FESW — a finite-element Fortran IV program for solving the shallow-water equations

I. M. NAVON and U. MULLER
National Research Institute for Mathematical Sciences of the CSIR, P.O. Box 395, Pretoria, South Africa

A Fortran IV computer program is documented, implementing a Galerkin finite-element method for solving the non-linear shallow-water equations on a limited domain. The resulting ordinary differential equations are integrated using a finite-difference discretization method in time. A time-extrapolated Crank–Nicolson numerical integration scheme is employed to quasi-linearize the non-linear advective terms. The three equations constituting the shallow-water equations are coupled at each time step, making it possible to use larger time steps. The output of the program includes a line printer plot contouring the height field. A compact storage scheme is provided in which advantage has been taken of the sparsity of the global matrices.

A Gauss–Seidel iterative procedure is employed to solve the linear systems of algebraic equations at each time step. Program options include the determination at each time step of the numerical integration of two of the integral invariants of the shallow-water equations.

Stable long-term runs were achieved using a 30-minute time step.

INTRODUCTION

The shallow-water equations are used in studies of tides and surface water run-off; they can also be used to study large-scale waves in the atmosphere and ocean if terms representing the effects of the earth's rotation (Coriolis terms) are included. Galerkin finite-element techniques have been applied to the shallow-water equations by the following workers, to cite but a few: Baker[1,2], Cullen[5–8], Brebbia and Partridge[3], Connor and Brebbia[4], Smith and Brebbia[9], Wang *et al.*[10] and Hinsman[11]. Only very few finite-element programs, however, have been published that are aimed at making it possible to practically apply the method to solve the shallow-water equations.

In the first part of this paper, illustrated by a test problem, a Galerkin finite-element application to the system of the shallow-water equations is described.

The second section is devoted to a description of the finite-difference method employed for the time integration, the implementation of boundary conditions in the finite-element model (f.e.m.) and a description of the different types of element matrices required for assembling the global matrices. A compact storage scheme is also briefly described, for which advantage has been taken of the sparsity of the assembled global coefficient matrices, and which makes it possible to reduce core storage to a minimum.

The remainder of the paper contains a detailed description of the program FESW and specifications for its use.

DESCRIPTION OF THE MODEL

Shallow-water equations

The shallow-water equations model can be written as follows:

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}+v\frac{\partial u}{\partial y}+\frac{\partial \varphi}{\partial x}-fv=0 \tag{1a}$$

$$\frac{\partial v}{\partial t}+u\frac{\partial v}{\partial x}+v\frac{\partial v}{\partial y}+\frac{\partial \varphi}{\partial y}+fu=0 \tag{1b}$$

$$\frac{\partial \varphi}{\partial t}+\frac{\partial}{\partial x}(\varphi u)+\frac{\partial}{\partial y}(\varphi v)=0 \tag{1c}$$

$$0\leqslant x\leqslant L, \qquad 0\leqslant y\leqslant D, \qquad t>0$$

where L and D are the dimensions of a rectangular domain of area $\bar{A}=LD$.

Here u and v are the velocity components in the x and y directions respectively; $\varphi=gh$ is the geopotential; h is the depth of the fluid; g is the acceleration of gravity; and f is the Coriolis parameter, required when we consider a fluid in a rotating frame of reference.

The Coriolis term f is given by:

$$f=\bar{f}+\beta(y-D/2) \qquad \beta=\frac{\partial f}{\partial y} \tag{2}$$

with $\bar{f}$ and β constants.

Periodic boundary conditions are assumed in the x direction while in the y direction the boundary condition is:

$$v(x, 0, t)=v(x, D, t)=0 \tag{3}$$

With these boundary conditions and with the initial condition:

$$w(x, y, 0)=\varphi(x, y) \tag{4}$$

where

$$w=(u, v, \varphi)^T \tag{5}$$

the total energy

$$E=\tfrac{1}{2}\int_0^L\int_0^D (u^2+v^2+\varphi)\frac{\varphi}{g}\,\mathrm{d}x\mathrm{d}y \tag{6}$$

is independent of time.

Also the average value of the height, which is proportional to the total mass:

$$\bar{h}=\frac{1}{A}\int_0^L\int_0^D h\mathrm{d}x\mathrm{d}y \tag{7}$$

is independent of time.

Test problem

The test problem used here is the initial height field condition no. 1 of Grammeltvedt[12]:

$$h(x, y)=H_0+H_1\tanh\left(\frac{9(D/2-y)}{2D}\right)+H_2\mathrm{sech}^2\left(\frac{9(D/2-y)}{2D}\right)\sin\left(\frac{2\pi x}{L}\right) \tag{8}$$

The initial velocity fields were derived from the initial height field using the geostrophic relationship:

$$u=-(g/f)\frac{\partial h}{\partial y}, \qquad v=(g/f)\frac{\partial h}{\partial x} \tag{9}$$

The constants used were

$$\begin{array}{ll} L=4400\ \mathrm{km} & g=10\ \mathrm{m/s} \\ D=6000\ \mathrm{km} & H_0=2000\ \mathrm{m} \\ f=10^{-4}\ \mathrm{s}^{-1} & H_1=220\ \mathrm{m} \\ \beta=1.5\times10^{-11}\ \mathrm{s}^{-1}\,\mathrm{m}^{-1} & H_2=133\ \mathrm{m} \end{array} \tag{10}$$

The time and space increments used were

$$\Delta x=\Delta y=400\ \mathrm{km}$$

$$\Delta t=1800\ \mathrm{s} \tag{11}$$

Formulation of the finite-element model

We approximate the shallow-water equations model equation (1), by the Galerkin f.e.m. The rectangular domain is subdivided into triangular elements forming a regular grid.

