Pergamon

Computers & Fluids Vol. 27, No. 4, pp. 479–508, 1998
© 1998 Elsevier Science Ltd. All rights reserved
Printed in Great Britain
0045-7930/98 $19.00 + 0.00

PII: S0045-7930(97)00071-6

AN UNSTRUCTURED FINITE-VOLUME ALGORITHM FOR PREDICTING FLOW IN RIVERS AND ESTUARIES

P. A. SLEIGH[1], P. H. GASKELL[2], M. BERZINS[3] and N. G. WRIGHT[4]

[1]School of Civil Engineering, University of Leeds, Leeds, LS2 9JT U.K.
[2]School of Mechanical Engineering, University of Leeds, Leeds, LS2 9JT, U.K.
[3]School of Computer Studies, University of Leeds, Leeds, LS2 9JT U.K.
[4]University of Nottingham, Department of Civil Engineering, Nottingham, NG7 2RD U.K.

(Received 3 February 1997; in revised form 11 September 1997)

Abstract—A numerical algorithm is presented for the solution of geometrically challenging two-dimensional river and estuary flows, based on an adaptive triangular tessellation of the flow domains of interest. The governing, shallow water, equations are discretised using a finite volume approach embodying variable step time integrators, to yield a method that is second order accurate in both space and time. An approximate Riemann solver is used to determine flow directionality in conjunction with an effective means of dealing with wetting and drying at the boundaries.

The approach is capable of handling complex flow domains and yielding solutions for which errors are controlled automatically by the use of spatial re-gridding and time stepping based on local error estimates. Its range of applicability is demonstrated through considering several problems involving super/sub-critical flow, wetting/drying, culminating in the solution of a complete estuary problem. © 1998 Elsevier Science Ltd. All rights reserved

1. INTRODUCTION

There has been considerable interest, of late, in modelling fluid flow in rivers and estuaries, much of which has been driven by increased public awareness of pollution and environmental issues in relation to construction projects, coastal defences, the dumping of effluent into rivers and the sea, discharges from power stations and the processing industries, flooding and radioactive waste disposal. The equations which govern the motion of such flows can be solved analytically for only the very simplest of problems—for flows of practical interest numerical methods offer the only viable way forward. The principal challenges the latter presents are two-fold: the capacity to resolve flow in convoluted topologies coupled with the capability to solve problems for a wide range of boundary and initial conditions.

In addition, any generally applicable numerical approach should embody the features necessary to: handle both steady and transient flow conditions (smooth or otherwise); describe and incorporate complex topography; simulate both sub- and super-critical conditions; accommodate flow around and through structures such as weirs and gates; allow for the wetting/drying of flood planes and river beds; enable inflow/outflow conditions to be specified at different points in the domain of interest. More exacting criteria involve: correct embodiment of the underlying physics; quantifiable accuracy and speed of computation.

A major issue is that of domain decomposition in the context of the topologies encountered in practise; although there are clearly difficulties associated with adopting a rectangular grid system for this purpose it has remained a popular choice, forming the basis of a number of widely used algorithms [1]. Another approach is to generate a computational grid based on a system of curvilinear coordinates [2–4]—the governing equations need to be transformed accordingly but the advantage is better geometric definition while retaining ease of discretisation. Molls and Chaudhry [5], have applied this methodology with apparent success to a number of challenging, steady-state problems, including transitional flow. However, they fail to disclose the nature of the computational grids employed, nor do they consider the solution of topologically complex flows when difficulties are likely to arise in regions where the grid becomes highly skewed.

A preferable alternative is to adopt an irregular, unstructured grid system, typical of those employed in finite element analyses. Recent work in the area has yielded some rather impressive results using TELEMAC [6,7], the crux of which has been the attainment of accurate descriptions of complex flow domains via triangulation. A key feature of the present work is to retain this flexibility but within a finite volume framework which experience dictates is much quicker to compute. On a different note, Garcia-Navarro, Hubbard and Priestly [8] introduced a novel finite volume formulation of the shallow water equations which, it is claimed, is 'genuinely' two-dimensional in the sense that the equations are never transformed to cell face normal directions in order to obtain fluxes. However, the method is rather more complex than the traditional finite volume approach and it remains unclear from their preliminary work whether the extra effort required gives way to improved solutions.

Until quite recently, the tendency has been to employ a non-conservative form of the shallow water equations to compute hydraulic flow models, a practice that should be avoided—when a problem involving rapidly changing transitional flow is encountered sharp fronts and rapid changes in velocity occur, resulting in these extreme conditions being predicted incorrectly. However, such features need no longer be afflicted in this manner since considerable effort has been expended of late (inspired by success in the field of aeronautics) in the adoption of shock capturing methods for the solution of the conservative form of the shallow water equations [9–15] and [8]. The latter enables a Riemann solver and flux limiter to be used within a finite volume formulation, a combination that offers the tantalising prospect of attaining solutions to complex problems on fully unstructured grids with accurate description and resolution of extreme conditions such as hydraulic jumps and bores.

The resolution of steep gradients can be improved further by employing mesh adaption—that is, increasing the mesh density in desired regions of the flow. While this can prove problematic within finite difference and finite element analyses, finite volume methods—particularly those based on a triangular mesh system—lend themselves quite naturally to automatic adaptive refinement/de-refinement procedures [16,17].

There is very little in the open literature concerning the application of these recent advances to general hydraulic flow problems. The exceptions are Yang and Hsu [13] who report results for the formation of shocks around a cylinder in a contraction, and Zhao *et al.* [1] who appear to have been the first to tackle a problem of practical interest—that of a river and flood plane. In their work a Riemann solver is used to form the fluxes on an unstructured, mixed quadrilateral-triangular grid system. The results are extremely encouraging with numerical predictions and field measurements comparing well for the steady-state problem investigated. The major drawbacks are that the associated solution times are seen to be rather excessive—a consequence of the CFL limit imposed by the explicit nature of their scheme—while the spatial discretisation procedure adopted is first order only.

The motivation for the present work is to improve upon every facet of the above and produce an algorithm with a wide range of practical applicability. More specifically the method:

- is centred on the conservative form of the shallow water equations;
- employs an finite volume formulation, the unstructured computational grid for which is generated automatically by triangular decomposition of the region of interest; any associated digital bathymetry data is similarly interpolated automatically;
- uses an approximate Riemann solver to calculate the convective fluxes together with an associated flux limiter which is monotonicity preserving;
- makes use of second order spatial and temporal discretisation;
- embodies the scope for automatic spatial and temporal error control;
- has the flexibility to specify general boundary conditions, including the capacity to handle wetting/drying characteristics.

The ethos which underpins the scheme is described fully in Sections 2 and 3. This is followed (see Section 4) by a description of how wetting/drying type boundary conditions are assimilated with application to two demanding one-dimensional test problems for which accurate results already exist for the purpose of comparison. In Section 5 the complete algorithm is used to solve a number of two-dimensional problems and comparisons drawn between existing experimental data and corresponding theoretical predictions. Conclusions follow in Section 6.

2. GOVERNING EQUATIONS AND DOMAIN DECOMPOSITION

Each of the problems considered later is modelled in terms of the two-dimensional shallow water equations, containing source terms representing frictional stress and momentum change due to a sloping bed, which when written in conservative form can be expressed as

$$U_t + E_x + H_y = S(U), \tag{1}$$

where subscripts t, x and y denote first derivatives with respect to these parameters, and

$$U = \begin{bmatrix} \phi \\ \phi u \\ \phi v \end{bmatrix}, E = \begin{bmatrix} \phi u \\ \phi u^2 + \frac{1}{2}\phi^2 \\ \phi uv \end{bmatrix}, H = \begin{bmatrix} \phi v \\ \phi uv \\ \phi v^2 + \frac{1}{2}\phi^2 \end{bmatrix}, S = \begin{bmatrix} 0 \\ g\phi(Sf_x + So_x) \\ g\phi(Sf_y + So_y) \end{bmatrix} \tag{2}$$

where u, v are the velocities in the x, y directions respectively; $\phi = gh$, h is the depth, g is the acceleration due to gravity, $So_{x/y}$ and $Sf_{x/y}$, are the bed slope and friction slope in the x/y direction, respectively. The latter is found using either the Manning or (equivalent) De Chezy formulae

$$Sf_x = \frac{n^2 u\sqrt{u^2+v^2}}{(\phi/g)^{4/3}} = \frac{u\sqrt{u^2+v^2}}{C^2(\phi/g)}, \tag{3}$$

where n is the Manning n and C the De Chezy C [18].

2.1. Mesh generation

The type of mesh employed is an unstructured triangulation of the solution domain enabling arbitrary shaped geometries to be accommodated more easily than with a square grid system. The production of a suitable mesh can be a complex and time consuming affair were it not for the many algorithms now available which generate automatically an initial mesh with the required properties. The property most desired is that the mesh is smooth, that is, that adjacent triangles do not have greatly differing qualities and that none of the triangles have a quality less than a prescribed value. Consequently, use was made of the GEOMPACK [19] mesh generator. GEOMPACK allows the mesh to be distributed as required via a function defining a weight between 0 and 1 for any point within the solution domain. The higher the function value the greater the mesh density. The reader is referred to Ref. [19] for a more detailed discussion of the consequence of using this function.

The above was interfaced to software requiring only an outline of the full solution domain and a minimum quality of triangle to be supplied. Although this in itself provides sufficient information to construct a finite volume solution scheme, it was further processed using TRIAD [20] to create data structures allowing efficient manipulation of the mesh, in particular adaption—refinement/de-refinement—throughout the solution domain.

3. METHOD OF SOLUTION

Construction of a suitable numerical algorithm for the solution of Equation (1) necessitates a firm base upon which to build. The one presented here is centred around SPRINT2D [20–22], software which utilises triangular domain decomposition and a cell centred finite volume formulation, with numerical flux determined by the solution of a local Riemann problem. This method enables accurate solutions to be found for both smooth and discontinuous flow problems. It allows flexibility of definition of problems and boundary conditions by passing control of the flux calculation, initial conditions, source term and boundary specification to external routines while performing the integration routine within. Thus the most complex of problem specification can be achieved—including wetting/drying—for flows involving tributaries, weirs, rating curves etc.

A key feature of SPRINT2D is its ability to quantify the solution error and to control it as necessary. Error control is complex with balance necessary between those arising from both temporal and spatial integration. SPRINT2D contains methods that control each interdependently—while the mesh is allowed to refine/de-refine, the time step is governed by the temporal integration procedure [21]. Full control of this adaption is allowed via external routines through the setting of error scaling and mesh adaption control parameters.

3.1. Discretisation procedure

A cell centred finite volume method is formulated for Equation (1) over a triangular shaped control volume with the dependent variables of the system represented as piecewise constants. The association of these variables with particular points enables the use of a high-order interpolation scheme—see later. Integrating Equation (1) over the *i*th triangle gives

$$\int_{A_i} \frac{\partial U}{\partial t} d\Omega - \int_{A_i} S(U) d\Omega = -\int_{A_i} (E(U)_x + H(U)_y) d\Omega, \tag{4}$$

where A_i is the area of the triangle and Ω is the integration variable defined on A_i. The area integrals on the left hand side are approximated by a one-point quadrature rule, the quadrature point being the centroid of the triangle. Using the divergence theorem, the right hand side of Equation (4) can be replaced by a line integral around the bounding volume, namely

$$A_i \frac{\partial U_i}{\partial t} = -\oint_{\Gamma_i} \big(E(U) n_x + H(U) n_y\big) ds + A_i S(U_i), \tag{5}$$

or

$$A_i \frac{\partial U_i}{\partial t} = -\oint_{\Gamma_i} F_n(U) ds + A_i S(U_i), \tag{6}$$

where Γ_i is the perimeter of the *i*th triangle, $F_n(U)$ is the normal flux vector and s is the integration variable along the perimeter; {n_x and N_y are the components of unit normal in the x and y directions respectively. The line integral is evaluated via a mid-point quadrature rule, that is, the numerical flux is calculated at the mid-point of each edge, giving

$$\frac{\partial U_i}{\partial t} = -\frac{1}{A_i} \left[F_n(U)_{ik} . l_{ik} + F_n(U)_{ij} . l_{ij} + F_n(U)_{il} . l_{il} \right] + S(U). \tag{7}$$

Referring to Fig. 1, side *ij* is common to the triangles associated with U_i and U_j, l_{ij} is the length of side *ij* and $F_n(U)_{ij}$ the flux in the outward (from the triangle associated with U_i) normal direction evaluated at the midpoint of this edge; these are defined similarly for sides *ik* and *il*.

3.2. Normal flux calculation: the Riemann approach

A feature of the shallow water equations, which is very useful when developing an unstructured finite volume scheme, is that they are rotationally invariant enabling problems to be posed as locally one-dimensional. Evaluation of the normal flux in Equation (7) is made by a series of solutions local to the lines which make up the triangular mesh. The Riemann problem is defined by the solutions on the left and right of the cell face (or internal and external to the finite volume), and the order of the numerical scheme is determined by the definition of these two data states.

If ($\bar{x}$, $\bar{y}$) is a local coordinate system centred at the mid-point of the cell face in question, with $\bar{x}$ in the normal outward direction and $\bar{y}$ lying tangentially, the condition for rotational invariance is

$$T F_n(U) = E(TU) = E(\overline{U}), \tag{8}$$

where

$$T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & n_x & n_y \\ 0 & -n_y & n_x \end{bmatrix}. \tag{9}$$

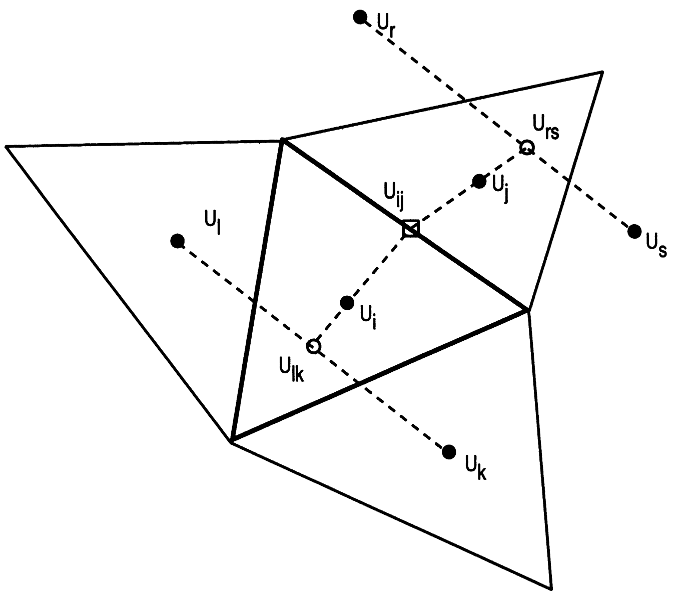

Fig. 1. Construction of U^L and U^R interpolants. ●, Centroid solution values; □, Midpoint of edge; ○, Interpolated solution values.

