

**Naval Research Laboratory**

Stennis Space Center, MS 39529-5004



NRL/FR/7322--97-9671

# **Insights into Unstructured Mesh Generation for Coastal Ocean Applications**

CHERYL ANN BLAIN  
ASHLEY P. McMANUS

*Ocean Dynamics and Prediction Branch  
Oceanography Division*

March 6, 1998

Approved for public release; distribution unlimited.

# REPORT DOCUMENTATION PAGE

*Form Approved*  
*OBM No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

|   |  |  |
|---|--|--|
| <b>1. AGENCY USE ONLY (Leave blank)</b> | <b>2. REPORT DATE</b><br>March 6, 1998 | <b>3. REPORT TYPE AND DATES COVERED</b><br>Final |
|---|--|--|

|   |  |
|---|--|
| <b>4. TITLE AND SUBTITLE</b><br>Insights into Unstructured Mesh Generation for Coastal Ocean Applications | <b>5. FUNDING NUMBERS</b><br>Job Order No. 573672000<br>Program Element No. 0602435N |
|---|--|

|   |   |
|---|---|
| <b>6. AUTHOR(S)</b><br>Cheryl Ann Blain and Ashley P. McManus | <b>Project No.</b><br><b>Task No.</b> BE-35-2-15<br><b>Accession No.</b> DN163783 |
|---|---|

|  |   |
|--|---|
| <b>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</b><br>Naval Research Laboratory<br>Oceanography Division<br>Stennis Space Center, MS 39529-5004 | <b>8. PERFORMING ORGANIZATION REPORT NUMBER</b><br>NRL/FR/7322--97-9671 |
|--|---|

|   |   |
|---|---|
| <b>9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)</b><br>Office of Naval Research<br>800 N. Quincy St.<br>Arlington, VA 22217-5000 | <b>10. SPONSORING/MONITORING AGENCY REPORT NUMBER</b> |
|---|---|

**11. SUPPLEMENTARY NOTES**

|   |                               |
|---|-------------------------------|
| <b>12a. DISTRIBUTION/AVAILABILITY STATEMENT</b><br>Approved for public release; distribution unlimited. | <b>12b. DISTRIBUTION CODE</b> |
|---|-------------------------------|

**13. ABSTRACT (Maximum 200 words)**

The finite element method for discretizing differential equations is introduced and briefly compared to finite difference approaches. Advantages of finite element approximations include ease of localized mesh refinement and maximization of computational resources through unstructured, graded meshes and element-based computations. The use of triangular elements lead to realistic representations of shoreline and bathymetric complexity; boundary conditions are naturally incorporated. Presentation of a theoretical framework and state-of-the-art research for determining required mesh resolution in relation to the modeled physics is followed by practical concerns of mesh generation itself. Known tools for mesh generation and promising developmental software are discussed. Detailed descriptions of mesh creation, editing, and model diagnostics using the *ACE/gredit* software, the most versatile and advanced mesh creation and modification tool available, comprises the remainder of this report. Step-by-step instructions for the semi-automatic generation of a finite element mesh are included, which has the potential to advance the rapid relocatability of finite element models.

|   |                                  |
|---|----------------------------------|
| <b>14. SUBJECT TERMS</b><br>wave modeling, tide modeling, coupled waves and tides | <b>15. NUMBER OF PAGES</b><br>63 |
|   | <b>16. PRICE CODE</b>            |

|  |   |  |   |
|--|---|--|---|
| <b>17. SECURITY CLASSIFICATION OF REPORT</b><br>Unclassified | <b>18. SECURITY CLASSIFICATION OF THIS PAGE</b><br>Unclassified | <b>19. SECURITY CLASSIFICATION OF ABSTRACT</b><br>Unclassified | <b>20. LIMITATION OF ABSTRACT</b><br>Same as report |
|--|---|--|---|

## CONTENTS

|  |    |
|--|----|
| 1.0 INTRODUCTION .....   | 1  |
| 2.0 AN OVERVIEW OF THE FINITE ELEMENT METHOD .....                             | 1  |
| 2.1 Mathematical Development .....   | 2  |
| 2.2 Comparison to Finite Difference Methods .....                              | 4  |
| 3.0 UNSTRUCTURED MESH RESOLUTION .....   | 7  |
| 3.1 Theoretical Aspects of Mesh Resolution .....                               | 7  |
| 3.2 Empirical Approaches to Mesh Resolution .....                              | 10 |
| 3.3 Recent Research in Mesh Generation .....                                   | 11 |
| 4.0 EXISTING TOOLS FOR MESH GENERATION .....                                   | 12 |
| 4.1 ACE/gredit .....   | 13 |
| 4.2 TriGrid .....  | 13 |
| 4.3 Software Packages Under Development .....                                  | 14 |
| 5.0 APPLICATIONS USING ACE/gredit .....  | 14 |
| 5.1 Definition of Terminology .....  | 15 |
| 5.2 Mesh Creation .....  | 17 |
| 5.3 Mesh Modification and Editing .....  | 23 |
| 5.4 Model Diagnostics .....  | 27 |
| 6.0 SUMMARY .....  | 30 |
| 7.0 ACKNOWLEDGMENTS .....  | 30 |
| 8.0 REFERENCES .....   | 30 |
| APPENDIX A — Examples of Semi-Automatic Grid Generation Using ACE/gredit ..... | 33 |

# INSIGHTS INTO UNSTRUCTURED MESH GENERATION FOR COASTAL OCEAN APPLICATIONS

## 1.0 INTRODUCTION

In recent times, an increasing number of coastal ocean model applications employed finite element modeling techniques. Significant advantages are associated with the finite element discrete form of equations governing the coastal ocean; the most obvious of these advantages being the tremendous grid flexibility afforded by finite elements. Variable mesh resolution allows enhancements in localized regions to represent sharp gradients of flow or bathymetric features or to capture the tortuous detail of a shoreline. Simultaneously, course mesh spacing can be utilized in deeper waters where changes in the ocean dynamics are known to occur more slowly or on larger scales. Such flexibility leads to questions of how best to construct a finite element mesh that will efficiently meet one's modeling aims and, secondly, what tools are available to accomplish this end in the most automated fashion possible.

This report offers a comprehensive presentation of the state-of-the art wisdom regarding the answers to these questions. Before moving to a discussion of mesh resolution, an overview of the basic premise and an abbreviated mathematical development of finite element approximations is presented for the uninitiated reader. Included in this introduction to the finite element approach is a comparison of finite element model formulations to the widely used finite difference modeling approaches. In the application of developed models that utilize finite element techniques, the user is required to understand the relationship between mesh resolution, domain size, boundary forcing, and the model computed fields to apply and assess the performance of finite element coastal ocean models.

As mentioned, the heart of any finite element model application is an unstructured computational mesh. Both theoretical and empirical approaches can be used to identify *a priori* general mesh resolution requirements for a particular application. These perspectives on mesh resolution are presented together with highlights of advances and current areas of research related to the determination of unstructured mesh resolution. The discussion of mesh resolution gives way to practical concerns of mesh generation. Several known tools for mesh generation are described along with mention of some promising software under development. Finally, implementation of one particular mesh generation and editing tool, *ACE/gredit*, is presented in depth. Currently, this package offers the most versatile and advanced features in mesh creation and editing within a user-friendly environment. Detailed descriptions on the usage of *ACE/gredit* are presented in the context of common finite element mesh applications such as mesh creation, modification, and model diagnostics. Note that step-by-step instructions for the automatic generation of a mesh given coastline and bathymetry data is also presented.

## 2.0 AN OVERVIEW OF THE FINITE ELEMENT METHOD

Differential equations are an attempt to describe the actual behavior of a physical system. In the coastal ocean, the governing partial differential equations, at a minimum, consist of the conservation

laws of mass and momentum. These equations, because of their hyperbolic nature and strong nonlinearity, preclude analytic solution. Instead, a discrete representation of the governing equations must be formulated for solution on a computer. Selection of an optimal numerical technique depends on the governing equations themselves, the particular application, and the geometry of the domain involved.

## 2.1 Mathematical Development

The finite element method is simply a general technique for constructing approximate solutions to boundary value problems. The *method of weighted residuals* (MWR) is a family of numerical approximation techniques from which the finite element equations are derived. The heart of the MWR is predicated on an approximation to the actual *solution* to the differential equation while maintaining the original differential operator. A functional form for the solution, or trial function, is selected to satisfy the differential operator only in an approximate way. A convenient form for the trial function is a set of basis functions in  $N$  dimensional space such that their linear combination approximates the problem solution, i.e.:

$$\hat{U}(\underline{x}) = \sum_{i=1}^N \alpha_i \varphi_i(\underline{x}), \quad (1)$$

where  $\hat{U}(\underline{x})$  is the approximate solution to a partial differential equation,  $\varphi_i(\underline{x})$  are the basis functions of the solution,  $\alpha_i(\underline{x})$  are scalar coefficients, and  $\underline{x}$  contains the independent variables. The problem, then (in the MWR) is to determine the coefficients  $\alpha_i$  such that  $\hat{U}(\underline{x})$  is a good approximation to the true solution of the partial differential equations. By substitution of the approximate solution, Eq. (1), into the governing partial differential equations, MWR algorithms achieve the fundamental tenet of numerical methods: replacement of a differential equation by a related system of algebraic equations, finite in number.

In choosing appropriate values for the coefficients  $\alpha_i$ , the MWR seeks to minimize the amount by which the approximate solution fails to satisfy the original governing equations. The measure of the error, termed the residual, is minimized by forcing it to 0 in a weighted, average sense over the entire domain. Mathematically, the MWR assumes the form:

$$\int_{\Omega} [L(\hat{U}(\underline{x})) - f(\underline{x})] w_i(\underline{x}) d\underline{x}, \quad (2)$$

where  $\Omega$  is the problem domain,  $L$  is the differential operator,  $\hat{U}(\underline{x})$  is the approximate solution,  $f(\underline{x})$  is the known forcing function,  $L(\hat{U}(\underline{x})) - f(\underline{x})$  is the residual, and  $w_i(\underline{x})$  are the weighting or test functions. Different choices of the weight functions are the distinguishing feature among various MWR approximations.

In the MWR, functional forms for both the approximate solution and the weights must be determined. Admissibility conditions on the trial functions used to approximate the solution include satisfaction of all of the boundary conditions over the entire domain and functional continuity as required by the differential operators of the governing equations. In practice, the trial space is most always chosen from finite-dimensional polynomial space because of the simple definitions and well-known properties of polynomials. A popular choice are the Lagrange polynomials since the

coefficients  $\alpha_i$  correspond exactly to discrete nodal points in the domain. Furthermore, Lagrange polynomials are non-zero over the entire domain except at the nodal points and are infinitely differentiable. Note, however, that the choice of Lagrange polynomials is one of convenience, not necessity, since basis sets are not unique.

A global set of functions that satisfy all of the prescribed boundary conditions over the domain is extremely difficult to determine, so it becomes advantageous to relax the admissibility requirements through a *fundamental weak form* of the MWR equations. In the fundamental weak form, only essential or Dirichlet boundary conditions must be satisfied with natural (flux) or Neumann boundary conditions met in a weighted residual sense; i.e., the approximate solution error associated with the boundary flux is explicitly included in the fundamental weak form of the MWR formulation:

$$\int_{\Omega} [L(\hat{U}(x)) - f(x)] w_i(x) dx + \int_{\Gamma} [-S_N(\hat{U}(x)) + g_N] w_i(x) d\Gamma, \quad (3)$$

where  $\Gamma$  is the domain boundary,  $S_N(\hat{U}(x))$  is the approximated natural boundary flux, and  $g_N$  is the prescribed value at the natural boundary.

Functional continuity requirements on the trial and test functions are further reduced through an application of Green's theorem (or integration by parts). A *symmetrical, weak weighted residual formulation* is reached when the order of the derivatives on both the trial and test functions match. Weak forms of the equations allow lesser conditions on the problem equations and its derivatives and, thus, lead to the accommodation of irregular solutions having irregular domains and forcing functions. Note it can be shown (Becker et al. 1981) that the solution to the original equations also satisfies the weak form of the equations and is the *only* solution of the weak form.

The most widely implemented MWR approximation, the Galerkin approach, is defined by choosing the trial function for the approximate solution and the weighting functions to be identical. This approach reduces the number of unknowns and significantly simplifies the boundary error term. Application of the Galerkin MWR yields a symmetric stiffness matrix that allows more expedient solution techniques.

While the functions representing the approximate solution and the weights can be defined globally, functional continuity is more readily enforced over local domains or finite elements. Thus, it is advantageous to define the basis functions to be piece-wise continuous over the finite elements. Satisfaction of the essential boundary conditions is significantly simplified for locally defined functions over globally defined ones. In addition, splitting the domain into intervals and using lower order approximations within each element cause the integral error to assume better accuracy on a point-wise basis.

For applications that utilize the finite element method, operation at the elemental level using a local element coordinate system offers several advantages. Basis function interpolation is simplified by the definition of one set of elemental functions for all elements regardless of shape, size, or location. As a consequence, integrals in the weak weighted residual form of Eq. (3) are evaluated element-wise, with functional continuity enforced at the element level, and then summed to form a global system of equations:

$$\sum_{j=1}^{NELE} \left[ \sum_{i=1}^N \alpha_i \int_{elem} L(\varphi_i(x)) w_i(x) dx + \int_{elem} f(x) w_i(x) dx \right] - S_N(\hat{U}(x)) w_N(x) |_N. \quad (4)$$

Essential boundary conditions are imposed on this global set of equations and the system is solved for the unknown nodal values,  $\alpha_i$ . Knowing these nodal coefficients, a solution to the governing equations is obtained from substitution into the approximate solution, Eq. (1).

The finite element method allows for systematic and efficient development of discrete approximation equations. Note in the development of the finite element equations above that no specific differential equation or operator was identified. The finite element method as described is a general technique that is readily implemented for a variety of governing partial differential equations. Basis functions that approximate the solution are selected piece-wise over subregions of the domain (finite elements) and can be chosen as simple functions (polynomials of low degree) over the element. Piece-wise linear basis functions further simplify the problem in that their coefficients are the values of the solution at the nodal points. These unknown functional values at the nodal points are then solved for by minimizing the error over the domain with respect to some weighting functions. Efficiency of the finite element method is enhanced in that the formulation lends itself to solution one element at a time.

For further details, discussion, and examples, the interested reader is referred to several texts on the finite element method including Celia and Gray (1992), Lapidus and Pinder (1982), and Becker et al. (1981).

## 2.2 Comparison to Finite Difference Methods

One of the most common and conceptually straightforward numerical approaches is the finite difference method. The fundamental basis of finite differences is the replacement of differential operators with a finite difference operator; i.e., continuous differential operators are replaced with discrete approximations written in terms of a finite number of point values of the unknown function. In a sense, finite difference expressions can be viewed as a reversal of the limit process used to define differential operators.

The finite difference method is implemented by first identifying a finite number of discrete points or nodes in the domain of interest. At each of the nodes, an approximation to the true solution is computed. Derivatives appearing in the governing equations are replaced by discrete nodal values as unknowns. A resulting algebraic system of equations is solved to yield a value of the unknown function at every nodal point in the domain.

Below, some comparisons between the finite difference and finite element approaches are offered with respect to three areas: grid flexibility, boundary condition implementation and efficiency. Certain advantages are inherent in the finite element method because of its piece-wise rather than point-wise approximation (Myers and Weaver 1995). Finite element approximations are especially appropriate for coastal ocean applications where the complexity of the shoreline significantly influences dynamic response and the phenomena of interest occur over several orders of magnitude requiring highly variable resolution throughout the domain. Be advised, however, that no single method can be identified as the "best" choice for all applications. One must consider all factors of the problem including the dynamics of the system, goals of a solution, and available resources when choosing an appropriate numerical technique.

### 2.2.1 Grid Flexibility

An obvious difficulty for regular grids with uniform rectangular elements of the type used for most finite difference schemes is that the edges of the grid cells are necessarily parallel to the two

coordinate axes, with the result that some parts of the coastline have to be approximated roughly by a “staircase” arrangement of line segments. This kind of approximation introduces the most error in the definition of boundary locations and is acceptable only in cases where the influence of the boundary is not dominant.

Formulation of the finite element approximations to the equations naturally deal with fully irregular-shaped domains. The use of triangular elements allows representation of complex features in the geometry, bathymetry, circulation, and point sources. A need to align elements with coordinate axes is removed in the finite element approach.

Secondly, refinement of a region within a finite difference mesh either leads to unnecessary resolution over most portions of the domain or problematic distortion of the rectangular cells in areas remote from the desired region of mesh refinement. When using triangular finite elements, the inclusion of fine grid spacing can be achieved through the local addition of elements in the vicinity of a singularity or steep gradient. This approach eliminates high aspect ratios (ratio of the largest side length to the smallest side length) caused by restrictions to the rectangular cell geometry of finite difference methods.

### 2.2.2 Boundary Condition Implementation

In general, finite difference equations must be specially formulated to accommodate various boundary conditions. For an essential boundary condition, formulation of the right-hand side (RHS) must be altered from the discretized interior equations to compute the influence of a known point source, i.e.:

$$Q\delta(\hat{x}-\hat{0}), \quad (5)$$

where  $Q$  is the magnitude of the point source and  $\delta(\hat{x}-\hat{0})$  is the Dirac Delta function that assumes the value of the point source at  $\hat{x}=\hat{0}$  and is 0 elsewhere in the domain. While the finite difference approximation requires a point value of the RHS function, point values of the Dirac Delta function are not meaningful. The usual approximation of the delta function is a function that is  $C^{-1}$  continuous over the domain, is as compact as the grid spacing will allow, and maintains the correct volume of extraction/injection of the source, i.e.:

$$f_{\delta}(x, y) = \left[ \begin{array}{c} \frac{Q}{\delta x_o \delta y_o} \\ 0 \text{ all other } (x, y) \in \Omega \end{array} \right], \quad (6)$$

where  $x_o, y_o = (0, 0)$  is the location of the concentrated source at node  $o$ .

The treatment of singularities in finite element approximations does not require replacement of the delta function. Finite element equations are based on integrations and the delta function is well defined in this context. Direct assignment of the essential boundary values to the appropriate boundary nodes is the usual treatment for both the finite element and the finite difference method.

The implementation of flux type boundary conditions in the finite difference domain is quite involved. The normal direction of a boundary is poorly approximated by normals to the numerical stepped boundary. Other coordinate systems may be employed for either the equations or the domain, but derivation of the finite difference approximations quickly becomes complex and cumbersome. Finite difference discretizations at flux boundaries is further complicated with the introduction of imaginary nodes exterior to the domain. Nodes on convex-out corners of the boundary require additional nodes in  $x$  and  $y$ , whereas concave-out corners need no additional nodes. This asymmetry adds to the burden of finite difference formulations and the need for multiple forms of the governing equations to properly represent a boundary value problem over the domain of interest.

The finite element method naturally accommodates flux terms via the boundary integrals that arise in the weak form of the equations from an application of Green's theorem (integration by parts). Boundary conditions can be quite general and only need be specified at a late stage of the analysis such that the specification of flux or mixed boundary conditions does not alter the stiffness matrix.

### 2.2.3 Efficiency

Conceptually, interior approximations to the equations are simple to derive using the finite difference approach. Admittedly, application of the finite element method to a set of equations is more mathematically involved though generality of the technique is gained. A significant advantage is realized if one wishes to change the order of the approximation to the equations. Since basis functions in the finite element method are defined at the element level, changing to higher order approximations is achieved with relative ease, whereas increasing the order of accuracy of the finite difference equations involves a complete reformulation of the difference operators.

The regular discretization of the finite difference method allows implementation of certain expedient matrix solution techniques such as those based on approximate factorization, e.g., alternating-direction methods. Efficiency within the finite element method is achieved largely by the execution of localized computations performed on an element-by-element basis. The stiffness matrix that results from the summation of elemental computations has properties of summability, sparseness, and symmetry that facilitate matrix solution. The use of sparse matrix iterative solvers over bandwidth solvers is considerably more efficient in terms of reduced storage requirements and computational speed. In addition, this approach is particularly well suited for computation on parallel architecture machines.

Clearly, the grid flexibility afforded by the finite element method translates into computational efficiency in the elimination of wasteful computations spent on a grid with uniform element size. Finite element grids offer substantial advantages when implemented in situations where water depth varies substantially over the area modeled.

To summarize, the finite element method allows for systematic and efficient development of discrete approximation equations. The approach lends itself to relatively easy implementation of boundary conditions, in particular for second and third type conditions, where proper imposition of normal flux conditions may become difficult. Finally, the finite element method provides sufficiently accurate geometric definition of irregular boundaries where sufficiency is defined by the relative influence that boundary conditions have on the solution of interest.

In general, implementation of the finite element modeling approach involves approximation of the governing equations, selection of a model domain, and discretization of the domain into discrete elements with variable resolution.

### 3.0 UNSTRUCTURED MESH RESOLUTION

The design of a reasonable spatial discretization requires an understanding of the underlying mathematics of the governing equations, the physical processes being described by those equations, and the numerical methods used to obtain the approximate solution. With such an understanding, general expectations concerning the behavior of the solution can be formulated. In truth, control of numerical discretization errors through intelligent grid design is only one, albeit important, way to control solution errors. Other mechanisms include judicious modification of the actual finite element approximation to the governing equations and the appropriate specification and placement of boundary forcing. This report will only address the issue of mesh resolution and the design of finite element grids.

In coastal regions, the large variability in bathymetry, the complexity of coastline features, and the resulting highly nonlinear hydrodynamic response suggest the use of unstructured meshes with graded grid spacing. The variable size and orientation of grid elements results in a better fit of coastlines having arbitrary directions. Furthermore, an unstructured mesh allows a higher density placement of nodes in regions where the dynamics are rapidly varying and leaves a sparse array of nodes in areas where changes are slow. Generally, for realistic simulation of coastal waters, unstructured, graded meshes provide an optimal discretization of the domain and yield the most accurate solution for the least computational resources (e.g., Blain et al. 1994, 1997; Luettich and Westerink 1995; Lynch et al. 1995).

Some theoretical criteria and empirical approaches for nodal placement and grid resolution are presented below, as well as the latest advances in the area of mesh design. One must keep in mind, however, that many possibilities exist in the construction of a mesh and that the necessary resolution cannot be known exactly *a priori*. Performance of a mesh must ultimately be tested through simulation of the physical processes of interest and a well-formulated convergence study. If the solution to the discrete problem is not well converged with respect to the mesh resolution, model error cannot be attributed to physical causes.

### 3.1 Theoretical Aspects of Mesh Resolution

#### 3.1.1 Spatial Truncation Error Analysis

Truncation error is defined as the difference between a true derivative and the discrete approximation to that derivative. In truncation error analyses, variables of a model equation are typically approximated using a Taylor series expansion (this approach actually forms the basis for the derivation of finite difference derivative operators, e.g., Lapidus and Pinder 1982; Celia and Gray 1992), i.e.:

$$u(x) = \sum_{n=0}^{\infty} \left. \frac{d^n u}{dx^n} \right|_{x_0} \frac{(x-x_0)^n}{n!} \quad (7)$$

or

$$u_{i+1} - u_i = \frac{du}{dx} \Big|_{x_i} (x_{i+1} - x_i) + \frac{d^2u}{dx^2} \Big|_{x_i} \frac{(x_{i+1} - x_i)^2}{2} + \dots, \quad (8)$$

where  $u(x)$  is the dependent variable and  $x_0$  is the point about which expansion takes place. From a Taylor series representation, the continuous form of the model equations is subtracted. The remaining terms of the Taylor series expansion constitute the truncation error:

$$TE \equiv \frac{(u_{i+1} - u_i)}{(x_{i+1} - x_i)} - \frac{du}{dx} \Big|_{x_i} = \frac{d^2u}{dx^2} \Big|_{x_i} \frac{(x_{i+1} - x_i)^2}{2} + \dots. \quad (9)$$

Typically, it is the higher order derivative terms of the Taylor series that are truncated from the discrete representation of a continuous derivative and account for the error in the approximation. These terms maintain a dependency on mesh spacing that can be utilized to establish resolution criterion.

Expectations regarding the behavior of a model solution together with knowledge of the truncation errors associated with the discrete equations influence the design of spatial resolution. Truncation error analysis generally confirms the intuitive arguments that finer resolution is required in regions of rapid changes of the computed solution. In areas of rapid variation, higher order derivatives in the truncation error terms are correspondingly large. Therefore, to control truncation error, a smaller grid spacing in these regions is needed. The truncation error associated with one-dimensional (1-D) forms of the model equations can provide guidelines for mesh design (see Celia and Gray 1992 for excellent examples). Truncation error analysis suggests that variable grid resolution may achieve a more accurate solution than the use of constant nodal spacing for the same number of nodes. As detailed in a later section, minimization of the spatial truncation errors can be applied as a constraint for the automatic generation of unstructured meshes.

### 3.1.2 Stability Constraints

Partial differential equations describing the coastal ocean environment are classified as initial value problems and require the discretization of time derivatives. Expressions for the time derivatives in discrete form result in an algorithm that marches the solution forward in time by small increments or time steps. These time marching algorithms can be subject to temporal instabilities that are characterized by unbounded growth of the discrete solution as time increases. The condition of stability, then, imposes a constraint that the difference between the analytical and numerical solutions for a set of equations remain bounded for all time. Formal stability analyses (e.g., the use of amplification factors, characteristic, matrix, and Fourier analyses) can be approached in several ways, but in general, their application is limited to linear approximating equations. This linear analysis, in turn, can provide guidelines for nonlinear solutions. A more detailed discussion of stability analysis is beyond the scope of this report and the reader is referred to texts such as Celia and Gray (1992).

A relationship between stability and mesh resolution arises from the form of the stability constraint derived for coastal ocean model equations. Fourier analysis of a standard second order, hyperbolic (wave) equation leads to a term of the form:

$$v \equiv (c\Delta t/\Delta x), \quad (10)$$

where  $\nu$  is referred to as the Courant number. The Courant stability criterion requires that  $\nu \leq 1$  for an explicit time stepping scheme (unknown model variables are at a single time level). Since water velocity,  $c$ , increases with water depth and variable time step sizes,  $\Delta t$  may not be practical; a uniform grid spacing leads to unnecessarily frequent computation everywhere except at the point of maximum depth. Most implicit time stepping schemes (unknown model variables are expressed at more than one time level) are stable if the Courant criterion is violated, but truncation error generally increases with local Courant number. Therefore, it is desirable to design grids whose elements change in size according to water depth so as to keep the Courant ratio fairly uniform over the entire grid. Recasting the Courant criterion as:

$$\frac{c^2 T^2}{A} \leq 1, \quad (11)$$

where  $A$  is the element area of an equilateral triangle and substituting  $c^2 = gh$ , we are left with  $h/A$ , the ratio of the element area to local depth. In practice, this should be kept as uniform as possible over the grid. The Courant criterion is another means by which mesh resolution can be designed.

