J. Fluid Mech., **294**, 71–92.

Appendix A

In this Appendix, we provide a brief review of the numerical scheme proposed by Wei and Kirby[22] to solve the equations of the Boussinesq type model of Nwogu[15]. Let η be the surface elevation, h be the local water depth, u, v are the x-, y-components of the velocity at the water depth z_α. Wei and Kirby[22] rewrite the equations as:

$$\eta_t = E(\eta, u, v) \tag{1}$$
$$U_t = F(\eta, u, v) + [F_1(v)]_t \tag{2}$$
$$V_t = G(\eta, u, v) + [G_1(u)]_t \tag{3}$$

where

$$U(u) = u + h\left[b_1 h u_{xx} + b_2 (hu)_{xx}\right] \tag{4}$$
$$V(v) = v + h\left[b_1 h v_{xx} + b_2 (hv)_{xx}\right] \tag{5}$$

are treated as simple variables in the time stepping scheme, and,

$$\begin{aligned} E(\eta, u, v) = & -\left[(h+\eta)u\right]_x - \left[(h+\eta)v\right]_y \\ & - \left\{a_1 h^3 (u_{xx} + v_{xy}) + a_2 h^2 \left[(hu)_{xx} + (hv)_{xy}\right]\right\}_x \\ & - \left\{a_1 h^3 (v_{yy} + u_{xy}) + a_2 h^2 \left[(hv)_{yy} + (hu)_{xy}\right]\right\}_y \\ = & -\left[(h+\eta)u\right]_x - \left[(h+\eta)v\right]_y \\ & - \Big\{3a_1 h^2 h_x (u_{xx} + v_{xy}) + a_1 h^3 (u_{xxx} + v_{xxy}) \\ & \quad + 2a_2 h h_x \left[(hu)_{xx} + (hv)_{xy}\right] + a_2 h^2 \left[(hu)_{xxx} + (hv)_{xxy}\right]\Big\} \\ & - \Big\{3a_1 h^2 h_y (v_{yy} + u_{xy}) + a_1 h^3 (v_{yyy} + u_{xyy}) \\ & \quad + 2a_2 h h_y \left[(hv)_{yy} + (hu)_{xy}\right] + a_2 h^2 \left[(hv)_{yyy} + (hu)_{xyy}\right]\Big\} \end{aligned} \tag{6}$$

$$F(\eta, u, v) = -g\eta_x - (uu_x + vu_y) \tag{7}$$
$$G(\eta, u, v) = -g\eta_y - (vv_y + uv_x) \tag{8}$$
$$F_1(v) = -h\left[b_1 h v_{xy} + b_2 (hv)_{xy}\right] \tag{9}$$
$$G_1(u) = -h\left[b_1 h u_{xy} + b_2 (hu)_{xy}\right]. \tag{10}$$

The constants a_1, a_2, b_1, b_2 are given by

$$a_1 = \beta^2/2 - 1/6, \quad a_2 = \beta + 1/2, \quad b_1 = \beta^2, \quad b_2 = \beta \tag{11}$$

where $\beta = z_\alpha/h$, and g is the gravitational acceleration.

These equations are solved numerically by a finite difference method. They are discretized on a uniform grid as: $x = i\Delta x, y = j\Delta y, t = n\Delta t$ with $i = 0, \cdots, N_x, j = 0, \cdots, N_y, n = 0, \cdots N_t$. Suppressing the time dependence, i.e. the index n, we have:

