Mesh adaption strategies for shallow water flow

M. Marrocu(1), D. Ambrosi(2)

(1) CRS4, Centro di Ricerca, Sviluppo e Studi Superiori in Sardegna,
Via Nazario Sauro, 10 Cagliari, I-09123 Italy.
(2) Dipartimento di Matematica, Politecnico di Torino,
Corso Duca degli Abruzzi 24, 10129 Torino, Italy

Manuscript submitted to

International Journal for Numerical Methods in FLUIDS

Manuscript version from March 3, 1998

Offset requests to:
M. Marrocu
CRS4, Via Nazario Sauro, 10
Cagliari, I-09123 Italy.
Email: marino@crs4.it Fax: (+39)-70-2796216

Send proofs to:
M. Marrocu
CRS4, Via Nazario Sauro, 10
Cagliari, I-09123 Italy.
Email: marino@crs4.it Fax: (+39)-70-2796216

Abstract

In this paper it is shown how the mesh adaption 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 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 that are obtained applying this adaptive approach by considering specific test cases of steady and unsteady flows.

keywords: mesh adaption, shallow water, finite element, error estimate.

Correspondence to: M. Marrocu

Introduction

The use of unstructured grids for the numerical approximation of partial differential equations of applied mathematics has two main attractives. The most commonly claimed one is the geometrical flexibility, that is the capability to handle 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 of unstructured grids that has even more relevance: the possibility to refine 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 to refine the mesh where the numerical error is large and to coarsen it where the error is small, in order to optimize the quality of the computed solution for a given computational effort.

Mesh adaption techniques have been used since many years in several fields of computational fluid dynamics, but adaptivity has not yet much explored in the framework of free surface hydrostatic flow. At our knowledge only a very recent paper (Walters and Barragy (1997)) compares and discusses the use of high order polynomial basis (p adaption) for the discretization of the shallow water equations versus local mesh refinement, where the the order of the polynomial approximation is kept unchanged (h adaption).

This paper deals with local mesh adaption strategies for shallow water flow. The numerical solver for the shallow water equations coupled with the mesh adaption procedure, is of fractional–step type. It has been already extensively tested and successfully used with static meshes in several applications (Ambrosi et al. (1996)).

The mesh adaption technique that is described in the present paper is based on the use of a background grid. The numerical simulation starts on a grid, possibly composed by few nodes, which is the coarsest mesh which may be used in the computation: its nodes are neither moved nor suppressed. Successive levels of refinement and coarsening lie on such background grid. This technique has been adopted because of the very complicated geometry of the boundary that characterizes environmental applications in river, 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 adaption is that little care has to be used in defining the initial grid: "with adaption, any initial grid will be transformed into a near optimal discretization" (Lohner (1992)).

A peculiar aspect of the mesh adaption 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.

1 The shallow water equations and the numerical scheme

The shallow water equations in conservative differential form read

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

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

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 (1-2) is a finite element scheme of fractional step type. Here the numerical scheme is briefly described, more details can be found in Ambrosi et al. (1996).

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 the 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

$$
\begin{aligned}
\mathbf{q}^{n+1/3} &= H^n \mathbf{v}^{n+1/3}, \\
\mathbf{q}^{n+2/3} + \Delta t\, g \frac{\mathbf{q}^{n+2/3}|\mathbf{q}^{n+1/3}|}{H^2 H^{1/3} K^2} &= \\
\mathbf{q}^{n+1/3} + \Delta t \left[\nabla \cdot \left(\mu \nabla \mathbf{q}^{n+1/3}\right) - 2\Omega \times \mathbf{q}^{n+1/3}\right], & \qquad (4)
\end{aligned}
$$

step 3

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

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

The symbol $\mathbf{v}^n \circ \mathbf{X}$ 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, the equations (5) and (6) are decoupled by subtracting the divergence of (5) from (6). One then solves the following Helmholtz–type equation:

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

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 (1987) and Zienkiewikz and Taylor (1989). 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 involve an elliptic equation for it.

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

In the integration of the weak formulation of Eqs. (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* we intend 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 nontrivial diagonalization has to be performed (contrary to the case of P1 elements), otherwise a singular matrix is recovered (see appendix 8 of Zienkiewikz and Taylor (1989)). 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 sub-triangle and the total integral is the sum of the sub-integrals automatically leading to a diagonal mass matrix.

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

1.1 Lagrangian scheme for the convective terms

At step 1, the advective part of the momentum equation is integrated by a Lagrangian scheme (Pironneau (1982), Benquè et al. (1982)). Rewriting the convective terms of equation (1) in Lagrangian form, results in the solution of two coupled ordinary differential equations:

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

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

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 we are interested in computing the solution at the mesh nodes, let us consider the node with coordinates $\mathbf{y}$. The initial condition associated with equation (9) is:

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

To integrate equation (9) we 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 equations (8) immediately give:

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

The Lagrangian approach for convective terms as described above is not the *weak* Lagrangian scheme discussed in Pironneau (1982); 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. We note 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 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 centered 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 grids. 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, that has been adopted for compatibility reasons, here ensures that the interpolation of the velocity at the foot of the pathline is not too dissipative.

2 Error estimate and error indicators

The mesh adaption 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. In this section we introduce 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 (Johnson (1987)) that the distance between the exact and the computed solution is bounded as follows:

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

where Ω is the computational domain. As the computational kernel of the numerical scheme described in Section 1 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 $\{\phi_k\}$, we can write

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

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

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

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 equations (1), and are then consistent with it. However, the finite element scheme illustrated above does not ensure mass conservation in a *finite–volume cell–centered* 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 of this is twofold. On one hand the use of quadratic polynomials to approximate the discharge $\mathbf{q}$ yields to 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 analyzed 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–centered finite volume technique (Sacco and Saleri (1997)).

