INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids **31**: 285–297 (1999)

ADAPTIVITY IN SPACE AND TIME FOR SHALLOW WATER EQUATIONS

M. MORANDI CECCHI* AND F. MARCUZZI[1]

Dipartimento Matematica Pura ed Applicata, Università di Padova, Via Belzoni 7, 35100 Padova, Italy

SUMMARY

In this paper, adaptive algorithms for time and space discretizations are added to an existing solution method previously applied to the Venice Lagoon Tidal Circulation problem. An analysis of the interactions between space and time discretizations adaptation algorithms is presented. In particular, it turns out that both error estimations in space and time must be present for maintaining the adaptation efficiency. Several advantages, for adaptivity and for time decoupling of the equations, offered by the operator-splitting adopted for shallow water equations solution are presented. Copyright © 1999 John Wiley & Sons, Ltd.

KEY WORDS: shallow waters; *a posteriori* error estimates; mesh adaptation; time step control

1. INTRODUCTION

A numerical method for solving the shallow water equations applied to the Venice Lagoon was developed earlier [1–3]. Here, adaptivity [4,5] for space and time discretizations and algorithm decoupling in time are introduced. The simplifications made at the model formulation stage and the way of splitting the operator have revealed some useful properties for this purpose.

The first of the two systems of equations that represents the split operator (Section 2) is solved explicitly in time and does not contain derivatives in space, i.e. it corresponds to a system of uncoupled differential equations. This fact makes it easy to perform the stability analysis and to calculate the critical time step above, which makes the system unstable. Actually, the system is time varying, so it changes its dynamics at each simulation step and hence the critical time step value. This suggests the need of adaptively changing the time step used for the simulation and for this purpose, *a posteriori* error estimates for the time discretization error are developed. The splitting and the Taylor series expansion method used for discretization in time [6] allow the estimates to be calculated easily.

The second system of equations also contains derivatives in space and *a posteriori* estimates for the space discretization error are simply obtained by computing two expressions for the gradient of the solution having different orders of approximation. This estimation method was developed for structural mechanics problems [7] and is now adapted and applied to time-dependent fluids.

* Correspondence to: Dipartimento Matematica Pura ed Applicata, Università di Padova, Via Belzoni 7, 35100 Padova, Italy. E-mail: mcecchi@math.unipd.it
[1] E-mail: marcuzzi@math.unipd.it

CCC 0271–2091/99/170285–13$17.50

The adaptive algorithm performs both mesh refinement and unrefinement. The mesh unrefinement problem is solved by eliminating nodes selected by *a posteriori* error estimates and remeshing the little polygons created, without adding interior nodes. This procedure works on completely unstructured grids, it is easy to implement and has the advantages that it does not alter (eventually improves locally) the quality of the mesh and allows the area of unrefined triangles be controlled (a necessary condition for obtaining a good mesh balancing).

As is well-known in all numerical methods that solve time-dependent problems governed by partial differential equations, there is a relationship between space–time discretizations granularity and properties of the computed solution. The most important properties are the convergence of the computed solution to the true one and the order of convergence as finer discretizations are used. This relation depends in general on the equations involved in the problem at hand and also on the methods used for discretization in space (e.g. the finite element space adopted) and in time. For example, it will be seen in the problem of shallow water solved with a particular splitting method, that the approximation of water elevation is more sensible to space discretization and the approximation of water flow velocity is more dependent on time discretization. In principle, if stability is of concern (i.e. the method is not unconditionally stable), a bound can be obtained of the kind $\Delta t \leq f(h)$, where Δt is the time step and h is the mesh size. Moreover, if the method is consistent for the approximation error, a bound can be obtained in which the dependence on h is explicitly formulated (see [8]), revealing that the error vanishes as h approaches zero with a certain rate. These bounds are useful indicators for choosing the optimal sizes for the time and space discretizations. Using finer discretizations than necessary corresponds to a waste of computing resources. The adaptive techniques arise with the aim of reaching automatically the optimal discretizations and, in principle, they are efficient because they act locally and so build the discretizations taking into consideration the various approximation problems along space and time dimensions. Their efficiency level depends mainly on the amount of extra computations required (with respect to the mere computation of the solution), and on their ability to approach the optimal Δt and h.

It will be seen in the sequel that adaptation can be corrupted by external phenomena. For time-dependent problems, the discretization error in time can alter the *a posteriori* error estimates in space, and hence produce a non-optimal refinement of the mesh, while the discretization error in space can alter the *a posteriori* error estimates in time, thereby making the adaptive algorithm choose a non-optimal Δt. A solution to this problem is proposed in Section 5. In Section 6, it is observed that critical time steps for the first system of equations reveal quite different values. Thanks to the independence of equations resulting from the splitting, this system can be solved in parallel without data movements between processors, and each with its appropriate time step (i.e. avoiding the execution of all of them at the minimum time step allowed).