The velocities in the $\overline{x}$ and $\overline{y}$ directions are $\overline{u}=u_n=un_x + vn_y$ and $\overline{v}=u_t=-un_y + vn_x$, respectively.

In this local coordinate system the one-dimensional problem is then

$$\overline{U}_t + [E(\overline{U})]_{\overline{x}} = 0, \tag{10}$$

which can be viewed as simply a Riemann problem—an initial value problem with discontinuous initial conditions on either side of the line, that is

$$\overline{U}(\overline{x},0) = \begin{cases} \overline{U}_L & \overline{x}<0 \\ \overline{U}_R & \overline{x}\geq 0 \end{cases}. \tag{11}$$

The approximate solution to this problem allows the correct flux value, $E(\overline{U})$, in Equation (10), to be computed which is then transformed, via Equation (8), to give the desired flux, $F_n(U)$.

Many Riemann solvers exist for application to the shallow water equations—most having been derived from versions used in gas dynamics—those of Roe [23] and Osher [24] being examples which have met with considerable success [25]. Toro [11] identifies several others specific to the shallow water equation, including an exact solution. In this study each of the above solvers were examined for both test and practical flow simulations in one- and two-dimensions. While most proved satisfactory—producing very similar solutions—the solver due to Roe was consistently more stable, providing solutions under extreme conditions where others failed.

The normal flux at the face of a control volume can be expressed [9] as

$$F_n = \frac{1}{2}\big(F_L + F_R - |A|(U_L - U_R)\big), \tag{12}$$

where $F_{L/R}$, $U_{L/R}$ are the fluxes and the solution vector on the left and right sides of the face respectively, and A is a Jacobian matrix given by

$$A = \frac{\partial F_n}{\partial U} = \left\{ \begin{array}{ccc} 0 & n_x & n_y \\ (c^2 - u^2)n_x - uvn_y & 2un_x - vn_y & un_y \\ -uvn_x + (c^2 - v^2)n_y & vn_y & un_x + 2vn_y \end{array} \right\}. \tag{13}$$

Following Roe [23] an equivalent [26] but linear system for the governing equations can be defined using an averaged matrix $\tilde{A}$ with eigenvalues

$$\tilde{\lambda}_1 = \tilde{u}n_x + \tilde{v}n_y + \tilde{c},\ \tilde{\lambda}_2 = \tilde{u}n_x + \tilde{v}n_y,\ \tilde{\lambda}_3 = \tilde{u}n_x + \tilde{v}n_y - \tilde{c}, \tag{14}$$

and eigenvectors

$$\tilde{e}_1 = (1, \tilde{u} + \tilde{c}n_x, \tilde{v} + \tilde{c}n_y)^T,\ \tilde{e}_1 = (0, -\tilde{c}n_y, \tilde{c}n_x)^T,\ \tilde{e}_1 = (1, \tilde{u} - \tilde{c}n_x, \tilde{v} - \tilde{c}n_y)^T, \tag{15}$$

where the averages are defined as

$$\tilde{u} = \frac{u_R\sqrt{\phi_R} + u_L\sqrt{\phi_L}}{\sqrt{\phi_R} + \sqrt{\phi_L}},\ \tilde{v} = \frac{v_R\sqrt{\phi_R} + v_L\sqrt{\phi_L}}{\sqrt{\phi_R} + \sqrt{\phi_L}},\ \tilde{c} = \frac{\sqrt{\phi_L + \phi_R}}{2},\ \tilde{\phi} = \sqrt{\phi_L\phi_R}. \tag{16}$$

where $\Phi_{L/R}$ are the values of Φ to the left and right sides of the face. The normal flux can then be written as

$$F_n = \frac{1}{2}\left(F_L + F_R - \sum_{k=1}^{3}\tilde{a}_k|\tilde{\lambda}_k|\tilde{e}_k\right), \tag{17}$$

where the $\tilde{a}$'s are the wave strengths given by

$$\tilde{a}_1 = \frac{1}{2}\Delta\phi + \frac{1}{2}\tilde{\phi}\Delta u/\tilde{c},\ \tilde{a}_2 = \tilde{\phi}\Delta v,\ \tilde{a}_3 = \frac{1}{2}\Delta\phi - \frac{1}{2}\tilde{\phi}\Delta u/\tilde{c}. \tag{18}$$

Hence the change in state across a wave is the product of the wave strength and the component of the right eigenvector.

3.3. Spatial integration

For a first order scheme the values of the variables to the left and right of side ij are $U_{L_{ij}} = U_i, U_{R_{ij}} = U_j$, the solution at the centre of adjacent cells. However, by employing an upwind weighted linear interpolation function second order accuracy can be achieved, see Berzins and Ware [22]. While second order schemes in general can exhibit under and over shoots in regions of steep gradients this method has been shown to allow only physically realistic values to be computed by means of limiting the gradients in the linear interpolants. It is worth noting that a careful choice of limiter is required to ensure positivity on irregular triangular meshes [22].

Accordingly, the limited values on the left and right of face ij (see Fig. 1) are given by

$$U_{L_{ij}} = U_i + \left(U^L_{ij} - U_i\right)\Phi\left(r^l_{ij}\right), \tag{19}$$

$$U_{R_{ij}} = U_j + \left(U^R_{ij} - U_j\right)\Phi\left(r^r_{ij}\right), \tag{20}$$

Here Φ is a suitable limiter, such as the modified Van Leer [22] one, where

$$\Phi(R) = \frac{R + |R|}{1 + \max(1,|R|)}, \tag{21}$$

r^l_{ij} and r^r_{ij} are internal and external upwind bias ratios of gradients

$$r^l_{ij} = \frac{U^C_{ij} - U_i}{U^L_{ij} - U_i},\ r^r_{ij} = \frac{U^C_{ij} - U_j}{U^R_{ij} - U_j}. \tag{22}$$

while U^L_{ij} and U^R_{ij} are internal and external linear upwind values, given by

$$U^L_{ij} = U_i + d_{ij,i}\frac{U_i - U_{lk}}{d_{i,lk}},\ U^R_{ij} = U_j + d_{ij,j}\frac{U_j - U_{rs}}{d_{j,rs}}. \tag{23}$$

The linear centred value at the cell interface, denoted by U^C_{ij}, is constructed from the six 'surrounding' triangles and is, for edge ij, either an interpolation between U_{ij_c} and U_{ks} or between

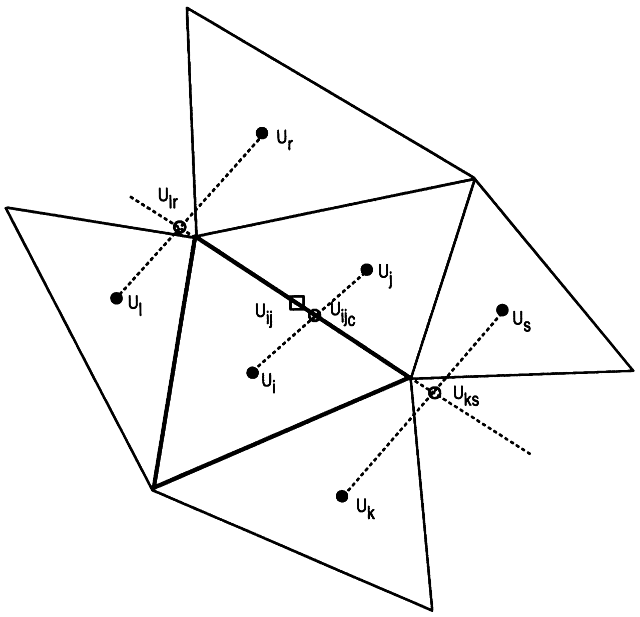

Fig. 2. Construction of U^C and U^R interpolant. ●, Centroid solution values; □, Midpoint of edge; ○, Interpolated solution values.

U_{ij_c} and U_{lr}, depending on which pair encompass the midpoint. This can be seen from Fig. 2 in which case the interpolation is between U_{ij_c} and U_{ks}. The term $d_{i,ij}$ denotes the distance between points, for example $d_{i,ij} = \sqrt{(x_i - x_{ij})^2 + (y_i - y_{ij})^2}$, while the values U_{lk} and U_{rs} are intermediate, linearly interpolated, quantities which can be most clearly seen by examining Fig. 1. For certain meshes the three centroid values may be collinear in which case the immediate upwind centroid value is used.

3.4. Time integration

Equation (7) may be rewritten as a system of ordinary differential equations

$$\frac{\partial U_i}{\partial t} = G_{N,i}(t, U(t)), \tag{24}$$

where $U(t) = U_1(t),..., U_N(t)$, and integrated numerically to give an approximation, $V(t)$, to the vector of exact solution values, $U(t)$. The difference between these two vectors is the global time integration error due to the numerical integration method applied. SPRINT2D contains a number of time integration modules—both explicit and implicit. The Theta method used here has automatic local error control, which when applied with local mesh refinement/de-refinement (see Section 3.5) enables a balance between spatial and temporal errors to be found, allowing optimum accuracy [21]. The solution at time $t_{n+1} = t_n + \kappa$, where κ is the time step, is written as

$$V(t_{n+1}) = V(t_n) + (1-\theta)\kappa \dot{V}(t_n) + \theta\kappa G_N\big(t_{n+1},\ V(t_{N+1})\big), \tag{25}$$

in which $V(t_n)$ and $\dot{V}(t_n)$ is the numerical solution and its time derivative at time level n, respectively. The value of θ lies in the range $\theta \in [0.5, 1.0]$ and can be set automatically within the software—a more common practise is to pre-set θ explicitly at a value of 0.55. The size of the time step κ is chosen to satisfy a local error control, as described in the next subsection, which may reflect a measure of spatial error.

The equation set can be solved within SPRINT2D by either a Newton–Krylov technique or functional iteration—see Berzins [21,22] for details. Functional iteration gives

$$V^{m+1}(t_{n+1}) = V(t_n) + (1-\theta)\kappa \dot{V}(t_n) + \theta\kappa G_N\big(t_{n+1},\ V^{(m)}(t_{n+1})\big). \tag{26}$$

where $m = 0,1,...,$ is generally less than 3, and use is made of a second order predictor or one based on the forward Euler method

$$V^{(0)}(t_{n+1}) = V(t_n) + \kappa G_N\big(t_n,\ V(t_n)\big). \tag{27}$$

The condition for functional iteration to converge with a rate of convergence, r_c, is that

$$\kappa\theta\|J\|<r_c \text{ where } J = \frac{\partial G_N}{\partial u}. \tag{28}$$

Berzins [21] shows that a CFL type stability condition is satisfied automatically if functional iteration converges sufficiently fast.

With the Newton–Krylov approach Equations (25) are solved for a correction, $\underline{\Delta V}$, to the solution. So for the $(m + 1)$th iteration of the modified Newton method

$$[I - \kappa\theta J]\Delta V = \underline{r}\big(t^m_{n+1}\big), \tag{29}$$

where

$$\underline{r}\big(t^m_{n+1}\big) = -V\big(t^m_{n+1}\big) + V\big(t_n\big) + (1-\theta)\kappa\dot{V}(t_n) - \theta\kappa G_n\big(t_{n+1},\ V\big(t^m_{n+1}\big)\big), \Delta V = \big[V\big(t^{m+1}_{n+1}\big) - V\big(t^m_{n+1}\big)\big]. \tag{30}$$

The solution of the system of Equations (26) or Equations (29) constitutes the major computational task. Numerical tests with both methods confirmed the results of Berzins and Furzeland [27] that functional iteration can be as much as four times faster than multidimensional Newton–Krylov methods for convection dominated problems of the type considered here.

3.4.1. Stability and time error control. Efficient time integration requires that the spatial and temporal errors are roughly the same order of magnitude. The need for spatial error estimates unpolluted by temporal error, requires that the spatial error is the larger of the two. Although one way of achieving this might be to use the approach of Berzins [21] which controls the local time error $\underline{l}_{n+1}(t_{n+1})$ to be a fraction of the growth in the spatial discretisation error over a time-step a standard local error approach was adopted in the present work, given by

$$\|\underline{l}_{n+1}(t_{n+1})\| < TOL, \tag{31}$$

where $\underline{l}_{n+1}(t_{n+1})$ is the local time incurred on the time step to t_{n+1}. TOL is a user-supplied tolerance and the time step limited both by imposing an explicit maximum step size and that functional iteration converges sufficiently fast.

The above approach is necessary because the problem of establishing a stable step size limit analytically, for time integration of the system of partial differential equations given by Equation (1), is non-trivial in the case of an unstructured mesh system. This is easily demonstrated by considering the following model advection equation

$$U_t + aU_x + bU_y = 0, \tag{32}$$

which when discretised in a finite volume sense leads to a system of ordinary differential equations, see Berzins [22]

$$G_i\big(t_n, V(t_n)\big) = -\tilde{a}_i\ V_i(t_n) + S^i_N\big(V(t_n)\big), \tag{33}$$

where $S^i_N\big(V(t_n)\big) = \sum_{j\neq i} c_{ij} V_j(t_n)$, $c_{ij} \geq 0$, thus making $S^i_N\big(V(t_n)\big)$ a positive function for positive values of $V(t_n)$. The coefficient $\tilde{a}_i$ plays a key role in the stability analysis, Berzins and Ware [22], in that the stability condition for flow alone is

$$1 - \theta\kappa\, m\, \tilde{a}_i \geq 0, \tag{34}$$

where m is the number of functional iterations. The coefficient $\tilde{a}_i$ is defined by considering the discretised form of Equation (32) which, for a triangular control volume, is

$$\frac{du_i}{dt} = -\frac{1}{A_i}\big(au^l_{ik}\Delta y_{0,1} - bu^r_{ik}\Delta x_{0,1} + au^l_{ij}\Delta y_{1,2} - bu^l_{ij}\Delta x_{1,2} + au^r_{il}\Delta y_{2,0} - bu^l_{il}\Delta x_{2,0}\big), \tag{35}$$

where, subscripts to the Δx and Δy terms indicate the vertices of the triangle, which are numbered in an anti-clockwise sense, such that

$$\Delta x_{0,1} + \Delta x_{1,2} + \Delta x_{2,0} = \Delta y_{0,1} + \Delta y_{1,2} + \Delta y_{2,0} = 0. \tag{36}$$

Consequently, the right side of Equation (35) may be rewritten as,

$$-\frac{1}{A_i}\big(a\big(u^l_{ik} - u^r_{il}\big)\Delta y_{0,1} + a(u^l_{ij} - u^r_{il})\Delta y_{1,2} - b(u^l_{ij} - u^r_{il})\Delta x_{1,2} - b(u^l_{il} - u^r_{ik})\Delta x_{2,0}\big). \tag{37}$$

Consider each term of the form $(u^l_{ij} - u^r_{il})$ independently; following the analysis in Berzins and Ware [22], their Equation (42) shows that the coefficient of u_{ij} in this term, denoted by $z_{ij,lk,il}$, may be defined as

$$z_{ij,lk,il} = 1 + \Phi(S_{ij})\frac{d_{ij,i}}{d_{i,lk}} - \frac{\Phi(R_{il})}{R_{il}}\gamma_{il}, \tag{38}$$

where $\gamma_{il} \leq 1$. Defining other coefficients of $z_{.,.,.}$ in the same way gives

$$\tilde{a}_i = \frac{1}{A_i}\big[a\, z_{ik,lj,il}\Delta y_{0,1} + a\, z_{ij,lk,il}\Delta y_{1,2} - b\, z_{ij,lk,ik}\Delta x_{1,2} - b z_{il,kj,ik}\Delta x_{2,0}\big] \tag{39}$$

The above expression can be bounded by noting from Equation (21) that $\Phi(.) \in [0,2]$ and hence

$$z_{ij,lk,il} \leq z_{\max} = 1 + 2d_{\max}, \tag{40}$$

where d_{max} is the maximum value of the ratio of distances, such as $d_{ij,i}/d_{i,lk}$ in Equation (38). Hence using Equations (33) and (35)

$$\tilde{a}_i \leq \frac{-1}{A_i} z_{\max}\big[-a\Delta y_{2,0} + b\Delta x_{0,1}\big], \tag{41}$$

where $-\Delta y_{2,0} \geq 0$. Now if L_i is the length of the longest edge of triangle i, a sufficient condition for positivity is

$$z_{\max}\kappa\frac{L_i}{A_i}(a + b) \leq \frac{1}{\theta m}. \tag{42}$$

The result is a CFL type stability condition which depends on the term L_i/A_i, which is also used as a measure of the quality of a triangle. Although only one possible alignment to the characteristic directions has been considered a similar result is obtained for other alignments.