Secondly, we observe that the substitution of the momentum equation into the continuity equation, that leads to Eq.(7) involves a new spatial derivation. This is an usual approach in the finite element context (Utnes (1991)), 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 (5-6) are first discretized in space and then the substitution is carried out, without further derivatives. This ensures local triangle-by-triangle mass conservation (Casulli and R.T. (1992)).

The considerations above suggest to use the check of local mass balance as an error indicator for the present scheme. We define then

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

where $\mathbf{\Gamma}$ is the contour of the e-th element so that ϵ_3 is the mass defect in the e-th element.

It is to be mentioned that other possible error indicators out 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 to appreciable results in comparison with the ones described above.

3 The mesh refinement technique

Any error estimator among the ones described in the section above allows to identify a set of elements of the mesh to be refined (or coarsened). Several techniques can then be used to this aim (Lohner (1992)).

1. Repositioning of the mesh (r-methods):. local refinement of the mesh is obtained 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 made in two ways:

1. edge bisection: the midpoint of an edge to be refined is connected with the vertex opposite to the edge itself.
2. 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 for a factor 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 adaption 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 flowfield. Adapted grids for these problems are in general very irregular both in side length and in shape (Habashi et al. (1996)) and the more appropriate strategy for these kind of problems seems to be is the h-method 2.1.

In the present paper we apply the h-method 2.2 for shallow water flow simulations. Main reason for this choice is that the h-methods keep 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 a quite coarse mesh, as regular as possible; then refinements and de-refinements are accomplished in such a way to preserve regularity.