### 3.1.3 Element Selection

Within the finite element method, a number of element types may be accommodated through a coordinate mapping from global elements  $(x, y)$  to local, master elements  $(\xi, \eta)$ , i.e.:

$$dx dy = \det J d\xi d\eta, \quad (12)$$

where  $\det J$  is the determinant of the Jacobian matrix defined

$$J \equiv \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix}. \quad (13)$$

There is a requirement that the Jacobian of these coordinate transformations remain nonzero at all points within the element area being mapped. As distortion of the element increases, evaluation of the element integrals becomes more difficult and more integration points become necessary. When the element becomes too distorted, the integral can no longer be evaluated due to a 0 value of the Jacobian within the element and the finite element approximation fails. Thus, the Jacobian can be used as a flag to check allowable element shapes and angles. A change in sign at the corner nodes of the element indicates that the Jacobian has become 0 somewhere within the element. Linear coordinate transformations should always be used except in special cases where higher order elements are needed, but care must be taken to avoid element distortion.

Linear triangular elements have easily defined basis functions, and the element integrals associated with linear triangles are readily evaluated. Furthermore, Walters and Henry (1995) demonstrate that equilateral triangles reduce truncation errors in the computed numerical solution. Linear triangles also allow isotropic wave propagation (Foreman 1984). One further observation is that no more

than six elements should be permitted to meet at a vertex to reduce the generation of numerical noise (Platzman 1981). Elements should vary smoothly in size with area ratios of neighboring elements not to exceed two.

## 3.2 Empirical Approaches to Mesh Resolution

### 3.2.1 Dimensionless Wavelength

A widely used strategy in the generation of unstructured grids utilizes the wavelength to grid size ratio:

$$\frac{\lambda}{\Delta x} = \frac{T\sqrt{gh}}{\Delta x}, \quad (14)$$

where  $\lambda$  is the wavelength,  $\Delta x$  is the grid spacing,  $T$  is the wave period,  $g$  is the gravitational acceleration, and  $h$  is the local water depth. This ratio is generally set to some constant value equal to 40 or less (LeProvost and Vincent 1986; Westerink et al. 1994a, b). In shallow water, wave speed,  $c$ , is related to the local water depth by  $c^2 = gd$ . This condition is equivalent to requiring that the spatial sampling interval (number of gridpoints per wavelength) be kept as nearly uniform as possible over the grid. LeProvost and Vincent (1986) assume a level of 13–14 points/wavelength as the standard for comparison of different meshes. Others (e.g., Gray and Lynch 1977; Kinnmark 1986; Luetlich et al. 1992) found that amplitude and phase characteristics of a tidal wave are best represented by 15–25 uniformly spaced grid cells.

Assuming that tidal waves propagate near their shallow-water wave speed:

$$c = \lambda/T = \sqrt{gh}, \quad (15)$$

then  $\lambda$  will decrease as a wave propagates into shallower water. To maintain a constant  $\lambda/\Delta x$ , the grid spacing,  $\Delta x$ , must decrease as  $\sqrt{h}$ .

The dimensionless wavelength criterion is not without shortcomings. First of all, the criterion stems from an assumption of 1-D linear, frictionless waves moving over constant depths. Generally, a target of 25–150 is sought for  $\lambda/\Delta x$ , but this is found to be unsatisfactory because of its inability to identify the two-dimensional (2-D) structure of the waves associated with features such as amphidromes and where circulation is forced by 2-D coastline or bathymetric features (Westerink et al. 1994a). In these cases, the actual wave number content of the response is much greater than that predicted by the 1-D criterion. Additionally, the rate of change of wavelength or associated significant gradients in the response, which occur as the waves propagate over steep topographic changes, is not considered. Such steep gradients are known to occur at the continental shelf break and continental slope. The modeler is also required to know the period and location of the characteristic waves associated with the hydrodynamic response of the system. Though this criterion is widely implemented, it must be used with caution.

### 3.2.2 Topographic Length Scale

Recently, a 1-D criterion, the topographic length scale, was introduced by Hannah and Wright (1995). This criterion requires the grid spacing to be less than the topographic length scale times a constant:

$$\Delta x \leq \alpha L_h, \quad (16)$$

where  $\Delta x$  is the mesh spacing,  $\alpha$  is a constant, and  $L_h$  is the topographic length scale. The topographic length scale can be expressed as a ratio of the bathymetry to the bathymetric gradient  $\nabla h$ :

$$L_h \sim h/(\Delta h/\Delta x), \quad (17)$$

which yields an expression

$$\nabla h/h \leq \alpha \quad (18)$$

over an element. This relationship in Eq. (18) can be used as a mesh generating tool where  $\alpha$  is the mesh generation parameter. Use of the topographic length scale incorporates both the bathymetry and the gradient of bathymetry into the mesh generation process. The criterion, however, fails in the limit as the bathymetric gradient approaches 0. Such conditions occur, for example, in locally constant depth regions, which severely limits the utility of the topographic length scale.

### 3.3 Recent Research in Mesh Generation

A basic difficulty with mesh generation lies in achieving an acceptable compromise among the following conflicting design criteria: (a) boundary line elements of the grid should fit the boundaries of the problem domain with sufficient accuracy, (b) element area should be proportional to the local value of some scalar field defined over the grid, and (c) the Jacobian of the element coordinate transformation should be non-zero for the sake of accuracy in subsequent numerical calculations. A general survey of grid generation approaches is given by Simpson (1979), and Cescotto and Wu (1989) discuss a number of specific algorithms for element construction.

#### 3.3.1 Local Truncation Error Analysis (LTEA)

Hagen and Westerink (1995) demonstrated that truncation errors are predominant at the coast and near the continental shelf break and slope. Use of the  $\lambda/\Delta x$  criterion does not indicate a need for resolution in these localized areas. Hagen and Westerink (1995, 1996) have reexamined the use of truncation error analysis to derive guidelines for the automatic generation of finite element meshes. Taylor series expansions in the generic discrete nodal form of the conservation equations of mass (cast in the Generalized Wave Continuity form (Kinnmark 1986)) and momentum lead to expressions for the local or nodal truncation error. Fine mesh computations provide estimates for the spatial derivatives of velocity and elevation in the local truncation error expressions. With a goal to reduce local, second-order truncation errors, the physics of the equation are incorporated into the grid generation process. Values of the truncation error, once obtained, are forced to maintain a specified level over the domain, thereby effectively generating a variably spaced mesh. Imposed multiples of change (IMCC) minimize the contribution of odd-order terms in the local truncation errors by restricting the magnitude and relative importance of even-order terms. The IMCC is ultimately utilized to generate a mesh given the minimum grid spacing desired at the coast. This approach appears promising for implementation as an automated mesh generation approach that incorporates the physics of the equations.

#### 3.3.2 Dynamic Gridding

In dynamic gridding, the spatial nodes of a grid are non-stationary in time. This approach takes advantage of the ideas of space-time truncation error cancellation. The majority of nodes can be

allocated to regions of the domain where they are most needed based on the dynamics of the problem, while relatively few nodes can be used in regions where the solution is changing slowly. Dynamic gridding is employed for the so-called characteristic methods that include Eulerian-Lagrangian methods, the method of characteristics, and particle tracking. The solution procedure in these approaches is obtained by tracking characteristic curves through space and time, where characteristic curves are defined by:

$$\frac{\partial x}{\partial t} = V, \quad (19)$$

where  $V$  is the advective velocity of the system and  $x$  and  $t$  are the space and time independent variables, respectively. Actually, numerical schemes track along secondary characteristics associated with the diffusive portion of the governing equation. Nodal spacing should increase as time increases since diffusion spreads a pulse spatially with increasing time.

The computational overhead of such approaches where nodal locations are a function of time can become quite high. The interdependence of nodal position and solution variable yields a highly nonlinear problem that requires significant iteration for solution at each time step. The idea of moving nodes with the advective velocity may be appealing and conceptually simple, but such computations can lead to severe grid distortion and the instance of tangled grids, places where characteristic lines cross, causing nodes to cross element boundaries. An even greater drawback is that optimal nodal placement depends on only one dependent variable precluding a good computational solution with respect to the remaining dependent variables. Furthermore, the problem remains to specify an initial number and distribution of nodal points that will allow propagation of disturbances from the boundary and remain adequate throughout the entire simulation.

Alternative approaches involve selective refinement on fixed grids that incorporate characteristic type approximations. Refinement of a particular region within the global mesh can be obtained by solving for the deviation of dependent variables within an element. Criteria for refinement can be established for each variable based on an error norm or estimates of deviation. Conversely, refinement may be eliminated via a 0 deviation threshold. A significant concern of the selective refinement approach is the strain placed on stability constraints with respect to the time step size. Computationally, this method is most attractive when time step can vary spatially over the domain.

Needless to say, the specification of unstructured mesh resolution remains an open question and an area subject to further extensive research. In the intervening period, mesh construction relies on the theoretical and empirical criteria described within modeling experience and intuition.

#### 4.0 EXISTING TOOLS FOR MESH GENERATION

High-quality, finite element predictions have at their core a well-constructed finite element mesh that provides the detail and resolution adequate to capture physical reality of the processes under study. As previously stated, one of the primary advantages of the finite element method is the inherent grid flexibility of the mesh for capturing complex shoreline geometry, irregular topographic features, and highly variable circulation patterns. It is precisely this desired grid flexibility that precludes the timely development of finite element grids "by hand" and has driven many to seek more automated procedures for finite element mesh generation.

In the following sections, several of the known, more advanced mesh generation tools are briefly described. Subsequent discussions pertain to the first of those mesh generation tools, *ACE/gredit*, which is selected for its multitude of features and its window-based, intuitive environment. Detailed information regarding the utilization and implementation of *ACE/gredit* in semi-automatic mesh construction and grid editing are presented. The authors' intent is to supplement the *ACE/gredit* User's Manual (Turner and Baptista 1991) and to aid the novice with the manipulation of grids within *ACE/gredit* to accomplish grid creation and editing for common applications. Insights from the authors' own experiences are also provided.

#### 4.1 *ACE/gredit*

*ACE/gredit* is a software package that facilitates flexible, interactive, semi-automatic generation of 2-D triangular finite element meshes and allows for mesh modification and display. This software was developed by Antonio M. Baptista and Paul J. Turner at the Oregon Graduate Institute and made available by the Center for Coastal and Land-Margin Research through its Software Affiliates Program (Turner and Baptista 1991, 1992). Within *ACE/gredit*, the user maintains full control of the grid design while letting the computer perform the algorithmic, time consuming tasks. Knowledge of some basic terminology for the components of a finite element mesh, together with the menu-driven interface in *ACE/gredit*, form a relatively efficient and easy to use grid editing tool.

A reference bathymetry field and raw, ordered coastal outline data are all that are needed to automatically generate a spatial distribution of points. The placement density of these points is subject to a scalar criterion defined by the user. In the triangulation phase, these points are mapped onto nodes and triangulated into three-node triangular elements with linear shape functions using the Fortune's sweepline method. The *ACE/gredit* software allows node-wise and element-wise regional and local editing and facilitates model diagnostics.

#### 4.2 *TriGrid*

The *TriGrid* software developed by Henry and Walters (1993) is aimed at automatic generation of finite element networks in two horizontal dimensions for use with models of coastal circulation. The philosophy contained within this grid generation tool is that partial automation of the most subjective stages of grid design is far more cost-effective than full automation. *TriGrid* assumes a knowledgeable and experienced user, such that it is more practical to include the modeler whenever necessary in the grid design process while maintaining automatically an up-to-date grid structure.

Within *TriGrid*, a network generator creates a depth-interpolation network using sub-sampled contour and shoreline data and depth soundings. An initial model network is generated from the interpolation network using geometric cells, the centroids of which become the nodal points. Interior points for the depth-interpolation grid are obtained from digitized depth soundings or selected points from digitized depth contours. Boundary points are selected from digitized land boundary data. Finally, a depth interpolation grid is constructed from the interior and boundary points using a triangularization algorithm.

The problem geometry is maintained using Delaunay triangulation where all boundary connections are preserved and all connections exterior to the boundary are removed. Linear interpolation between nodes is implemented in part to avoid overshoot with higher interpolation schemes. Nodal densities of the triangulated mesh are determined by some scalar function. Ultimately, a

graphics-based editor permits interactive modification of the network and performance quality checks after a preliminary grid has been constructed.

### 4.3 Software Packages Under Development

*FEGPAK* is a series of finite element mesh generation and modification utilities developed by Dr. Christopher E. Naimie at Dartmouth College for use in a research environment. As such, implementation of these programs requires significant familiarity, knowledge, and care on the part of the user. However, in *FEGPAK* the user maintains considerable control of the triangulation procedure in relation to placement of the boundary and the location of the nodes in relation to bathymetric contours. Operations on portions of the mesh are handled with ease as is the joining of separate meshes.

*GENESIS* is a finite element grid generation effort currently under development at the Bedford Institute of Oceanography, Dartmouth, Nova Scotia, under Dr. David A. Greenberg in conjunction with Dr. Florent Lyard from the Laboratoire des Ecoulements Geophysiques et Industriels (LEGI) in France. This software package is merging the various sources for bathymetry and shoreline data such that they are directly accessible to the automatic finite element grid generation process.

At the Mississippi State University, Engineering and Research Center for Computational Field Simulation, an intensive effort headed by Dr. David L. Whitfield is underway to fully automate unstructured grid generation for computational fluid dynamics problems. The system has built in options for mesh generation such that the combination of techniques which produce the "best" grid for ocean model applications may be utilized. In this comprehensive system, extraction routines will generate the mesh geometry from available data bases with intelligent processing which, for instance, will filter a specified level of detail if not desired, compute a grid where the element size is coupling with depth data or some other scalar parameter, and interpolate depth values from multiple sources of bathymetry. Point densities on the boundary and the approximate number of points/elements desired in the mesh must be specified. Measures are in place for checking min/max angles, skewness, and length ratios of triangular elements. The edge-swapping algorithm allows either the minimization of maximum angles (preferred/advancing front triangulation) or the maximization of minimum angles (Delauney triangulation). The package currently operates from a mix of batch and interactive modes.

The grid generation and editing packages presented by no means comprise an exhaustive list. The software discussed is intended to familiarize the reader with the existence of other packages and to emphasize that finite element grid generation is an evolving effort combining new research in grid generation algorithms with advanced computer graphics capabilities.

### 5.0 APPLICATIONS USING ACE/gredit

The following sections detail implementation of the *ACE/gredit* grid editing and generation software for a number of common mesh applications. These applications include generation of a mesh from raw bathymetric and shoreline coordinate information, the addition or deletion of regions within an existing grid, refinement of shoreline or mesh resolution, localized node and element edits, and model analysis that includes the contouring of bathymetry or other model surface fields and the location of nodes and elements within the mesh for diagnostic purposes.

### 5.1 Definition of Terminology

A few of the building blocks used in the construction of a finite element mesh, as well as the terminology employed by the *ACE/gredit* software, are defined here and graphically shown in Figs. 1–3.

An edit grid is a collection of triangular elements and nodal points that compose the finite element mesh. Elements within *ACE/gredit* are restricted to linear triangles and so have three vertices labeled as nodes. As will later be detailed, the edit grid is the final product of the semi-automatic grid generation process and may be modified using local or regional node and element edits in *ACE/gredit*. The coastal outline of an edit grid plus the outline of any islands contained within the mesh domain are labeled the edit grid boundary. Any elements created during the triangulation process that lie outside the edit boundary will be clipped. Build points are a collection of points not necessarily identical to the nodes in an edit grid that are triangulated to form a new edit grid. The build points also contain point-wise bathymetric information that can be triangulated to form a background grid having a convex hull boundary. A background grid supplies the reference bathymetry and is comprised of elements and nodes just as the edit grid. Bathymetry from the background grid will be interpolated onto the edit grid nodal points. A coastal boundary is the raw

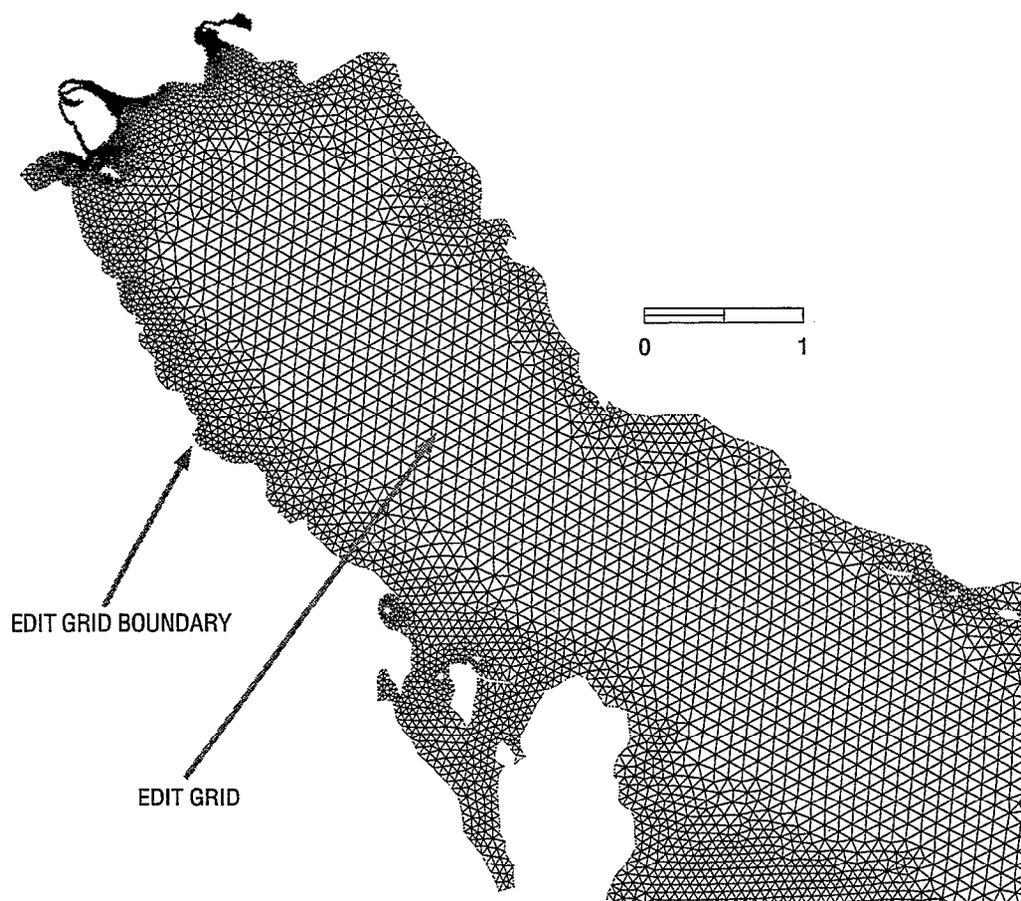


Fig. 1 — Graphical depiction of the edit grid boundary and edit grid used within the *ACE/gredit* software

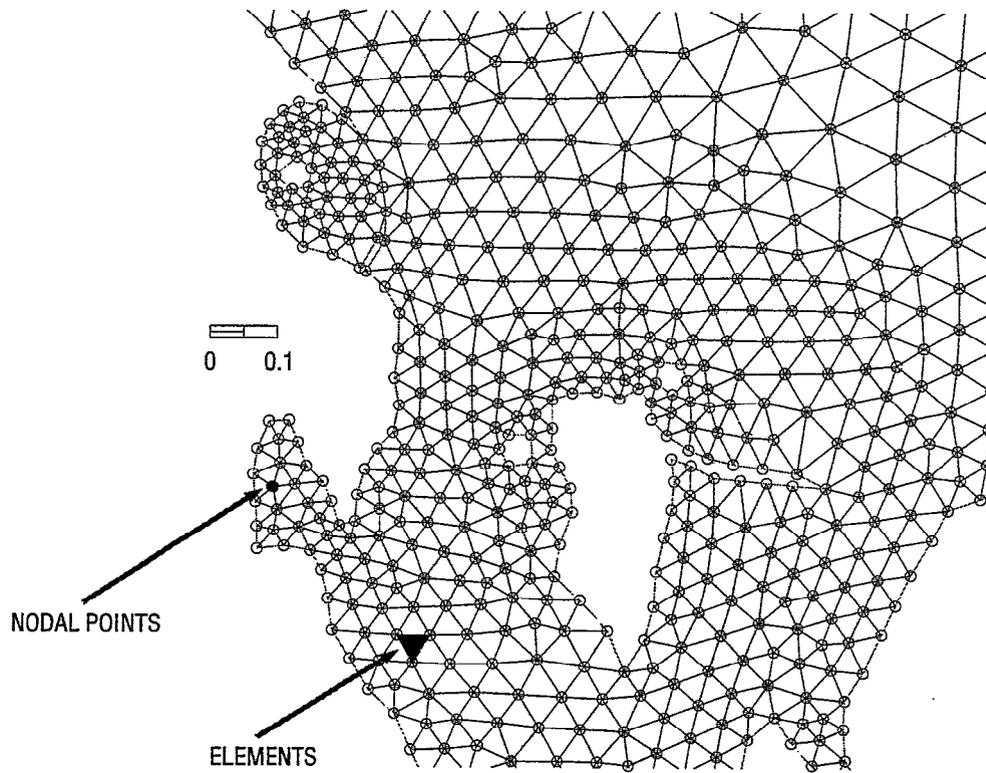


Fig. 2 — Graphical depiction of nodal points and elements used within the *ACE/gredit* software

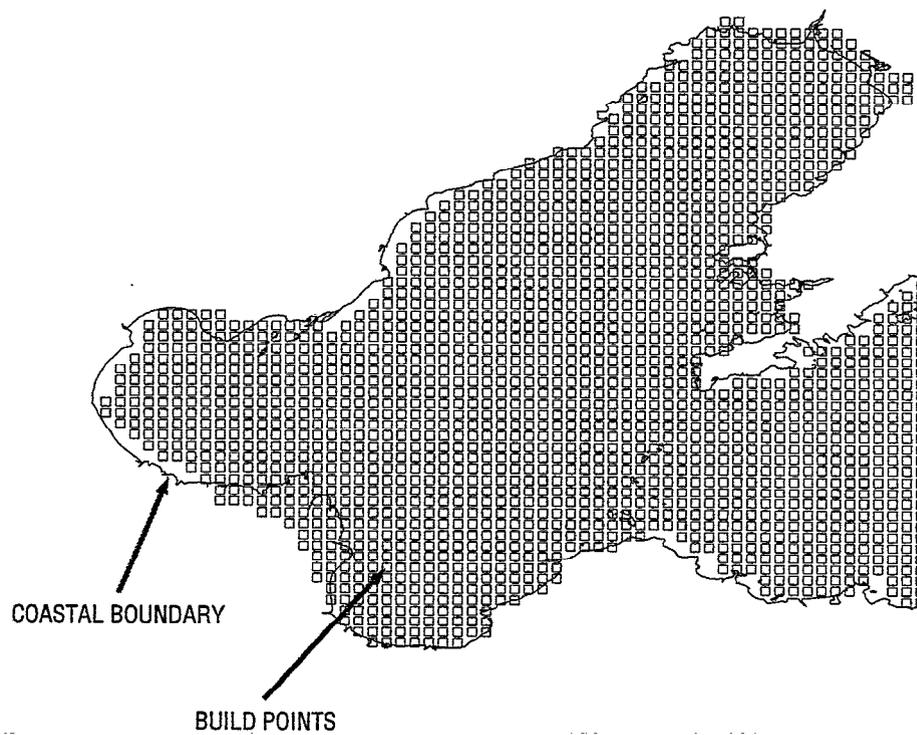


Fig. 3 — Graphical depiction of the coastal boundary and build points used within the *ACE/gredit* software

shoreline data used to automatically create an edit grid boundary or it can be used as a guide in the manual creation of an edit grid boundary.

The various grid components just defined can be viewed individually or in groups within *ACE/gredit* by selecting their icons from the main panel on the left and scaling the display according to the grid components selected. The main panel consists of three groups of grid components each associated with either the edit grid, background grid, or the build points.

Found under *EdGr* in the main panel are eight items associated with edit grid: the mesh (first square in the left column), isolines of a nodal variable (i.e., bathymetry, first square in the right column), the edit grid boundary (second square in the left column), a filled edit grid (second square in right column), node numbers (third square in right column), element numbers (third square in right column), values of the nodal variable (i.e., bathymetric elevations, fourth square in the left column), and enlarged circles at the nodal points (fourth item in right column). The background grid, noted by *BkGr* in the main panel, has three components for display: the mesh (first square in the left column), isolines of the node values (first square in the right column), and the background grid boundary (second square in the left column). Build points themselves can be viewed under *BldP*.

To display an individual or set of grid components, simply click the corresponding boxes in the main panel and scale the view if necessary. To scale the view, select DISPLAY/AUTOSCALE from the menu bar. This opens a DISPLAY window, which lists the following grid components that can be used to scale the view: EDIT GRID, EDIT BOUNDARY, BUILD POINTS, BACKGROUND GRID, and COASTAL OUTLINE. Click on the square next to the item or items to be viewed and then click ACCEPT. The grid components selected in the main panel for the area covered by the item(s) selected in the DISPLAY window will be shown.

## 5.2 Mesh Creation

The building blocks of a finite element mesh are the coastal outline and bathymetry data. The mesh creation feature in *ACE/gredit* uses these two pieces of information to construct a 2-D finite element grid composed of three-node triangular elements. The resulting mesh is termed the "edit grid." The shoreline and bathymetric data sources are often cast in a variety of forms that must be converted into file formats accepted by *ACE/gredit*. Within *ACE/gredit* these raw data files are then transformed through a series of steps into an edit boundary and background grid, the building blocks for automatic grid generation. Detailed below are the steps necessary to generate a background grid containing bathymetric information and create an edit boundary from coastal outline data. Having the background grid and edit boundary, a step-by-step procedure for automatically creating a triangular mesh whose nodal distribution is defined by some user-defined criterion is presented.