It is the difficulty of evaluating Equation (42) that leads one to adopt the approach of imposing stability through requiring the fast convergence of functional iteration—see also Berzins [21]. For the model problem, Equation (32), the condition for convergence via functional iteration in this case, with matrix norm $||.||_\infty$, is (ignoring the dependence of $\tilde{a}$ and c_{ij} on the solution) from Equation (33), approximately

$$\kappa\theta\left\{|\tilde{a}_i| + \sum_{j\neq i}|c_{ij}|\right\} = \lambda r_c, \tag{43}$$

for $0 < \lambda < 1$. This expression may be substituted in Equation (34) to give

$$\begin{aligned} 1 - \theta\kappa\, m|a_i| &= 1 - (\lambda r_c - \kappa\theta\sum_{j\neq i}|c_{ij}|)m, \\ &= 1 + \kappa\theta\sum_{j\neq i}|c_{ij}|m - \lambda r_c m. \end{aligned} \tag{45}$$

Hence for a rate of convergence of functional iteration, r_c, that is sufficiently small, condition (34) holds.

3.5. Error measurement and adaptation

At each time step, t_{n+1}, for each triangular area, a local error estimate of the temporal growth in the spatial error is calculated from the difference between the solution found at t_{n+1} with a first order spatial interpolation and that for a second order spatial interpolation scheme for each partial differential equation in the system. The precise form of the error estimate used and its justification can be found in Berzins [21,22]. In order to obtain a measurement suitable for use as a refinement indicator an average scaled error (*serr*) measurement is found for all of the partial differential equations using prescribed absolute and relative tolerances

$$serr = \sum_{i=1}^{npde} \frac{err_i}{atol_i \times area + rtol_i \times u_i}, \tag{45}$$

where *err* is the local error estimate, *npde* is the number of partial differential equations and *atol* and *rtol* are the absolute and relative tolerances, respectively.

This form of scaled error provides a flexible way of specifying the refinement—by adjusting *atol* and *rtol* for each partial differential equation the error measurement may be weighted toward any one equation.

The refinement indicator obtained from this scaled error

$$\text{ceiling}\big(\log_4(serr) + 1\big), \tag{46}$$

yields an integer value indicating the number of times the triangle should be refined (if positive) or de-refined (if negative).

Refinement is performed by regular subdivision of existing triangles. Each triangle identified for refinement is divided into four by joining together the centre points of each of its edges, see Fig. 3. In this way the smoothness of the original mesh is preserved as the aspect ratio of the new triangles will not be larger than that of the original. In the case when the adjacent triangle is not refined then a ‘hanging node’ would be present were it not for the use of temporary ‘green triangles’ which surround any area of refinement and are removed at the earliest opportunity.

In order to avoid refinement/de-refinement occurring too often, two controls are introduced into the adaptivity algorithm. The first is a restriction on de-refinement to one level at a time and this only after the triangle has been flagged to de-refine twice. The second control is a layer of extra refinement which is placed around each refined triangle. This layer is normally fixed at 2 triangle widths but may be varied for each problem. Further details of the algorithm can be found in Ware [20]. This method of subdivision of triangles allows a tree structure of coarse to fine triangles to be constructed and makes for simple de-refinement.

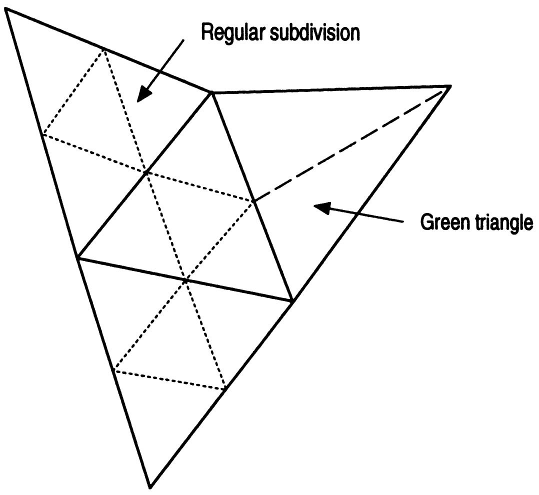

Fig. 3. Schematic of adaption technique.

4. BOUNDARY CONDITIONS

The idea of using a Riemann solver to calculate the flux at the face of a cell can be adopted in the description of boundary conditions in that a condition which relates, at most, one of the flow variables to the local flow field can be dealt with in this way—typical examples are that of unit inflow or a rating curve (a relationship between the discharge and water depth)—and when combined with equations obtained from characteristics theory give sufficient information for the boundary flux to be calculated.

Characteristics theory tells us that the Riemann invariants of the one-dimensional shallow water equations are

$$R^- = u + 2c,\ R^+ = u - 2c, \tag{47}$$

and define the relationships

$$\frac{d}{dt}(u - 2c) = 0 \text{ on } \frac{dx}{dt} = u - c,;$$
$$\frac{d}{dt}(u + 2c) = 0 \text{ on } \frac{dx}{dt} = u + c, \tag{48}$$

such that $u \pm 2c$ is constant along $u \pm c$, respectively; c is the local wave speed given by $c = \sqrt{\phi}$; R^+ represents the state to the left, R^- that to the right. At a boundary the right side is outside the domain, that is, the R^- relationship is replaced by the boundary condition itself. The R^+ relationship can be written as

$$u_L + 2c_L = u_* + 2c_*, \tag{49}$$

in which the subscripts L and $*$ indicate the left and solution variables, respectively. This can be combined with the boundary condition to obtain a solution for u_* and c_*. In general, the normal flux may then be calculated at the boundary face, namely

$$Fn(U_*) = \begin{bmatrix} \phi_* u_n \\ \phi_* u_n u_* + \frac{1}{2}\phi_*{}^2 n_x \\ \phi_* u_n v_* + \frac{1}{2}\phi_*{}^2 n_y \end{bmatrix}, \tag{50}$$

where $u_n = u_* n_x + v_* n_y$.

4.1. Inflow, outflow and solid boundaries

Upstream inflow boundary conditions are implemented in the form of a unit discharge or discharge per unit length. At any given time the following discharge condition holds

$$q = h_* u_*, \tag{51}$$

which when combined with the relationship $c = \sqrt{hg}$ and Equation (50) gives

$$2c_*^3 - (u_L + 2c_L)c_*^2 + qg = 0, \tag{52}$$

which can be solved iteratively for c_*, and the normal boundary flux calculated via Equation (51).

In the case of a specified depth boundary condition, $c_* = \sqrt{gh_*}$, where h_* is known. From U_L, the normal and tangential velocity components at the face of a control volume are

$$u_{L_n} = u_L n_x + v_L n_y,\ u_L t = -u_L n_y + v_L n_x. \tag{53}$$

From which u_n follows directly from Equation (50),

$$u_n = u_L n + 2c_L - 2c_*. \tag{54}$$

The transverse velocity is passively advected [11], so $u_t = u_{L_t}$. Transforming back to the

Cartesian system using

$$u_* = u_n n_x - u_t n_y,\ v_* = u_n n_y + u_t n_x, \tag{55}$$

enables the normal flux to be calculated.

A useful downstream boundary condition is a free out-fall—an outlet boundary which imposes no influence on the fluid in the domain—where depths are maintained and waves travel through without reflection. This type of boundary is specified by fixing the boundary flux using the internal flow conditions—simply $U_* = U_L$ with the normal flux calculated from Equation (50).

At stationary solid boundary walls the no slip condition is assumed to hold, that is normal and tangential velocity components are zero there. Consequently, from Equation (49) $c_* = c_L$ giving the normal flux at a solid wall as simply

$$F_n(U_*) = \begin{bmatrix} 0 \\ \frac{1}{2}\phi_*^2 n_x \\ \frac{1}{2}\phi_*^2 n_y \end{bmatrix}. \tag{56}$$

4.2. *Wetting and drying*

In practical two-dimensional river and coastal flows, boundaries exist at which the water depth approaches zero i.e. wetting/drying occurs. Clearly, an inadequate treatment of this boundary condition will affect the accuracy of the solution.

The exact solution to a dry bed problem can be obtained by examining the Riemann invariants present, in much the same way as the the approach described above, for the initial conditions: ϕ_L, $u_L = 0$ on the left and no fluid ($\phi_R = u_R = 0$) to the right. The exact solution to this sub-critical Riemann problem [11] is simply

$$\phi = \left(\frac{1}{3}\left(u_L + 2\sqrt{\phi_L}\right)\right)^2,\ u = \frac{1}{3}\left(u_L + 2\sqrt{\phi_L}\right). \tag{57}$$

Unfortunately this exact, and other approximate, Riemann solutions produce unrealistically high velocities when the depth of flow is very small. In the past, two not dissimilar approaches have been used in an attempt to alleviate this problem. The first examines the solution depth in a cell (or at a node in finite difference schemes) and removes from the computation any cell whose depth is less that some prescribed value [2]. The second approach is to reformulate the problem when the depths are *small* only removing them from the calculation when they are *very small* [1,28]. The reformulation is usually one of removal of the inertial terms or to make the flow friction dominated. The second approach, being more sophisticated, would be expected to produce more accurate solutions. The increase in accuracy is, however, difficult to quantify and no comparative work has yet been published.

In this work a scheme based on the second approach is used; it is similar to that of Zhao *et al.* [1], and involves monitoring solution variables at both cell centres and faces. The implementation differs however, by not excluding cells from the solution update but rather using a modified flux to achieve the same result.

Following Zhao *et al.* the (limited) flow variables at each cell face are monitored and that face is classed as being either a *dry land boundary* or a *wet boundary*. Similarly, flow variables in the cell are monitored with each cell denoted as being *wet*, *dry* or *partially dry*. Depending on the combination of *wet/dry/partial* cells and cell-faces, a choice of flux calculation for each cell face is made and as limiter functions are used to calculate the cell face flux, limited face values are used to determine the face state. The method proceeds as follows:

- A tolerance depth, *htol*1, is chosen and the cell-face nominated a *land boundary* if the left depth $h_L < htol1$ and the right depth $h_R < htol1$ or both the left and right cells are dry.

- A cell is deemed *dry* if the depth is less than *htol*1 and all cell-faces are *land boundaries*.
- A cell is deemed *partially dry* if the depth of fluid is greater that *htol*1 but less than a higher valued tolerance *htol*2, or when the depth is less than *htol*1 but one of the cell-faces is not a *land boundary*. For a *partially dry* cell the momentum fluxes are set to zero and only the mass flux considered.
- If the depth in a cell is greater than *htol*2 then the cell is *wet* and the complete set of fluxes are calculated via the Riemann solver.

The above wetting/drying procedure was tested thoroughly via controlled numerical experiments, employing a one-dimensional version of the algorithm, before applying it to two-dimensional flow problems, see below. The first problem considered is that of a dam-break onto a dry bed having an exact solution, the second flow on a sloping beach driven by an oscillating sea-ward depth variation.

4.2.1. Dam-break induced flow onto a dry bed. The initial conditions for the dry bed dam-break problem are $\phi_L = 1$, $U_L = 0$ and $\phi_R = U_R = 0$, the exact solution to which can be found elsewhere, Toro [11]. The values of *htol*1 and *htol*2 are 0.0005 and 0.005 respectively, the cell width is 1/160 and the time-step governed via Equation (31), described earlier.

Figure 4 shows analytical and numerical solutions of water surface and velocity profiles after 0.04 and 0.1 sec have elapsed which can be seen to agree very well. The major discrepancy occurs, as expected, at the moving foot of the wave where the dry-bed condition is applied. However, the effect is local, differences are small height wise and the wave appears to be travelling at the correct speed.

4.2.2. Sea-ward driven flow onto a dry sloping beach. Although no exact solution exists for a dry bed problem with non-zero bed slope, recent numerical results [29] provide excellent material for comparison purposes. The problem, originally specified by Watson and Peregrine [30], was investigated by Pennington and Berzins [29] via a simple coordinate transformation from

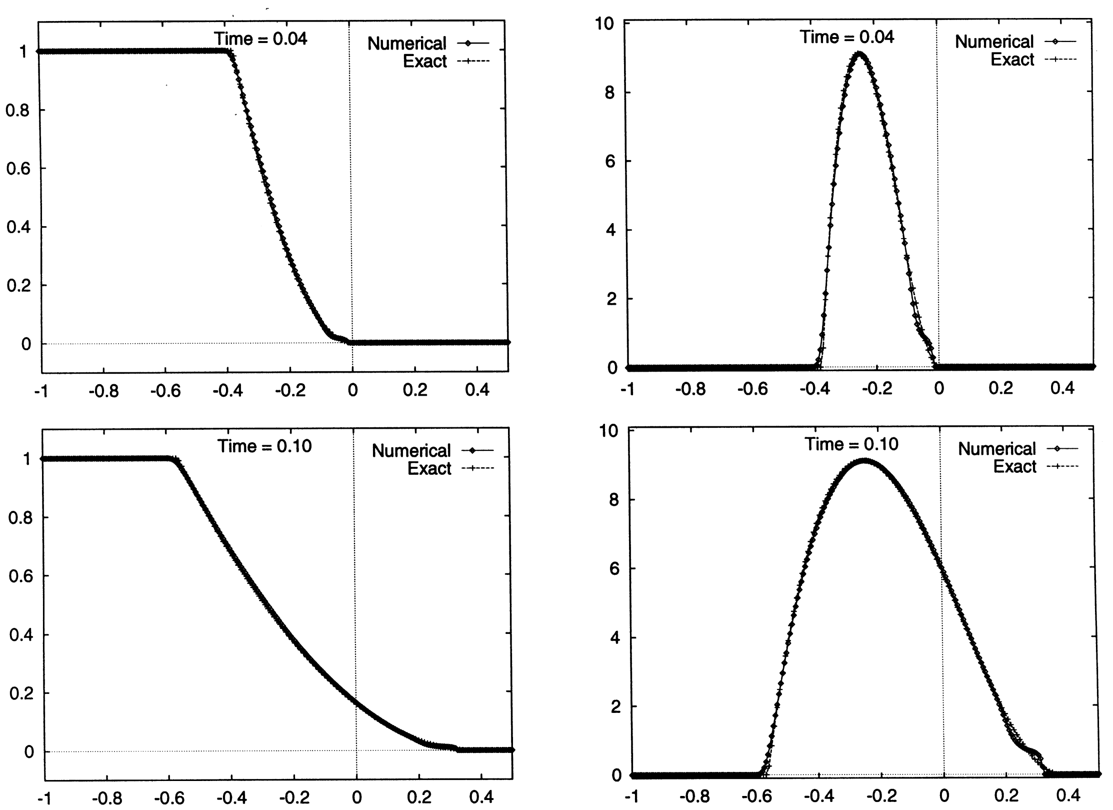

Fig. 4. The dry-bed dam-break problem: height (left) and discharge (right). Comparison of numerical and exact solution at two instants of time.