2. AN ALGORITHM FOR SHALLOW WATER EQUATIONS

In a previous paper [1], an algorithm for solving the shallow water equations was presented. Here we will sketch it briefly and make some considerations. The solution of the shallow water problem applied to the Venice Lagoon led to the following semi-linear system:

$$\frac{\partial \eta}{\partial t} + \frac{\partial H u_1}{\partial x_1} + \frac{\partial H u_2}{\partial x_2} = 0,$$

$$\frac{\partial u_1}{\partial t} + g\frac{\partial \eta}{\partial x_1} = fu_2 - \frac{g\sqrt{u_1^2+u_2^2}}{C^2 H}u_1, \tag{2.1}$$
$$\frac{\partial u_2}{\partial t} + g\frac{\partial \eta}{\partial x_2} = fu_1 - \frac{g\sqrt{u_1^2+u_2^2}}{C^2 H}u_2,$$

where, having performed the integration of the variables along the vertical dimension,

$\eta = \eta(x_1, x_2) =$ the height of the free surface,

$u_1 = u_1(x_1, x_2) =$ the velocity component along the x_1-direction,

$u_2 = u_2(x_1, x_2) =$ the velocity component along the x_2-direction.

Equation set (2.1) can be written, in vector form, as

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}_1}{\partial x_1} + \frac{\partial \mathbf{F}_2}{\partial x_2} = \mathbf{R}_s, \tag{2.2}$$

where

$$\mathbf{U} = [\eta, u_1, u_2]^T, \quad \mathbf{F}_1 = [Hu_1, g\eta, 0]^T, \quad \mathbf{F}_2 = [Hu_1, 0, g\eta]^T$$

and

$$R_s = \left[0,\ fu_2 - \frac{g|u|u_1}{C^2 H},\ -fu_1 - \frac{g|u|u_2}{C^2 H}\right]^T.$$

By taking $\mathbf{F}_i = \mathbf{F}_i^* + \mathbf{F}_i^{**}$ $(i = 1, 2)$, $\mathbf{U} = \mathbf{U}^* + \mathbf{U}^{**}$, with $\mathbf{F}_i^* = 0$, $\mathbf{F}_i^{**} = \mathbf{F}_i$, system (2.2) can be split into

$$\frac{\partial \mathbf{U}^*}{\partial t} = \mathbf{R}_s, \tag{2.3}$$

$$\frac{\partial \mathbf{U}^{**}}{\partial t} + \frac{\partial \mathbf{F}_1^{**}}{\partial x_1} + \frac{\partial \mathbf{F}_2^{**}}{\partial x_2} = \mathbf{0}. \tag{2.4}$$

2.1. Solution of Equation (2.3)

Now, thanks to the particular operator-splitting adopted, Equation (2.3) does not contain derivatives in space and so must be fulfilled at each point of the domain independently; in particular, we will evaluate it at nodal points of the mesh used for the solution of (2.4). This corresponds to saying that discretizing (2.3) in space creates a diagonal system of equations. After performing the discretization in time, the independence of the equations also simplifies considerably the stability analysis, because this can be performed for each equation separately; for this reason an explicit formula for the eigenvalues ρ can be obtained and the condition $|\rho| < 1$ easily checked. Moreover, the discretization in time can be easily evaluated by adopting a second-order Taylor series expansion [6] as follows:

$$\mathbf{U}^{*(n+1)} = \mathbf{U}^{*(n)} + \Delta t\left(\frac{\partial \mathbf{U}^*}{\partial t}\right)^{(n)} + \frac{\Delta t^2}{2}\left(\frac{\partial^2 \mathbf{U}^*}{\partial t^2}\right)^{(n)}. \tag{2.5}$$

Since

$$\frac{\partial \mathbf{U}^*}{\partial t} = \mathbf{R}_s \text{ and } \frac{\partial^2 \mathbf{U}^*}{\partial t^2} = \frac{\partial \mathbf{R}_s}{\partial t} = \frac{\partial \mathbf{R}_s}{\partial \mathbf{U}^*}\frac{\partial \mathbf{U}^*}{\partial t},$$

the system (2.5) can be rewritten as

$$\mathbf{U}^{*(n+1)} = \mathbf{U}^{*(n)} + \Delta t(\mathbf{R}_s)^{(n)} + \frac{\Delta t^2}{2}(\mathbf{G}\mathbf{R}_s)^{(n)}, \tag{2.6}$$

where

$$\mathbf{G} = \frac{\partial \mathbf{R}_s}{\partial \mathbf{U}^*},$$

and solved doing straightforward calculations for evaluating $(\mathbf{R}_s)^{(n)}$ and $(\mathbf{G}\mathbf{R}_s)^{(n)}$. This discretization will be advantageous for obtaining *a posteriori* error estimates for the time discretization error.