### 5.2.1 Bathymetry

The first step in creating an edit grid is converting the bathymetric information into a background grid. Bathymetry must initially enter *ACE/gredit* through an ASCII build point file that contains either latitude and longitude or Cartesian coordinate pairs and the corresponding bathymetric depth at that location. Note that one should strive to define the areal extent of the background grid

bathymetry to be much larger than the coastal outline of the desired mesh. The build point file format is as follows:

Line 1: string, usually the name of the build point file or source of bathymetry

Line 2: number\_of\_build\_points

Line 3: build\_point\_number longitude latitude depth

or (x , y)

...repeated number\_of\_build\_point times

Having created a build point file with the necessary bathymetric information, this file is now read into the *ACE/gredit* software. The steps within *ACE/gredit* are outlined below:

\* Start *ACE/gredit*:

– type *xmgredit edit.grd*

The file *edit.grd* is any edit grid created previously and is used here simply to start the *ACE/gredit* software.

\* Select FILE/OPEN from the menu bar, which opens a *read data* window.

– Select the build point file to load.

– Click on the box next to *read object*. Select *build points* from that list.

– Click on *OK*.

*ACE/gredit* requires that grid manipulations be exercised in a Cartesian reference frame and bathymetry coordinates be expressed in meters. Build points defined relative to a spherical latitude/longitude coordinate system must be transformed to a Cartesian coordinate system. This is accomplished using the Carte Parallelogramatique Projection (Pearson 1990) or CPP transformation. Alternative projections may be used but must be executed by the user outside of the *ACE/gredit* software. To apply the CPP projection:

\* Select EDIT/*edit over grid/regions* from the menu bar.

\* Select *CPP projection...* from the *edit region* window.

\* The *CPP map projection* window requires the following input:

– Median longitude and median latitude values for the region covered by the bathymetry data.

– Data type to be converted. This can be specified by clicking on the box next to *Apply to:*. The default is *ALL*. Select *build points*.

– Click on *OK* in *CPP map projection* window.

To save the CPP conversion of the bathymetric build points:

\* Select FILE/SAVE from the main menu.

– Select *build points* as the write object.

– Enter the file name.

– Click *accept*. The file will be written in the directory where *ACE/gredit* was started.

– Click on *done* and the *write data* window will disappear.

To view the CPP-converted build points:

- \* Select DISPLAY/AUTOSCALE from the main menu.
  - Click on the square next to *build points* and then *accept* in the *scaling* window.
  - Click on *done* in the *scaling* window. The *scaling* window will disappear.

To utilize the bathymetric information contained in the CPP or other planar referenced build points, a background grid must be created. The first step is triangulation of the build points followed by computations of the boundary of the background grid. The result is a grid whose boundary consists of a convex hull. The nodal locations for this grid are identical to that of the build points and are assigned the bathymetry value of each build point. The procedure for generation of this background grid is described below:

- \* Select BOUNDARIES from menu bar.
  - If *clear external boundary* is selectable, it means that an external boundary is present and should be cleared, so select *clear external boundary*.
- \* Select BUILD/TRIANGULATE BUILD POINTS... from menu bar.
  - Enter a minimum distance in *triangulate build points* window. Experience dictates that the minimum distance must be greater than 0.
  - Click on *apply* in *triangulate build points* window.
- \* Select BOUNDARIES/COMPUTE BOUNDARY from menu bar.
  - Save background grid.
- \* Select FILE/SAVE from the main menu.
  - Select *background grid* as the write object.
  - Enter the file name.
  - Click *accept*. The file will be written in the directory where *ACE/gredit* was started.
  - Click on *done* and the *write data* window will disappear.

### 5.2.2 Coastline

The second step toward the automatic generation of an edit grid is the specification of a coastline. Shoreline data must initially enter *ACE/gredit* through an ASCII coastal outline file. A coastal outline file contains either the latitude and longitude or Cartesian coordinate pairs of the shoreline and can be created manually or extracted from an outside source. *ACE/gredit* ultimately requires coordinates to be expressed in meters. The format for the coastal outline file follows:

Line 1: string, usually the name of the coastal outline file or source of shoreline data

Line 2: 1 (for the number of boundaries)

Line 3: number\_of\_points\_in\_the\_boundary

Line 4: longitude latitude

or (x , y)

...repeated number\_of\_points\_in\_the\_boundary times

The most widely available source for shoreline data is the World Vector Shoreline (WVS) and the WDBII-CIA data bases. The detail of the coastal outline within these sources is represented by short line segments assigned to regularly spaced bins covering the entire globe. The line segment format is problematic for any mesh generation software in that it is most difficult to order shoreline points from unrelated line segments and distinguish which segments are associated with the closed coastlines of islands. *ACE/gredit* relies on boundaries that are organized such that the outer boundary of the mesh is oriented counterclockwise and the internal boundaries or islands are oriented clockwise.

A *background* coastline can be read in as a *coastal boundary* and used as a stencil.

- \* Select FILE/OPEN from menu bar.
- \* Select *coastalboundary* as the read object.
  - Select the file name.
  - Click *OK*.

Manual creation of the coastline then begins by selecting points along the stencil coastal outline. Initially, a crude representation of the coastal outline is acceptable. This manually created coastline is achieved by:

- \* Select BOUNDARIES/CREATE EXTERNAL BOUNDARY from the menu bar.
  - Click the left mouse button along the coastal boundary to generate points in the shape of the coastline.
  - Click the middle mouse button when complete.
- \* Select the DISPLAY BOUNDARY button under the *EdGr* portion of the main panel to show the created coastline.

Additional points can be added to the crude coastal outline to improve the realism of the coastline shape. Boundary points can be added in the following manner:

- \* Select BOUNDARIES/INSERT BOUNDARY POINTS from menu bar.
  - Click the left mouse button on the two boundary points that will be on either side of the new point; the two points must be selected in a clockwise fashion.
  - Click the left mouse button on the location of the new point.
  - Click the middle mouse button when all of the new boundary points have been added.
  - Click the *DRAW* button on the left panel to see a current version of the coastline.

Making the coastline shape more realistic sometimes only requires shifting the locations of a few coastal boundary points; these points can be moved by:

- \* Select BOUNDARIES/MOVE EXTERNAL POINT from menu bar.
  - Click the left mouse button on the point to move and click a second time on its new location.
  - Click the middle mouse button when finished moving boundary points.
  - Click the *DRAW* button on the left panel to see a current version of the coastline.

To save the created coastal outline:

- \* Select FILE/SAVE from the menu bar.
  - Select *edit boundary (X,Y)* as the write object.
  - Enter the file name.
  - Click *accept*. The file will be written in the directory where *ACE/gredit* was started.
  - Click on *done* and the *write data* window will disappear.

Again, conversion of the coastline to a Cartesian reference frame is necessary. It is important to apply the same transformation used to convert the build points associated with the background grid, in this case the CPP transformation. To perform the CPP transformation on the coastline:

- \* Select *EDIT/edit over grid/regions* from the menu bar.
- \* Select *CPP projection...* from the *edit region* window.
- \* The *CPP map projection* window requires the following input:
  - The same median longitude and median latitude values used to create the background grid so that the coastline and bathymetry information will be consistent.
  - Data type to be converted. This can be specified by clicking on the box next to *apply to*. The default is *ALL*. Select *coastal boundaries*.
  - Click on *OK* in *CPP map projection* window.

To save the CPP conversion of the coastal outline:

- \* Select FILE/SAVE from the main menu.
  - Select *edit boundary (X,Y)* as the write object.
  - Enter the file name.
  - Click *accept*. The file will be written in the directory where *ACE/gredit* was started.
  - Click on *done* and the *write data* window will disappear.

To view the CPP-converted coastal outline:

- \* Select DISPLAY/AUTOSCALE from the main menu.
  - Click on the square next to *coastal boundaries* and then *accept* in the *scaling* window.
  - Click on *done* in the *scaling* window. The *scaling* window will disappear.

### 5.2.3 Automatic Grid Generation

The automatic mesh generation capability in *ACE/gredit* utilizes the background grid and coastlines created in the previously outlined steps to create a spatial distribution of build points that are subsequently triangularized to form a mesh of elements. The procedure to obtain such a finite element grid is detailed below:

The CPP transformed background grid containing the bathymetric information and the appropriate CPP coastal outline for the desired mesh region are to be read into *ACE/gredit*.

- \* Select FILE/OPEN from the menu bar, which opens a *read data* window.
  - Select the CPP background grid file to load.

- Click on the box next to *read object*. Select *background grid* from the list.
- Click on *OK*.
- \* Select FILE/OPEN from the menu bar, which opens a *read data* window.
  - Select the coastline file to load.
  - Click on the box next to *read object*. Select *edit boundary (X, Y)* from the list.
  - Click on *OK*.

To verify the coordinate system correspondence of the background grid and coastal outline, view these components together by:

- \* Select DISPLAY/AUTOSCALE from the main menu.
  - Click on the square next to *coastal boundary* and then *accept* in the *scaling* window.
  - Click on *done* in the *scaling* window. The *scaling* window will disappear.

The next step in the process of creating a finite element mesh is to generate a distribution of build points within the coastal outline that utilizes the bathymetric data supplied by the background grid. This array of build points is triangulated to form the nodes and elements of the new edit grid. The density of the build points in the mesh can be defined by the user according to a scalar criterion. Several options are provided by *ACE/gredit*. In the first option, the user selects the minimum dimensionless wavelength or the minimum number of points needed to represent one wavelength. This condition also requires input of the shortest allowable wave in hours and a grid delta or spacing. Other options include the minimum element area and a maximum Courant number criterion using previously computed velocity field. The latter two options will not be discussed or demonstrated in this report. To automatically generate build points according to the minimum dimensionless wavelength criterion:

- \* Select BUILD/AUTOMATIC PLACEMENT from menu bar.
  - Enter the minimum dimensionless wavelength, the shortest allowable wave in hours, and the grid delta.

See App. A for examples of applications employing the minimum dimensionless wavelength criterion for semi-automatic grid generation.

- Select the square next to CREATE BOUNDARY.
- Specify the minimum depth at the boundary and the minimum depth desired in the created mesh or retain the default values.
- Click on *accept*.

If a *no boundary formed* window appears, re-read the background grid and coastal boundary and retry the automatic placement.

Within the automatic build point placement, the created boundary is one that conforms to exterior locations of the background build points if the background grid falls within the coastal outline over any portion of the domain. One may desire to preserve the original coastal outline boundary. Once the placement of build points along the boundary of the background grid and on the interior of that boundary is complete, the points from the original coastal outline can be included in the current set of build points by:

- \* Select BUILD/MERGE BOUNDARY TO BUILD POINTS from the menu bar.

Once the edit grid build points are created and the boundary points are included, triangulate the build points inside the external boundary:

- \* Select BUILD/TRIANGULATE BUILD POINTS... from menu bar.
  - Enter a minimum distance in *triangulate build points* window. Experience dictates that the minimum distance must be greater than 0.
  - Click on *apply* in *triangulate build points* window.

For finite element model stability and to reduce computational errors associated with the mesh, triangular elements within the mesh should be as close to equilateral as possible. To make minor adjustments to the nodal positions to achieve this goal, a *springs* utility is included in *ACE/gredit*. To apply *springs*:

- \* Select BUILD/SPRINGS from the menu bar.

To keep elements inside of the exterior boundary and outside of any interior boundaries:

- \* Select BOUNDARIES/COMPUTE BOUNDARY from menu bar.

Of course, it is highly desirable to save the semi-automatically generated edit grid before proceeding with any further modifications:

- \* Select FILE/SAVE from the main menu.
  - Select *edit grid* as the write object.
  - Enter the file name.
  - Click *accept*. The file will be written in the directory where *ACE/gredit* was started.
  - Click on *done* and the *write data* window will disappear.

### 5.3 Mesh Modification and Editing

Once edit grids are formed and loaded into *ACE/gredit*, modifications can be made to enhance or degrade resolution in local or regional areas or point edit individual elements or nodes either on the interior or boundary of the domain.

#### 5.3.1 Element Skewness

To reiterate, one factor associated with a viable, accurate computational mesh is the proximity of the triangular elements to equilateral triangles that have 60° angles at each nodal vertex. As noted during the discussion of automatic grid generation, *springs* is one mechanism that assists in the adjustment of nodes toward more equilateral elements. The *springs* approach, however, is incomplete and a final phase of *hand* edits is required to adjust element skewness to a particular user-defined threshold.

The skewness of an element is a measure of its deviation from an equilateral triangle. More specifically, skewness is computed as the ratio of the longest edge of an element divided by the element's equivalent radius equal to  $\sqrt{Area_{element}/\pi}$ . A skewness check of grid elements marks elements that have skewness values higher than a specified value. Generally, skewness between 3.9 and 3.4 is considered achievable. Smaller skewness values indicate elements that more closely approach the ideal equilateral condition. To run the skewness check:

- \* Select EDIT/EDIT over grid/regions...
- \* Select ACCEPTABLE SKEWNESS from edit region window.
  - A SKEWNESS window will pop up.
  - Enter a skewness value, generally between 3.9 and 3.4.
  - Click on the PREVIEW button. Note that selection of the *accept* button will result in the deletion of all elements violating the skewness criterion. It is NOT recommended that the *accept* button be selected.

To correct isolated points and entire regions within the grid where the skewness criterion has been violated, one can locally edit nodes or elements or refine an entire region through element splitting. Mesh adjustments can also be achieved by adding, deleting, or modifying build points and retriangulating the mesh.

### 5.3.2 Local Node and Element Editing

The alteration of elements whose skewness values have been flagged for exceeding the prescribed threshold can be accomplished through a number of element-wise and nodal edits. Several approaches can be taken including moving nodes, swapping the shared line between elements, or increasing the number of elements by splitting the elements into three or four new elements. Also note that improvements made to the edit grid through point- and element-wise edits are largely experimental, so it is highly recommended that the user save frequently as edit grid improvements are made. Reloading the latest saved version of an edit grid can quickly undo unexpected or detrimental grid modifications.

To move a node:

- \* Select EDIT/EDIT TRIANGLES... from menu bar.
- \* Select *move node* from EDIT NODES/ELEMENTS window.
  - Click with left mouse button on the node to move and then click on its new location. The prompts for each step are written at the bottom border of the main *ACE/gredit* window.
  - Nodes can be moved as often as needed without reselecting *move node* from the EDIT NODES/ELEMENTS window. To end, hit the middle or right mouse button.

To swap a side shared by two triangular elements:

- \* Select EDIT/EDIT TRIANGLES... from menu bar.
- \* Select *swap lines* from EDIT NODES/ELEMENTS window.
  - Click with left mouse button on two adjacent elements. The nodes that define the new line are the two nodes of the elements not used to define the previous line. Prompts are written at the bottom border of the main *ACE/gredit* window.
  - *Swap lines* can be used to undo an unproductive swap by selecting the two new elements that were created from the first line swap.
  - Line swapping can be repeated as often as needed without reselecting *swap lines* from the EDIT NODES/ELEMENTS window.
  - To end, hit the middle or right mouse button.

To add a node:

- \* Select EDIT/EDIT TRIANGLES... from menu panel.
- \* Select *add node* from EDIT NODES/ELEMENTS window.
  - Click with left mouse button on the new node location. The prompts for each step are written at the bottom border of the main *ACE/gredit* window.
  - Additional nodes can be added by clicking with the left mouse button until the middle or right mouse button is clicked.
- \* To dismiss the EDIT NODES/ELEMENTS window, click on *done*.
- \* To see the new nodes, click on the DISPLAY NODES icon and then *DRAW* on the main panel.

To add an element:

New elements can be created from new nodes (see the adding new nodes section) or from existing nodes that remain following an element or node deletion.

- \* Select EDIT/EDIT TRIANGLES... from menu panel.
- \* Select *add element* from EDIT NODES/ELEMENTS window.
  - Click, in counterclockwise fashion, the three nodes that will serve as the vertices of the new element. The prompts for each step are written at the bottom border of the main *ACE/gredit* window.
  - More than one element can be made in a single session and new elements are displayed as they are made.
  - To end, click with middle or right mouse button.
- \* To dismiss the EDIT NODES/ELEMENTS window, click on *done*.

To delete an element:

- \* Select EDIT/EDIT TRIANGLES... from menu panel.
- \* Select *delete element* from EDIT NODES/ELEMENTS window. Note that the cursor is now represented as a skull and crossbones; the bottom border of the main window displays instructions on deletion.
  - Click on the elements to be deleted with left mouse button. Elements selected for deletion are identified by a black square that appears at the element center.
  - Deletion does not occur until it is confirmed by clicking the middle button and can be cancelled by clicking the right mouse button.

To view the effects of local node and element modifications on the edit grid, select the *edit grid fill* icon to the left on the main panel. Elements are shaded gray and islands or portions of the domain not represented by an element are indicated in white. This tool can be used to identify irregularities in the local node and element edits.

### 5.3.3 Mesh Refinement

Element skewness problems in a region arise due to sharp transitions between finely and coarsely gridded areas. In these regions, it is common for six elements to all have a shared vertex.

The smoothing of sharp changes in resolution can be achieved by increasing resolution within a transition region in two ways: splitting an element into three or four elements or adding elements in the transition area.

To refine a region through element splitting:

First, select the region of elements to be split:

- \* Select REGIONS/CREATE REGION from the menu bar.
  - Click with the left mouse button to create the points that define this region.
  - Click with middle or right mouse button to end the creation phase; the created region will be enclosed in a thick black line.

Then, split elements in the region:

- \* Select EDIT/EDIT OVER GRID/REGION from the menu bar.
- \* Select *split elements...* from EDIT REGION window.
- \* Select a preference for the type of element split from the list next to *split elements* in the SPLIT IN REGION window.
  - *Split into 3* is preferable for transition regions. *Split into 4* will result in three-node element sides where the defined region meets the coarser mesh resolution.

An initial sweep to adjust element skewness or refine a mesh may best be achieved by adding build points in problem areas and retriangulating the mesh.

To refine or modify resolution within a region:

- \* Select BUILD/LOAD GRID TO BUILD POINTS from menu panel. To have an unobstructed view of the build points, the only icon selected should be that under *BlidP* on the left panel.

To add build points:

- \* Select BUILD/PLACE BUILD POINTS from the menu panel.
  - Click with the left mouse button to place new build points.
  - Click with middle or right mouse button to end.

To move build points:

- \* Select BUILD/MOVE BUILD POINTS from the menu panel.
  - Click with the left mouse button on the build point to be moved and then click the new location.
  - To end, hit the middle or right mouse button.

To remove build points:

- \* Select BUILD/DELETE BUILD POINTS from the menu panel. The cursor becomes a skull and crossbones and the bottom border of the main window displays instructions on deletion.
  - Click on the build points to be deleted with left mouse button. Upon selection, the build point center becomes black.

- Click on the middle mouse button to confirm deletion.
- Click on the right mouse button to cancel deletion.

Once the new build points are in place, the grid must be retriangulated:

- \* Select BUILD/TRIANGULATE BUILD POINTS... from menu bar.
  - Enter a minimum distance in *triangulate build points* window. Experience dictates that the minimum distance must be greater than 0.
  - Click on *apply* in *triangulate build points* window.

To smooth the elements, making them more equilateral, the nodal positions are adjusted through the *springs* utility:

- \* Select BUILD/SPRINGS from the menu bar.

More adjustments and element skewness checks will likely be required after either of these procedures.

## 5.4 Model Diagnostics

### 5.4.1 Edit Gridpoint Locator

Locate can be used to identify the position of nodes and elements, the bathymetry at a point, or specific coordinates within the edit grid. Such a capability is extremely useful when tracking model instabilities or assessing computational effects of the mesh or bathymetry on model computed fields. The tools available under LOCATE in the menu bar display information at a particular point including a node or element number, the coordinates, and the bathymetry in the bottom border of the main window. To implement these features:

- \* Select LOCATE from the menu bar.
  - *Find nearest node* displays the node number, coordinate location, and depth of the node nearest a point selected with the cursor and the left mouse button.
  - *Find element* displays the element number, associated node numbers, area, and an equivalent radius of an element selected with the cursor and the left mouse button.
  - *Find nearest element* displays the same information as *find element* but for the nearest element to the point selected with the cursor and the left mouse button.
  - *Find depth in edit grid* displays for a selected point the nearest element number, coordinate location, and the interpolated depth from the edit grid at that location.
  - *Find depth in background* provides identical information as *find depth in edit grid* but the information pertains to the background grid.
  - *Find nearest build point* displays coordinates for the build point nearest the selected point.
  - *Find depth in all* displays the depth values relative to the edit and background grids for the selected point.
  - *Goto node/element...* positions the cursor on the node or element specified by the user in the *goto node/element* window.
  - *Goto X,Y...* positions the cursor on the x, y coordinate position entered by the user in the *goto X, Y* window.

### 5.4.2 Graphical Display of Mesh Fields

*ACE/gredit* also acts as a powerful tool for viewing the finite element mesh itself, its boundary, and the bathymetry. Other fields computed over the mesh or pertaining to the mesh may be displayed as well using *ACE/gredit*. For example, one may wish to contour mesh spacing or model computed elevations. Any scalar quantity may be contoured by substituting nodal values in place of the bathymetry values in the grid file. A useful zoom in/out capability, as well as color bar options, also exist for the display window. These graphical display features, coupled with the edit gridpoint locator, form a very powerful analysis tool for finite element meshes and model results.

To view a contoured field over the mesh:

- \* Select DISPLAY/ISOLINES OF BATHYMETRY (EDIT GRID).

The DISPLAY ISOLINES icon on the left panel (first icon in right column under *EdGr*) must be selected to view. Several options for viewing isolines are possible including contour lines, color fill, or both; contour intervals can be automatically determined or user specified, and the precision of the color bar contour values can be set. To invoke these options:

- \* In the *edit grid isolines* window:
  - Select type of contour from the contour box next to *draw*, either lines, filled, or both.
  - Select *auto place labels* to label bathymetric values at a user-defined number of points on a contour line specified in the box next to *label every*.
  - Select # of *isolines* ranging from 1–16.
  - *Spacing by* permits user flexibility in selection of a contour interval.
    - *Start, step* spacing allows specification of the starting value and increment of the isolines.
    - *Specified values* spacing allows each contour value to be defined separately. The values #X and #X + 1 determine the minimum and maximum values for contour interval #X.
  - Select *precision* to define the number of significant digits associated with the contours. The range is from 0–9 decimal positions.
  - Select *vertical* or *horizontal* next to *layout* for the desired legend orientation.
  - Select *legend* to enable display of the contour legend.
  - Select *place legend* to specify the position of the contour legend in the plotting window. Note the cursor becomes a cross-hair in the plotting window.
  - Click on the new location of the contour legend. Click on *accept*. Then click *DRAW* on the main panel to view.

Repositioning of the legend can be accomplished by repeating the last three steps.

To add a map scale:

- \* Select DISPLAY/MAP SCALE... from menu bar.
  - Enter the mapscale length in the box next to *mapscale legend length*.
  - Select the legend units from the list next to *mapscale legend units*.
  - Though the legend label font and mapscale legend color can be changed, it is recommended that they remain unchanged.

- Click *display mapscale legend* and *accept*.
- Click on *place* to locate the position of the mapscale legend in the plotting window.
- Click on the new location of the map legend. Click *accept*.
- Click on *draw* and the map scale will appear.

To change the color map used for color fill:

- \* Select DISPLAY/COLORS... from menu bar and a COLOR EDIT window opens.
  - Bathymetric colors are the colors number 16–31.
- \* A color can be modified by adjusting its RGB or its HSV value. As a color is edited, the number assigned to it is displayed above the RGB and HSV slide bars. The color resulting from the current value is shown in both the COLOR EDIT window and by filled isolines in the main window.
  - RGB colors are derived according to numerical values assigned to red, green, and blue. The values for red, green, and blue can separately range from 0–255. For example, RGB values of (0, 0, 255) result in the solid blue color.
  - HSV colors are derived according to numerical values assigned to hue, saturation, and value. Hue sets the color that can range from 0–360. Saturation controls the amount of white that a color contains and can range from 0.01–1.0. Value controls the portion of black that a color contains and its range is from 0.01–1.0.
- \* An alternate approach to editing the color bar is to interpolate between two specified colors:
  - Type the numbers associated with the beginning and ending colors for interpolation in the boxes next to *from color#* and *to color#*.
  - Click on the type of interpolation desired, RGB, or HSV.
  - Click *interpolate*.
  - To revoke the interpolation, click on the *undo* box.

To add text to the display window:

- \* Select DISPLAY/TEXT... from menu bar.

Any number of text properties can be user controlled including font type, font color, justification, text position reference, rotation, and text size. In addition, the depth at a user-selected point can be included along with a symbol identifying the depth point. Existing text strings may also be edited. In practice, the text editing features are often unreliable and other means of including text on a graphic are recommended.

To either print directly to a printer or save the graphic as a file:

- \* Select FILE/SAVE...
  - Select *printer* from box next to *print to* for output directly to a printer.
    - Edit *lpr* statement in window to send to correct printer.
  - Select *file* from box next to *print to* for output to a postscript file.
    - Enter the name of the postscript file.

## 6.0 SUMMARY

One of the primary advantages associated with finite element approximations is the flexibility associated with variably graded meshes. Construction of the finite element mesh with appropriate mesh resolution throughout the domain of interest remains a challenging problem in the application of finite element models.

Included in this report is a brief introduction to the finite element method and a comparison of finite element versus more common finite difference approaches that provides the reader with an initial foundation and motivation for the implementation of finite element models. Advantages of finite element approximations include the ease of localized refinement and a maximization of computational resources through unstructured, graded meshes and element-based computations. The use of a triangular element results in more realistic representations of shoreline and bathymetric complexities; boundary conditions are naturally incorporated into the finite element approximations, eliminating the need for additional forms of the discrete equations.

Both theoretical and empirical approaches for determining mesh resolution *a priori* for a particular application were presented. Drawbacks for several criterion were highlighted. Clearly, no one technique provides "the answer" to the question of what mesh resolution is adequate for a particular application. In fact, two areas of recent research, local truncation error analysis and dynamic gridding, were introduced as promising approaches for the future. The adequacy of mesh resolution is ultimately determined from a well-formulated convergence study in the area of interest.

