Development of a 1-D Error-Minimizing Moving Adaptive Grid Method *

Mart Borsboom †
WL | Delft Hydraulics
P.O. box 177
2600 MH Delft
the Netherlands

September, 2000

Abstract

Instead of developing an adaptive grid technique for some discretization method, we develop a discretization technique designed for grid adaptation. This so-called compatible scheme allows to translate the leading term of the local residual directly in terms of a local error in the numerical solution (the numerical modeling error). An error-dependent smoothing technique is used to ensure that higher-order error terms are negligible. The numerical modeling error is minimized by means of grid adaptation. Fully converged adapted grids with strong local refinements are obtained for a steady-state shallow-water application with a hydraulic jump. An unsteady application confirms the importance of taking the error in time into account when adapting the grid in space. We discuss the shortcomings of the present implementation and the remedies currently under development.

1 Introduction

In moving adaptive grid methods both the numerical solution and the grid on which that numerical solution is defined are considered unknown. This offers an enormous increase in numerical modeling flexibility, but also raises the far from easy question of how to couple the grid to the numerical solution. The standard answer is to apply an equidistribution principle: the grid is defined by the equidistribution of a solution-dependent error measure or monitor function over the grid cells. Many different error measures have been proposed in the literature, often chosen heuristically, with little or no justification [12]. Ideally,

*Contribution to the book *Adaptive Method of Lines*, A. vande Wouwer, P. Saucez, B. Schiesser (eds), CRC Press, to be published.

†E-mail: mart.borsboom@wldelft.nl

error measures include all important sources of numerical solution errors, i.e., terms that indicate how grid resolution, grid stretching, grid curvature and grid skewness affect the accuracy of the numerical simulation.

If essential error information is missing or not used properly, grid adaptation may not give any improvement [27]. The use of an incomplete or incorrect error estimate could even lead to an increase of solution errors due to the incorrect grid point distributions and/or the severe grid distortions that it may induce. This can be avoided by adding terms that ensure some form of grid regularity or that limit grid variation, thereby reducing at the same time the adaptability of the grid. The resulting adaptive grid technique will probably require problem-dependent fine tuning as well. These arguments clearly illustrate the importance of a reliable error measure when designing a moving adaptive grid algorithm [12, 17].

Residual-based, a posteriori error estimates for hyperbolic problems of practical interest (in particular nonlinear flow problems) still lack sharpness or are not generally applicable [11, 15, 17]. Using the residual as such is generally not a good idea. The model equations (and hence the residual) can be multiplied by any smooth positive function without changing the solution. Depending on this scaling, a large/small residual may therefore not correspond with a large/small solution error. The relation between the solution error and the residual can be estimated by considering a global dual problem, obtained after linearization, which is usually complex and expensive to solve. Solving a dual problem locally is practically more feasible [23], although it does indicate only the locally generated error and ignores solution errors that are the result of the accumulation during propagation of errors generated elsewhere [14]. However, a small local residual may be due to the cancellation of numerical errors originating from different modeling terms and may therefore not correspond with small errors as such. It is usually also not clear how the local residual depends on the local grid parameters (size, stretching, curvature, skewness), which makes this approach less suited for error-minimizing grid adaptation. Furthermore, it may be difficult to develop a meaningful local dual problem with appropriate local (inflow, outflow) boundary conditions.

Propagation, and hence error propagation, is typical of flow and transport problems; it makes the development of an error-minimizing adaptive grid technique for such applications extremely complicated, because of the obscure and complex relation that exists between the regions where the solution errors are generated and the regions where the solution errors manifest themselves. This applies to numerical errors as well as to physical errors and errors due to incorrect data. Examples of physical modeling errors are the approximate description of turbulence or the neglect of certain aspects like a space dimension or viscous effects. Data errors may consist of uncertainties in, e.g., the initial and boundary conditions, geometry and certain model parameters. When reducing numerical errors through grid adaptation, the presence of these other modeling errors should be taken into consideration as well [7]. In particular, refining the grid is not useful in regions where the solution error is mainly due to the applied physical model or caused by unreliable or incomplete input data.

This strongly limits the usefulness of an adaptive grid technique that merely attempts to minimize the numerical solution error. In complex applications it is however virtually impossible to quantify the effect of physical modeling errors and data errors on the solution, let alone to take the relative importance of this effect into account in the grid adaptation.

We have, therefore, not opted for the development of a grid adaptation method that aims to reduce some upper error bound to a given tolerance level. Instead, our goal is to minimize the part of the numerical solution error that is generated locally as a result of using a numerical approximation of the model equations. We will refer to this error as the *numerical modeling error*. Note that it may be feasible to compare the numerical modeling error with physical modeling errors and data errors. This could then be used to assess its relative importance and hence the usefulness of local (adaptive) grid refinements.

The basis of our method is the approximation of the local residual in terms of a local solution error. We want to avoid the use of the residual as such in combination with a local dual problem because of the disadvantages mentioned before. Investigating this issue more closely, we found that it is generally impossible to reformulate the residual directly in terms of local errors in the numerical solution. The reason for this seems to be the discrepancy that often exists between the discretization of the model equations and the way the numerical solution is represented. For example, the use of piecewise linear polynomials to reconstruct the numerical solution over the whole domain is consistent with the use of a second-order accurate discretization technique. This does not mean however that the second-order interpolation error is representative of the numerical modeling errors.

So instead of analyzing the residual of some numerical scheme, we decided to investigate numerical schemes for their suitability of rewriting the residual as a local solution error. This has led to the development of a discretization technique designed for error analysis and hence for grid adaptation. Using this technique, the residual of any flow or transport equation discretized on a non-uniform, moving grid can indeed be reformulated, at least in 1-D, in terms of a local solution error.

The moving adaptive grid equations are obtained by solving an optimization problem, minimizing the numerical modeling error in the L_1 norm. The L_1 norm has been selected because of its physical relevancy (see also [29]). In particular, the numerical approximation of a discontinuity spread over a fixed number of grid points shows only the expected first-order error behavior if it is measured in the L_1 norm. This is consistent with the local order of accuracy of the numerical scheme.

2 Two-step numerical modeling

In the numerical error analysis that we will present we will make extensive use of truncated Taylor-series expansions. A smoothing technique is applied prior to the discretization to ensure that higher-order error terms are negligible. This

makes the numerical modeling process essentially a two-step procedure. In the first step the problem to be solved is regularized by adding a suitable form of smoothing; in the second step the regularized problem with smooth solution is discretized. The amount of smoothing in the first step is controlled by the error that is made in the second step by an error feedback loop. Smoothing or regularization is used frequently in adaptive grid techniques [5, 8, 12, 17].

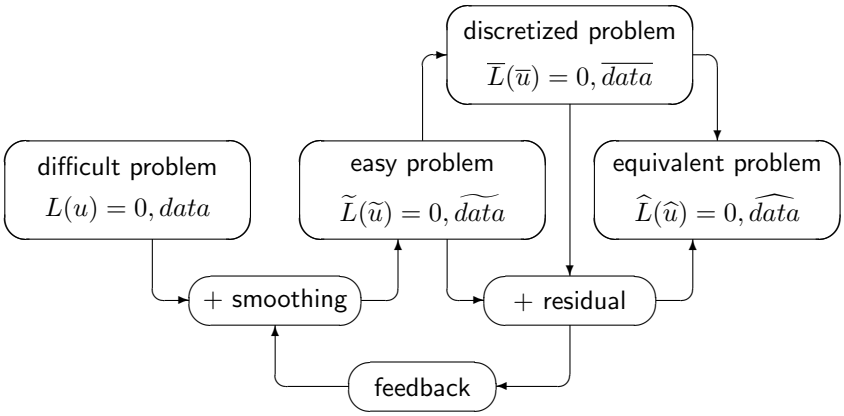

Figure 1: Outline of the two-step numerical modeling technique

Figure 1 shows the elements of the two-step method. We discern a smoothing step and a discretization step connecting four different problems:

- the **difficult problem**, i.e., the original physical model problem, which solution may include steep gradients and discontinuities,
- the **easy problem**, i.e., the regularized problem, which solution is smooth enough to be discretized with sufficient accuracy,
- the **discretized problem**, obtained upon the discretization of the easy problem,
- the **equivalent problem**, i.e., the differential problem equivalent with the discretized problem.

The smoothing step converts the difficult problem into the easy problem by explicitly adding suitable smoothing terms. The discretization step converts the easy problem into the equivalent problem by means of a suitable discretization of the easy problem. To obtain the second conversion in an explicit form, we determine the residual associated with the discretized problem. The residual can be viewed as the continuous equivalent of the discretization error.

The difficult, easy and equivalent problem are all differential problems. They each consist of a system of partial differential equations ($L(u) = 0$, $\widetilde{L}(\widetilde{u}) = 0$, $\widehat{L}(\widehat{u}) = 0$) and a set of data ($data$, $\widetilde{data}$, $\widehat{data}$) containing, e.g., the boundary and initial conditions. Functions and parameters required to fully specify each

problem (geometry and source terms, for example) are also included in the data. The discretized problem on the other hand consists of a system of algebraic equations ($\overline{L}(\overline{u}) = 0$) supplemented with a discrete data set ($\overline{data}$).

The discretized problem is usually the only problem that is solvable. However, we do not know the differential problem corresponding with its solution $\overline{u}$. In contrast, the difficult and easy problem are fully specified but their solution (respectively u and $\widetilde{u}$) is largely unknown. The link between both is the equivalent problem. The equivalent solution $\widehat{u}$ and data $\widehat{data}$ are a close and smooth approximation of the solution and data of the discretized problem, while the equivalent system of equations $\widehat{L}(\widehat{u}) = 0$ is an approximation of the system of partial differential equations satisfied by $\widehat{u}$ given $\widehat{data}$. If the discretization is consistent, the equivalent problem is also an approximation of the easy problem that in turn is an approximation of the difficult problem.

Some form of two-step modeling is used in virtually any discretization method for flow and transport problems of practical interest. Smoothing may have been added explicitly (i.e., prior to the discretization) in the form of artificial dissipation terms in the equations, or implicitly by incorporating a dissipative mechanism like flux limiters inside the discretization method. Usually, the purpose of adding dissipation is to reduce wiggles or to ensure monotonicity. Here we demand that it guarantees a sufficiently smooth solution, i.e., negligibly small higher-order discretization errors, to allow a meaningful error analysis that can be used in a grid adaptation procedure.

A simple example may illustrate the connection between smoothing and grid adaptation. We consider a smooth function with a discontinuity at $x = 0$ and approximate this function by means of a continuous piecewise linear discrete function defined on the grid $x = \pm 1, \pm 3, \ldots$. Because the discontinuity is at the center of a grid cell, its best possible numerical approximation is only one grid cell wide. We now discretize the function on the grid $x = 0, \pm 1, \pm 2, \ldots$ and observe that there is no improvement in accuracy, despite the fact that the resolution has been doubled. The reason for this is obvious: the discontinuity is now at a grid point and must be spread over two grid cells. It is even possible to get worse results on a finer grid, simply because the position of the discontinuity changes from a cell center to a grid point (consider the grid $x = 0, \pm 1.5, \pm 3, \ldots$). Such a function has been considered in [2]. The results presented in that paper show that, although the global error behavior in the L_1 norm is indeed first order, without smoothing of the discontinuity the approximation error is an irregular function of the number of grid points. When the function is first properly smoothed, the L_1 approximation error decreases uniformly as the number of grid points increases, showing a clear first-order behavior independent of the position of the grid points.

The dependence of the error on grid point position is unacceptable in the context of the use of moving adaptive grid techniques where we need a monotone relation between the numerical accuracy and the grid resolution. The example shows that the behavior of higher-order error components can be rather erratic and becomes dominant if the solution is not smooth. This can be avoided by

smoothing the function that is to be approximated (e.g., the solution of a differential problem) over a sufficient number of grid cells. It can easily be verified that the smoother the function, the less sensitive its numerical approximation is to the position of the grid points. However, smoothing introduces another form of numerical error, so in practice a compromise needs to be found between the amount of smoothing (should be as small as possible) and the amount of spreading (must be large enough). In particular, no additional smoothing is required if the function is already sufficiently smooth. These requirements are precisely the design criteria of the two-step method.

If designed correctly, explicitly added artificial dissipation weakens (infinitely) steep gradients by spreading them over enough but not too many grid cells. The numerical scheme should then be able to handle those gradients correctly, meaning that the effect of discretization errors should be sufficiently small. The mechanism is comparable with upwind techniques and flux limiters, but an essential difference is that flux limiters are mainly designed to spread steep gradients over the smallest number of grid cells possible without causing any over- or undershoots (see, e.g., [13]). They are not designed to keep numerical errors below an acceptable level. Using upwind methods, solutions can therefore be obtained that look reasonable and yet are completely wrong due to a lack of spatial resolution (see, e.g., [1, 19]). This is consistent with the conclusion from the above example and indicates that the preservation of monotonicity is no guarantee for numerical accuracy. On the other hand, explicitly added artificial dissipation generally does not give that guarantee either.

The design of a suitable form of artificial dissipation has been the subject of many publications (see, e.g., [13] for an overview). We prefer the approach proposed half a century ago by Von Neumann and Richtmyer who suggested to 'utilize the well-known effect on shocks of dissipative mechanisms, such as viscosity' [26]. The advantage of the artificial viscosity concept is that no physically unrealistic terms are introduced. In particular, no dissipation terms that would destroy mass conservation are added to the continuity equation.

There remains the problem of how to scale the artificial viscosity. Since its purpose is to regularize the difficult problem, it must depend in some way on the discretization error. This feedback should ensure that the easy problem will always be easy enough for the discretization error to be sufficiently small: the discretization error should be such that the neglected higher-order error terms are indeed negligible.

Once these elements have been designed properly, the development of an error-minimizing moving adaptive grid technique is relatively straightforward. The discretization error is a function of the grid, and can be minimized by moving the grid points. When the discretization error is minimized, the artificial viscosity is minimized as well because of the feedback loop. It will be shown that this minimizes all coefficients in the equivalent differential equation that are different from the original differential equation of the difficult problem. In other words, the perturbations introduced in the difficult problem by the numerical solution technique are minimized and thereby their local effect on the solution (the numerical modeling error).

Notice that increasing the viscosity artificially allows to compare it directly with the true physical viscosity, to assess the relative importance of the artificial dissipation. This may prove to be very useful in view of physical modeling errors. When numerical results are compared with measurement data, it is then easily verified whether deviations should be attributed to an incorrect physical viscosity model (e.g., a turbulence model) or to a large artificial viscosity. It may even be possible to incorporate knowledge about the validity of the physical model in the grid adaptation procedure, to avoid that grid points are moved to regions where the model is not very accurate anyway.

3 1-D shallow-water equations

The physical problem that we consider is the modeling of free-surface flow through an open channel section. In many practical applications the vertical and transversal length scales are small compared to the longitudinal scales and can be neglected. Assuming constant density ρ and restricting ourselves to channels of constant width W, the flow can then be modeled by the 1-D equations (see, e.g., [3]):

$$\frac{\partial d}{\partial t} + \frac{\partial q}{\partial x} = 0 \,, \tag{1}$$

$$\frac{\partial q}{\partial t} + \frac{\partial q^2/d}{\partial x} + gd\frac{\partial h}{\partial x} + g\frac{Pq|q|}{Wd^2C^2} = \frac{\partial}{\partial x}\left(\nu_{\text{art}} d \frac{\partial q/d}{\partial x}\right) \,. \tag{2}$$

Space coordinate x [m] is defined along the axis of the channel. The unknowns of the flow equations are the water depth d [m] and the depth-integrated flow velocity in x-direction q [m^2/sec]. Depth-averaged flow velocity u [m/sec] is equal to q/d. Gravitational acceleration g [m/sec^2] is a given constant parameter. Chézy coefficient C [$\text{m}^{1/2}$/sec] is a friction parameter to account for friction losses due to the bottom friction. Water level h [m] is the sum of d and the given bottom level of the channel z_b [m]:

$$h = d + z_b \,, \tag{3}$$

while wetted perimeter P [m] is defined as $P = W + 2d$.