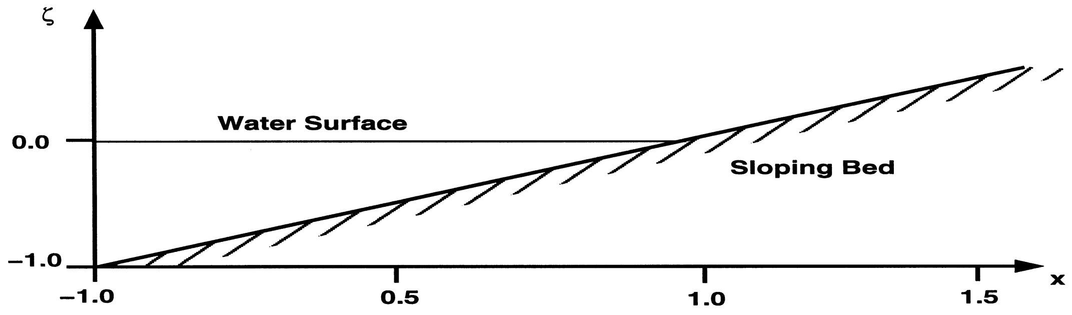

Fig. 5. Sloping dry-bed test problem: initial conditions.

which they derived a homogeneous set of equations that could be solved numerically with relative ease. Taking $\zeta = (x + 1)/(x_s + 1)$, where $x_s(t)$ is the position of the moving shoreline, the one-dimensional shallow water equations can be written as

$$(1 + x_s)\frac{\partial h}{\partial t} - \zeta\frac{dx_s}{dt}\frac{\partial h}{\partial \zeta} + \frac{\partial(uh)}{\partial \zeta} = 0, \tag{58}$$

$$(1 + x_s)\frac{\partial u}{\partial t} - \zeta\frac{dx_s}{dt}\frac{\partial u}{\partial \zeta} + \frac{\partial}{\partial \zeta}\left(\frac{1}{2}u^2 + h\right) = -(1 + x_s), \tag{59}$$

with boundary conditions $h(1,t) = 0$, $u(1,t) = dx_s/dt$.

The transformation has the effect of confining the numerical solution to a fixed domain, $\zeta \in [-1,1]$, with a constant boundary condition at the dry bed boundary, thus the problem of a moving dry-bed is removed. The initial conditions are those of a flat water surface with a non-dimensional depth of 1 at the upstream end decreasing as x increases to the value zero where it meets the bed of slope −1 at the point $x = 0$, see Fig. 5. The flow is driven by imposing a sinusoidal change in the depth at $x = -1$ given by $f(t) = 0.6((2\pi t/0.475) - sin(2\pi t/0.525))$—see Fig. 6. This function oscillates greatly causing sharp fronts to appear in the flow which travel toward the sloping bed.

Results of the water surface heights at three successive times are shown in Fig. 7. The sharp wave fronts which develop are seen to agree well with the corresponding solution of Pennington and Berzins [29]. Most interestingly, although very slightly retarded, the height and position of the front travelling on the sloping bed is in close agreement.

5. TWO-DIMENSIONAL FLOW PROBLEMS AND RESULTS

In order to demonstrate the flexibility of the algorithm outlined above three two-dimensional flow problems are considered, that of: a dam-break in a rectangular domain; a dam-break in a

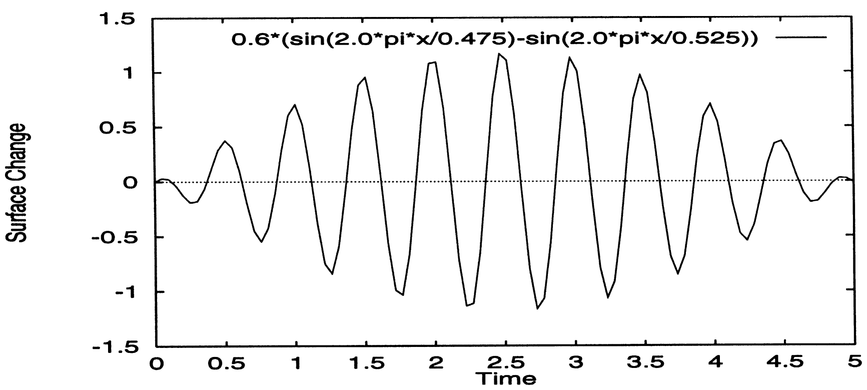

Fig. 6. Upstream surface oscillation used to drive the sloping bed problem illustrated in Fig. 5.

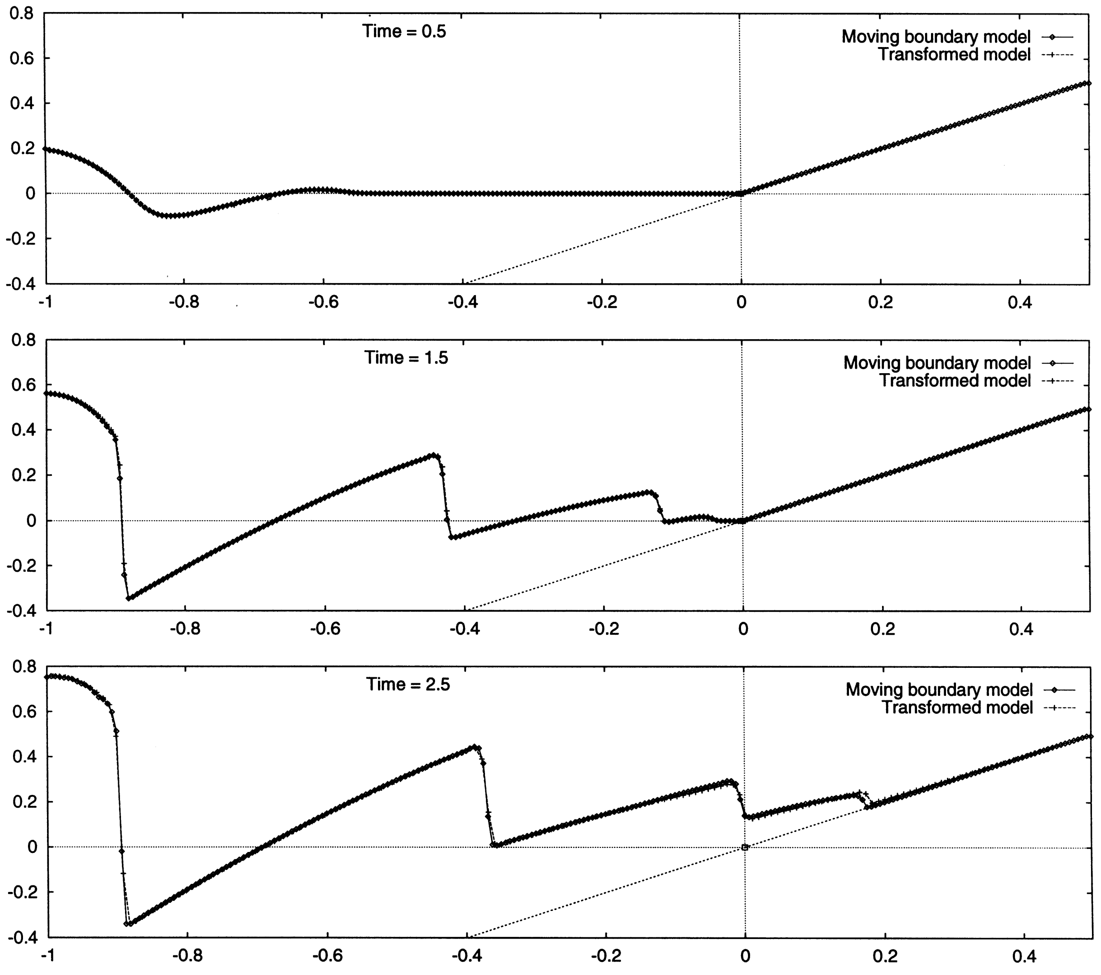

Fig. 7. Comparison of depths of flow, ζ onto a sloping dry-bed, at three different times.

slowly diverging/converging channel together with comparisons with experimental measurements; flow in a natural estuary driven by both tidal changes and river inflow.

5.1. Standard dam-break problem

A commonly cited problem in hydraulic flow simulations is that of a two-dimensional dam-break, see for example Zhao *et al.* [1], Alcrudo *et al.* [9,26], Glaister [31] and Katapodes [32]. This type of flow will reveal any weakness in relation as to how well a particular strategy deals with combined sub- and super-critical flow. The flow domain, detailed in Fig. 8, is 200 m square and contains a wall with an offset gap of 75 m across its centre. The initial conditions are zero flow with higher depth to the left than to the right of the wall, and is equivalent to a partial dam failure. The resultant flow is two-dimensional, highly non-linear and possibly trans-critical. All boundaries are treated as solid non-slip walls.

Two scenarios are considered: the first with initial conditions of zero flow and depths of 1.0 m to the left and 0.2 m to the right of the dividing wall; the second with the same condition to the left but with a dry bed to the right.

Solutions to the first scenario are shown in Figs 9 and 11 at four instants in time. In Fig. 9 the left and right images represent the mesh configuration and velocity vectors, respectively. Figure 11 shows a 3-dimensional representation of the water surface at the same four instants in time as those in Fig. 9. It is useful to examine all three mesh, vector and surface plots together.

The collapsing vertical wall of water moves in the form of two wave fronts to the left and right. To the right the wave moving into the shallow region has a flat profile (in plan) at 4 sec.

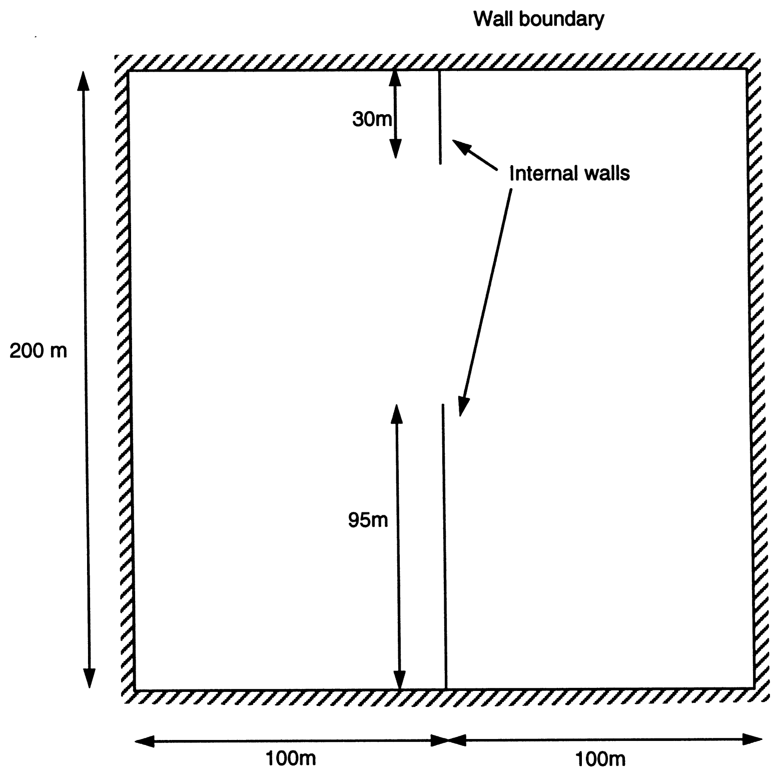

Fig. 8. Flow domain of standard two-dimensional dam-break test problem.

It is at this initial stage where traditional methods have failed to capture the conditions well and it should be noted how the current method has ensured that no under or over shoots are present and that the gradient across the steep fronts and the horizontal region is sharp. The right front curvature as well as the height and surface gradient increase with time. On the vector plots two vortices are seen to form on either side of the breach. These appear on the surface plots as indents which deepen in time, contributing to the highly undulating surface shape at 28 sec. The mesh can be seen adapting itself to satisfy the error control requirements as the solution evolves in time. The adaption tolerances were set as in Table 1, the significance of these tolerances being that only the absolute errors are taken into account for adaption, and weight is given toward the momentum equations. The solution which began on a coarse grid (level 0) of 400 triangles, evolves in time with refinement up to level 2 (the equivalent of a full mesh of 6400 triangles). The gradation of the mesh from coarse to fine is shown particularly clearly at 12 sec, see Fig. 9. The maximum mesh density was restricted to two levels of refinement to ensure clarity of presented meshes. More refined regions would have been created had this restriction not been implemented.