Linear piecewise polynomial interpolation functions were employed, to save computing time and also for the sake of simplicity. Over a given triangular element, each variable was represented as a linear sum of the interpolation functions, i.e.

$$u_{el}=\sum_{j=1}^{3} u_j(t)V_j \tag{12}$$

where $u_j(t)$ represents the scalar nodal value of the variable u at the node j of the triangular element, and V_j is the basis function (interpolation function) which can be defined by the coordinates of the nodes.

Galerkin's f.e.m. is a particular weighted-residual method in which the trial functions are the same as the basis functions used for representing the variables.

For instance, given the system of equations:

$$L(u)-f=0 \qquad x\in A \tag{13}$$

with the boundary conditions:

$$S(u)=p \qquad x\in S \tag{14}$$

and an approximating function:

$$u=\sum_{k=1}^{N}\alpha_k V_k \tag{15}$$

which satisfies the boundary conditions (14), the residual

$$\varepsilon=L(\Sigma\alpha_k V_k)-f \tag{16}$$

is orthogonalized with respect to the trial functions V_i

$$\iint_A [L(\Sigma\alpha_k V_k)-f]V_i\mathrm{d}A=0, \qquad i=1, 2, \ldots, N \tag{17}$$

and this relationship can also be written:

$$\langle\varepsilon, V_i\rangle=0 \qquad i=1, 2, \ldots, N \tag{18}$$

The notation of equation (18) defines the inner product.

Relation (18) holds also for an arbitrary subdomain or element of the whole domain and we may focus our attention on individual elements, provided the basis functions V_i guarantee the interelement continuity necessary for the assembly process.

We have a set of equations such as (17) for each element of the whole domain. The next step in the Galerkin f.e.m. is to assemble the equation for the whole domain following well-known assembly rules. The complete system of equations thus assembled is next solved for the time-dependent coefficients of the basis functions. The advantage of the Galerkin f.e.m. is that it enables us to proceed and derive a finite-element model even in the absence of a classical variational principle.

Let us now start with the continuity equation which is the first to be solved during a time step.

Writing it following the Galerkin f.e.m. we obtain:

$$\left\langle\frac{\partial\varphi}{\partial t}, V_i\right\rangle+\left\langle\frac{\partial}{\partial x}(\varphi u), V_1\right\rangle+\left\langle\frac{\partial}{\partial y}(\varphi v), V_i\right\rangle=0 \tag{19}$$

where by the notation of the inner product of each term with the trial function we mean:

$$\langle f(x, y), V_i\rangle=\sum^{M}\iint_{\text{element}} f(x, y)V_i\mathrm{d}x\mathrm{d}y=\iint_{\text{global}} f(x, y)V_i\mathrm{d}x\mathrm{d}y \tag{20}$$

where M is the number of elements in the integration domain.

The advection terms in the continuity equation are usually integrated by parts (using Green's theorem) to shift from derivatives of the variable to derivatives of the basis function. This permits the use of basis functions with lower-order interelement continuity and often offers a convenient way of introducing the natural boundary conditions that must be satisfied on some portion of the boundary. This integration gives:

$$\langle\frac{\partial\varphi}{\partial t}, V_i\rangle+\int(\varphi u V_i)\Big|_{x|0}^{L}\mathrm{d}y-\langle(\varphi u),\frac{\partial V_i}{\partial x}\rangle+$$

$$\int(\varphi v V_i)\Big|_{y|0}^{D}\mathrm{d}x-\langle(\varphi v),\frac{\partial V_i}{\partial y}\rangle=0 \qquad (21)$$

Taking into account the cyclic boundary conditions in the x direction and the boundary condition on v, the component of velocity in the y direction, the second and fourth terms of equation (21) vanish.

The final expression for the continuity equation is:

$$\langle\frac{\partial\varphi}{\partial t}, V_i\rangle-\langle(\varphi u),\frac{\partial V_i}{\partial x}\rangle-\langle(\varphi v),\frac{\partial V_i}{\partial y}\rangle=0 \qquad (22)$$

Following the Galerkin f.e.m., the momentum equations (1a) and (1b) are written as:

$$\langle\frac{\partial u}{\partial t}, V_i\rangle+\langle u\frac{\partial u}{\partial x}, V_i\rangle+\langle v\frac{\partial u}{\partial y}, V_i\rangle-\langle fv, V_i\rangle+\langle\frac{\partial\varphi}{\partial x}, V_i\rangle=0 \qquad (23)$$

$$\langle\frac{\partial v}{\partial t}, V_i\rangle+\langle u\frac{\partial v}{\partial x}, V_i\rangle+\langle v\frac{\partial v}{\partial y}, V_i\rangle+\langle Fu, V_i\rangle+\langle\frac{\partial\varphi}{\partial y}, V_i\rangle=0 \qquad (24)$$

We assume that over an element the same basis functions V apply for the u, v, φ unknowns, i.e. that:

$$\varphi\simeq\sum_{j=1}^{3}\varphi_j(t)V_j$$

$$u\simeq\sum_{j=1}^{3}u_j(t)V_j \qquad (25)$$

$$v\simeq\sum_{j=1}^{3}v_j(t)V_j$$

where $\varphi_j(t)$, $u_j(t)$, $v_j(t)$ are the time-dependent nodal values of the variables φ, u, v respectively.

Upon substituting these expressions into equations (22)–(24) one obtains:

$$\langle\frac{\partial\varphi_j}{\partial t}V_j, V_i\rangle-\langle\varphi_j u_k V_j V_k, \frac{\partial V_i}{\partial x}\rangle-\langle\varphi_j v_k V_j V_k, \frac{\partial V_i}{\partial y}\rangle=0 \qquad (26)$$

$$\langle\frac{\partial u_j}{\partial t}V_j, V_i\rangle+\langle u_k V_k u_j\frac{\partial V_j}{\partial x}, V_i\rangle+\langle v_k V_k u_j\frac{\partial V_j}{\partial y}, V_i\rangle-$$

$$\langle fv_k V_k, V_i\rangle+\langle\varphi_k\frac{\partial V_k}{\partial x}, V_i\rangle=0 \qquad (27)$$

$$\langle\frac{\partial v_j}{\partial t}V_j, V_i\rangle+\langle u_k V_k v_j\frac{\partial V_j}{\partial x}, V_i\rangle+\langle v_k V_k v_j\frac{\partial V_j}{\partial y}, V_i\rangle+$$

$$\Delta\langle fu_k V_k, V_i\rangle+\langle\varphi_k\frac{\partial V_k}{\partial y}, V_i\rangle=0 \qquad (28)$$

IMPLEMENTATION OF THE GALERKIN f.e.m.