Continuity equation (1) is a mass conservation law. It describes the balance between the rate of change of mass in a cross-section and the net mass flow entering that cross-section for constant ρ and W. The terms in the left-hand side of momentum equation (2) represent the rate of change of momentum, the net momentum flow, the hydrostatic pressure force and the bottom friction force respectively. In the right-hand side of (2) we have the added artificial smoothing term, so (1) and (2) form in fact the equations of an easy problem (cf. Figure 1). Since we will only consider the easy problem, we have omitted the tilde for convenience. The form of the smoothing term is equal to the physical viscosity term integrated over the water depth, neglecting non-conservative parts to avoid artificial loss of momentum. Its viscosity coefficient ν_{art} [m^2/sec] is of

course artificial, although the same formulation could also be used to improve the modeling of physical effects [18].

The equations (1) and (2) can be recombined to the characteristic equations:

$$(c \mp u) \text{ cont.eq. } (1) \pm \text{ mom.eq. } (2) \ , \tag{4}$$

with $c = \sqrt{gd}$ the wave celerity. These equations describe the propagation of the Riemann variables $(c \mp u)\partial d \pm \partial q$ along the characteristics $dx/dt = u \pm c$. The flow behavior depends to a large extent on the direction of the characteristics. Introducing the Froude number $\mathrm{Fr} = |u|/c$ we make a distinction between subcritical flow ($\mathrm{Fr} < 1$; propagation in both directions), critical flow ($\mathrm{Fr} = 1$; one propagation direction vanishes) and supercritical flow ($\mathrm{Fr} > 1$; only propagation in downstream direction). Note that in the shallow-water *model* there will always be some information moving upstream (even if $\mathrm{Fr} > 1$), due to the diffusive transport introduced by the artificial viscosity. This is perfectly acceptable if the effect is small enough.

In regions where the solution of (1) and (2) is differentiable, the equations can be recombined to the energy equation:

$$\begin{aligned} &\frac{\partial}{\partial t}\left(d\left(\frac{1}{2}u^2 + gh - \frac{1}{2}gd\right)\right) + \frac{\partial}{\partial x}\left(ud\left(\frac{1}{2}u^2 + gh\right)\right) = \\ &\quad = -g\frac{Pu^2|u|}{WC^2} - \nu_{\mathrm{art}} d\left(\frac{\partial u}{\partial x}\right)^2 + \frac{\partial}{\partial x}\left(u\nu_{\mathrm{art}} d\frac{\partial u}{\partial x}\right) \ . \end{aligned} \tag{5}$$

Ignoring the dissipation terms in the right-hand side, the steady-state solution of equation (5) reads (from (1) we obtain $ud = q = \text{constant}$):

$$h + \frac{u^2}{2g} = \text{constant} \ . \tag{6}$$

This Bernoulli equation shows that the energy head, $h + \frac{1}{2}u^2/g$, is constant in smooth stationary frictionless flow; no energy is dissipated. The equation is not valid across a steady discontinuous hydraulic jump where we have to apply the Rankine-Hugoniot relations obtained from (1) and (2):

$$\begin{aligned} (ud)_1 &= (ud)_2 \ , \\ \left(u^2 d + g\frac{d^2}{2}\right)_1 &= \left(u^2 d + g\frac{d^2}{2}\right)_2 \ . \end{aligned} \tag{7}$$

The indices 1 and 2 indicate the state left and right of the discontinuity. Solving (7) across a hydraulic jump gives the energy loss across the jump. Further details can be found in [4]. Using the steady-state solution of (1), discharge $q = ud = \text{constant}$, the solution of (6) (in regions where the solution is smooth) and (7) (across a hydraulic jump) can be determined analytically, given suitable boundary conditions. This provides a useful analytical bench-mark solution to compare numerical solutions with [10, 21].

We point out that the discontinuous hydraulic jump is only a solution of (1) and (2) if the artificial viscosity is set to zero, i.e., if the model equations are considered to form a difficult problem (cf. Figure 1). With the addition of the artificial smoothing term it has become an easy problem; discontinuities are spread and 'replaced' so to speak by gradients of limited steepness. It is easily verified that the spreading is proportional to the value of ν_{art}.

At some distance of a smooth but nevertheless steep gradient the solution gradients will be small again, and the effect of the artificial viscosity negligible. Neglecting the variations in channel geometry, it follows from momentum equation (2) that the jump relations (7) (or the unsteady equivalent) are still satisfied across the jump region. So if only a moderate amount of smoothing is applied, the behavior of discontinuities is still modeled accurately. The condition to be satisfied here is that within a smoothing region the channel variation must be small. This condition was formulated 50 years ago already by Von Neumann and Richtmyer: '... for the assumed form of dissipation, and for many others as well, the Rankine-Hugoniot equations are satisfied provided the thickness of the shock layers is small in comparison with other physically relevant dimensions of the system' [26].

An interesting aspect of the easy problem is that, because the solution is smooth and differentiable, energy equation (5) is valid everywhere, also across the jump regions. In fact, the sudden energy drop across the jump is replaced by a steep decrease of the energy head due to the artificial viscosity. Since jump conditions (7) are still valid, the net result must be the same.

White presents an analysis of the structure of a 1-D viscous aerodynamic shock wave [28]. The analysis can be applied to hydrodynamic shocks as well. His results confirm that the thickness of the shock is proportional to the size of the viscosity coefficient. A rather surprising result is the entropy overshoot in the shock which is due to the energy redistribution caused by the viscous term. In the inviscid limit the overshoot becomes a peak of zero thickness and vanishes, leaving only the well-known discontinuous increase of entropy. In our model the entropy overshoot corresponds with an undershoot of the energy head. The last term in the right-hand side of (5) is responsible for this effect. It is first negative (u decreases, hence $\partial u/\partial x$ becomes strongly negative inside the shock) but becomes positive at the end of a viscous shock layer. Overall the third term does not affect the energy across a shock because of its conservative form. The energy across a shock decreases because of the negative-definite second term in the right-hand side that also prevents the development of non-physical expansion shocks (the entropy condition, cf. [13]). At the end of a shock however, where the third term is strongly positive and dominant, the right-hand side of (5) becomes positive causing a small energy uplift. As explained before, the overall energy loss across a viscous shock can still be a very close approximation of the inviscid shock loss.

The conclusion that can be drawn from this discussion is that, although the solutions are different locally, globally the solution of the easy problem and the difficult problem may be expected to be virtually the same provided that not too much artificial viscosity is added. This is precisely the purpose of grid

adaptation; the amount of artificial dissipation required depends both on the solution and on the grid, so by optimizing the grid as a function of the solution the artificial viscosity can be minimized and the accuracy maximized. This too was perceived already by Von Neumann and Richtmyer: 'the qualitative influence of these terms (read: the artificial viscosity) can be made as small as one wishes by choice of a sufficiently fine mesh' [26].

A condition to be fulfilled by the numerical scheme is that the solution of the easy flow problem is calculated with sufficient accuracy. To be able to capture all details of the (artificially thickened) viscous shock, including the undershoot of the energy head, shocks must be spread over at least several grid cells. It seems that this argument can be reversed: if a shock is not modeled as a discontinuity (any shock-capturing method), it should be modeled as a viscous shock of a certain thickness in order to be physically realistic. Upwind schemes that spread shocks over only one or two grid cells may not be capable of modeling the entropy overshoot, and hence the dynamics, of that shock correctly. This observation is in line with the discussion of Section 2 and emphasizes the importance of sufficient resolution and hence smoothing.

4 Compatible discretization

Our investigations have revealed that there seems to be only one way to obtain a discretization of flow and transport equations with the property that the residual can be formulated in terms of errors in the numerical solution and other variables. The key element of this approach is the definition of unique approximations per grid cell of *all* variables. We have used piecewise polynomial approximations that are defined on a uniform grid in the parameter space or computational space (ξ, τ). It is convenient for the discretization of the model equations and for the error analysis to define the differential problem in this computational space as well.

Since we consider moving grids, the mapping of the computational space (ξ, τ) onto the physical space (x, t) is defined by the two functions:

$$x = x(\xi, \tau) \ , \ t = t(\tau) \ . \tag{8}$$

Actually, it is defined by several of such functions because each problem that we consider in the two-step modeling technique (cf. Figure 1) requires its own coordinate transformation. This is only relevant for the two problems related to the numerical scheme: the discretized problem and the equivalent problem. We will see later that it is indeed essential to consider for each of these two problems a separate coordinate transformation.

The discretization in time that we will apply is a two-level method. As a consequence, each time step can be considered separately, i.e., each time step the transformation in time can be redefined including a change of the size of the time step. This permits to restrict ourselves to transformation functions (8) that are linear in τ:

$$t_\tau = 1 \ , \ x_{\tau\tau} = 0 \ , \tag{9}$$

where we have used the convention that a subscript that is a coordinate denotes differentiation with respect to that coordinate.

The (moving) space-time grid that we use is as indicated in Figure 2.

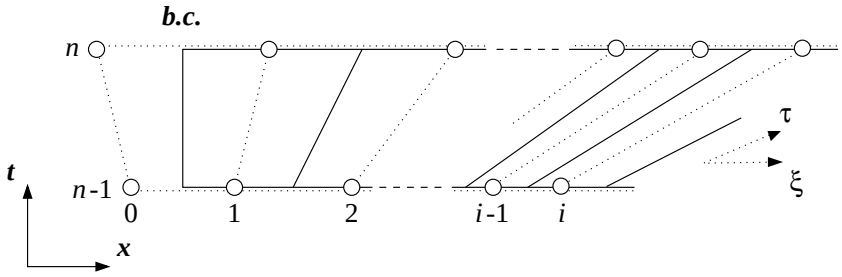

Figure 2: Grid in physical time and space

The coordinates (x_i^n, t^n) of the grid points (i, n), indicated in Figure 2 by the circles, determine the coordinate transformation of the discretized problem. Using (bi)linear interpolations per grid cell, the transformation is defined by:

$$\begin{aligned}\overline{x}(\xi,\tau) &= \frac{\xi_i - \xi}{\Delta\xi}\left(\frac{\tau^n - \tau}{\Delta\tau}x_{i-1}^{n-1} + \frac{\tau - \tau^{n-1}}{\Delta\tau}x_{i-1}^{n}\right) \\ &\quad + \frac{\xi - \xi_{i-1}}{\Delta\xi}\left(\frac{\tau^n - \tau}{\Delta\tau}x_{i}^{n-1} + \frac{\tau - \tau^{n-1}}{\Delta\tau}\Delta\xi x_{i}^{n}\right)\ , \\ \overline{t}(\xi,\tau) &= \frac{\tau^n - \tau}{\Delta\tau}t^{n-1} + \frac{\tau - \tau^{n-1}}{\Delta\tau}t^{n}\ ,\end{aligned} \tag{10}$$

with:

$$\begin{aligned}\xi_{i-1} \le \xi \le \xi_i\ , i = 1,\ldots,I+1\ , \\ \tau^{n-1} \le \tau \le \tau^n\ , n = 1,\ldots,N\ .\end{aligned}$$

We have $\xi_i = \xi_{i-1} + \Delta\xi, i = 1,\ldots,I+1$, with $\Delta\xi$ the size of the uniform grid in computational space. The grid consists of $I+1$ grid cells, with I grid points inside the domain and 2 virtual grid points outside the boundaries (cf. Figure 2). The reason for this will become clear in Subsection 4.1. In order to satisfy (9) we take $\tau^n - \tau^{n-1} = \Delta\tau = \Delta t$ within each time step $[\tau^{n-1}, \tau^n]$. As explained before, this does not preclude the use of different values of Δt in different time steps. The total number of time steps is N.

Using (8) and (9), the equations (1) and (2) transformed to computational space can be written as:

$$\frac{\partial(x_\xi d)}{\partial\tau} + \frac{\partial(q - x_\tau d)}{\partial\xi} = 0\ , \tag{11}$$

$$\frac{\partial(x_\xi q)}{\partial\tau} + \frac{\partial((q/d - x_\tau)q)}{\partial\xi} + gd\frac{\partial h}{\partial\xi} + x_\xi g\frac{Pq|q|}{Wd^2C^2} = \frac{\partial}{\partial\xi}\left(\frac{\nu_{\rm art}d}{x_\xi}\frac{\partial q/d}{\partial\xi}\right)\ . \tag{12}$$

This result has been obtained upon multiplying the transformed equations by x_ξ and rearranging the terms to recover the conservative form of the original system formulated in physical space. Notice that the coefficients t_τ have vanished because of (9).

For the discretization of (11) and (12) we introduce uniquely defined finite-dimensional function approximations of all variables: piecewise linear interpolations in computational space but backward piecewise constant (and hence discontinuous) extrapolations in computational time. The latter makes that the discretization in time will have the form of a backward Euler scheme. It provides the amount of dissipation in time required to ensure stability for any size of the time step, and also simplifies the analysis of the error in time. The accuracy obtained with this first-order scheme in time may still be quite acceptable, provided that we are able to design a moving adaptive grid method capable of aligning the grid properly with the solution. So we decided to use:

$$\overline{d}(\xi,\tau) = \frac{\xi_i - \xi}{\Delta\xi} d_{i-1}^n + \frac{\xi - \xi_{i-1}}{\Delta\xi} d_i^n \ , \tag{13}$$

$$\xi_{i-1} \le \xi \le \xi_i \ , \ i = 1, \ldots, I+1 \ ,$$
$$\tau^{n-1} < \tau \le \tau^n \ , \ n = 1, \ldots, N \ .$$

Similar expressions are used for unknown $\overline{q}$, wetted perimeter $\overline{P}$ and artificial viscosity coefficient $\overline{\nu}_{\rm art}$. The expression is assumed to exist for discretized water level $\overline{h}$. As before, we have used the overbar to indicate discrete functions (cf. Figure 1).

All discrete functions, i.e., $\overline{x}$ (eq. (10)), $\overline{d}$ (eq. (13)), $\overline{q}$, $\overline{h}$, $\overline{P}$ and $\overline{\nu}_{\rm art}$, must be (made) sufficiently smooth (the smoothness of $\overline{t}$ is guaranteed because of (9)). This condition must be fulfilled in order that higher-order error terms can be neglected in the error analysis. The smoothing step with error feedback loop (cf. Figure 1) should take care of that. As for the unknowns $\overline{d}$ and $\overline{q}$, their smoothness is realized by the artificial viscosity term. To ensure smoothness of the artificial viscosity coefficient (an essential part of the method!), a separate equation is applied:

$$\nu_{\rm art} - \alpha\Delta\xi^2 \frac{\partial^2 \nu_{\rm art}}{\partial \xi^2} = c_\nu Err_\nu \ . \tag{14}$$

Function Err_ν is an error expression of dimension [m^2/sec] that will be specified later. For the moment it is sufficient to mention that Err_ν (and hence $\nu_{\rm art}$) is $O(\Delta x^3)$ in regions where the solution is smooth and $O(\Delta x)$ in regions where steep gradients develop. This makes that the artificial viscosity mechanism does not affect the formal second-order accuracy of the scheme, while discontinuities will be spread over a fixed number of grid cells that depends on the value of constant scaling coefficient c_ν.

The amount of smoothing applied in (14) is determined by constant coefficient α. The smoothing of $\nu_{\rm art}$ is required to reduce the unreliable higher-order error information that may be present in Err_ν to an insignificant level. These errors are partly due to the neglect of higher-order errors in the error analysis

and partly introduced by approximating Err_ν by means of a discretization (details later). The smoothing of $\nu_{\rm art}$ is in computational space because Err_ν will be determined and discretized in computational space. The use of a constant smoothing coefficient α is sufficient because the artificial viscosity coefficient by itself is already a higher-order term.

