# Numerical Analysis of the Shallow Water Equations

P.J. van der Houwen, B.P. Sommeijer,
J.G. Verwer, F.W. Wubs
*Centre for Mathematics and Computer Science*
*P.O. Box 4079, 1009 AB Amsterdam, The Netherlands*

## 1. Introduction

In this contribution we will give an expository account of the numerical analysis of hyperbolic differential equations. Recently, these equations have become of particular interest to the Numerical Mathematics Department of the CWI. Our main purpose is to apply this analysis to the *shallow water equations* (SWEs) and therefore, throughout this paper, we will illustrate the analysis by giving theoretical as well as numerical results for the SWEs. In this introductory section we start with a description of the origin of the SWEs.

A windfield (or tidal forces) perturbing a water surface which is initially at rest will generate two types of water waves: long (or tidal) waves and short waves. In the long waves the wave length is large compared with the height of the water surface and the vertical accelerations are small compared with the horizontal accelerations. In the short waves the wave length is smaller than the depth of the water and the vertical accelerations are no longer insignificant. We will concentrate on *long waves in shallow water* generated by wind forces (or tidal forces).

Due to the movement of the water, three other forces will become active: (i) bottom friction (ii) Coriolis force (iii) gravity. Let **R** denote the total resulting horizontal force, then we have the following equation

$$\frac{D\mathbf{u}}{Dt} := \frac{\partial \mathbf{u}}{\partial t} + \left[ u\frac{\partial}{\partial x} + v\frac{\partial}{\partial y} \right] \mathbf{u} = \mathbf{R}, \tag{1.1a}$$

where $\mathbf{u}=(u,v)^T$ denotes the depth-averaged velocity of the water and $(x,y)$ represents an orthogonal coordinate system. In addition to this equation of motion we have the continuity equation (e.g. [7, p. 179])

$$\frac{\partial h}{\partial t} + \frac{\partial hu}{\partial x} + \frac{\partial hv}{\partial y} = 0 \tag{1.1b}$$

where $h$ denotes the depth. Combining these equations and deriving expressions for the forces due to gravity and Coriolis (see e.g. [7, p. 190] we arrive at the SWEs:

$$\mathbf{w}_t = A(\mathbf{w})\mathbf{w}_x + B(\mathbf{w})\mathbf{w}_y + C(y)\mathbf{w} + \mathbf{r}(\mathbf{w}) \tag{1.2}$$

where $\mathbf{w} = (u,v,h)^T$ and where the matrices $A,B$ and $C$ are defined by

$$A(\mathbf{w}) = -\begin{bmatrix} u & 0 & g \\ 0 & u & 0 \\ h & 0 & u \end{bmatrix}, \quad B(\mathbf{w}) = -\begin{bmatrix} v & 0 & 0 \\ 0 & v & g \\ 0 & h & v \end{bmatrix}, \quad C(y) = \begin{bmatrix} 0 & f & 0 \\ -f & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

The vector $\mathbf{r}(\mathbf{w})$ represents the bottom friction, forces due to bottom irregularities, to the wind field and other atmospheric forces (for a discussion of this term we refer to [7]); $f$ and $g$ denote the Coriolis parameter and the acceleration due to gravity, respectively.

Omitting the external forces represented by $\mathbf{r}(\mathbf{w})$ we obtain a system of *hyperbolic* equations *in two space dimensions* (notice that the matrices $A$ and $B$ have the distinct, *real* eigenvalues $\{-u \pm \sqrt{gh}, -u\}$ and $\{-v \pm \sqrt{gh}, -v\}$, respectively; these eigenvalues correspond to the characteristic directions of the SWEs).

A particularly difficult problem in deriving a mathematical model for a shallow sea is the formulation of the boundary conditions along the 'non-coastal' boundaries of the sea (along 'coastal' boundaries one usually imposes the 'rigid wall' condition which requires that the velocity component normal to the coast vanishes). For a discussion of boundary conditions we refer to [7]. In our examples we will use periodic boundary conditions along the non-coastal boundaries.

## 2. The space-discretization

A flexible approach in the numerical solution of evolutionary problems in PDEs is obtained by applying the so-called *method of lines*. Herewith the numerical solution process is considered as to consist of two parts, viz. *space-discretization* and *time-integration*.

In the space-discretization the PDE is converted into a system of ODEs by discretizing the space variables, while the time variable is left continuous. Usually, the space-discretization is performed, either by the finite difference method [32], or by the finite element method [42]. Spectral methods can also be applied, however [10]. In this paper we restrict our attention to the finite difference method since this method is easier to present to the nonspecialist. Moreover, in the field of hyperbolic PDEs the finite difference method is still most widely used. We note that Praagman [35] has implemented the finite element method for the shallow water equations (1.2).

In the time-integration the resulting system of ODEs, often called the semi-discrete problem, is integrated by one of the many existing integration formulas which is most appropriate for the problem at hand. This part of the discretization process will be the subject of Section 3.

### *2.1. Two simplified models*

Throughout this contribution we will discuss examples and numerical experiments with the aim of illustrating the various aspects and difficulties which are encountered in the numerical solution of hyperbolic problems such as (1.2). For that purpose we will resort to two simplified models which we give first.

*Model 1. A conservative shallow water equation*

Following [12] we consider the nonlinear hyperbolic system

$$\mathbf{w}_t = A(\mathbf{w})\mathbf{w}_x + B(\mathbf{w})\mathbf{w}_y + C(y)\mathbf{w}, \quad (x,y)\in\Omega, \quad t\geqslant 0, \tag{2.1.1}$$

for the dependent vector variable $\mathbf{w}=[u,v,\phi]^T$, where $u$ and $v$ have the same meaning as in (1.2) and $\phi=2\sqrt{gh}$. Further

$$A(\mathbf{w}) = -\begin{bmatrix} u & 0 & \frac{1}{2}\phi \\ 0 & u & 0 \\ \frac{1}{2}\phi & 0 & u \end{bmatrix}, \quad B(\mathbf{w}) = -\begin{bmatrix} v & 0 & 0 \\ 0 & v & \frac{1}{2}\phi \\ 0 & \frac{1}{2}\phi & v \end{bmatrix}, \quad C(y) = \begin{bmatrix} 0 & f & 0 \\ -f & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

It can be verified, using a simple transformation, that (2.1.1) can be obtained from (1.2), provided all external forces except the Coriolis force $f$ are neglected. We prefer to use the same notation $\mathbf{w}$ for the dependent variable, although its third component has a different meaning than in (1.2). Any confusion is precluded. We observe that in the numerical solution process the treatment of the external forces is relatively simple. Hence, unless otherwise stated, we neglect these forces in our examples and experiments.

An important tool in the analysis of evolutionary problems in PDEs, analytically as well as numerically, is the total energy integral. For (2.1.1) the total energy can be expressed as

$$E(t) = \frac{1}{2g}\iint_{\Omega}(u^2+v^2+\tfrac{1}{4}\phi^2)\tfrac{1}{4}\phi^2 dxdy. \tag{2.1.2}$$

The origin of the name energy integral is that in many applications the physical energy of the physical system underlying to the partial differential equations can be expressed by an integral expression. This expression, in turn, is a convenient tool for examining well-posedness questions [36]. For example, if the physical energy is conserved, $E$ must remain constant in time. If energy dissipates, $E$ should monotonically decrease in time. $E$ is also useful for finding sensible boundary conditions.

Following the aforementioned authors, we define the rectangle

$$\Omega = \{(x,y):0\leqslant x\leqslant L, \quad 0\leqslant y\leqslant D\}. \tag{2.1.3}$$

Then, using the boundary conditions,

$$\mathbf{w}(x,y,t) = \mathbf{w}(x+L,y,t), \quad v(x,0,t)=v(x,D,t)=0, \tag{2.1.4}$$

a straightforward computation reveals that $\dot{E}(t)=0$, i.e., these boundary conditions imply that (2.1.1) conserves the total energy $E(t)$. The conservation of energy property should be accounted for in the discretization of (2.1.1). Observe that (2.1.4) implies periodicity in the $x$-direction and that no boundary conditions are necessary for $u$ and $\phi$ at $y=0$, $D$. Other boundary conditions may also lead to well-posedness of (2.1.1) on $\Omega\times(t>0)$. It is also of interest to observe that if we add bottom friction to (2.1.1), i.e., if we replace the matrix $C$ by

$$C = \begin{bmatrix} -\lambda & f & 0 \\ -f & -\lambda & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad \lambda=\lambda(x,y,w)>0,$$

it follows that $\dot{E}(t)<0$, which means energy dissipation. Finally, for the general problem (1.2) it may happen that $E$ will increase in time due to the influence of the source term or energy transport through boundaries.

For future reference we list some specific problem parameters for the just described initial-boundary value problem [11]:

Coriolis force $f=\hat{f}+\beta(y-D/2)$
Initial height function

$$h(x,y) = H_0+H_1\tanh(9(D/2-y)/2D)+H_2\mathrm{sech}^2(9(D/2-y)/D)\sin(2\pi x/L)$$

Initial velocities $u=-gf^{-1}\partial h/\partial y$, $v=gf^{-1}\partial h/\partial x$

$L=6.0\,10^6$ m, $D=4.4\,10^6$ m, $\hat{f}=10^{-4}$ sec$^{-1}$, $\beta=1.5\,10^{-11}$ sec$^{-1}m^{-1}$, $g=10$ m sec$^{-2}$, $H_0=2000$ m, $H_1=220$ m, $H_2=133$ m.

*Model 2. A one-dimensional incompressible flow equation*
Consider the one-space dimensional hyperbolic system

$$\mathbf{w}_t = B(\mathbf{w})\mathbf{w}_y, \quad y\in\Omega=[0,D], \quad t\geq 0,$$
$$B(\mathbf{w}) = -\begin{bmatrix} v & \frac{1}{2}\phi \\ \frac{1}{2}\phi & v \end{bmatrix}, \quad \mathbf{w}=[v,\phi]^T \tag{2.1.5}$$

which is obtained from equation (2.1.1) by dropping the $x$-dependent term $A(\mathbf{w})\mathbf{w}_x$ and the term $C(y)\mathbf{w}$. We again impose the boundary condition (cf. (2.1.4))

$$v(0,t) = v(D,t)=0, \tag{2.1.6}$$

while no condition for $\phi$ is prescribed. Hence, for $\phi$ the boundaries $y=0,D$ are open. The total energy integral $E(t)$ now reads (apart from a constant)

$$E(t) = \int_0^D (v^2+\tfrac{1}{4}\phi^2)\tfrac{1}{4}\phi^2 dy, \tag{2.1.7}$$

and again we have conservation of total energy, i.e., $\dot{E}(t)=0$ for $t\geq 0$. Numerically this one-space dimensional flow problem has similar properties as problem (2.1.1)-(2.1.4).

### *2.2. Finite difference space-discretization*

Let

$$\mathbf{w}_t = \mathcal{F}(x,y,t,\frac{\partial}{\partial x},\frac{\partial}{\partial y})\mathbf{w}, \quad (x,y)\in\Omega, \quad t\geqslant 0, \tag{2.2.1}$$

or, in case of one-space dimension,

$$\mathbf{w}_t = \mathcal{F}(x,t,\frac{\partial}{\partial x})\mathbf{w}, \quad x\in\Omega, \quad t\geqslant 0, \tag{2.2.1$'$}$$

formally represent a system of hyperbolic equations, given on the space domain $\Omega$. Suppose that on the boundary $\partial\Omega$ of $\Omega$ correct boundary conditions

$$B(\mathbf{w},t) = 0, \quad \text{on } \partial\Omega \tag{2.2.2}$$

are defined. The space-discretization of this problem essentially consists of three steps: (i) A grid $\Omega_\Delta$ must be defined covering $\Omega\cup\partial\Omega$. We use the symbol $\Delta$ as a formal notation for the grid distance which, of course, may vary over $\Omega_\Delta$. (ii) Appropriate finite difference replacements for the operators $\partial/\partial x$, $\partial/\partial y$ must be selected at all points of $\Omega_\Delta$. (iii) The boundary conditions must be taken into account.