$$
\begin{aligned}
&E_{i,j}(\eta, u, v) = \\
&- \left\{ \frac{(h+\eta)_{i+1,j} u_{i+1,j} - (h+\eta)_{i-1,j} u_{i-1,j}}{2dx} \right\} - \left\{ \frac{(h+\eta)_{i,j+1} u_{i,j+1} - (h+\eta)_{i,j-1} u_{i,j-1}}{2dy} \right\} \\
&- \left\{ 3a_1 h_{i,j}^2 (h_x)_{i,j} \left(\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{dx^2} + \frac{v_{i+1,j+1} - v_{i+1,j-1} - v_{i-1,j+1} + v_{i-1,j-1}}{4dxdy} \right) \right\} \\
&- a_1 h_{i,j}^3 \left\{ \frac{u_{i+2,j} - 2u_{i+1,j} + 2u_{i-1,j} - u_{i-2,j}}{2dx^3} \right. \\
&\qquad \left. + \frac{v_{i+1,j+1} - v_{i+1,j-1} - 2(v_{i,j+1} - v_{i,j-1}) + v_{i-1,j+1} - v_{i-1,j-1}}{2dx^2 dy} \right\} \\
&- 2a_2 h_{i,j} (h_x)_{i,j} \left\{ \frac{(hu)_{i+1,j} - 2(hu)_{i,j} + (hu)_{i-1,j}}{dx^2} \right. \\
&\qquad \left. + \frac{(hv)_{i+1,j+1} - (hv)_{i+1,j-1} - (hv)_{i-1,j+1} + (hv)_{i-1,j-1}}{4dxdy} \right\} \\
&- a_2 h_{i,j}^2 \left\{ \frac{(hu)_{i+2,j} - 2(hu)_{i+1,j} + 2(hu)_{i-1,j} - (hu)_{i-2,j}}{2dx^3} \right. \\
&\qquad \left. + \frac{(hv)_{i+1,j+1} - (hv)_{i+1,j-1} - 2(hv)_{i,j+1} - 2(hv)_{i,j-1} + (hv)_{i-1,j+1} - (hv)_{i-1,j-1}}{2dx^2 dy} \right\} \\
&- 3a_1 h_{i,j}^2 (h_y)_{i,j} \left\{ \frac{v_{i,j+1} - 2v_{i,j} + v_{i,j-1}}{dy^2} + \frac{u_{i+1,j+1} - u_{i+1,j-1} - u_{i-1,j+1} + u_{i-1,j-1}}{4dxdy} \right\} \\
&- a_1 h_{i,j}^3 \left\{ \frac{v_{i,j+2} - 2v_{i,j+1} + 2v_{i,j-1} - v_{i,j-2}}{2dy^3} \right. \\
&\qquad \left. + \frac{u_{i+1,j+1} - u_{i-1,j+1} - 2(u_{i+1,j} - u_{i-1,j}) + u_{i+1,j-1} - u_{i-1,j-1}}{2dy^2 dx} \right\} \\
&- 2a_2 h_{i,j} (h_y)_{i,j} \left\{ \frac{(hv)_{i,j+1} - 2(hv)_{i,j} + (hv)_{i,j-1}}{dy^2} \right. \\
&\qquad \left. + \frac{(hu)_{i+1,j+1} - (hu)_{i+1,j-1} - (hu)_{i-1,j+1} + (hu)_{i-1,j-1}}{4dxdy} \right\} \\
&- a_2 h_{i,j}^2 \left\{ \frac{(hv)_{i,j+2} - 2(hv)_{i,j+1} + 2(hv)_{i,j-1} - (hv)_{i,j-2}}{2dy^3} \right. \\
&\qquad \left. + \frac{(hu)_{i+1,j+1} - (hu)_{i-1,j+1} - 2(hu)_{i+1,j} - 2(hu)_{i-1,j} + (hu)_{i+1,j-1} - (hu)_{i-1,j-1}}{2dy^2 dx} \right\}
\end{aligned}
\tag{12}
$$

where

$$(h_x)_{i,j} = \frac{h_{i+1,j} - h_{i-1,j}}{2dx} \tag{13}$$

$$(h_y)_{i,j} = \frac{h_{i,j+1} - h_{i,j-1}}{2dy} \tag{14}$$

The spatial discretization of U, V are given by:

$$
\begin{aligned}
U_{i,j}(u) &= u_{i,j} \\
&\quad + h_{i,j}\left[b_1 h_{i,j}\left(\frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{dx^2}\right)+b_2\left(\frac{(hu)_{i+1,j}-2(hu)_{i,j}+(hu)_{i-1,j}}{dx^2}\right)\right] \\
&= \frac{h_{i,j}}{dx^2}\left[b_1 h_{i,j}+b_2 h_{i-1,j}\right]u_{i-1,j}+\left[1-\frac{2h_{i,j}^2}{dx^2}(b_1+b_2)\right]u_{i,j} \\
&\quad + \frac{h_{i,j}}{dx^2}\left[b_1 h_{i,j}+b_2 h_{i+1,j}\right]u_{i+1,j} \\
&\equiv \sum_k (M^{[u]}_{[j]})_{i,k}u_{k,j} \qquad (15)
\end{aligned}
$$