An implicit smoothing equation like (14) is used frequently in moving adaptive grid methods. To see this, replace $\nu_{\rm art}$ by smoothed grid-point concentration n, α by $\alpha(1+\alpha)$ and $c_\nu Err_\nu$ by non-smooth point concentration $\tilde{n}$, and discretize the equation by means of finite differences. The result is the equation that Dorfi and Drury use to ensure grid smoothness [5]. Huang and Russell show that smoothing of the monitor function and of the grid point concentration are equivalent [16]. Implicit smoothing of grid point concentration has been used in, e.g., [9, 25, 30, 31]. An explicit monitor smoothing technique approximating implicit smoothing has been used in [20, 22].

We will use an equation like (14) also for the smoothing of the error expressions that are used in the moving adaptive grid procedure, to eliminate higher-order errors that are not included properly. This automatically takes care of grid smoothness, i.e., no special measures are required to ensure that $\overline{x}$ is sufficiently smooth.

One variable may not be sufficiently smooth: $\overline{h}$. This is presently one of the main shortcomings of the method. Water level h is equal to the sum of bottom level z_b and water level d (cf. equation (3)) and considered as a function of d. It is obvious that smoothness of $\overline{d}$ is no guarantee for smoothness of $\overline{h}$; that depends entirely on the given profile z_b that in practice may be highly irregular. Moreover, while $\overline{d}$ is a function that is piecewise linear in computational space and piecewise constant in computational time, the behavior of $\overline{h}$ can be anything depending on how z_b is specified. Usually, z_b is given as a piecewise linear function based on the available data of a channel's geometry. This does not make $\overline{h}$ a function that is linear per grid cell because the coordinates where the geometry is specified will rarely coincide with grid points. Discrete function $\overline{h}$ will generally not be piecewise constant in time either, even though z_b is constant in time. This is because z_b is constant in time in physical space, not in computational space.

All this is ignored at present. That is, also for $\overline{h}$ an expression like (13) is used, but a rather rough approximation is used to define the grid point values h_i^n:

$$h_i^n = d_i^n + z_b(x_i^{n-\frac{1}{2}}) \;, \tag{15}$$

with:

$$x_i^{n-\frac{1}{2}} = \overline{x}(\xi_i, \tau^{n-\frac{1}{2}}) = \frac{1}{2}\left(x_i^{n-1} + x_i^n\right) \;.$$

To obtain the best possible accuracy in time we evaluate the bottom level at the position of the moving grid halfway the time steps.

4.1 Discretized shallow-water equations

In the previous part we have defined function approximations that are piecewise linear in computational space and piecewise constant in computational time. It is reasonable to assume that this is sufficient to construct a discretization that is second-order accurate in space and first-order accurate in time, in agreement with the leading interpolation errors. However, the first derivative of a piecewise linear approximation is piecewise constant, and only second-order accurate at cell centers. The second derivative of a piecewise linear function is not defined. As a result, (11) and (12) cannot be discretized directly.

The obvious solution is to consider a weak formulation, integrating the model equations in space from cell center to cell center which are the only positions where the viscous fluxes can be evaluated with second-order accuracy. This defines automatically a finite volume technique, with volumes as indicated by the solid lines in Figure 2. Since all discrete functions have been specified, the discretization in space is now fully defined and straightforward to obtain.

It is important to evaluate the integrals in space of the different terms, with the functions replaced by their discrete approximations, with *at least* fourth-order accuracy. Second-order accurate approximations of the integrals, obtained by using, e.g., 1-point Gauss quadrature rules, are not allowed. Although that would lead to a second-order accurate discretization in space as well, it would *not* lead to a *compatible* discretization. It would introduce errors that are of the same order as the interpolation errors, thereby effectively modifying the defined discrete functions. In consequence, the interpolation error of these functions would not be the only source of discretization errors anymore. To make sure that the second-order interpolation errors are included with at least second-order accuracy, the integrals must be approximated fourth-order accurate or better.

We give two examples of how this compatible discretization in space works out in practice, keeping everything in time continuous for the moment (Method Of Lines approach). The time derivative of (12) is discretized in space according to:

$$\begin{aligned}\int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \frac{\partial (x_\xi q)}{\partial \tau}\, d\xi &\approx \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \frac{\partial (\overline{x}_\xi \overline{q})}{\partial \tau}\, d\xi = \\ &= \frac{\partial}{\partial \tau}\left((x_i - x_{i-1}) \frac{q_{i-1} + 3q_i}{8} + (x_{i+1} - x_i) \frac{3q_i + q_{i+1}}{8} \right) ,\end{aligned} \tag{16}$$

while the space discretization of the artificial viscosity term of (12) reads:

$$\int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \frac{\partial (\nu_{\mathrm{art}} d / x_\xi \partial (q/d)/\partial \xi)}{\partial \xi}\, d\xi \approx \left(\frac{\overline{\nu}_{\mathrm{art}} \overline{d}}{\overline{x}_\xi} \frac{\partial \overline{q}/\overline{d}}{\partial \xi} \right)_{i+\frac{1}{2}} - \left(\frac{\overline{\nu}_{\mathrm{art}} \overline{d}}{\overline{x}_\xi} \frac{\partial \overline{q}/\overline{d}}{\partial \xi} \right)_{i-\frac{1}{2}} , \tag{17}$$

with:

$$\left(\frac{\overline{\nu}_{\text{art}}\overline{d}}{\overline{x}_\xi}\frac{\partial\overline{q}/\overline{d}}{\partial\xi}\right)_{i+\frac{1}{2}} = \left(\frac{\overline{\nu}_{\text{art}}}{\overline{x}_\xi}\frac{\partial\overline{q}}{\partial\xi} - \frac{\overline{\nu}_{\text{art}}\overline{q}}{\overline{x}_\xi\overline{d}}\frac{\partial\overline{d}}{\partial\xi}\right)_{i+\frac{1}{2}} =$$

$$= \frac{\nu_{\text{art},i}+\nu_{\text{art},i+1}}{2}\left(\frac{q_{i+1}-q_i}{x_{i+1}-x_i} - \frac{q_i+q_{i+1}}{d_i+d_{i+1}}\frac{d_{i+1}-d_i}{x_{i+1}-x_i}\right).$$

Note that this discretization has *not* been obtained by taking the average of $\overline{\nu}_{\text{art}}\overline{d}/\overline{x}_\xi$ and the difference of $\overline{q}/\overline{d}$ evaluated at the grid points. That would imply that certain combinations of variables rather than the variables themselves are approximated piecewise linearly. That is incompatible with the piecewise linear approximation of those variables used in the other terms (e.g., in time discretization (16)).

Our research has indicated that it is this mix of different variable approximations often encountered in numerical discretizations that makes it impossible to express discretization errors made in the equations in terms of errors in the numerical solution. One would expect this error to be some interpolation error, but if several interpolation errors have been mixed together during the discretization process, it is obviously not possible anymore to recover a well-defined one afterwards. See also [2].

There are still a few details to be filled in. One concerns the discretization of (14) which reads:

$$\left(\frac{3}{4}+2\alpha\right)\nu_{\text{art},i} + \left(\frac{1}{8}-\alpha\right)(\nu_{\text{art},i-1}+\nu_{\text{art},i+1}) = \frac{c_\nu}{\Delta\xi}\int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} Err_\nu \, d\xi \,. \quad (18)$$

The discretization of the integral in the right-hand side is not critical; higher-order errors are irrelevant in the determination of ν_{art} and are damped by the smoothing anyway. The expression for Err_ν will be given in Subsection 5.3.

The discretization in time is simply obtained by evaluating the equations at the central time level $t^{n-\frac{1}{2}} = \frac{1}{2}(t^{n-1}+t^n)$ using the discrete function approximations in time. This is within $O(\Delta\tau^2)$ equal to the discretization obtained by integrating the equations over one time step, which is sufficient for compatibility since the time discretization is only first-order accurate.

As for the boundary conditions, they have to be applied at a cell center to allow the whole domain to be covered with finite volumes (cf. Figure 2). This is rather fortunate since both Dirichlet and Neumann boundary conditions can then be discretized with second-order accuracy using only two grid points. Compatibility is no problem either. We will consider only subcritical flow at boundaries, and apply Dirichlet conditions (either q or h imposed) supplemented with equation (4) describing the behavior of the outgoing characteristic. The discretization of the characteristic equation is obtained in two steps:

- The flow equations (11) and (12) are discretized at the boundaries using the defined discrete approximation of the variables. Because the approximation is linear in space, second derivatives vanish, but the artificial

viscosity term is still contributing to the discretization. To be able to discretize that term at the boundary, we write it as:

$$\frac{\partial}{\partial\xi}\left(\frac{\nu_{\rm art}d}{x_\xi}\frac{\partial q/d}{\partial\xi}\right) = \\ = \frac{\nu_{\rm art}}{x_\xi}\left(\frac{\partial^2 q}{\partial\xi^2} - \frac{q}{d}\frac{\partial^2 d}{\partial\xi^2}\right) + \frac{1}{x_\xi}\left(\frac{\partial\nu_{\rm art}}{\partial\xi} - \frac{\nu_{\rm art}}{d}\frac{\partial d}{\partial\xi}\right)\left(\frac{\partial q}{\partial\xi} - \frac{q}{d}\frac{\partial d}{\partial\xi}\right) .$$

Here, we have used $x_{\xi\xi} = 0$ since the grid is assumed to be uniform at the boundaries. The discretization of the second derivatives in the right-hand side vanishes at the boundaries; the compatible discretization of the other terms is straightforward.

- Using (4), the discretized flow equations at the boundaries are recombined to the equation for the outgoing characteristic.

We end this Subsection with the boundary conditions for equation (14) that takes care of the smoothing of $\nu_{\rm art}$. Assuming a constant error level and hence constant $\nu_{\rm art}$ outside the domain, the discretization of this equation at the left boundary becomes (compare with (18)):

$$\left(\frac{1}{2} + \alpha\right)\nu_{{\rm art},0} + \left(\frac{1}{2} - \alpha\right)\nu_{{\rm art},1} = c_\nu Err_{\nu,\frac{1}{2}} . \tag{19}$$

A similar condition is applied at the right boundary. It is not exactly compatible, but it ensures that the smoothing of Err_ν is extended up to and including the boundaries. Note that for $\alpha \to \infty$, condition (19) converges to a homogeneous Neumann condition for $\nu_{\rm art}$. This is the condition that is usually applied in combination with implicit smoothing [5, 16].

4.2 Iterative solution algorithm

With reference to Figure 1, we write the discretization of the flow equations per time step or steady state in the compact form $\overline{L}(\overline{u}^{n-1}, \overline{u}^n) = 0$. Solution vector $\overline{u}$, not to be confused with velocity u, consists of the two discrete functions $\overline{d}, \overline{q}$ which vectors of grid point values $(d_i)^T$, $(q_i)^T$ are to be determined each time step or each steady state. We will omit in the description of the solution algorithm the solution at the previous time level $\overline{u}^{n-1} = (d_i^{n-1}, q_i^{n-1})^T$ since its contribution, only relevant in unsteady calculations, is known. Superscript n will be omitted as well. The determination of the grid-point coordinates x_i^n at the next time level or steady-state level will be considered later.

The basis of the iterative algorithm that we use to solve the algebraic system of nonlinear discretized flow equations is the Newton method. We employ a so-called pseudo or dual time-step technique to improve its robustness, solving each iteration:

$$\left(\frac{\Delta x}{\Delta t_{\rm pseu}^{m-1}} + \left.\frac{\partial\overline{L}}{\partial\overline{u}}\right|^{m-1}\right)(\overline{u}^m - \overline{u}^{m-1}) = -\overline{L}(\overline{u}^{m-1}) . \tag{20}$$

Superscript m denotes the iteration level. The size of pseudo time step Δt_{pseu} is in [sec]. The added pseudo time-step coefficient $1/\Delta t_{\text{pseu}}^{m-1}$ is multiplied by local grid size Δx, reflecting the integration over the finite volumes of the flow equations. The residual in the right-hand side of (20) and the Jacobian $\partial\overline{L}/\partial\overline{u}$ (the block-tridiagonal linearization matrix) are evaluated at the previous iteration level $m-1$. Linearization of complicated coefficient ν_{art} is not feasible. Instead, we use underrelaxation with an underrelaxation coefficient of 0.8 and multiply the part of $\partial\overline{L}/\partial\overline{u}$ stemming from the linearization of the artificial viscosity term by a factor 1.3. Without the latter, a much stronger underrelaxation of ν_{art} is usually required. We found that the combination of both measures is sufficient to stabilize the implicit iterative solver while maintaining a good performance, despite the fact that ν_{art}, which can be very important locally, is treated explicitly.

To ensure stability at the beginning of an iteration process and at the same time obtain a high convergence speed, we have made the pseudo time step per grid point inversely proportional to the local previous solution correction $(\overline{u}^{m-1} - \overline{u}^{m-2})$. The result is a pseudo time step that automatically compensates for the local Newton linearization error. To reduce the sensitivity of the method to irregular convergence behavior, the pseudo time steps are slightly underrelaxed. The iteration process is initialized by taking the initial pseudo CFL number $CFL^0_{\text{pseu}} = (|\overline{q}^0|/\overline{d}^0 + (g\overline{d}^0)^{1/2})\Delta t^0_{\text{pseu}}/\Delta x$ equal to 1. This gives a very conservative initial pseudo time-step value, which in fact is only required for steady-state problems where the initial condition $\overline{u}^0$ is generally strongly different from the final solution $\overline{u}^*$ satisfying $\overline{L}(\overline{u}^*) = 0$.

The discretization in space of the pseudo time-step term in (20) is partially upwind to stabilize the iteration process for high-speed flow calculations. To avoid stability problems due to sudden changes at the boundaries, we use solution-correction dependent underrelaxation of the boundary conditions that vanishes upon convergence. These two measures turn out to be very effective in practice.

The convergence behavior of the present method is as described in [24]: slow convergence in the first or *searching* phase where the algorithm tries to get in the neighborhood of $\overline{u}^*$ ($CFL_{\text{pseu}} = O(1)$); fast convergence in the second or *converging* phase where the algorithm feels the attraction of $\overline{u}^*$ ($CFL_{\text{pseu}} \gg 1$). We have observed that, depending on the difficulty of the system of equations to be solved, the searching phase can be virtually absent or can take up to several hundreds of iterations. The converging phase invariably takes only about 20 to 30 iterations to reduce the convergence error down to $(\max_i |d_i^m - d_i^{m-1}| < 10^{-11}, \max_i |q_i^m/d_i^m - q_i^{m-1}/d_i^{m-1}| < 10^{-13})$, the convergence criterion that we have used in all flow calculations. This type of convergence behavior is for problems with steep-gradient solutions where the underrelaxation of ν_{art} precludes quadratic convergence. Very easy problems with smooth solutions that require virtually no artificial smoothing converge faster.

An example of a system of equations that is difficult to solve is the calculation of a complex steady-state flow through a complex geometry, modeled on

a grid consisting of 2000 grid points. The difficulty is entirely due to the fact that the initial condition (uniform discharge qW, constant water level h) is very different from the solution that is sought. In contrast, a system is easy to solve whenever a reasonable initial condition is available. This situation is encountered in unsteady calculations where there is hardly any searching phase; the initial condition is the solution of the previous physical time step and usually already very close to the solution of the next physical time step. Flow calculations inside the adaptive grid algorithm are also nearly all easy. In adaptive grid calculations we solve within each time step or steady state alternately the grid equations and the flow equations until convergence of the coupled grid-flow problem (the outer grid iteration algorithm, cf. Section 6). The initial condition of the flow calculation on an updated grid is obtained by interpolation of the solution on the previous grid. As soon as the grid starts converging, this condition is reasonably to very close to the next solution, both for unsteady and for steady-state calculations. Remark that the present iterative flow solver is relatively slow for easy systems of equations. The conservative initialization of the pseudo time step with $CFL^0_{\text{pseu}} = 1$ in combination with the applied underrelaxation of Δt_{pseu} prevent the occurrence of large pseudo time steps in the first few iterations, even in situations where that would be possible.