In Fig. 1 is shown the subdivision technique known as *standard* or *regular* or *red* refinement is shown. The elements to be refined are first split in a standard way. The surrounding triangles have then one, two or three sub-divided edges that are not connected to nodes not belonging to 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 Fig. 1). At this aim, there are few possible strategies; we have chosen the one described in the following in 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);
```

`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 place, to not worsen further the quality of the mesh. When executing the first `for`-block, only elements which have an error greater than a fixed error threshold `thresh` are refined. `NHang(i)` and `EType(i)` are functions which return number of hanging nodes and type (standard or green) of the triangle `i`, respectively. In the second `for`-block, standard triangles which have more than one hanging node, or green ones which have at least one, are standardly refined. If some element has been refined, 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 which 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.

3.1 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 stating first the computational resource that can be afforded, that is the maximum number of nodes to be used in the numerical simulation. Then, starting with a quite coarse mesh, it is refined until that the desired number of nodes is reached.

When dealing with unsteady flow, also coarsening is useful. In this paper we are addressing smooth flow, that is sub-critical shallow water flow or, at most, locally trans-critical flow. In such 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 hours. As time goes by, the flowfield changes and a proper mesh adaption strategy should modify the mesh, refining it in a 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 significative 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.

4 Numerical results: steady flow

To investigate numerically the performance of the mesh adaption strategy outlined above, we have chosen a test case involving several characteristic features of shallow water flow.

In Fig. 2 we show the geometry of the channel and the initial computational mesh used for the present calculations. 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 kilometers long and 100 meters large, 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 of 3 meters, in the final part, just behind the island, it has a jump, the depth rapidly becoming 6 meters. The inflow boundary conditions of the main channel and of the secondary channel are 300 and 100 cubic meters per second, respectively. At the outflow, a constant water elevation is imposed.