The more theoretical discussion of mesh resolution turns to practical concerns of the mesh generation itself. Several known tools for mesh generation are described along with promising developmental software. The mesh generator, *ACE/gredit*, currently offers the most versatile and advanced features in mesh creation and modification within a user-friendly environment. Detailed descriptions of the usage of *ACE/gredit* are presented in the context of common finite element mesh applications such as mesh creation, editing, and model diagnostics. One highlight is step-by-step instructions for the automatic generation of a mesh, given coastline and bathymetry data. This procedure offers a first-generation, rapid relocatable capability for finite element models.

## 7.0 ACKNOWLEDGMENTS

This work has been funded by the Office of Naval Research through the NRL 6.2 "Coastal Simulation" project (Program element number 0602435N). Thanks are extended to A. Baptista and P. Turner at the Oregon Graduate Institute for the privilege of using the *ACE/gredit* software.

## 8.0 REFERENCES

- Becker, E. B., G. F. Carey, and J. T. Oden, *Finite Elements, An Introduction, Volume I*, (Prentice-Hall, Inc., Englewood Cliffs, NJ, 1981).
- Blain, C. A., J. J. Westerink, and R. A. Luettich, "The Influence of Grid Structure on the Response Characteristics of a Hurricane Storm Surge Model," *Int. J. Num. Methods Fluids* **26**, 369-401 (1998).

- Blain, C. A., J. J. Westerink, and R. A. Luetlich, "The Influence of Domain Size on the Response Characteristics of a Hurricane Storm Surge Model," *J. Geophys. Res.* **99**(C9), 18467–18479 (1994).
- Celia, M. A. and W. G. Gray, *Numerical Methods for Differential Equations* (Prentice-Hall, Inc., Englewood Cliffs, NJ, 1992).
- Cescotto, S. and Z. D. Wu, "A Variable-Density Mesh Generation for Planar Domains," *Commun. Appl. Numer. Methods* **5**, 473–481 (1989).
- Foreman, M. G. G., "A Two-Dimensional Dispersion Analysis of Selected Methods for Solving the Linearized Shallow Water Equations," *J. Computational Physics* **56**, 287–323 (1984).
- Gray, W. G. and D. R. Lynch, "Time Stepping Schemes for Finite Element Tidal Model Computations," *Advances in Water Resources* **1**(2), 83–95 (1977).
- Hagen, S. C. and J. J. Westerink, "Finite Element Grid Resolution Based on Second- and Fourth-Order Truncation Error Analysis," Second International Conference on Computer Modelling of Seas and Coastal Regions, Computational Mechanics, UK, 283–290, 1995.
- Hagen, S. C. and J. J. Westerink, "Utilizing an Imposed Multiple of Change in Finite Element Grid Generation," *Proceedings of the Third Asian-Pacific Conference on Computational Mechanics*, Chang-Koon Choi (ed.), 1996, pp. 1817–1822.
- Hannah, C. G. and D. G. Wright, "Depth Dependent Analytical and Numerical Solutions for Wind-Driven Flow in the Coastal Ocean," in *Quantitative Skill Assessment for Coastal Ocean Models*, Coastal and Estuarine Studies, Vol. 47, 125–152, American Geophysical Union, 1995.
- Henry, R. F. and R. A. Walters, "Geometrically Based, Automatic Generator for Irregular Triangular Networks," *Communications in Numerical Methods in Engineering* **9**, 555–566 (1993).
- Kinnmark, I. P. E., "Shallow Water Wave Equations: Formulation, Analysis and Application," C. A. Brebbia and S. A. Orzag (eds.), *Lecture Notes in Engineering*, Vol. 15, Springer-Verlag, New York, NY, 187 p., 1986.
- Lapidus, L. and G. F. Pinder, *Numerical Solution of Partial Differential Equations in Science and Engineering* (John Wiley and Sons, Inc., New York, NY, 1982).
- LeProvost, C. and P. Vincent, "Some Tests of Precision for a Finite Element Model of Ocean Tides," *J. Computational Physics* **65**, 273–291 (1986).
- Luetlich, R. A., J. J. Westerink, and N. W. Scheffner, "ADCIRC: An Advanced Three-Dimensional Circulation Model for Shelves, Coasts and Estuaries, Report 1: Theory and Methodology of ADCIRC-2DDI and ADCIRC-3DL," Technical Report DRP-92-6, U.S. Army Corps of Engineers Waterways Experiment Station, Vicksburg, MS, 137 p., 1992.
- Luetlich, R. A. and J. J. Westerink, "Continental Shelf Scale Convergence Studies with a Barotropic Tidal Model," in *Quantitative Skill Assessment for Coastal Ocean Models*, Coastal and Estuarine Studies, Vol. 47, 349–372, American Geophysical Union, 1995.

- Lynch, D. R., J. T. Ip, C. E. Naime, and F. E. Werner, "Convergence Studies of Tidally-Rectified Circulation on Georges Bank," in *Quantitative Skill Assessment for Coastal Ocean Models*, Coastal and Estuarine Studies, Vol. 47, 153-174, American Geophysical Union, 1995.
- Myers, P. G. and A. J. Weaver, "A Diagnostic Barotropic Finite-Element Ocean Circulation Model," *J. Atmos. Oceanic. Tech.* **12**, 511-526 (1995).
- Pearson, F., *Map Projections: Theory and Applications* (CRC Press, Inc., Boca Raton, FL, 1990).
- Platzman, G. W., "Some Response Characteristics of Finite Element Tidal Models," *J. Computational Physics* **40**, 36-63 (1980).
- Simpson, R. B., "A Survey of Two-Dimensional Finite Element Mesh Generation," in Proceedings 9th Manitoba Conf. Num. Math and Computing, 1979, pp. 49-124.
- Turner, P. J. and A. M. Baptista, "ACE/gredit User's Manual, Software for Semi-Automatic Generation of Two-Dimensional Finite Element Grids," SDS2 91-2, Center for Coastal and Land-Margin Research, Beaverton, OR, 1991.
- Turner, P. J. and A. M. Baptista, "ACE/gredit Tutorial, Draft Report," Center for Coastal and Land-Margin Research, Beaverton, OR, 1992.
- Walters, R. A. and R. F. Henry, "TRIGRID Users Manual," GKS Version, Tacoma, WA, 1995.
- Westerink, J. J., R. A. Luettich, and J. C. Muccino, "Modeling Tides in the Western North Atlantic Using Unstructured Grids," *J. Computational Physics* **46A**, 178-199 (1994a).
- Westerink, J. J., R. A. Luettich, and S. C. Hagen, "Meshing Requirements of Large Scale Coastal Ocean Tidal Models," *Numerical Methods in Water Resources*, A. Peters, G. Wittum, B. Herrling, and U. Meissner (eds.), (Kluwer Academic Publishers, The Netherlands, 1994b).

## Appendix A

### EXAMPLES OF SEMI-AUTOMATIC GRID GENERATION USING ACE/GREDIT

The semi-automatic grid generation feature was implemented for a single geographical region with grids created for two portions of that region. The first mesh covers the Yellow and Bohai Seas together, which are bordered to the east by Korea and to the north and west by China. The second grid is a subset of the first and encompasses only the Bohai Sea. Raw coastal outline data for this geographical area was obtained from the WVS and CIA II data bases. Bathymetry values were taken from a 5-min resolution data source covering most of the region of interest. Note, however, that this bathymetry field did not cover near coastal portions of the western Bohai Sea as defined by the WVS and CIA II shoreline data. One can see in the examples shown the importance of utilizing a bathymetry field that includes all areas within the coastal outline of the region of interest.

In applying the automatic grid generation feature of *ACE/gredit*, the dimensionless wavelength criterion was used exclusively for nodal point placement. Tables A1 and A2 provide summaries of 25 cases of grid generation in which various values of the three parameters shortest wave, grid delta, and dimensionless wavelength, are selected. Included in these tables are values of the mesh generation parameters, the number of nodes and elements in the created mesh, and the CPU times associated with each stage of mesh generation. The automatically generated mesh for each of the cases, A-Y, is depicted in Figs. A.1-A.25, respectively, for the Yellow and Bohai Seas and in Figs. A.26-A.50, respectively, for the Bohai Sea.

These examples are intended to give the reader a visual feel for products of the automatic nodal placement feature of *ACE/gredit*, as well as an appreciation for the computational time required. Computed correlations for various mesh generation parameters are also presented in Table A3 for the interested reader.

Table A1 — A Summary of Automatic Grid Generation Parameters of Meshes Created for the Yellow and Bohai Seas

| Case | Shortest Wave | Grid Delta | Min. Dim. Wavelength | # of Nodes | # of Elements | Total Time (min) | P1 (s) | P2 (s) | P3 (s) | P4 (s) | P5 (s) |
|------|---------------|------------|----------------------|------------|---------------|------------------|--------|--------|--------|--------|--------|
| a    | 12.0          | 5000       | 25                   | 997        | 1750          | 11.97            | 534    | 43     | 24     | 57     | 50     |
| b    | 12.0          | 2500       | 25                   | 1026       | 1807          | 16.52            | 539    | 176    | 26     | 61     | 189    |
| c    | 12.0          | 1250       | 25                   | 1047       | 1847          | 36.50            | 535    | 664    | 72     | 62     | 847    |
| d    | 12.0          | 10000      | 25                   | 925        | 1604          | 10.25            | 537    | 12     | 20     | 57     | 19     |
| e    | 12.0          | 20000      | 25                   | 729        | 1211          | 10.75            | 545    | 3      | 20     | 57     | 8      |
| f    | 6.0           | 5000       | 25                   | 3588       | 6590          | 23.88            | 984    | 100    | 24     | 111    | 204    |
| g    | 24.0          | 5000       | 25                   | 248        | 395           | 6.17             | 275    | 20     | 24     | 23     | 15     |
| h    | 12.0          | 5000       | 50                   | 3588       | 6590          | 24.42            | 1008   | 100    | 23     | 140    | 188    |
| i    | 12.0          | 5000       | 15                   | 363        | 601           | 7.60             | 347    | 24     | 24     | 30     | 21     |
| j    | 3.0           | 5000       | 25                   | 10376      | 19332         | 55.67            | 2214   | 234    | 24     | 333    | 464    |
| k    | 1.5           | 5000       | 25                   | 19776      | 36418         | 121.67           | 5021   | 500    | 75     | 778    | 880    |
| l    | 12.0          | 5000       | 30                   | 1409       | 2510          | 13.9             | 599    | 53     | 37     | 60     | 73     |
| m    | 12.0          | 5000       | 40                   | 2398       | 4352          | 19.17            | 809    | 78     | 23     | 99     | 130    |
| n    | 9.0           | 5000       | 25                   | 1715       | 3082          | 15.3             | 656    | 59     | 25     | 82     | 88     |
| o    | 9.0           | 5000       | 35                   | 3155       | 5768          | 22.15            | 924    | 91     | 24     | 127    | 151    |
| p    | 24.0          | 5000       | 15                   | 89         | 126           | 4.30             | 173    | 19     | 25     | 14     | 10     |
| q    | 24.0          | 5000       | 50                   | 997        | 1750          | 12.50            | 564    | 47     | 23     | 65     | 53     |
| r    | 6.0           | 5000       | 15                   | 1409       | 2510          | 13.98            | 605    | 54     | 24     | 72     | 72     |
| s    | 6.0           | 5000       | 50                   | 10376      | 19332         | 52.67            | 2113   | 233    | 24     | 596    | 420    |
| t    | 12.0          | 2500       | 15                   | 370        | 614           | 9.42             | 324    | 91     | 32     | 32     | 77     |
| u    | 12.0          | 15000      | 50                   | 1997       | 3418          | 20.99            | 992    | 12     | 20     | 142    | 44     |
| v    | 6.0           | 2500       | 25                   | 3942       | 7296          | 38.98            | 992    | 406    | 32     | 142    | 697    |
| w    | 6.0           | 15000      | 25                   | 1997       | 3418          | 20.15            | 700    | 31     | 14     | 126    | 23     |
| x    | 1.5           | 2500       | 25                   | 40399      | 77568         | 258.4            | 5739   | 2290   | 36     | 765    | 6662   |
| y    | 3.0           | 2500       | 25                   | 14142      | 26872         | 96.72            | 2160   | 944    | 34     | 329    | 23     |

P1 = Processing coastal outlines

P2 = Initializing auxiliary grid

P3 = Computing depths

P4 = Processing boundary

P5 = Processing interior

Table A2 — A Summary of Automatic Grid Generation Parameters of Meshes Created for the Bohai Sea

| Case | Shortest Wave | Grid Delta | Min. Dim. Wavelength | # of Nodes | # of Elements | Total Time (min) | P1 (s) | P2 (s) | P3 (s) | P4 (s) | P5 (s) |
|------|---------------|------------|----------------------|------------|---------------|------------------|--------|--------|--------|--------|--------|
| a    | 12.0          | 5000       | 25                   | 281        | 484           | 5.06             | 218    | 17     | 22     | 22     | 15     |
| b    | 12.0          | 2500       | 25                   | 297        | 515           | 6.67             | 218    | 60     | 34     | 26     | 55     |
| c    | 12.0          | 1250       | 25                   | 300        | 521           | 13.83            | 220    | 228    | 65     | 32     | 279    |
| d    | 12.0          | 10000      | 25                   | 257        | 437           | 4.75             | 220    | 5      | 20     | 23     | 5      |
| e    | 12.0          | 20000      | 25                   | 181        | 287           | 4.72             | 224    | 3      | 17     | 23     | 1      |
| f    | 6.0           | 5000       | 25                   | 991        | 1809          | 8.92             | 370    | 32     | 24     | 52     | 46     |
| g    | 24.0          | 5000       | 25                   | 69         | 104           | 2.82             | 98     | 9      | 23     | 4      | 5      |
| h    | 12.0          | 5000       | 50                   | 991        | 1809          | 8.83             | 370    | 33     | 24     | 51     | 100    |
| i    | 12.0          | 5000       | 15                   | 104        | 164           | 3.38             | 140    | 11     | 24     | 13     | 5      |
| j    | 3.0           | 5000       | 25                   | 2534       | 4680          | 18.20            | 767    | 72     | 22     | 117    | 102    |
| k    | 1.5           | 5000       | 25                   | 3942       | 7055          | 36.50            | 1545   | 149    | 23     | 254    | 148    |
| l    | 12.0          | 5000       | 30                   | 398        | 693           | 5.75             | 247    | 20     | 23     | 29     | 19     |
| m    | 12.0          | 5000       | 40                   | 669        | 1199          | 7.17             | 291    | 26     | 24     | 51     | 21     |
| n    | 9.0           | 5000       | 25                   | 480        | 852           | 6.17             | 261    | 23     | 23     | 32     | 22     |
| o    | 9.0           | 5000       | 35                   | 879        | 1597          | 8.30             | 346    | 31     | 23     | 47     | 93     |
| p    | 24.0          | 5000       | 15                   | 22         | 25            | 1.78             | 59     | 7      | 23     | 3      | 4      |
| q    | 24.0          | 5000       | 50                   | 281        | 484           | 5.10             | 218    | 17     | 23     | 23     | 15     |
| r    | 6.0           | 5000       | 15                   | 398        | 693           | 5.70             | 243    | 20     | 23     | 28     | 20     |
| s    | 6.0           | 5000       | 50                   | 2534       | 4680          | 19.03            | 780    | 71     | 23     | 153    | 75     |
| t    | 12.0          | 2500       | 15                   | 105        | 166           | 4.35             | 144    | 36     | 33     | 13     | 24     |
| u    | 12.0          | 15000      | 50                   | 472        | 777           | 7.58             | 366    | 5      | 20     | 49     | 5      |
| v    | 6.0           | 2500       | 25                   | 1125       | 2069          | 13.92            | 367    | 124    | 33     | 57     | 185    |
| w    | 6.0           | 15000      | 25                   | 472        | 777           | 8.13             | 402    | 5      | 19     | 51     | 6      |
| x    | 1.5           | 2500       | 25                   | 9819       | 18786         | 65.23            | 1587   | 591    | 33     | 258    | 1435   |
| y    | 3.0           | 2500       | 25                   | 3912       | 7423          | 30.07            | 756    | 280    | 32     | 121    | 601    |

P1 = Processing coastal outlines

P2 = Initializing auxiliary grid

P3 = Computing depths

P4 = Processing boundary

P5 = Processing interior

Table A3 — Correlations Between Grid Generation Parameters for the Yellow and Bohai Seas and the Bohai Sea Meshes for all 26 Cases on (a) a Cartesian Scale, (b) a Logarithmic Scale, and (c) an Inverse Scale

|     |                      |                    |                     |                    |                     |                      |
|-----|----------------------|--------------------|---------------------|--------------------|---------------------|----------------------|
| (a) | Cartesian Scale      | Time               | # of Nodes          | Shortest Wave      | Grid Delta          | Min. Dim. Wavelength |
|     | Time                 |                    | 0.8936<br>0.8883    | -0.6172<br>-0.5704 | -0.2626<br>-0.2505  | -0.0017<br>-0.0235   |
|     | # of Nodes           | 0.8936<br>0.8883   |                     | -0.5852<br>-0.5851 | -0.2488<br>-0.2327  | 0.0144<br>0.0080     |
|     | Shortest Wave        | -0.6172<br>-0.5704 | -0.5851<br>-0.5852  |                    | 0.06083<br>0.06083  | 0.1201<br>0.1201     |
|     | Grid Delta           | -0.2626<br>-0.2505 | -0.2488<br>-0.2327  | 0.06083<br>0.06083 |                     | 0.1611<br>0.1611     |
|     | Min. Dim. Wavelength | -0.0017<br>-0.0235 | 0.0144<br>0.0080    | 0.1201<br>0.1201   | 0.1611<br>0.1611    |                      |
| (b) | Logarithmic Scale    | Time               | # of Nodes          | Shortest Wave      | Grid Delta          | Min. Dim. Wavelength |
|     | Time                 |                    | 0.9458<br>0.9502    | -0.8883<br>-0.8884 | -0.3353<br>-0.334   | -0.2575<br>-0.2603   |
|     | # of Nodes           | 0.9458<br>0.9502   |                     | -0.8643<br>-0.8768 | -0.1895<br>-0.1588  | 0.3875<br>0.385      |
|     | Shortest Wave        | -0.8883<br>-0.8884 | -0.8643<br>-0.8768  |                    | 0.1737<br>0.1737    | 0.07855<br>0.07855   |
|     | Grid Delta           | -0.3353<br>-0.334  | -0.1895<br>-0.1588  | 0.1737<br>0.1737   |                     | 0.2132<br>0.2132     |
|     | Min. Dim. Wavelength | -0.2575<br>-0.2603 | 0.3875<br>0.385     | 0.07855<br>0.07855 | 0.2132<br>0.2132    |                      |
| (c) | Inverse Scale        | Time               | # of Nodes          | Shortest Wave      | Grid Delta          | Min. Dim. Wavelength |
|     | Time                 |                    | 0.9831<br>0.9908    | -0.5621<br>-0.5561 | -0.2644<br>-0.2574  | 0.5035<br>0.5242     |
|     | # of Nodes           | 0.9831<br>0.9908   |                     | -0.3057<br>-0.3224 | -0.06393<br>-0.0533 | 0.5222<br>0.5384     |
|     | Shortest Wave        | -0.5621<br>-0.5561 | -0.3057<br>-0.3224  |                    | 0.1347<br>0.1347    | 0.0083<br>0.0083     |
|     | Grid Delta           | -0.2644<br>-0.2574 | -0.06393<br>-0.0533 | 0.1347<br>0.1347   |                     | 0.1561<br>0.1561     |
|     | Min. Dim. Wavelength | 0.5035<br>0.5242   | 0.5222<br>0.5384    | 0.0083<br>0.0083   | 0.1561<br>0.1561    |                      |

NOTE: The first value in each cell is the correlation from the Bohai grid cases; the second value is the correlation from the Yellow and Bohai Seas grid cases.