5 Error analysis

The error analysis consists of the following steps (cf. Figure 1):

- define the close and smooth approximation $\widehat{u}$ of the numerical solution $\overline{u}$,
- determine the continuous differential operator $\widehat{L}$ satisfied by $\widehat{u}$,
- determine the difference between $\widehat{L}$ and the continuous differential operator $\widetilde{L}$ of the easy problem,
- express this difference in terms of the effect that is has on $\widehat{u}$.

The first step is fairly straightforward. The second step is more complicated and requires the correct application of Taylor-series expansions. This is realized by anticipating on the result of the third step:

$$R(\widehat{u}) = \widehat{L}(\widehat{u}) - \widetilde{L}(\widehat{u}) \; . \tag{21}$$

Residual R indicates the numerical approximation error that is introduced in the easy system of partial differential equations by a particular numerical solution. The purpose of the fourth step is to transform residual R into the effect that it has locally on the numerical solution (the numerical modeling error).

The grid-dependent numerical modeling error is used in the right-hand side of (14) for the determination of ν_{art} (the feedback loop in Figure 1). Hence, minimizing the numerical modeling error by means of grid adaptation will not only minimize in some sense the difference between $\widehat{L}$ and $\widetilde{L}$ as far as relevant

for the accuracy of the numerical solution, but also the difference between $\widetilde{L}$ and L. Error minimization as such cannot be guaranteed, but the effect of all mechanisms responsible for the difference between the original solution u and the numerical solution $\overline{u}$ will be minimized.

The compatible scheme that we apply consists of the superposition of a discretization in space and a discretization in time (Method Of Lines). This permits to analyze the error separately in computational space and in computational time.

5.1 Error analysis in space

The first step is to define a suitable smooth and infinitely differentiable approximation $\widehat{u}$ of the piecewise linear numerical function approximation $\overline{u}$ that we have been using. In principle that is easy since the space of infinitely differentiable functions is dense in the space of piecewise polynomial functions. In practice it is slightly more complicated, because we need an algebraic description of that smooth function.

There are two reasonable ways to construct such a function: one is to connect grid-point values by a smooth function; the other is to connect cell-center values. Both possibilities are illustrated in Figure 3 by respectively the function $\widehat{u}(\gamma = 0)$ and $\widehat{u}(\gamma = 1)$. It can be seen that the closest smooth fit to $\overline{u}$ lies somewhere in between these two functions.

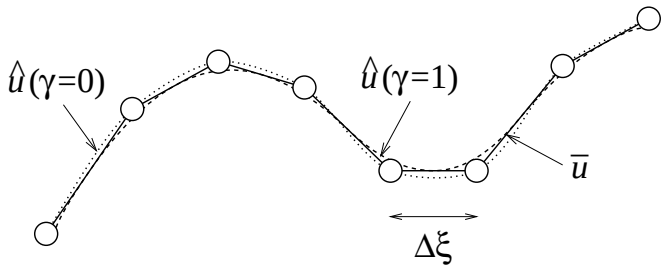

Figure 3: Smooth approximations of a piecewise linear function

This suggests to consider the one-parameter family of functions obtained by means of linear interpolation between $\widehat{u}(\gamma = 0)$ and $\widehat{u}(\gamma = 1)$, with γ the interpolation coefficient. It is not relevant to determine how these functions can be constructed from the grid-point values of $\overline{u}$. What matters is the inverse: $\overline{u}$ expressed in terms of $\widehat{u}$ and its derivatives at the grid points, since this is the information required to construct the Taylor-series expansions. Skipping the details of the construction, we will just present the result:

$$\overline{u}(\xi) = \overline{u}(\xi_{i-1} + \beta\Delta\xi)) = f_i^-(1-\beta) - \gamma g_i^-(1-\beta) + O(\Delta\xi^6) \,, \qquad (22)$$

or:

$$\overline{u}(\xi) = \overline{u}(\xi_{i-1} + \beta\Delta\xi)) = f_{i-1}^+(\beta) - \gamma g_{i-1}^+(\beta) + O(\Delta\xi^6) \,, \qquad (23)$$

with $\xi_{i-1} \le \xi \le \xi_i, i = 1, \ldots, I+1$, local coordinate $\beta = (\xi - \xi_{i-1})/\Delta\xi \in [0,1]$, and with:

$$f_i^{\pm}(\beta) = \widehat{u}_i + \beta \left(\pm \Delta\xi \widehat{u}_i^{i} + \frac{\Delta\xi^2}{2}\widehat{u}_i^{ii} \pm \frac{\Delta\xi^3}{6}\widehat{u}_i^{iii} + \frac{\Delta\xi^4}{24}\widehat{u}_i^{iv} \pm \frac{\Delta\xi^5}{120}\widehat{u}_i^{v} \right) , \quad (24)$$

$$g_i^{\pm}(\beta) = \frac{\Delta\xi^2}{8}\widehat{u}_i^{ii} \pm \beta\frac{\Delta\xi^3}{8}\widehat{u}_i^{iii} + (24\beta - 5)\frac{\Delta\xi^4}{384}\widehat{u}_i^{iv} \pm \beta\frac{\Delta\xi^5}{128}\widehat{u}_i^{v} . \quad (25)$$

The superscripts in (24) and (25) indicate the order of differentiation with respect to ξ while subscript i indicates the grid point (e.g., $\widehat{u}_i^{iii} = \partial^3\widehat{u}/\partial\xi^3\big|_{\xi=\xi_i}$). All functions in (22), (23), (24) and (25) are also a function of computational time coordinate τ, but for clarity this has not been indicated.

For $\beta = 1$ expression (24) is the well-known Taylor-series expansion up to $O(\Delta\xi^6)$ of grid-point value $u_{i\pm1}$ in terms of derivatives at $\xi = \xi_i$ when $\widehat{u}$ passes through the grid-point values ($\gamma = 0$). The expressions (22) and (23) for $\gamma = 0$ are obtained by combining this with $\overline{u}(\xi) = \overline{u}(\xi_{i-1} + \beta\Delta\xi) = (1-\beta)u_{i-1} + \beta u_i$ (cf. (13)).

The second term in the right-hand side of (22) and (23) can be viewed as a γ-dependent correction that takes care of the shift required when the smooth approximation is defined differently. From both (22) and (23) we obtain (expand the right-hand sides at $\xi = \xi_{i-\frac{1}{2}}$):

$$\overline{u}(\xi_{i-\frac{1}{2}}) = \widehat{u}(\xi_{i-\frac{1}{2}}) + (1-\gamma)\left(\frac{\Delta\xi^2}{8}\widehat{u}_{i-\frac{1}{2}}^{ii} + \frac{\Delta\xi^4}{384}\widehat{u}_{i-\frac{1}{2}}^{iv} \right) + O(\Delta\xi^6) . \quad (26)$$

This shows that $\gamma = 1$ corresponds with a smooth function fit through the cell-center values. In fact, the equation $\overline{u}(\xi_{i-\frac{1}{2}}) = \widehat{u}(\xi_{i-\frac{1}{2}})$ has been used to derive the expression for $g_i^{\pm}$. Another equation that has been used in that derivation is $g_{i-1}^{+}(\beta) = g_i^{-}(1-\beta) + O(\Delta\xi^6)$. We also have $f_{i-1}^{+}(\beta) = f_i^{-}(1-\beta) + O(\Delta\xi^6)$. The latter two equations must hold for the two expressions (22) and (23) to be equivalent, and hence continuous at the grid points. This is trivial for $\gamma = 0$ when $\widehat{u}$ passes through the grid-point values, but it is a condition to be imposed when deriving the expression for $g_i^{\pm}$. It turns out that this, together with $\overline{u}(\xi_{i-\frac{1}{2}}) = \widehat{u}(\xi_{i-\frac{1}{2}})$ for $\gamma = 1$, fixes $g_i^{\pm}$ completely.

The compatible discretization in space of the shallow-water equations has been obtained upon the integration of the easy equations with added artificial smoothing term over the finite volumes $[\xi_{i-\frac{1}{2}}, \xi_{i+\frac{1}{2}}]$, replacing each continuous function by its piecewise linear approximation (Subsection 4.1). In other words, the discretized flow equations $\overline{L}(\overline{u}) = 0$ can also be written as:

$$\overline{L}_i(\overline{u}) = \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \widetilde{L}(\overline{u})\, d\xi = 0 , \; i = 1, \ldots, I , \quad (27)$$

where $\widetilde{L}$ stands for the easy shallow-water equations transformed to computational space (eqs. (11) and (12)), while $\overline{u}$ represents this time the collection of

piecewise linear functions that have been used ($\overline{x}$, $\overline{d}$, $\overline{q}$, $\overline{h}$, $\overline{P}$ and $\overline{\nu}_{\rm art}$). See Figure 1 and the construction of the discretization in Subsection 4.1.

It is important to realize that a straightforward Taylor-series expansion of the left-hand side of (27) at $\xi = \xi_i$ does *not* produce the second-order residual. This approach can only be used if the discretization represents an approximation of the model equations at a specific point (finite difference methods, collocation-point methods). Finite volume discretizations (and finite element discretizations) however approximate a (weighed) *integral* of the model equations. A one-point Taylor-series expansion would then introduce an integral approximation error that is first order in general, and second order if the expansion is at the central quadrature point, as is the case here. This is one order lower or of the same order as the residual to be determined and hence unacceptable.

There is a simple way to avoid the problem of integration errors. The equivalence between the discretized problem and the equivalent problem implies that we have (cf. Figure 1 and (27)):

$$\overline{L}_i(\overline{u}) = \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \widetilde{L}(\overline{u})\, d\xi = \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \widehat{L}(\widehat{u})\, d\xi \ , \ i = 1, \ldots, I \ . \tag{28}$$

Since the discretization is obtained by integration over finite volumes, the equivalence becomes an identity per finite volume. Using (28), we obtain from (21):

$$\int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} R(\widehat{u})\, d\xi = \overline{L}_i(\overline{u}) - \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \widetilde{L}(\widehat{u})\, d\xi \ , \ i = 1, \ldots, I \ . \tag{29}$$

The integral approximation error that makes it impossible to determine the residual from a Taylor-series expansion of (27) at a single point does not pose a problem here. By expanding $\overline{L}_i(\overline{u})$ as well as the finite volume integrals of R and $\widetilde{L}(\widehat{u})$ at the same point (not necessarily the quadrature point), the terms that would perturb the error analysis cancel out. Moreover, residual R can be determined with an accuracy of $O(\Delta\xi^2)$ by just developing the right-hand side of (29) at the quadrature point $\xi = \xi_i$ and approximating the left-hand side by $\Delta\xi R_i$.

There is yet another snag to be avoided. Analyzing (29) would imply that we consider a problem in computational space. The result would be a residual that indicates the error in computational space, not in physical space. However, we are dealing with a physical problem in physical space and are only concerned with the numerical accuracy in physical space. On the other hand, it is perfectly acceptable to formulate the numerical modeling error pertaining to the physical problem in computational space. The computational space is a convenient means to construct and analyze discretizations, but its use should not have a negative effect on the error analysis.

The error in (29) is a subtle one and related to the coordinate transformation. The coordinate transformation used for the discretized equations is defined by $\overline{x}(\xi, \tau)$, while the one of the equivalent equations is $\widehat{x}(\xi, \tau)$. Hence, integrating

both over the *same* finite volume in computational space means that they are integrated over finite volumes of *different* size in physical space. We see here why it is essential to consider separate coordinate transformations for the different problems that we are dealing with (see the beginning of Section 4). To make sure that the equivalent equations will be equivalent with the discretized equations in *physical* space we should consider:

$$\int_{\overline{x}_{i-\frac{1}{2}}}^{\overline{x}_{i+\frac{1}{2}}} \frac{R(\widehat{u})}{\widehat{x}_\xi}\, d\widehat{x} = \int_{\overline{x}_{i-\frac{1}{2}}}^{\overline{x}_{i+\frac{1}{2}}} \frac{\widetilde{L}(\overline{u})}{\overline{x}_\xi}\, d\overline{x} - \int_{\overline{x}_{i-\frac{1}{2}}}^{\overline{x}_{i+\frac{1}{2}}} \frac{\widetilde{L}(\widehat{u})}{\widehat{x}_\xi}\, d\overline{x}\ ,\ i = 1, \ldots, I\ . \tag{30}$$

$\widetilde{L}(\overline{u})/\overline{x}_\xi = 0$ and $\widetilde{L}(\widehat{u})/\widehat{x}_\xi = 0$ can be viewed as equations formulated in physical space, cf. the equations in computational space (11) and (12) that have been obtained upon multiplying the original equations (1) and (2) by x_ξ.

Using (26) to replace the integral boundaries $\overline{x}_{i-\frac{1}{2}}$ and $\overline{x}_{i+\frac{1}{2}}$ of the second term in the right-hand side, we obtain from (30) (recall that $\widehat{x}_\xi$ and $\widehat{x}^i$ both denote $\partial\widehat{x}/\partial\xi$):

$$\begin{aligned}
&\Delta\xi \left(R(\widehat{u})|_{\xi_i} + O(\Delta\xi^2) \right) = \\
&= \int_{\overline{x}_{i-\frac{1}{2}}}^{\overline{x}_{i+\frac{1}{2}}} \frac{\widetilde{L}(\overline{u})}{\overline{x}_\xi}\, d\overline{x} - \int_{\widehat{x}_{i-\frac{1}{2}} + \frac{1-\gamma}{8}\Delta\xi^2 \widehat{x}^{ii}_{i-\frac{1}{2}} + O(\Delta\xi^4)}^{\widehat{x}_{i+\frac{1}{2}} + \frac{1-\gamma}{8}\Delta\xi^2 \widehat{x}^{ii}_{i+\frac{1}{2}} + O(\Delta\xi^4)} \frac{\widetilde{L}(\widehat{u})}{\widehat{x}_\xi}\, d\widehat{x} = \\
&= \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \widetilde{L}(\overline{u})\, d\xi - \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \widetilde{L}(\widehat{u})\, d\xi + \\
&\quad - \frac{1-\gamma}{8}\Delta\xi^3 \frac{\partial}{\partial\xi}\left(\frac{\widehat{x}^{ii}\widetilde{L}(\widehat{u})}{\widehat{x}^i} \right)\Bigg|_{\xi_i} + O(\Delta\xi^5)\ ,\ i = 1, \ldots, I\ .
\end{aligned} \tag{31}$$

The essential difference between (29) and (31) is the physical-to-computational-space correction in the right-hand side of (31). It vanishes for $\gamma = 1$ when we have $[\overline{x}_{i-\frac{1}{2}}, \overline{x}_{i+\frac{1}{2}}] = [\widehat{x}_{i-\frac{1}{2}}, \widehat{x}_{i+\frac{1}{2}}]$.