$$
\begin{aligned}
V_{i,j}(u) &= v_{i,j} \\
&\quad + h_{i,j}\left[b_1 h_{i,j}\left(\frac{v_{i,j+1}-2v_{i,j}+v_{i,j-1}}{dy^2}\right)+b_2\left(\frac{(hv)_{i,j+1}-2(hv)_{i,j}+(hv)_{i,j-1}}{dy^2}\right)\right] \\
&= \frac{h_{i,j}}{dy^2}\left[b_1 h_{i,j}+b_2 h_{i,j-1}\right]v_{i,j-1}+\left[1-\frac{2h_{i,j}^2}{dy^2}(b_1+b_2)\right]v_{i,j} \\
&\quad + \frac{h_{i,j}}{dy^2}\left[b_1 h_{i,j}+b_2 h_{i,j+1}\right]v_{i,j+1} \\
&\equiv \sum_k (M^{[v]}_{[i]})_{j,k}v_{i,k}\,. \qquad (16)
\end{aligned}
$$

All terms in equation (15) have the same index j, and thus it can be treated as a linear system connecting u and U via the tridiagonal matrix $M^{[u]}_{[j]}$. The quantity U is treated as a simple variable in the time stepping scheme, i.e. its values are obtained directly. The values of u are then obtained by solving N_y tridiagonal equations. Note that the matrix is constant and involves only one dimension. The boundary terms have a slightly different form, that can be handled either by redefining the matrix or by the right side of the equation, we redefine the right side. Similar remarks apply for equation (16). The remaining quantities are given by:

$$
\begin{aligned}
F_{i,j}(\eta,u,v) &= -g\left(\frac{\eta_{i+1,j}-\eta_{i-1,j}}{2dx}\right) \\
&\quad - u_{i,j}\left(\frac{u_{i+1,j}-u_{i-1,j}}{2dx}\right)-v_{i,j}\left(\frac{u_{i,j+1}-u_{i,j-1}}{2dy}\right) \\
&= -g\left(\frac{\eta_{i-2,j}-8\eta_{i-1,j}+8\eta_{i+1,j}-\eta_{i+2,j}}{12dx}\right) \\
&\quad - u_{i,j}\left(\frac{u_{i-2,j}-8u_{i-1,j}+8u_{i+1,j}-u_{i+2,j}}{12dx}\right) \\
&\quad - v_{i,j}\left(\frac{u_{i,j-2}-8u_{i,j-1}+8u_{i,j+1}-u_{i,j+2}}{12dy}\right), \qquad (17)
\end{aligned}
$$

$$G_{i,j}(\eta,u,v) = -g\left(\frac{\eta_{i,j-1}-\eta_{i,j-1}}{2dy}\right)$$
$$- v_{i,j}\left(\frac{v_{i,j+1}-v_{i,j-1}}{2dy}\right) - u_{i,j}\left(\frac{v_{i+1,j}-v_{i-1,j}}{2dx}\right)$$
$$= -g\left(\frac{\eta_{i,j-2}-8\eta_{i,j-1}+8\eta_{i,j+1}-\eta_{i,j+2}}{12dy}\right)$$
$$- v_{i,j}\left(\frac{v_{i,j-2}-8v_{i,j-1}+8v_{i,j+1}-v_{i,j+2}}{12dy}\right)$$
$$- u_{i,j}\left(\frac{v_{i-2,j}-8v_{i-1,j}+8v_{i+1,j}-v_{i+2,j}}{12dx}\right), \tag{18}$$

$$(F_1(v))_{i,j} = -h_{i,j}\Big[b_1 h_{i,j}\Big(\frac{v_{i+1,j+1}-v_{i+1,j-1}-v_{i-1,j+1}+v_{i-1,j-1}}{4dxdy}\Big)$$
$$+ b_2\Big(\frac{(hv)_{i+1,j+1}-(hv)_{i+1,j-1}-(hv)_{i-1,j+1}+(hv)_{i-1,j-1}}{4dxdy}\Big)\Big], \tag{19}$$

$$(G_1(u))_{i,j} = -h_{i,j}\Big[b_1 h_{i,j}\Big(\frac{u_{i+1,j+1}-u_{i+1,j-1}-u_{i-1,j+1}+u_{i-1,j-1}}{4dxdy}\Big)$$
$$+ b_2\Big(\frac{(hu)_{i+1,j+1}-(hu)_{i+1,j-1}-(hu)_{i-1,j+1}+(hu)_{i-1,j-1}}{4dxdy}\Big)\Big]. \tag{20}$$

We have used both (selected by a flag) the three-point and five-point formula in equations (17) and (18).