The initial computational mesh is intentionally quite coarse, being composed by 120 nodes only. In Fig. 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 appreciable recirculation neither behind the abrupt enlargement nor behind the island, which in fact do exist.

It has been chosen to refine the initial grid adaptively up to three levels of refinement, the final mesh being composed by about 360 nodes. In Fig. 4 is shown 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. We have verified, plotting the errors defined in the following in Eq.20 (not shown here) that these regions are the ones where 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 refining uniformly three times the background grid, up to a final number of elements which is 64 times the initial one (u_3 solution). Taking u_3 as 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. The Figs. 5-6 show plots of the error, defined as

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

where $\xi_{3,i}$, $v_{3,i}$ is the reference solution in the i-th 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 mostly refined in all the adapted meshes. Moreover, the results in Figs. 5-6 and in Table 1 show that both locally and globally the error indicator ϵ_2 performs better in computing the water elevation and gives better global results in computing the velocity field.

The maximum relative error in the water elevation is about 5% and it is confined in a small area behind the island. Maximum value of the error obtained with the estimator ϵ_1 are comparable but, values are generally greater of about 1% in a great part of the domain. The analysis of the numerical error performed in computing the flow speed suggests more or less the same remarks. The maximum error induced by the criteria ϵ_2 is, again, strongly confined around the island, on the contrary the other indicator 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 ($\overline{\eta^\xi} = 7\%, \overline{\eta^q} = 0.054m/s$). The errors of the u_{ϵ_2} solution ($\overline{\eta^\xi} = 1.1\%, \overline{\eta^q} = 0.010m/s$) are a little bit lower than those of the u_{ϵ_1} solution ($\overline{\eta^\xi} = 2.3\%$, $\overline{\eta^q} = 0.013$) but this could be related to the different number of nodes in the two final meshes (see last column of Table 1).