Since everything is now formulated in computational space again, the determination of R is fairly straightforward. Term by term the finite volume integrals of $\widetilde{L}(\overline{u})$ and of $\widetilde{L}(\widehat{u})$ are expanded in a Taylor series at $\xi = \xi_i$, using (22) and (23) to replace $\overline{u}$ by $\widehat{u}$. For example, using the discretization in space (16), the evaluation of the right-hand side of (31) for the time derivative $\partial(x_\xi q)/\partial\tau$ reads:

$$
\begin{aligned}
&\int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \frac{\partial(\overline{x}_\xi \overline{q})}{\partial\tau}\, d\xi - \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \frac{\partial(\widehat{x}_\xi \widehat{q})}{\partial\tau}\, d\xi - \frac{1-\gamma}{8}\Delta\xi^3 \frac{\partial}{\partial\xi}\left(\frac{\widehat{x}^{ii}}{\widehat{x}^i}\frac{\partial(\widehat{x}_\xi\widehat{q})}{\partial\tau}\right)\Big|_{\xi_i} = \\
&= \Delta\xi^3 \frac{\partial}{\partial\tau}\left(\frac{2-3\gamma}{24}\widehat{x}^i\widehat{q}^{ii} + \frac{1}{24}\widehat{x}^{ii}\widehat{q}^i + \frac{1-\gamma}{8}\widehat{x}^{iii}\widehat{q}\right)_{\xi_i} + \\
&\quad - \frac{1-\gamma}{8}\Delta\xi^3\left(\frac{\partial}{\partial\tau}\left(\widehat{x}^{ii}\widehat{q}^i + \widehat{x}^{iii}\widehat{q}\right) - \frac{\partial}{\partial\xi}\left(\widehat{x}_\xi\widehat{q}\frac{\partial}{\partial\tau}\frac{\widehat{x}^{ii}}{\widehat{x}^i}\right)\right)_{\xi_i} + O(\Delta\xi^5) = \\
&= \frac{2-3\gamma}{24}\Delta\xi^3\frac{\partial}{\partial\tau}\left(\widehat{x}^i\left(\widehat{q}^{ii} - \frac{\widehat{x}^{ii}}{\widehat{x}^i}\widehat{q}^i\right)\right)_{\xi_i} + \\
&\quad + \frac{1-\gamma}{8}\Delta\xi^3\frac{\partial}{\partial\xi}\left(\widehat{q}\left(\widehat{x}^{ii}_\tau - \frac{\widehat{x}^{ii}}{\widehat{x}^i}\widehat{x}^i_\tau\right)\right)_{\xi_i} + O(\Delta\xi^5)\,.
\end{aligned}
\tag{32}
$$

In the first term of the right-hand side of (32) one recognizes the second-order interpolation error of $\overline{q}$ in *physical* space integrated over the finite volume $[\overline{x}_{i-\frac{1}{2}}, \overline{x}_{i+\frac{1}{2}}]$ and formulated in computational space:

$$
\begin{aligned}
&\int_{\overline{x}_{i-\frac{1}{2}}}^{\overline{x}_{i+\frac{1}{2}}} (\overline{q} - \widehat{q})\, dx = \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \overline{x}_\xi \overline{q}\, d\xi - \int_{\xi_{i-\frac{1}{2}}}^{\xi_{i+\frac{1}{2}}} \widehat{x}_\xi \widehat{q}\, d\xi - \frac{1-\gamma}{8}\Delta\xi^3 \left.\frac{\partial\widehat{x}^{ii}\widehat{q}}{\partial\xi}\right|_{\xi_i} = \\
&= \Delta\xi^3\left(\frac{2-3\gamma}{24}\widehat{x}^i\widehat{q}^{ii} + \frac{1}{24}\widehat{x}^{ii}\widehat{q}^i + \frac{1-\gamma}{8}\widehat{x}^{iii}\widehat{q}\right)_{\xi_i} + \\
&\quad - \frac{1-\gamma}{8}\Delta\xi^3\left(\widehat{x}^{ii}\widehat{q}^i + \widehat{x}^{iii}\widehat{q}\right)_{\xi_i} + O(\Delta\xi^5) = \\
&= \frac{2-3\gamma}{24}\Delta\xi^3\widehat{x}^i_i\left(\widehat{q}^{ii} - \frac{\widehat{x}^{ii}}{\widehat{x}^i}\widehat{q}^i\right)_{\xi_i} + O(\Delta\xi^5)\,.
\end{aligned}
\tag{33}
$$

Using $\partial^2\widehat{q}/\partial x^2 = 1/\widehat{x}_\xi\partial(\widehat{q}_\xi/\widehat{x}_\xi)/\partial\xi = (\widehat{q}^{ii} - \widehat{x}^{ii}\widehat{q}^i/\widehat{x}^i)/(\widehat{x}^i)^2$, this can also be written as:

$$
\int_{\overline{x}_{i-\frac{1}{2}}}^{\overline{x}_{i+\frac{1}{2}}} (\overline{q} - \widehat{q})\, dx = \frac{2-3\gamma}{24}\Delta\overline{x}_i^3 \left.\frac{\partial^2\widehat{q}}{\partial x^2}\right|_{\xi_i} + O(\Delta\overline{x}_i^5)\,,
$$

with $\Delta\overline{x}_i = \overline{x}_{i+\frac{1}{2}} - \overline{x}_{i-\frac{1}{2}}$.

The space discretization of the other terms in the flow equations is analyzed in the same way. This shows for example that the second term in the right-hand side of (32) cancels against the part of the residual of the convection term $\partial(\overline{q}/\overline{d} - \overline{x}_\tau)\overline{q}/\partial\xi$ stemming from the interpolation error in physical space of $\overline{x}_\tau$. Skipping the lengthy derivation and reformulating the residual in terms of errors in the variables, we obtain for the momentum equation:

$$
\begin{aligned}
&\frac{\partial}{\partial\tau}\left[\widehat{x}_\xi\left(\widehat{q}+(\gamma'-\frac{1}{24})D_\xi(\widehat{q})\right)\right]+\\
&\quad+\frac{\partial}{\partial\xi}\left[\left(\frac{\widehat{q}+\gamma' D_\xi(\widehat{q})}{\widehat{d}+\gamma' D_\xi(\widehat{d})}-\widehat{x}_\tau\right)\left(\widehat{q}+\gamma' D_\xi(\widehat{q})\right)\right]+\\
&\quad+g\left(\widehat{d}+\gamma' D_\xi(\widehat{d})\right)\frac{\partial}{\partial\xi}\left[\widehat{h}+\gamma' D_\xi(\widehat{h})\right]+g\frac{\Delta\xi^2}{24}\left(\widehat{d}^i\widehat{h}^{ii}-\widehat{d}^{ii}\widehat{h}^i\right)=\\
&=\frac{\partial}{\partial\xi}\left[\left(\widehat{\nu}_{\rm art}+\gamma' D_\xi(\widehat{\nu}_{\rm art})\right)\times\right.\\
&\qquad\left.\times\left(\frac{1}{\widehat{x}_\xi}\frac{\partial}{\partial\xi}\left(\widehat{q}+(\gamma'-\frac{1}{12})D_\xi(\widehat{q})\right)+\frac{D_\xi(\widehat{q})}{12}\frac{\partial(1/\widehat{x}_\xi)}{\partial\xi}\right)\right]+\\
&\quad-\frac{\partial}{\partial\xi}\left[\frac{\left(\widehat{\nu}_{\rm art}+\gamma' D_\xi(\widehat{\nu}_{\rm art})\right)\left(\widehat{q}+\gamma' D_\xi(\widehat{q})\right)}{\widehat{d}+\gamma' D_\xi(\widehat{d})}\times\right.\\
&\qquad\left.\times\left(\frac{1}{\widehat{x}_\xi}\frac{\partial}{\partial\xi}\left(\widehat{d}+(\gamma'-\frac{1}{12})D_\xi(\widehat{d})\right)+\frac{D_\xi(\widehat{d})}{12}\frac{\partial(1/\widehat{x}_\xi)}{\partial\xi}\right)\right]+\\
&\quad+O(\Delta\xi^4)\,,
\end{aligned}
\tag{34}
$$

with $\gamma'=(1-\gamma)/8$ and with

$$
D_\xi=\Delta\xi^2\left(\frac{\partial^2}{\partial\xi^2}-\frac{\widehat{x}^{ii}}{\widehat{x}^i}\frac{\partial}{\partial\xi}\right) \tag{35}
$$

the differential operator for the second-order interpolation error in physical space.

Equivalent partial differential equation (34) is an extended version of the result presented in [2]. For convenience we have omitted bottom friction term $\overline{x}_\xi g(\overline{P}\overline{q}|\overline{q}|)(W\overline{d}^2C^2)$ which equivalent expression is obtained by replacing $\overline{x}_\xi$ by $\widehat{x}_\xi$ and all other variables by their equivalent smooth approximation plus average interpolation error like in the time derivative term in the left-hand side of (34).

One recognizes in (34) momentum equation (12) with the artificial viscosity term written in the form used for its discretization (17). The difference is due to discretization errors and consists, if error terms of $O(\Delta\xi^4)$ can be neglected, almost entirely of second-order interpolation errors in physical space (the D_ξ-terms). With a small exception in the viscosity term, these errors perturb the local value of variables just like piecewise linear interpolations would do. For example, the error in the time derivative equals the second-order interpolation error averaged over the finite volumes (cf. (33)) and vanishes for $\gamma'-1/24=0\rightarrow\gamma=2/3$. The error in the convection term is determined by the second-order interpolation error at the cell centers $\xi_{i-\frac{1}{2}}$ which is where the fluxes are evaluated, and vanishes for $\gamma'=0\rightarrow\gamma=1$. It seems that the optimal value of γ, defining the best possible smooth fit in space, is somewhere in between.

This result demonstrates the compatibility between the discretization of the flow equations and the applied discrete function approximations. If the compatible scheme is used and sufficient smoothing is applied, the leading part of the residual (21), i.e., the difference between equivalent equation (34) and easy equation (12), can be reformulated directly in terms of local errors in the numerical solution that are of the same form and size as the leading interpolation errors. There is no need for solving a local dual problem.

Only the equivalent form of the hydrostatic pressure term (the third line of (34)) contains an error term that does not depend on interpolation errors in physical space. This is because the discretization of this *non*-conservative term involves the *weighed* integration (by $\overline{d}$) of $\partial\overline{h}/\partial\xi$ over the volume $[\xi_{i-\frac{1}{2}}, \xi_{i+\frac{1}{2}}]$, while this term is only a second-order accurate approximation of $\partial h/\partial\xi$ at the volume boundaries. Using (3), we can write that term as:

$$g\frac{\Delta\xi^2}{24}\left(\widehat{d}^i\widehat{h}^{ii} - \widehat{d}^{ii}\widehat{h}^i\right) = g\frac{\Delta\xi^2}{24}\left(\widehat{d}^i z_b^{ii} - \widehat{d}^{ii} z_b^i\right) . \tag{36}$$

The effect of this term is small for a nearly horizontal bottom, but can become quite large if the bottom is non-smooth. On the other hand, since z_b is usually given as a piecewise linear function in space we have $z_b(x) = Cx$ almost everywhere, with $|C| \ll 1$ because of the assumptions underlying the shallow-water model (Section 3). Hence $\Delta\xi^2|\widehat{d}^i z_b^{ii} - \widehat{d}^{ii} z_b^i| \ll |\widehat{x}^i D_\xi(\widehat{d})|$, except near the locations where the value of z_b is specified. Some form of geometry smoothing (not included yet) is required to ensure that the effect of (36) is negligible everywhere. See also Section 4.

5.2 Error analysis in time

The analysis of the error in time is similar to the error analysis in space, but less complex. The accuracy of the scheme is lower (first order in time as opposed to second order in space), only two time levels are used (as opposed to the three-point stencil in space), the coordinate transformation is considered to be linear in time (9), and there are no second derivatives in time. It is fairly easy to derive that the residual in time, assuming components of $O(\Delta\tau^2)$ and higher can be neglected, can be rewritten in expressions of the form:

$$\frac{1}{2}D_\tau(\widehat{d}) = \frac{1}{2}\Delta\tau\frac{\partial\widehat{d}}{\partial\tau} . \tag{37}$$

This is the first-order interpolation error in *computational* time that can be reduced by decreasing the size of the time step but also by aligning the grid with the solution. The latter property is the result of using a higher-order piecewise linear approximation in time for the coordinate mapping (i.e. (10)) in combination with the backward piecewise constant discretization (13) for the other variables.

5.3 Error in discretized shallow-water equations

Combining the results of the previous two Subsections, we conclude that the local effect of the discretization error on the numerical solution of the shallow-water equations (the numerical modeling error) can reasonably well be approximated by the sum of the second-order interpolation error in physical space and the first-order interpolation error in computational time:

$$Err_d = Err_d^{\text{space}} + Err_d^{\text{time}} \approx |D_\xi(\widehat{d})| + c_\gamma |D_\tau(\widehat{d})| \approx |D_\xi(\overline{d})| + c_\gamma |D_\tau(\overline{d})| \ , \quad (38)$$

$$Err_q = Err_q^{\text{space}} + Err_q^{\text{time}} \approx |D_\xi(\widehat{q})| + c_\gamma |D_\tau(\widehat{q})| \approx |D_\xi(\overline{q})| + c_\gamma |D_\tau(\overline{q})| \ , \quad (39)$$

with D_ξ and D_τ as in (35) and (37). Since all functions are smooth (they should be, by construction), the derivatives of $\widehat{d}$, $\widehat{q}$ and $\widehat{x}$ in D_ξ and D_τ can be approximated very well by the derivatives of $\overline{d}$, $\overline{q}$ and $\overline{x}$ using simple finite differences. We see that, although the smooth fits of the piecewise polynomial numerical approximations are the basis of the error analysis, these smooth functions are actually not required. It suffices to know that they exist.

Scaling of the error expressions (38) and (39) with a constant coefficient is irrelevant. However, the relative weighing between the interpolation error in space and the one in time is important. It is determined by parameter c_γ that, in view of the coefficients of the errors in space in (34) and the coefficient of the error in time (37), should have a value of about $(1/2)/(1/12) = 6$.

The reliability of error approximations (38) and (39) depends to a large extent on the smoothness of the bottom, cf. the discussion above definition (15) and below equation (36). It may therefore be useful to consider instead of (38):

$$Err_h = Err_h^{\text{space}} + Err_h^{\text{time}} \approx |D_\xi(\widehat{h})| + c_\gamma |D_\tau(\widehat{h})| \approx |D_\xi(\overline{h})| + c_\gamma |D_\tau(\overline{h})| \ . \quad (40)$$

For the determination of the artificial viscosity coefficient ν_{art} and the adaptation of the grid, the error in water depth d or water level h and in depth-integrated velocity q should be combined in some way to a single error expression. A convenient option is to base that combination on an energy measure like the energy head. Since the energy head itself may be (nearly) constant (cf. expression (6)) we will consider the sum of the error in the potential part and in the kinetic part of the flow energy. From a physical point of view this seems to be a fairly meaningful choice. So for the grid adaptation we consider:

$$\begin{aligned} Err = Err^{\text{space}} + Err^{\text{time}} \approx & |D_\xi(\overline{d})| + \left| \frac{\overline{q} D_\xi(\overline{q})}{g\overline{d}^2} - \frac{\overline{q}^2 D_\xi(\overline{d})}{g\overline{d}^3} \right| \\ & + c_\gamma \left(|D_\tau(\overline{d})| + \left| \frac{\overline{q} D_\tau(\overline{q})}{g\overline{d}^2} - \frac{\overline{q}^2 D_\tau(\overline{d})}{g\overline{d}^3} \right| \right) \ , \end{aligned} \quad (41)$$

which expression has been obtained by linearizing $\frac{1}{2}\overline{q}^2/(g\overline{d}^2) - \frac{1}{2}\widehat{q}^2/(g\widehat{d}^2)$, the interpolation error in the kinetic part $\frac{1}{2}u^2/g$ of the energy head, with respect to $D_\xi(\overline{q})$, $D_\xi(\overline{d})$, $D_\tau(\overline{q})$ and $D_\tau(\overline{d})$.

The dimension of Err is [m]. Therefore, expression (41) cannot be used for Err_ν, the error in the right-hand side of artificial viscosity equation (14). Instead we use:

$$Err_\nu = \Delta\xi\overline{x}_\xi \left(\frac{1}{2}\sqrt{\frac{g}{\overline{d}}}\,|D_\xi(\overline{h})| + \sqrt{\frac{1}{2}}\left|\frac{D_\xi(\overline{q})}{\overline{d}} - \frac{\overline{q}D_\xi(\overline{d})}{\overline{d}^2}\right| \right) , \tag{42}$$

obtained by linearizing the interpolation error in the squareroot of the potential part and the kinetic part of the energy head, multiplied by $\sqrt{g}$ and by grid size $\Delta\xi\overline{x}_\xi$. Only the interpolation error in space is considered; it is not necessary to include the error in time in the error feedback mechanism because the dissipative backward Euler scheme that we use provides already enough smoothing in time. It is easily verified that (42) is as described below equation (14) and satisfies the artificial dissipation requirements mentioned in Section 2.