The time stepping scheme proposed by Wei and Kirby[22] is a Predictor-Corrector seheme. The predictor step is the third order explicit Adams-Bashforth scheme, given by:

$$\eta_{i,j}^{n+1} = \eta_{i,j}^{n} + \frac{\Delta t}{12}\left[23E_{i,j}^{n} - 16E_{i,j}^{n-1} + 5E_{i,j}^{n-2}\right] \tag{21}$$

$$U_{i,j}^{n+1} = U_{i,j}^{n} + \frac{\Delta t}{12}\left[23F_{i,j}^{n} - 16F_{i,j}^{n-1} + 5F_{i,j}^{n-2}\right]$$
$$+ 2(F_1)_{i,j}^{n} - 3(F_1)_{i,j}^{n-1} + (F_1)_{i,j}^{n-2} \tag{22}$$

$$V_{i,j}^{n+1} = V_{i,j}^{n} + \frac{\Delta t}{12}\left[23G_{i,j}^{n} - 16G_{i,j}^{n-1} + 5G_{i,j}^{n-2}\right]$$
$$+ 2(G_1)_{i,j}^{n} - 3(G_1)_{i,j}^{n-1} + (G_1)_{i,j}^{n-2}. \tag{23}$$

The corrector step is the fourth order explicit Adams-Moulton scheme, given by:

$$\eta_{i,j}^{n+1} = \eta_{i,j}^{n} + \frac{\Delta t}{24}\left[9E_{i,j}^{n+1} + 19E_{i,j}^{n} - 5E_{i,j}^{n-1} + E_{i,j}^{n-2}\right] \tag{24}$$

$$U_{i,j}^{n+1} = U_{i,j}^{n} + \frac{\Delta t}{24}\left[9F_{i,j}^{n+1} + 19F_{i,j}^{n} - 5F_{i,j}^{n-1} + F_{i,j}^{n-2}\right]$$
$$+ (F_1)_{i,j}^{n+1} - (F_1)^{n})_{i,j} \tag{25}$$

$$V_{i,j}^{n+1} = V_{i,j}^{n} + \frac{\Delta t}{24}\left[9G_{i,j}^{n+1} + 19G_{i,j}^{n} - 5G_{i,j}^{n-1} + G_{i,j}^{n-2}\right]$$
$$+ (G_1)_{i,j}^{n+1} - (G_1)^{n})_{i,j} \tag{26}$$

The scheme is applied as follows. We assume that the dependent variables η, u, v are known at three successive time steps: $n-2, n-1, n$. All terms on the right sides of the equations (21 – 24) can be calculated using these values and equations (12 – 20). This gives the "predicted" values of η, U, V at the next time step $n+1$. The values of u, v are then obtained by solving the tridiagonal systems in equations (15, 16). In the corrector step equations (24 – 26), the next time step $n+1$ occurs on both sides of this equation i.e. this is an implicit equation which is solved by iteration. The "predicted" values obtained from the predictor equations (21 – 23) gives the starting point of this iteration. The values at $k-th$ stage of the iteration is used in the right side of equations (24 – 26) to obtain the values at stage $k+1$. The iteration stops if convergence is achieved or the number of iteration exceeds some preset value. The convergence criteria is that either the relative error in L^1 norm or the L^1 is less than some prescribed tolerance ϵ for each of η, u, v. This procedure can be repeated for the following time step.

This is multi-level scheme. The successive time steps are required to start the scheme. The $0-th$ step is given by the initial condition. As in other multi-steps schemes, a single stem scheme such as the Euler scheme is used to obtain the next two time steps. However, use of Euler scheme is not necessary in this problem. This is due to the fact that we begin with still water i.e. η, u, v are identically zero at $t=0$. It is easily seen that this condition can be assumed for the first three time steps without any loss of generality.

We note that the program must store the values for three successive time steps as well as two sets of values needed in the iteration stage of the corrector scheme. If one wishes to "restart" a simulation, the variables at three successive step must be stored at the end of first run and retrieved at the start of next run.

Finally, we note that the formulae for the one-dimensional case can be obtained from the two-dimensional formula by reduction. To obtain the one dimensional equations we set v, V, G, F_1, G_1 to zero, the y derivatives to zero, and all discretized quantities to be independent of j in equations (1 – 26).