Example 2.2.1. The standard finite difference space-discretization of our second model (2.1.5) proceeds as follows. The interval $[0,D]$ is divided into $N_y$ subintervals of equal length $\Delta y$, thus defining the grid

$$\{y_k : y_k = k\Delta y \quad \text{for} \quad k=0(1)N_y\}.$$

On this grid we introduce the so-called grid function

$$\mathbf{W} = [\mathbf{W}_0, \ldots, \mathbf{W}_{N_y}]^T, \quad \mathbf{W}_k = [V_k, \Phi_k]^T,$$

where each component vector $\mathbf{W}_k(t)=[V_k(t),\Phi_k(t)]^T$ depends on time $t$ and is meant to approximate $\mathbf{w}(y,t)$, the exact solution of problem (2.1.5)-(2.1.6), at the gridpoint $y_k$. Hence $\mathbf{W}$ is still time continuous. The approximation is defined by the choice of the finite difference formulas for approximating the space derivative $\mathbf{w}_y$. At this place we have to face our first difficulty, i.e. the open boundary for $\phi$ which forces us to approximate $v_y$ in a different way at the points $y_0=0$, $y_{N_y}=D$. We consider the standard second order difference formula

$$\mathbf{w}_y(y_k,t) \simeq \frac{1}{2\Delta y}(\mathbf{W}_{k+1}(t)-\mathbf{W}_{k-1}(t)) \tag{2.2.3}$$

for $k=1(1)N_y-1$. For $k=0$, $N_y$ define $V_k(t)=0$ according to the boundary conditions (2.1.6) and use one-sided first order differences for $v_y$ at these points, i.e.,

$$v_y(0,t) \simeq \frac{1}{\Delta y}\,(V_1(t)-V_0(t)),$$
$$v_y(D,t) \simeq \frac{1}{\Delta y}\,(V_{N_y}(t)-V_{N_y-1}(t)). \tag{2.2.4}$$

After replacing $w_y$ on $\{y_k\}$ by these finite difference quotients, the system of ODEs

$$\begin{aligned}
\dot{\mathbf{W}}_0 &= B(\mathbf{W}_0)\frac{\mathbf{W}_1-\mathbf{W}_0}{\Delta y}\,, \quad V_0(t)=0,\\
\dot{\mathbf{W}}_k &= B(\mathbf{W}_k)\frac{\mathbf{W}_{k+1}-\mathbf{W}_{k-1}}{2\Delta y}\,, \quad k=1(1)N_y-1,\\
\dot{\mathbf{W}}_{N_y} &= B(\mathbf{W}_{N_y})\frac{\mathbf{W}_{N_y}-\mathbf{W}_{N_y-1}}{\Delta y}\,, \quad V_{N_y}(t)=0,
\end{aligned} \tag{2.2.5}$$

results. This system is a time continuous, semi-discrete version of the original initial-boundary value problem (2.1.5)-(2.1.6). □

The above example illustrates that the process of space-discretization converts an initial-boundary value problem for a PDE into an initial value problem for a system of ODEs with $t$ as independent variable. Henceforth we will denote this latter system by

$$\dot{\mathbf{W}} = \mathbf{F}(t,\mathbf{W}), \quad t\geqslant 0, \quad \mathbf{W}(0) \text{ prescribed.} \tag{2.2.6}$$

This system is usually called the time-continuous, semi-discrete system. Obviously, there is an intimate relationship with the grid $\Omega_\Delta$. The vector function $\mathbf{F}$ is always parameterized with the grid distance $\Delta$. $\mathbf{F}$ approximates the hyperbolic operator $\mathcal{F}$ on the grid $\Omega_\Delta$. The length of the vector $\mathbf{W}$, the gridfunction which approximates $w$ on $\Omega_\Delta$, depends on $\Delta$ too. Occasionally, if this clarifies the discussion, we will therefore use the notation

$$\dot{\mathbf{W}}_\Delta = \mathbf{F}_\Delta(t,\mathbf{W}_\Delta) \tag{2.2.6$'$}$$

instead of (2.2.6). Further, we shall mostly use the autonomous notation $\dot{\mathbf{W}}=\mathbf{F}(\mathbf{W})$ as our two example models are autonomous.

As a further illustration we describe the space-discretization of our first model (2.1.1)-(2.1.4). Because $\Omega$ is a rectangle the derivation is nearly the same as in Example 2.2.1.

Example 2.2.2. Divide the $x$-interval and $y$-interval into $N_x$ and $N_y$ subintervals of length $\Delta x$ and $\Delta y$, respectively. On the grid

$$\{(x_j,y_k) : x_j=j\Delta x,\ j=1(1)N_x \text{ and } y_k=k\Delta y,\ k=0(1)N_y\},$$

we define $\mathbf{W}_{jk}=[U_{jk},V_{jk},\Phi_{jk}]^T$ as the time continuous approximation for $\mathbf{w}(x_j,y_k,t)$ which results from the application of second order symmetrical differences at all interior points and first order one-sided differences at the boundary points $(x_j,y_k)$, $k=0,N_y$. In the $x$-direction symmetrical differencing is possible everywhere because of the periodicity, i.e., $\mathbf{W}_{0k} = \mathbf{W}_{N_x k}$ and $\mathbf{W}_{N_x+1,k} = \mathbf{W}_{1k}$. Note that $V_{j0} = V_{jN_y} = 0$ due to (2.1.4). □

*Grid staggering.* Grid staggering, originally introduced by HANSEN [13], is often applied in space-discretization. By this technique $u,v$ and $\phi$ are calculated at different grid points. Herewith, it is possible to decrease the storage requirements by a factor four without loss of accuracy with respect to the main terms of the SWEs. We will show the idea using the one-dimensional equation (2.1.5) of which the main part is described by:

$$\begin{aligned} v_t &= -\frac{1}{2}\phi_0\phi_y \\ \phi_t &= -\frac{1}{2}\phi_0 v_y, \end{aligned} \tag{2.2.7}$$

where we have frozen the coefficients of $\phi_y$ and $v_y$. If this system is semi-discretized in the usual way, we obtain

$$\begin{aligned} (V_t)_i &= -\frac{1}{2}\Phi_0(\Phi_{i+1} - \Phi_{i-1})/2\Delta y, \\ (\Phi_t)_j &= -\frac{1}{2}\Phi_0(V_{j+1} - V_{j-1})/2\Delta y. \end{aligned} \tag{2.2.8}$$

Observe that in the case where $i$ is running through even values and $j$ through odd values, the set of equations is independent of its complement $\{i\ odd,\ j\ even\}$. Thus we may omit one of these sets, without loss of accuracy, thereby reducing the number of equations by a factor two. Applying the same technique in the $y$-direction will lead to a final reduction by a factor four. A part of the resulting grid is depicted in Fig. 2.2.1.

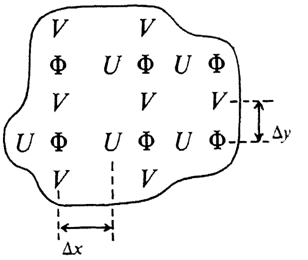

FIGURE 2.2.1

The neglected terms have to be composed by use of the variables of the reduced set, implying a small loss of accuracy due to averaging techniques.

### 2.3. *Some fundamental topics*

The choice of a difference method for discretizing hyperbolic equations, such as the shallow water equations, depends on many factors. These may vary from theoretical to practical and will always depend on the specific application area (see e.g. [33], p. 718). In this subsection we will briefly discuss some basic topics concerning the space-discretization. These include *consistency, stability, conservation laws, open boundaries, curved boundaries.* Our purpose is to give some insight in the choice and analysis of finite difference models. It is emphasized that the field is so diverse that completeness is impossible in the present paper. The above topics, however, play a role in a lot of investigations and are of a fundamental nature.

*Consistency.* The approximation is useful only if it is consistent, i.e., upon grid refinement the approximation should converge to the continuous. problem. Normally there is no difficulty in setting up a consistent approximation. A difficulty may lie in finding approximations which converge sufficiently fast if Δ→0. Further, an always returning and important question is, how accurate is the numerical solution computed on a certain grid? We will briefly consider these matters for the space-discretization error.

Consider a general initial-boundary value problem (2.2.1)-(2.2.2) and a corresponding semi-discrete approximation (2.2.6 ′ ). Let $\mathbf{w}_\Delta$ denote the restriction of the fully continuous function $\mathbf{w}$ to the space grid. Hence, in the case of Example 2.2.1, we have

$$\mathbf{w}_\Delta(t) = [\mathbf{w}(0,t), \dots ,\mathbf{w}(k\Delta y,t), \dots ,\mathbf{w}(N_y\Delta y,t)]^T.$$

Further, let $\alpha_\Delta$ denote the *space-approximation error*

$$\alpha_\Delta = \mathbf{F}_\Delta(\mathbf{w}_\Delta)-\dot{\mathbf{w}}_\Delta. \tag{2.3.1}$$

This error is obtained by substituting the exact solution $\mathbf{w}$ into the semi-discrete problem. It measures how much the semi-discrete operator deviates from the partial differential operator including the boundary conditions. Next, let $\eta_\Delta$ denote the *space-discretization error*

$$\eta_\Delta = \mathbf{W}_\Delta-\mathbf{w}_\Delta. \tag{2.3.2}$$

It follows that $\eta_\Delta$ is a solution of the ordinary differential system

$$\dot{\eta} = \mathbf{F}_\Delta(\mathbf{w}_\Delta+\eta_\Delta)-\mathbf{F}_\Delta(\mathbf{w}_\Delta)+\alpha_\Delta,$$

which can be rewritten to

$$\begin{aligned}\dot{\eta}_\Delta &= M_\Delta(t)\eta_\Delta+\alpha_\Delta,\\ M(t) &= \int_0^1 F'_\Delta(\mathbf{w}_\Delta+\theta\eta_\Delta)d\theta.\end{aligned} \tag{2.3.3}$$

Here $F'_\Delta$ denotes the Jacobian matrix of the vector function $\mathbf{F}_\Delta$ which is assumed to exist. Note that we have used the mean value theorem for vector functions [28, p. 71].

The above derivation reveals three properties of the space error $\eta_\Delta$ which are worth mentioning. (i) Though $\eta_\Delta$ comes into existence only by discretizing space variables, this error is really time-dependent, even when $\alpha_\Delta \neq \mathbf{0}$ is constant. (ii) The space error depends on the stability behaviour of the ordinary differential equation (2.3.3) when proceeding in time. Evidently, this equation should possess similar stability properties as the underlying partial differential equation. (iii) The smaller the approximation error $\alpha_\Delta$, the smaller the space error $\eta_\Delta$, certainly if (2.3.3) is a stable system. Hence, if the approximation is consistent, i.e., $\alpha_\Delta \to \mathbf{0}$ if $\Delta \to 0$, the space error $\eta_\Delta$ will converge to zero, for all $t$, upon grid refinement.

*Consistency and stability.* To clarify the aspect of stability in the above reasoning we will give a typical stability estimate for $\eta_\Delta$. This stability estimate gives insight in the dependence of the space error $\eta_\Delta$ on the approximation error (2.3.1).

Let $\|\cdot\|$ be some norm on the finite dimensional solution space of the system $\dot{\mathbf{W}}_\Delta = \mathbf{F}_\Delta(\mathbf{W}_\Delta)$, e.g., a known $l^p$-norm. Then, according to [2, p. 13], it follows that

$$\|\eta_\Delta(t)\| \leqslant e^{\mu_{\max} t}\|\eta_\Delta(0)\| + \int_0^t e^{\mu_{\max}(t-\tau)}\|\alpha_\Delta(\tau)\|d\tau,$$

where

$$\mu_{\max} = \max_{\mathbf{W}} \mu[F'_\Delta(\mathbf{W})],$$

$\mu$ being the logarithmic matrix norm belonging to $\|\cdot\|$ (for specific details about this result and the use and meaning of the logarithmic norm, the reader may also consult [6]). Let us assume a zero space error at the initial time $t=0$. Then

$$\|\eta_\Delta(t)\| \leqslant \int_0^t e^{\mu_{\max}(t-\tau)}\|\alpha_\Delta(\tau)\|d\tau. \tag{2.3.4}$$

In many instances the quantity $\mu_{\max}$ can be proved to be independent of the grid distance $\Delta$. In that case, this worst case estimate proves that

$$\|\eta_\Delta(t)\| \leqslant c(t)\max_t \|\alpha_\Delta(t)\|,$$

$c(t)$ being independent of $\Delta$. Consequently, if the finite difference formula in all gridpoints is consistent of order $q$, i.e., in a formal notation,

$$\alpha_\Delta = O(\Delta^q), \quad \Delta \to 0, \tag{2.3.5}$$

it follows that $\eta_\Delta(t) = O(\Delta^q)$ as $\Delta \to 0$, establishing $q$-th order convergence for the semi-discrete solution $\mathbf{W}(t)$. Apparently, in the above derivation time $t$ was kept fixed, i.e., (2.3.5) applies for all $t$ but the constant involved still depends on $t$[1].