To evaluate better the advantages that can be achieved in terms of computational efficiency by using the adaption procedure, we show in Table 1 average errors of the solutions obtained by refining globally the background mesh one and two times (u_1 and u_2 solutions). The solution on the background grid (u_0) is poor and differs not much 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. But the more striking result, that can be deduced by the numbers in the table, and from the comparison of local errors (Figs. 5,6), is that the adapted solution u_{ϵ_2} (375 P1 nodes) is even better than the solution obtained uniformly refining two times the initial mesh (u_2) which have a number of nodes (1451 P1 nodes), and then a computational cost, about four times greater.

5 Numerical results: unsteady flow

To investigate further the performance of both the error indicators and the adaption procedure, an unsteady flow has been considered too. In particular, it has been chosen the case of the circulation inside a closed basin induced by a tidal current. This kind of unsteady hydrodynamic problems, commonly studied using the shallow water model, have a great environmental interest. For example, the coupling a fluid dynamics code with a transport-diffusion-reaction model of nutrients, as nitrogen and phosphorus, enables to recognize conditions leading to eutrophication and then to anoxic crisis, in basins characterized by a long time of retention (Balzano et al. (1996)).

The computational domain used here is geometrically the same illustrated previously for the steady flow case (see Fig. 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:

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

This boundary condition simulates a tide of about one meter of amplitude, a typical value in the Mediterranean area, with a period of 48hours due to the superimposition of a period of 12hours and one of 24hours respectively (see Fig. 7). 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, we have performed first a simulation using the background mesh of Fig. 2 (u_0 solution) and another one using the grid obtained by uniformly refining three times the background one (u_3 solution).

Using u_3 as reference solution, we calculate the maximum relative deviation of the solution u_0 as:

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

where u can be either ξ or q.

It has been seen 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 very difficult, if not useless, the location of the elevation errors for mesh adaption purposes for the present test case. The deviation of the magnitude of the discharge is instead significant although - as already remarked - absolute values of discharge in this test are very small. These characteristics of the flowfield 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 comparing the solutions u_0 and u_3 with those obtained 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 1, we have calculated their spatial correlation:

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

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 Fig. 7). 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 we show the temporal evolution of the elevation at the boundary: it may be seen that periods of low correlation are in correspondence with periods of stagnation of the tide ($d\xi/dt = 0$). 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 also Figs. 8-9). Spatial correlation for the other two estimators (not shown) give similar results and for this reason we will use in the following tests only the estimator ϵ_3. The criterion adopted during the adaption procedure is the one described in Section 3.1, with an adaption call every 13 minutes of simulated time. In addition, we have added the constraint that the number of nodes should not exceed 360 (we will refer to this solution as

u_a). Temporal evolution of the mean value of the error of the magnitude of the discharge is showed in figure 8.

In the same figure we also show the errors of the solutions u_0, u_1 and u_2.

The average errors for the solution u_a are very near to those of the solution u_1 (see also Table 2). Moreover, after about the first six hours 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 2, the errors, spatially averaged and temporally averaged, excluding the first six hours 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 adaption procedure is working correctly can be deduced from Fig. 10: the number of vertices as a function of the integration time is proportional to the error (see also Figs. 8, 9).

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 (Tab. 2), we cannot say in general that the elevation field of the first solution is more accurate than the of the second one.

6 Conclusions

The numerical tests show that mesh adaption is a very reliable tool for numerical simulation of shallow water steady flows. Any of the used error indicators produce numerical results that are strongly 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 also in codes that have not been originally thought in an adaptive perspective. Last, but not least, the initial mesh generation does not require any a priori knowledge of the flowfield, 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 behavior is probably that only mass defect check involves both 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 adaption 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 adaption 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. We thank Prof. Alfio Quarteroni for several suggestions and our colleagues L. Formaggia, L. Paglieri and L. Trotta for many helpful discussions.