Notice that the interpolation error in the water *depth* is used in (41), while in (42) we look at the interpolation error in the water *level*. Ideally, these two should be nearly equivalent; at present they are not because of the absence of geometry smoothing. Since a smooth water surface implies a smooth hydrostatic pressure term in (2) and hence a smooth flow, the use of $D_\xi(\overline{h})$ is prefered in (42). This avoids that unnecessarily large values of ν_{art} occur in regions with large variations in water depth but with small flow velocities. However, numerical experiments have shown that it is best to use $D_\xi(\overline{d})$ and $D_\tau(\overline{d})$ in (41), probably because in this way the algorithm 'feels' to some extent the non-smoothness of the bottom. All this is rather a matter of compromising and certainly not ideal, which is confirmed by the results that we will present.

6 Error-minimizing grid adaptation

In the previous Section we have derived expression (41): a physically meaningful approximation of the numerical modeling error. Err is a function of the numerical solution that is sought, as well as of the piecewise linear coordinate transformation $\overline{x}(\xi,\tau)$ defined by the grid point coordinates x_i^n (cf. 10). The grid points form a set of degrees of freedom that can be chosen in any convenient way. Here, we choose to determine them by considering the optimization problem:

$$\text{solve: } \min_{\overline{x}(\xi,\tau)} \| \, Err \, \|_1 \, . \tag{43}$$

Numerical modeling error Err is measured in L_1-norm for reasons explained in Section 1. Putting (41) and (43) together, we obtain:

$$\begin{aligned} \text{solve: } \min_{\overline{x}(\xi,\tau)} \int_{t^0}^{t^N} \int_{x_{\text{left}}}^{x_{\text{right}}} \left(Err^{\text{space}} + Err^{\text{time}}\right) dx\, dt = \\ = \text{solve: } \min_{\overline{x}(\xi,\tau)} \int_{t^0}^{t^N} \int_{\xi_{\frac{1}{2}}}^{\xi_{I+\frac{1}{2}}} \overline{x}_\xi \left(Err^{\text{space}} + Err^{\text{time}}\right) d\xi\, dt \, , \end{aligned} \tag{44}$$

with x_{left} and x_{right} the coordinates of the left and right boundary in physical space, and $\xi_{\frac{1}{2}}$ and $\xi_{I+\frac{1}{2}}$ the coordinates of these boundaries in computational space (cf. Figure 2 and (10)).

Solving the global optimization problem (44) as such would be prohibitively complicated and expensive. Following the usual approach in moving adaptive grid methods, we therefore determine the grid point coordinates x_i^n time level by time level, keeping the grid points x_i^{n-1} fixed at the previous time level. So instead of (44) we consider the sequence of optimization problems:

$$\text{solve: } \min_{\overline{x}^n(\xi)} \int_{t^{n-1}}^{t^n} \int_{\xi_{\frac{1}{2}}}^{\xi_{I+\frac{1}{2}}} \overline{x}_\xi \left(Err^{\text{space}} + Err^{\text{time}} \right) \, d\xi \, dt \ , \ n = 1, \ldots, N \ . \quad (45)$$

This way of grid optimization may however lead to very poor grid point redistributions. It forces the grid points to move in a direction that may be totally different from the direction of propagation of steep gradients, in order to get the points concentrated at the right locations. For example, after a sudden change of the condition imposed at a boundary, grid points have to move toward that boundary which may seriously conflict with other requirements. Even the 'optimal' grid may then yield a poor error reduction. Small time steps should be taken to compensate for the loss in accuracy when the grid lines cannot be aligned anymore with steep gradients in the space-time domain. This strongly reduces the efficiency and hence the advantages of moving grid adaptation.

The above problem is caused by the fact that (45) can be a very poor approximation of (44). By minimizing the error only in subsequent time steps we do not minimize the error globally in time. This is a consequence of keeping the grid points fixed once they have been determined, which forces the grid to be continuous in time. As the results will illustrate, this drawback is amplified by the presence of a variable bottom geometry.

An elegant solution would be to optimize the grid per time step at both the previous and the next time level, and to allow the grid to be discontinuous in time. This would strongly augment the possibilities for error minimization. Such a procedure is quite feasible in the present approach where we use a two-level method for the discretization in time; the coordinate transformation per time step would then be truly independent (cf. (9)), i.e., also the number of grid points per time step could be optimized. Moreover, solving (45), minimizing the error per time step with respect to $\overline{x}^{n-1}(\xi)$ as well, would become virtually equivalent with solving (44).

From (45) we obtain the algebraic system of equations that defines the optimal value of the x_i^n. Because Err^{space} and Err^{time} are grid dependent, the equations are nonlinear and have to be solved iteratively. Each iteration would involve a complicated interpolation procedure to handle the changing grid, which is not convenient. We have therefore opted for a different approach, determining per time step the optimal coordinate transformation $\overline{x}(\xi', \tau')$ given the coordinate transformation $\overline{x}(\xi, \tau)$. By solving the flow equations on the grid defined by $\overline{x}(\xi, \tau)$ we obtain the flow solution and hence Err^{space} and Err^{time} on that grid. From this we calculate $\overline{x}(\xi', \tau')$ which then defines the next approximation

of optimal coordinate transformation $\overline{x}(\xi,\tau)$ on which we solve again the flow equations, etcetera. This process, that defines the *outer grid iteration loop*, is repeated until convergence, i.e., until $\overline{x}(\xi',\tau')$ and $\overline{x}(\xi,\tau)$ are virtually identical. That is, in principle, because the present algorithm is not yet able to get the grid fully converged under all circumstances (see the next Section). Each outer grid iteration we solve the flow equations given the grid defined by $\overline{x}(\xi,\tau)$, and solve the grid equations to obtain $\overline{x}(\xi',\tau')$ given the flow solution. Both are iterative solution procedures; the former has been described in Section 4.2, while the latter is the *inner grid iteration loop* described below.

For clarity, we will explain the grid optimization procedure for a single unknown a. Approximating the integral in time by means of a one-point quadrature rule, the optimization problem to be solved per time step is:

$$\min_{\overline{x}^n(\xi')} \Delta t \int_{\xi'_{\frac{1}{2}}}^{\xi'_{I+\frac{1}{2}}} \overline{x}_{\xi'}^{n-\frac{1}{2}} \left(\Delta\xi'^2 |\overline{a}_{\xi'\xi'} - \frac{\overline{x}_{\xi'\xi'}}{\overline{x}_{\xi'}} \overline{a}_{\xi'}| + c_\gamma \Delta\tau' |\overline{a}_{\tau'}| \right)^{n-\frac{1}{2}} d\xi' \,, \qquad (46)$$

where we have used an expression like (40) for a, substituting (35) and (37). For the optimization of the grid in shallow-water applications the more complicated error expression (41) is used, but the principle remains the same.

Optimization problem (46) has been formulated in the computational space (ξ',τ') corresponding with optimal coordinate transformation $\overline{x}(\xi',\tau')$. To be able to solve it, we transform it to the current computational space (ξ,τ), introducing $\xi'(\xi,\tau)$ and $\tau'=\tau$, the transformation from current to optimal computational space. It is most convenient to consider $\xi'(\xi,\tau)$ and not $\overline{x}(\xi',\tau')$ as the unknown function to be determined, since $\xi'(\xi,\tau)$ is defined (and hence can be approximated) on the current computational grid. The coordinates of the optimal computational space corresponding with the grid points in the current computational space will be denoted by $\xi_i'^{,n} = \xi'(\xi_i,\tau^n)$. Notice that $\xi_i'^{,n} - \xi_{i-1}'^{,n}$ is *not* a grid size and hence in general not equal to $\Delta\xi'$, the size of the uniform grid in the optimal computational space.

At this point we can take the size of the grid in computational space (any computational space) equal to any convenient value. We will use $\Delta\xi = \Delta\xi' = 1$ (recall that we have $\Delta\tau = \Delta\tau' = \Delta t$ because of (9)). The position of the boundaries of a computational domain are fixed, $\xi'_{\frac{1}{2}} = \xi_{\frac{1}{2}} = \frac{1}{2}$ and $\xi'_{I+\frac{1}{2}} = \xi_{I+\frac{1}{2}} = I+\frac{1}{2}$, and so optimization problem (46) formulated in (ξ,τ) becomes:

$$\text{solve: } \min_{\xi'^{,n}(\xi)} \Delta t \int_{\xi_{\frac{1}{2}}}^{\xi_{I+\frac{1}{2}}} \overline{x}_{\xi}^{n-\frac{1}{2}} \left(\frac{|D_\xi(\overline{a})|}{(\xi'_\xi)^2} + c_\gamma \Delta\tau |\overline{a}_\tau - \frac{\xi'_\tau}{\xi'_\xi} \overline{a}_\xi| \right)^{n-\frac{1}{2}} d\xi \,. \qquad (47)$$

In order to be able to solve this problem we assume that $\overline{a}$ is a function of the physical coordinates only, and independent of the grid size. This is obviously not true in general, especially in regions of steep gradients, but a reasonable approximation for small grid perturbations (in particular perturbations around the optimal grid) because of the smoothness of the solution. Since we have fixed

momentarily coordinate transformation $x(\xi,\tau)$, this implies that $\overline{x}$ and $\overline{a}$ in (47) are both considered to be independent of ξ'.

Under this assumption it is straightforward to solve (47). The integral is approximated by a sum over the grid cells using straightforward discretizations, and differentiated with respect to the $\xi_i'^{,n}, i = 1,\ldots,I$. Equating the result to zero, the system of equations is obtained that defines the optimal grid:

$$\begin{aligned}
&\frac{8\Delta x_{i+\frac{1}{2}}^{n-\frac{1}{2}}}{(\xi_{i+1}'^{,n}-\xi_i'^{,n}+1)^3}e_{i+\frac{1}{2}}^{\xi,n-\frac{1}{2}} - \frac{8\Delta x_{i-\frac{1}{2}}^{n-\frac{1}{2}}}{(\xi_i'^{,n}-\xi_{i-1}'^{,n}+1)^3}e_{i-\frac{1}{2}}^{\xi,n-\frac{1}{2}} = \\
&= c_\gamma\Delta\tau\left(\frac{2\Delta x_{i+\frac{1}{2}}^{n-\frac{1}{2}}(\xi_{i+1}'^{,n}-i)}{(\xi_{i+1}'^{,n}-\xi_i'^{,n}+1)^2}e_{i+\frac{1}{2}}^{\tau,n-\frac{1}{2}} - \frac{2\Delta x_{i-\frac{1}{2}}^{n-\frac{1}{2}}(\xi_{i-1}'^{,n}-i)}{(\xi_i'^{,n}-\xi_{i-1}'^{,n}+1)^2}e_{i-\frac{1}{2}}^{\tau,n-\frac{1}{2}}\right), \\
&i = 1,\ldots,I\,,
\end{aligned} \tag{48}$$

with:

$$\begin{aligned}
e^\xi &= \text{smoothed } \left|\overline{a}_{\xi\xi} - \overline{a}_\xi\overline{x}_{\xi\xi}/\overline{x}_\xi\right|, \\
e^\tau &= \text{smoothed } \overline{a}_\xi\tanh\left(\frac{\Delta\tau(\overline{a}_\tau - \overline{a}_\xi\xi_\tau'/\xi_\xi')}{c_\tau|\overline{a}|}\right).
\end{aligned} \tag{49}$$

The latter is an approximation of $e^\tau = \text{sign}(\overline{a}_\tau - \overline{a}_\xi\xi_\tau'/\xi_\xi')$ that, not surprisingly, turns out to lead to serious stability problems. The argument of the tanh function is scaled with $|\overline{a}|$ to make it non-dimensional. For the shallow-water equations we scale with the energy-head-like expression $\overline{d} + \frac{1}{2}\overline{q}^2/(g\overline{d}^2)$ and use $c_\tau = 10$ for the scaling parameter. The smoothing of e^ξ and e^τ is the same as the one applied for ν_{art}, i.e., like equation (14) which discretization is given in (18), using the same value of constant smoothing coefficient α.

Equation (48) with boundary conditions $\frac{1}{2}(\xi_0'^{,n} + \xi_1'^{,n}) = \frac{1}{2}$ and $\frac{1}{2}(\xi_I'^{,n} + \xi_{I+1}'^{,n}) = I + \frac{1}{2}$ is solved iteratively by means of a Newton-type method, evaluating e^τ explicitly using a very strong underrelaxation. The diagonal of the implicit part of the linearized system of equations per iteration is increased for additional stability. The approach is similar to that used for the flow equations (see Subsection 4.2). Although we obtain convergence down to 10^{-10}, the convergence is usually slow and certainly needs to be improved.

Once equation (48) has been solved, we use the grid point values $\xi_i'^{,n}$ to define the piecewise linear function $\overline{\xi}'^{,n}(\xi)$. The coordinates of the next approximation of the optimal grid are determined by $x_i'^{,n} = \overline{x}^n(\xi)$, with ξ the solution of $\overline{\xi}'^{,n}(\xi) = i, i = 1,\ldots,I$. In practice, we do not use $\overline{x}^n(\xi)$ here but a smooth approximation based on a monotonicity-preserving cubic Hermite interpolation [6]. This leads to smoother grid updates in the outer grid iteration loop without changing the final, fully converged grid. The coordinates of the virtual grid points are obtained from the grid boundary conditions $\frac{1}{2}(x_0'^{,n} + x_1'^{,n}) = x_{\text{left}}$ and $\frac{1}{2}(x_I'^{,n} + x_{I+1}'^{,n}) = x_{\text{right}}$.

Once the outer grid iteration loop has converged we have $\xi_i'^{,n} \to \xi_i = i$ in which case the equations (48) reduce to:

$$\begin{aligned}&\Delta x_{i+\frac{1}{2}}^{n-\frac{1}{2}} e_{i+\frac{1}{2}}^{\xi,n-\frac{1}{2}} - \Delta x_{i-\frac{1}{2}}^{n-\frac{1}{2}} e_{i-\frac{1}{2}}^{\xi,n-\frac{1}{2}} = \\ &= \frac{1}{2} c_\gamma \Delta\tau \left(\Delta x_{i+\frac{1}{2}}^{n-\frac{1}{2}} e_{i+\frac{1}{2}}^{\tau,n-\frac{1}{2}} + \Delta x_{i-\frac{1}{2}}^{n-\frac{1}{2}} e_{i-\frac{1}{2}}^{\tau,n-\frac{1}{2}} \right) \ , \ i = 1, \ldots, I \ ,\end{aligned} \tag{50}$$

or, in the equivalent continuous form:

$$\frac{\partial}{\partial \xi} \left(e^\xi \frac{\partial x}{\partial \xi} \right) = c_\gamma \Delta\tau e^\tau \frac{\partial x}{\partial \xi} \ . \tag{51}$$

In the left-hand side one recognizes an equidistribution principle based on the numerical modeling error in space. It is corrected with the term in the right-hand side that takes account of the effect that the movement of the grid has on the numerical modeling error in time.

For steady-state problems the grid optimization boils down to the equidistribution of the second-order interpolation error $|\overline{a}_{\xi\xi} - \overline{a}_\xi \overline{x}_{\xi\xi} / \overline{x}_\xi| = |(\overline{x}_\xi)^2 \overline{a}_{xx}|$. The equations (48) solved inside the outer grid iteration loop simplify in that case to:

$$\frac{\Delta x_{i+\frac{1}{2}}^n}{(\xi_{i+1}'^{,n} - \xi_i'^{,n})^3} e_{i+\frac{1}{2}}^{\xi,n} - \frac{\Delta x_{i-\frac{1}{2}}^n}{(\xi_i'^{,n} - \xi_{i-1}'^{,n})^3} e_{i-\frac{1}{2}}^{\xi,n} = 0 \ , \ i = 1, \ldots, I \ , \tag{52}$$

with n some steady-state level.