Taking stability of the solution as time goes to infinity (A-stability) into consideration, the lack of derivatives in space make expressions of the kind $\Delta t \leq f(h)$ senseless, while other punctual constraints arise; in particular, the following eigenvalue analysis indicates the relation between A-stability and friction with the bottom (Chezy's force [2]). At each point where we evaluate Equation (2.3), we must solve iteratively the following system:

$$\begin{pmatrix} \eta^{(n+1)} \\ u_1^{(n+1)} \\ u_2^{(n+1)} \end{pmatrix} = J^* \begin{pmatrix} \eta^{(n)} \\ u_1^{(n)} \\ u_2^{(n)} \end{pmatrix}.$$

In Reference [9] an explicit formula for the eigenvalues ρ of the matrix J^* and the expression for the critical time step value up to which $|\rho| < 1$, are given. Hence,

$$\tau_c = \frac{b}{3a} - \frac{2^{1/3}(-b^2+3ac)}{3aq^{1/3}} + \frac{q^{1/3}}{2^{1/3}3a},$$

where

$$q = 2b^3 - 9abc + 27a^2d + \sqrt{4(-b^2+3ac)^3} + (2b^3 - 9abc + 27a^2d^2),$$

$$a = (K_0^2 + D^2)^2, \quad b = 4D(K_0^2 + D^2), \quad c = 8D^2, \quad d = 8D.$$

2.2. Solution of Equation (2.4)

For the discretization of Equation (2.4) it is convenient to write it in the extended form

$$\begin{cases} \dfrac{\partial}{\partial t}(\eta^{**}) + \dfrac{\partial}{\partial x_1}(Hu_1) + \dfrac{\partial}{\partial x_1}(Hu_2) = 0 \\ \dfrac{\partial}{\partial t}(u_1^{**}) + \dfrac{\partial}{\partial x_1}(g\eta) = 0 \\ \dfrac{\partial}{\partial t}(u_2^{**}) + \dfrac{\partial}{\partial x_2}(g\eta) = 0 \end{cases}. \tag{2.7}$$

This is discretized in time according to the θ method proposed in [10], giving

$$\begin{cases} (\Delta\eta^{**})^{(n+1)} + \Delta t\left[\dfrac{\partial}{\partial x_1}(Hu_1^{(n+\theta_1)}) + \dfrac{\partial}{\partial x_2}(Hu_2^{(n+\theta_1)})\right] = 0 \\ (\Delta u_1^{**})^{(n+1)} + \Delta t\dfrac{\partial}{\partial x_1} p^{(n+\theta_2)} = 0 \\ (\Delta u_2^{**})^{(n+1)} + \Delta t\dfrac{\partial}{\partial x_2} p^{(n+\theta_2)} = 0 \end{cases}, \tag{2.8}$$

where $p = g\eta$, θ_1 and θ_2 are real parameters in $[0, 1]$, $(\Delta\eta^{**})^{(n+1)} = \eta^{**(n+1)} - \eta^{**(n)}$, $(\Delta u_i^{**})^{(n+1)} = u_i^{**(n+1)} - u_i^{**(n)}$ and

$$\begin{cases} u_i^{(n+\theta_1)} = u_i^{(n)} + \theta_1[(\Delta u_i^{*})^{(n+1)} + (\Delta u_i^{**})^{(n+1)}], \quad i = 1, 2 \\ p^{(n+\theta_2)} = p^{(n)} + \theta_2(\Delta p^{**})^{(n+1)} \end{cases}. \tag{2.9}$$

After some manipulation and discretizing in space using linear triangular finite elements, we have

$$\left(\frac{1}{g}\mathbf{M} + \Delta t^2\theta_1\theta_2\mathbf{S}\right)(\Delta p^{**})^{(n+1)} = -\Delta t\left\{\sum_{i=1}^{2}\mathbf{Q}_i[H(u_1^{(n)} + \theta_1(\Delta u_i^{*})^{(n+1)})] + \Delta t\theta_1\mathbf{S}p^{(n)}\right\}, \tag{2.10}$$

$$\mathbf{S} = \sum_{i=1}^{2}\left(\int_\Omega \frac{\partial[\phi]}{\partial x_i} H \frac{\partial[\phi]^T}{\partial x_i}\,\mathrm{d}\Omega\right) \quad \text{and} \quad \mathbf{Q}_i = \int_\Omega [\phi]\frac{\partial[\phi]^T}{\partial x_i}\,\mathrm{d}\Omega, \tag{2.11}$$

where ϕ is the vector of shape functions at all nodes in the mesh. Once the increment in pressure has been evaluated from (2.10), it can be used for the calculation of $(\Delta u_i^{**})^{(n+1)}$:

$$\mathbf{M}(\Delta u_i^{**})^{(n+1)} = -\Delta t\mathbf{Q}_i[p^{(n)} + \theta_2(\Delta p^{**})^{(n+1)}], \quad i = 1, 2, \tag{2.12}$$

where $\mathbf{M}_i = \int_\Omega [\phi][\phi]^T\,\mathrm{d}\Omega$, in order to obtain the velocity subincrements $(\Delta u_1^{**})^{(n+1)}$ and $(\Delta u_2^{**})^{(n+1)}$.

The algorithm for evaluating $(\Delta U^{**})^{(n+1)}$ is unconditionally stable for suitable choices of θ_1 and θ_2 [10] and can run at higher time steps than the algorithm that computes $(\Delta U^{*})^{(n+1)}$. For this reason it is convenient to run them asynchronously with different time steps.

3. ADAPTIVITY IN SPACE DISCRETIZATION

The adaptivity in space is done through mesh refinement and unrefinement. These are performed where *a posteriori* error estimates are above (respectively below) a certain error tolerance.