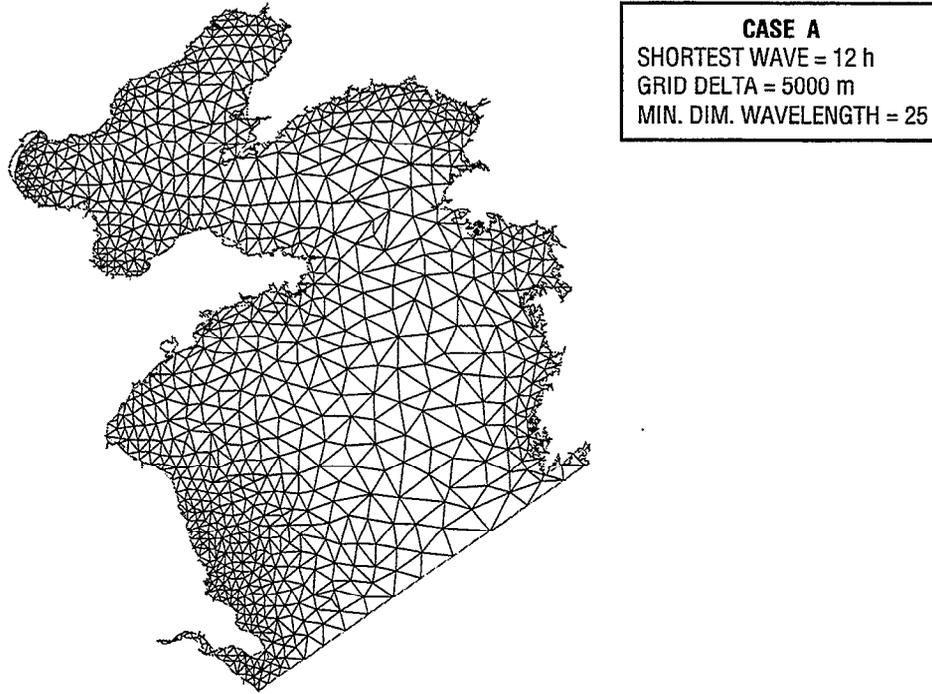


Fig. A.1 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case A

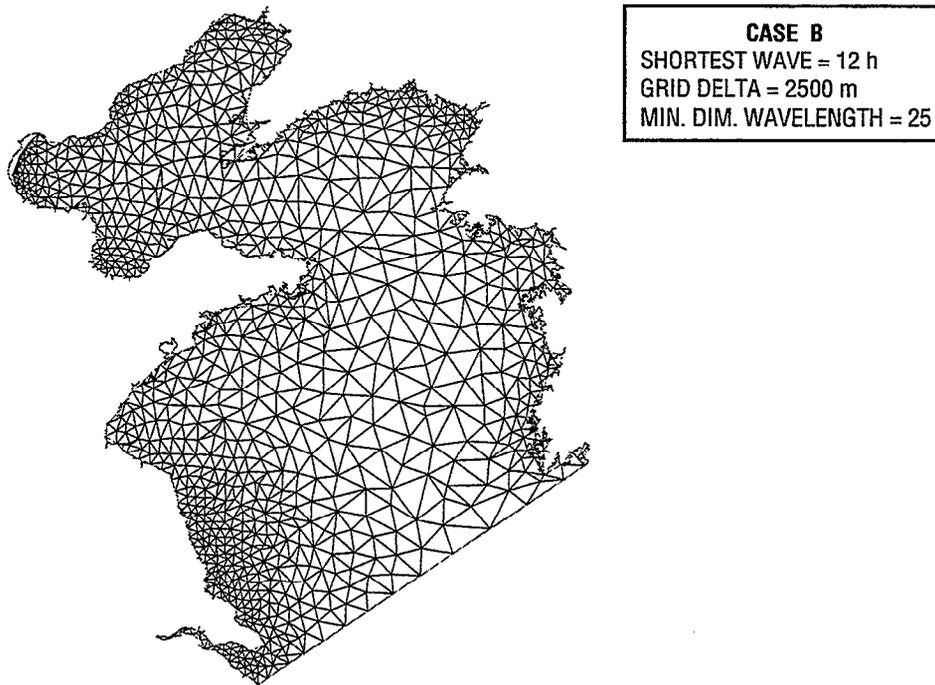


Fig. A.2 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case B

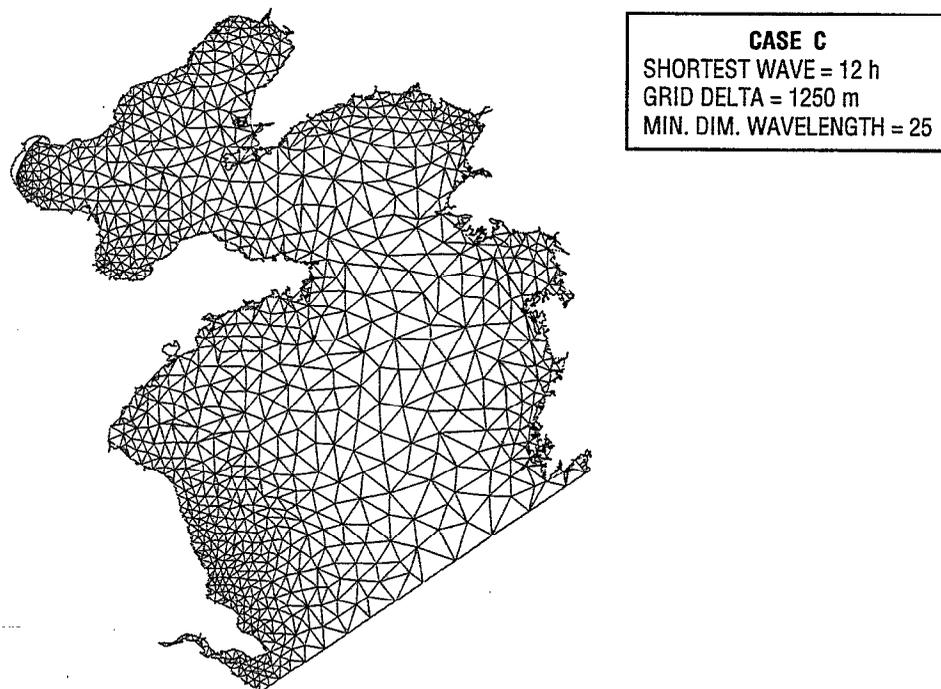


Fig. A.3 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case C

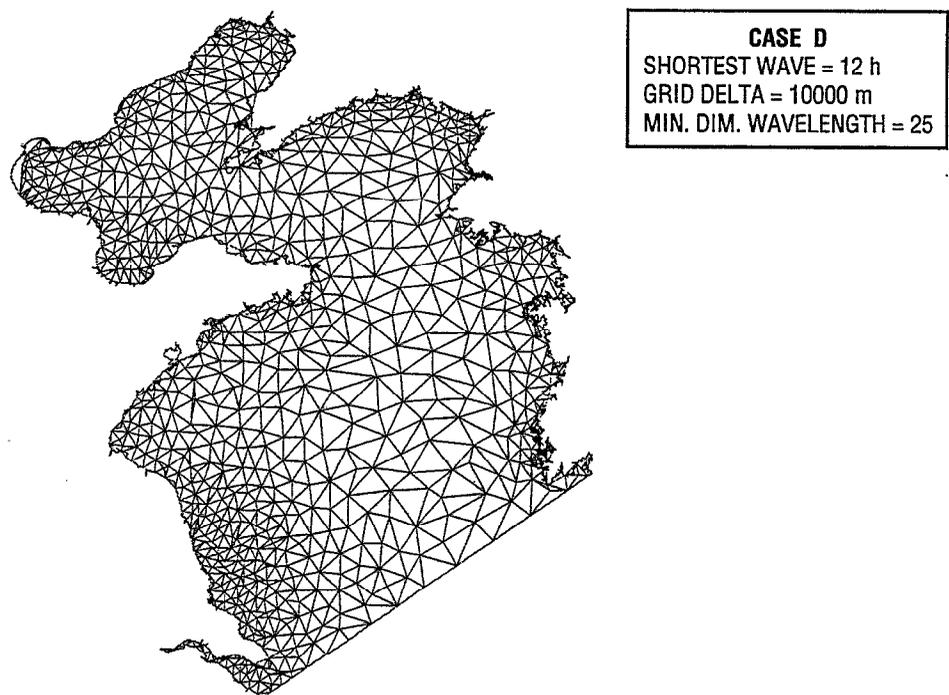
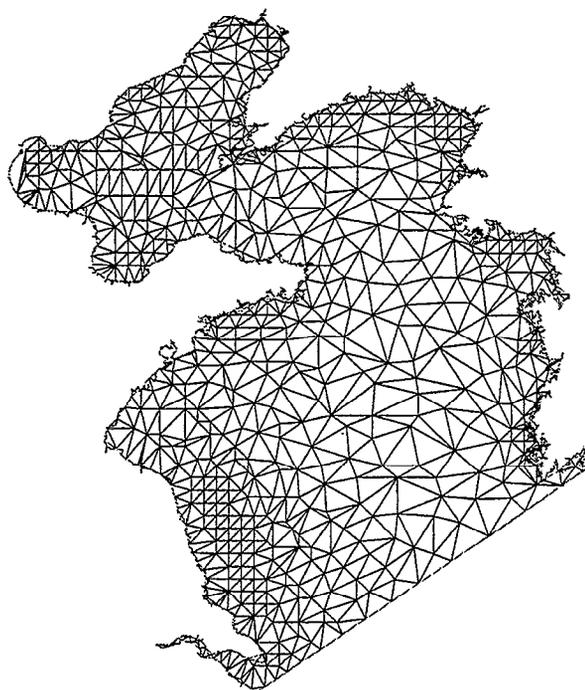
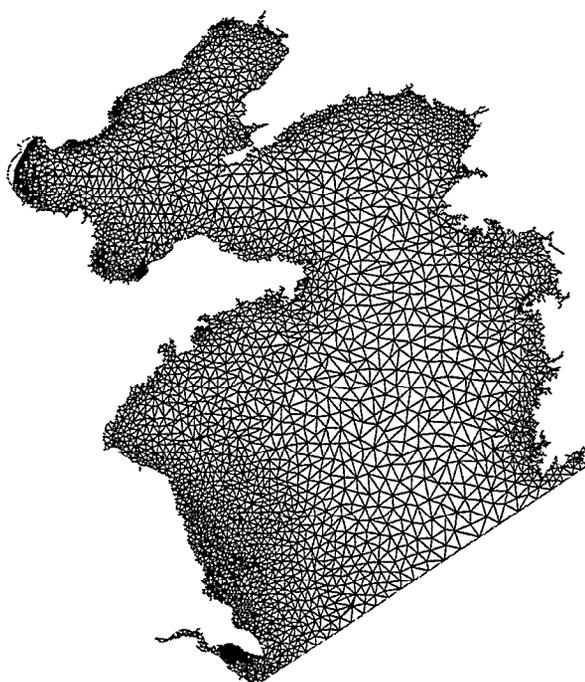


