INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids **31**: 497–512 (1999)

MESH ADAPTATION STRATEGIES FOR SHALLOW WATER FLOW

M. MARROCU[a,*] AND D. AMBROSI[b]

[a] *CRS4, Centro di Ricerca, Sviluppo e Studi Superiori in Sardegna, C.P. 94, 09010 Uta (CA), Italy*
[b] *Dipartimento di Matematica, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy*

SUMMARY

This paper shows how the mesh adaptation technique can be exploited for the numerical simulation of shallow water flow. The shallow water equations are numerically approximated by the Galerkin finite element method, using linear elements for the elevation field and quadratic elements for the unit width discharge field; the time advancing scheme is of a fractional step type. The standard mesh refinement technique is coupled with the numerical solver; movement and elimination of nodes of the initial triangulation is not allowed. Two error indicators are discussed and applied in the numerical examples. The conclusion focuses the relevant advantages obtained by applying this adaptive approach by considering specific test cases of steady and unsteady flows. Copyright © 1999 John Wiley & Sons, Ltd.

KEY WORDS: mesh adaptation; shallow water; finite element; error estimate

1. INTRODUCTION

The use of unstructured grids for the numerical approximation of partial differential equations of applied mathematics has two main attractions. The most common one is the geometrical flexibility, i.e. the capability of handling problems characterized by computational domains with complex boundaries that would be almost impossible to face by a structured approach. However, there is a second aspect to unstructured grids that has even more relevance: the possibility of refining the computational mesh, where needed, in order to minimize the computational error in some proper sense. Suitable indicators of the accuracy of the solution allow refinement of the mesh where the numerical error is large and coarsening where the error is small, in order to optimize the quality of the computed solution for a given computational effort.

Mesh adaptation techniques have been used for many years in several fields of computational fluid dynamics (CFD), but adaptivity in the framework of free surface hydrostatic flow has not yet been much explored. To the knowledge of the authors only a very recent paper by Walters and Barragy [1] compares and discusses the use of high-order polynomial basis (*p* adaptation) for the discretization of the shallow water equations versus local mesh refinement, where the order of the polynomial approximation is kept unchanged (*h* adaptation).

* Correspondence to: CRS4, Centro di Ricerca, Sviluppo e Studi Superiori in Sardegna, C.P. 94, 09010 Uta (CA), Italy. Fax: +39 70 2796216; e-mail: marino@crs4.it

CCC 0271–2091/99/180497–18$17.50

Received April 1998
Revised August 1998

This paper deals with local mesh adaptation strategies for shallow water flow. The numerical solver for the shallow water equations coupled with the mesh adaptation procedure is of fractional step type. It has been extensively tested and successfully used with static meshes in several applications [2].

The mesh adaptation technique described in this paper is based on the use of a background grid. The numerical simulation starts on a grid, possibly composed of only a few nodes, which is the coarsest mesh that may be used in the computation: its nodes are neither moved nor suppressed. Successive levels of refinement and coarsening lie on such a background grid. This technique has been adopted because of the very complicated geometry of the boundary that characterizes environmental applications in rivers, coastal areas and so on. By keeping a background grid, the information about the position of the boundaries and batimetry can be preserved and it is not lost by further interpolations due to node movements. Remarkably, an important by-product of the mesh adaptation is that little care has to be used in defining the initial grid: 'with adaptation, any initial grid will be transformed into a near optimal discretization' [3].

A peculiar aspect of the mesh adaptation is the necessity to devise proper indicators of the numerical error that should drive the grid refinement and de-refinement. In the present paper, two possibilities are proposed and investigated in some detail: the second-order derivative of the elevation field and the local mass conservation in a specific sense. The first of the error indicators has a mathematical basis and the second is suggested by numerical and physical reasons. The performance of these error estimators is discussed in the last section of the paper together with the numerical results.

2. THE SHALLOW WATER EQUATIONS AND THE NUMERICAL SCHEME

The shallow water equations in conservative differential form read

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla\cdot(\mathbf{q}\otimes\mathbf{q}/H) - \nabla\cdot(\mu\nabla\mathbf{q}) + gH\nabla\xi = -g\frac{\mathbf{q}|\mathbf{q}|}{H^2H^{1/3}K^2} - 2\Omega\times\mathbf{q}, \tag{1}$$

$$\frac{\partial \xi}{\partial t} + \nabla\cdot\mathbf{q} = 0, \tag{2}$$

where $\mathbf{q}(x, y, t) = (q_x, q_y)^T$ is the unit width discharge, $\xi(x, y, t)$ is the elevation over a reference plane, $H(x, y)$ is the total depth of the water, μ is the dispersion coefficient, g is the gravity acceleration, Ω is the angular velocity of the Earth and K is the Strickler coefficient.

The numerical scheme used to approximate the shallow water equations (1) and (2) is a finite element scheme of a fractional step type. In this section the numerical scheme is briefly described, more details can be found in Reference [2].

The main idea of the advancing-in-time scheme is to split the equations at every time step, in order to decouple the different physical contributions: convection, propagation, diffusion and bottom friction. The discretization in time of system (1)–(2) then reads:

Step 1

$$\mathbf{v}^n = \mathbf{q}^n/H^n, \qquad \mathbf{v}^{n+1/3} = \mathbf{v}^n \circ \mathbf{X}, \tag{3}$$

Step 2

$$\mathbf{q}^{n+1/3} = H^n \mathbf{v}^{n+1/3},$$

$$\mathbf{q}^{n+2/3} + \Delta t g \frac{\mathbf{q}^{n+2/3}|\mathbf{q}^{n+1/3}|}{H^2 H^{1/3} K^2} = \mathbf{q}^{n+1/3} + \Delta t[\nabla\cdot(\mu\nabla\mathbf{q}^{n+1/3}) - 2\Omega \times \mathbf{q}^{n+1/3}], \tag{4}$$

Step 3

$$\mathbf{q}^{n+1} - \mathbf{q}^{n+2/3} + \Delta t g H^n \nabla(\theta\xi^{n+1} + (1-\theta)\xi^n) - \frac{\mathbf{q}^{n+2/3}}{H^n}(\xi^{n+1} - \xi^n) = 0, \tag{5}$$

$$\xi^{n+1} - \xi^n + \Delta t \nabla\cdot(\theta\mathbf{q}^{n+1} + (1-\theta)\mathbf{q}^n) = 0. \tag{6}$$

The symbol $v^n \circ \mathbf{X}$ indicates the value of the velocity obtained by a Lagrangian integration along the pathline. The parameter θ ranges between 0.5 and 1.

At the third integration step, Equations (5) and (6) are decoupled by subtracting the divergence of (5) from (6). One then solves the following Helmholtz-type equation:

$$\xi^{n+1} - \theta^2(\Delta t)^2 g \nabla\cdot(H^n \nabla\xi^{n+1}) + \Delta t \nabla\cdot\left(\frac{\mathbf{q}^{n+2/3}}{H^n}\xi^{n+1}\right)$$

$$= \xi^n + \theta(1-\theta)(\Delta t)^2 g \nabla\cdot(H^n\nabla\xi^n) - \Delta t \nabla\cdot\mathbf{q}^{n+2/3} + \Delta t \nabla\cdot\left(\frac{\mathbf{q}^{n+2/3}}{H^n}\xi^n\right). \tag{7}$$

The new water elevation is finally used to update Equation (5).