Solutions to the second scenario are shown in Figs 10 and 11. The same adaption tolerances were used with wetting and drying tolerances of $htol1 = 0.00001$, $htol2 = 0.00002$, respectively. Again, the similar phenomena of the mesh adapting over the steep surface gradient as the water flows to the right is seen. Two important differences are apparent: the shape of the front remains flatter and the flow more unidirectional than the non-dry bed case; no steep surface gra-

Table 1. Adaption tolerances for two-dimensional dam-break

Equation	Atol	Rtol
continuity	0.1	0.0
x-momentum	0.0001	0.0
y-momentum	0.0001	0.0

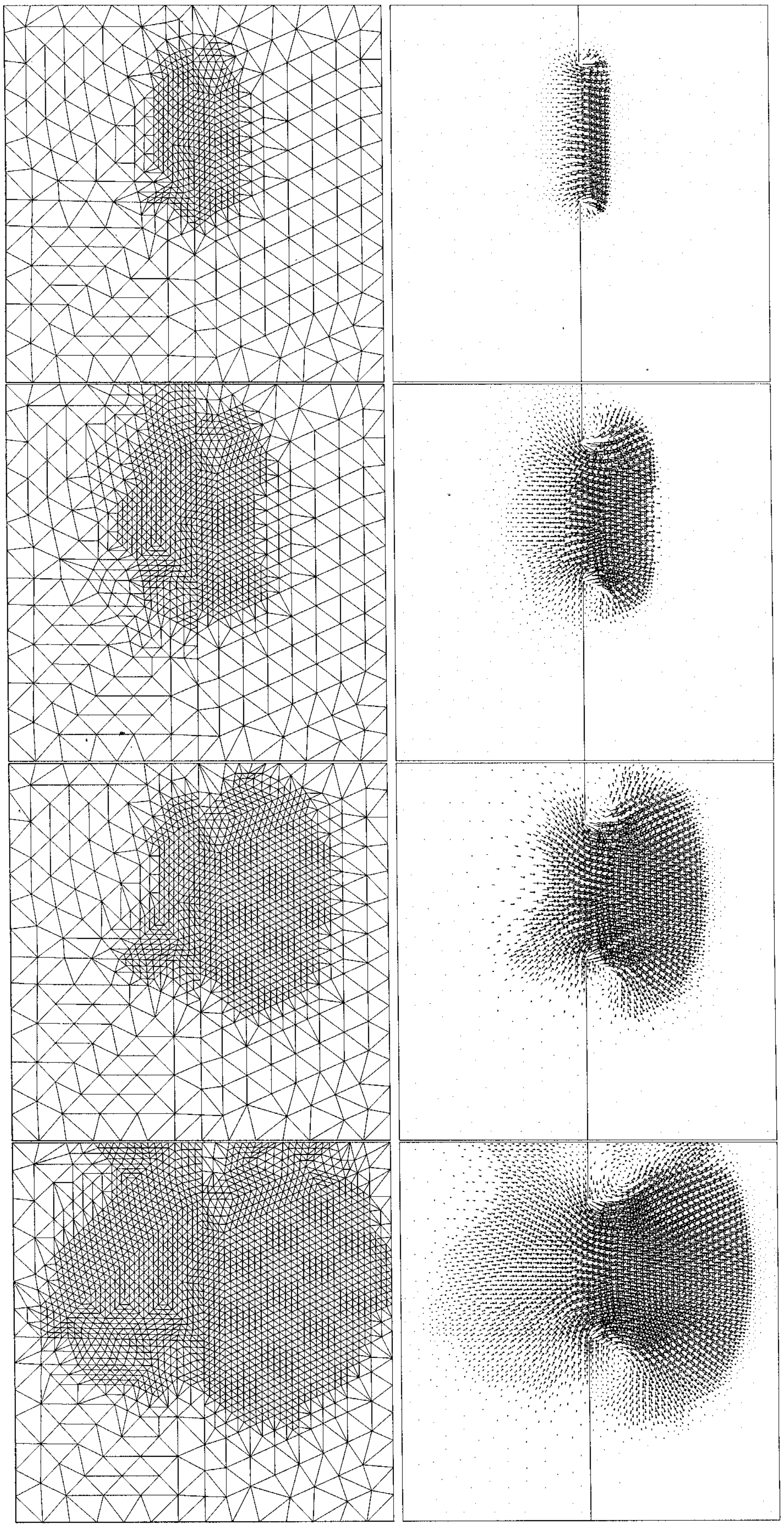

Fig. 9. Two-dimensional dam-break problem—left 1.0 m, right 0.2 m; mesh (left) and velocity vectors (right) at four instants in time—from the top t = 4, 12, 20 and 28 sec.

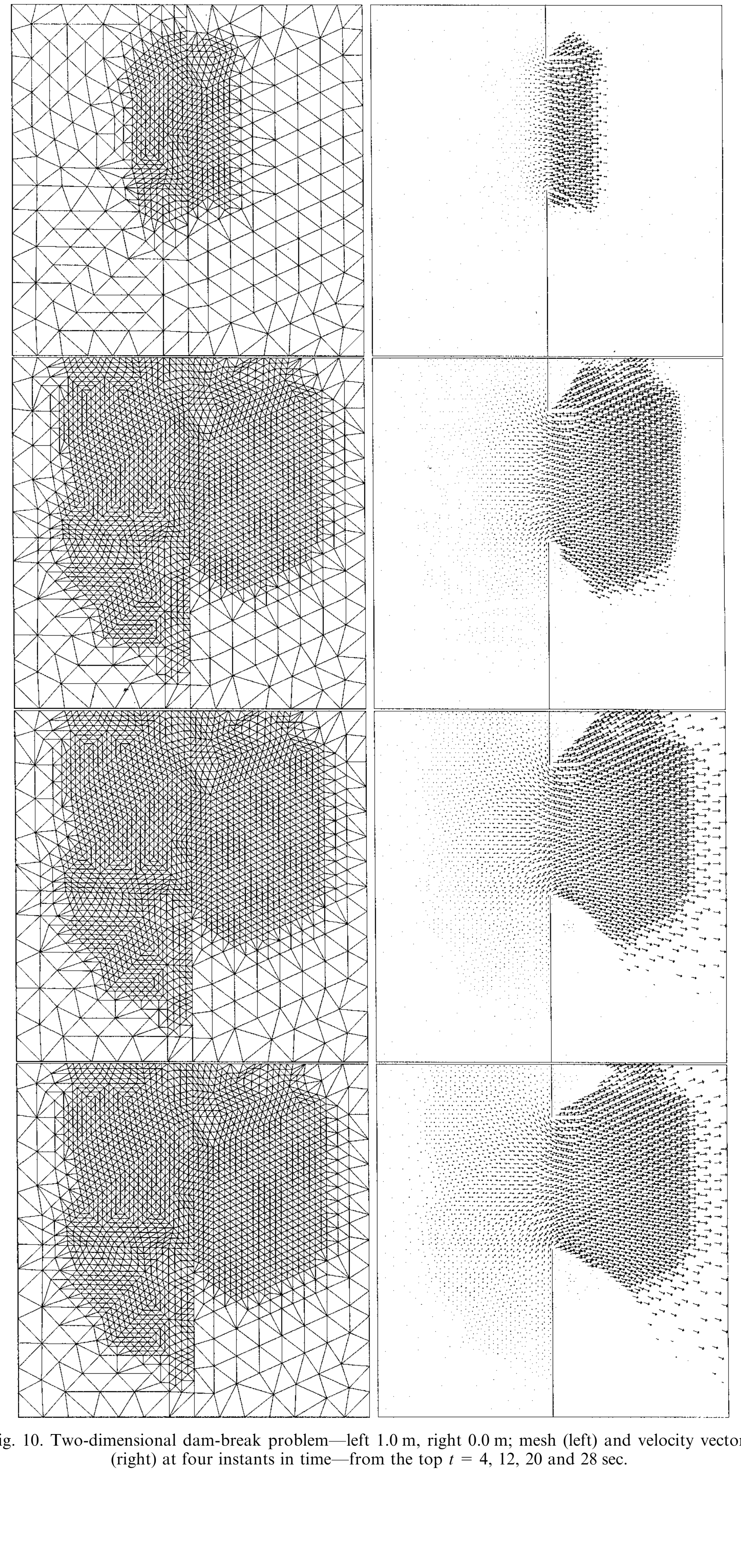

Fig. 10. Two-dimensional dam-break problem—left 1.0 m, right 0.0 m; mesh (left) and velocity vectors (right) at four instants in time—from the top t = 4, 12, 20 and 28 sec.

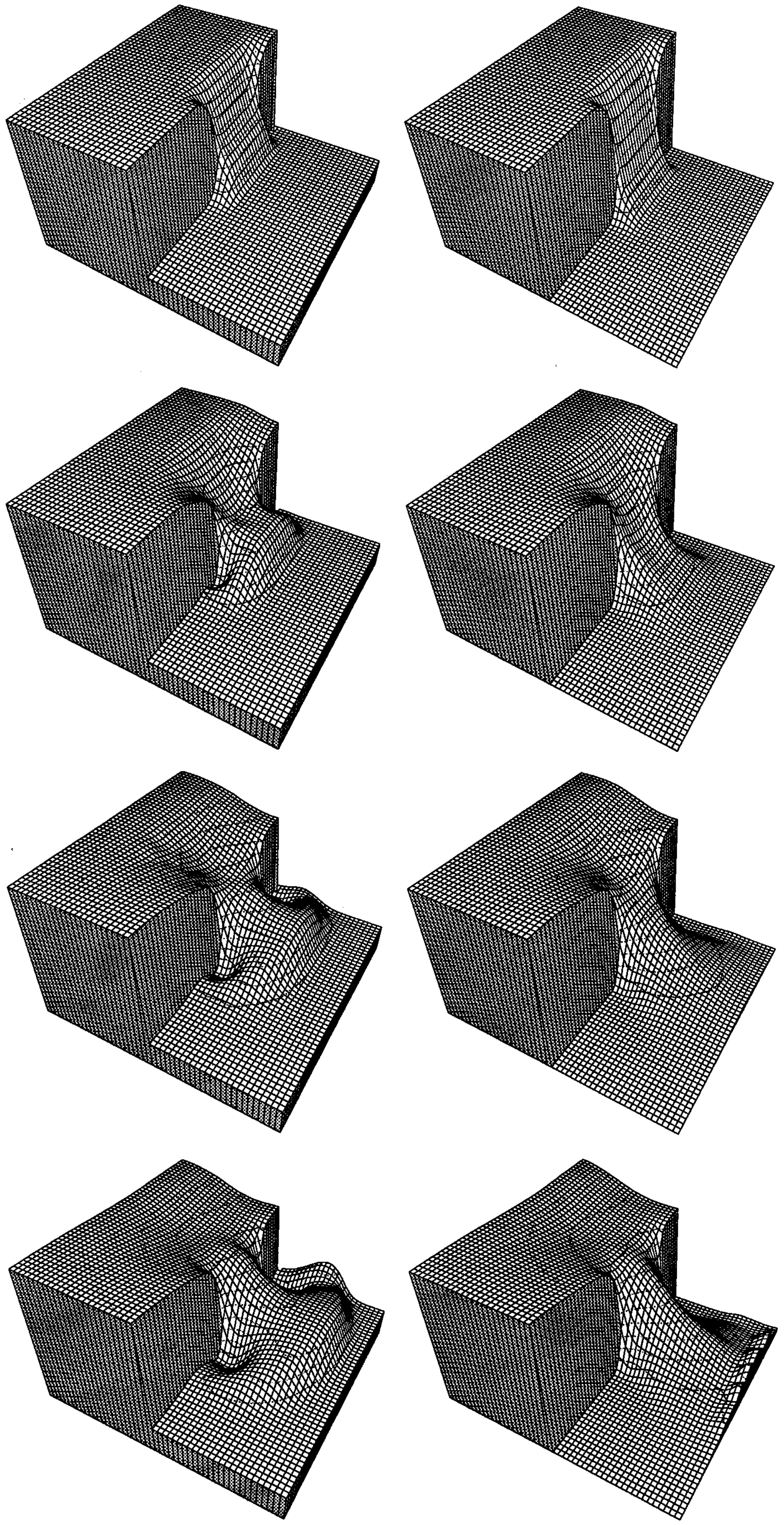

Fig. 11. Two-dimensional dam-break problem—wet (left) and dry (right) at four instants in time—from the top t = 4, 12, 20 and 28 sec.

dient occurs at the tip of the front moving onto the dry-region. The front also moves faster having reached the right hand side of the domain at 28 sec. The absence of a steep front and faster movement for this dry case are both consistent with the solution of a one-dimensional dam-break problem [11]. Adaption has extended further into the upstream region. This, although difficult to see on the surface plots, is because the surface has been drawn down further by the increased speed of the out-flowing water.

Although direct comparison with other published solutions is not possible, the general shape of the surfaces appear similar. The two flows demonstrate how the method handles steep gradients and trans-criticality well without special treatment of troublesome terms.

5.2. The Bellos dam-break problem

Bellos *et al.* [33] detail several simulations of an instantly failing dam and the subsequent flow on to a dry downstream bed. These tests were performed in a channel 20 m long by 1.4 m wide at the upstream and downstream ends with a smooth non-symmetric contraction to 0.6 m wide at a position 8.5 m along the length. The channel geometry is shown in Fig. 12.

The experimental series consisted of setting different upstream depths and slopes of channel. A plate simulating a dam separates the upstream and downstream regions, this was removed quickly to approximate an instantly failing dam. The experimental measurements consist of depth histories along the centreline of the channel for each test. The depths in the upstream (wet) region were recorded using wire resistance type depth gauges while those downstream of the dam, which were dry initially, were recorded using pressure transducers. Data at five positions were collected in the mid-width position at the following distances from the upstream end: 0.0 m, 4.5 m, 8.5 m, 13.5 and at 18.5 m (see Fig. 12). A more detailed description of the experimental arrangements and series of tests can be found in Bellos *et al.* [33].

In the numerical simulation the walls of the channel and the upstream end are solid, no slip, boundaries. At the downstream end for a dry bed a free out-fall is applied—see Section 4.1. For situations with a set depth downstream it is not clear from the experimental description what condition actually prevailed. For this reason the depth history measurements at 18.5 m have been used to provide specified depth boundary condition for the numerical simulations.

The first problem considered is that for an upstream depth of 0.3 m and a downstream depth of 0.053 m with a horizontal bed and having a Manning friction coefficient of 0.012. Computations were carried out on a fixed mesh of 4025 triangles. The water surface profiles for each of the first 6 sec of simulation are shown in Fig. 13. The steep front is seen to be preserved from the outset and propagates to the end of the channel without dissipation. The effect of the two dimensional nature of the channel on these profiles can be seen—there is a reduction in depth behind the right moving front before an increase at the upstream end.

Figure 15 shows depth histories at five points in the channel. Included for comparison are the measurements from the experimental study of Bellos *et al.* and the results of Garcia-Navarro *et al.* [8]. It can be seen how, at each point, the depth is resolved well with the steep fronted wave being captured and its shape maintained. Both sets of numerical results fail to position the wave

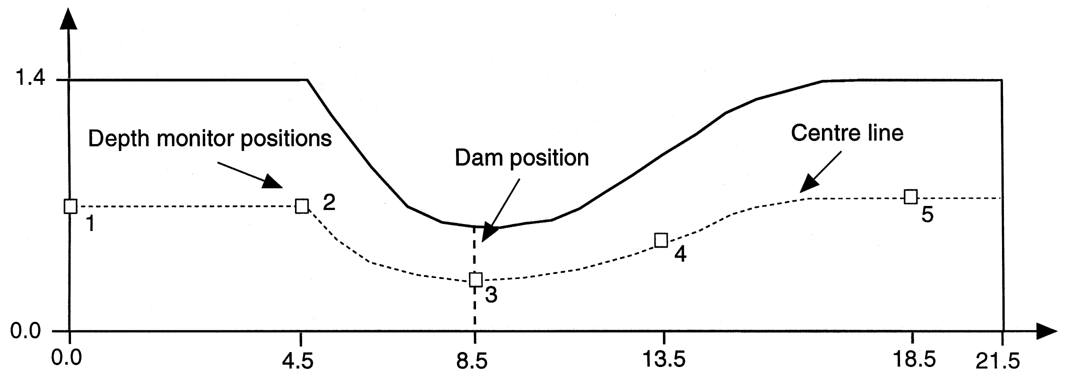

Fig. 12. Solution domain for the Bellos Channel problem, showing depth gauge positions (all dimensions in metres).