A posteriori error estimates are obtained comparing two approximations for the first derivatives of the solution having different approximation orders. More precisely,

$N = [N_1, \ldots, N_n]$	nodal shape functions
$U = \begin{bmatrix} U_1 \\ \vdots \\ U_n \end{bmatrix}$	nodal computed solution functions
NU	computed solution function
$\hat{\sigma}_i = \dfrac{\partial N}{\partial x_i} U \quad (i = 1, 2)$	computed solution derivatives

A better approximation for the derivatives of the computed solution can be obtained using the same nodal functions used for the computation of the solution, and imposing the Galerkin orthogonality condition, for finding a continuous piecewise linear approximation σ^* of the solution gradient:

$$\int_\Omega N^T(\sigma_i^* - \hat{\sigma}_i)\,\mathrm{d}\Omega = 0 \quad (i = 1, 2),$$

that is

$$\int_\Omega \left[N^T N \sigma_i^* - N^T \frac{\partial N}{\partial x_i} U \right] \mathrm{d}\Omega = 0, \tag{3.1}$$

$$\left[\int_\Omega N^T N \,\mathrm{d}\Omega \right] \sigma_i^* = \left[\int_\Omega N^T \frac{\partial N}{\partial x_i} \,\mathrm{d}\Omega \right] U,$$

$$M\sigma_i^* = Q_i U.$$

Note that matrices M and Q were previously evaluated for computing the solution, cf. Section 2.2. Thus, the amount of work in computing σ_i^* is quite small. Defining

$$\epsilon_i = \sigma_i^* - \hat{\sigma}_i = N\sigma_i^* - \frac{\partial N}{\partial x_i} U \quad (i = 1, 2),$$

and the *a posteriori* error estimates can be expressed as the sum of each element contribution

$$\|\epsilon\|_\Omega^2 = \int_\Omega \epsilon^T \epsilon \,\mathrm{d}\Omega = \sum_{k=1}^{m} \left[\int_\Omega \epsilon^T \epsilon \,\mathrm{d}e_k \right] = \sum_{k=1}^{m} \|\epsilon\|_{e_k}^2,$$

where e_k is the kth element of the mesh.

This method of estimating the approximation error was introduced by Zienkiewicz and Zhu [7] for solving elasticity problems in structural mechanics. In recent years, its properties of efficiency and of equivalence to the error have been revealed with mathematical rigor [4,11].

Here we apply it to time-dependent fluid flow problems. The estimates on water flow approximation error are performed separately for velocity and for elevation. The error estimates are then used for deciding local mesh refinement (i.e. element subdivision) or coarsening. This process of mesh adaptation turns out to be effective for improving the approximation of water elevation and less useful for improving the approximation of water velocity (for which time discretization error plays a greater role).

The mesh adaptation algorithm uses *a posteriori* error estimates for obtaining a balanced mesh (i.e. each element gives the same contribution to the error) with the total error below a certain predetermined tolerance. It is a slight modification of one given in [4] for first balancing the mesh and then lowering the total error. Mesh adaptation is performed considering the error in water elevation in OR with the error in water velocity.

Mesh generation and refinement was done by performing Delaunay triangulation. This can be accomplished today reliably and fast thanks to existing algorithms that can be found on the Internet (see the Acknowledgement section).

The unrefinement is a little more delicate. Here nodes are eliminated whose adjacent triangles have error indicators below the desired tolerance, creating in this way 'polygons' that are successively triangulated (again Delaunay) without adding interior nodes. This procedure seems to have nice properties:

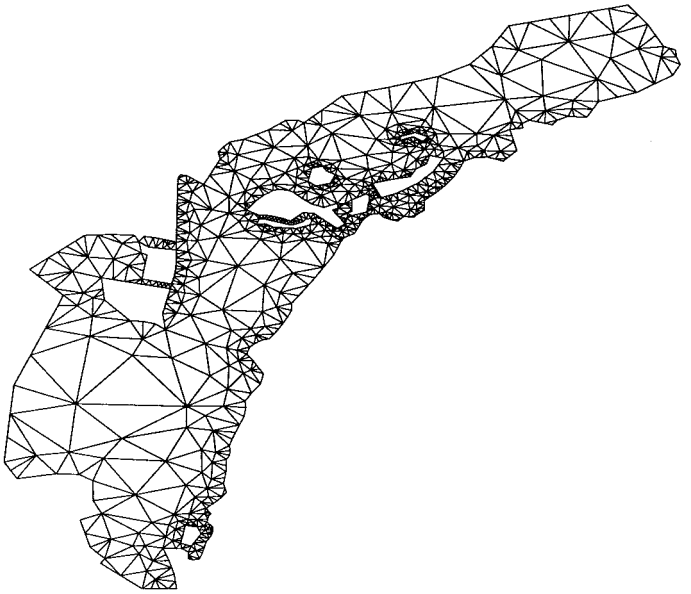

Figure 1. Automatically generated coarse mesh.

- the number of triangles eliminated is fixed (two every node cancelled); in this way we can control the areas of triangles also during unrefinement and this is what the adaptive algorithm requires to reach a 'balanced mesh';
- the unrefinement does not degrade mesh quality, and enhances it locally (see Figures 1–3).