The spatial discretization of Equations (4)–(6) is based on a standard Galerkin finite element method; the basic theory of the Galerkin approach may be found, for example, in Johnson [4] and Zienkiewikz and Taylor [5]. An important aspect of the spatial discretization is that two different spaces of representation have been used for the unknowns: the elevation is interpolated by $P1$ functions, whilst the unit width discharge is interpolated by $P2$ functions. As usual, $P1$ is the set of piecewise linear functions on triangles, $P2$ is the set of piecewise quadratic functions on triangles. The choice of these interpolation spaces eliminates the spurious oscillations that arise in the elevation field when a $P1$–$P1$ representation is used inside a scheme that involves an elliptic equation for it.

The main advantage of this fractional step procedure is that the wave travelling at speed $\sqrt{gh}$ is decoupled in the equations and treated implicitly. Therefore, the CFL condition owing to celerity is cheaply circumvented. The horizontal diffusion term in Equation (4) is typically of minor importance in many applications and often it can be discretized explicitly. However, when refining the computational mesh, the creation of small triangles in crucial zones often occurs, so that an implicit discretization of diffusion can be advantageous.

In the integration of the weak formulation of Equations (4) and (5) the lumping technique can be adopted for the mass matrices deriving from the discretization of the $\mathbf{q}$ terms. By the term mass lumping, the intention is the use of a low-order quadrature formula for the evaluation of the integrals involving the non-differential terms, yielding a diagonal stiffness mass matrix. It is well known that for $P2$ elements a non-trivial diagonalization has to be performed (contrary to the case of $P1$ elements), otherwise a singular matrix is recovered (see Appendix 8 of [5]). This difficulty has been overcome in the following way: each triangle of the mesh is split into four parts by connecting the midpoints of the sides. It is then possible to use the three vertex points rule on each subtriangle and the total integral is the sum of the subintegrals automatically leading to a diagonal mass matrix.

The computational effort required for the solution of the linear systems involved in the scheme (3)–(5) and (7), therefore, consists of the inversion of one symmetric matrix, whose size is equal to the number of triangle vertices.

2.1. Lagrangian scheme for the convective terms

At Step 1, the advective part of the momentum equation is integrated by a Lagrangian scheme [6,7]. Rewriting the convective terms of Equation (1) in Lagrangian form results in the solution of two coupled ordinary differential equations:

$$\frac{\mathrm{d}\mathbf{v}(\mathbf{X}(t), t)}{\mathrm{d}t} = 0, \tag{8}$$

$$\frac{\mathrm{d}\mathbf{X}}{\mathrm{d}t} = \mathbf{v}(\mathbf{X}(t), t). \tag{9}$$

The curve $\mathbf{X}(t)$ is the characteristic line and its slope is the velocity itself, so that at this stage it coincides with the pathline. The velocity field plays a double role: it is the unknown to be determined as well as the slope of the characteristic curve. As this study is interested in computing the solution at the mesh nodes, consider the node with co-ordinates $\mathbf{y}$. The initial condition associated with Equation (9) is

$$\mathbf{X}(t^{n+1}) = \mathbf{y}. \tag{10}$$

To integrate Equation (9) you need to know the slope of the characteristic curve at $\mathbf{y}$ at time t^{n+1}, which, unfortunately, is the unknown velocity itself. Therefore, the slope of the characteristic line has to be approximated in some way, for instance, by a zero-order extrapolation in time. Assuming the use of a second-order Runge–Kutta scheme to integrate Equation (9), the algorithm is as follows:

$$\bar{\mathbf{X}} = \mathbf{y} - \frac{\Delta t}{2}\mathbf{v}^n(\mathbf{y}), \tag{11}$$

$$\mathbf{X}(t^n) = \mathbf{y} - \Delta t\,\mathbf{v}^n(\bar{\mathbf{X}}), \tag{12}$$

and Equation (8) will immediately give

$$\mathbf{v}^{n+1/3}(\mathbf{y}) = \mathbf{v}(\mathbf{X}(t^{n+1}), t^{n+1}) = \mathbf{v}(\mathbf{X}(t^n), t^n). \tag{13}$$

The Lagrangian approach for convective terms as described above is not the weak Lagrangian scheme discussed in [6]; there is no transport of quadrature points because Equation (13) is written in strong form. This choice is consistent with the mass matrix lumping that has been used in the explicit parts of the scheme.

As the Lagrangian integration requires the primitive form of the equations, the fourth term that appears in the left-hand-side of (5) has been added to ensure consistency with Equation (1), which is written in conservation form. It should be noted that, apart from this term, the discrete counterpart of Equations (3)–(7) requires the inversion of symmetric matrices only. However, this consistency term is of minor relevance in all flows in which the typical time scale is much larger than the time step (as is the case of tides). Therefore, the usual conjugate gradient (CG) algorithm can be confidently used in this kind of simulations.

The Lagrangian discretization of the transport terms has many attractive features: it avoids spurious oscillations due to a centred treatment, without the inclusion of any unphysical viscosity coefficients and it eliminates any restriction on the time step. However, when using unstructured grids, the pathline reconstruction, which requires the knowledge of the element in which the foot of the pathline falls, yield a much greater algorithmic effort than using a structured grid. In practice, this difficulty has been overcome in the code by defining an ordered list containing all the elements that are adjacent to a node or to a given element. In

this way the search for the element in which the pathline foot falls is restricted to clusters of elements. To avoid that the foot of the pathline reconstructed falls outside of the domain, the rigid boundary of the domain is always assumed to be a streamline.

It is worthwhile to remark that the quadratic representation of velocities, which here has been adopted for compatibility reasons, ensures that the interpolation of the velocity at the foot of the pathline is not too dissipative.

3. ERROR ESTIMATE AND ERROR INDICATORS

The mesh adaptation technique requires some *a posteriori* estimate of the error of the numerical solution based on the computed solution itself: it is necessary to state how much the numerical solution differs locally, in a proper sense, from the exact solution of the differential problem. This section introduces two ways to determine the regions where the computational mesh should be refined or coarsened by two error indicators.

(1) For linear elliptic problems it is possible to estimate rigorously the numerical error in terms of the second derivative of the solution. Let u be the exact solution of the Poisson problem in weak form

$$(\nabla u, \nabla v) = (b, v), \tag{14}$$

for all v belonging to a suitable space and where (,) indicates the usual internal product in L^2, and let u_h be the solution of the discrete equation

$$(\nabla u_h, \nabla v_h) = (b, v_h), \tag{15}$$

where $\{v_h\}$ is a finite subset of test functions. Then, given a triangulation with maximum side length h, it can be shown [4] that the distance between the exact and the computed solution is bounded as follows:

$$\|\nabla(u - u_h)\|_{L^\infty} \leq Ch^2 \max_{\Omega} \max_{ij} \left| H \frac{\partial u}{\partial x_i \, \partial x_j} \right|. \tag{16}$$

where Ω is the computational domain. As the computational kernel of the numerical scheme described in Section 2 is an elliptic equation for the elevation of the water (7), a possible approach in the refining–coarsening stage is to use the estimate (16), where the right-hand-side has to be calculated using the computed solution u_h.