[1] More details concerning the actual application of the above derivation can be found in 'Convergence of Methods of Lines Approximations to PDEs', J.G. Verwer and J.M. Sanz-Serna, CWI Report NM-R8404.

EXAMPLE 2.3.1. Let us examine the approximation error $\alpha_\Delta$ for the semi-discrete one-dimensional incompressible flow equation (2.2.5). We will denote the $k$-th component of $\alpha_\Delta$ by $\alpha_{\Delta,k}$. If $\mathbf{w}$ is at least two times differentiable, a straightforward Taylor expansion of $\mathbf{w}(y_k \pm \Delta y, t)$ at all grid points $y_k$ shows that

$$\alpha_{\Delta,O} = O(\Delta y),\ \alpha_{\Delta,k} = O((\Delta y)^2)\ (1 \leq k \leq N_y - 1),\ \alpha_{\Delta,N_y} = O(\Delta y)$$

as $\Delta y \rightarrow 0$. Consequently, due to the first order approximations at the boundary, $q = 1$ in relation (2.3.5) instead of $q = 2$.

A decrease of accuracy at a boundary may be reduced by using a higher-order difference formula. This may, however, destroy the stability of the space-discretization. In our terminology this means that the error equation (2.3.3) becomes unstable. We will illustrate this later. First we proceed with the topic conservation laws which provides us further means for examining stability. □

*Conservation laws and stability.* Let us once more consider the stability estimate (2.3.4) for the differential system (2.3.3) which determines the space error $\eta_\Delta$. Obviously, if it is required to solve the initial value problem over a large time interval it is highly desirable that the semi-discrete system itself is stable. Stability corresponds to a nonpositive logarithmic matrix norm, so the worst case estimate then reads

$$||\eta_\Delta(t)|| \leq \int_0^t ||\alpha_\Delta(\tau)|| d\tau \leq t \max_{0 \leq \tau \leq t} ||\alpha_\Delta(\tau)||. \tag{2.3.6}$$

This estimate still allows a linear growth of the space error, but should be considered as rather pessimistic. If system (2.3.3) is stable, it is to be expected that an eventual growth of $\eta_\Delta$ is less than the linear growth of the above estimate. Certainly this is true if $M_\Delta(t)$ is a constant matrix, i.e. if $F'_\Delta$ is constant.

One must reckon with a much more serious situation if the semi-discrete system is unstable, which corresponds to a positive $\mu_{\max}$ in the estimate (2.3.4). Then the worst case estimate allows an exponential growth of the space error which may be fatal. From practical experiences we know that exponential growth, also called 'blow up', really occurs. The next example serves to illustrate this.

EXAMPLE 2.3.2. [6]. Consider the ODE system described in Example 2.2.2 which is a semi-discrete approximation to Model 1 of subsection 2.1. On the space grid $\Omega_\Delta$ we approximate the total energy $E(t)$, given by (2.1.2), by the trapezoidal approximation

$$E_\Delta = \frac{\Delta x \Delta y}{2g} \sum_{j=1}^{N_x} \{ \sum_{k=1}^{N_y - 1} \hat{\mathbf{W}}_{jk}^T \hat{\mathbf{W}}_{jk} + \tfrac{1}{2}(\hat{\mathbf{W}}_{j0}^T \hat{\mathbf{W}}_{j0} + \hat{\mathbf{W}}_{jN_y}^T \hat{\mathbf{W}}_{jN_y}) \}, \tag{2.3.7}$$

where

$$\hat{\mathbf{W}}_{jk} := [\tfrac{1}{2}U_{jk}\Phi_{jk},\ \tfrac{1}{2}V_{jk}\Phi_{jk}, \tfrac{1}{4}\Phi_{jk}^2]^T,$$

$U_{jk}$, $V_{jk}$ and $\Phi_{jk}$ being the components of $\mathbf{W}_{jk}$ defined in Example 2.2.2. Because we have conservation of total energy in Model 1 a legitimate requirement is that the semi-discrete model conserves the semi-discrete total energy (2.3.7), i.e., $\dot{E}_\Delta(t)=0$. It turns out that this requirement is not fulfilled. For $\Delta x = \Delta y = 200$ km we have computed $E_\Delta$ over a relatively large time interval of approximately 17 days by means of a highly accurate, stable numerical integration method. Figure 2.3.1 shows a plot of $E_\Delta$. One can see that after approximately 17 days a sudden 'blow up', or energy explosion, occurs. Of course, this explosion completely ruins all results of the numerical computation. □

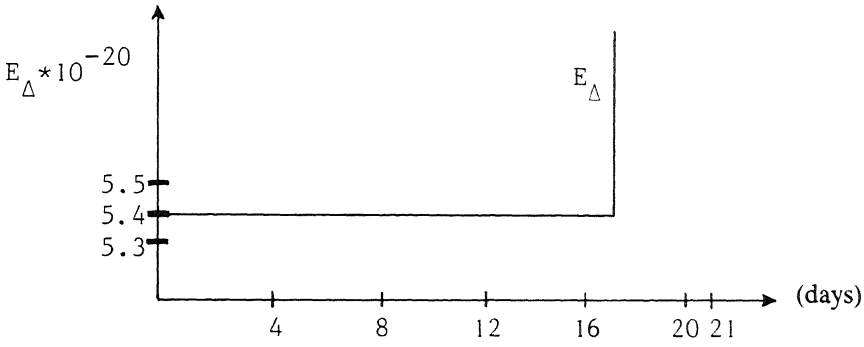

FIGURE 2.3.1. Explosion of semi-discrete total energy

From the stability estimate (2.3.4) one can deduce that the 'blow up' will be delayed if one refines $\Omega_\Delta$. In practice this is much too costly, however, and instead one imposes artificial damping or one tries to obey the conservation laws. The (ad hoc) technique of artificial damping is widely known. Alternative names are artificial dissipation or artificial viscosity. The basic idea is to add small terms to the original PDE such that the semi-discrete approximation becomes stable. Observe that bottom friction in our Model 1 also has a stabilizing influence. The inherent difficulty of this technique is that one has to make a compromise between stability and accuracy, for one changes the original PDE and thus solves a different problem. Fortunately, in many applications one is satisfied with a rough accuracy and then the technique of artificial dissipation performs quite satisfactorily.

The instability illustrated in Figure 2.3.1 is essentially due to the fact that in the space-discretization described in Example 2.2.2 the conservation of total energy $E(t)$ has not been taken into account. If the PDE conserves physical quantities such as mass, total energy, momentum, it is sensible to transfer these properties to the finite difference approximation in order to improve it. In this connection the conservation law of total energy is an important tool for the stability analysis due to the fact that $E$ and $E_\Delta$ can always be written as quadratic functionals. Hence if $E_\Delta$ is constant in time 'blow up' simply cannot happen. In numerical literature one has introduced the name energy method for stability analyses along the lines of energy conservation laws [36]. The energy method is of great use for examining the stability of particularly nonlinear models, since here the standard classical approach of Fourier analysis cannot be applied.

EXAMPLE 2.3.3. Following [6] we will briefly illustrate the energy method for the one-dimensional model (2.1.5) which we prefer for reasons of presentation. All results go through for the related two-dimensional model (2.1.1). Let $\dot{\mathbf{W}}=\mathbf{F}(\mathbf{W})$ denote a semi-discrete version of the PDE (2.1.5). The first step in the standard energy method is to select an appropriate energy norm, i.e., a norm such that $\|\mathbf{W}(t)\|^2=E_\Delta(t)$. Suppose that we can deal with an inner product norm $\|\mathbf{W}\|^2=<\mathbf{W},\mathbf{W}>$. Then, if $E_\Delta$ is required to be constant in time, say, we have

$$\frac{d}{dt}\|\mathbf{W}(t)\|^2 = <\mathbf{F}(\mathbf{W}(t)),\ \mathbf{W}(t)>=0.$$

A function $\mathbf{F}$ which satisfies this property for all vectors $\mathbf{W}$ is called conservative, on the analogy of the term used for the PDE.

Now consider the energy integral (2.1.7) and define the transformation of variables $v\to\frac{1}{2}\phi v$, $\phi\to\frac{1}{4}\phi^2$. Then $E$ is in the form of a quadratic functional, viz.

$$E(t) = \int_0^D (v^2+\phi^2)dy.$$

Next, introduce the inner product generated by the trapezoidal rule approximation $E_\Delta$ for $E$:

$$\begin{aligned} <\mathbf{W},\tilde{\mathbf{W}}> &= \Delta y[\sum_{k=1}^{N_y-1}\mathbf{W}_k^T\tilde{\mathbf{W}}_k+\frac{1}{2}(\mathbf{W}_0^T\tilde{\mathbf{W}}_0+\mathbf{W}_{N_y}^T\tilde{\mathbf{W}}_{N_y})], \\ \|\mathbf{W}\|^2 &= E_\Delta. \end{aligned} \tag{2.3.8}$$

With an elementary calculation it can now be proved that space-discretization of the transformed PDE

$$\begin{aligned} v_t &= -\frac{1}{2}v\phi^{-1/2}v_y-\frac{1}{2}(v^2\phi^{-1/2})_y-\phi^{1/2}\phi_y, \\ \phi_t &= -(\phi^{1/2}v)_y, \end{aligned} \tag{2.3.9}$$

in the same way as described in Example 2.2.1, yields a semi-discrete approximation $\dot{\mathbf{W}}=\mathbf{F}(\mathbf{W})$ which is conservative with respect to the given energy norm. This particular ODE system reads

$$\begin{aligned}
\dot{\Phi}_0 &= -\frac{1}{\Delta y}\,\Phi_1^{1/2} V_1, \\
\dot{V}_k &= D_k - \frac{1}{2\Delta y}\,\Phi_k^{1/2}(\Phi_{k+1}-\Phi_{k-1}), \quad k = 1(1)N_y-1, \\
\dot{\Phi}_k &= -\frac{1}{2\Delta y}\,(\Phi_{k+1}^{1/2}V_{k+1}-\Phi_{k-1}^{1/2}V_{k-1}), \quad k = 1(1)N_y-1, \\
\dot{\Phi}_{N_y} &= \frac{1}{\Delta y}\,\Phi_{N_y-1}^{1/2}V_{N_y-1},
\end{aligned} \tag{2.3.10}$$

where $D_k$ is given by

$$D_k = -\frac{1}{4\Delta y}\,[V_k\Phi_k^{-1/2}(V_{k+1}-V_{k-1})+(V_{k+1}^2\,\Phi_{k+1}^{-1/2}-V_{k-1}^2\,\Phi_{k-1}^{-1/2})].$$

See [6] for more details. □

*Open boundaries and stability..* For a dependent variable a boundary is called open if no boundary condition is present. Open boundaries normally lead to inaccuracies (see the discussion in Example 2.3.1) due to the use of one-sided difference approximations which tend to be less accurate than symmetric approximations, at least for problems with smooth solutions. An additional difficulty with open boundaries is that the use of higher order one-sided approximations may turn a stable approximation into an unstable one. We will illustrate the nuisance of unstable boundary conditions for the just described PDE (2.3.9). Recall that $v$ is zero at the boundaries $y=0$, $D$, while $\phi$ is not prescribed. Hence for $\phi$ the boundary is open.