Time integration

The time-extrapolated Crank–Nicolson method was used for integrating in time the system of ordinary differential equations resulting from the application of the Galerkin f.e.m. to the shallow-water equations model.

In this method, previously used by Douglas and Dupont[13] and Hinsman[11], an average is taken at time levels N and $N+1$ of expressions involving space derivatives, while the non-linear advective terms are quasi-linearized by estimating them at time level $N+\frac{1}{2}$ using the following second-order approximation in time:

$$u^{N+\frac{1}{2}}=u^*=\tfrac{3}{2}u^N-\tfrac{1}{2}u^{N-1}+0(\Delta t^2)$$

$$v^{N+\frac{1}{2}}=v^*=\tfrac{3}{2}v^N-\tfrac{1}{2}v^{N-1}+0(\Delta t^2) \qquad (29)$$

At each time step the shallow-water equations system was coupled, i.e. the solution of each equation after one iteration at a given time step was used to solve the other two equations for the same iteration for the same time step.

Upon introducing time discretization in the continuity equation (26), which is the first to be solved at a given time step, one obtains:

$$\langle(\varphi_j^{n+1}-\varphi_j^n)V_j, V_i\rangle-$$

$$\frac{\Delta t}{2}\left[\langle\varphi_j^{n+1}u_k^*V_jV_k, \frac{\partial V_i}{\partial x}\rangle+\langle\varphi_j^{n+1}v_k^*V_jV_k, \frac{\partial V_i}{\partial y}\rangle\right]-$$

$$\frac{\Delta t}{2}\left[\langle\varphi_j^{n}u_k^*V_jV_k, \frac{\partial V_i}{\partial x}\rangle+\langle\varphi_j^{n}v_k^*V_jV_k, \frac{\partial V_i}{\partial y}\rangle\right]=0 \qquad (30)$$

By defining the following matrices:

$$M=\int\int V_jV_i\mathrm{d}A$$

$$K_1=\iint_A V_jV_ku_k^*\frac{\partial V_i}{\partial x}\mathrm{d}A+\iint_A V_jV_kv_k^*\frac{\partial V_i}{\partial y}\mathrm{d}A \qquad (31)$$

the continuity equation can be written as:

$$M(\varphi_j^{n+1}-\varphi_j^n)-\frac{\Delta t}{2}K_1(\varphi_j^{n+1}+\varphi_j^n)=0 \qquad (32)$$

Introducing time discretization in the same way into the momentum equations (27) and (28), one obtains:

$$\langle (u_j^{n+1}-u_j^n)V_j, V_i\rangle +$$

$$\frac{\Delta t}{2}\left[\langle u_j^{n+1}u_k^* V_k \frac{\partial V_j}{\partial x}, V_i\rangle + \langle u_j^{n+1}v_k^* V_k \frac{\partial V_j}{\partial y}, V_i\rangle + \langle \varphi_k^{n+1}\frac{\partial V_k}{\partial x}, V_i\rangle\right] +$$

$$\frac{\Delta t}{2}\left[\langle u_j^n u_k^* V_k \frac{\partial V_j}{\partial x}, V_i\rangle + \langle u_j^n v_k^* V_k \frac{\partial V_j}{\partial y}, V_i\rangle + \langle \varphi_k^n \frac{\partial V_k}{\partial x}, V_i\rangle\right] -$$

$$\Delta t\langle f v_k^* V_k, V_i\rangle = 0 \tag{33}$$

and

$$\langle (v_j^{n+1}-v_j^n)V_j, V_i\rangle +$$

$$\frac{\Delta t}{2}\left[\langle v_j^{n+1}u_k^{n+1} V_k \frac{\partial V_j}{\partial x}, V_i\rangle + \langle v_j^{n+1}v_k^* V_k \frac{\partial V_j}{\partial y}, V_i\rangle + \langle \varphi_k^{n+1}\frac{\partial V_k}{\partial y}, V_i\rangle\right] +$$

$$\frac{\Delta t}{2}\left[\langle v_j^n u_k^n V_k \frac{\partial V_j}{\partial x}, V_i\rangle + \langle v_j^n v_j^* V_k \frac{\partial V_j}{\partial y}, V_i\rangle + \langle \varphi_k^n \frac{\partial V_k}{\partial y}, V_i\rangle\right] +$$

$$\Delta t\langle f u_k^{n+1} V_k, V_i\rangle = 0 \tag{34}$$

Using the following matrix definitions:

$$M = \int\int_A V_j V_i \mathrm{d}A$$

$$K_2 = \int\int_A u_k^* V_k V_i \frac{\partial V_j}{\partial x}\mathrm{d}A + \int\int v_k^* V_k V_i \frac{\partial V_j}{\partial y}\mathrm{d}A \tag{35}$$

$$K_{21} = \int\int_A \varphi_k \frac{\partial V_k}{\partial x} V_i \mathrm{d}A$$

$$P_2 = -\int\int_A f v_k^* V_k V_i \mathrm{d}A$$

the u-momentum equation becomes:

$$M(u_j^{n+1}-u_j^n) + \frac{\Delta t}{2}K_2(u_j^{n+1}+u_j^n) + \frac{\Delta t}{2}(K_{21}^{n+1}+K_{21}^n) + \Delta t P_2 = 0 \tag{36}$$