7 Results

7.1 Steady-state application

We consider a steady-state shallow-water problem for which an exact solution is available. As explained in Section 3, exact steady-state solutions of (1) and (2) can be constructed easily if the dissipation terms (bottom friction and artificial viscosity) are set to zero. This has been realized by setting $WC^2 = 10^{18}$. Artificial viscosity coefficient ν_{art} is set to zero to obtain the exact solution but plays of course an important role in the computations.

Figure 4 shows the geometry that we have considered, the exact solution of the water level, and the numerical solution of water level, velocity, Froude number and energy head on a uniform grid consisting of 100 finite volumes. It is a fully converged solution (see Subsection 4.2) and has been obtained by setting $\alpha = 3$ and $c_\nu = 10$ in (14). With these parameter values, steep variations in ν_{art} and the numerical solution are spread over about 5 to 6 grid cells. It can be shown by means of a Fourier-mode accuracy analysis of the compatible scheme that this is sufficient and approximately required to ensure that higher-order error terms are indeed negligible. Notice however that the geometry is *not* smooth.

The boundary conditions that we have applied are $q = 19.8656\text{m}^2/\text{sec}$ at the left inflow boundary ($\rightarrow$ water level at left boundary is $h = 12.0000\text{m}$) and $h = 9.0000\text{m}$ at the right outflow boundary ($\rightarrow$ position of the hydraulic jump is $x = 439.2647\text{m}$). Because of the critical flow condition on top of the bar, the flow upstream of the bar (and hence the water level at the inflow boundary) is independent of the flow downstream of the bar and of the water level imposed at the outflow boundary.

Small dashes in Figure 4 indicate the level Fr = 1. It can be observed that the position of the hydraulic jump is predicted within a fraction of the grid size. This confirms that, although smoothing does affect the solution locally, global details like the position of the jump are modeled very accurately (cf. the theoretical considerations in Section 3).

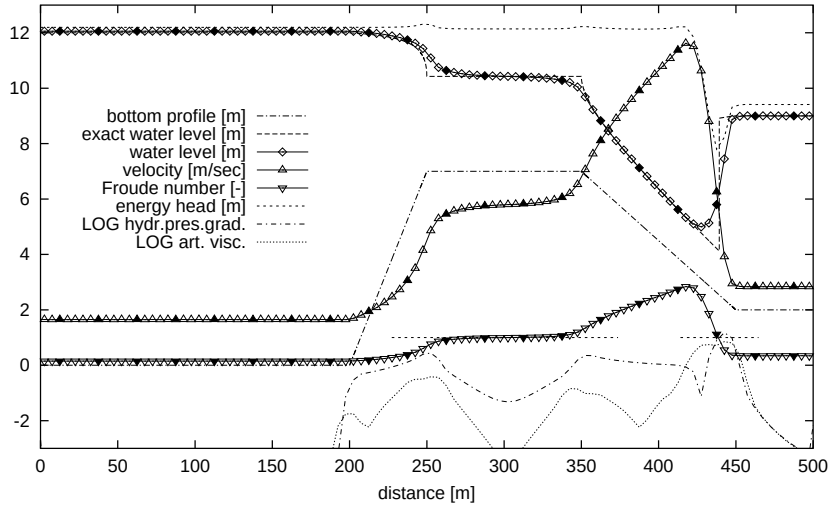

Figure 4: Shallow-water solution on uniform grid, 100 finite volumes

Since the bottom friction has been set to zero and a physical viscosity model has not been included yet, the size of the artificial viscosity term cannot be compared with a physical dissipation term. Instead, we compare it in Figure 4 with the hydrostatic pressure term. The comparison shows, not surprisingly, that the artificial viscosity is fairly large in the neighborhood of the edges of the bar and equally important as the hydrostatic pressure gradient at the location of the hydraulic jump. The dominant character of the artificial viscosity around the hydraulic jump is explained by the fact that near discontinuities this term must balance the other terms in momentum equation (2). The fairly large value near the edges of the bar is mainly due to the lack of geometry smoothness.

Less artificial viscosity would have been required near the edges if geometry smoothing would have been applied. At first sight it may seem a matter of taste to smooth the solution near the edges indirectly by means of data smoothing rather than directly by means of the artificial viscosity mechanism, but this is

not true. This is because of the way the numerical solution method attempts to approximate the infinitely steep gradient at the top two edges (see the exact water level in Figure 4). The artificial viscosity takes care of these difficult details, but without additional geometry smoothing this mechanism is unable to provide enough smoothing, especially on very fine grid when the details are felt strongly. The net result is that on very fine grids small wiggles tend to become important, thereby violating the assumption underlying the whole method that higher-order errors should be negligible. See also the adaptive grid results below.

The effect of the artificial viscosity on the solution is evidenced by the difference between the exact water level and its numerical approximation (see Figure 4). In practical applications an exact solution would not be available. A comparison between the artificial viscosity and other modeling terms is however always possible. This information becomes especially useful once bottom friction and a turbulent viscosity model will be present, since it indicates immediately if deviations from measurement data are to be attributed to physical or to numerical modeling errors. See also Section 2.

The energy head in Figure 4 downstream and upstream of the hydraulic jump is nearly constant, in agreement with the exact solution (see Section 3). Also the energy undershoot of a viscous hydraulic jump is predicted by the model. Not shown is the mass conservation 'error'. Because continuity equation (1) is discretized over the finite volumes $[\xi_{i-\frac{1}{2}}, \xi_{i+\frac{1}{2}}]$, the mass flow at the cell centers is constant within $O(10^{-12})$ which is the remaining convergence error. Perfect mass conservation at the discrete level does not prevent however a difference of 0.0352% between the value of q at the even-numbered grid points and the value of q at the odd-numbered grid points. This small mass flow wiggle is mainly due to the non-smoothness of the geometry.

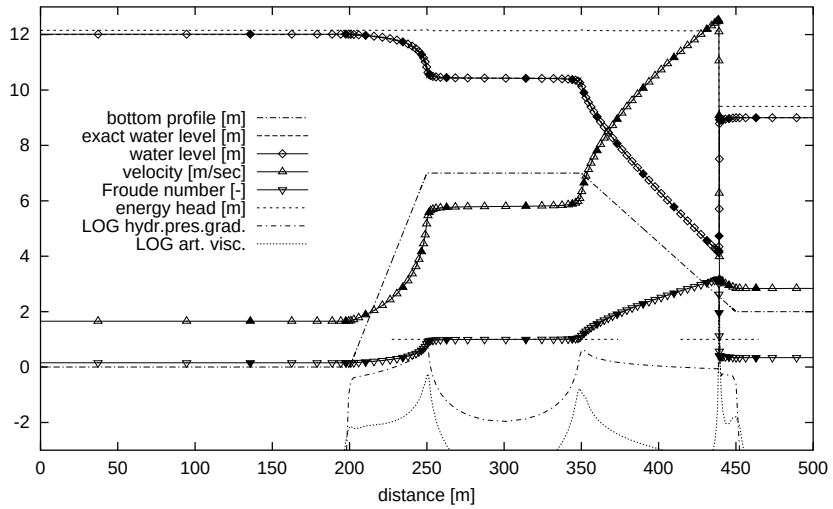

Figure 5: Shallow-water solution on adapted grid, 100 finite volumes

Solving steady-state grid adaptation equation (52) and the flow equations alternately in 50 outer grid iterations gives the fully converged adaptive grid result shown in Figure 5. The outer grid iteration loop starts with a maximum relative grid correction $\max_i |\xi'_i - \xi_i| = \max_i |\xi'_i - i|$ of 23.5 in the first iteration, and reduces it to 4.72×10^{-6} in the last iteration. With such small grid corrections there is of course no gain in numerical accuracy anymore. Comparing the exact solution and numerical solution in L_1 norm, it appears that the grid can be considered fully converged when $\max_i |\xi'_i - i| < 0.1$. This grid convergence criterion is reached after 13 outer grid iterations. The numerical solution error is then within 1% of its value on the fully converged grid.

A comparison between Figure 4 and Figure 5 shows clearly the significant accuracy improvement. The energy head upstream and downstream of the hydraulic jump is virtually constant, there is hardly any difference visible between the exact solution of the water level and its numerical approximation, and the level of the artificial viscosity is very low except at the locations where the solution is not smooth. However, the mass conservation error is 0.0156% which is barely a factor 2 better than on the uniform grid.

A detail of the adaptive grid solution is shown in Figure 6 together with the uniform grid solution. It can be observed that the structure of the numerical approximation of the hydraulic jump on the adapted grid is identical to the one shown in Figure 4. The jump is again spread over about 5 to 6 grid cells while the levels of the undershoot of the energy head are the same, indicating that the artificial viscosity mechanism has been dimensioned correctly.

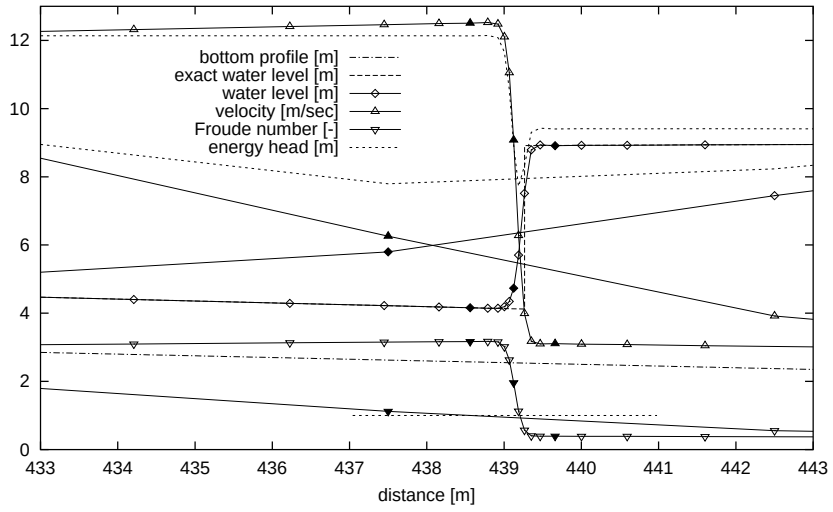

Figure 6: Comparison between uniform grid result and adaptive grid result at position of hydraulic jump, 100 finite volumes

Parameter value $\alpha = 3$ has also been used for the smoothing of the error Err^{space} (see (41)) that has been minimized. With this value, the grid stretching

is limited to 78.9% or $|\ln(\Delta x_{i+\frac{1}{2}}/\Delta x_{i-\frac{1}{2}})| \leq 0.582$. This result is obtained from discretization (18) of smoothing equation (14) that limits the rate of decay of the smoothed interpolation error to a factor $(\frac{3}{4} + 2\alpha + 2\sqrt{\frac{1}{8} + \alpha})/(2\alpha - \frac{1}{4})$ per grid cell. Substituting $\alpha = 3$ in this factor gives 1.789 ($\ln 1.789 = 0.582$), which is also the maximum grid stretching since the grid size is inversely proportional to the smoothed error (the equidistribution principle, equation (52)).

The grid size and grid stretching as a function of computational space coordinate ξ are shown in Figure 7, with the grid size non-dimensionalized by the size of the uniform grid $\Delta x_{\text{unif}} = 5\text{m}$. One recognizes the moderate grid refinement near the four edges of the bar (a stronger refinement near the top two edges) and the large refinement near the hydraulic jump.

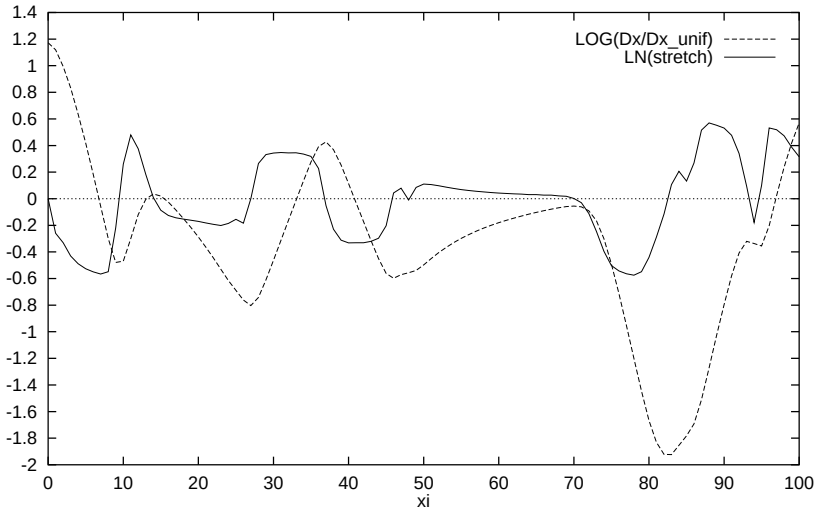

Figure 7: Computed optimal coordinate transformation for steady-state shallow-water calculation, 100 finite volumes

The maximum stretching is indeed reached at a number of places. However, the maximum stretching is not reached at the boundaries where we would have expected it. The solution near the boundaries is uniform, the interpolation error becomes zero, and so the decay of the smoothed interpolation error and hence the grid stretching should have been maximal. They are not, which means that the interpolation error does not become zero near the boundaries. This can only be due to a small but noticeable wiggle, indicating the presence of non-negligible higher-order errors that are most likely caused by the non-smooth edges of the geometry. The same wiggle is also responsible for the comparatively large mass conservation error.

Table 1 summarizes the results of a number of adaptive grid experiments varying the number of finite volumes. We used the somewhat severe grid convergence criterion $\max_i |\xi'_i - i| < 0.1$. The number of outer grid iterations

required to reach that criterion is given in the second column of the table. Convergence was not obtained for 280 and 400 volumes; the grid convergence error did not get lower than $O(1)$ in these calculations. We also needed to apply underrelaxation for the grid updates in the outer grid iteration loop when more than 100 volumes were used in order to stabilize the solution algorithm. These convergence problems are clearly related to the small but nevertheless important wiggles near the boundaries, as is evidenced by the maximum grid size $\Delta x_{\max}$. One would expect $\Delta x_{\max}$ to be roughly independent of the number of finite volumes, in accordance with the fact that the grid in the inlet section near the left boundary can have maximum stretching because of the very smooth solution. The solution is however less smooth than expected and so quite a number of grid points are drawn to the inlet section. This explains the large reduction of $\Delta x_{\max}$, the moderate gain in accuracy and the convergence problems when the number of volumes is larger than 100.

# vols	# iters	$\Delta x_{\min}$	$\Delta x_{\max}$	$\|\overline{h} - h\|_1$	order	CPU (sec.)
25	8	5.79E0	109.4	2.59E-1		0.38
35	9	2.22E0	100.9	1.81E-1	1.06	0.44
50	11	7.29E-1	92.7	9.61E-2	1.78	0.77
70	11	2.32E-1	78.4	4.07E-2	2.55	1.10
100	13	5.96E-2	74.4	1.34E-2	3.31	1.48
140	25	1.81E-2	29.7	5.15E-3	2.68	3.90
200	30	6.71E-3	11.3	2.27E-3	2.30	7.09
280	50	3.44E-3	7.3	1.01E-3	2.40	24.22
400	50	2.49E-3	3.8	5.43E-4	1.74	40.15

Table 1: Convergence behavior of adaptive grid shallow-water calculations

To confirm the hypothesis that these problems are essentially caused by the non-smooth geometry, we did the same series of test with a geometry where the edges were rounded over a length of 10 meter. This is however only a partial solution to the problem since pointwise geometry approximation (15) is not able to 'see' the curved shape of the smoothed edges in between the grid points. As a consequence, each time the grid is adapted and the points move to a different position, the flow solver sees a slightly different geometry and hence calculates a slightly different solution. This is obviously not favorable to the grid convergence process and explains why the results were only marginally better.