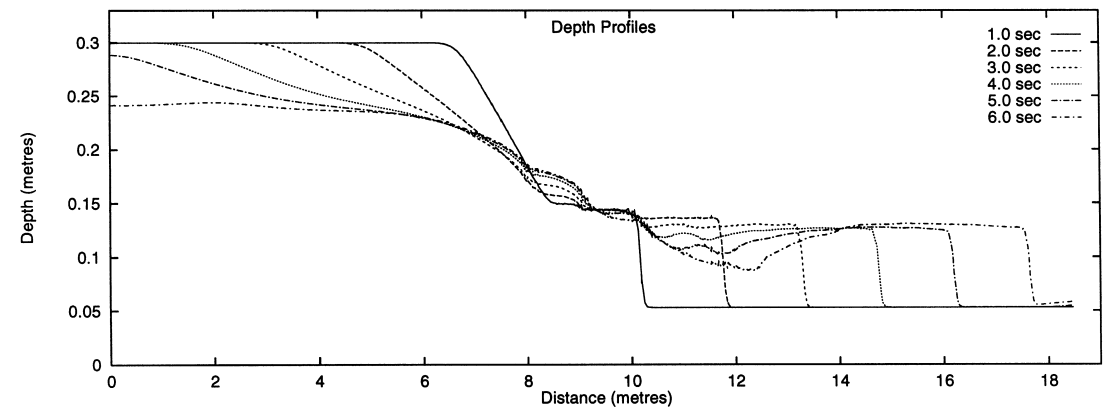

Fig. 13. Water surface profiles for Bellos Channel at six instants in time—Expt. 1.

travelling from the downstream end well. This is probably due to the imposition of inappropriate boundary conditions which, as mentioned above, are not specified in the description of the experimental procedure.

The second problem has the same upstream depth and Manning coefficient but for flow onto a horizontal *dry* bed. The results were obtained from a 3095 triangular mesh with tolerances for wetting/drying being *htol*1 = 0.001, *htol*2 = 0.011. Water surface profiles are shown in Fig. 14 where it can be seen that the wave front moves over the dry bed until it eventually leaves the domain at the downstream end. In Fig. 16 the depth histories from the experimental study and this work are shown. The first four plots agree very well with the experimental data; only the last, at 18.5 m, differs in that the depth never reaches the measured levels. This again is probably due to uncertainty and lack of agreement in the boundary conditions applied at the downstream boundary in the case of the numerical solution.

5.3. Flow in the river axe estuary

A practical problem which demonstrates the full capabilities of the approach described here is that of flow in the River Axe estuary. The associated field data, kindly provided by Wessex Water, is of a high quality enabling the physical problem to be described in terms of: upstream discharge from the River Axe; downstream sea-ward tide-cycle measurements; cross-sectional measurements detailing the topographical (bed-elevation) information required to compute the solution domain, Fig. 17. The river position and path was constructed from OS maps [34].

The unstructured gridding routine is able to take full advantage of the field data by describing the highly complex geometry to greatest effect in that the mesh is concentrated in user specified

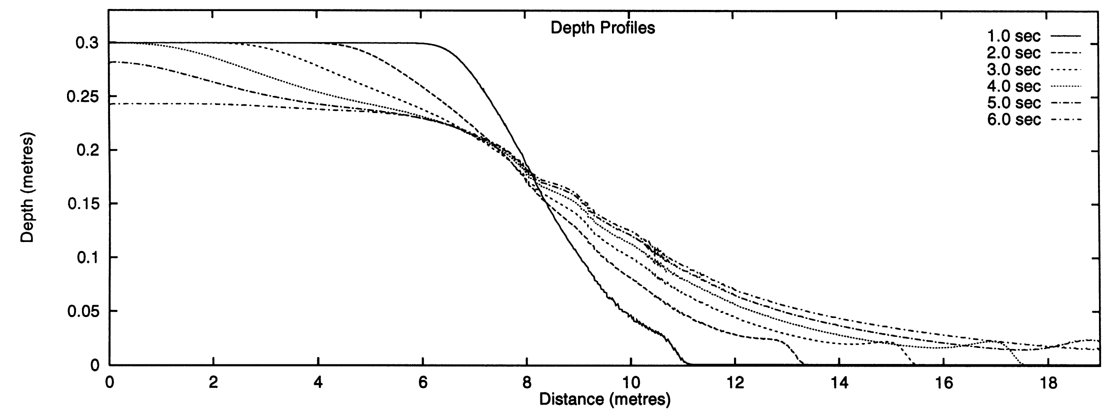

Fig. 14. Water surface profiles for Bellos Channel at six instants in time—Expt. 2.

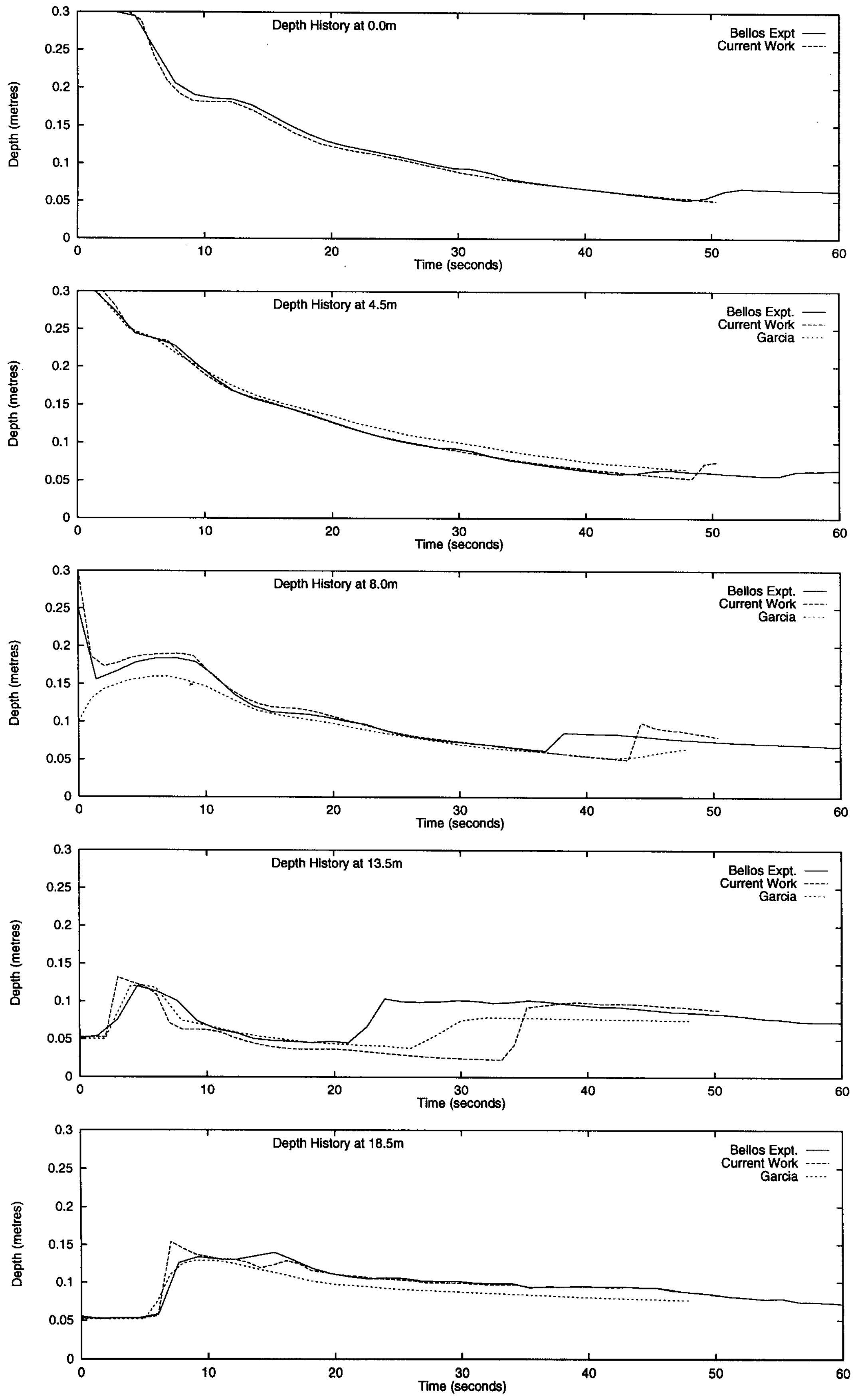

Fig. 15. Comparison of depth histories at five points in the Bellos Channel—Expt. 1.

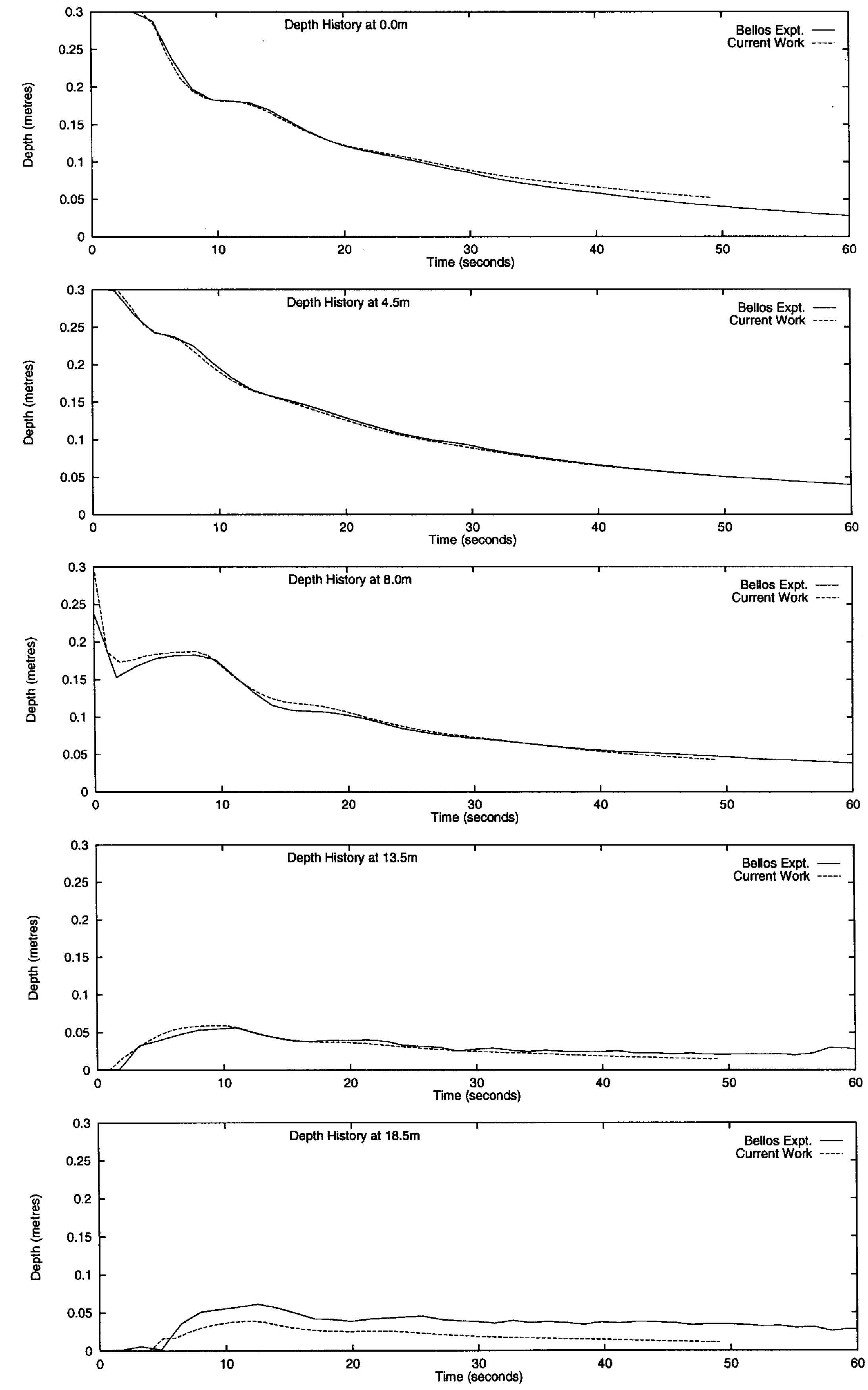

Fig. 16. Comparison of depth histories at five points in the Bellos Channel—Expt. 2.

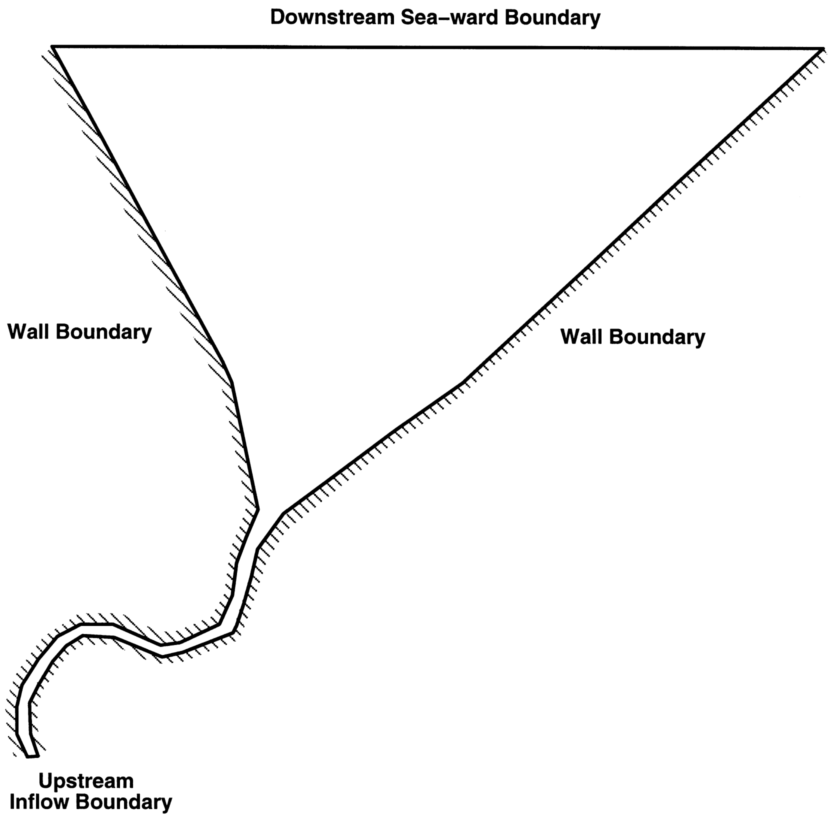

Fig. 17. River Axe: domain and boundary conditions.

regions of complex topology via the mesh weighting function described in Section 2.1. The result is that steep bed gradients are resolved adequately with a moderate number of triangles.