while by defining:

$$K_3 = \int\int_A u_k^{n+1} V_k \frac{\partial V_j}{\partial x} V_i \mathrm{d}A + \int\int_A v_k^* V_k \frac{\partial V_j}{\partial y}\mathrm{d}A$$

$$K_{31} = \int\int_A \varphi_k \frac{\partial V_k}{\partial y} V_i \mathrm{d}A \tag{37}$$

$$P_3 = \int\int_A f u_k^{n+1} V_k V_i \mathrm{d}A$$

The v-momentum equation becomes:

$$M(v_j^{n+1}-v_j^n) + \frac{\Delta t}{2}K_3(v_j^{n+1}+v_j^n) + \frac{\Delta t}{2}(K_{31}^{n+1}+K_{31}^n) + \Delta t P_3 = 0 \tag{38}$$

Element matrices

Using linear basis functions over triangular elements and introducing the well-known natural or area coordinates[14] one can obtain exact integrations using the following formula for area integrals[15]:

$$\int\int_A L_1^a L_2^b L_3^c \mathrm{d}x\mathrm{d}y = \frac{a!b!c}{(a+b+c+2)!} \tag{39}$$

(a, b, c integers), where $L_i (i = 1, 2, 3)$ are the basis functions for the triangular linear element as well as the natural coordinate variables.

The natural coordinates in terms of the Cartesian coordinates for a given triangle are:

$$L_i \frac{1}{2A}(a_i y + b_i x + c_i) \qquad i = 1, 2, 3 \tag{40}$$

where

$$2A = \begin{vmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{vmatrix} = 2(\textit{area of triangle } 1\text{-}2, \text{-}3) \tag{41}$$

$$a_i = y_j - y_k, \qquad b_i = x_k - x_j, \qquad c_i = x_j y_k - v_k y_j \tag{42}$$

for instance

$$a_1 = y_2 - y_3, \qquad b_1 = x_3 - x_2, \qquad c_1 = x_2 y_3 - x_3 y_2 \tag{43}$$

i, j, k cyclically permuted ($i, j, k = 1, 2, 3$).

The derivatives of the shape functions L_i are:

$$\frac{\partial L_i}{\partial x} = \frac{b_i}{2A}, \qquad \frac{\partial L_i}{\partial y} = \frac{c_i}{2A} \qquad i = 1, 2, 3 \tag{44}$$

There are basically four types of element (3×3) matrices required in the Galerkin f.e.m. of the shallow-water equations, as follows:

(a) $$M=\iint_A V_j V_i \mathrm{d}A \qquad i,j=1,2,3 \tag{45}$$

The integration formula (39) yields:

$$\iint_A V_j V_i \mathrm{d}A=\frac{A}{12}\begin{pmatrix}2 & 1 & 1\\ 1 & 2 & 1\\ 1 & 1 & 2\end{pmatrix} \tag{46}$$

(b) $$\iint_A V_k \frac{\partial V_i}{\partial x}\mathrm{d}A \quad \text{or} \quad \iint_A V_k \frac{\partial V_i}{\partial y}\mathrm{d}A \tag{47}$$

The use of integration formula (39) in conjunction with equation (44) yields, for instance,

$$\iint_A V_k \frac{\partial V_i}{\partial x}\mathrm{d}A=\iint_A V_k \frac{b_i}{2A}\mathrm{d}A=2A\cdot\frac{1}{6}\frac{b_i}{2A}=\frac{b_i}{6} \tag{48}$$

i.e.

$$\iint V_k \frac{\partial V_i}{\partial x}\mathrm{d}A=\frac{1}{6}\begin{pmatrix}b_1 & b_2 & b_3\\ b_1 & b_2 & b_3\\ b_1 & b_2 & b_3\end{pmatrix} \tag{49}$$

(c) $$\iint_A V_k p_j \frac{\partial V_j}{\partial x} V_i \mathrm{d}A \quad \text{or} \quad \iint_A V_k p_j \frac{\partial V_j}{\partial y} V_i \mathrm{d}A \tag{50}$$

where p_j stands for either u_j, v_j or φ_j.

For instance

$$\iint_A V_k p_j \frac{\partial V_j}{\partial x} V_i \mathrm{d}A=\iint_A V_k p_j \frac{b_j}{2A} V_i \mathrm{d}A$$

$$=\frac{1}{2A}p_j b_j \iint V_i V_k \mathrm{d}A$$

$$=\frac{1}{2A}\Sigma p_j b_j \frac{A}{12}\begin{bmatrix}2 & 1 & 1\\ 1 & 2 & 1\\ 1 & 1 & 2\end{bmatrix}=\frac{1}{24}\Sigma p_j b_j \begin{bmatrix}2 & 1 & 1\\ 1 & 2 & 1\\ 1 & 1 & 2\end{bmatrix} \tag{51}$$

(d) $$\iint_A V_j p_j V_k \frac{\partial V_i}{\partial x}\mathrm{d}A \quad \text{or} \quad \iint_A V_j p_j V_k \frac{\partial V_i}{\partial y}\mathrm{d}A \tag{52}$$

For instance

$$\iint_A V_j p_j V_k \frac{\partial V_i}{\partial x}\mathrm{d}A=\iint_A V_j p_j V_k \frac{\partial b_i}{2A}\mathrm{d}A=$$

$$\frac{b_i}{2A}\left(\sum_{\substack{j\neq k\\ j=1}}^{3}\frac{A}{12}p_j+\frac{A}{6}p_k\right)$$

$$=\frac{1}{24}\begin{pmatrix}(2p_1+p_2+p_3)b_1 & (2p_1+p_2+p_3)b_2 & (2p_1+p_2+p_3)b_3\\ (p_1+2p_2+p_3)b_1 & (p_1+2p_2+p_3)b_2 & (p_1+2p_2+p_3)b_3\\ (p_1+p_2+2p_3)b_1 & (p_1+p_2+2p_3)b_2 & (p_1+p_2+2p_3)b_3\end{pmatrix} \tag{53}$$