A series of tests with $\alpha = 6$, in an attempt to compensate for the non-smooth geometry by increasing the artificial viscosity smoothing, was more successful. This time we also obtained grid convergence when 280 volumes were used, and lower errors for 200 finite volumes and more, despite the fact that the grid was less refined near the hydraulic jump (the location of minimum grid size $\Delta x_{\min}$). From this we conclude that the adaptive grid modeling of the discontinuous jump is not posing a problem. Since the hydraulic jump is at a position where the geometry is smooth, this is consistent with our interpretation. The solution error was however larger when less than 200 volumes were used, because with

$\alpha = 6$ the grid stretching is limited to 51%.

The last column in Table 1 gives the CPU time of the calculations that were all executed on a 400 MHz Pentium notebook computer. To assess the efficiency of the adaptive grid method they should be compared with the CPU time of the same calculation on a uniform grid. Steady-state shallow-water calculations on a uniform grid are however very fast, mainly because of the effort that we have spent in optimizing the iterative flow solver (see Subsection 4.2). For example, a fully converged steady-state calculation for the same problem on a uniform grid of 400 finite volumes takes only 0.82 CPU seconds. The L_1 error in the water level, $\|\overline{h} - h\|_1$, turns out to be 2.06E-2 for this calculation. This result is more accurate and obtained faster than the result of the adaptive grid calculation with 70 finite volumes (see Table 1). On the other hand, the adaptive grid convergence criterion that we applied was rather severe. There is also still room for improvement of the adaptive grid solver (see, e.g., Subsection 4.2). We especially expect a significant gain in accuracy and efficiency once geometry smoothing will have been implemented.

7.2 Unsteady application

We present an unsteady adaptive grid application that, strictly speaking, is not even a genuine unsteady application. It is a fairly simple and yet extremely complicated test for moving adaptive grid methods, and shows clearly both the advantages and the shortcomings of our moving adaptive grid approach.

The test is simple: starting from the steady-state solution shown in Figure 4, calculate the steady-state solution on the adapted grid of Figure 5 by solving the flow equations both in time and in space using a time step of 1 second. Although a steady-state problem in physical space, it is *not* a steady-state problem in computational space where we solve (11) and (12).

We will first solve this problem in the standard way, applying the equidistribution principle in space and ignoring the effect that this has on the error in computational time. This is realized by solving per outer grid iteration the equations (48) with $c_\gamma = 0$. As can be seen in Figure 8, the effect is dramatic. Virtually all grid points are drawn toward the hydraulic jump region in the very first time step. As a consequence, many grid points move to a position with a totally different water depth and velocity. This leads to very large variations in computational time, and, since time derivatives and space derivatives are coupled, also to very large perturbations of the solution in space. This applies in particular to grid points at and in the vicinity of the bar.

The reason for this erroneous behavior is the non-uniform bottom profile in combination with the fact that the adaptive grid algorithm did not take into account the effect of the grid point redistribution on the error in time. Still, even a uniform bottom may be difficult to handle if the time discretization error is not taken into account. Similar solution perturbations as shown here (although probably not that strong) may for example occur if a sudden change at one of the boundaries forces the grid points to move through the domain in the direction of that boundary, crossing a region where due to the presence of waves

there may be a significant variation in water depth and velocity (cf. Section 6).

The erroneous motion of grid points can be suppressed by smoothing the grid velocity [5, 30, 31]. However, this slows down the entire grid adaptation process and can be expected to have in general a negative effect on the quality of the solution [22]. This can be compensated for by using a small time step but then the advantages of grid adaptation are not so clear anymore. A calculation on a uniform grid using a large number of points may actually be more efficient and more accurate.

A moving adaptive grid algorithm that minimizes the combined space-time numerical modeling error should perform better. The right-hand side of (48) should be able to detect large errors in time and avoid excessive grid point displacements whenever this would lead to large errors in time. On the other hand, large grid adaptations in regions where it has little effect on the accuracy in time are still possible.

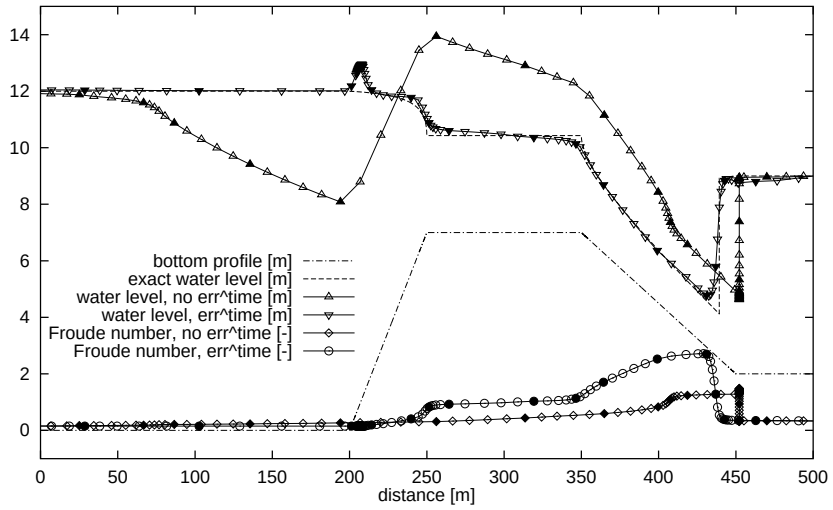

Figure 8: Unsteady shallow-water solution at first time level on adapted grid with and without compensation of time discretization error, 100 finite volumes

The result obtained in the first time step using (48) with $c_\gamma = 6$ (a typical value, see Subsection 5.3) and with $c_\tau = 10$ in (49) is shown in Figure 8 as well. The result is much better, but still not satisfactory. One notices the large concentration of grid points at the base of the bar which is due to the fact that the grid points are again drawn toward the hydraulic jump. The difference with the previous result without time error compensation is that this time the error minimization in time prevents the grid points from crossing the bar.

The overshoot in the water level at the base is probably due to the non-smoothness of the geometry. The results obtained in subsequent time steps show that this overshoot initiates a small wave moving essentially upstream and decaying rapidly. However, grid points start following that wave and are

then not available anymore to improve the accuracy elsewhere. We have observed that the solution error $\|\overline{h} - h\|_1$ did decrease from the very first time step, although very slowly (less than a factor 2 in 20 time steps) due to the perturbations introduced in the first time step.

No full convergence has been obtained for the adapted grids of this Subsection, although we did not spend much effort in finding suitable values for the different convergence parameters. We found that rather useless because of the fundamental shortcoming already mentioned in Section 6: the lack of adaptive flexibility when the grid is continuous in time. The results presented here reveal that this is a major drawback in shallow-water applications where grid points may have to move continuously across non-uniform parts of a channel geometry in order to get a high resolution in certain dynamically changing regions. This will always lead to relatively large errors in time and will always require some compensation mechanism. In our method the compensation is included automatically, based on the error in time and only active when necessary. Nevertheless, this compensation (and any such compensation mechanism) will inevitably limit the grid speed and hence the potential gain in accuracy. There may then be no advantage in using a moving adaptive grid technique.

The solution to this problem has already been suggested in Section 6: consider a grid that is piecewise discontinuous in time and minimize the error per time step (45) also with respect to $\overline{x}^{n-1}(\xi)$. The derivation of the optimal grid equations is straightforward and leads to an interesting result that can be summarized as follows. Average grid size $x_i^{n-\frac{1}{2}} - x_{i-1}^{n-\frac{1}{2}}$ will be optimized for minimal error in space, while grid displacement $x_i^n - x_i^{n-1}$ will be optimized for minimal error in time. This virtually independent minimization of the numerical modeling error in space and in time indicates that a discontinuously moving adaptive grid method may prove to be very efficient.

8 Conclusions

We have presented the development of a moving adaptive grid method that aims at minimizing the numerical modeling error (the part of the numerical solution error generated locally as a result of solving the model equations numerically), rather than the numerical solution error. Besides being virtually impossible to realize for problems of practical interest, the usefulness of the latter is limited because of the presence of physical modeling errors and data errors which effect should also be taken into account.

The idea behind the present approach is firstly to ensure that physics-based artificial smoothing terms form the dominant numerical modeling error to allow a direct comparison with physical modeling terms; and secondly to minimize the effect of the artificial smoothing terms by means of grid adaptation. We have shown that this is feasible in 1-D, although the presented numerical algorithm is still incomplete. One element that needs to be added is geometry smoothing, the importance of which is illustrated by the results.

Smoothness is the key element of the proposed method; it ensures that

the effect of discretization errors on the numerical solution is small, which is essential for a meaningful error analysis. The use of a compatible scheme is required to be able to analyse that effect, and to obtain a useful approximation of the numerical modeling error in space and in time as a function of the local grid parameters. The combination of artificial smoothing and a compatible discretization has enabled us to develop an error-minimizing moving adaptive grid algorithm, minimizing the effect of both discretization errors and smoothing errors.

The results clearly show the importance of taking the error in time into account when adapting the grid in space. However, grid adaptation can be very inefficient if the grid is forced to move continuously. This applies in particular to the unsteady shallow-water applications with non-uniform bottom that we are interested in. An extension that needs to be considered is therefore the development of a discontinuously moving adaptive grid method.

The complexity of the developed method is large. On the other hand, a high gain in accuracy is possible since the method is capable of using grid points very efficiently. For unsteady calculations this will only be true after the method has been extended with a grid that can move discontinuously in time.

There are indications that the 1-D error analysis of the compatible scheme that we have presented can be extended to several space dimensions, showing the correspondence between the multi-D numerical modeling error and multi-D interpolation errors. Multi-D interpolation errors depend however in a complicated way on the grid size, grid stretching, grid curvature and grid skewness. Minimizing these errors by means of grid adaptation will be very difficult to realize. On the other hand, elements of such a technique may possibly be combined with a more heuristic adaptive grid approach to arrive at better monitor functions and hence more efficient grid adaptation procedures for multi-D applications of practical interest.

References

[1] M. Arora and P. L. Roe, On postshock oscillations due to shock capturing schemes in unsteady flows, *J. Comput. Phys.* **130** (1997) 25–40.

[2] M. Borsboom, Development of an error-minimizing adaptive grid method, *Appl. Num. Math.* **26** (1998) 13–21.

[3] M. H. Chaudhry, *Open-Channel Flow* (Prentice-Hall, 1993).

[4] V. T. Chow, *Open-Channel Hydraulics* (McGraw-Hill, 1973).

[5] E. A. Dorfi and L. O'C. Drury, Simple adaptive grids for 1-D initial value problems, *J. Comput. Phys.* **69** (1987) 175–195.

[6] R. L. Dougherty, A. Edelman and J. M. Hyman, Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic Hermite interpolation, *Math. Comp.* **52** (1989) 471–494.

[7] K. Eriksson, D. Estep, P. Hansbo and C. Johnson, Introduction to adaptive methods for differential equations, *Acta Numerica* **4** (1995) 105–158.

[8] K. Eriksson and C. Johnson, Adaptive streamline diffusion finite element methods for stationary convection-diffusion problems, *Math. Comp.* **60** (1993) 167–188.

[9] R. M. Furzeland, J. G. Verwer and P. A. Zegeling, A numerical study of three moving-grid methods for one-dimensional partial differential equations which are based on the method of lines, *J. Comput. Phys.* **89** (1990) 349–388.

[10] P. Garcia-Navarro, F. Alcrudo and J. M. Saviron, 1-D open-channel flow simulation using TVD-McCormack scheme, *J. Hydr. Engrg.* **118** (1992) 1359–1372.

[11] T. Geßner, D. Kröner, B. Schupp and M. Wierse, Finite volume methods for conservation laws and convection-dominated diffusion equations, in: F. Benkhaldoun and R. Vilsmeier, eds., *Finite Volumes for Complex Applications – Problems and Perspectives* (Hermes, Paris, 1996) 61–76.

[12] D. F. Hawken, J. J. Gottlieb and J. S. Hansen, Review of some adaptive node-movement techniques in finite-element and finite-difference solutions of partial differential equations, *J. Comput. Phys.* **95** (1991) 254–302.

[13] C. Hirsch, *Numerical Computation of Internal and External Flows, Volume 2* (Wiley, 1990).

[14] P. Houston, J. A. Mackenzie, E. Süli and G. Warnecke, A posteriori error analysis for numerical approximations of Friedrichs systems, *Numer. Math.* **82** (1999) 433–470.

[15] P. Houston, R. Rannacher and E. Süli, A posteriori error analysis for stabilised finite element approximations of transport problems, *Tech. Rep. NA-99/04* (Oxford University Computing Laboratory, 1999).

[16] W. Huang and R. D. Russell, Analysis of moving mesh partial differential equations with spatial smoothing, *SIAM J. Numer. Anal.* **34** (1997) 1106–1126.

[17] C. Johnson, R. Rannacher and M. Boman, Numerics and hydrodynamic stability: toward error control in computational fluid dynamics, *SIAM J. Numer. Anal.* **32** (1995) 1058–1079.

[18] A. A. Khan and P. M. Steffler, Physically based hydraulic jump model for depth-averaged computations, *J. Hydr. Engrg.* **122** (1996) 540–548.

[19] R. J. LeVeque and H. C. Yee, A study of numerical methods for hyperbolic conservation laws with stiff source terms, *J. Comput. Phys.* **86** (1990) 187–210.

[20] J. A. Mackenzie, The efficient generation of simple two-dimensional adaptive grids, *SIAM J. Sci. Comput.* **19** (1998) 1340–1365.

[21] E. A. Meselhe, F. Sotiropoulos and F. M. Holly Jr., Numerical simulation of transcritical flow in open channels, *J. Hydr. Engrg.* **123** (1997) 774–783.

[22] Y. Qiu and D. M. Sloan, Numerical solution of Fisher's equation using a moving mesh method, *J. Comput. Phys.* **146** (1998) 726–746.

[23] T. Sonar and E. Süli, A dual graph-norm refinement indicator for finite volume approximations of the Euler equations, *Numer. Math.* **78** (1998) 619–658.

[24] B. van Leer and W. A. Mulder, Relaxation methods for hyperbolic equations, in: F. Angrand, A. Dervieux, J. A. Desideri and R. Glowinsky, eds., *Numerical Methods for the Euler Equations of Fluid Dynamics* (SIAM, 1985) 312–333.

[25] J. G. Verwer, J. G. Blom, R. M. Furzeland and P. A. Zegeling, A moving grid method for one-dimensional PDEs based on the method of lines, in: J. E. Flaherty, P. J. Paslow, M. S. Shephard and J. D. Vasilakis, eds., *Adaptive Methods for Partial Differential Equations* (SIAM, 1989) 160–175.

[26] J. von Neumann and R. D. Richtmyer, A method for the numerical calculation of hydrodynamic shocks, *J. Applied Physics* **21** (1950) 232–237.

[27] G. P. Warren, W. K. Anderson, J. L. Thomas and S. L. Krist, Grid convergence for adaptive methods, in: M. J. Baines and K. W. Morton, eds., *Numerical Methods for Fluid Dynamics 4* (Oxford University Press, 1993) 317–328.

[28] F. M. White, *Viscous Fluid Flow* (McGraw-Hill, 1974).

[29] Y. Xiang, N. R. Thomson and J. F. Sykes, Fitting a groundwater contaminant transport model by L_1 and L_2 parameter estimators, *Adv. Water Res.* **15** (1992) 303–310.

[30] P. A. Zegeling, Moving-grid methods for time-dependent partial differential equations, *CWI-tract No. 94* (Centre for Math. and Comp. Science, Amsterdam, 1993).

[31] P. Zegeling, M. Borsboom and J. van Kester, Adaptive moving grid solutions of a shallow-water transport model with steep vertical gradients, in: V. N. Burganos, ed., *Proc. 12th Int. Conf. On Comput. Methods in Water Resources, Volume 2* (Computational Mechanics Publications, 1998) 427–434.