References

Ambrosi, D., Corti, S., Pennati, V., and Saleri, F., Numerical simulation of unsteady flow at Po river delta, *J. of Hydraulic Engineering*, *122*(12), 1996.

Balzano, A., Pastres, R., and Rossi, C., in *Coupling hydrodynamic and biochemical mathematical models for lagoon ecosystem.*, XI Int. Conf. on Comp. Meth. in Water Resources, Cancun, Mexico, 1996.

Benquè, J., Cunge, J., Feuillet, J., Hauguel, A., and Holly, F., New method for tidal current computation, *J. of Waterway, Port, Coastal and Ocean Engineering*, *108*, 396–417, 1982.

Casulli, V. and R.T., C., Semi-implicit finite difference methods for Three dimensional shallow water flow, *Int. J. Numer. Meth. Fluids*, *15*, 629–648, 1992.

Habashi, W., Fortin, M., Dompierre, J., Vallet, M.-G., and Bourgault, Y., Certifiable CFD through mesh optimization, *submited to AIAA Journal*, pp. 1–22, 1996.

Johnson, C., *Numerical solution of partial differential equations by the finite element method*, Cambridge University Press, 1987.

Lohner, R., Finite element methods in CFD: grid generation, adaptivity and parallelization, in *Unstructured grids methods for advection dominated flows*, AGARD-R-787, 1992.

Pironneau, O., On the Transport–Diffusion Algorithm and its Application to the Navier–Stokes Equations, *Numerische Matematik*, *38*, 339–332, 1982.

Sacco, R. and Saleri, F., Mixed Finite Volume Methods for Semiconductor Device Simulation, *Numerical Methods for Partial Differential Equations*, *3*(13), 215–236, 1997.

Utnes, T., Finite element modelling of quasi-three-dimensional nearly horizontal flow, *Int. J. Numer. Meth. Fluids*, *12*, 559–576, 1991.

Walters, R. and Barragy, E., Comparison of *H* and *P* finite element approximations of the shallow water model, *Int. J. Numer. Meth. Fluids*, *24*, 61–79, 1997.

Zienkiewikz, O. and Taylor, R., *The finite element method*, Mc Graw–Hill, 1989.

Tables

solution	$\overline{\eta}_\xi (\%)$	$\overline{\eta}_v$ (m/s)	$\overline{\eta}_v$ direction (deg)	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

Table 1. Average errors of the computed solutions and number of P1 nodes (NV1) of the corresponding grids.