Representing the unknown ξ using the standard finite elements linear basis $\{\phi_k\}$, it can be written

$$\xi(\mathbf{x}) = \sum_k \phi_k(\mathbf{x}) \xi_k. \tag{17}$$

The error estimate (16) then suggests to define the following non-dimensional error indicator:

$$\epsilon_{1,m} = \frac{H_m}{\xi_m} \max_{ij} \left| \sum_k \xi_k \int_\Omega \frac{\partial \phi_k}{\partial x_i} \frac{\partial \phi_m}{\partial x_j} \, \mathrm{d}\Omega \right|, \tag{18}$$

where the water elevation is supposed to be never vanishing in the computational domain.

(2) An important feature that a numerical scheme for shallow water flow should possess is mass conservation. This property is accomplished by the scheme described above in a weak sense, as the discrete equations are obtained from the continuity equation (1), and are then consistent with it. However, the finite element scheme illustrated above does not ensure mass

conservation in a *finite volume cell centred* sense: the mass variation inside a triangle during a time step is not exactly equal to the flux through the edges of the triangle itself.

The reason for this is twofold. On one hand the use of quadratic polynomials to approximate the discharge **q** results in a larger computational stencil than the one involved when using a linear representation of the unknown. Mass conservation checking should be performed on a stencil consistent with the stencil of the analysed scheme. The mass conservation, triangle by triangle, can be properly advocated only for finite elements of mixed type, when a special mass lumping is used: only finite elements of mixed type RT0 just recover the cell centred finite volume technique [8].

Secondly, we observe that the substitution of the momentum equation into the continuity equation, which leads to Equation (7), involves a new spatial derivation. This is a common approach in the finite element context [9], but does not ensure local mass conservation. Conversely, in the finite difference finite volumes framework, such a substitution is usually carried out at an algebraic level: Equations (5) and (6) are first discretized in space and then the substitution is carried out, without further derivatives. This ensures local triangle-by-triangle mass conservation [10].

The considerations above suggest the use of checking the local mass balance as an error indicator for the present scheme. Thus, it can be defined

$$\epsilon_{2,e} = \left| \frac{1}{\Delta t} \int_e (\xi^n - \xi^{n-1}) \, \mathrm{d}\Omega + \oint_e \mathbf{q}^n \cdot \mathrm{d}\Gamma \right|, \tag{19}$$

where Γ is the contour of the eth element so that ϵ_2 is the mass defect in the eth element.

It should be mentioned that other possible error indicators outside of ϵ_1 and ϵ_2 have been considered and numerically investigated by the authors. However, other candidates (as second-order derivatives of the velocity) did not yield appreciable results in comparison with the ones described above.

4. THE MESH REFINEMENT TECHNIQUE

Any error estimator among the ones described in the section above allows identification of a set of elements of the mesh to be refined (or coarsened). Several techniques can then be used towards this aim [3].

1. *Repositioning of the mesh* (*r-methods*). Local refinement of the mesh is obtained by moving its nodes. Of course, the local refinement generates coarsening in the remaining part of the domain. Since the topology of the mesh does not change during repositioning, this strategy is easy and cheap to implement: the connectivity of the grid is unchanged. Nevertheless, it is a strategy not often used because of the constrain of using a fixed number of elements may lead to unacceptable deformation of the grid.

2. *Enrichment of the mesh* (*h-methods*). Grid triangles are split into triangles of lower average side length h. As the error of the numerical solution behaves like ch^{λ}, with c and λ constants depending on the discretization scheme, these methods if used in a proper way ensure convergence of order λ. For this reason h-methods are more popular, although their implementation is more complex than repositioning methods as, at every subdivision, the topology of the mesh changes. The splitting of the elements can be typically performed in two ways:

(i) edge bisection: the midpoint of an edge to be refined is connected to the vertex opposite to the edge itself.
(ii) standard refinement: a marked triangle (*father*) is subdivided into four similar triangles (*sons*) joining the midpoints of the edges. The number of edges and triangles increases by a factor of four, the local length of triangles is halved.

3. *Re-meshing* (*m-methods*). To produce a higher quality triangulation, creation, destruction and repositioning of the nodes is allowed. This is the most general procedure but also the heaviest from a computational point of view.

The choice of the most suitable mesh adaptation procedure depends on the problem at hand. For example, regularity of the element size and shape can be a requirement or not, depending on the characteristics of the flow. In compressible flow simulations, very stretched elements are useful because where there are shocks we have lines (surfaces) of discontinuity in the flow field. Adapted grids for these problems are in general very irregular, both in side length and in shape [Habashi *et al.*, 'Certifiable CFD through mesh optimization', submitted to *AIAA J.*, 1–22 (1996)] and the more appropriate strategy for these kind of problems seems to be the *h*-method 2(i).

In the present paper, the *h*-method 2(ii) is applied to shallow water flow simulations. The main reason for this choice is that the *h*-method keeps the information on the original grid unchanged. When a new node is added in the middle of an edge, batimetry, velocity and water elevation in it are obtained by interpolating the values in the element. In this way the original values of the batimetry and of the physical unknowns in the initial mesh are never abandoned and the degradation of the solution by interpolation is minimized. The computations are started by using quite a coarse mesh, as regular as possible; then refinements and de-refinements are accomplished in such a way as to preserve regularity.

Figure 1 shows the subdivision technique known as *standard* or *regular* or *red* refinement. The elements to be refined are first split in a standard way. The surrounding triangles have then one, two or three subdivided edges that are not connected to nodes belonging to the original triangle. The nodes in the midpoint of the edges of the elements not yet refined are called *hanging nodes*. Elements having *hanging nodes* must then be refined in a proper way to ensure consistency of the triangulation (*green refinement*, see Figure 1). For this aim there are only a few possible strategies; we have chosen the following one, described in a C-like form:

```
for (i=0; i<NEL; i++)
  if (Err(i)>thresh) StandRef(i);
for (i=0; i<NEL; i++)
  if ((NHang(i)>1)||(NHang(i)==1 && EType(i)==green)) StandRef(i);
if StandRef has been called at least one time in the last block then it
is re-executed;
for (i=0; i<NEL; i++)
  if (NHang(i)==1) MakeGreen(i);
```

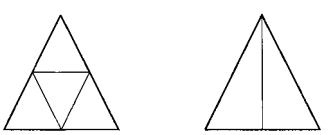

Figure 1. Red and green refinement of a triangle.

`Err (i)` is a function evaluating the error on the `i`th triangle by one of the methods described above. `StandRef (i)` refines the `i`th triangle in the standard way; if the triangle is *green* because of a previous refinement, its father is refined in its place, so as not to worsen further the quality of the mesh. When executing the first `for` block, only elements that have an error greater than a fixed error threshold `thresh` are refined. `NHang (i)` and `EType (i)` are functions that return the number of hanging nodes and type (standard or green) of the triangle `i` respectively. In the second `for` block, standard triangles that have more than one hanging node, or green ones that have at least one, are standardly refined. If some element has been refined, it may be that some other hanging node has been created; for this reason the last block is re-executed until possible. In the last block, the function `MakeGreen` green-refines all those triangles that have one hanging node, to ensure consistency of the mesh.

This algorithm converges in a finite number of iterations; at most, all the elements of the initial grid will be standard-refined. The grid refined by the algorithm listed above has a minimum number of *green* elements, which are the only elements deteriorating the quality of the initial mesh.