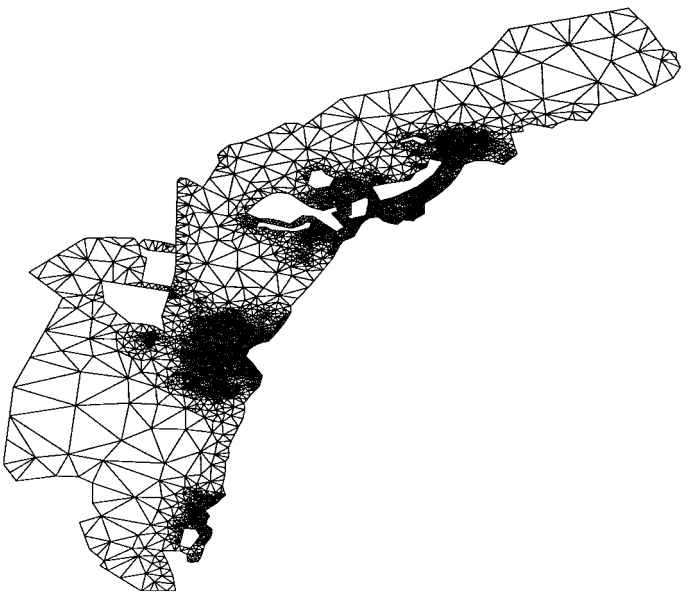

Figure 2. Adaptive refinement.

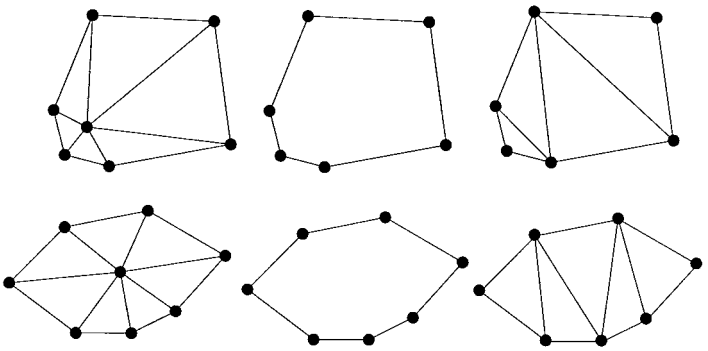

Figure 3. Two unrefinement examples.

4. ADAPTIVITY IN TIME DISCRETIZATION

4.1. Error estimates for $(\Delta U^*)^{(n+1)}$

Since for discretizing U^* we have used a Taylor series expansion, it seems natural to calculate the discretization error using the remainder in the Lagrange form; it holds that

$$(\Delta \mathbf{U}^*)^{(n+1)} = \Delta t \left(\frac{\partial \mathbf{U}^*}{\partial t}\right)^{(n)} + \frac{\Delta t^2}{2}\left(\frac{\partial^2 \mathbf{U}^*}{\partial t^2}\right)^{(n)} + \frac{\Delta t^3}{6}\left(\frac{\partial^3 \mathbf{U}^*}{\partial t^3}\right)^{(\xi)}. \tag{4.1}$$

Now, taking into account (2.6), we obtain the expression for the third derivative of U^*:

$$\left(\frac{\partial^3 \mathbf{U}^*}{\partial t^3}\right)^{(\xi)} = \frac{\partial}{\partial t}\left(\frac{\partial^2 \mathbf{U}^*}{\partial t^2}\right)^{(\xi)} = \frac{\partial}{\partial t}(\mathbf{GR}_s)^{(\xi)}, \tag{4.2}$$

where ξ is a point in $[t_n, t_{n+1}]$.

In this way we have obtained an expression for the third time derivative of U^* as a function of the first time derivative, namely the nodal acceleration. Hence, it is possible to argue the range of possible values of $U^{(\xi)}$, given for example, $U^{(n-1)}$, and take the one that produces the maximum value for the discretized version of the time derivative of $U^{(n)}$,

$$\begin{aligned}\arg \sup_{\hat{U}\in[U_{\mathrm{MIN}},U_{\mathrm{MAX}}]} \left\{\frac{(\mathbf{GR}_s)(\hat{U}) - (\mathbf{GR}_s)(U)^{(n-1)}}{2\Delta t}\right\} &= (\text{given } U^{(n-1)}) \\ &= \arg \sup_{\hat{U}\in[U_{\mathrm{MIN}},U_{\mathrm{MAX}}]} \{(\mathbf{GR}_s)(\hat{U})\},\end{aligned} \tag{4.3}$$

where U_{MIN} and U_{MAX} are determined having calculated $U^{(n-1)}$ and the maximum acceleration seen at the nodes or argued *a priori* based on physical considerations. The maximum value could be computed by putting the first derivatives of the bracketed expression equal to zero, but this would require solving a non-linear system of equations; actually, we are interested in the maximum over a small interval of values for U and this can be done in a simple way looking at the behaviour of the relatively smooth function $(\mathbf{GR}_s)(\hat{U})$. The efficiency of the estimator depends much on the choice of U_{MIN} and U_{MAX}; a too small interval would bring about underestimation of the error, while a too large interval would bring about overestimation of the error. The range of values indicated by U_{MIN} and U_{MAX} would, however, remain in a certain measure close to the optimal one (i.e. the range that would give the correct error estimate) since an unstable growth of the acceleration due to a too large Δt (i.e. underestimation of the error) would bring about a raise in the range (i.e. a potential increase of the error

estimates and consequently a reduction of the Δt), while an ever stable velocity value would bring about a reduction in the range (i.e. a potential decrease of the error estimates and consequently an increase of the Δt). Hence, the adaptive scheme is at least stable.