Fig. A.4 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case D



**CASE E**  
SHORTEST WAVE = 12 h  
GRID DELTA = 20000 m  
MIN. DIM. WAVELENGTH = 25

Fig. A.5 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case E



**CASE F**  
SHORTEST WAVE = 6 h  
GRID DELTA = 5000 m  
MIN. DIM. WAVELENGTH = 25

Fig. A.6 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case F

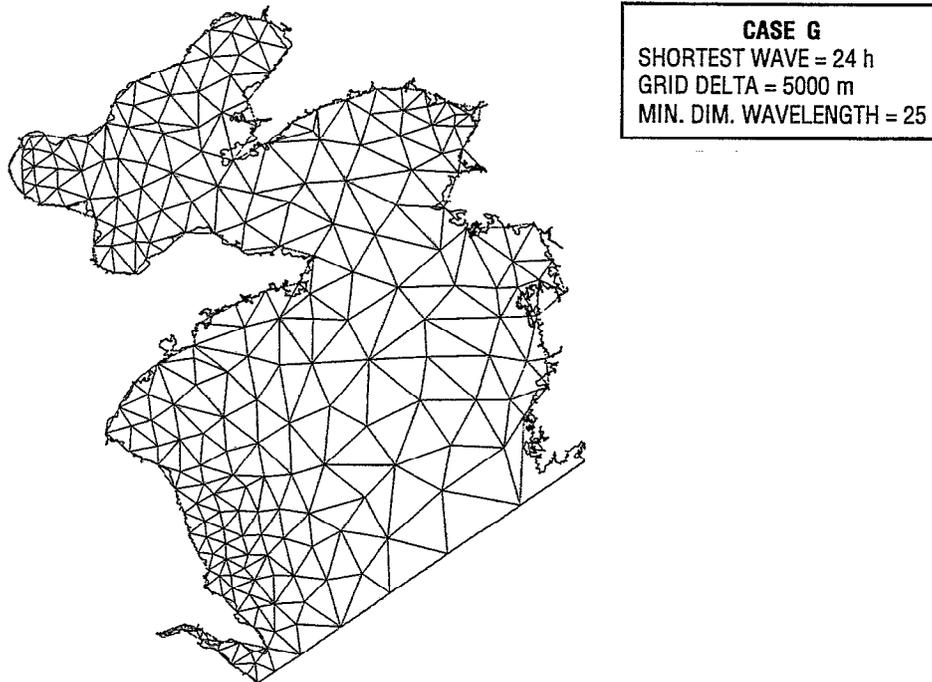


Fig. A.7 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case G

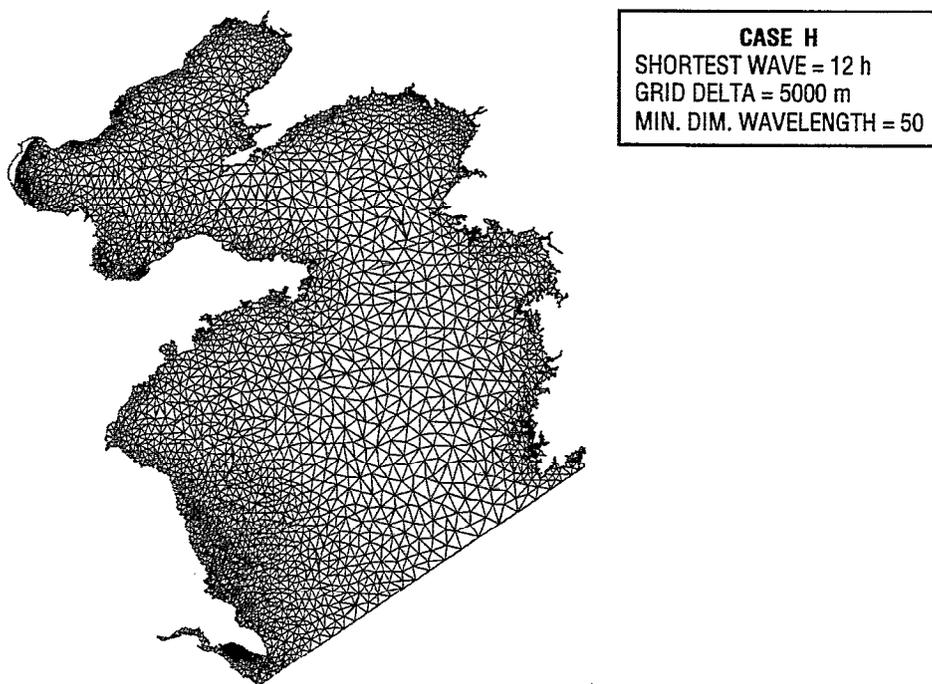


Fig. A.8 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case H

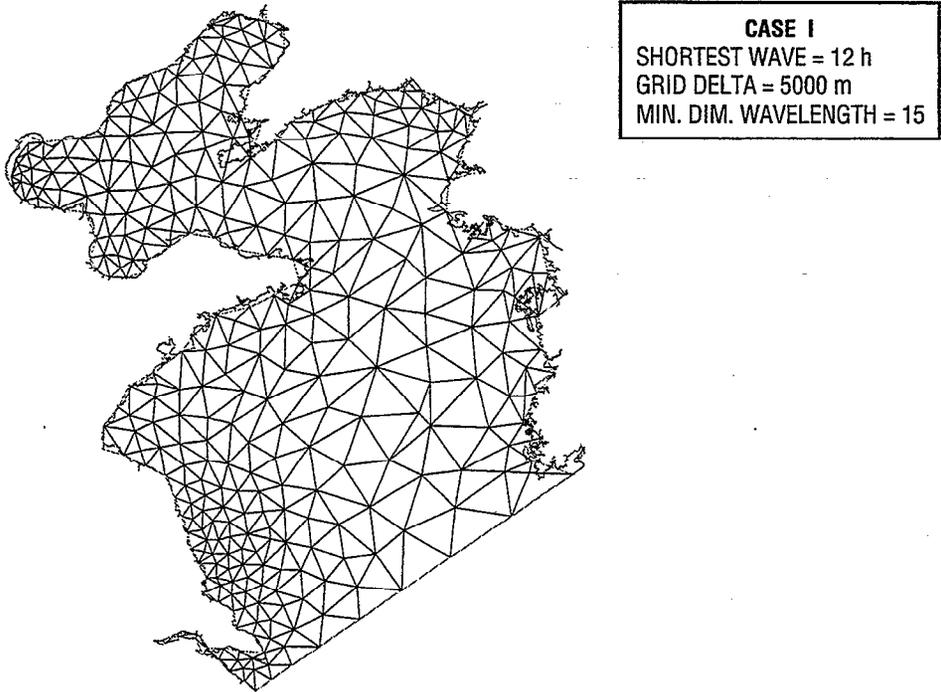


Fig. A.9 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case I

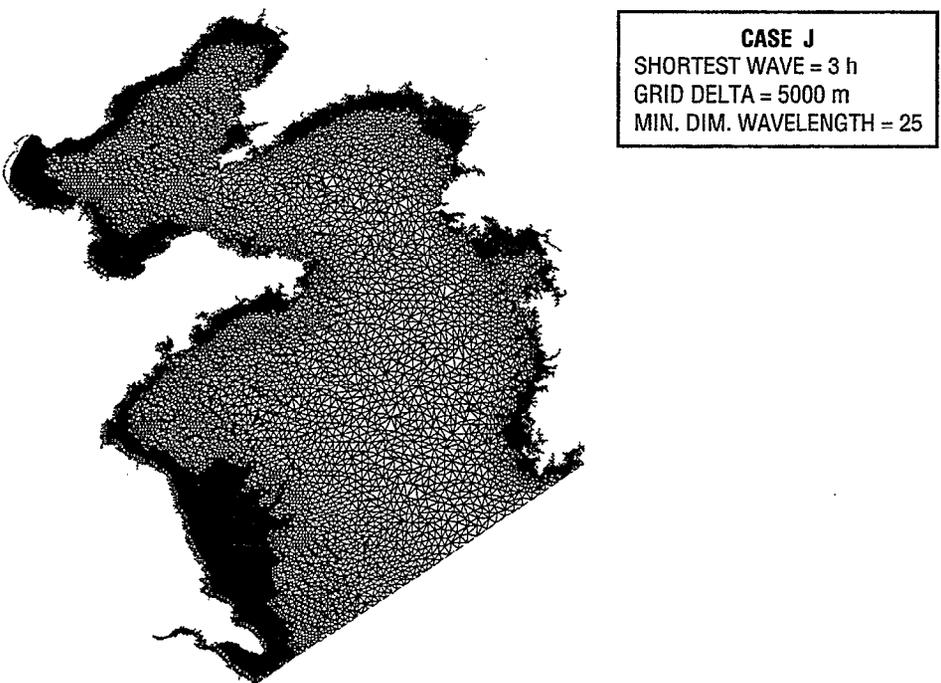


Fig. A.10 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case J

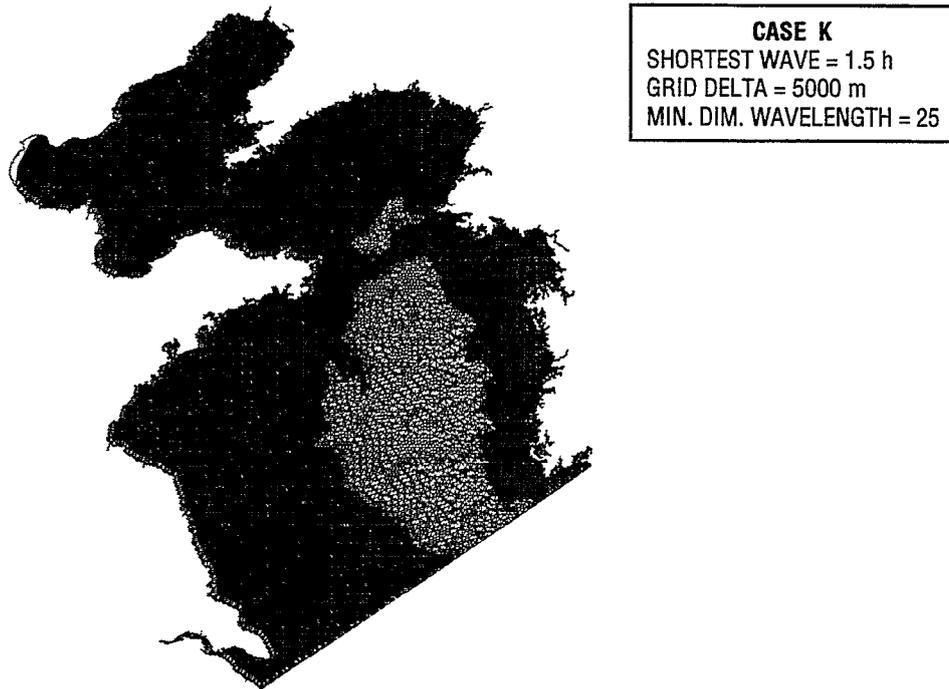


Fig. A.11 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case K

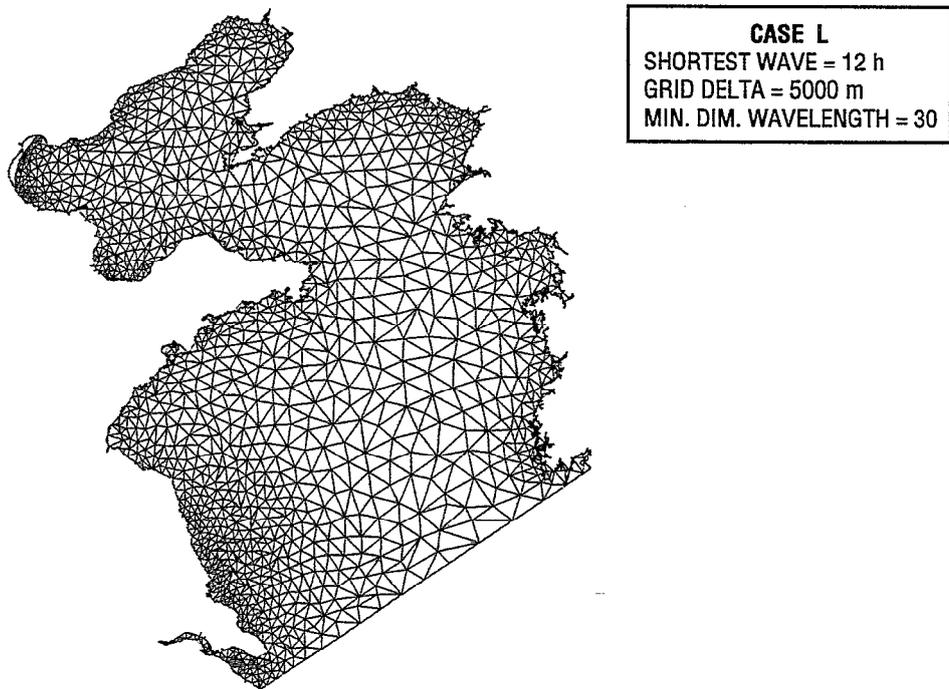


Fig. A.12 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case L

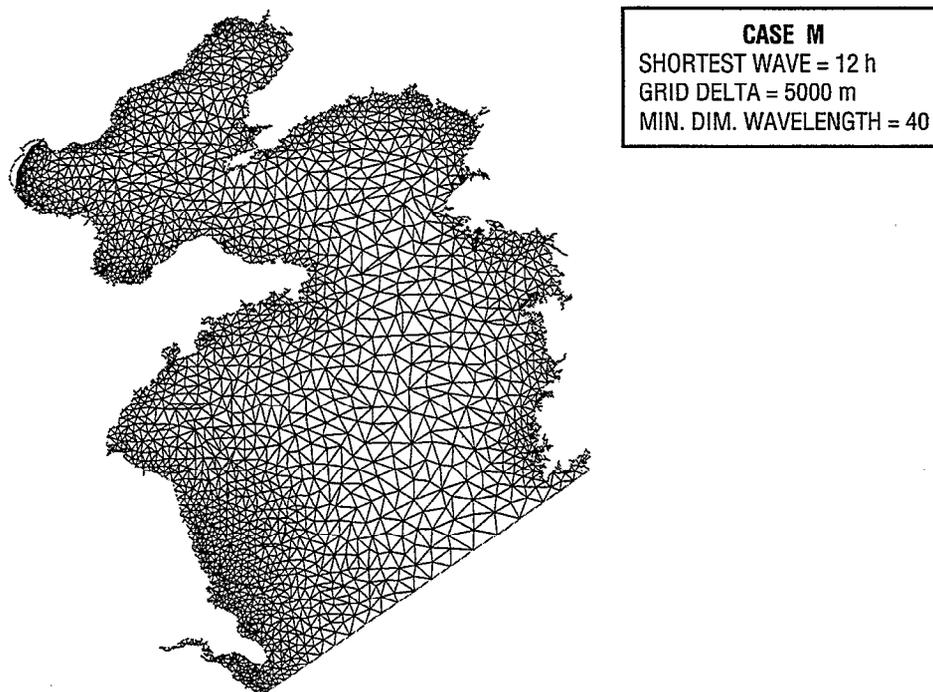


Fig. A.13 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case M

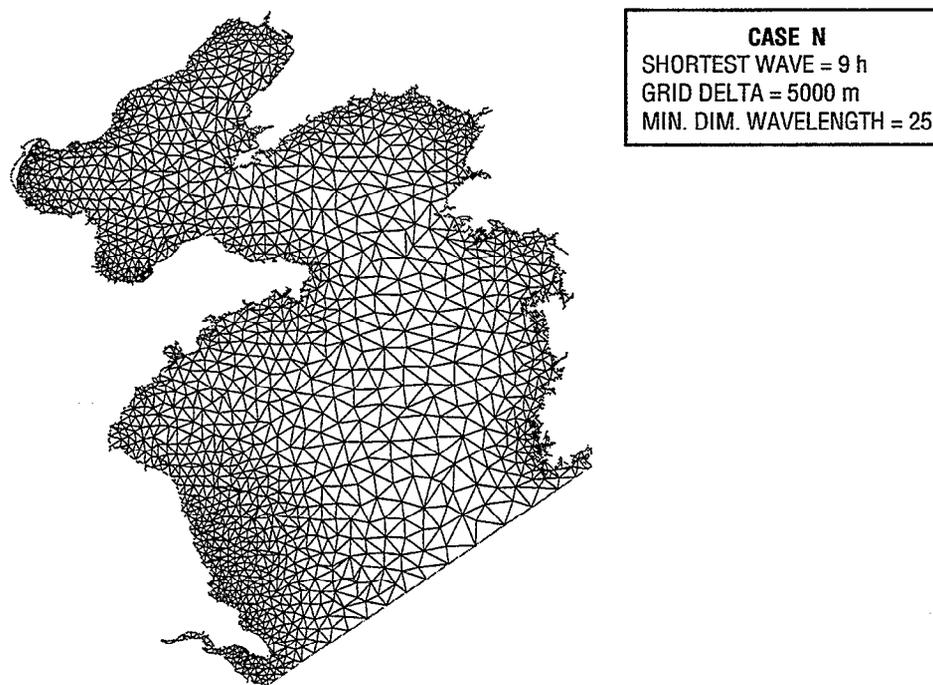


Fig. A.14 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case N

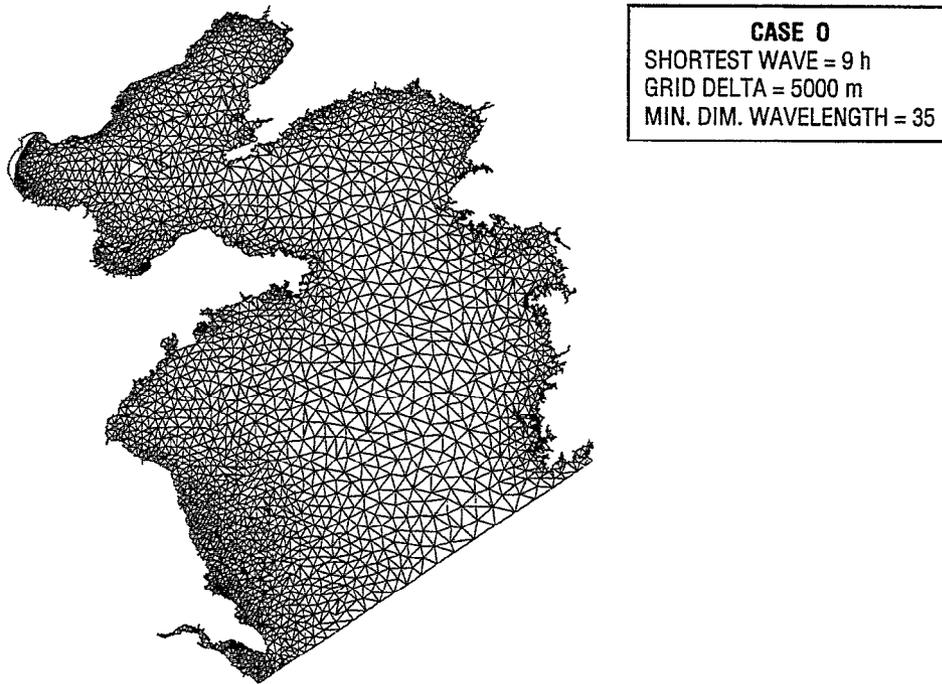


Fig. A.15 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case O

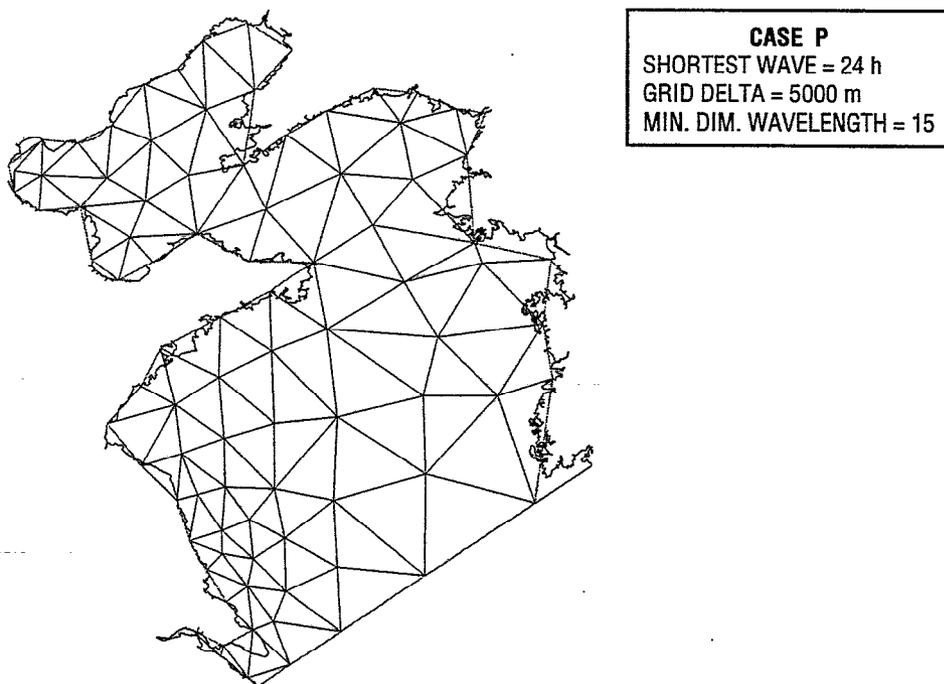


Fig. A.16 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case P

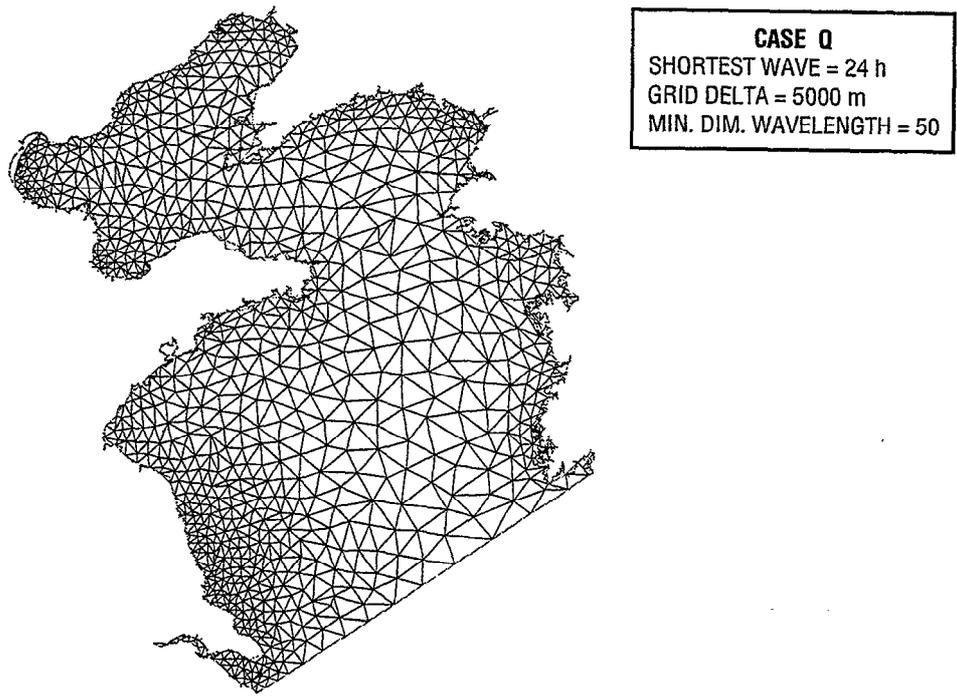


Fig. A.17 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case Q

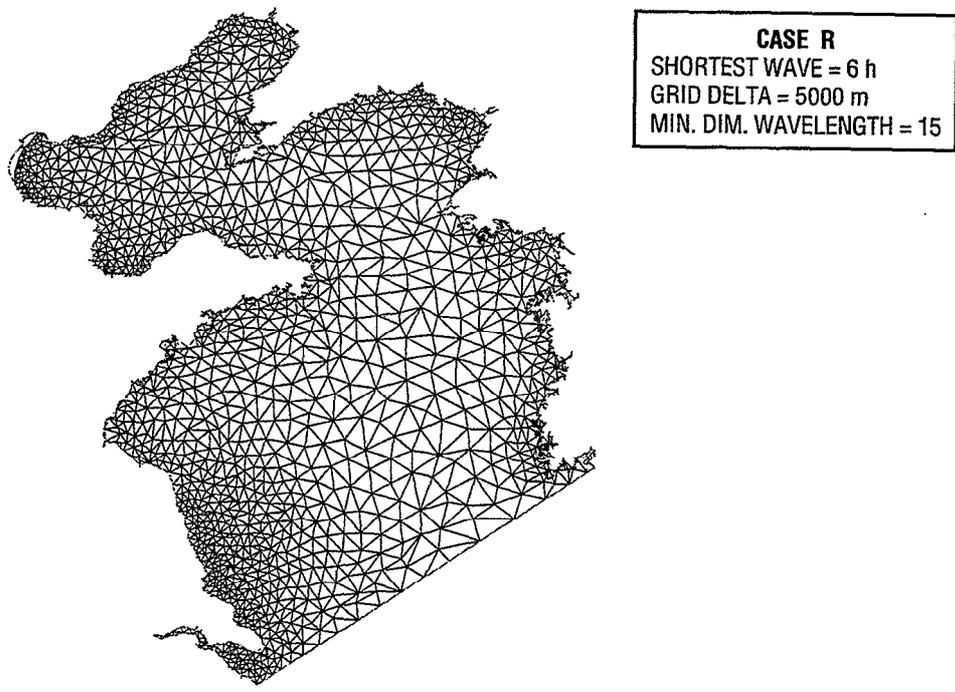


Fig. A.18 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case R

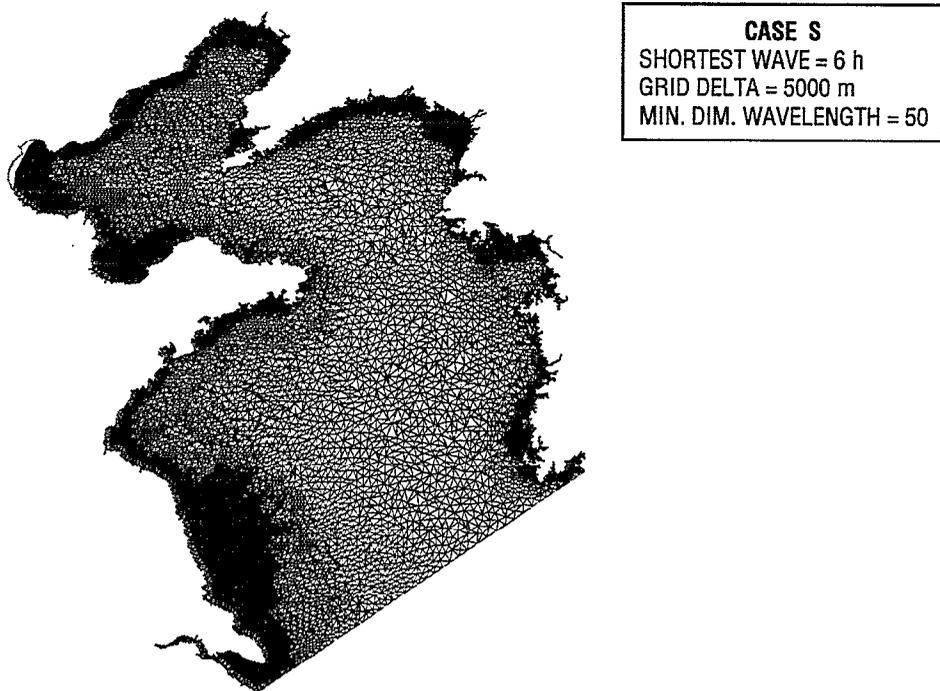


Fig. A.19 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case S

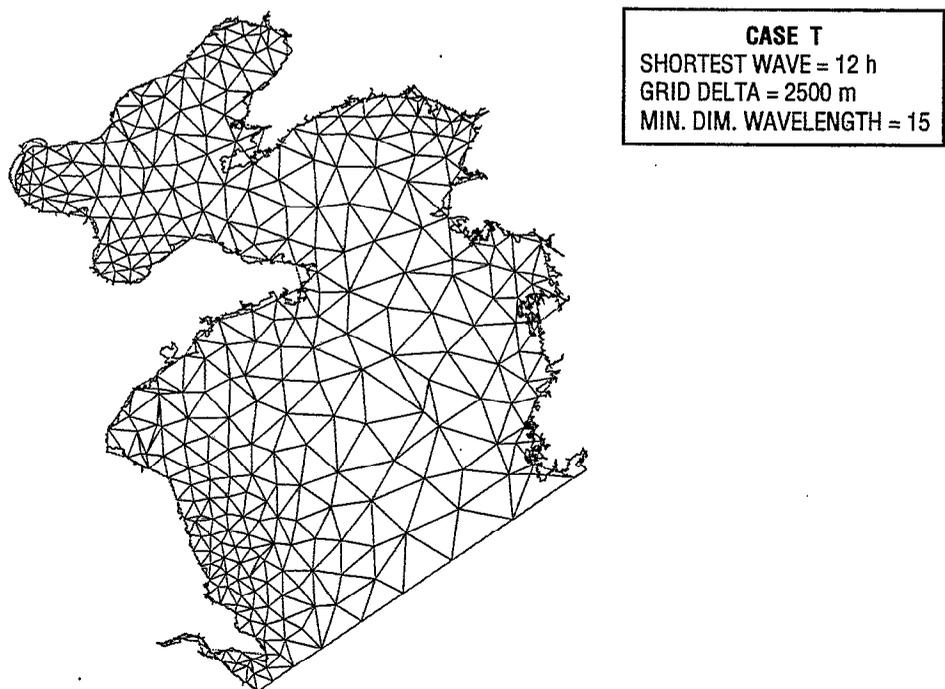


Fig. A.20 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case T

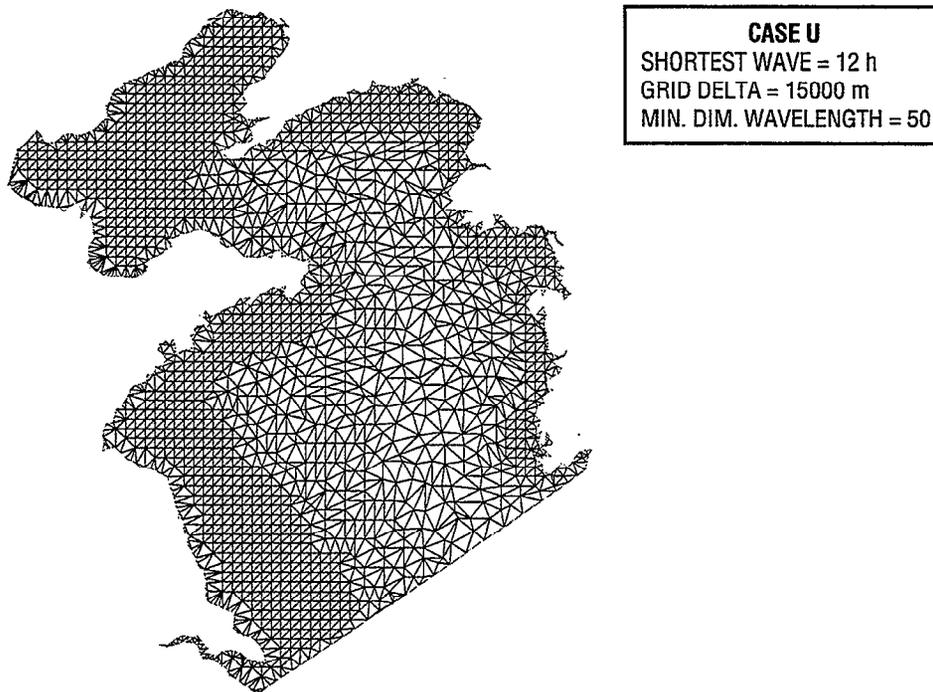


Fig. A.21 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case U

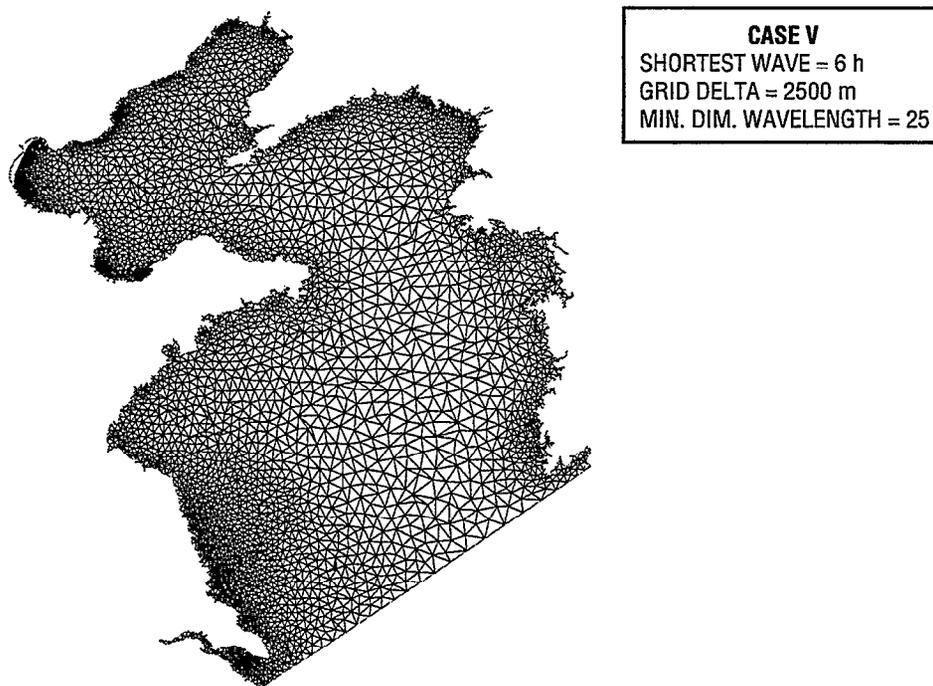


Fig. A.22 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case V

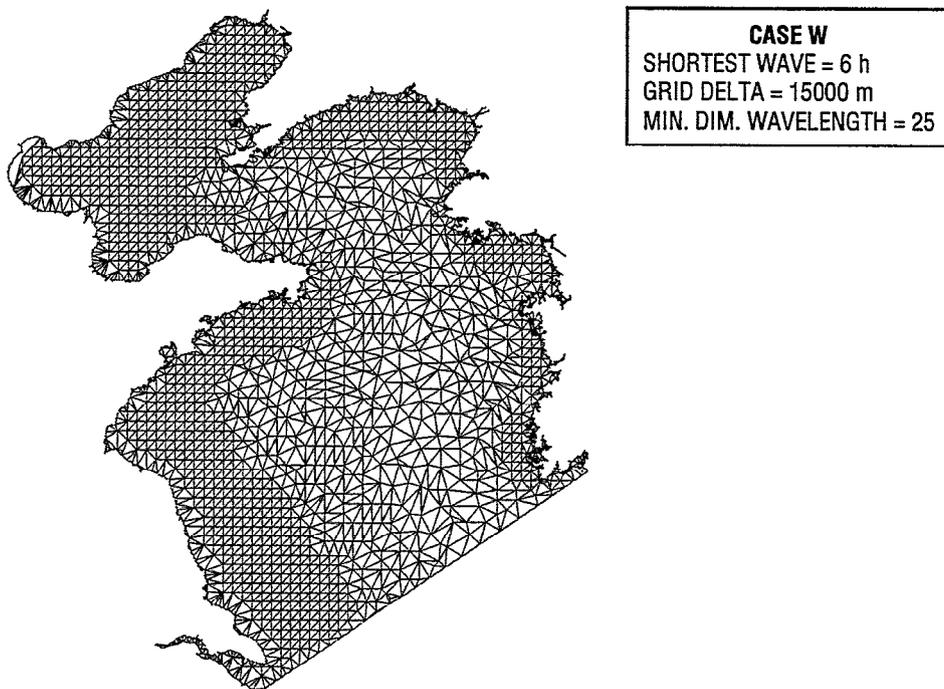


Fig. A.23 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case W

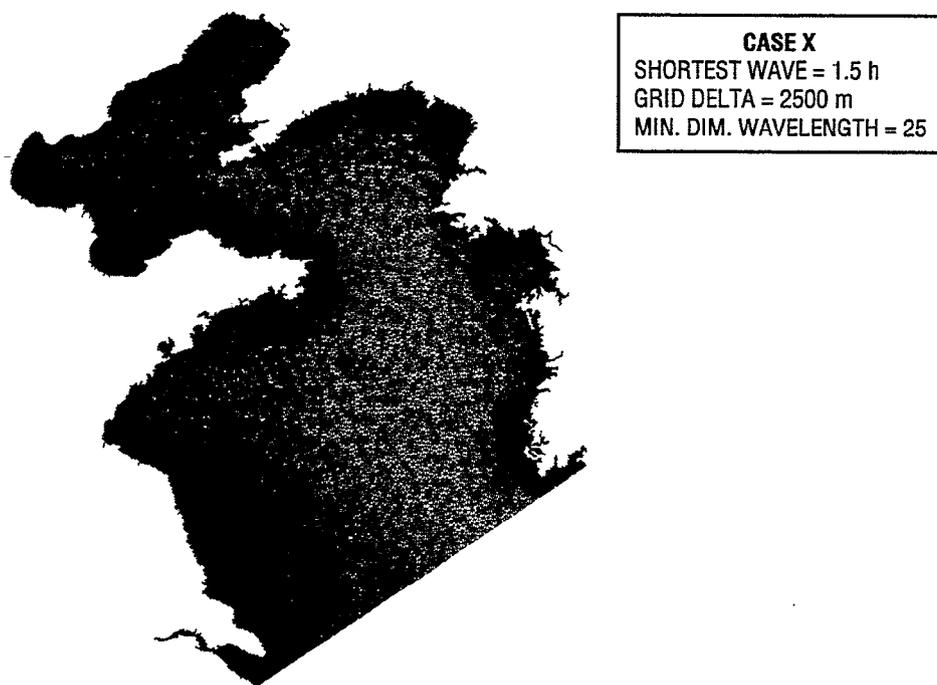


Fig. A.24 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case X

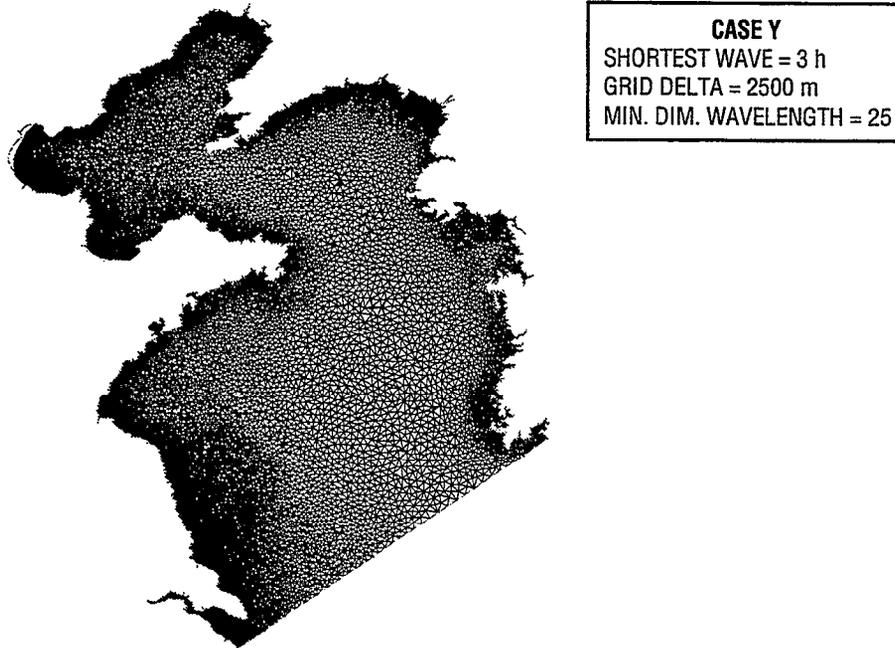


Fig. A.25 — An automatically generated finite element mesh for the Yellow and Bohai Seas, case Y

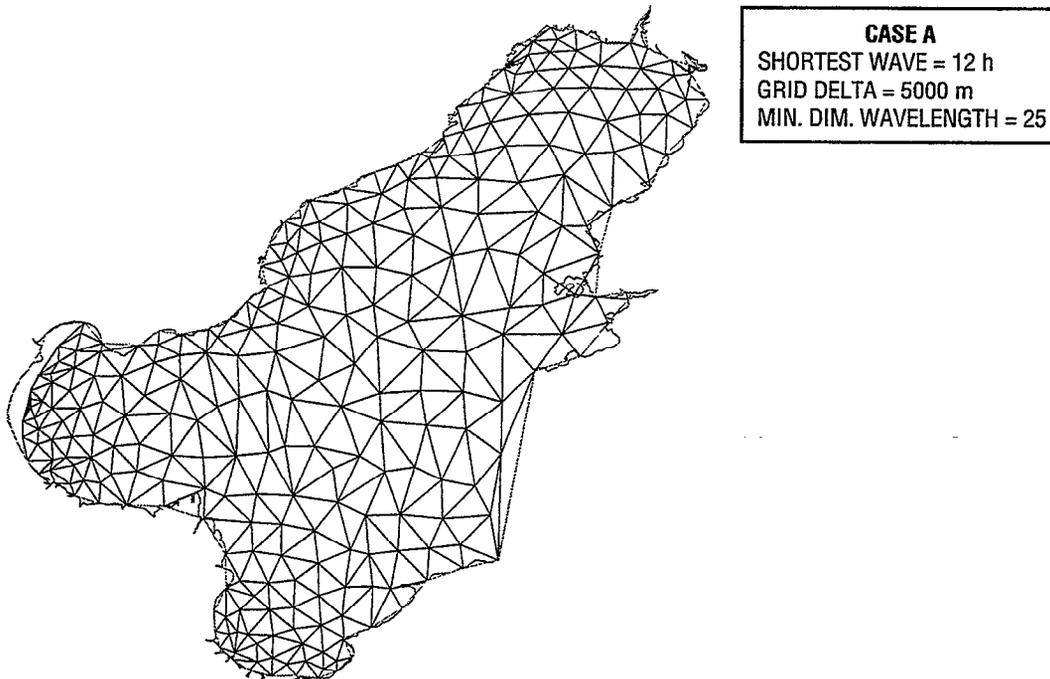


Fig. A.26 — An automatically generated finite element mesh for the Bohai Sea, case A

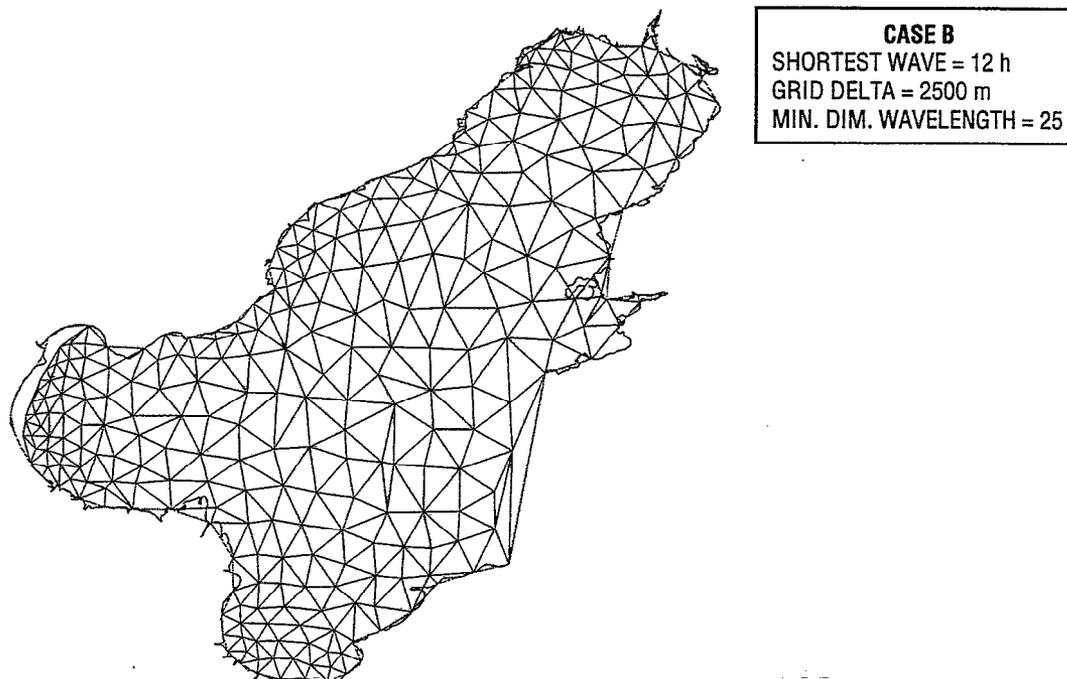


Fig. A.27 — An automatically generated finite element mesh for the Bohai Sea, case B

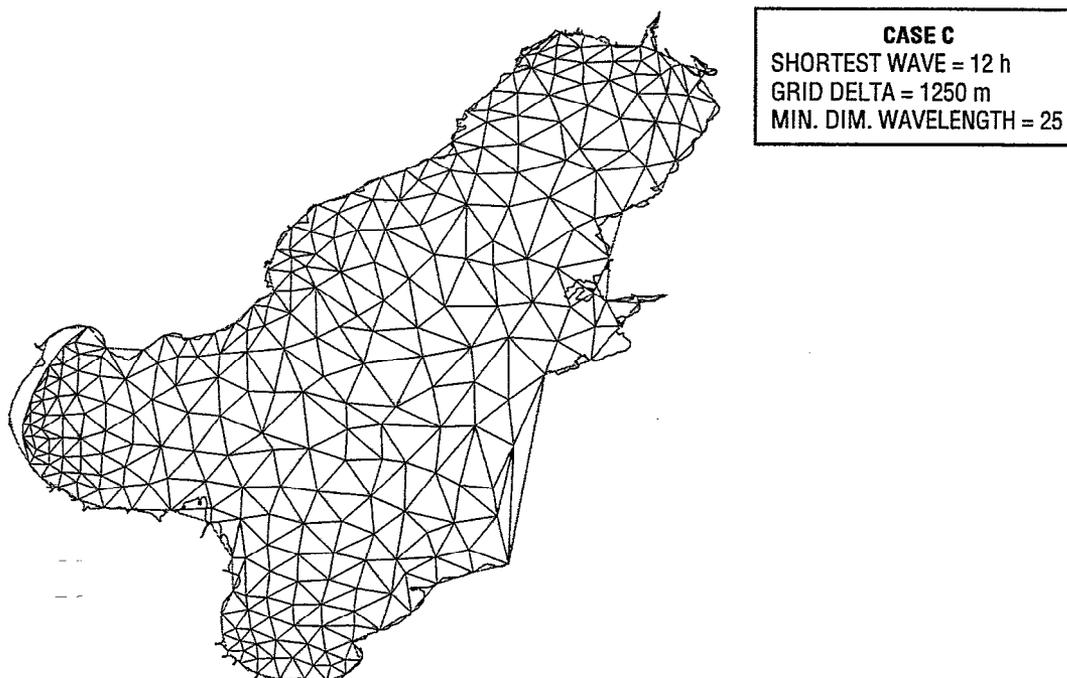


Fig. A.28 — An automatically generated finite element mesh for the Bohai Sea, case C