The bed surface, which is highly non-uniform, consists of two very flat regions which slope up toward each side together with a narrow steep-sided main channel. The available field measurement data for the bed consists of four cross-section profiles from which the data are interpolated on to the computational mesh. Figure 19(a) is a three-dimensional view of this bed surface containing 1200 evenly distributed triangles, highlighting the main topographic features, but reveal-

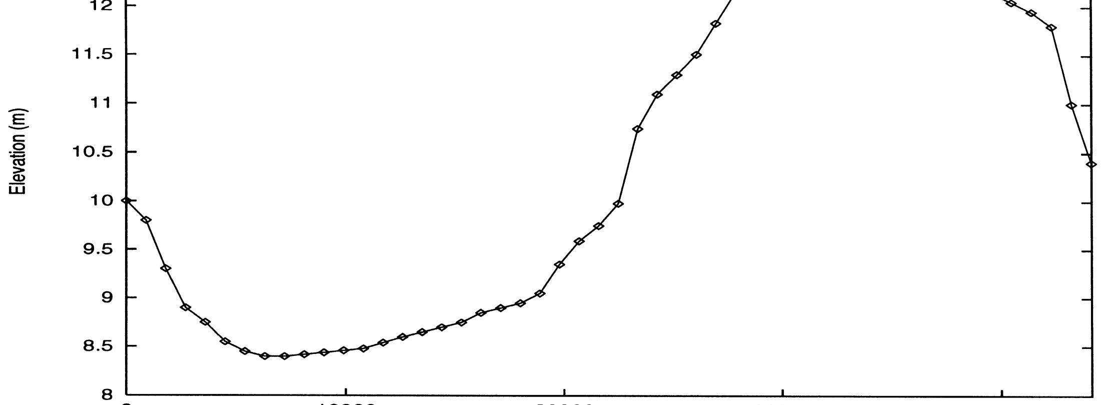

Fig. 18. Surface oscillation specifying the downstream boundary condition for the River Axe.

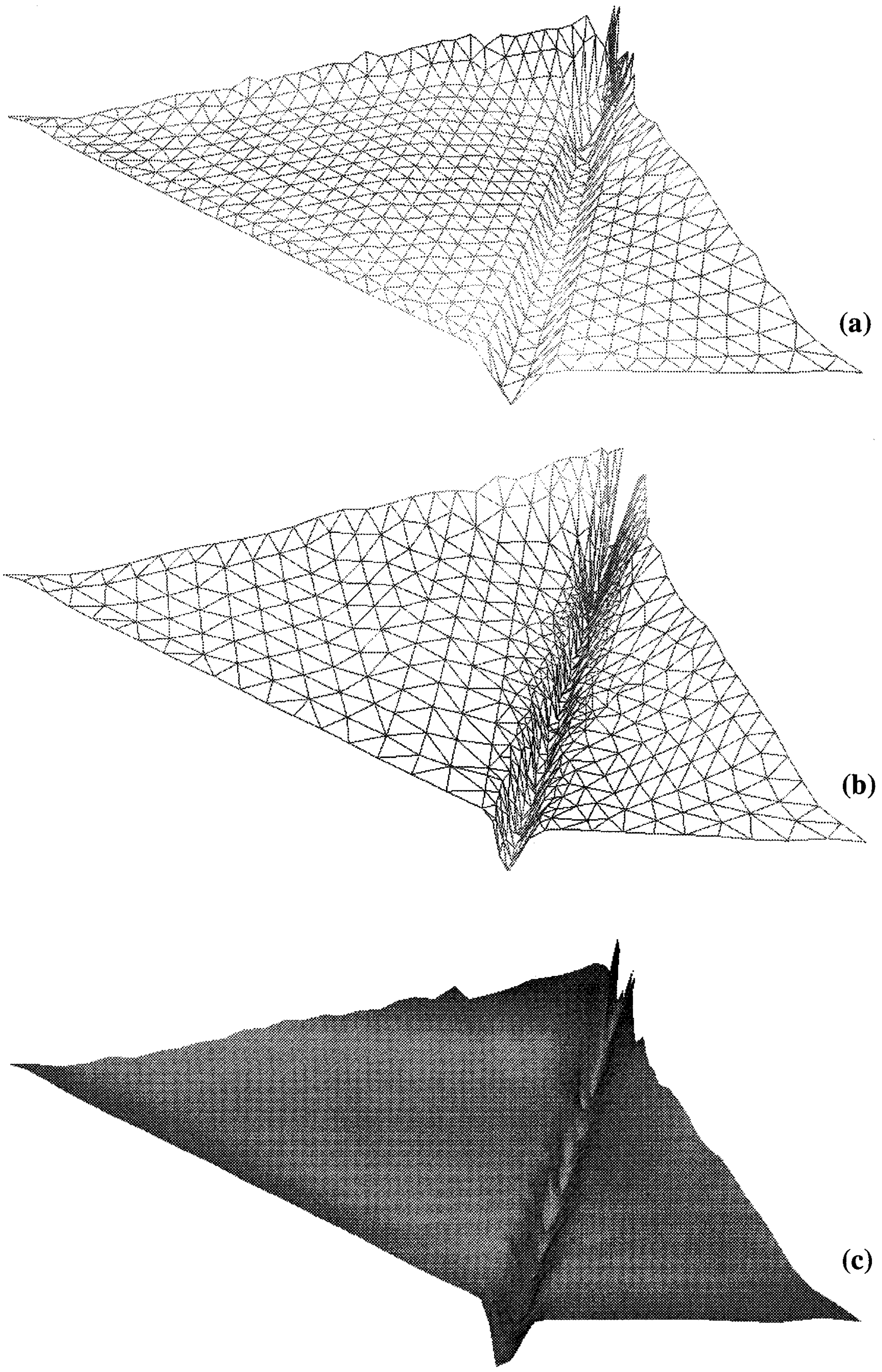

Fig. 19. Three representations of the bed topography: 1200 uniform triangulation (a), 1067 weighted triangulation (b), 4262 weighed triangulation (c), solid view of the bed surface from (a).

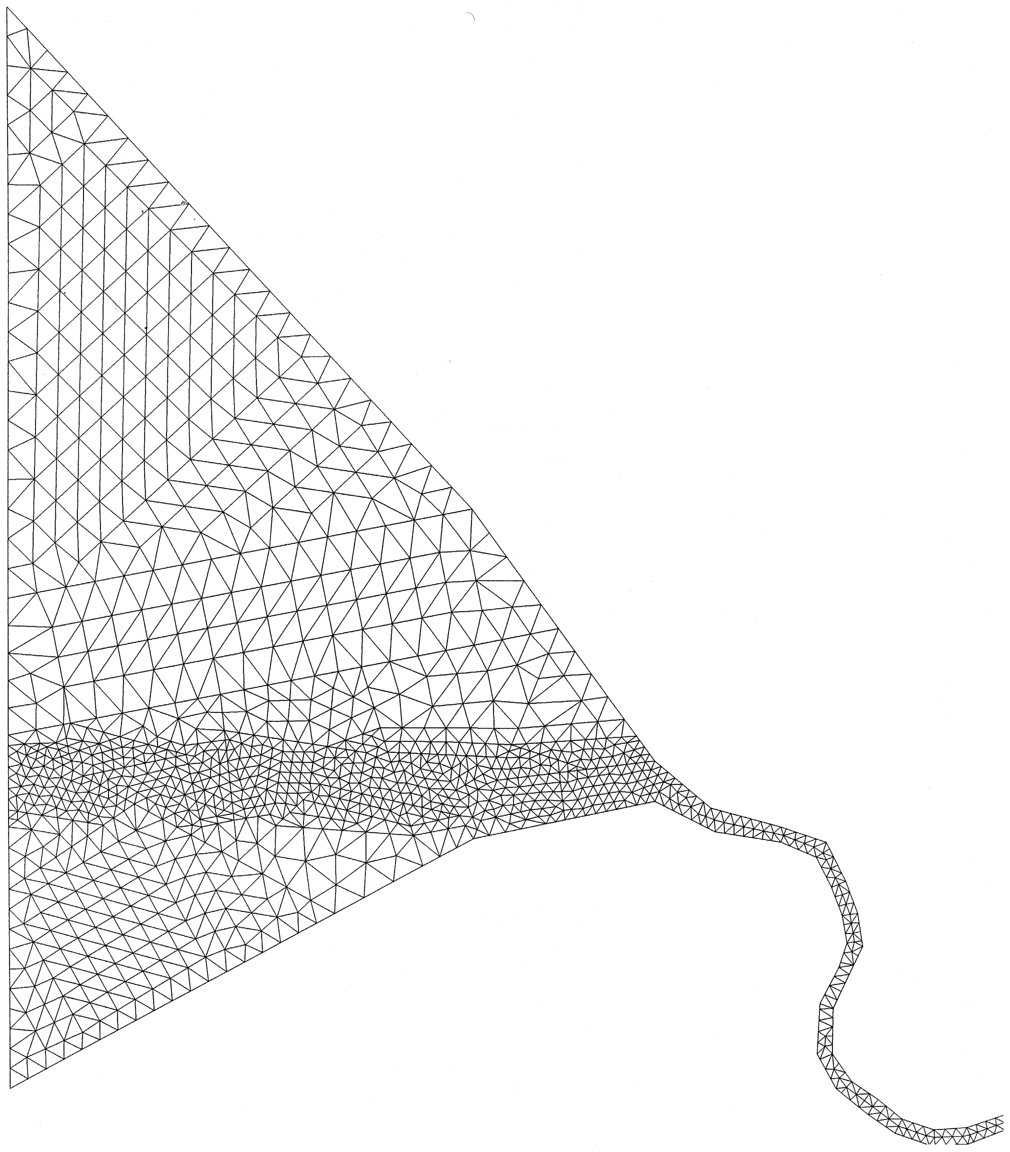

Fig. 20. Plan view of the River Axe Estuary mesh with 2126 triangles.

ing that with this number and arrangement of triangles the main narrow channel is not resolved particularly well. This can be overcome without the need to employ an excessive number of triangles over the two comparatively broad bounding flanks, by biasing the triangular decomposition toward the channel region as shown in Fig. 19(b). Comparing Fig. 19(a) and (b), reveals that the latter grid distribution is necessary to best resolve the steepness of the 'V' shaped channel side walls. For the sake of clarity the bottom image, Fig. 19(c), shows a 'solid' view of the bed surface produced from one uniform refinement of the middle image—a plan view of the corresponding mesh is shown in Fig. 20.

Examination of Fig. 17 shows that three types of boundary condition need to be specified: to the left and right are solid walls; at the upstream end, inflow is specified using a constant discharge condition; at the downstream or sea-ward boundary, a specified depth is imposed. The latter, depth data, is obtained from recorded tide height history data—a typical tidal cycle is shown in Fig. 18. The initial conditions are assumed to be zero flow over the entire solution domain with a water surface elevation of 10.0 m. For this elevation and prescribed tidal boundary condition no wetting/drying occurs so that the *htol* values do not require setting.

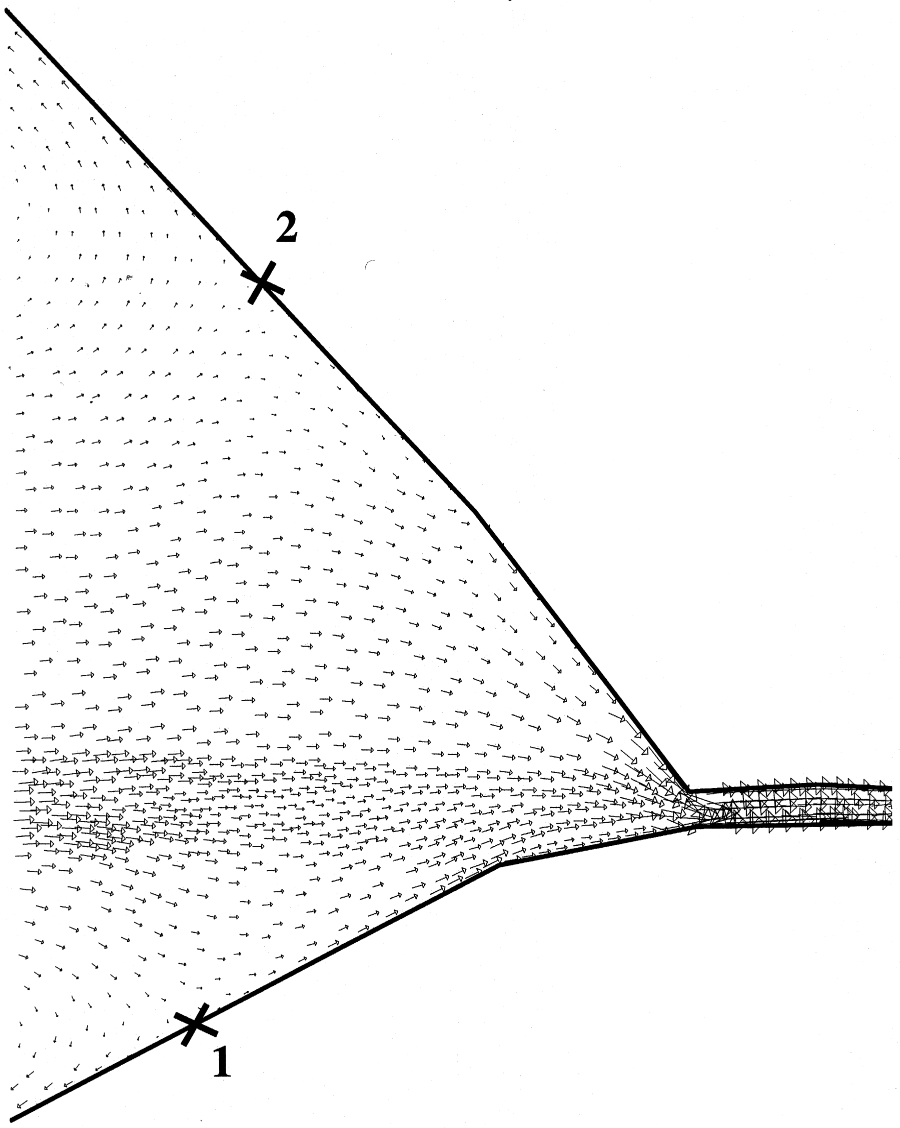

Fig. 21. Typical velocity vectors and position of stagnation points (1 and 2) a flood tide.

The velocity vectors for a typical flow due to a flood tide are shown in Fig. 21. The flow is funnelled predominantly into the river channel where its velocity increases rapidly. Note how, at the river mouth, the flow is non-symmetric which could indicate a possible navigation problem if this were a shipping channel. At the edges of the estuary close to the sea-ward boundary the flow is reversed resulting in the two stagnation points shown and flow being forced out of the domain by the rise in sea level at these points. Figure 22 shows typical velocity vectors for an ebb tide—flow out to sea—in which the rapid velocities in the river reduce, as expected, as the estuary widens and flows into the sea. There are no stagnation points apparent in this flow.