4.1. Refining and coarsening

When designing a practical strategy of mesh refinement–coarsening, it would be very useful to state first an acceptable numerical error and then use it as a yardstick: coarsen the mesh where the error indicator is lower than the reference one, refine when larger. Unfortunately, the error indicators described above only give, at best, estimates of the numerical error, or hints about *where* the error is larger: they do not ensure any absolute evaluation of its magnitude.

When computing steady flows, this difficulty is overcome by stating first the computational resource that can be afforded, i.e. the maximum number of nodes to be used in the numerical simulation. Then, starting with quite a coarse mesh, it is refined until that the desired number of nodes is reached.

When dealing with unsteady flow, coarsening is also useful. In this paper, smooth flow is addressed, i.e. subcritical shallow water flow or, at most, locally transcritical flow. In such a regime, there are no discontinuities in the physical variables, and typically, the time dependency is due to the change of boundary conditions which is smooth in time and has a period of 12–24 h. As time goes by, the flow field changes and a proper mesh adaptation strategy should modify the mesh, refining it in an optimal way with respect to the adopted error indicator. In this framework, instead of keeping constant the number of nodes, as was the case of steady flow, it is more significant to keep constant the maximum error indicator of the grid at any time. This ensures a constant control of the error during the whole simulation.

5. NUMERICAL RESULTS: STEADY FLOW

To investigate numerically the performance of the mesh adaptation strategy outlined above, a test case involving several characteristic features of shallow water flow has been chosen.

In Figure 2 the geometry of the channel and the initial computational mesh used for the present calculations are shown. The test is designed to collect, in a sketchy fashion, a number of typical situations that occur in river flow. The main channel is 2 km long and 100 m wide, it has an abrupt enlargement that doubles its width, a smaller inflowing branch and a small square island in the middle. In the initial part, the bottom of the channel has a constant depth

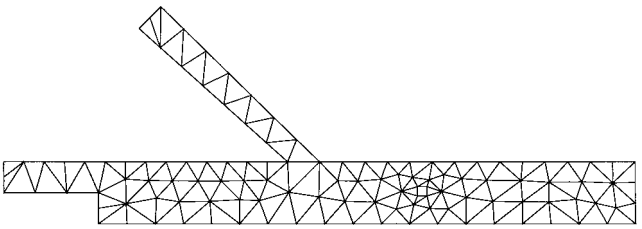

Figure 2. Background mesh.

of 3 m, in the final part, just behind the island, it has a jump, the depth rapidly increasing to 6 m. The inflow boundary conditions of the main channel and of the secondary channel are 300 and 100 $m^3 s^{-1}$ respectively. At the outflow, a constant water elevation is imposed.

The initial computational mesh is intentionally quite coarse, being composed of 120 nodes only. In Figure 3, the velocity field computed on such a background grid is illustrated. The simulation performed on the initial grid (u_0 solution) does not reveal any appreciable recirculations, neither behind the abrupt enlargement nor behind the island, which in fact exist.

It was decided to refine the initial grid adaptively up to three levels of refinement, the final mesh being composed of about 360 nodes. Figure 4 shows the refined mesh obtained by using the error indicator ϵ_2. The adapted mesh obtained by using the error indicator ϵ_1 is very similar and for this reason it is not shown.

The use of both the error indicators leads to meshes refined in the regions of interest: around the corner of the primary inflow channel, behind the enlargement, near the corners of the secondary inflow channel, around the island and in correspondence of the batimetry jump. It has been verified, plotting the errors defined in the following in Equation (20) (not shown here), that these regions are the ones where the greatest errors in the discharge field are located.

The global mass error performed by the coarse initial mesh, defined as the difference between the inflowing and the outflowing water, is about 3%. By using the grid refined as driven by the ϵ_2 indicator, the mass defect is reduced to 1%.

To get a quantitative evaluation of the quality of the adaptively refined meshes, a numerical simulation has been run on a grid obtained by refining uniformly three times the background grid, up to a final number of elements that is 64 times the initial one (u_3 solution). Taking u_3 as a reference solution, the numerical error is then evaluated as the difference between the water elevation and velocity computed on the finest mesh and the values computed on the adaptively refined meshes. Figures 5 and 6 show plots of the error, defined as

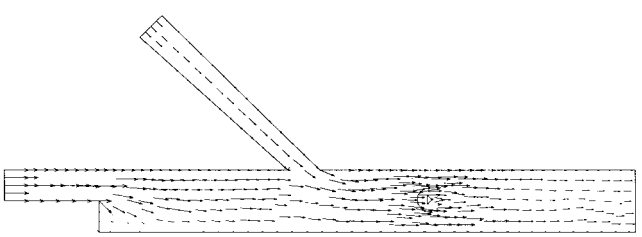

Figure 3. Velocity field computed on the background mesh.

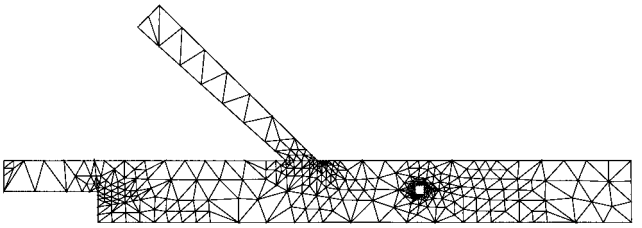

Figure 4. Refined mesh obtained using the error indicator ϵ_2.

$$\eta_i^{\xi} = \frac{|\xi_{3,i} - \xi_i|}{\xi_{3,i}}, \qquad \eta_i^{v} = |v_{3,i} - v_i|, \tag{20}$$

where $\xi_{3,i}$, $v_{3,i}$ is the reference solution in the ith node computed on the finest grid. The absolute value of the velocity error is considered, so as not to overestimate the contribution due to the stagnation points.

Generally speaking, all the simulations show that the bigger residual error is located around the island, which is correctly the region most refined in all the adapted meshes. Moreover, the results shown in Plates 1 and 2 and in Table I show both locally and globally that the error indicator ϵ_2 performs better in computing the water elevation and gives better global results in computing the velocity field.