Implementation of the boundary conditions

After the assembly process we obtain a global $N\times N$ matrix K, and the system of linear equations to be solved has the form:

$$\underset{(n\times n)}{K}\ \underset{(n\times 1)}{X}=\underset{(n\times 1)}{R} \tag{54}$$

In order to implement boundary conditions in the Galerkin f.e.m. we have here adopted an approach suggested by Payne and Irons[16] and mentioned by Huebner[17]. This approach consists in modifying the diagonal terms of K associated with the specified nodal variables by multiplying each term by a large number, say 10^{16} (chosen by consideration of the significant number of digits for the given computer and the size of the field variables), while the corresponding term in R is replaced by the specified nodal variable multiplied by the same large factor times the corresponding diagonal term. The procedure is repeated until all prescribed boundary nodal variables have been treated.

After having made these modifications one can then proceed to solve the set of equations using the modified matrix K and the modified vector R.

For instance if in the matrix K we wish to implement the boundary condition:

$$X_r=\beta_r \tag{55}$$

Then the modification is:

$$\begin{bmatrix}k_{11} & k_{12} & & & k_{1N}\\ & \ddots & & & \\ k_{r1} & k_{r2} & k_{rr}10^{16} & & k_{rN}\\ & & & \ddots & \\ k_{N1} & k_{N2} & & & k_{NN}\end{bmatrix}\begin{bmatrix}X_1\\ X_2\\ \vdots\\ X_r\\ \vdots\\ X_N\end{bmatrix}=\begin{bmatrix}R_1\\ R_2\\ \vdots\\ \beta_r\cdot k_{rr}\cdot 10^{16}\\ \vdots\\ R_N\end{bmatrix} \tag{56}$$

If the rth equation is considered, it can be observed that the desired boundary condition has been imposed as:

$$k_{r1}X_1+k_{r2}X_2+\ldots+k_{rr}\cdot 10^{16}X_r+\ldots+k_{rN}X_N=\beta_r k_{rr}\cdot 10^{16} \tag{57}$$

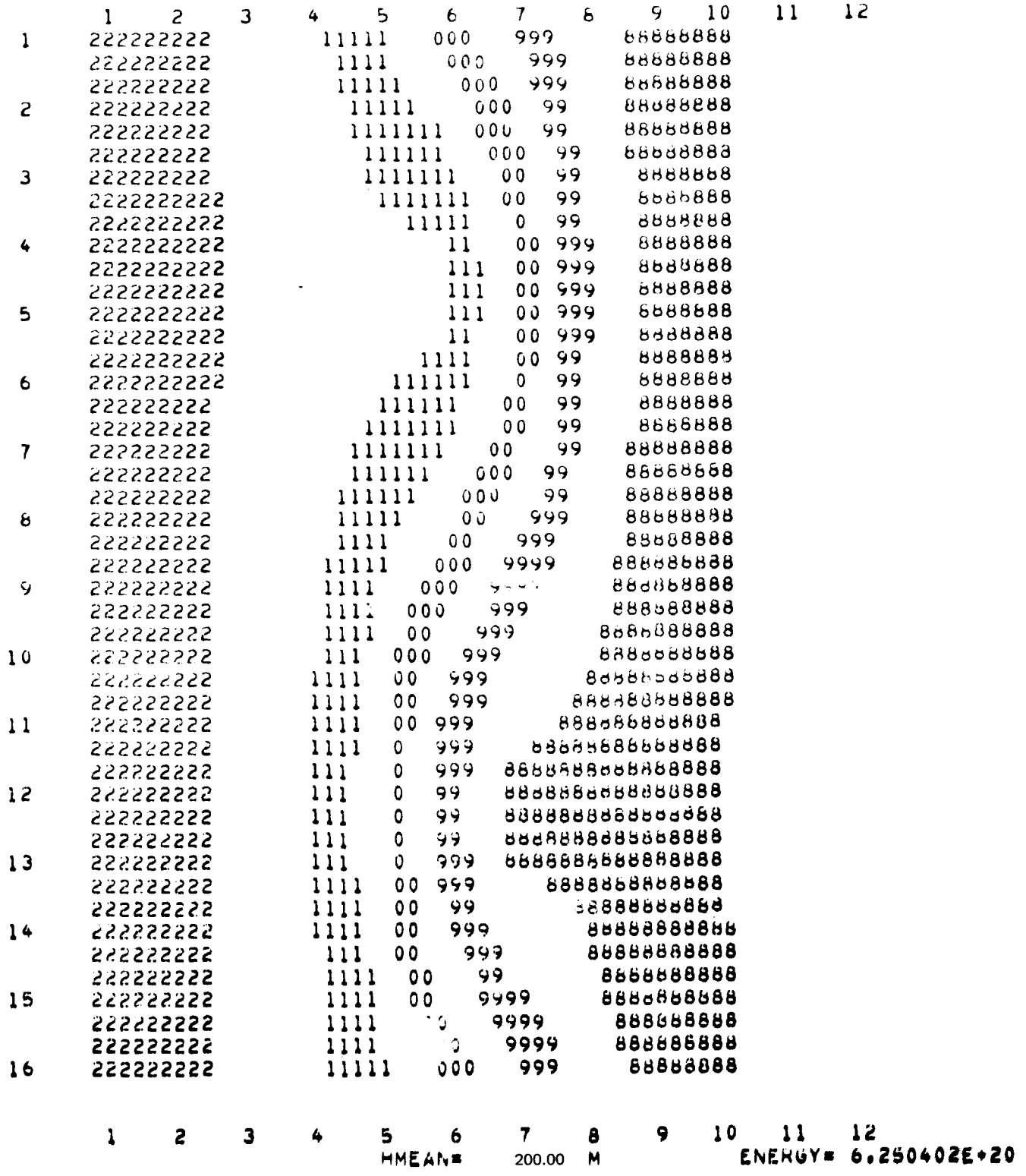

Figure 1. Initial height field contours (every 50 m). $\Delta x = \Delta y = 400$ km

i.e.