Example 2.3.4. As shown above the semi-discrete approximation (2.3.10) conserves the semi-discrete energy $E_\Delta$. Let us apply the usual one-sided second-order difference approximation at the boundaries, instead of (2.2.4), for approximating $(\sqrt{\phi}\,v)_y$. The first and last equation of (2.3.10) are then replaced by

$$\begin{aligned}
\dot{\Phi}_0 &= \frac{-1}{2\Delta y}\,(4\Phi_1^{1/2}V_1-\Phi_2^{1/2}V_2), \\
\dot{\Phi}_{N_y} &= \frac{1}{2\Delta y}\,(4\Phi_{N_y-1}^{1/2}V_{N_y-1}-\Phi_{N_y-2}^{1/2}V_{N_y-2}).
\end{aligned}$$

It is straightforward to prove that now

$$\dot{E}_\Delta = \frac{1}{4\Delta y}\,[(3\Phi_{N_y-1}^{1/2}V_{N_y-1}-\Phi_{N_y-2}^{1/2}V_{N_y-2})\Phi_{N_y}+(\Phi_2^{1/2}V_2-3\Phi_1^{1/2}V_1)\Phi_0].$$

Hence, $\dot{E}_\Delta \neq 0$ and the energy will increase as soon as the right hand side expression becomes positive. From then on we have to face severe instabilities the origin of which lies in the use of the second order one-sided differences at the boundary points. Note that $E_\Delta$ is again constant for the (physically unrealistic) boundary condition $\Phi=0$. □

*Curved boundaries.* The domain $\Omega$ of our first model in subsection 2.1 is the rectangle (2.1.3). Finite differences are easily implemented on such simple domains. In applications, however, $\Omega$ may be rather irregular leading to curved boundaries. For example, part of $\partial\Omega$ may consist of a curved coastline. It shall be clear that such a domain is poorly approximated by an orthogonal grid. This poor representation of $\Omega$ will cause larger approximation errors $\alpha_\Delta$ near the boundary $\partial\Omega$ than in the interior of the domain. These larger approximation errors, in turn, may increase the space approximation error $\eta_\Delta$ over a considerable part of $\Omega$, if not the whole of $\Omega$. To some extent it depends on the application whether this specific error increase is unacceptable. For in many practical computations the physical data, for example at a boundary, already contain inaccuracies which overshadow numerical errors due to a bad boundary representation or other numerical errors. In such applications one is satisfied with low accuracy finite difference models and orthogonal grids are still useful.

A cure for the above mentioned boundary inaccuracies is the use of curvilinear grids. Gridlines then can be chosen coincident with boundaries leading to a significantly more accurate discretization of the domain $\Omega$. Clearly, the use of curvilinear grids does complicate the implementation of finite differences. Already the creation of $\Omega_\Delta$ itself may become very cumbersome. For that reason one has developed so-called grid or mesh generators, computer programs which assist the engineer in setting up nonrectangular grids without irregularities such as too small corners between grid lines.

Loosely speaking, the derivation of approximations for partial derivatives on nonrectangular grid-elements are always based on a (local or global) coordinate transformation $T$ which maps the nonrectangular grid-element onto a rectangular one where standard approximations are applicable. The effect of $T$ is that one performs a standard space-discretization of a transformed PDE on a rectangular grid-element. The choice of $T$ influences the accuracy of the discretization of course. DEKKER [4] has developed a method which minimizes the errors of derivative approximations on nonrectangular grid-elements. In case an explicit parametrization of the curvilinear grid-lines in the $(x,y)$-plane is available a suitable transformation is easily found [25]. The next example illustrates this.

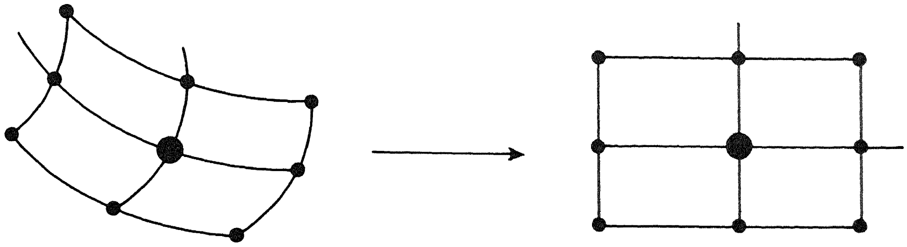

FIGURE 2.3.4. A curvilinear and a square grid-element

EXAMPLE 2.3.5. Consider Figure 2.3.4. Suppose that in the $(x,y)$-domain $\Omega$ of the PDE curvilinear gridlines $X = X(x,y)$, $Y = Y(x,y)$ have been defined. We seek a transformation $T$ which maps the curvilinear grid onto a square grid in the $(X, Y)$-plane with grid distance $\Delta$. Hence $T$ is defined by

$$X(x,y) = j\Delta, \quad Y(x,y) = k\Delta,$$

$j$ and $k$ being gridindices in the square grid. Supposing that $X$ and $Y$ are differentiable, it holds that

$$\frac{\partial}{\partial x} = \frac{\partial X}{\partial x}\frac{\partial}{\partial X} + \frac{\partial Y}{\partial x}\frac{\partial}{\partial Y}, \quad \frac{\partial}{\partial y} = \frac{\partial X}{\partial y}\frac{\partial}{\partial X} + \frac{\partial Y}{\partial y}\frac{\partial}{\partial Y},$$

where $X_x$, $X_y$, $Y_x$ and $Y_y$ are explicitly known. Because of the transformation $T$ standard differences can be used for approximating $\partial/\partial X$ and $\partial/\partial Y$ in the $(X, Y)$-plane.

In case no explicit parameterization $X(x,y)$, $Y(x,y)$ is available, one can approximate $X_x, \dots, Y_y$ on the curvilinear grid in the $(x,y)$-plane. There are various possibilities to do so [4], [25]. Of course, in applications one mostly has to make this approximation. □

REMARK 2.3.6. By nature the finite element method leads to an easier way of handling curved boundaries. Differently from the field of hyperbolic PDEs, in recent years the finite element method has become quite popular for parabolic equations. PRAAGMAN [35] has implemented the finite element method for the shallow water equations and reports satisfactory results. More research is needed however for more definite conclusions how finite differences and finite elements compare to each other in the extensive and diverse field of hyperbolic equations, such as in shallow water applications. □

## 3. Time integrators

In this section we start with the time-continuous, semi-discrete system (2.2.6). In principle, any ODE solver can now be applied to this equation in order to obtain a numerical approximation to the solution of (2.2.6). Thus, using an initial value problem solver from a program library such as NAG or IMSL will provide us with a numerical solution of the SWEs. However, the costs both in terms of computation time and of storage will be enormous. The reason is that such library programs, being designed as general purpose methods, do not take into account the two characteristic properties of semi-discrete hyperbolic systems, in particular the semi-discrete SWEs:

A. *The large number of component equations in the system* (2.2.6) (3 *times the number of spatial grid points used in the semidiscretization*).

B. *The large, almost imaginary eigenvalue interval of the Jacobian matrix* $\partial F/\partial W$ *of the right-hand side in* (2.2.6)

Property A is obviously responsible for the excessive storage requirements when applying a general purpose method, at the same time implying that each integration step is relatively expensive. Property B causes the system (2.2.6) to be marginally stable; it is therefore expected that a numerical approximation to (2.2.6) will easily become unstable unless either small integration steps or special numerical approximations are used.

It is the purpose of this section to give a survey of possible integration techniques for solving (2.2.6) that take into account the properties A and B.

### *3.1. Runge-Kutta methods*

Let $\mathbf{W}_n$, $n=0,1,2,\ldots$ denote numerical approximations to the exact solution $\mathbf{W}(t)$ of (2.2.6) at $t_n=t_0+n\Delta t$, $\Delta t$ being the integration step. Then an important class of numerical approximations to (2.2.6) is given by

$$\begin{aligned} \mathbf{W}_{n+1}^{(j)} &= \mathbf{W}_n+\Delta t\sum_{l=1}^{m} a_{j,l}\mathbf{F}_{n+1}^{(l)}, \\ \mathbf{F}_{n+1}^{(l)} &:= \mathbf{F}(t_{n+1}^{(l)},\mathbf{W}_{n+1}^{(l)}),\ \ j=1,2,\ \ldots\ ,m, \\ \mathbf{W}_{n+1} &= \mathbf{W}_n+\Delta t\sum_{l=1}^{m} b_l\mathbf{F}_{n+1}^{(l)}. \end{aligned} \tag{3.1.1}$$

This method is called an *m-stage Runge-Kutta method.* The Runge-Kutta parameters $a_{j,l}$ and $b_l$ are determined by accuracy and stability conditions. The intermediate points $t_{n+1}^{(j)}$ are usually defined by

$$t_{n+1}^{(j)} = t_n+\Delta t\sum_{l=1}^{m} a_{j,l}. \tag{3.1.2}$$

In this case, the $\mathbf{W}_{n+1}^{(j)}$ are approximations to $\mathbf{W}(t_{n+1}^{(j)})$. We will assume that (3.1.2) is always satisfied.

Example 3.1.1. The most famous (and at the same time an appropriate time integrator for the SWEs) is given by (KUTTA [26])

$$\begin{aligned}
\mathbf{W}^{(1)}_{n+1} &= \mathbf{W}_n, \quad \mathbf{W}^{(2)}_{n+1} = \mathbf{W}_n + \tfrac{1}{2}\Delta t \mathbf{F}^{(1)}_{n+1}, \\
\mathbf{W}^{(3)}_{n+1} &= \mathbf{W}_n + \tfrac{1}{2}\Delta t \mathbf{F}^{(2)}_{n+1}, \quad \mathbf{W}^{(4)}_{n+1} = \mathbf{W}_n + \Delta t \mathbf{F}^{(3)}_{n+1}, \qquad (3.1.3) \\
\mathbf{W}_{n+1} &= \mathbf{W}_n + \tfrac{1}{6}\Delta t[\mathbf{F}^{(1)}_{n+1} + 2\mathbf{F}^{(2)}_{n+1} + 2\mathbf{F}^{(3)}_{n+1} + \mathbf{F}^{(4)}_{n+1}].
\end{aligned}$$

If $\mathbf{F}(t, \mathbf{W})$ is sufficiently smooth it can be proved that (see e.g. [27])

$$\mathbf{W}_{n+1} - \mathbf{W}(t_{n+1}) = O((\Delta t)^4) \text{ as } \Delta t \to 0, \ t_{n+1} \text{ constant.}$$

The method is said to be of order 4. Notice that the $\mathbf{W}^{(j)}_{n+1}$ are defined by (3.1.3) explicitly. In the particular case of the SWEs, this method when implemented on a computer requires 3 arrays for storing the $\mathbf{W}^{(j)}_{n+1}$ and $\mathbf{F}^{(j)}_{n+1}$ during the computation of an integration step. □

In Table 3.1.1 we present a few numerical results obtained by this method when applied to model 1 (equations (2.1.1)-(2.1.4)) in the semi-discrete form (2.2.6). These results refer to the relative height deviation defined by

$$\epsilon := \frac{h - h_{ref}}{\max|h_{ref} - \bar{h}_{ref}|},$$

with $h_{ref}$ a sufficiently accurate reference solution and $\bar{h}_{ref}$ the mean value of $h_{ref}$. For a few grid points we have listed the accuracy expressed in terms of the number of correct significant digits, i.e.

$$sd := -{}^{10}\log(\max|\epsilon|).$$

TABLE 3.1.1. Model 1 with $\Delta x = \Delta y = 100 * 10^3$ m and $t_{end} = 48 * 3600$ sec.

| Grid point | $\Delta t = 1200$ | $\Delta t = 600$ | $\Delta t = 300$ |
|---|---|---|---|
| (48,00) | 2.80 | 3.77 | 5.01 |
| (48,12) | 2.37 | 3.35 | 4.68 |
| (48,24) | 2.48 | 3.28 | 4.54 |
| (48,36) | 2.41 | 3.34 | 4.53 |

The reference solution for (2.2.6) was obtained by using a high order method with extremely small integration steps.

These results clearly reflect the fourth order behaviour of the Runge-Kutta integrator, i.e. on halving the integration step the *sd*-value should increase by $4\log 2 \approx 1.2$. For $\Delta t \leqslant 600$ this (asymptotic) order property is shown.