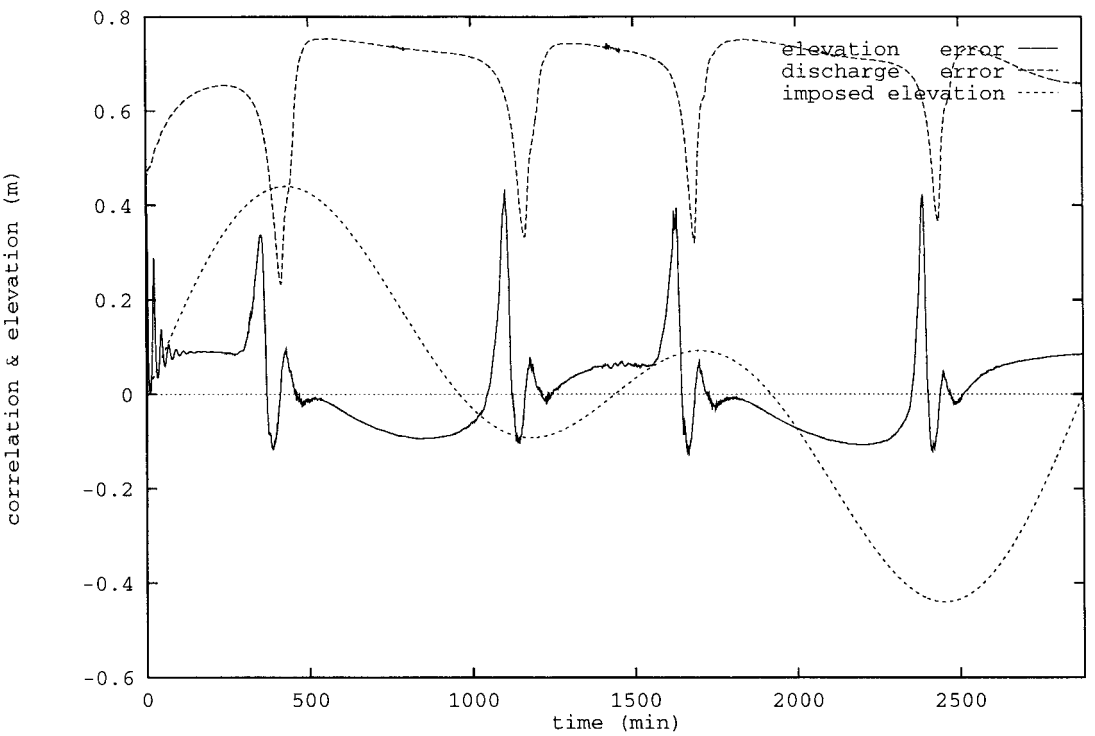

Figure 5. Spatial correlation between the ϵ_3 error estimators and the error as a function of the integration time. Dotted line is the imposed elevation expressed in metres.

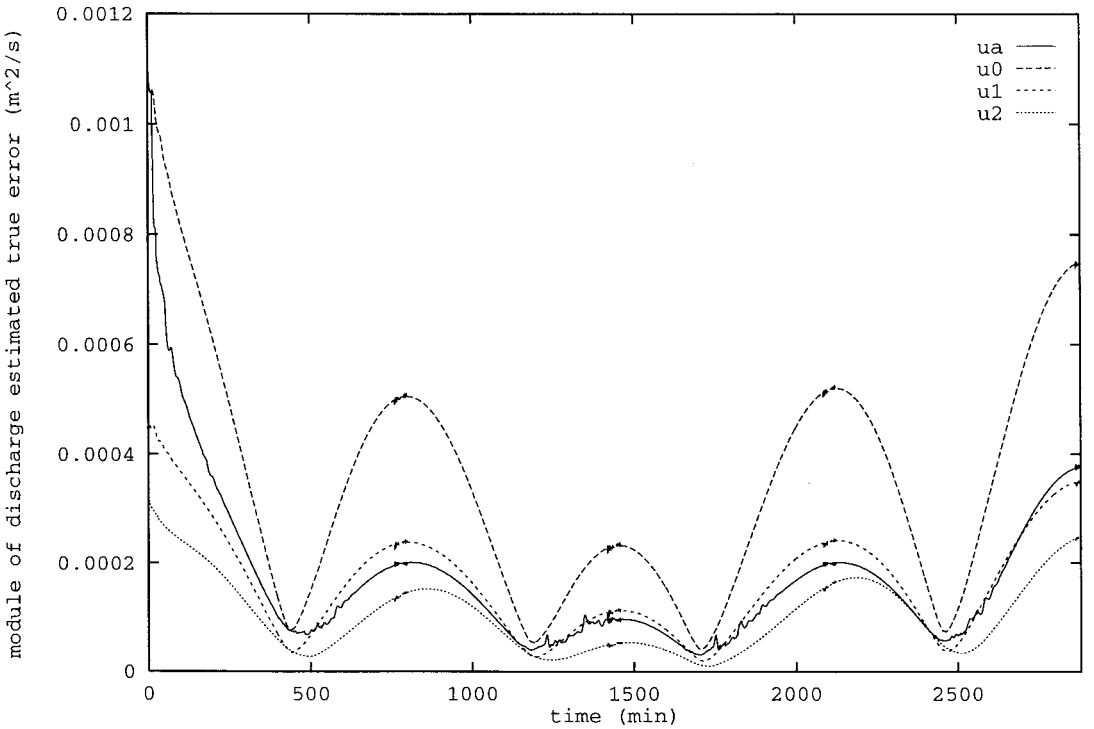

Figure 6. Average error of the discharge magnitude as a function of the integration time; solutions u_0, u_1, u_2, u_a.

The maximum relative error in the water elevation is about 5% and it is confined to a small area behind the island. The maximum value of the error obtained with the estimator ϵ_1 is comparable but values are generally greater by about 1% in a great part of the domain. The analysis of the numerical error performed in computing the flow speed deserves more or less the same comments. The maximum error induced by the criteria ϵ_2 is, again, strongly confined around the island, in contrast to the other indicator which lead to a relevant error also near to the right inflow corner of the secondary channel. The average errors of the two adapted solutions are much lower than those of the u_0 solution ($\overline{\eta^{\xi}} = 7\%$, $\overline{\eta^{q}} = 0.054$ m s^{-1}). The errors of the u_{ϵ_2} solution ($\overline{\eta^{\xi}} = 1.1\%$, $\overline{\eta^{q}} = 0.010$ m s^{-1}) are a little bit lower than those of the u_{ϵ_1} solution ($\overline{\eta^{\xi}} = 2.3\%$, $\overline{\eta^{q}} = 0.013$ m s^{-1})', but this could be related to the different number of nodes in the two final meshes (see the last column of Table I).

Table I. Average errors of the computed solutions and number of $P1$ nodes (NV1) of the corresponding grids

Solution	$\bar{\eta}_{\xi}$ (%)	$\bar{\eta}_{v}$ (m s^{-1})	$\bar{\eta}_{v}$ Direction (°)	NV1
u_0	7.0	0.054	3.9	120
u_{ϵ_1}	2.3	0.013	1.7	353
u_{ϵ_2}	1.1	0.010	1.7	375
u_1	6.0	0.030	1.8	403
u_2	1.6	0.022	1.6	1458

To better evaluate the advantages that can be achieved in terms of computational efficiency by using the adaptation procedure, average errors of the solutions obtained by refining globally the background mesh one and two times (u_1 and u_2 solutions) are shown in Table I. The solution on the background grid (u_0) is poor and differs little with respect to that on the grid uniformly refined one time (u_1). Errors of the u_2 solution are instead much smaller, indicating that the solution on the finest grid is converging. However, the more striking result that can be deduced by the numbers in the table and from the comparison of local errors, Plates 1 and 2, is that the adapted solution u_{ϵ_2} (375 $P1$ nodes) is even better than the solution obtained by uniformly refining two times the initial mesh (u_2), which has a larger number of nodes (1451 $P1$ nodes), and then a computational cost about four times greater.

6. NUMERICAL RESULTS: UNSTEADY FLOW

To further investigate the performance of both the error indicators and the adaptation procedure, an unsteady flow has been considered. In particular, the case of the circulation inside a closed basin induced by a tidal current has been chosen. This kind of unsteady hydrodynamic problem, commonly studied using the shallow water model, has a great environmental interest. For example, the coupling a the fluid dynamics code with a transport–diffusion–reaction model of nutrients, as nitrogen and phosphorus, enable recognition of conditions leading to eutrophication and then to anoxic crisis in basins characterized by a long retention time [11].