$$X_r = \beta_r,$$

$$k_{ri} << k_{rr} \cdot 10^{16} \qquad i = 1, 2, \ldots n \qquad i \neq k \qquad (58)$$

Assembly and numbering of nodes

The element matrices relevant to the Galerkin f.e.m. of the shallow-water equations are first set up according to equations (45)–(53) and then assembled into global matrices. Each triangle has local element (3×3) matrices which need to be scattered into global position in the global coefficient matrices. An element connectivity table is established which assigns the local (3×3) matrices to their place in the $(N \times N)$ global matrix. Whenever a term is assigned to a matrix location where another term has already been placed, it is added to whatever value is there.

This procedure is usually termed Boolean assembly. In order to proceed with the assembly, it is necessary to number the nodes and the elements. By labelling the

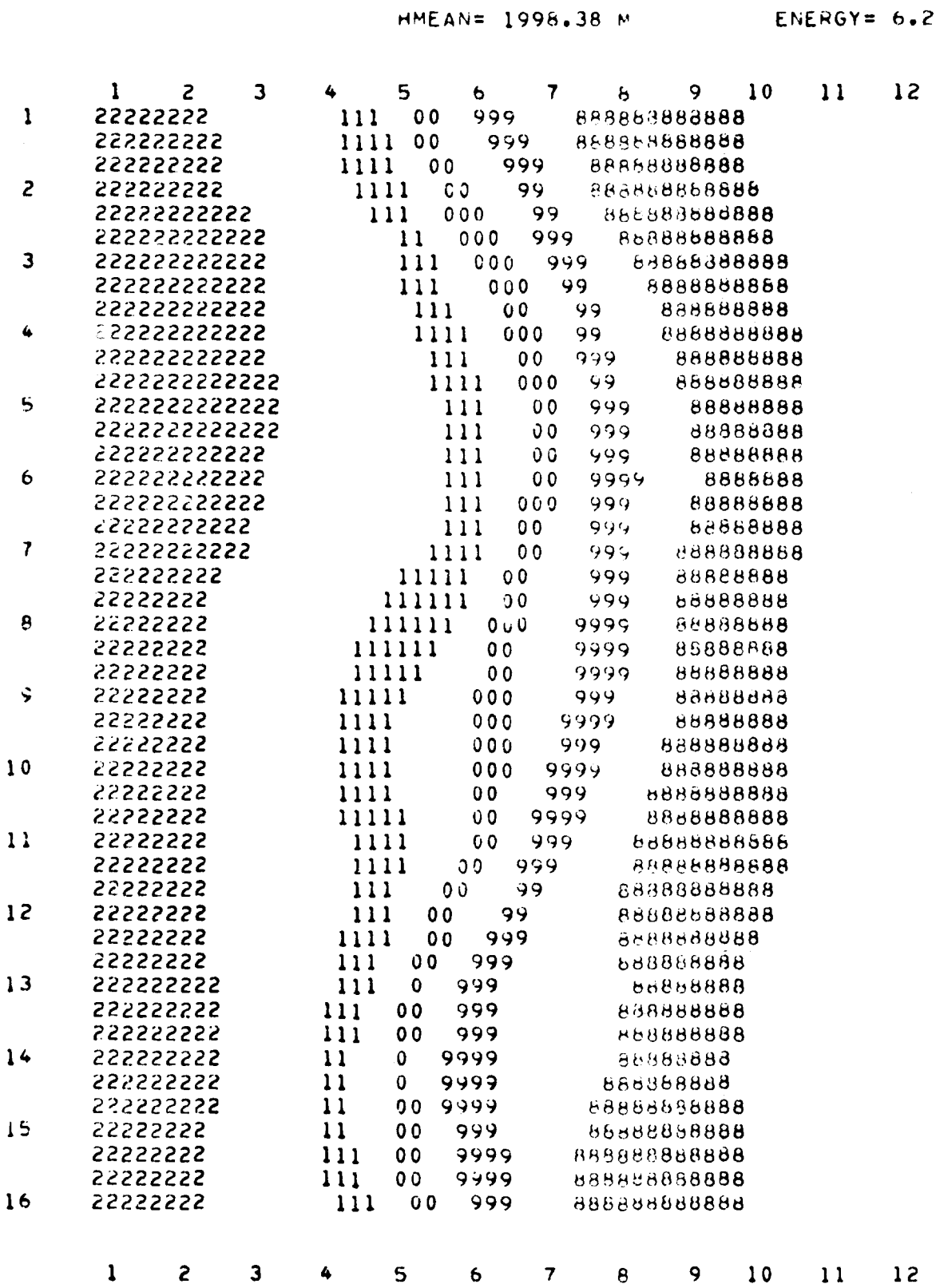

Figure 2. Height field contours after 2 days. $\Delta x = \Delta y = 400$ km, $\Delta t = 1800$ sec using Galerkin f.e.m.

nodes across the shortest dimension of the domain, it is possible partially to minimize the bandwidth of the global coefficient matrix. On the other hand, the numbering of the elements does not affect the computational aspects of the problem.

For sophisticated routines of node-ordering in order to obtain a small bandwidth, see George[18], Felippa and Clough[19] and Desai and Abel[14].