In order to analyse the stability characteristics of a numerical method one often uses the *linear* equation

$$\dot{\mathbf{W}}(t) = \lambda \mathbf{W}(t), \quad \lambda \in \Lambda_n \tag{3.1.4}$$

as a stability test model. Here, $\Lambda_n$ denotes the eigenvalue spectrum of $\partial \mathbf{F}/\partial \mathbf{W}$ at $t_n$. Applying (3.1.1) to (3.1.4) leads to the relation

$$\mathbf{W}_{n+1} = R(\lambda \Delta t)\mathbf{W}_n, \quad \lambda \in \Lambda_n, \tag{3.1.5}$$

where $R(z)$ is a rational function in $z$ the coefficients of which are expressions in terms of the parameters $a_{j,l}$ and $b_l$. It can be shown [39] that

$$R(z) = \frac{\det[I - Az + \mathbf{e}\mathbf{b}^T z]}{\det[I - Az]}, \quad \mathbf{e} = [1, \ldots, 1]^T, \tag{3.1.6}$$

where $A$ is the matrix $(a_{j,l})$; $j,l = 1, \ldots, m$, and $\mathbf{b}$ is the vector $(b_1, \ldots, b_m)^T$ (observe that the Runge-Kutta method is completely defined by the matrix $A$ and the vector $\mathbf{b}$). The *stability region* $\mathbb{S}$ of a Runge-Kutta method is defined as the region in the complex $z$-plane where $|R(z)| \leqslant 1$. The method is said to be stable for a given problem at $t_n$ if $\Delta t \lambda$ lies in the stability region. Notice that the stability region $\mathbb{S}$ is completely defined by the numerical method without reference to the particular problem to be solved. Evidently, if a method is stable at $t_n$, the numerical solutions of the test equations (3.1.4) satisfy the condition

$$\|\mathbf{W}_{n+1}\| \leqslant \|\mathbf{W}_n\|. \tag{3.1.7}$$

In many cases, the (linear) stability condition $\Delta t \Lambda_n \subset \mathbb{S}$ leads to satisfactory numerical solutions of nonlinear problems. But we should bear in mind that the above given analysis is based on the test equations (3.1.4) and should be applied with care to more general problems. For a discussion of nonlinear stability analysis we refer to Dekker and Verwer [6].

Adopting $\Delta t \Lambda_n \subset \mathbb{S}$ as the stability condition it follows from property B that the SWEs require numerical methods the stability regions of which contain a relatively large imaginary interval $[-i\beta, i\beta]$ (notice that $\mathbb{S}$ is symmetric with respect to the real axis). For *implicit* methods this is easily achieved. However, from a practical point of view we are mainly interested in *explicit* Runge-Kutta methods ($a_{j,l} = 0$ for $j \geqslant l$) which turn out to have rather modest $\beta$-values.

Example 3.1.2. Here we mention some well-known explicit Runge-Kutta methods and the corresponding stability function $R(z)$, which reduces to a polynomial for these explicit methods. Also the imaginary stability boundary $\beta$, the number of stages $m$ and the order $p$ are given. For the coefficients $a_{j,l}$ and $b_l$, defining the Runge-Kutta schemes, we refer to e.g. [27]

*method of Euler*; $R(z) = 1 + z$, $\beta = 0$, $m = 1$, $p = 1$,

*method of Runge*; $R(z) = 1 + z + \frac{1}{2}z^2$, $\beta = 0$, $m = 2$, $p = 2$,

*method of Heun*; $R(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3$, $\beta = \sqrt{3}$, $m = 3$, $p = 3$,

*method of Kutta*; $R(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4$, $\beta = 2\sqrt{2}$, $m = 4$, $p = 4$.

Thus, the methods using 3 or 4 stages in this example possess a nonvanishing stability boundary $\beta$ on the imaginary axis. The corresponding stability condition $\Delta t \Lambda_n \subset \mathbb{S}$ can be written as

$$\Delta t \leqslant \frac{\beta}{S}, \quad S := \text{spectral radius of } \partial \mathbf{F}/\partial \mathbf{W}, \tag{3.1.8}$$

provided that $\Lambda_n$ is purely imaginary. Since in the case of the SWEs, $S = O(\Delta^{-1})$, $\Delta$ being the mesh size on the spatial grid, condition (3.1.8) allows us to use grid parameters $\Delta$ and $\Delta t$ of comparable magnitude as $\Delta$, $\Delta t \rightarrow 0$. However, in an actual computation the order constant in $S = O(\Delta^{-1})$ may be large (e.g. in computations with large values for the depth function); also, it is often allowed to use $\Delta t$-values which are large compared with $\Delta$ (see [38, p. 214]). In such cases, (3.1.8) may impose a severe limitation on the integration step $\Delta t$, just *for the sake of stability* and *not for the sake of accuracy*. It is therefore of interest to look for (explicit) Runge-Kutta methods with a large stability boundary $\beta$ on the imaginary axis, that is to look for methods possessing a (so-called) stability polynomial $R(z) = P_m(z)$ of the form (3.1.6) which assumes values on the unit disc on the largest possible interval $[-i\beta, i\beta]$. Before giving results for this minimax problem we give a theorem which relates the order of the method to the specific form of $P_m(z)$.

THEOREM 3.1.1. *If the Runge-Kutta method* (3.1.1) *is of order p then*

$$\frac{d^j P_m}{dz^j}(0) = 1$$

*for* $j = 0, 1, \dots, p$. *($P_m(z)$ is called consistent of order p.)* □

THEOREM 3.1.2. *If* $p \geqslant 1$ *and* $m \geqslant 2$ *then* $\beta \leqslant m - 1$ *(for m odd see* [17], *for m even see* [40]*).* □

THEOREM 3.1.3. *If* $p = 1,2$ *and* $m = 2k + 1$, $k = 0, 1, 2, \dots$, *then*

$$P_m(z) = T_k\left[1 + \frac{z^2}{2k^2}\right] + \frac{z}{k}\left[1 + \frac{z^2}{4k^2}\right] U_{k-1}\left[1 + \frac{z^2}{2k^2}\right] \tag{3.1.9}$$

*solves the minimax problem and the corresponding* $\beta$ *value is the largest possible, i.e.* $\beta = m - 1$ [17]. □

This result has recently been extended [37]; now a polynomial is available for all values of $m$:

THEOREM 3.1.4. *For* $p = 1$ *the polynomial*

$$P_m(z) = i^{m-1}T_{m-1}\left[\frac{z}{i(m-1)}\right] + \frac{1}{2}i^m\left\{T_m\left[\frac{z}{i(m-1)}\right] - T_{m-2}\left[\frac{z}{i(m-1)}\right]\right\} \tag{3.1.10}$$

*is the optimal polynomial and has $\beta = m-1$.* □

THEOREM 3.1.5. *If $p=2,3,4$ and $m=4$ then*

$$P_4(z) = 1+z+\frac{1}{2}z^2+\frac{1}{6}z^3+\frac{1}{24}z^4 \tag{3.1.11}$$

*solves the minimax problem and $\beta = 2\sqrt{2}$* [17]. □

In selecting a stability polynomial one should take into account that the larger $m$ the more expensive an integration step. Hence, using large $m$-values in order to increase $\beta$, has to be paid for by $m$ right-hand side evaluations. This suggests considering the effective (or scaled) stability boundary $\beta/m$. From Theorem 3.1.2 it follows that $\beta/m \leqslant 1-1/m$ so that it hardly pays to use a large value for $m$. In this connection, we observe that the *fourth order consistent* polynomial $P_4(z)$ given in Theorem 3.1.5 has an effective stability boundary $\beta/m = ½\sqrt{2}$ which is already more than 70% of the asymptotic value of the *second order consistent* minimax polynomial of degree infinity. Therefore, the polynomial (3.1.11) is recommended as a stability polynomial if one decides to use an explicit Runge-Kutta method for the SWEs.

The next step is the choice of a Runge-Kutta method possessing (3.1.11) as its stability polynomial. An obvious choice is the fourth order method of Kutta (3.1.3) (see also Example 3.1.2), and in fact PRAAGMAN [35] used this method in solving the SWEs. An alternative might be the methods of MERLUZZI and BROSILOW [31] who (following ideas of STETTER [39]) developed methods which allow for global error estimation with low extra costs.

### *3.2. Linear multistep methods*

A second important class of numerical approximations to the ODE (2.2.6) are the *linear k-step methods* defined by

$$\sum_{l=0}^{k}[a_l\mathbf{W}_{n+1-l} - \Delta t b_l \mathbf{F}_{n+1-l}] = 0, \tag{3.2.1}$$

$$\mathbf{F}_{n+1-l} := \mathbf{F}(t_{n+1-l}, \mathbf{W}_{n+1-l}),$$

where the coefficients $a_l$ and $b_l$ are determined by accuracy and stability conditions.

As in the case of Runge-Kutta methods we are particularly interested in explicit methods, i.e. $b_0 = 0$. However, *implicit* methods are also important for us as a starting point in constructing special predictor-corrector methods (see subsection 3.2.2).

*3.2.1. Explicit linear multistep methods.* Just as for Runge-Kutta methods, stability requirements for linear multistep methods are obtained by applying (3.2.1) to the test equations (3.1.4). This yields a relation of the form

$$[\rho(E)-\lambda\Delta t\sigma(E)]\mathbf{W}_{n+1-k} = 0, \quad \lambda\in\Lambda_n, \quad n+1\geqslant k, \tag{3.2.2}$$

where $E$ is the shift operator defined by $E\mathbf{W}_n=\mathbf{W}_{n+1}$ and where $\{\rho,\sigma\}$ are the so-called *characteristic polynomials* defined by

$$\rho(\zeta) := \sum_{l=0}^{k} a_l\zeta^{k-l}; \ \sigma(\zeta) := \sum_{l=0}^{k} b_l\zeta^{k-l}, \quad b_0=0. \tag{3.2.3}$$

The *stability region* $\mathbb{S}$ is now defined as the region in the complex $z$-plane where the *characteristic function* $\pi(\zeta,z):=\rho(\zeta)-z\sigma(\zeta)$ has all its roots $\zeta$ on the unit disc.

The stability condition, widely adopted in practical computations, reads $\Delta t\Lambda_n\subset\mathbb{S}$ so that we are again faced with the problem to construct a method the stability region $\mathbb{S}$ of which contains a large imaginary interval $[-i\beta,i\beta]$.

EXAMPLE 3.2.1. As an example of explicit linear multistep methods we mention the extensively used $k$-step *Adams-Bashforth* methods, which are characterized by their $\rho$-polynomial possessing the form $\rho(\zeta)=\zeta^k-\zeta^{k-1}$. The $\sigma$-polynomials may be found in [27]. For $k=2,3,4$ (yielding methods of order $p=2,3,4$, respectively) we have the respective imaginary stability boundaries $\beta=0$, $\beta=.72$ and $\beta=.43$. □

The stability boundaries given in this example are at the same time the effective boundaries because each integration step requires just one **F**-evaluation. A comparison with the results obtained for Runge-Kutta methods of the same order reveals that (cf. Example 3.1.2) the Adams-Bashforth method of order 3 has a larger effective stability interval along the imaginary axis, but the fourth order method does not. Both second order methods have $\beta=0$.

The maximization of the imaginary stability interval of explicit multistep method has been studied in JELTSCH and NEVANLINNA [24]:

THEOREM 3.2.1.

(a) *The imaginary stability boundary $\beta$ of an explicit linear multistep method cannot exceed* 1.

(b) *Let $r\in[0,1)$ and $k\in\{2,3,4\}$ be given. Then there exists an explicit linear $k$-step method of order $p=k$ with $\beta=r$.* □

EXAMPLE 3.2.2.

*Leap-frog* method: $k=2$

$$\rho(\zeta) = \zeta^2-1, \quad \sigma(\zeta)=2\zeta, \quad \beta=1, \quad p=2$$

*Jeltsch-Nevanlinna methods:* $k=3,4$ $(\epsilon>0)$

$$\rho(\zeta) = (\zeta-1)(\zeta+1-\tfrac{2}{3}\epsilon)(\zeta-1+\epsilon),$$

$$\sigma(\zeta)=2\zeta(\zeta-1)+\tfrac{2}{3}\epsilon(2\zeta+1)+\frac{1}{18}\epsilon^2(\zeta^2-8\zeta+5),\ \ \beta=1-O(\epsilon),\ \ p=3.$$

$$\rho(\zeta) = (\zeta^2-1)(\zeta^2-2\frac{3-4\epsilon}{3-\epsilon}\,\zeta+1),\ \ \sigma(\zeta)=\frac{6}{3-\epsilon}\,\zeta(\zeta^2+2(\epsilon-1)\zeta+1),$$

$$\beta=1-O(\epsilon),\ \ p=4.\quad \Box$$

In Table 3.2.1 results obtained by the leap-frog method are listed for problem (2.2.6) corresponding to model 1.