The computational domain used here is geometrically the same as illustrated previously for the steady flow case (see Figure 2). Now, the two inflowing branches from the left of the domain are supposed to be closed and the water elevation is imposed at the right open boundary as given by the following expression:

$$\xi(t) = \frac{1}{4}\left[\cos\frac{2\pi t}{24} + \cos\frac{2\pi t}{12}\right]. \tag{21}$$

This boundary condition simulates a tide of about 1 m of amplitude, a typical value in the Mediterranean area, with a period of 48 h due to the superimposition of a period of 12 h and one of 24 h respectively (see Figure 5). It can be expected that for such a tide-driven flow, the typical values of the velocity field will be very small because the tidal current varies very slowly in time.

To estimate quantitatively the amplitude of the error of the numerical solution, a simulation using the background mesh of Figure 2 (u_0 solution) has been performed and another one using the grid obtained by uniformly refining three times the background one (u_3 solution).

Using u_3 as a reference solution, the maximum relative deviation of the solution u_0 is calculated as

$$\eta_{\text{rel}}^{\max} = \max_{x,y,t} \frac{|u_3(t) - u_0(t)|}{\max_{x,y} |u_3(t)|}, \tag{22}$$

where u can be either ξ or q.

It was observed that the maximum deviation for the elevation is very small ($\simeq 10^{-4}$), while the one related to the discharge magnitude is not negligible ($\simeq 0.33$). The very small deviation of ξ makes the location of the elevation errors for mesh adaptation purposes for the present test case very difficult, if not impossible. The deviation of the magnitude of the discharge is instead significant, although—as already remarked—absolute values of discharge in this test

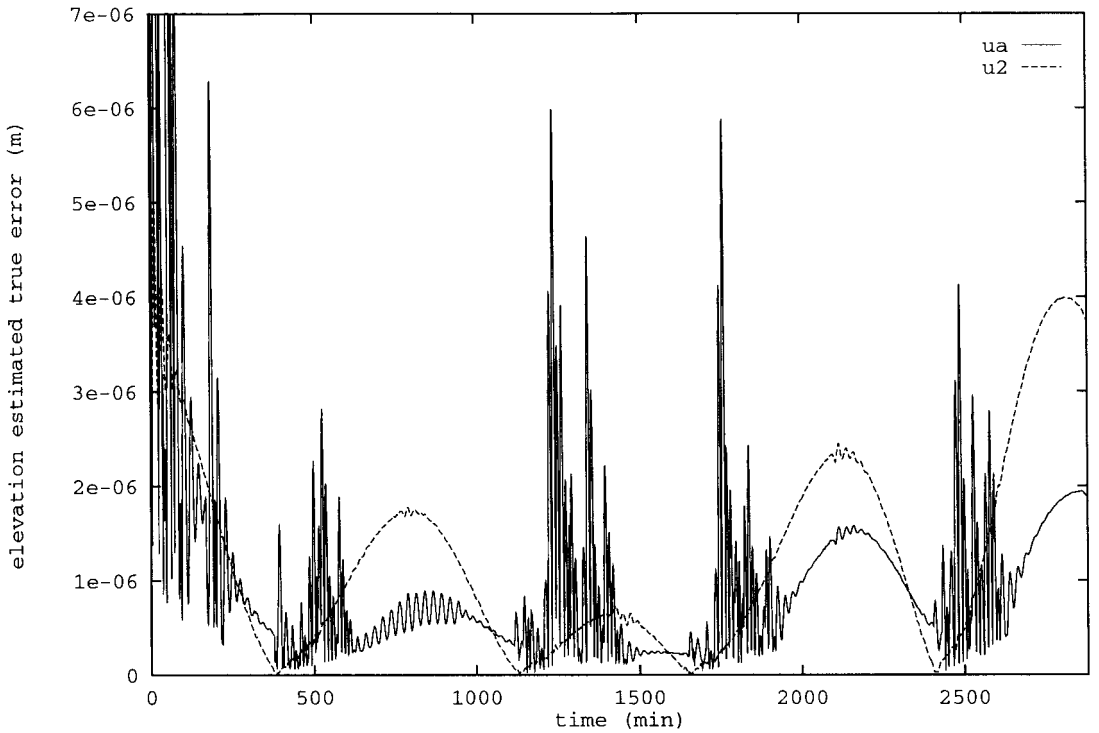

Figure 7. Average error of the elevation as a function of the integration time; solutions u_2 and u_a.

are very small. These characteristics of the flow field of the specific test case at hand should be kept in mind in the following discussion.

The rate of convergence of the numerical solution to the exact one by refining globally the grid has been obtained by comparing the solutions of u_0 and u_3 with those obtained by refining the background mesh uniformly one and two times (u_1 and u_2 solutions). The maximum deviation of the magnitude of the discharge of solutions u_1 and u_2 is $\simeq 0.12$ and $\simeq 0.06$ respectively. From now on we will refer as *error* the difference between the given and the reference solution u_3, evaluated on the nodes of the background mesh.

To verify the performance of the three error estimators proposed in Section 2, their spatial correlation has been calculated

$$\rho = \sum_i \frac{\epsilon_i \eta_i}{\sigma^\epsilon \sigma^\eta}, \tag{23}$$

with the corresponding error for the solution u_0. It is found that the correlation is practically zero for the elevation field: this is due to the fact that the errors of the solution for this variable are negligible (see Figure 5). The temporal evolution of the correlation for the magnitude of the discharge appears to be more interesting: its value is typically equal to about 0.8 but periodically falling to very low values. In the same figure the temporal evolution of the elevation at the boundary is shown: it may be seen that periods of low correlation are in correspondence with periods of stagnation of the tide ($\mathrm{d}\xi/\mathrm{d}t = 0$). When this happens, the velocity of the water vanishes in a large part of the domain, because the direction of the flow changes. For this reason, the low correlation obtained should not be surprising, nor of

concern: when the correlation is low all the estimated errors have a minimum (see Figures 6 and 7). Spatial correlation for the other two estimators (not shown) give similar results and for this reason only the estimator ϵ_2 will be used in the following tests. The criterion adopted during the adaptation procedure is the one described in Section 4.1, with an adaptation call every 13 min of simulated time. In addition, the constraint that the number of nodes should not exceed 360 is added (this solution will be referred to as u_a). Temporal evolution of the mean value of the error of the magnitude of the discharge is shown in Figure 6.

In the same figure, the errors of the solutions u_0, u_1 and u_2 are also shown.

The average errors for the solution u_a are very near to those of the solution u_1 (see also Table II). Moreover, after about the first 6 h of integration, during which there is a slow relaxation of the initial condition to the boundary conditions, errors of the adaptive solutions are between those of solution u_1 and solution u_2. In Table II, the errors, spatially averaged and temporally averaged, excluding the first 6 h of integration, are shown. As can be seen, although the average number of nodes used in the adaptive run is lower than that of the u_1 run and about one fifth of that in the u_2 run, the global errors of the adaptive solutions are between those of those solutions. Another indication that the adaptation procedure is working correctly can be deduced from Figure 8: the number of vertices as a function of the integration time is proportional to the error (see also Figures 6 and 7).

The estimated true error of the elevation field shows a more complicated picture: every time that the mesh is changed, the solution is interpolated in the new nodes. In general, this generates a high frequency component of unphysical noise in the solution. In the case of tidal current, the relative errors of the elevation are very small; such an interpolation noise is comparable with the numerical error. Therefore, although the mean global error of the adaptive solution is also smaller than that of u_2 (Table II), it cannot be said in general that the elevation field of the first solution is more accurate than the of the second one.