Compact storage scheme for sparse matrices

The global $(N \times N)$ coefficient matrix generated by the assembly process is very sparse, as the maximum number of triangles incident on one point of six. Therefore each row in the global $N \times N$ matrix has at most seven entries and it is an advantage to store the matrix in a compact manner to save fast-core storage. The overhead for reducing an $(N \times N)$ matrix to an $(N \times 7)$ matrix is the need

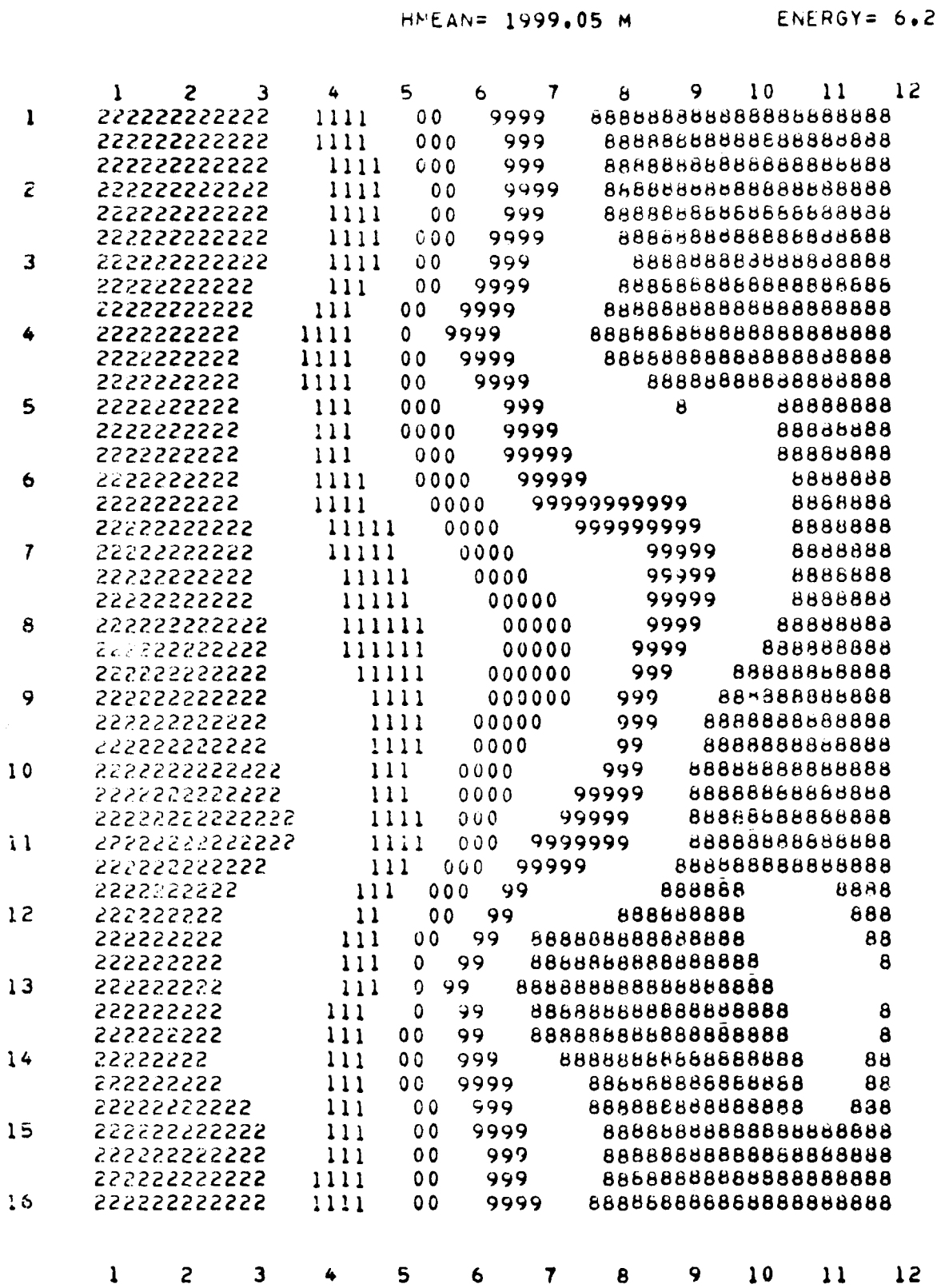

Figure 3. Height field contours after 10 days. $\Delta x = \Delta y = 400$ km, $\Delta t = 1800$ sec using Galerkin f.e.m.

for a correlation matrix also of size $N \times 7$, to tell us the seven nodes involved in any one row of the coefficient matrix. The correlation matrix is set up by searching the global correspondence table for the six triangles containing the node i. The six triangles found are then compared and sorted to produce the seven nodes interacting with node i (this includes node i itself). The seven nodes are then arranged in ascending order. Other efficient compact storage schemes for sparse matrices have been proposed by George[18] and Duff and Reid[20].

Solution of the linear system of equations

For solving the system of linear equations, the method adopted in this program was the iterative one of Gauss–Seidel[21,22] which has the virtue of simplicity.

If a direct solution is preferred, use can be made of the

frontal technique program of Irons[23] as documented by Hinton and Owen[24].

User-supplied S.O.R., ADI or conjugate–gradient methods can be used if a reduction of the computing time is desired.

COMPUTER IMPLEMENTATION

Description of main program

Main program FESW. The main program FESW reads the only data card of the program and after some preliminary calculations calls the subroutines NUMBER, CORRES, INCOND and AREAA.

These subroutines set up the triangular elements, number the nodes, determine the non-zero entries of the global matrix, calculate the initial fields and finally calculate the derivatives of the shape functions.

Next those of the element matrices which remain invariant during the simulation period are calculated by calling subroutine ASSEM. After finding the boundary nodes the program enters the main do-loop, which is executed once for every new time step. In this loop the simulation time is adjusted and then the subroutines ASSEM and MAMULT set up and assemble the different global matrices. The global matrices are then added following equations (32), (36) and (38).

Subroutine SOLVER is called upon to solve the linear systems of equations thus obtained, first for the continuity equation and then for the u and v momentum equations.