The time step can be adaptively modified with the simple rule: $\Delta t_{n+1} = q * \Delta t_n$, where $q \leq (\epsilon/\text{error}_{\text{estimated}})^{1/p}$ and ϵ is the predefined error tolerance, $\text{error}_{\text{estimated}}$ is the *a posteriori* time discretization error estimates derived above, and here $p = 2$ because the Taylor series expansion was of order two (see [12]).

We have obtained an expression for the so-called 'local error' (i.e. the error in one step). This serves us for adaptively changing the time step to maintain the local error below a predetermined tolerance. For purposes that will be explained in Section 5, we need also an estimate of the so-called 'global error', i.e. the total approximation error in the mesh nodes due to time discretization. For this purpose we must 'transport to current time all previous local error and add them up' (see [12], pp. 160–161). In this situation, this means finding for each mesh node k, two constants C_k and L_k, namely vector $L = [L_k]$ is an upper bound for the norm of $\mathbf{G} = \partial \mathbf{R}_s / \partial \mathbf{U}^*$ and can be obtained by evaluating at for each node the maximum velocity observed; $C = [C_k]$ is the vector of constants where $|e_k^{(n)}| \leq C_k * (h^{(n-1)})$ is satisfied at each mesh node, with $e_k^{(n)}$ the local error estimate obtained before. Then the following expression gives the global error for the node k:

$$|E_k^{(n)}| \leq h^p \frac{C_k}{L_k} (\exp(L_k(U_k^{(n)} - U_k^{(0)})) - 1).$$

4.2. Error estimates for $(\Delta U^{**})^{(n+1)}$

Consider Equation (2.12); a simple local error estimate for this series expansion is the following:

$$e_{u_i^{**}}^{(n+1)} = \frac{\theta_1^2}{2} [\Delta u_i^{*(n+1)} + \Delta u^{**(n+1)}_i - \Delta u_i^{*(n)} - \Delta u^{**(n)}_i] \quad (i = 1, 2),$$

$$e_{\eta^{**}}^{(n+1)} = \frac{\theta_2^2}{2} [\Delta \eta^{**(n+1)} - \Delta \eta^{**(n)}].$$

5. INTERACTIONS BETWEEN SPACE AND TIME DISCRETIZATIONS

When using a mesh adaptation algorithm for solving a time-dependent problem, one must consider the possibility that the discretization error in time may corrupt the *a posteriori* error estimates in space and the discretization error in space may corrupt the *a posteriori* error estimates in time. In fact, $U = u + e_s + e_t$, where e_s is the discretization error in space and e_t is the analogous in time, and the *a posteriori* error estimators work on U.

Note also that the stability properties as time increases are affected by the discretization errors in space and in time. Figure 4 shows the mean critical time step (as calculated in Section 2) for all mesh nodes in three situations; note that mesh adaptation allows higher stability limits than the initial mesh. This can also be seen in the equations of the stability analysis: the critical time step results to be inversely proportional to $D = |U|/(k_1^2 H)$, where k_1 is Chezy's constant, g is the gravity, $|U|$ is the modulus of the computed velocity (i.e. comprises the approximation errors in time and in space) and H is the depth.

Numerical experiments indicate that two adaptive-in-space simulations with different Δt, after 3 h of simulation, yield the following results:

Δt (s)	Number of mesh nodes	$\|\sigma e_s\|$ for water elevation
10	1465	45
30	1736	38

From *a posteriori* error estimates (in space only), the simulation with $\Delta t = 30$ s appears to be more accurate than the other, but Figure 5 shows that actually the adaptive-in-space simulation with $\Delta t = 30$ s is less accurate than that with $\Delta t = 10$ s. Hence, *a posteriori* error estimates in space are not reliable estimates of the total approximation error and we will see below that they are not reliable estimates of the space discretization error too. An intuitive explanation of this fact, confronted by numerical experiments, is that with increasing time steps, the spatial pattern of water elevation becomes rougher; hence, the *a posteriori* error estimators 'see' a simpler function to approximate than the true one, so they believe the approximation error is lower than it is. We could say that due to time discretization error, finer-scale phenomena are lost. *A posteriori* error estimates of the time discretization error can rescue this situation.

Now, it may be difficult to determine *a priori* if with the tolerances specified there will be relevant interaction between space and time adaptations as cited above.