TABLE 3.2.1. Model 1 with $\Delta x=\Delta y=100*10^3$ m and $t_{end}=48*3600$ sec

| Grid point | $\Delta t=400$ | $\Delta t=225$ | $\Delta t=75$ |
|---|---|---|---|
| (48,00) | 1.56 | 2.26 | 3.19 |
| (48,12) | 2.30 | 2.41 | 3.30 |
| (48,24) | 1.95 | 2.27 | 3.16 |
| (48,36) | 2.44 | 2.79 | 3.31 |

With the exception of the last grid point, the second order behavior is clearly shown for $\Delta t=75$. Note that the results of the fourth order *RK*-method (cf. Table 3.1.1) are much more accurate.

*3.2.2. Predictor-corrector methods.* The rather modest stability results obtained for explicit Runge-Kutta and linear multistep methods lead us to consider *implicit* methods. In particular we will study implicit linear multistep methods because of their simple structure. Thus, let $\mathbf{W}_{n+1}$ be defined by (3.2.1) with $b_0\neq0$, or briefly

$$\mathbf{W}_{n+1}-b_0\Delta t\mathbf{F}(t_{n+1},\mathbf{W}_{n+1}) = \Sigma_n, \tag{3.2.4}$$

where $\Sigma_n$ is a linear combination of $\mathbf{W}_j$ and $\mathbf{F}_j$ values with $j\leqslant n$. In order to solve this implicit relation we employ a *predictor-corrector method.*

Following [21] we define the $m$-point iteration scheme

$$\mathbf{W}_{n+1}^{(j)} = \sum_{l=1}^{j}[\mu_{jl}\mathbf{W}_{n+1}^{(l-1)}+\bar{\mu}_{jl}\Delta t\mathbf{F}(t_{n+1},\mathbf{W}_{n+1}^{(l-1)})]+\lambda_j\Sigma_n, \tag{3.2.5}$$

$$j=1,\ \dots\ ,m$$

where $\mathbf{W}_{n+1}^{(0)}$ is obtained by a suitable predictor method (e.g. an explicit linear multistep method) and $\mathbf{W}_{n+1}^{(m)}$ is accepted as an approximation to the exact solution of (3.2.4). By requiring that the $j$-th row sum of the matrices $M=(\mu_{jl})$ and $\overline{M}=(\bar{\mu}_{jl})$ are respectively given by $1-\lambda_j$ and $b_0\lambda_j$, we achieve that if

$\mathbf{W}_{n+1}^{(j)} \rightarrow \mathbf{W}$ as $j \rightarrow \infty$ then $\mathbf{W}$ equals the exact solution of (3.2.4). In doing so, (3.2.4) may be considered as the *corrector equation* and (3.2.5) as an correction iteration. The scheme (3.2.5) is said to be consistent with (3.2.4).

The iteration scheme (3.2.5) is conveniently characterized by the iteration polynomials

$$P_0(z) = 1, \quad P_j(z) = \sum_{l=1}^{j} [\mu_{jl} + \tilde{\mu}_{jl} z] P_{l-1}(z), \qquad (3.2.6)$$
$$j = 1, 2, \ldots, m.$$

The consistency condition implies that $P_j(1/b_0) = 1$ for all $j$. The following theorem determines the accuracy of the predictor-corrector method [21]:

THEOREM 3.2.2. *Let* $\mathbf{W}_j = \mathbf{W}(t_j)$ *for* $j \leqslant n$, *then*

$$\mathbf{W}_{n+1}^{(m)} - \mathbf{W}(t_{n+1}) = [I - P_m(Z)]\,[\mathbf{W}_{n+1} - \mathbf{W}(t_{n+1})] + P_m(Z)[\mathbf{W}_{n+1}^{(0)} - \mathbf{W}(t_{n+1})] + O(\Delta t^q),$$
$$q \geqslant 3 + 2min\{\tilde{p}, p\}, \quad Z := \Delta t \frac{\partial \mathbf{F}}{\partial \mathbf{W}}(t_{n+1}, \mathbf{W}_{n+1})$$

*where p and* $\tilde{p}$ *are the orders of accuracy of the corrector and the predictor respectively.* □

This theorem expresses the (local) error of the predictor-corrector method in terms of those of the predictor and the corrector plus higher order terms. It clearly shows that the solution of (3.2.4) is approximated better as $\|P_m(Z)\|$ is smaller. Furthermore, the theorem gives us the exact order of the method: let $P_m(z)$ have a zero at $z = 0$ of multiplicity $r$, then the predictor-corrector method is of order $p^* = \min\{p, \tilde{p} + r, 2 + 2\min\{\tilde{p}, p\}\}$.

Next, we consider the stability of (3.2.5). Assuming that we can find a predictor and an iteration polynomial $P_m(z)$ such that (3.2.4) is solved with sufficient accuracy, the stability properties are determined by those of (3.2.4). The following theorem is known:

THEOREM 3.2.3.

(a) *Only for* $p \leqslant 2$ *there exist linear multistep methods with an infinite imaginary interval of stability* [22].

(b) *For* $p > 2$ *the imaginary interval of stability cannot exceed* $[-i\sqrt{3}, i\sqrt{3}]$ [5],[23]. □

EXAMPLE 3.2.3. A few methods possessing an infinite imaginary interval of stability are [27]: *implicit Euler* ($k = 1$, $p = 1$), *the trapezoidal-rule* ($k = 1$, $p = 2$) and the *backward differentiation method* ($k = 2$, $p = 2$). Within the class of methods with $p > 2$, the fourth order, 2-step *Milne-Simpson method* has the maximal attainable imaginary stability boundary $\beta = \sqrt{3}$. □

Theorem 3.2.3 indicates that we should be content with a first or second order corrector in our attempt to construct predictor-corrector methods with large imaginary stability boundaries. In particular, the implicit Euler and the second order backward differentiation method are recommendable because of their strong damping of higher frequencies (which are easily introduced by round-off errors).

Let us now consider the imaginary stability boundary $\beta$ of the complete predictor-corrector method. We will do this by relating $\beta$ to the *real* stability boundary $\beta_{real}$ of the method. Since the derivation of $\beta_{real}$ has been studied in some detail [21] we can avoid a lot of tedious computations. The following theorem is easily proved.

THEOREM 3.2.4. *Let $\beta_{real}(m)$ be the real stability boundary of the predictor-corrector pair using the iteration polynomial $Q_m(z)$. Then this predictor-corrector pair using the iteration polynomial $P_{2m}(z)=Q_m(b_0z^2)$ has the imaginary stability boundary $\beta=\sqrt{\beta_{real}(m)/b_0}$.* □

EXAMPLE 3.2.4. In [21] a predictor-corrector pair consisting of the predictor

$$\mathbf{W}_{n+1} = 2\mathbf{W}_n-\mathbf{W}_{n-1}$$

and the second order backward differentiation corrector (see Example 3.2.3) is considered for which iteration polynomials are constructed such that $\beta_{real}(m)\uparrow 1.37m^2$ as $m\rightarrow\infty$. Hence, by Theorem 3.2.4 we can construct a $2m$-stage predictor-corrector method with $\beta\uparrow 1.43m$. Effectively, however, we obtain the value .72 for $m$ sufficiently large. □

The methods suggested by Theorem 3.2.4 are not optimal. In order to get some insight into how good or poor these methods are we have done a numerical search for the optimal iteration polynomial $P_2(z)$ in the case of the predictor-corrector pair mentioned in Example 3.2.4. We found

$$P_2(z) = 1.0-.408z+.272z^2, \quad \beta=1.97.$$

### 3.3. Multigrid methods

A multigrid method can be used for solving the implicit relations obtained when an implicit Runge-Kutta or multistep method is applied to (2.2.6). Let us consider an implicit $k$-step method which requires the solution of equation (3.2.4) in each integration step. In the multigrid technique we do not only consider this equation but we define on a sequence of successively coarser grids a similar equation. Thus, we have a sequence of problems of the form

$$\mathbf{W}_{n+1}-b_0\Delta t\mathbf{F}(t_{n+1},\mathbf{W}_{n+1}) = \Sigma_n \tag{3.3.1}$$

where the number of components in $\mathbf{W}_{n+1}$, $\mathbf{F}$ and $\Sigma_n$ correspond to the number of grid points in the grid considered. In fact, in the more advanced applications of the multigrid method the right-hand side vectors $\Sigma_n$ are modified except for the $\Sigma_n$ corresponding to the finest grid. We will not discuss these modification but refer to the literature (e.g. [1], [15]). By first solving the coarsest grid problem one may construct a rather good initial approximation to the solution of the next finer grid problem, and so on. Usually, a linear interpolation procedure is applied. These approximations can be improved by adding certain correction terms for which we again refer to the literature. As a result we obtain initial approximations which differ from the solution of (3.3.1) only in the high frequency range. Thus, in order to solve the problems (3.3.1) one may use any iteration scheme (usually called *relaxation method*) that takes advantage of the fact that the initial approximation is incorrect only in the high frequency range. For functions $\mathbf{F}(t, \mathbf{W})$ the Jacobian matrix of which has a *real spectrum with an orthogonal eigensystem*, Gauss-Seidel relaxation or Incomplete LU relaxation [44] are widely used. Chebyshev relaxation (advocated in [20]) may be another possibility, particularly when vectorcomputers are to be used (cf. subsection 4.1). However, in the present case we do not have a Jacobian matrix with a real spectrum, but with an *imaginary spectrum* instead. For such problems there is hardly any experience.

Let us apply the iteration scheme (3.2.5) to (3.3.1), that is $\Sigma_n$ is replaced by $\tilde{\Sigma}_n$. Omitting higher order terms in $\Delta t$ we deduce from Theorem 3.2.2

$$\mathbf{W}_{n+1} - \mathbf{W}_{n+1}^{(m)} = P_m(Z)[\mathbf{W}_{n+1} - \mathbf{W}_{n+1}^{(0)}].$$

Since $\mathbf{W}_{n+1} - \mathbf{W}_{n+1}^{(0)}$ is supposed to contain only high frequency components, we should look for polynomials $P_m(z)$ such that the matrix $P_m(Z) = P_m(\Delta t \partial \mathbf{F}/\partial \mathbf{W})$ damps all high frequencies. In the case of the SWEs, the eigenvectors of $\partial \mathbf{F}/\partial \mathbf{W}$ corresponding to the eigenvalues with large imaginary parts present the high frequencies so that we should construct polynomials $P_m(z)$ satisfying the condition $P_m(1/b_0) = 1$, such that $|P_m(iy)|$ is as small as possible on an interval $a \leqslant |y| \leqslant b$. Here, $a = \alpha \Delta t S$ and $b = \Delta t S$, $\alpha$ being some parameter $<1$ (say $\alpha = ½$), and $S$ is the spectral radius of $\partial \mathbf{F}/\partial \mathbf{W}$. As far as the authors know this minimax problem has not yet been solved for general $m$. For $m = 1$ we straightforwardly find that

$$P_1(z) = \frac{1 + z/b_0 b^2}{1 + 1/b_0^2 b^2} \quad \text{with} \quad \max_{[ia,ib]} |P_1(z)| = \frac{1}{\sqrt{1 + 1/b_0^2 b^2}} \tag{3.3.2}$$

is the minimax polynomial for all $0 \leqslant a \leqslant b$. For $m = 2$ and $m = 4$ a numerical computation for several intervals $[ia, ib]$ resulted in minimax polynomials with extremely small values for the odd degree coefficients. This suggests considering the polynomials $P_m(z) = Q_{m/2}(z^2)$ with $m$ even. The minimax problem on $[ia, ib]$, i.e. $a^2 \leqslant -z^2 \leqslant b^2$, is solved by Chebyshev polynomials:

$$P_m(z) = \frac{T_{m/2}\left[\dfrac{b^2 + a^2 + 2z^2}{b^2 - a^2}\right]}{T_{m/2}\left[\dfrac{b^2 + a^2 + 2/b_0^2}{b^2 - a^2}\right]}, \quad m \text{ even}, \tag{3.3.3}$$

from which the damping factor immediately follows.