The new field values are updated as soon as obtained, and used in solving the coupled shallow-water equations system. After a predetermined number of time steps subroutine OUT is called to print out the height and velocity fields. Subroutine OUT in turn calls the output subroutines LOOK and MAPPA to calculate and print-out the total energy and the mean height invariants and to obtain a line printer plot of the height field contours.

Input specifications. The input to the program consists of only one data card, as follows: CARD 1: FORMAT (F5.0, 5I5) which contains the following six parameters:

DT	the time step in seconds;
NLIMIT	total number of time steps;
MF	a parameter controlling output operations of the program, i.e. specifying that after MF time steps subroutine OUT is to be called;
NOUTU	a parameter taking the value 0 or any integer $\neq 0$; IF NOUTU = 0, the U field is not printed; IF NOUTU$\neq$0, the U field is printed by the subroutine OUT;
NOUTV	a parameter taking the value 0 or any integer $\neq 0$; IF NOUTV = 0, the V field is not printed; IF NOUTV$\neq$0, the V field is printed by subroutine OUT;
NPRINT	a parameter taking the value 0 or any integer $\neq 0$; IF NPRINT$\neq$0, the global nodal numbers of each element, the indices of all non-zero entries of the global matrix as well as the coordinates of all the nodes are printed in subroutine NUMBER; IF NPRINT = 0, none of the above-mentioned items is printed.

Output specifications. The constants used in setting up the initial fields as well as the time step and the space dimensions are printed by the main program FESW.

The initial height field and velocity fields are printed out in subroutine INCOND.

Examples of output

Examples of FESW output are provided so as to demonstrate the different options of the program. The initial height field using a space resolution of $\Delta z = \Delta y = 400$ km is shown in Fig. 1, while Fig. 2 shows the height field contours after two days of simulation using a time step of 1800 s, and Fig. 3 shows the height field contours for a ten-day long-term integration using the same time step and spatial resolution.

REFERENCES

1 Baker, A. J. Finite element solution algorithm for viscous incompressible fluid dynamics, *Int. J. Num. Meth. Eng.*, 1973, **6**, 89

2 Baker, A. J. A finite element solution algorithm for the Navier–Stokes equations, *NASA Report CR*-2391, 1974, 73 pp.

3 Brebbia, C. A. and Partridge, P. W. Simulation of water circulation in the North Sea, *Appl. Math. Modelling*, 1976, **1**, 101

4 Connor, J. J. and Brebbia, C. A. *Finite Element Techniques for Fluid-flow*, Newnes-Butterworths, London, 1976, 310 pp.

5 Cullen, M. J. P. A simple finite element method for meteorological problems, *J. Inst. Math. Appl.* 1973, **11**, 15

6 Cullen, M. J. P. A finite element method for a nonlinear initial value problem, *J. Inst. Math. Appl.* 1974, **13**, 233

7 Cullen, M. J. P. Application of the finite element method to numerical weather prediction, *Ph.D. Thesis*, (University of Reading) 1975

8 Cullen, M. J. P. On the use of artificial smoothing in Galerkin and finite difference solutions of the primitive equations, *J. R. Meteorol Soc.* 1976, **103**, 77

9 Smith, S. L. and Brebbia, C. A. Finite element solution of Navier–Stokes equations for transient two-dimensional incompressible flow, *J. Comp. Phys.* 1975, **17**, (3), 235

10 Wang, H. H., Halpern, P., Douglas, J. Jr. and Dupont, T. Numerical solutions of the one-dimensional primitive equations using Galerkin approximations with localized basis functions, *Mon. Weather Rev.* 1972, **100**, 738

11 Hinsman, D. E. Application of a finite element method to the barotropic primitive equations, *M.Sc. Thesis*, (Naval Postgraduate School, Monterey) 1975

12 Grammeltvedt, A. A survey of finite difference schemes for the primitive equations for a barotropic fluid, *Mon. Weather Rev.* 1969, **97**, 384

13 Douglas, J. and Dupont, T. Galerkin methods for parabolic problems, *SIAM J. Num. Analysis*, 1970, **7**, 575

14 Desai, C. S. and Abel, J. F. *Introduction to the Finite-Element Method*, Van Nostrand/Reinhold, New York, 1972, 477 pp.

15 Eisenberg, M. A. and Malvern, L. E., On finite element integration in natural coordinates, *Int. J. Num. Meth. Eng.* 1973, **7**, 574

16 Payne, N. A. and Irons, B. M. personal communication with O. Zienkiewicz, 1963

17 Huebner, K. H. *The Finite Element Method for Engineers*, John Wiley, New York, 1975, 500 pp.

18 George, J. A. Computer implementation of the finite element method, *Ph.D. Thesis*, (Stanford University), 1971, 222 pp.

19 Felippa, C. and Clough, R. W. The finite element method in solid mechanics, in *Numerical Solution of Field Problems in Continuum Mechanics, SIAM-AMS Proc.*, A.M.S., Providence, (G. Birkhoff and R. S. Varga eds), 1970

20 Duff, I. S. and Reid, J. K. *Some design features of a sparse matrix code*, Computer Science and System Division CSS48, Harwell, 1977, 31 pp.

21 Dahlquist, G. and Björk, A. *Numerical Methods in Automatic Computation*, Prentice-Hall, Englewood Cliffs, 1969

22 Ralston, A. *A First Course in Numerical Analysis*, McGraw-Hill, New York, 1965
23 Irons, B. M. A frontal solution program, *Int. J. Num. Meth. Eng.* 1970, **2**, 5
24 Hinton, E. H. and Owen, D. R. J. *Finite Element Programming* Academic Press, London, 1977, 305 pp.

COMPUTER PROGRAM

Copies of the program listing may be obtained from C. M. L. Publications upon application. A small charge for duplication and post and packing will be made.