For this reason it would be interesting to estimate the deviation produced in *a posteriori* time/space error indicators from errors in space/time. This is an object of ongoing work. For

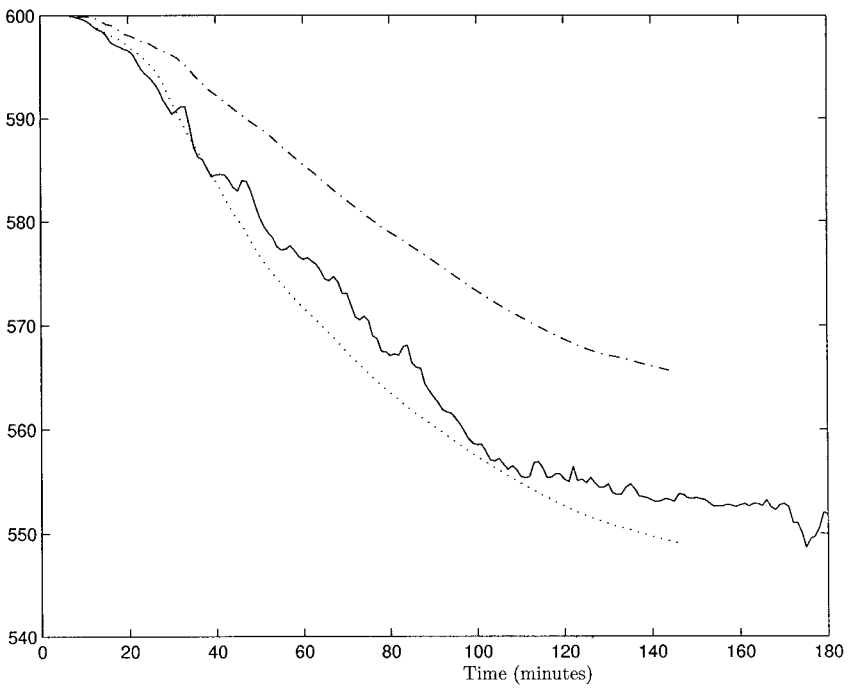

Figure 4. Mean critical time step with different space discretizations: adapted (continuous), coarse (dotted), uniformly refined (dash–dotted).

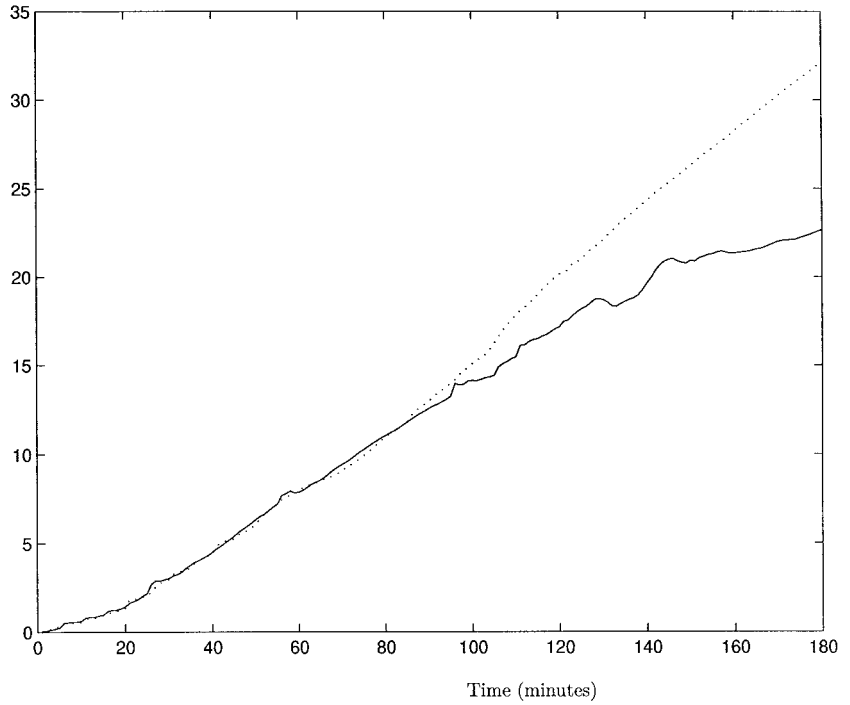

Figure 5. Total approximation error with different constant time steps: 10 s (continuous), 30 s (dotted).

eliminating the bad influence of time discretization error from *a posteriori* error estimates in space, we note that $U = u + e_s + e_t$, where e_s is the approximation error due to space discretization and e_t is the approximation error due to time discretization. An error estimate for e_t is obtained in Section 4. Let the Zienkiewicz–Zhu criterion (cf. Section 3) be applied to U to obtain *a posteriori* error estimates as done in Section 3.

$$\int_\Omega N^T(\sigma_U^* - \hat{\sigma}_U)\,\mathrm{d}\Omega = 0,$$

where now $\sigma_U = \sigma_{u+e_s} + \sigma_{e_t}$; hence

$$\int_\Omega N^T(\sigma_{u+e_s}^* + \sigma_{e_t}^* - \hat{\sigma}_{u+e_s} - \hat{\sigma}_{e_t})\,\mathrm{d}\Omega = \int_\Omega N^T(\sigma_{u+e_s}^* - \hat{\sigma}_{u+e_s})\,\mathrm{d}\Omega + \int_\Omega N^T(\sigma_{e_t}^* - \hat{\sigma}_{e_t})\,\mathrm{d}\Omega = 0,$$

that in matrix form, becomes $M\sigma_{u+e_s}^* = Q\hat{\sigma}_{u+e_s} + (Q\hat{\sigma}_{u+e_t} - M\sigma_{u+e_s}^*)$. Note that this expression is equivalent to that calculated in Section 3:

$$M\sigma_U^* = Q\hat{\sigma}_U.$$

If we choose $\sigma_{e_t}^*$ such that $M\sigma_{e_t}^* = Q\hat{\sigma}_{e_t}$, we obtain the *a posteriori* estimate of the discretization error in space, free from perturbations produced by the discretization error in time:

$$\|\sigma_{u+e_s}^* - \hat{\sigma}_{u+e_s}\| = \|(\sigma_U^* - \sigma_{e_t}^*) - (\hat{\sigma}_U - \hat{\sigma}_{e_t})\|.$$