solution	$\overline{\eta}_\xi \cdot (10^{-4}\mathrm{m})$	$\overline{\eta}_q \cdot (10^{-2}\mathrm{m}^2/\mathrm{s})$	$\overline{\eta}_v \cdot (10^{-3}\mathrm{m/s})$	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

Table 2. Average maximum and average error of elevation ξ, magnitude of discharge q and of velocity v. NV1 is the number of vertices in each mesh; in the adaption case NV1 is the average number during all the integration. Only solutions after the first six hours are used in the temporal averages.

Figure Captions

Fig. 1. Red and green refinement of a triangle.

Fig. 2. Background mesh.

Fig. 3. Velocity field computed on the background mesh.

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

Fig. 5. 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.

Fig. 6. 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.

Fig. 7. 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 meters.

Fig. 8. Average error of the discharge magnitude as a function of the integration time; solutions u_0, u_1, u_2, u_a.

Fig. 9. Average error of the elevation as a function of the integration time; solutions u_2 and u_a.

Fig. 10. Number of vertices of the grid as a function of the integration time using the adaption.

Figures

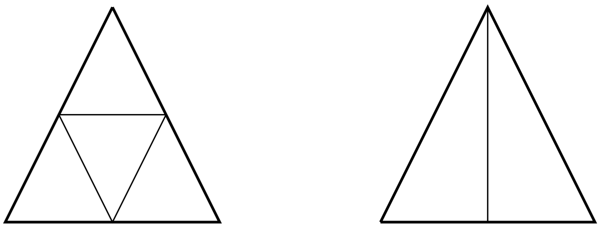

Fig. 1. Red and green refinement of a triangle.

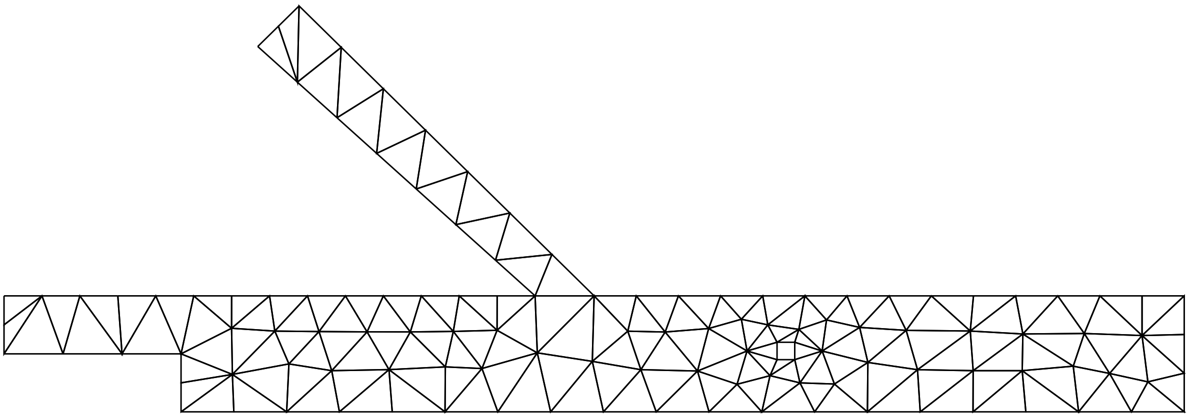

Fig. 2. Background mesh.

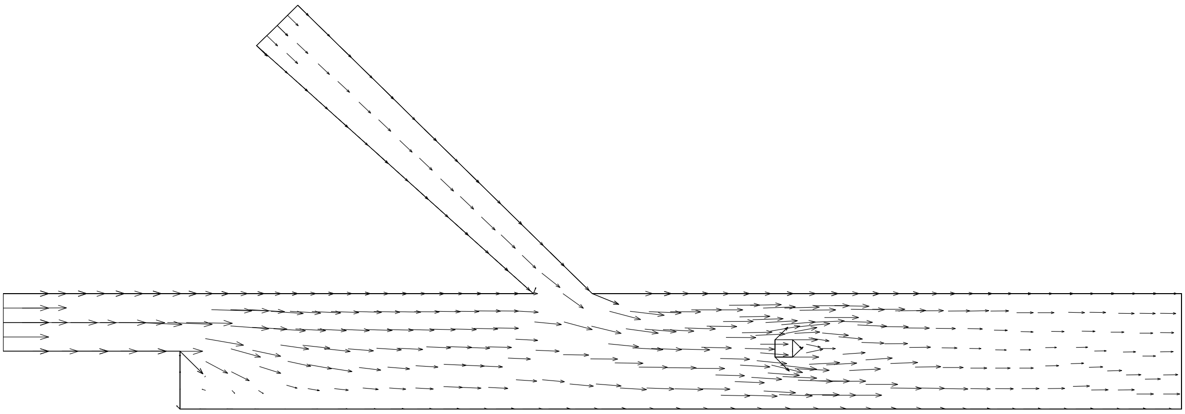

Fig. 3. Velocity field computed on the background mesh.

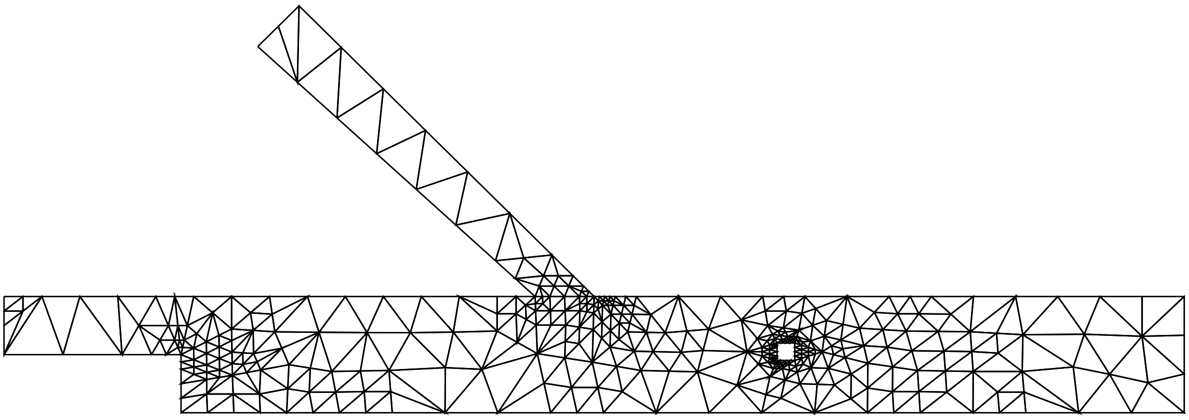

Fig. 4. Refined mesh obtained using the error indicator ϵ_2.

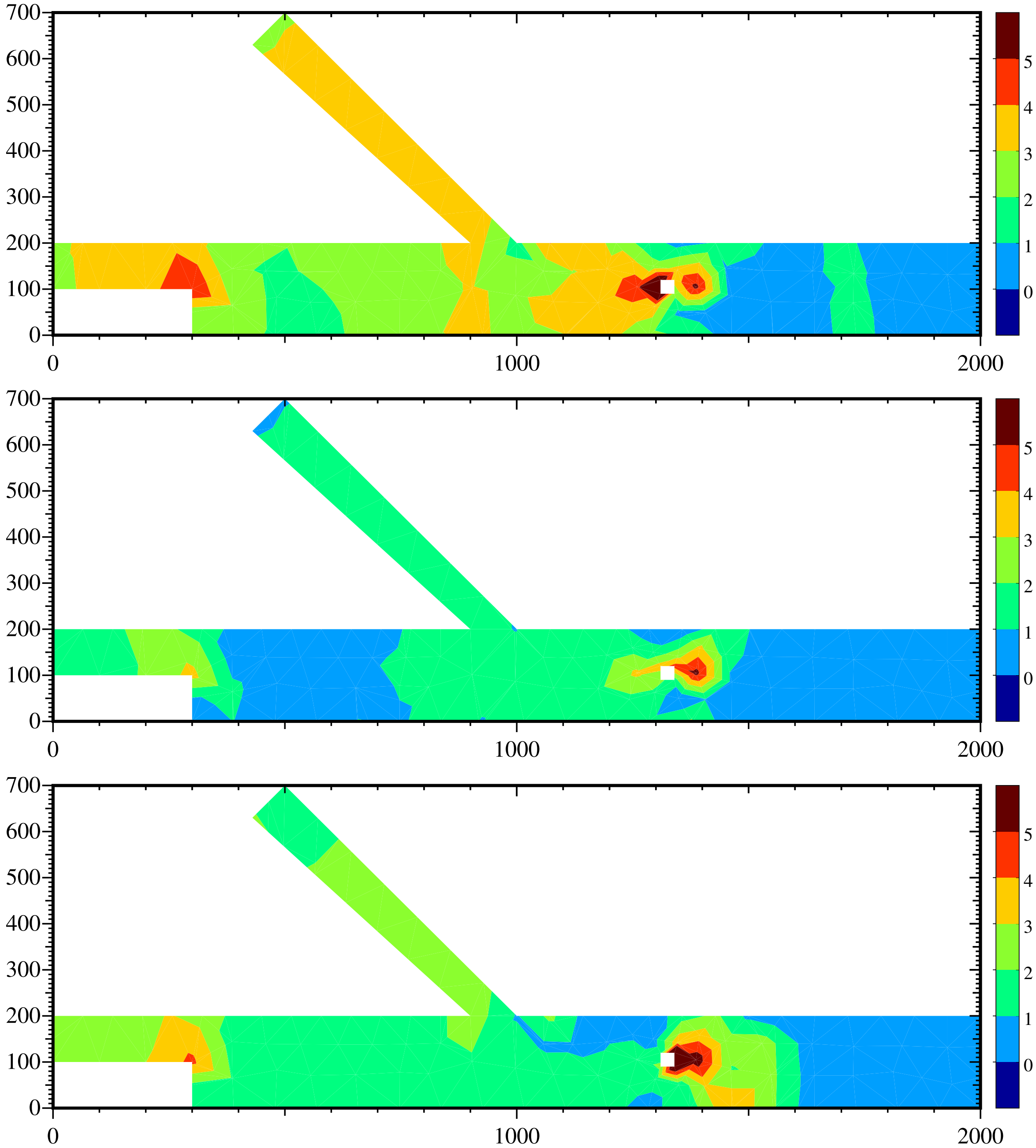

Fig. 5. Relative error in the water elevation value obtained by using the ϵ_1 and the ϵ_2 error indicators and by uniformly refining, two times, the background mesh.

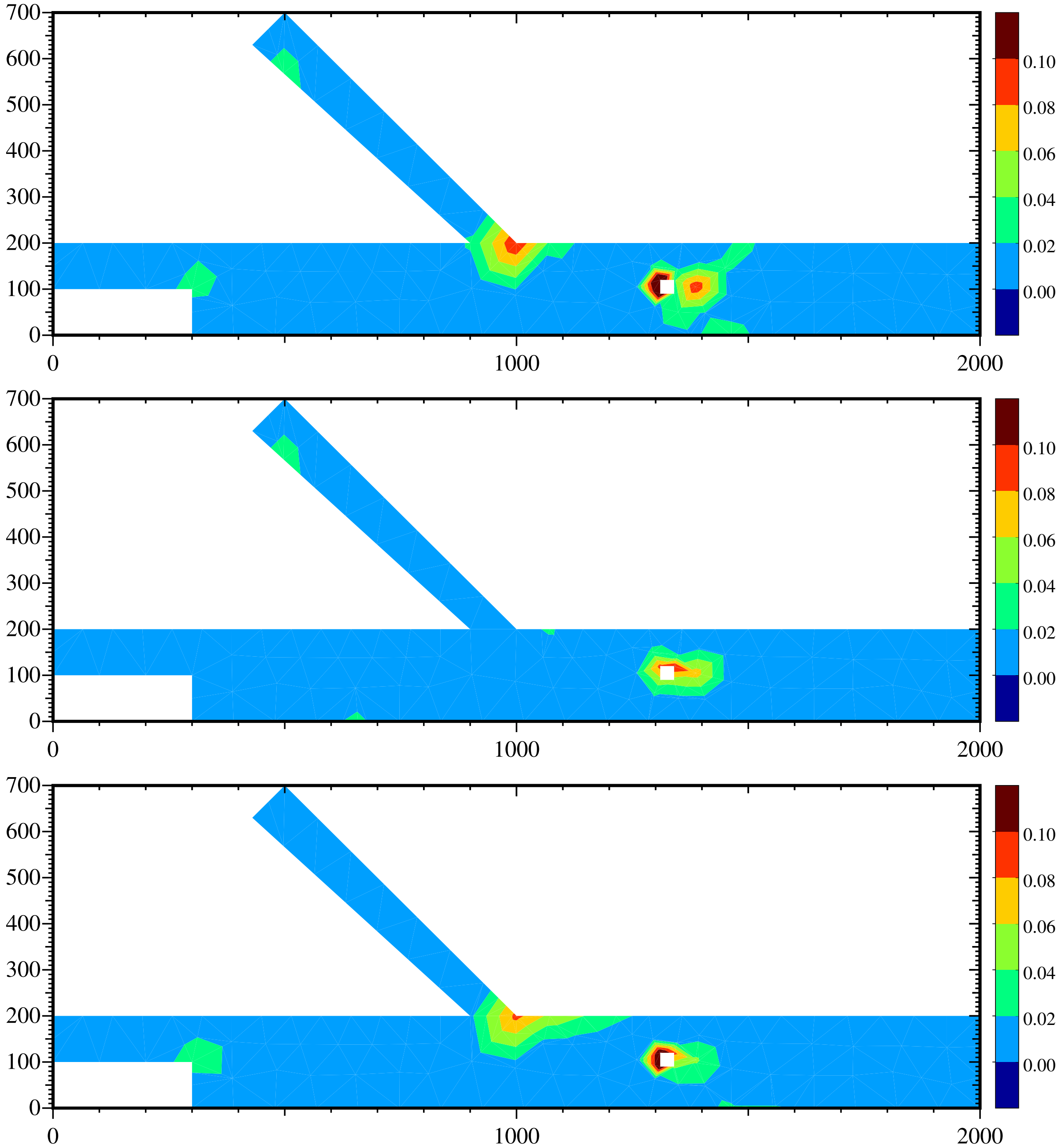

Fig. 6. Absolute error in the water speed obtained by using the ϵ_1 and the ϵ_2 error indicators and by uniformly refining, two times, the background mesh.

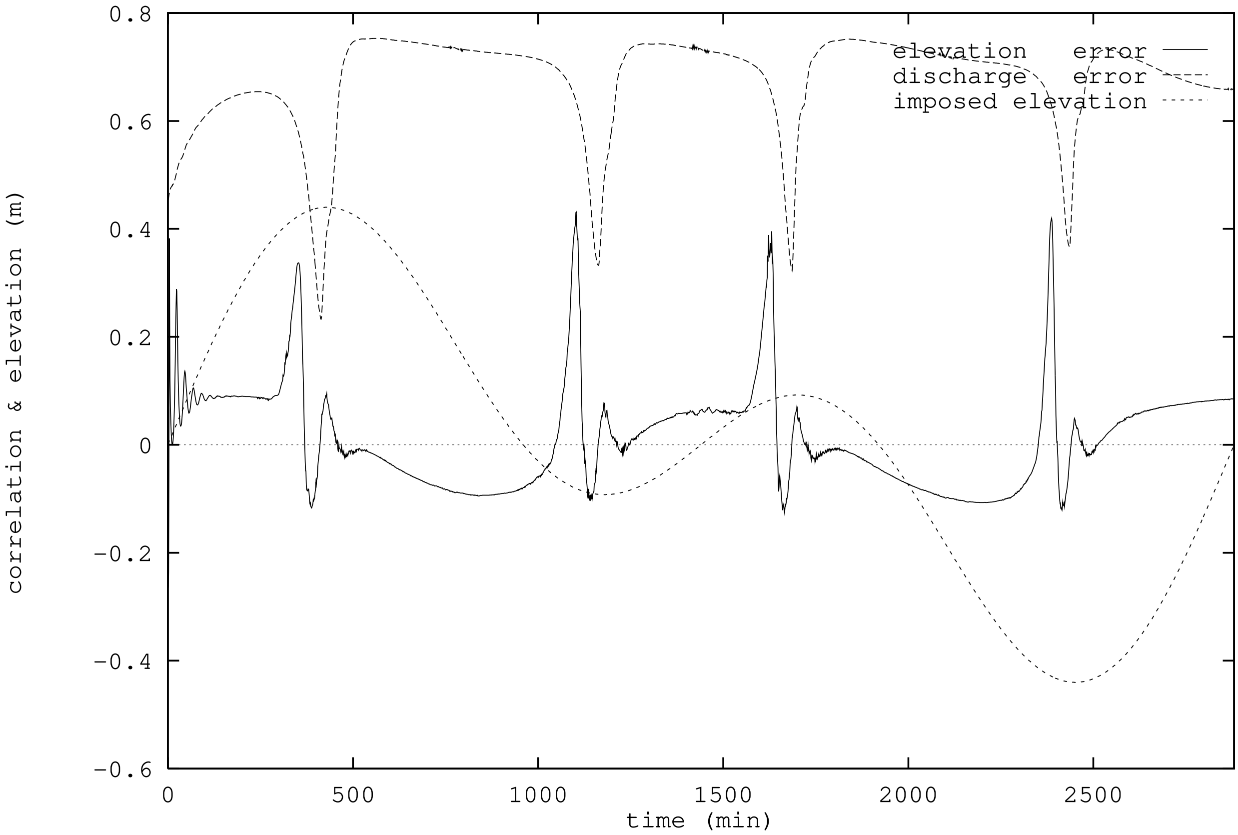

Fig. 7. Spatial correlation between the ϵ_3 error estimators and the error as a function of the integration time. Dotted line is the imposed elevation expressed in meters.

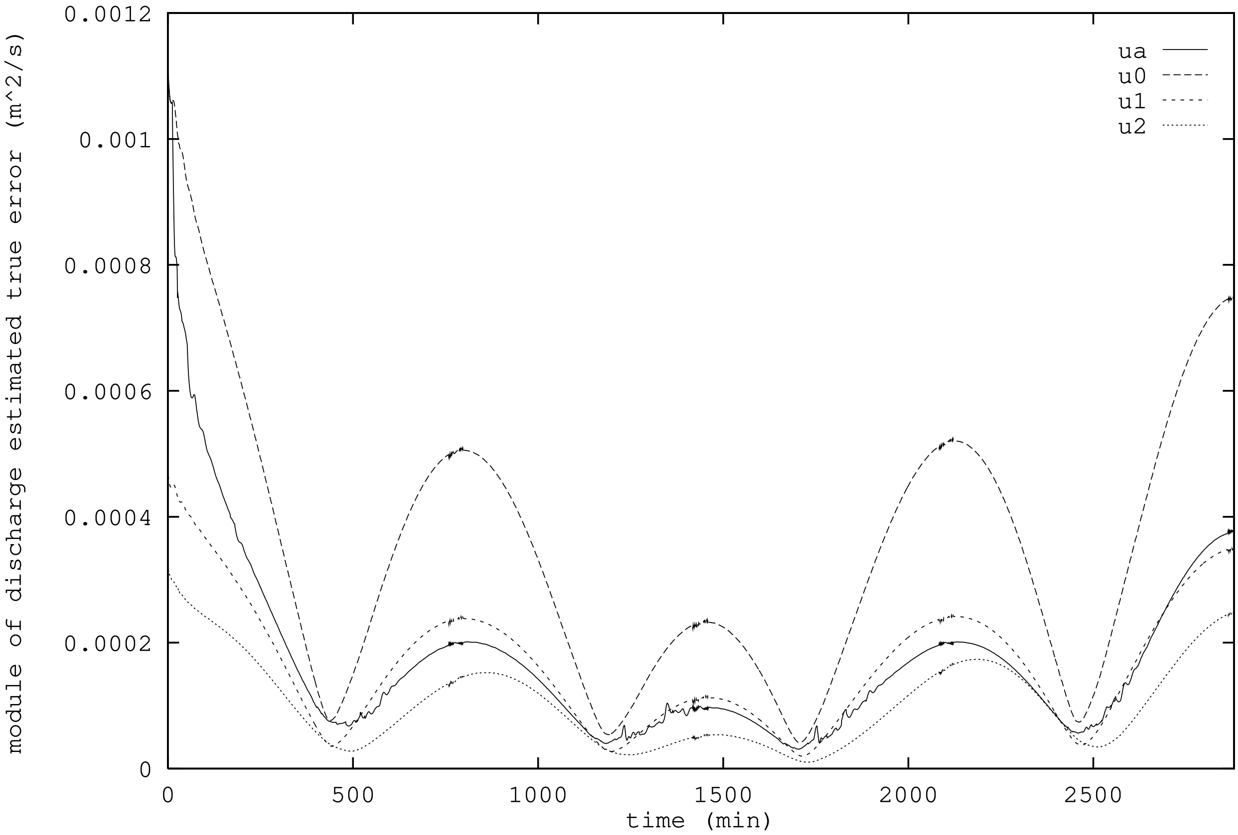

Fig. 8. Average error of the discharge magnitude as a function of the integration time; solutions u_0, u_1, u_2, u_a.

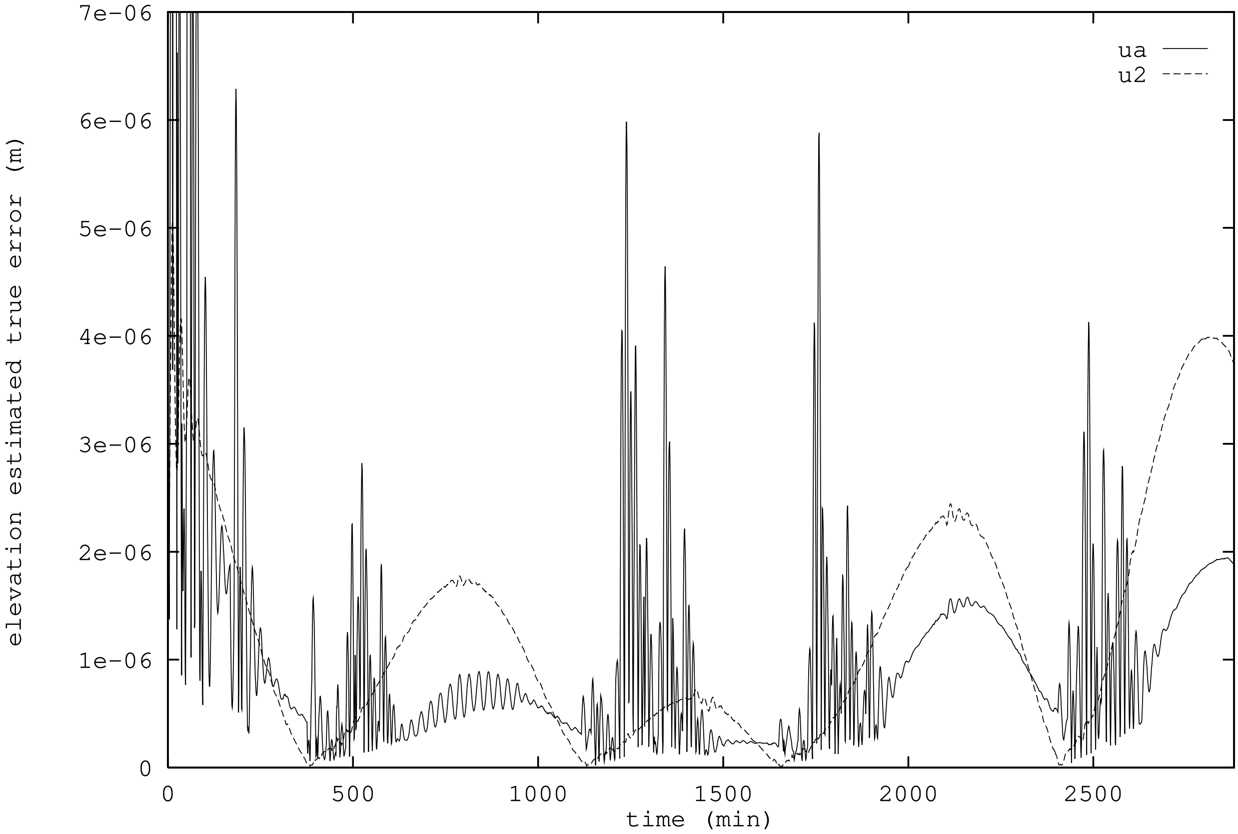

Fig. 9. Average error of the elevation as a function of the integration time; solutions u_2 and u_a.

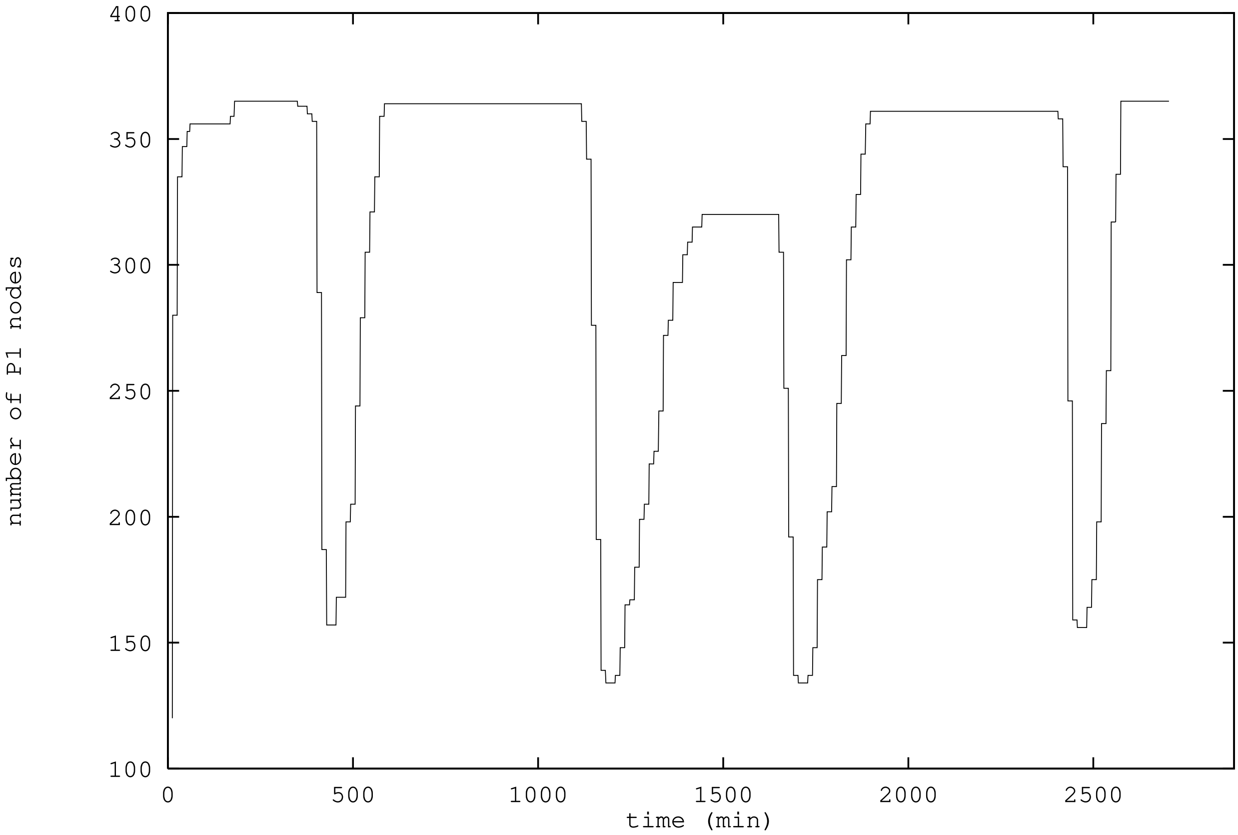

Fig. 10. Number of vertices of the grid as a function of the integration time using the adaption.