EXAMPLE 3.3.1. Let (3.3.1) correspond to the second order backward differentiation method ($b_0 = 2/3$) and choose $\alpha = ½$, i.e. $a = b/2$. Then the polynomials (3.3.3.) damp the high frequencies by a factor

$$1/T_{m/2}\left[\frac{5}{3}+\frac{6}{b^2}\right].$$

In Table 3.3.1 a few values are listed.

TABLE 3.3.1. Damping factors obtained by (3.3.2) and (3.3.3) for $a = b/2$ and $b_0 = 2/3$

| $b$ | $m=1$ | $m=2$ | $m=4$ | $m=6$ | $m=8$ | $m=10$ | $m=12$ |
|---|---|---|---|---|---|---|---|
| 1 | .83 | .13 | .009 | $<10^{-3}$ | $\sim 0$ | $\sim 0$ | $\sim 0$ |
| 2 | .95 | .32 | .05 | .009 | .001 | $\sim 0$ | $\sim 0$ |
| 4 | .986 | .49 | .13 | .04 | .009 | .003 | $\sim 0$ |
| 8 | .997 | .57 | .19 | .06 | .02 | .006 | .002 |
| 16 | .999 | .59 | .21 | .07 | .02 | .008 | .003 |

In order to illustrate that these polynomials are close to the optimal polynomials we explicitly give the fourth degree polynomial for $2a = b = 4$:

$$[P_m(z)]_{optimal} \approx .58801 + .028363z + .144657z^2 + .002135z^3 + .007261z^4$$

$$[P_m(z)]_{Chebyshev} \approx .62092 + 0.0z + .15144z^2 + 0.0z^3 + .007572z^4$$

with respective damping factors .1344 and .1363. □

*3.4. Splitting methods*

Sofar we did not exploit the special structure of the right-hand side function $\mathbf{F}(\mathbf{W})$. We will now consider time integrators which take advantage of the specific form of $\mathbf{F}(\mathbf{W})$. These methods fall into the class of *one-stage splitting methods* defined by

$$\mathbf{W}_{n+1} = \mathbf{W}_n + \lambda\Delta t[\mathbf{G}(\mathbf{W}_{n+1},\mathbf{W}_n) + (\frac{1}{\lambda}-1)\mathbf{G}(\mathbf{W}_n,\mathbf{W}_n)] \qquad (3.4.1)$$

or into the class of *two-stage splitting methods* defined by [19]

$$\begin{aligned}\mathbf{W}^{(1)}_{n+1} &= \mathbf{W}_n + \frac{1}{2}\Delta t[\mathbf{G}(\mathbf{W}^{(1)}_{n+1},\mathbf{W}_n) + (2\lambda-1)\mathbf{G}(\mathbf{W}_n,\mathbf{W}_n)] \\ \mathbf{W}_{n+1} &= \mathbf{W}_n + \frac{1}{2}\Delta t[\mathbf{G}(\mathbf{W}^{(1)}_{n+1},\mathbf{W}_n) + (2-\frac{1}{\lambda})\mathbf{G}(\mathbf{W}_n,\mathbf{W}_{n+1}) + \\ &\quad (\frac{1}{\lambda}-1)\ \mathbf{G}(\mathbf{W}^{(1)}_{n+1},\mathbf{W}_{n+1})].\end{aligned} \qquad (3.4.2)$$

Here, $\mathbf{G}(\mathbf{W},\tilde{\mathbf{W}})$ is a so-called splitting function satisfying the splitting condition

$\mathbf{G}(\mathbf{W},\mathbf{W}) \equiv \mathbf{F}(\mathbf{W})$. The one-stage method is first order accurate for all $\lambda$. The two-stage method is second order for all $\lambda$. In the literature one meets also two-stage methods which use different splitting functions in the successive stages.

*3.4.1. One-stage methods.* The one-stage methods may be considered as a method in between the explicit and implicit Euler method. If $\lambda = 0$ we have the *explicit* Euler method and if $\lambda = 1$ with $\mathbf{G}(\mathbf{W},\tilde{\mathbf{W}}) = \mathbf{F}(\mathbf{W})$ we obtain the *implicit* Euler method.

In the examples given below we have $\lambda = 1$. Furthermore, the function $\mathbf{F}(\mathbf{W})$ is defined by the discretization of the right-hand side of (1.2) omitting the force term $\mathbf{r}$. Thus,

$$\mathbf{F}(\mathbf{W}) = -\begin{bmatrix} UD_x + VD_y & -f & gD_x \\ f & UD_x + VD_y & gD_y \\ HD_x & HD_y & UD_x + VD_y \end{bmatrix} \mathbf{W},$$

where $D_x$ and $D_y$ are discretizations of $\partial/\partial x$ and $\partial/\partial y$, and $U, V$, and $H$ are the diagonal matrices $diag(\mathbf{U})$, $diag(\mathbf{V})$ and $diag(\mathbf{H})$; we will also write $H_x$ instead of $D_x H$, etc..

EXAMPLE 3.4.1. Fischer -Sielecki method [9]

$$\mathbf{G}(\mathbf{W},\tilde{\mathbf{W}}) = -\begin{bmatrix} 0 & 0 & 0 \\ f & 0 & 0 \\ \tilde{H}_x + \tilde{H}D_x & \tilde{H}_y + \tilde{H}D_y & 0 \end{bmatrix} \mathbf{W} \; - \begin{bmatrix} \tilde{U}D_x + \tilde{V}D_y & -f & gD_x \\ 0 & \tilde{U}D_x + \tilde{V}D_y & gD_y \\ 0 & 0 & 0 \end{bmatrix} \tilde{\mathbf{W}}.$$

When implemented this splitting function generates a completely explicit scheme. The stability condition reads $\Delta t \leqslant 2\sqrt{\|g\mathbf{H}\|(S_x^2 + S_y^2)}$ where $S_x$ and $S_y$ are the spectral radii of $D_x$ and $D_y$. For a detailed discussion of the Fischer-Sielecki method and its modifications we refer to [18].

*Navon's method* [34]

$$\mathbf{G}(\mathbf{W},\tilde{\mathbf{W}}) = -\begin{bmatrix} UD_x + \tilde{V}D_y & 0 & gD_x \\ f & UD_x + VD_y & gD_y \\ 0 & 0 & \tilde{U}D_x + \tilde{V}D_y + \tilde{U}_x + \tilde{V}_y \end{bmatrix} \mathbf{W} \; - \begin{bmatrix} 0 & -f & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tilde{\mathbf{W}}.$$

By first solving the (linear) equation for $\mathbf{H}_{n+1}$ we obtain a (nonlinear) equation in $\mathbf{U}_{n+1}$ alone and a (nonlinear) equation for $\mathbf{V}_{n+1}$ alone.

*3.4.2. Two-stage methods.* In the case of (3.4.2), both stages should be defined in such a way that $\mathbf{W}^{(1)}_{n+1}$ and $\mathbf{W}_{n+1}$ can be 'conveniently' obtained, that is $\partial\mathbf{G}/\partial\mathbf{W}$ and $\partial\mathbf{G}/\partial\tilde{\mathbf{W}}$ are required to have a simple structure.

EXAMPLE 3.4.2. *Classical ADI splitting*

$$\mathbf{G}(\mathbf{W},\tilde{\mathbf{W}}) = -\begin{bmatrix} UD_x & 0 & gD_x \\ f & UD_x & 0 \\ HD_x & 0 & UD_x \end{bmatrix}\mathbf{W} - \begin{bmatrix} \tilde{V}D_y & -f & 0 \\ 0 & \tilde{V}D_y & gD_y \\ 0 & \tilde{H}D_y & \tilde{V}D_y \end{bmatrix}\tilde{\mathbf{W}}. \quad (3.4.3)$$

This splitting is the most natural one (cf.[19] for a survey), and leads to an ADI type splitting method. In fact, this type of method (with $\lambda = ½$) was investigated by GUSTAFSSON [12] who gives a detailed discussion of the solution of the implicit relations (notice that the stages in (3.4.2) are one-dimensional implicit when using (3.4.3)). A linearized version of this ADI method has been considered by FAIRWEATHER and NAVON [8]. The (linear) stability analysis indicates unconditional stability for linear models with constant coefficients.

*Leendertse's method*

The scheme of Leendertse, in the way it was originally introduced [29], can also be formulated as a two-stage method of the form (3.4.2) with $\lambda = ½$. To that end, however, we have to define two splitting functions, which are different for both stages. In the first stage, this method uses $\mathbf{G}(\mathbf{W},\tilde{\mathbf{W}})$ with components

$$\begin{aligned} \mathbf{G}_1 &= -[\tilde{U}_x\mathbf{U} + \tilde{U}_y\tilde{\mathbf{V}} - f\tilde{\mathbf{V}} + g\mathbf{H}_x] \\ \mathbf{G}_2 &= -[U\tilde{\mathbf{V}}_x + V\tilde{\mathbf{V}}_y + f\mathbf{U} + g\tilde{\mathbf{H}}_y] \qquad (3.4.4) \\ \mathbf{G}_3 &= -[U\mathbf{H}_x + H\mathbf{U}_x + \tilde{V}\tilde{\mathbf{H}}_y + \tilde{H}\tilde{\mathbf{V}}_y]. \end{aligned}$$

By first (simultaneously) solving **U** and **H** and afterwards (explicitly) calculating **V**, only tridiagonal systems have to be solved.

In the second stage a slightly different splitting function is used. To be more precise, the advective terms are replaced by

$$-[U_x\tilde{\mathbf{U}} + U_y\tilde{\mathbf{V}}], \quad -[U\mathbf{V}_x + \tilde{V}\mathbf{V}_y].$$

Solving this stage is similar to the first one, but now the rôles of **U** and **V** are interchanged.

In a later version of this scheme [30] there has been used a staggering in time for the velocity components; this results in a calculation of **U** and **H** at time levels $n + ½$ and of **V** and **H** at levels $n$.

*Stelling's method* [38]
This method uses $\lambda = \frac{1}{2}$ and a splitting function $\mathbf{G}(\mathbf{W},\tilde{\mathbf{W}})$ with components

$$\begin{aligned} \mathbf{G}_1 &= -[\tilde{U}_x\mathbf{U}+\tilde{U}_y\mathbf{V}-f\mathbf{V}+g\mathbf{H}_x] \\ \mathbf{G}_2 &= -[\tilde{U}\mathbf{V}_x+\tilde{V}\mathbf{V}_y+f\tilde{\mathbf{U}}+g\tilde{\mathbf{H}}_y] \\ \mathbf{G}_3 &= -[U\mathbf{H}_x+H\mathbf{U}_x+\tilde{V}\tilde{\mathbf{H}}_y+\tilde{H}\tilde{\mathbf{V}}_y] \end{aligned} \tag{3.4.5}$$

In the first stage (implicit in $\mathbf{W}=\mathbf{W}^{(1)}_{n+1}$) the vector $\mathbf{V}=\mathbf{V}^{(1)}_{n+1}$ can be solved 'conveniently' by using one-sided difference operators; then $\mathbf{U}=\mathbf{U}^{(1)}_{n+1}$ can explicitly be expressed in terms of $\mathbf{V}$ and $\mathbf{H}=\mathbf{H}^{(1)}_{n+1}$, and substitution into the equation for $\mathbf{H}$ yields an equation in $\mathbf{H}$ alone. The second stage is treated in a similar manner. Notice the strong-implicit treatment of the convection term. This method is claimed to be highly stable even in the presence of nonlinear terms.

## 4. Future developments

In this section we will briefly discuss a few aspects which may become important for the solution of the SWEs.

### *4.1. Vector processing*

Since the numerical solution of a large-scale, realistic shallow water model is a tremendous task, a huge increase in computer speed as well as memory capacity is needed in order to obtain a sufficiently detailed simulation for engineering purposes. A useful alternative to the traditional scalar computer may be the so-called vector computer. The last type of machine is designed to enhance the concurrency of arithmetic operations, which results in a high system throughput. To be more precise, vectors (i.e., ordered sets of values) are operated with one single instruction.