7. CONCLUSIONS

The numerical tests show that mesh adaptation is a very reliable tool for numerical simulation of shallow water steady flows. Any of the used error indicators produce numerical results that are greatly improved with respect to a uniform mesh, with only a minor increase in the computational effort. The mesh refinement–coarsening technique is almost decoupled from the numerical integration aspects and can then be easily introduced in codes that have not been originally thought in an adaptive perspective. Last, but not least, the initial mesh generation

Table II. Average maximum and average error of elevation ξ, magnitude of discharge q and of velocity v

Solution	$\bar{\eta}_\xi$ $\times 10^{-4}$ m	$\bar{\eta}_q$ $\times 10^{-2}$ m^2 s^{-1}	$\bar{\eta}_v$ $\times 10^{-3}$ m s^{-1}	NV1
u_0	2.2	3.1	9.7	120
u_1	1.2	1.5	4.6	403
u_2	1.2	0.9	2.7	1458
u_a	0.9	1.3	4.6	310

NV1 is the number of vertices in each mesh; in the adaptation case NV1 is the average number during all the integration. Only solutions after the first 6 h are used in the temporal averages.

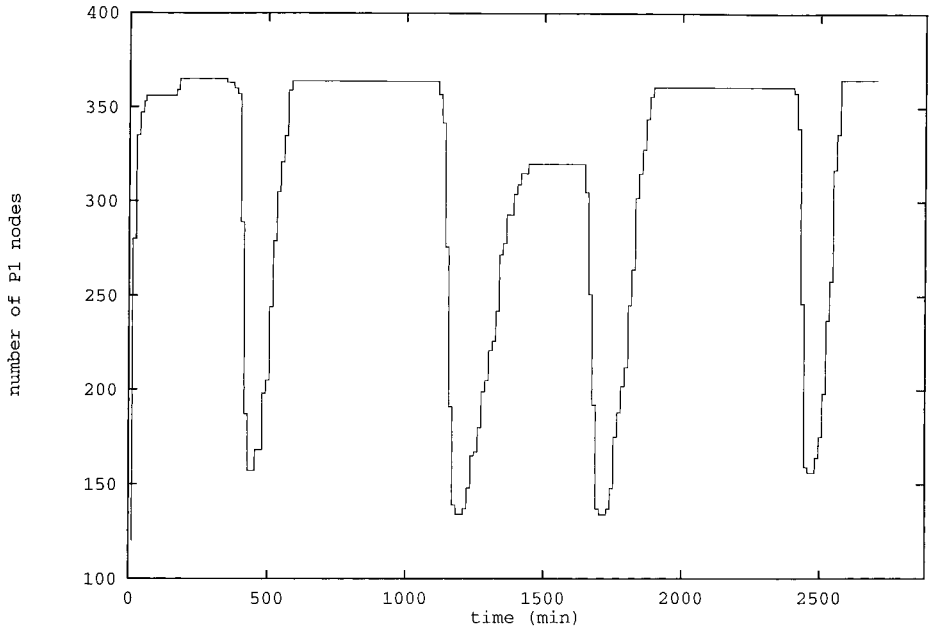

Figure 8. Number of vertices of the grid as a function of the integration time using the adaptation.

does not require any *a priori* knowledge of the flow field, as any sufficiently regular background grid is automatically refined to an almost optimal one.

Although the numerical results have been obtained for two test cases that have been designed to include more general scenarios, general conclusions cannot be drawn about a possible *best performing* error indicator. However, it is possible to say that both the error indicators work correctly, detecting regions where truncation error is relevant. The local mass conservation criteria performs better in the test case of steady flow, yielding always to lower errors in a global sense. The reason for this behaviour is probably that only the mass defect check involves both of the unknowns of the problem (velocity and elevation).

In the unsteady flow test case, the performance of all error indicators is good, since spatial correlation with the error of the magnitude of the discharge has the value of about 0.8, during periods where the speed of the water is not negligible. The lack of correlation of the error of the elevation field is not relevant, since amplitude of the relative error is very small ($\sim 10^{-5}$) in this particular case. The adaptation procedure gives encouraging results, with global errors always smaller than those obtained by uniformly refining the whole mesh. Some other test case should be carried out to quantify the benefits obtainable by mesh adaptation when the relative errors of the elevation field are not as small as the one that we have observed when computing tidal currents.

ACKNOWLEDGMENTS

This research has been supported by the Sardinia Region Authorities and by ENEL Ricerca, Polo Idraulico. The authors thank Professor Alfio Quarteroni for several suggestions and their colleagues L. Formaggia, L. Paglieri and L. Trotta for many helpful discussions.