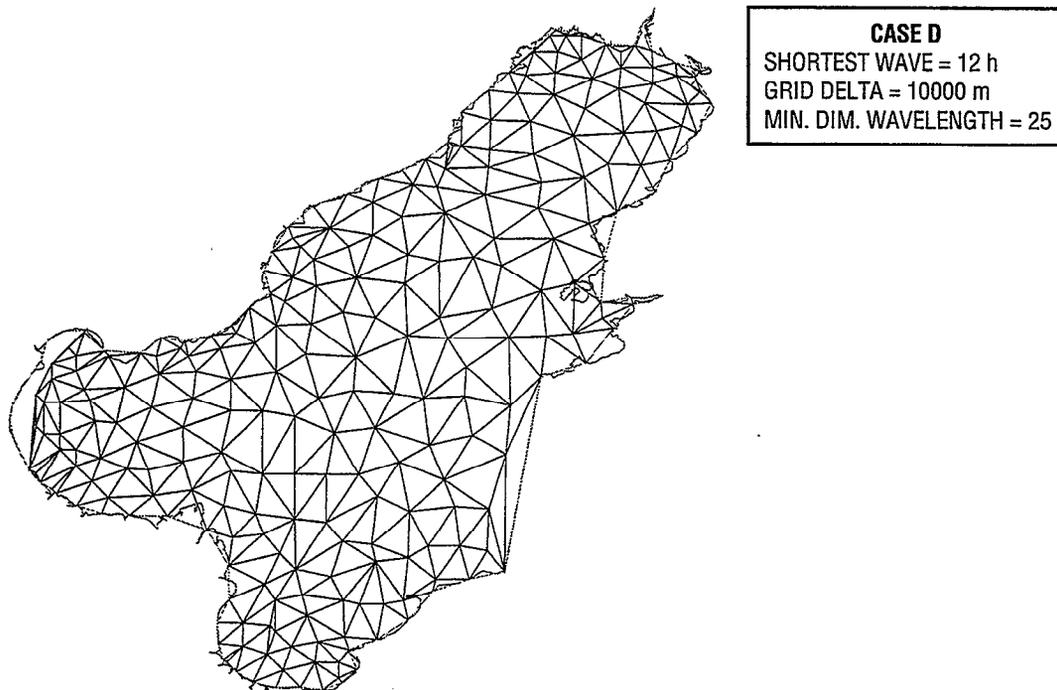


Fig. A.29 — An automatically generated finite element mesh for the Bohai Sea, case D

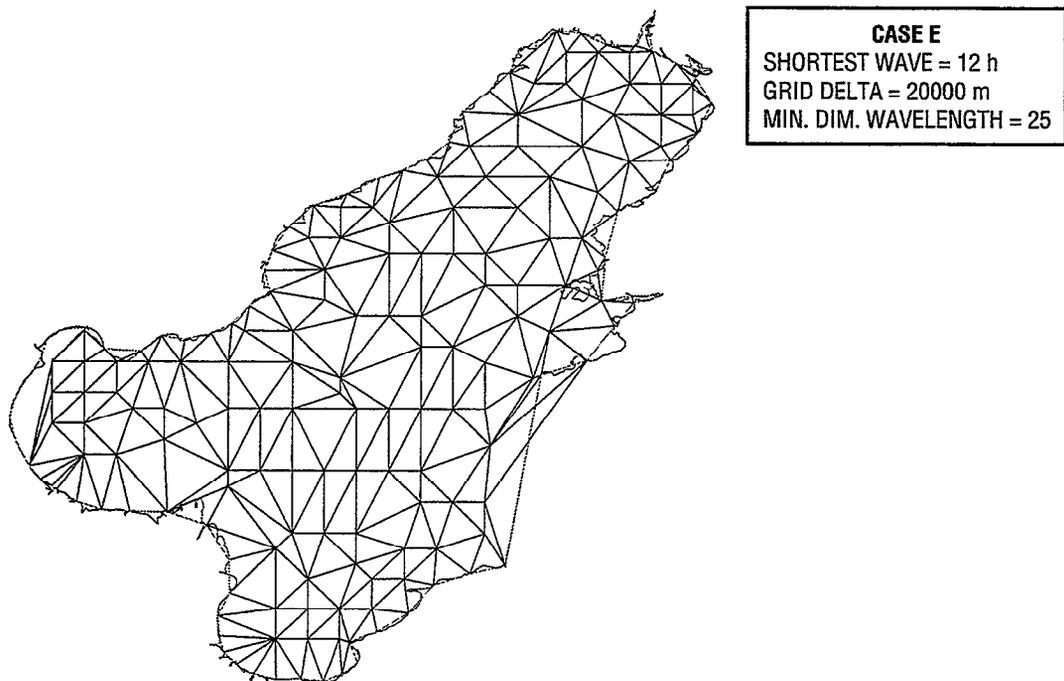


Fig. A.30 — An automatically generated finite element mesh for the Bohai Sea, case E

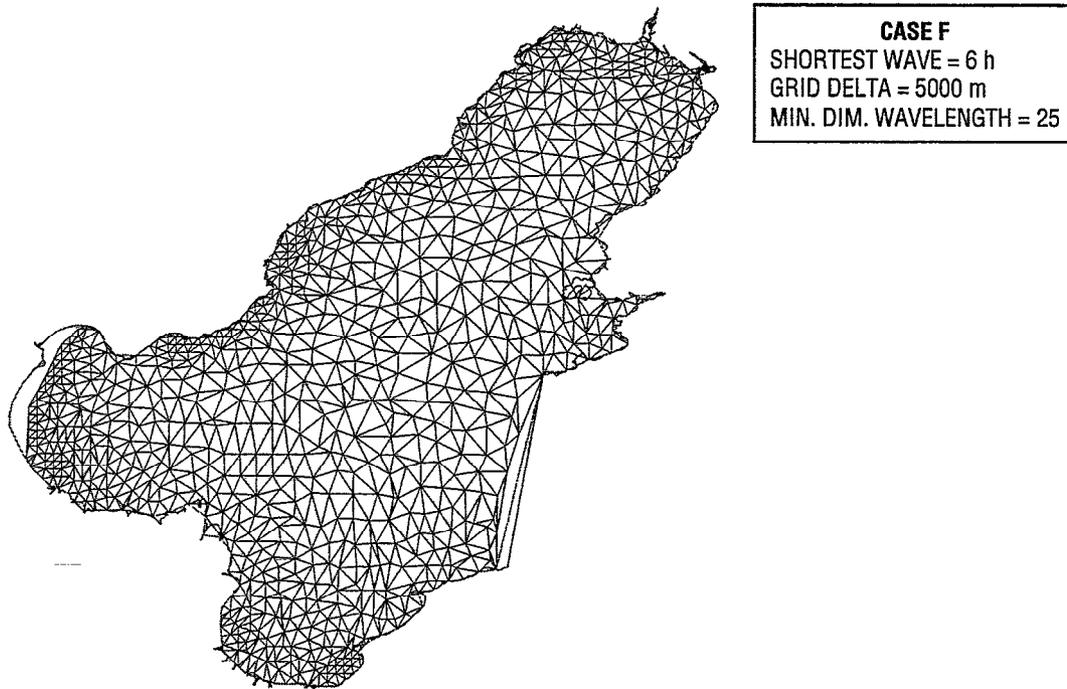


Fig. A.31 — An automatically generated finite element mesh for the Bohai Sea, case F

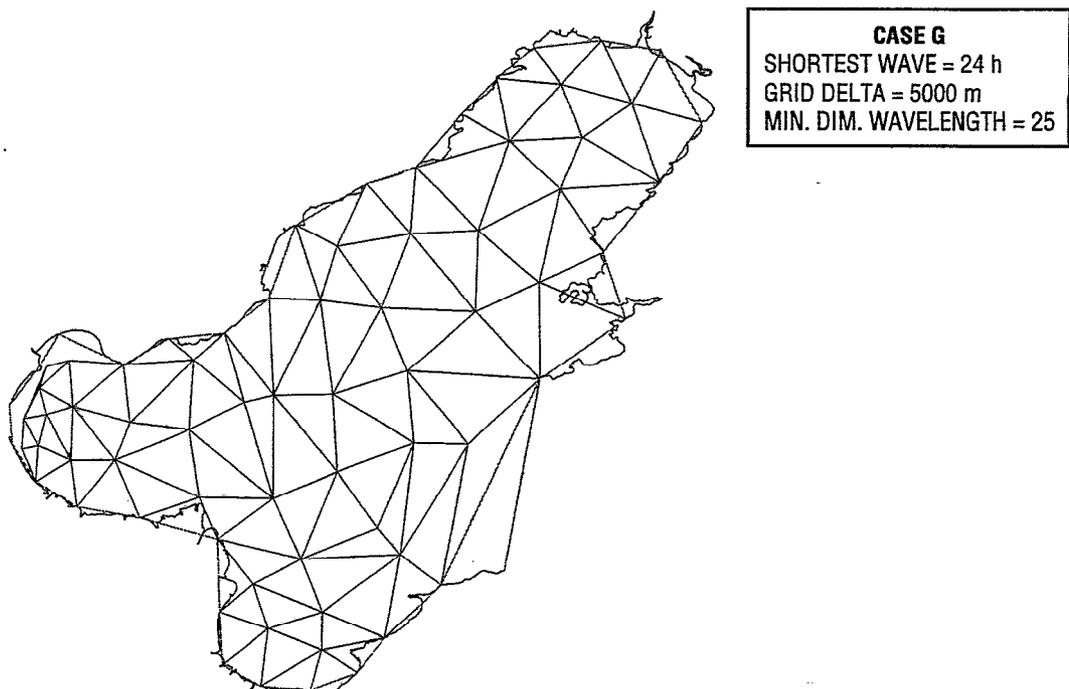


Fig. A.32 — An automatically generated finite element mesh for the Bohai Sea, case G

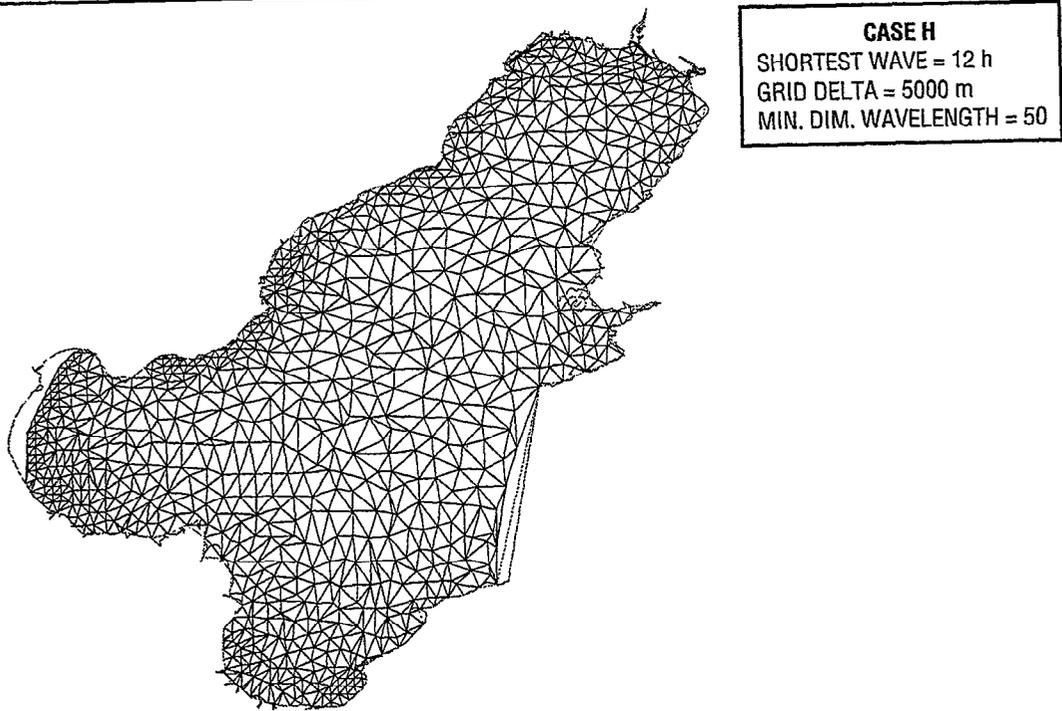


Fig. A.33 — An automatically generated finite element mesh for the Bohai Sea, case H

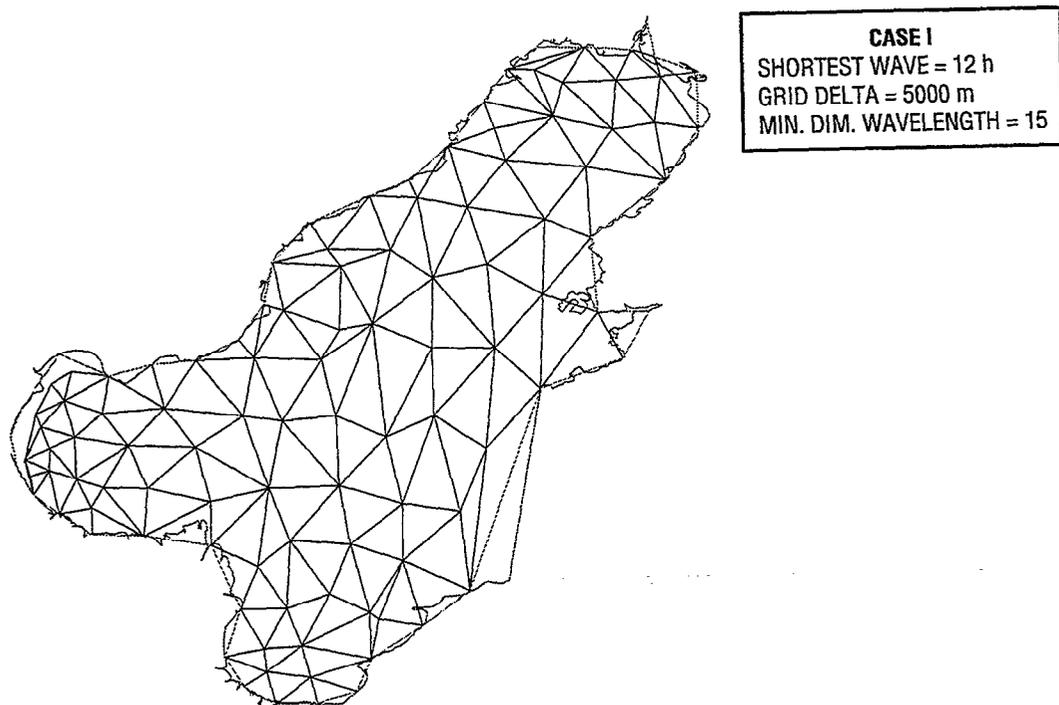


Fig. A.34 — An automatically generated finite element mesh for the Bohai Sea, case I

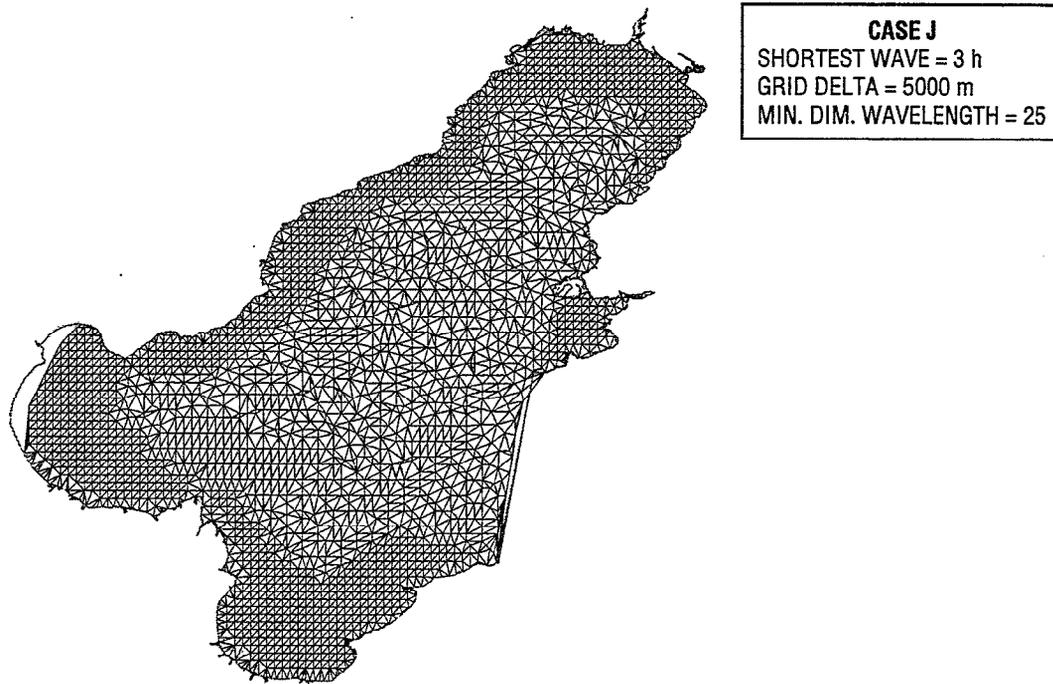


Fig. A.35 — An automatically generated finite element mesh for the Bohai Sea, case J

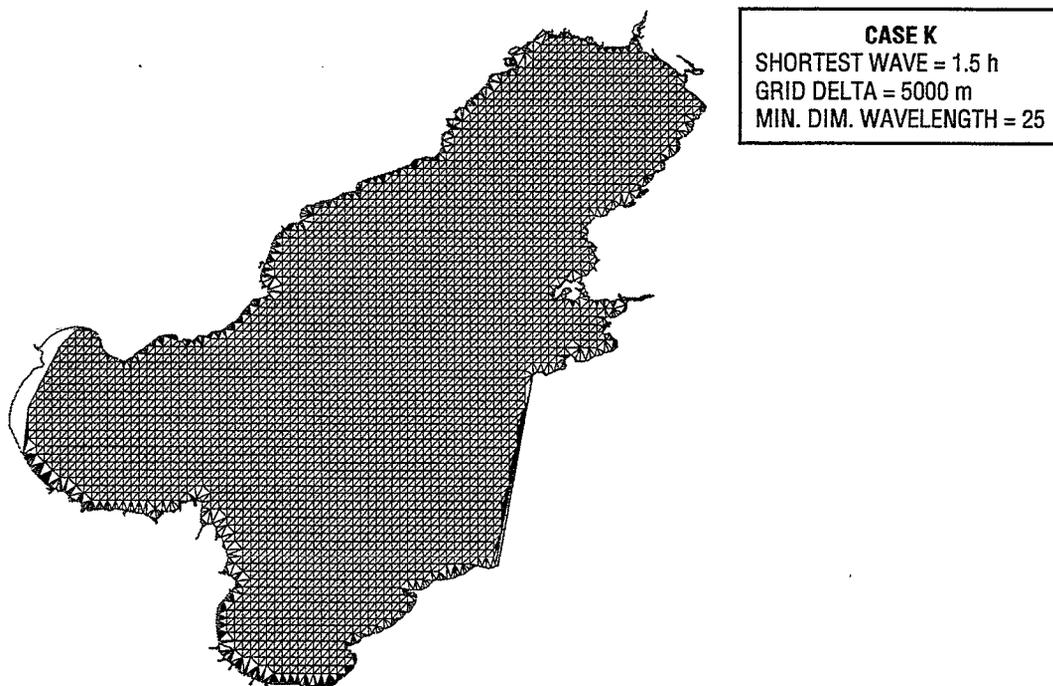


Fig. A.36 — An automatically generated finite element mesh for the Bohai Sea, case K

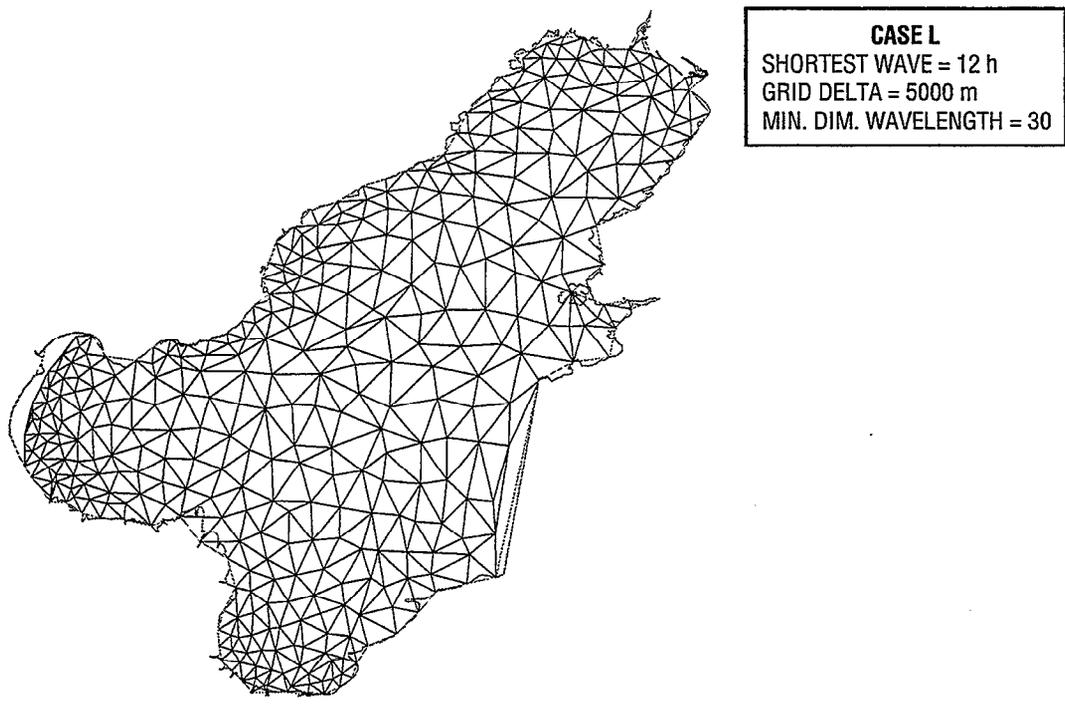


Fig. A.37 — An automatically generated finite element mesh for the Bohai Sea, case L

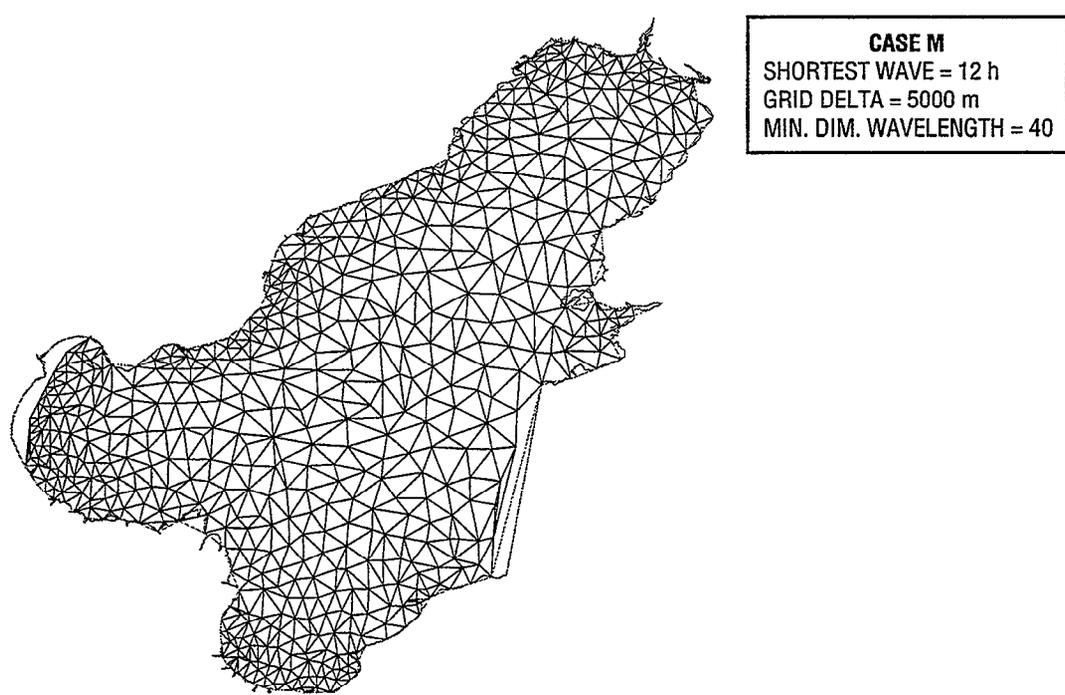


Fig. A.38 — An automatically generated finite element mesh for the Bohai Sea, case M

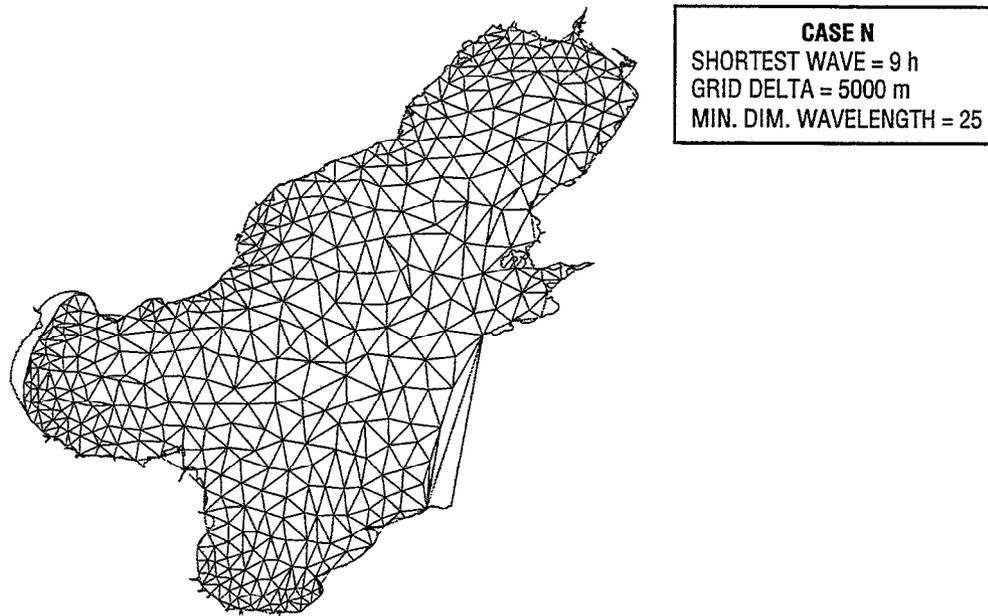


Fig. A.39 — An automatically generated finite element mesh for the Bohai Sea, case N