In Fig. 23 velocity vectors are shown for the region close to the mouth of the river at a point in time when flow is being reversed by a change in tide. Note the well-defined diagonal line along which flow is directed when the contra-directional velocities meet which is terminated at each end by a stagnation point reflecting an increase in sea-level. This is an interesting example of a truly two-dimensional flow feature occurring during a tide cycle in this estuary.

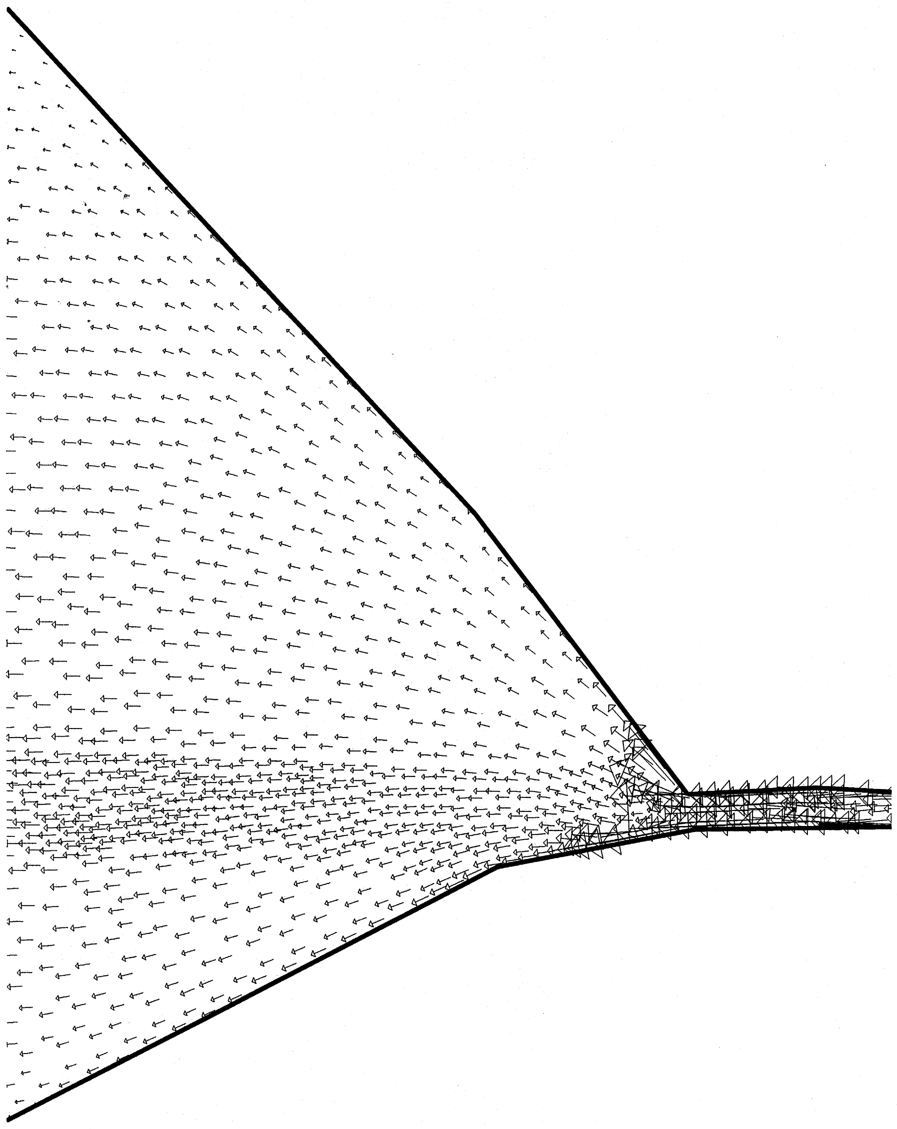
Fig. 22. Typical velocity vectors for an ebb tide.

6. CONCLUSIONS

An efficient finite volume based numerical algorithm for the solution of the shallow water equations is presented which embodies adaptive meshing, refinement/de-refinement, and variable time-stepping for application to geometrically challenging practical river and estuary flow problems. Accurate results are obtained for trans-critical flow situations via shock capturing techniques—in the form of an approximate Riemann solver—coupled with a weighted triangular tessellation of the solution domains of interest. The latter is achieved using a readily available mesh generator which is then further processed to create the appropriate data structures.

Key features of the methodology are flexibility and accuracy coupled with the option of automatic error control. Of practical interest is the ability to prescribe easily a wide range of boundary conditions including the wetting/drying of flood planes.

A convincing demonstration of the ability to deal with a convoluted, practical flow problem is given, in the form of the River Axe estuary, where it is shown that the flow is truly two-dimensional, casting doubt on the validity of predictions that might be obtained via simplified one-dimensional models of such flows.

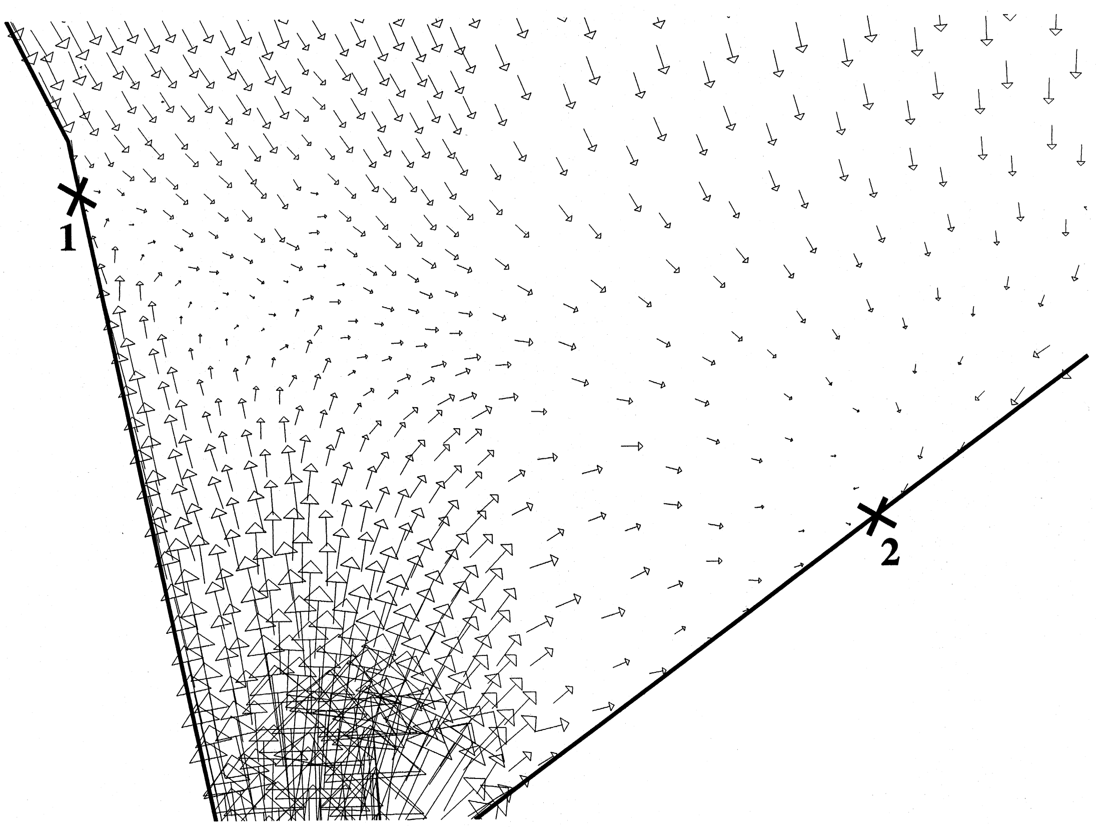

Fig. 23. Detail of flow and position of stagnation points (1 and 2) near the river mouth during tidal reversal, obtained using a mesh containing 4262 triangle mesh.

Acknowledgements—The authors would like to thank the EPSRC for their financial support of this work, staff at Sir William Halcrow and Partners, and Wessex Water.

REFERENCES

1. Zhao, D. H., Shen, H. W., Tabios, G. Q., Tan, W. Y. and Lai, J. S., Finite-volume 2-dimensional unsteady-flow model for river basins. *Journal of Hydraulic Engineering, ASCE7*, 1994, **120**(No.), 863–883.
2. Falconer, R. A. and Chen, Y. P., An improved representation of flooding and drying and wind stress effects in a 2-dimensional tidal numerical-model. *Proceedings of the Institution of Civil Engineers, Part 2. Research and Theory*, 1991, **91,** 659–678.
3. Klevanny, K. A., Matveyev, G. V. and Voltzinger, N. E., An integrated modeling system for coastal area dynamics. *International Journal for Numerical Methods in Fluids3*, 1994, **19**(No.), 181–206.
4. Chandler-Wilde, S. N. and Lin, B., A finite-difference method for the shallow-water equations with conformal boundary-fitted mesh generation. In *Hydraulic and Environmental Modeling—Coastal Waters*, 1992, pp. 507–518.
5. Molls, T. and Chaudhry, M. H., Depth-averaged open-channel flow model. *Journal of Hydraulic Engineering, ASCE6*, 1995, **121**(No.), 453–465.
6. Galland, J. C., Goutal, N. and Hervouet, J. M., TELEMAC—A new numerical-model for solving shallow-water equations. *Advances in Water Resources3*, 1991, **14**(No.), 138–148.
7. Hervouet, J. M., Hubert, J. L., Janin, J. M., Lepeintre, F. and Peltier, E., The computation of free surface flows with TELEMAC: an example of evolution towards hydroinformatics. *Journal of Hydraulic ResearchIssue*, 1994, **32**(Extra), 45–63.
8. Garcia-Navarro, P., Hubbard, M. E. and Preistley, A., Genuinely multidimensional upwinding for the 2d shallow water equations. *Journal of Computational Physics*, 1995, **121,** 79–93.
9. Alcrudo, F. and Garcia-Navarro, P., A high-resolution Godunov-type scheme in finite volumes for the 2d shallow-water equations. *International Journal for Numerical Methods in Fluids6*, 1993, **16**(No.), 489–505.
10. Garcia-Navarro, P. and Priestley, A., A conservative and shape-preserving semi-lagrangian method for the solution of the shallow-water equations. *International Journal for Numerical Methods in Fluids3*, 1994, **18**(No.), 273–294.
11. Toro, E. F., Riemann problems and the WAF method for solving the 2-dimensional shallow-water equations. Philosophical Transactions of the Royal Society of London, Series A. *Physical Sciences and Engineering1649*, 1992, **338**(No.), 43–68.
12. Ambrosi, D., Approximation of shallow-water equations by Roe Riemann solver. *International Journal for Numerical Methods in Fluids2*, 1995, **20**(No.), 157–168.

13. Yang, J. Y. and Hsu, C. A., Computations of free-surface flows 2-dimensional unsteady bore diffraction. *Journal of Hydraulic Research3*, 1993, **31**(No.), 403–414.
14. Glaister, P., Approximate riemann solutions of the 2-dimensional shallow-water equations. *Journal of Engineering Mathematics1*, 1990, **24**(No.), 45–53.
15. Glaister, P., Shock capturing for steady, supersonic, 2-dimensional isentropic flow. *International Journal for Numerical Methods in Fluids7*, 1991, **13**(No.), 883–894.
16. Sleigh, P. A., Gaskell, P. H., Berzins, M., Ware, J. L. and Wright, N. G., A reliable and accurate technique for the modelling of practically occuring open channel flow. In *Proceedings of the Ninth International Conference on Numerical Methods in Laminar and Turbulent Flow*, 881–892, 1995.
17. Berzins, M., Gaskell, P. H., Sleigh, P. A., Spears, W. S., Tomlin, A. and Ware, J. L., An adaptive CFD solver for time dependent environmental flow problems. In *Proceedings of the Institute of Computational Fluid Dynamics Conference*, 1995.
18. Chow, V. T., *Open-Channel Hydraulics*. Civil Engineering Series. McGraw-Hill, 1959.
19. Joe, B. and Simpson, R. B., Triangular meshes for regions of complicated shape. *International Journal for Numerical Methods in Engineering*, 1991, **23,** 987–997.
20. Ware, J. L., *The Adaptive Solution of Time-Dependent Partial Differential Equations in Two Space Dimensions*. PhD thesis, School of Computer Studies, University of Leeds, 1993.
21. Berzins, M., Temporal error control in the method of lines for convection dominated equations. *SIAM Journal of Scientific Computing*, May 1995.
22. Berzins, M. and Ware, J. L., Positive cell-centred finite volume discretisation methods for hyperbolic equations on irregular meshes. *Applied Numerical Mathematics*, 1995.
23. Roe, P. L., Approximate Riemann solvers, parameter vectors, and difference-schemes. *Journal of Computational Physics2*, 1981, **43**(No.), 357–372.
24. Osher, S. and Solomon, F., Upwind difference-schemes for hyperbolic systems of conservation-laws. *Mathematics of Computation158*, 1982, **38**(No.), 339–374.
25. Tan, W. Y., *Shallow Water Hydrodynamics*. Elsevier Oceanography Series. Elsevier, 1992.
26. Alcrudo, F., Garcia-Navarro, P. and Saviron, J. M., Flux difference splitting for 1-D open channel flow equations. *International Journal for Numerical Methods in Fluids9*, 1992, **14**(No.), 1009–1018.
27. Berzins, M. and Furzeland, R. M., An adaptive theta-method for the solution of stiff and nonstiff differential-equations. *Applied Numerical Mathematics1*, 1992, **9**(No.), 1–19.
28. Usseglio-Polatera, J. M. and Sauvaget, P., Dry beds and small depths in 2-d codes for costal and river engineering. In *Computer Methods and Water Resources: Computational Hydraulics*, edited by Barthet Ouzar, Brebbia, 1988.
29. Pennington, S. V. and Berzins, M., New NAG library software for 1st-order partial-differential equations. *ACM Transactions on Mathematical Software1*, 1994, **20**(No.), 63–99.
30. Watson, G. and Peregrin, D. H., Low frequency waves in the surf zone. In *Proceedings of 23rd International Conference on Costal Engineering, Venice*, 1992.
31. Glaister, P., Flux difference splitting for open-channel flows. *International Journal for Numerical Methods in Fluids7*, 1993, **16**(No.), 629–654.
32. Katapodes, N. D., Computing two-dimensional dam-break flood waves. *Journal of Hydraulic Engineering, ASCE9*, 1978, **104**(No.), 1269–1288.
33. Bellos, C. V., Soulis, J. V. and Sakkas, J. G., Computation of 2-dimensional dam-break-induced flows. *Advances in Water Resources1*, 1991, **14**(No.), 31–41.
34. Ordnance Survey. *1:50000 First Series*. Sheet 182, Weston-Super-Mare and Bridgwater.