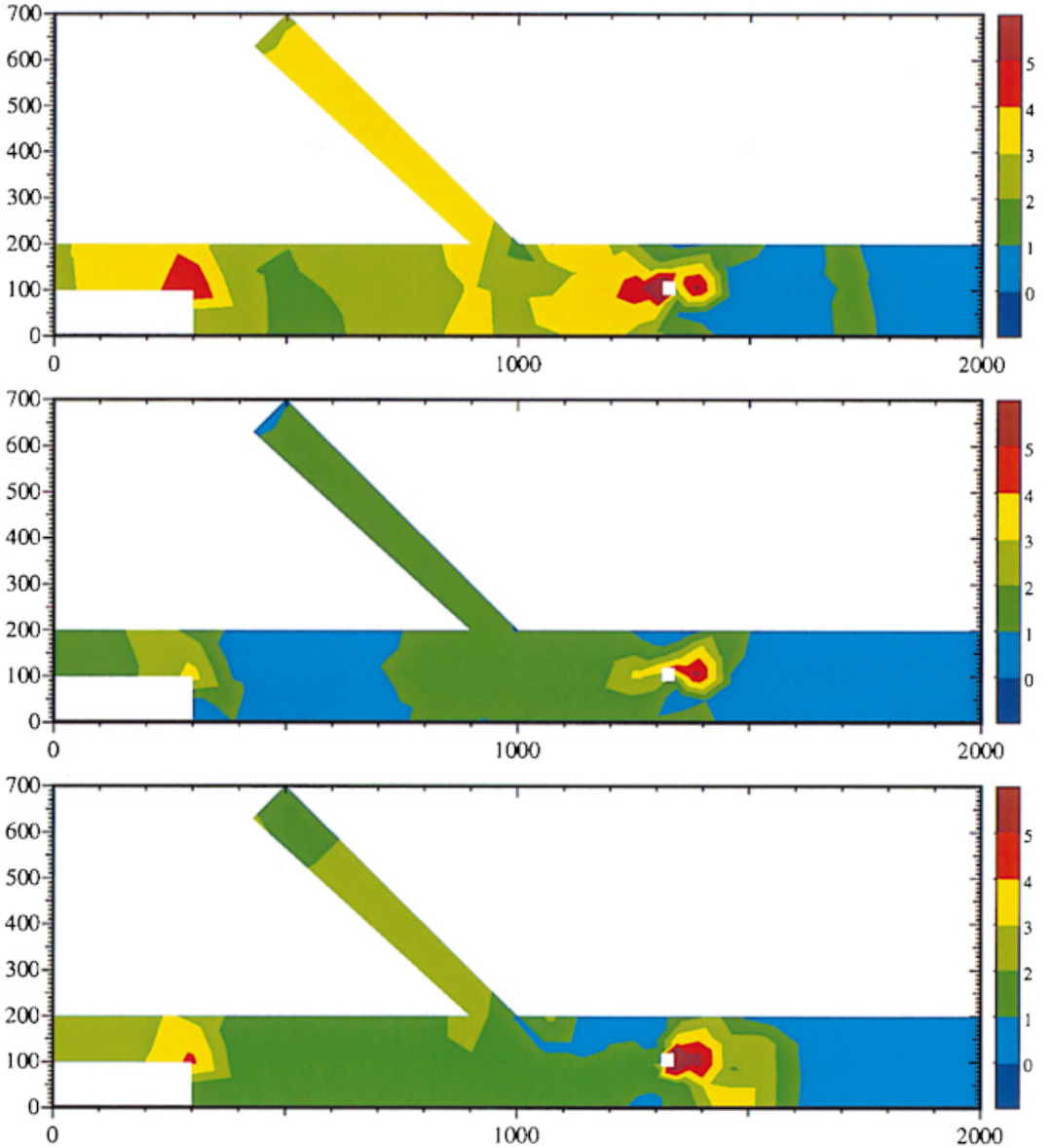

Plate 1. Relative error in the water elevation value obtained by using the ϵ_1 and the ϵ_2 error indicators and by uniformly refining, two times, the background mesh.

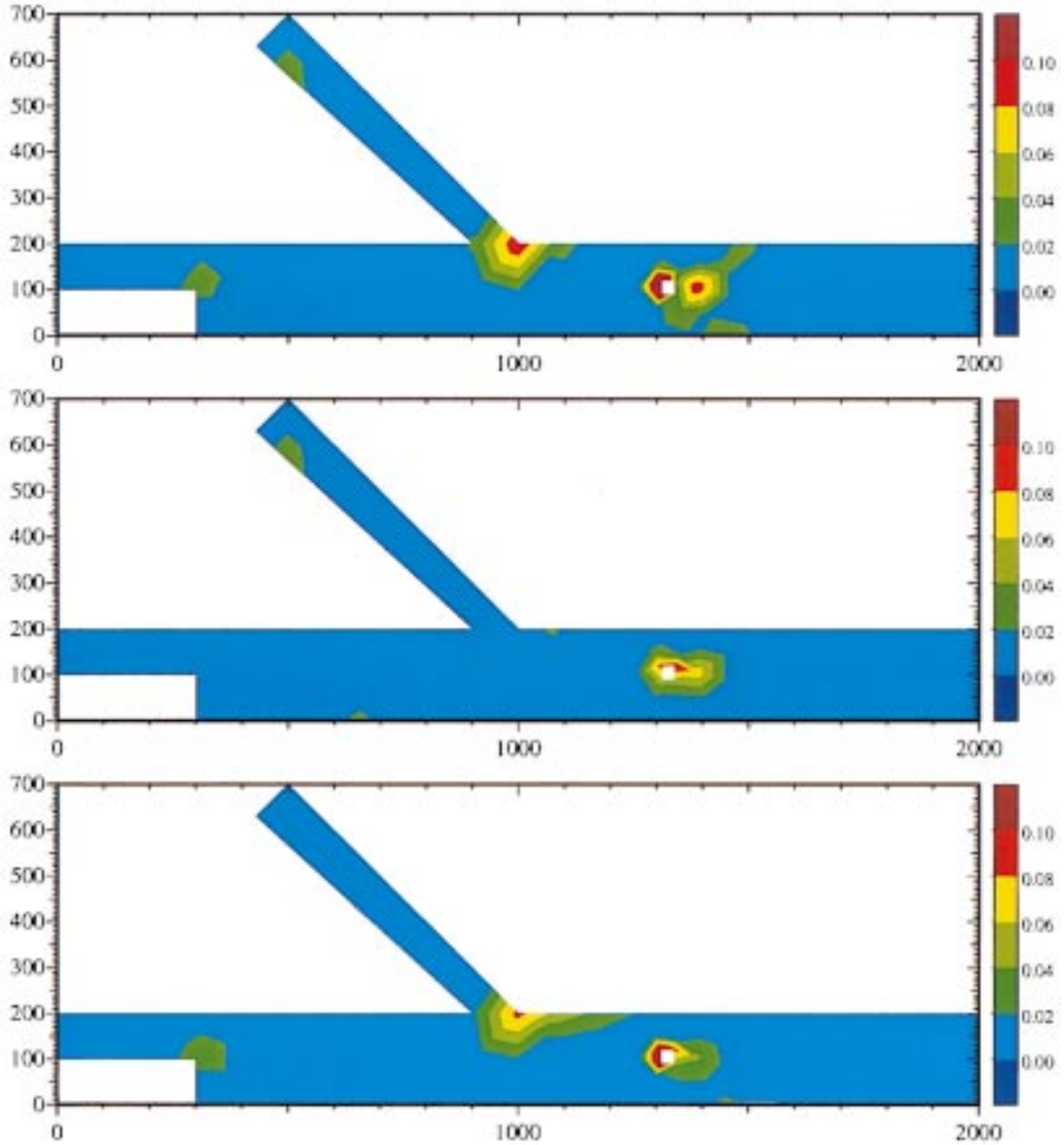

Plate 2. Absolute error in the water speed obtained by using the ϵ_1 and the ϵ_2 error indicators and by uniformly refining, two times, the background mesh.

REFERENCES

1. R. Walters and E. Barragy, 'Comparison of *H* and *P* finite element approximations of the shallow water model', *Int. J. Numer. Methods Fluids*, **24**, 61–79 (1997).
2. D. Ambrosi, S. Corti, V. Pennati and F. Saleri, 'Numerical simulation of unsteady flow at Po river delta', *J. Hydraul. Eng.*, **122**, 743 (1996).
3. R. Lohner, 'Finite element methods in CFD: grid generation, adaptivity and parallelization', in *Unstructured Grids Methods for Advection Dominated Flows*, AGARD-R-787, 1992.
4. C. Johnson, *Numerical Solution of Partial Differential Equations by the Finite Element Method*, Cambridge University Press, Cambridge, 1987.
5. O. Zienkiewikz and R. Taylor, *The Finite Element Method*, McGraw-Hill, New York, 1989.
6. O. Pironneau, 'On the transport–diffusion algorithm and its application to the Navier–Stokes equations', *Numer. Matematik*, **38**, 339–332 (1982).
7. J. Benquè, J. Cunge, J. Feuillet, A. Hauguel and F. Holly, 'New method for tidal current computation', *J. Waterway Port Coast. Ocean Eng.*, **108**, 396–417 (1982).
8. R. Sacco and F. Saleri, 'Mixed finite volume methods for semiconductor device simulation', *Numer. Methods Part. Different. Equat.*, **3**, 215–236 (1997).
9. T. Utnes, 'Finite element modelling of quasi-three-dimensional nearly horizontal flow', *Int. J. Numer. Methods Fluids*, **12**, 559–576 (1991).
10. V. Casulli and R.T. Cheng, 'Semi-implicit finite difference methods for three-dimensional shallow water flow', *Int. J. Numer. Methods Fluids*, **15**, 629–648 (1992).
11. A. Balzano, R. Pastres and C. Rossi, 'Coupling hydrodynamic and biochemical mathematical models for lagoon ecosystem', *XI Int. Conf. on Computational Methods in Water Resources*, Cancun, Mexico, 1996.