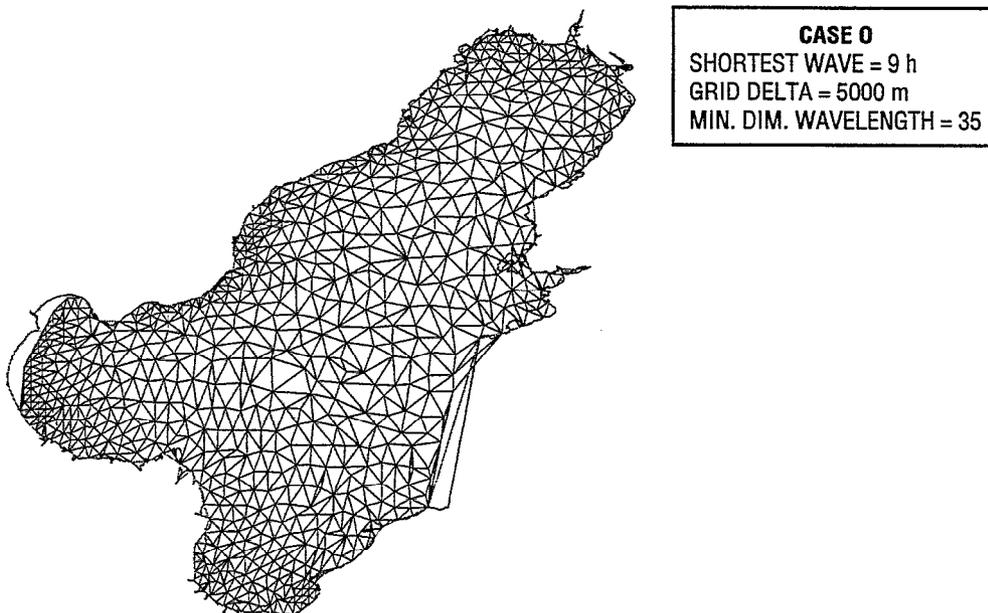


Fig. A.40 — An automatically generated finite element mesh for the Bohai Sea, case O

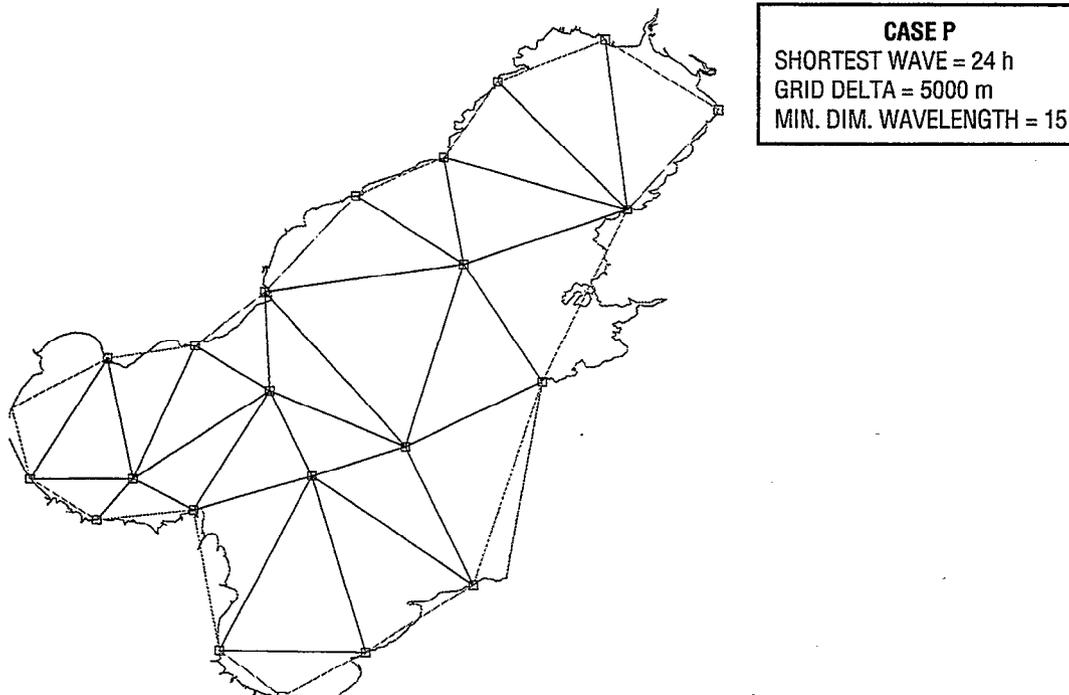


Fig. A.41 — An automatically generated finite element mesh for the Bohai Sea, case P

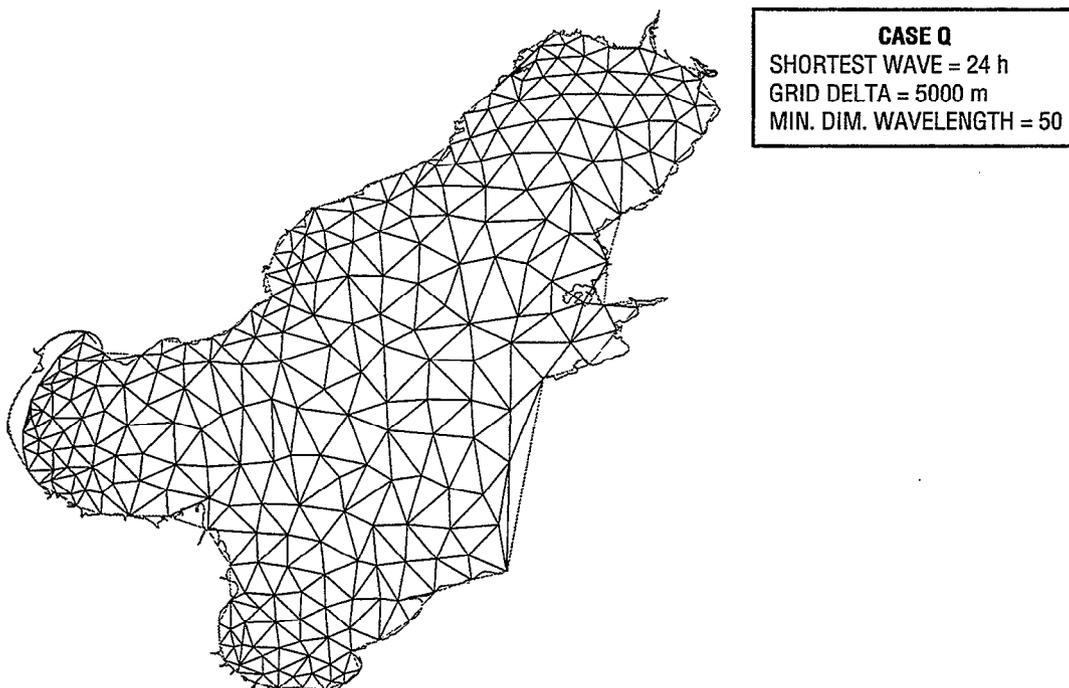


Fig. A.42 — An automatically generated finite element mesh for the Bohai Sea, case Q

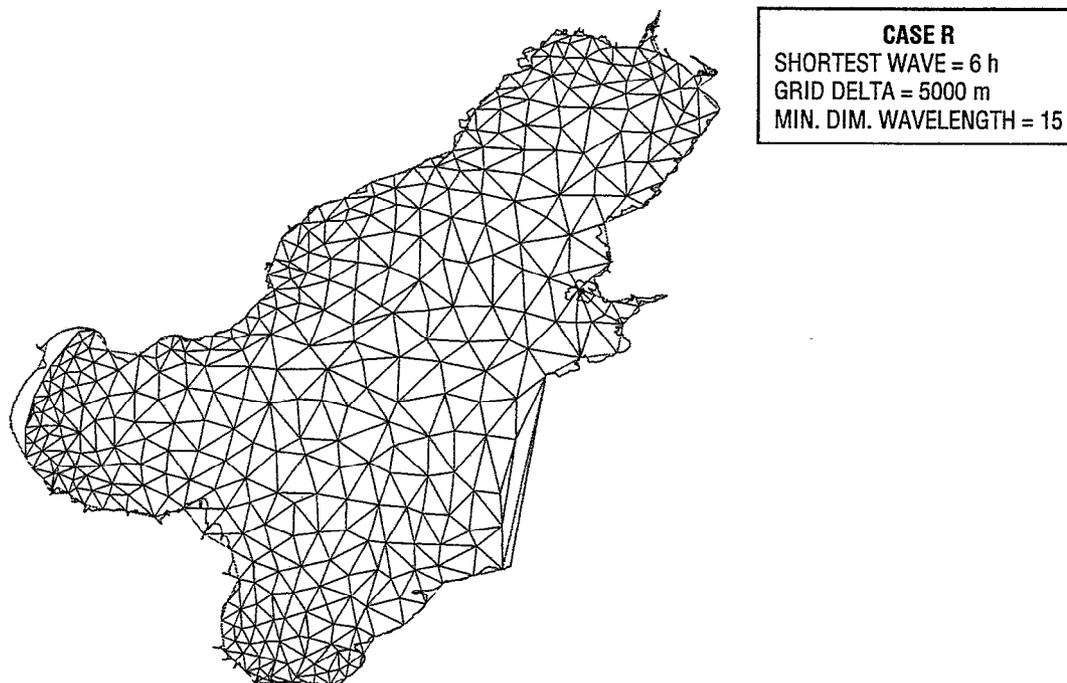


Fig. A.43 — An automatically generated finite element mesh for the Bohai Sea, case R

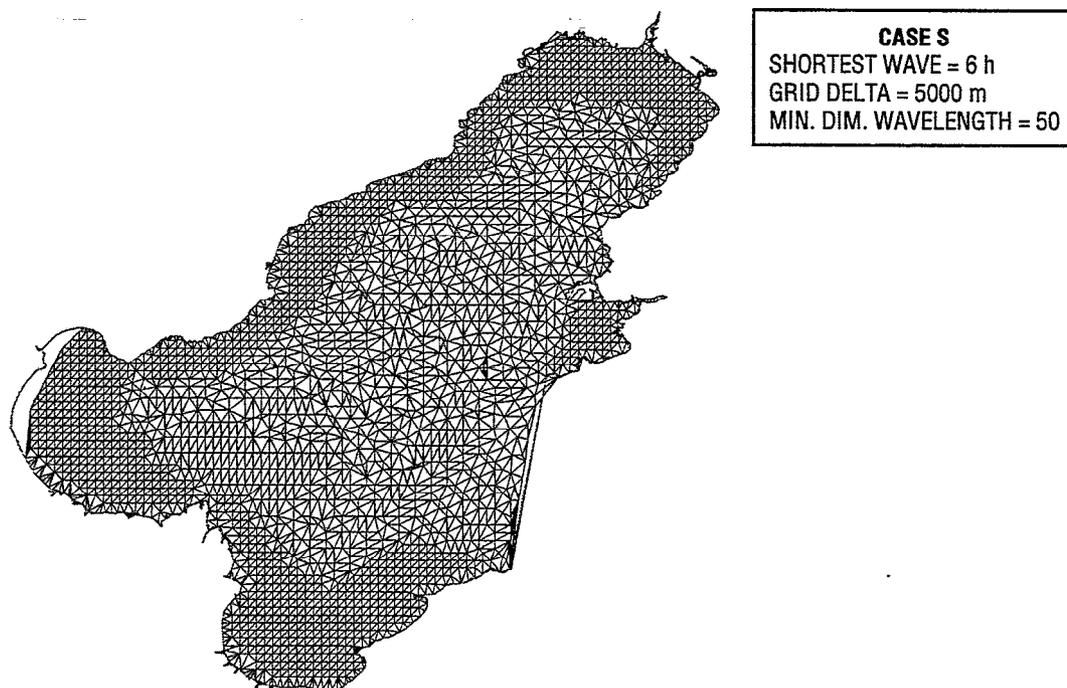


Fig. A.44 — An automatically generated finite element mesh for the Bohai Sea, case S

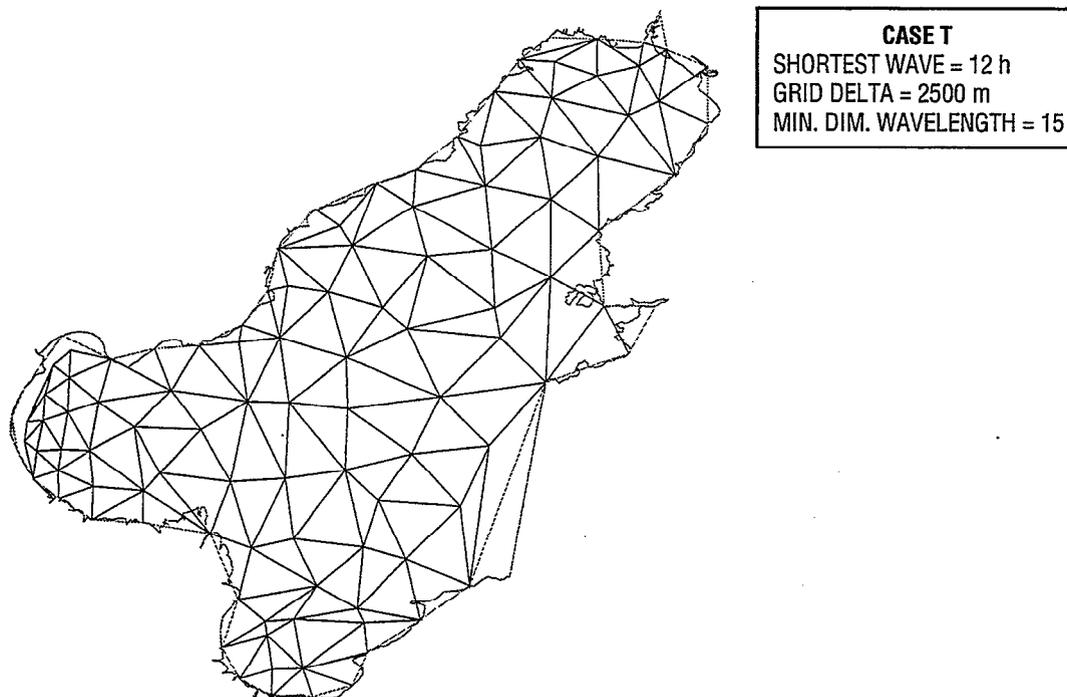


Fig. A.45 — An automatically generated finite element mesh for the Bohai Sea, case T

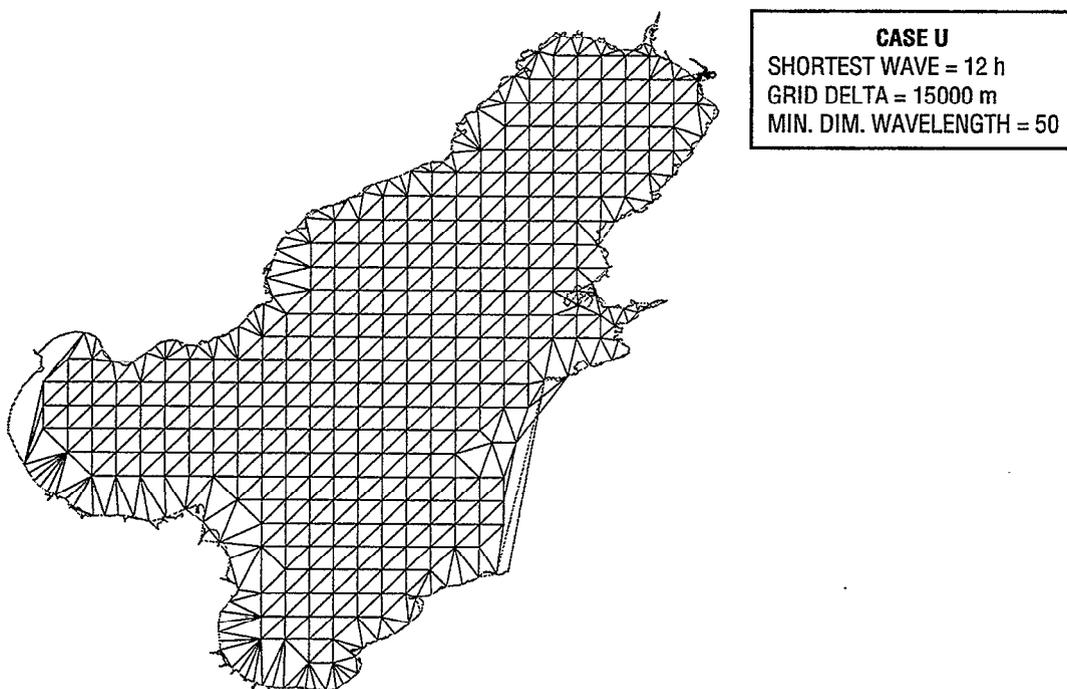


Fig. A.46 — An automatically generated finite element mesh for the Bohai Sea, case U

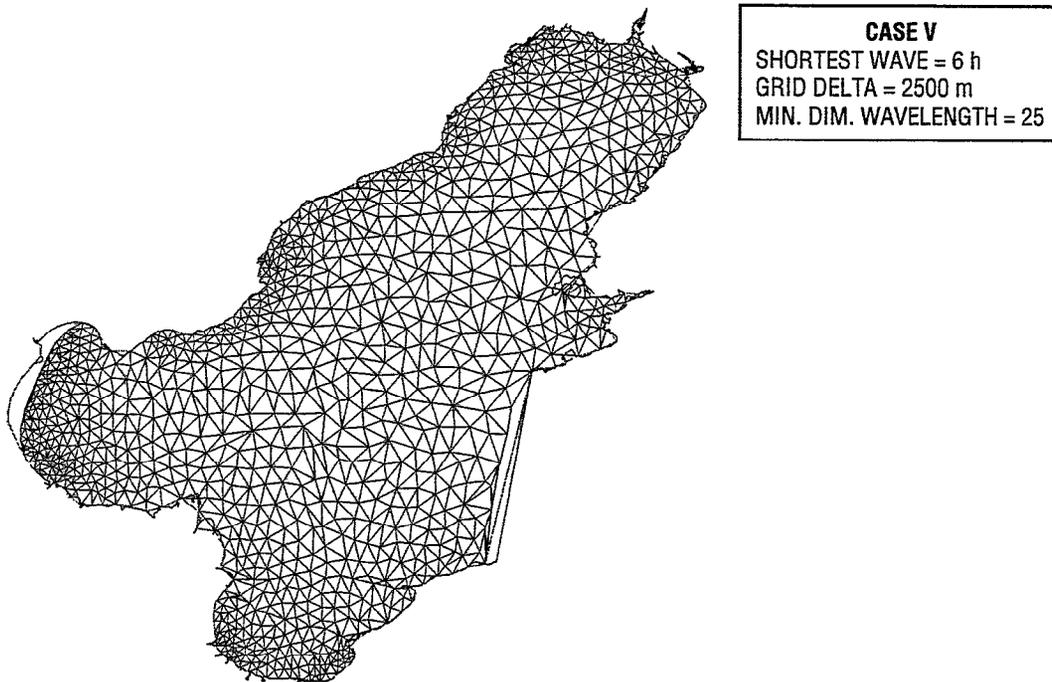


Fig. A.47 — An automatically generated finite element mesh for the Bohai Sea, case V

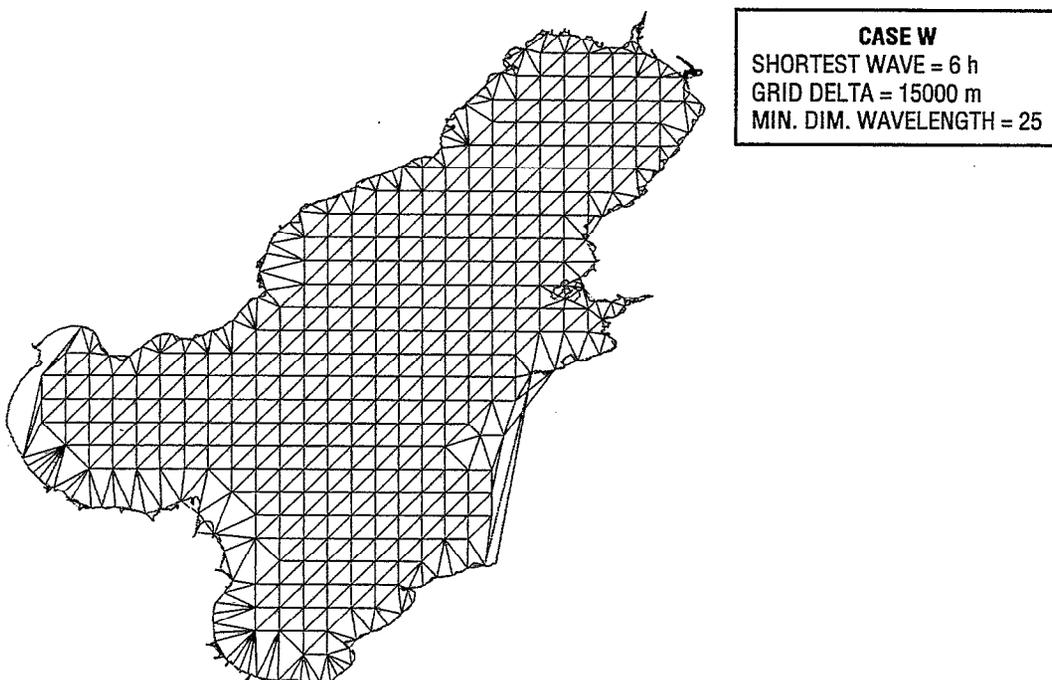


Fig. A.48 — An automatically generated finite element mesh for the Bohai Sea, case W

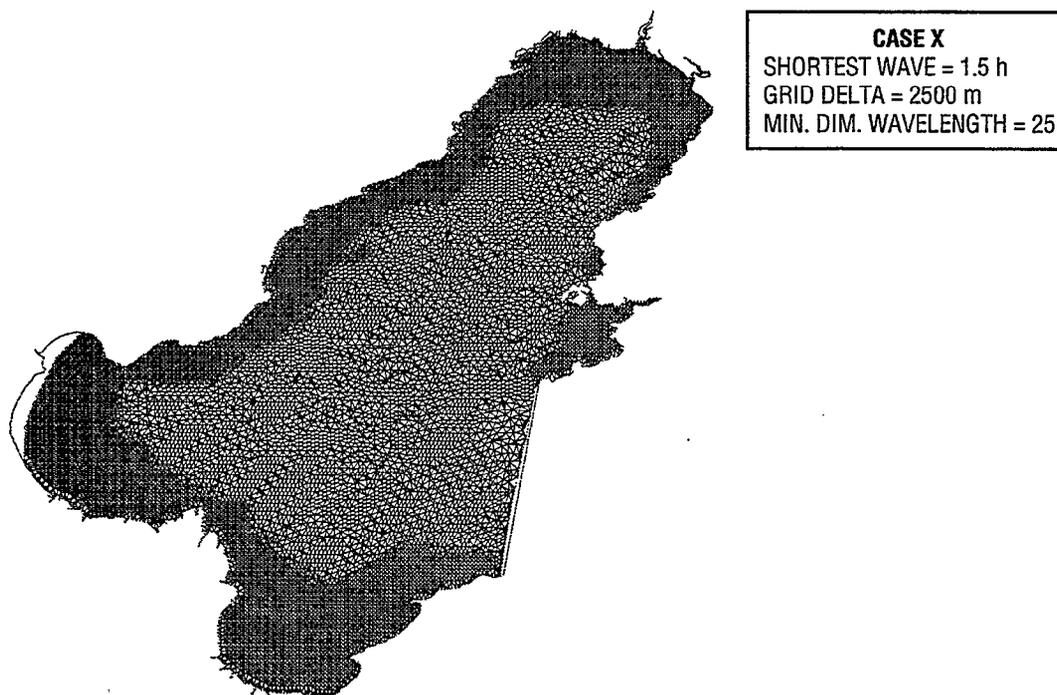


Fig. A.49 — An automatically generated finite element mesh for the Bohai Sea, case X

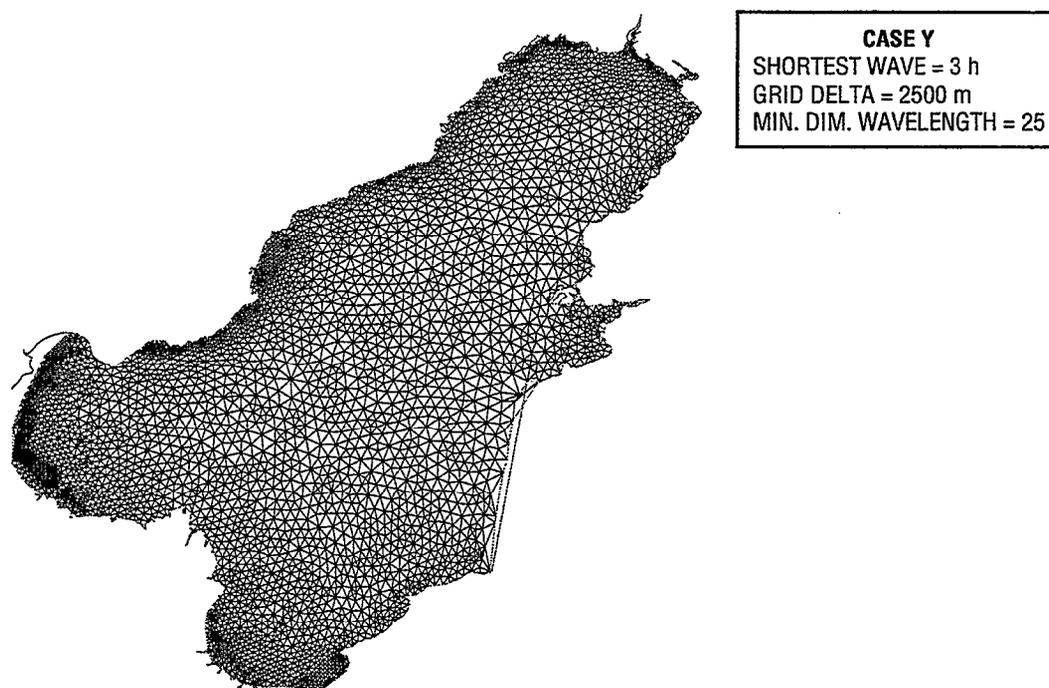


Fig. A.50 — An automatically generated finite element mesh for the Bohai Sea, case Y