The same can be formulated, with the necessary modifications, for eliminating the influence of discretization error in space on *a posteriori* error estimates of discretization error in time.

6. DOMAIN DECOMPOSITION

6.1. Computation of $(\Delta U^*)^{(n+1)}$

We have seen in Section 2 that Equation (2.3) represents a system of uncoupled differential equations. Numerical experiments on the Venice Lagoon data shows that the time steps to use for these equations lies in a quite large interval of values. Adopting an implementation that solves these equations, divided into groups according to the time step to be used for each one, in parallel and asyncronously, leads to a certain global execution time saving. As Equation (2.3) does not have derivatives in space, nodes corresponding to a group of equations do not have to be contiguous and there are no data movements between groups (i.e. between processors). This 'domain decomposition' strategy combined with time discretization error estimates allows the algorithm to work on different time scales in an adaptive way.

6.2. Computation of $(\Delta U^{**})^{(n+1)}$

Domain decomposition for Equation (2.4) was still considered in [1]. It is based on non-overlapping subdomains defined by choosing regions (not necessarily contiguous) of the lagoon at constant depth.

7. CONCLUDING REMARKS

Simplifications on the model based on physical assumptions and a suitable splitting of the operator before discretization, have shown the possibility of introducing adaptivity in space and in time using relatively simple mathematical tools.

With semi-discretization methods there are two shortcomings in doing adaptivity in space and in time. The first one is that, in space, the solution is computed globally in one step, and hence the error indicators of the solution are computed in one step. In time, the solution is built incrementally—there is not a global scheme for computing it—hence, the error indicators are built incrementally, which diminishes to some measure their effectiveness. The second is that two independent adaptive mechanisms control two discretizations that actually interact with each other. These problems could be overcome by adopting finite element discretizations in both space and time. This is the subject of current research.

ACKNOWLEDGMENTS

The authors express their appreciation for Triangle, the Delaunay triangulation library created by J.R. Shewchuk and available on the Internet at http://www.cs.cmu.edu/ ~ quake/triangle.html. This work has been carried out with the support of the M.U.R.S.T., Ministry of Research and Technology, project *Advanced Numerical Methods for Scientific Computing*, Scientific Research Programs of Relevant National Interest scheme, year 1998.

REFERENCES

1. M. Morandi Cecchi, A. Pica and E. Secco, 'A projection method for shallow water equations', *Int. J. Numer. Methods Fluids*, **27**, 81–95 (1998).
2. M. Morandi Cecchi and L. Salasnich, 'Shallow water theory and its application to the Venice Lagoon', *Comput. Methods Appl. Mech. Eng.*, **151**, 63–74 (1998).

3. O.C. Zienkiewicz, J. Wu and J. Peraire, 'A new semi-implicit or explicit algorithm for shallow water equations', *Math. Model. Sci. Comput.*, **1**, 31–49 (1993).
4. R. Verfurth, *A Review of A Posteriori Error Estimation and Adaptive Mesh-Refinement Techniques*, Wiley-Teubner, 1997.
5. K. Eriksson, D. Estep, P. Hansbo and C. Johnson, 'Introduction to adaptive methods for differential equations', *Acta Numer.*, 105–179 (1995).
6. J. Donea, 'A Taylor–Galerkin method for convective transport problems', *Int. J. Numer. Methods Eng.*, **20**, 101–119 (1984).
7. O.C. Zienkiewicz and J.Z. Zhu, 'A simple error estimator and adaptive procedure for practical engineering analysis', *Int. J. Numer. Methods Eng.*, **24**, 337–357 (1987).
8. T.J.R. Hughes, *The Finite Element Method*, Prentice-Hall, Englewood Cliff, NJ, 1987.
9. M. Morandi Cecchi and L. Salasnich, 'Convergence of the splitting method for shallow water equations', *AIICSR 97*, Smolenice, Slovakia, 1997, pp. 1–12.
10. V. Thomee, 'Finite difference methods for linear parabolic equations', in P.G. Ciarlet and J.L. Lions (eds.), *Handbook of Numerical Analysis*, North-Holland, Amsterdam, 1991, pp. 12–171.
11. R. Rodriguez, 'Some remarks on Zienkiewicz–Zhu estimator', *Numer. Methods PDE*, **10**, 625–635 (1994).
12. E. Hairer, S.P.N. Torsett and G. Wanner, *Solving Ordinary Differential Equations I*, Springer-Verlag, Berlin, 1987.