Since 1984 the CWI has access to a CYBER 205, which is a vector processor, also called pipeline machine. Therefore, we will globally consider the consequences for solving the SWEs when using such a computer. (For a detailed discussion of parallel computing we refer to [16].)

In order to utilize the potential speed of a vector computer we have to satisfy certain constraints.

First of all there is the necessity to adapt the *computer program* to the architecture of the particular computer. More or less, this argument holds for any type of computer but on a vector processor the effects are more pronounced. More serious is the requirement to suit the *algorithm* to the specific architectorial nature of the computer. Traditionally, numerical algorithms were selected on their 'mathematical' qualities, for instance, the rate of convergence in iterative processes. When using a vector computer it is no longer true that algorithms which are 'mathematically' superior to others will result in a better (i.e., faster) performance, which is usually the case on a scalar computer. Therefore, to obtain optimal performance from a vector machine, it is necessary to construct an algorithm which is best suited to that particular machine (running with a particular compiler). In doing so, we have to consider several aspects which will inhibit vectorisation. If vectorisation has to be performed by the compiler, only the source code will be examined. Evidently, DO-loops will be the most likely places where suitable sequences of operations can be found. Therefore, we should keep these loops going on, working on vectors the length of which is as large as possible. Hence, loops containing IF-statements, GOTO-statements or I/0-statements will inhibit vectorisation. Also certain index expression are a barrier to vectorisation, such as indirect addressing or nonlinear index expression. Moreover, calls to subroutines or functions within DO-loops make these loops nonvectorisable. The reason for this is that subprograms generally are compiled seperately.

However, the most restrictive aspect with respect to vectorisation is *recursion*, which means that in a sequence of evaluations the latest term depends on one or more of the previously computed terms, as, for example, in

```
      DO 10 I=2,N
   10 A(I) = A(I-1)+SCALAR*B(I)
```

Because the evaluation of a recurence relation *essentially* is a sequential process, recurrency conflicts with the nature of vectorisation. Recurrences are quite common in all fields of numerical analysis; examples are the calculation of the innerproduct of two vectors, solutions of linear equations by Gaussian elimination and in principle any iterative process in which a new approximation is calculated using previous approximations. Fortunately, for some of these problems manufacturers of vector computers provide a solution, e.g. the CYBER 205 has a special innerproduct instruction. The recursion which has our special attention occurs in the Gaussian elimination process used for solving tridiagonal systems. These systems frequently occur in the splitting methods as described in subsection 3.4. Therefore, in using these splitting methods we will have to use other techniques to solve the tridiagonal systems (such as recursive doubling or cyclic reduction). However, this usually requires additional arithmetic operations and storage.

A last aspect which we want to consider is the *portability*. Because FORTRAN originates from the fifties it lacks any feature for a standard treatment of vector processing. In consequence, each manufacturer developed his own dialect. Needless to say that this is disastrous for portability.

Notwithstanding these reservations, we think that it will be possible to take advantage of this vector processing machine by adapting both the program and the algorithm to this particular computer. Extensive tests have to show which algorithm is maximally benefitted from this type of computer.

Considering the various methods which are described in Section 3 we make a few remarks.

An aspect which seems to be in favour of the explicit methods (such as the Runge-Kutta schemes) is that these methods work with long vectors, whereas the splitting methods typically operate on vectors the length of which equals the number of points in one space-direction. This aspect is especially important for the CYBER 205 because this machine has a relatively large start-up time, which is greatly amortized by executing long vectors. Furthermore, the explicit schemes require only **F**(**W**)-evaluations to solve (2.2.6). Because of their explicit nature, the vector **W** is known prior to the calculations within **F**, which makes these schemes — at least in principle — highly vectorisable.

The refinement of the model describing the SWEs will also have influence on the performance of a vector computer. Evidently, the more sophisticated the model is, the more complicated the program will be. For example, very irregular shapes of the boundary (or even time-dependent boundaries which require a flooding and drying procedure) or special treatment of the advection terms in the SWEs in the neighbourhood of the boundaries, etc. Such situations will cause the program to perform a lot of tests to detect these irregularities. These IF-statements as well as the enormous overhead will prevent the program from optimal performance.

A last facet is the ambiguity of the word *performance.* Is it merely the CPU time that counts or do we also take into account the time needed for transput of data? Another definition of performance could be in terms of costs on a particular computer installation or in terms of memory.

### *4.2 Vertical stratification*

The general equations describing the motion of flow in shallow water are, in principle, three-dimensional. However, the enormous computational task such a 3-D system would require in numerical computations, is out of the scope of nowadays computers.

Therefore, in the momentum equation for the vertical velocity component, usually the following assumptions are made:

(i) the vertical acceleration is small with respect to the acceleration of gravity.

(ii) as the horizontal dimensions are large compared with the depth, the vertical velocity is small with respect to the horizontal velocity.

By these assumptions this momentum equation can be drastically simplified yielding a relation between pressure and gravity [7]. Then, this relation can be used to eliminate the pressure from the other momentum equations. A next step is introducing the depth-averaged horizontal velocity components

$$\bar{u} = \frac{1}{h}\int_0^h u\,dz \quad \text{and} \quad \bar{v} = \frac{1}{h}\int_0^h v\,dz.$$

Now, assuming that the free surface and the bottom are streamlines, which serve as vertical boundary conditions, the equations are integrated over the depth which eliminates the vertical velocity component from the system. It is this system which was considered in the previous sections, where the bars were dropped.

In many applications, this traditional approach has proven to be rather satisfactorily. However if the fluid is not homogeneous with respect to temperature or salinity it may be necessary to consider a model with more than one layer. Now, in each of these layers the SWEs as defined in (1.2) are used, extended with a variable $\rho$ denoting the density in that particular layer. Consequently, $\rho$ will be a function of the temperature or the salinity. Evidently, these layers have to be connected by appropriate interaction conditions, i.e., vertical boundary conditions are imposed assuming that the borders between the layers are streamlines again. These models with vertical stratification have been studied in the literature (see e.g. [14], [41]) but are still in a rather premature stage of development. Further research in this field is necessary in order to obtain a more flexible treatment of realistic models. This technique of using layers is also valuable for modelling the three-dimensional circulation of a homogeneous sea (cf. [3]). Thanks to the ever-increasing computer power the refinement of the models is possible. This may eventually lead to models with many layers or even to fully three-dimensional calculations. Perhaps, in the end, it may give a better comprehension of the phenomenon of turbulence which is still so poorly understood.

### *4.3 Error estimation*

A step forward in shallow-water calculations would be an estimation of the *global* error. The costs of such an estimation may be considerable. For this reason error estimation got little attention. However, a good estimation of the error would increase the reliability of the results, which will be a good starting point for probability calculations in civil engineering projects and thereby may lead to cheaper designs.

In future research on shallow-water equations at the CWI we will concentrate on the use of vector computers and on error estimation.

## References

1. A. Brandt, N. Dinar (1979). Multi-grid solutions to elliptic flow problems. S.V. Parter (ed.). *Numerical Methods for Partial Differential Equations,* Academic Press.
2. G. Dahlquist (1959). Stability and error bounds in the numerical integration of ordinary differential equations (thesis). *Transactions of the Royal Institute of Technology,* No. 130, Stockholm.
3. A.M. Davies (1980). On formulating a three-dimensional hydronamic sea model with an arbitrary variation of vertical eddy viscosity. *Computer Methods in Applied Mechanics and Engineering 22,* 187-211.
4. K. Dekker (1980). Semi-discretization methods for partial differential equations on non-rectangular grids. *Int. J. Num. Meth. Engng. 15,* 405-419.
5. K. Dekker (1981). Stability of linear multistep methods on the imaginary axis. *BIT 21,* 66-79.
6. K. Dekker, J.G. Verwer (1984). *Stability of Runge-Kutta Methods for Stiff Nonlinear Differential Equations,* North-Holland, Amsterdam.

7. J.J. DRONKERS (1964). *Tidal Computations in Rivers and Coastal Waters,* John Wiley & Sons, New York.
8. G. FAIRWEATHER, I.N. NAVON (1980). A linear ADI method for the shallow-water equations. *J. of Comp. Phys. 37,* 1-18.
9. G. FISCHER (1959). Ein numerisches Verfahren zur Errechnung von Windstau und Gezeiten in Randmeeren. *Tellus 11,* 60-76.
10. D. GOTTLIEB, S.A. ORSZAG (1977). Numerical analysis of spectral methods: theory and applications. *CBMS-WSF. Regional Conference Series in Applied Mathematics, No 26,* SIAM Philadelphia.
11. A. GRAMMELTVEDT (1969). A survey of finite-difference schemes for the primitive equations for a barotropic fluid. *Monthly Weather Review 97, no 5,* 384-404.
12. B. GUSTAFSSON (1971). An alternating direction implicit method for solving the shallow water equations. *J. of Comp. Physics 7,* 239-254.
13. W. HANSEN (1956). Theorie zur Errechnung des Wasserstandes und der Strömungen in Randmeeren nebst Anwendungen. *Tellus 8,* 289-300.
14. N.S. HEAPS (1981). Three-dimensional models for tides and surges with vertical eddy viscosity prescribed in two layers. Part I, Mathematical formulation. *Geophys. J.R. astr. Soc. 64,* 291-302.
15. P.W. HEMKER (1981). Introduction to multi-grid methods. *Nieuw Arch. Wiskunde, Ser. (3) 29,* 71-101.
16. R.W. HOCKNEY, C.R. JESSHOPE (1981). *Parallel Computers,* Adam Hilger Ltd, Bristol.
17. P.J. VAN DER HOUWEN (1977). *Construction of Integration Formulas for Initial Value Problems,* North-Holland Publishing Company, Amsterdam.
18. P.J. VAN DER HOUWEN (1977). *Berekening van Waterstanden in Zeeën en Rivieren* (Dutch), MC Syllabus 33, Mathematical Centre, Amsterdam.
19. P.J. VAN DER HOUWEN, J.G. VERWER (1979). One-step splitting methods for semi-discrete parabolic equations. *Computing 22,* 291-309.
20. P.J. VAN DER HOUWEN, B.P. SOMMEIJER (1983). Analysis of Chebyshev relaxation in multigrid methods for non linear parabolic differential equations. *ZAMM 63,* 193-201.
21. P.J. VAN DER HOUWEN, B.P. SOMMEIJER (1983). Predictor-corrector methods with improved absolute stability regions. *JIMANA 3,* 417-437.
22. R. JELTSCH (1978). Stability on the imaginary axis and A-stability of linear multistep methods. *BIT 18,* 170-174.
23. R. JELTSCH, O. NEVANLINNA (1979). *Stability and Accuracy Discretizations for Initial Value Problems,* Oulu report, Helsinki.
24. R. JELTSCH, O. NEVANLINNA (1981). Stability of explicit time discretizations for solving initial value problems. *Num. Math. 37,* 61-91.
25. J. KOK, P.J. VAN DER HOUWEN, P.H.M. WOLKENFELT (1978). *A Semi-discretization Algorithm for Two-dimensional Partial Differential Equations,* Report NW 54/78, Mathematical Centre, Amsterdam.

26. W. KUTTA (1901). Beitrag zur näherungsweisen Integration totaler Differentialgleichungen. *Z. Math. Phys. 46,* 435-453.

27. J.D. LAMBERT (1973). *Computational Methods in Ordinary Differential Equations,* John Wiley & Sons, London.

28. P. LANCASTER (1969). *Theory of Matrices,* Academic Press, New York and London.

29. J.J. LEENDERTSE (1967). *Aspects of a Computational Model for Long-period Water-wave Propagation,* Rand Corp., Mem. RM-5294, Santa Monica.

30. J.J. LEENDERTSE (1970). A water-quality simulation model for well-mixed estuaries and coastal seas. *Volume I, Principles of Computation,* Rand Corp. Mem. RM-6230- RC, Santa Monica.

31. P. MERLUZZI, C. BROSILOW (1978). Runge-Kutta integration algorithms with built-in estimates of the accumulated truncation error. *Computing 20,* 1-16.

32. A.R. MITCHELL, D.F. GRIFFITHS (1980). *The Finite Difference Method in Partial Differential Equations,* John Wiley & Sons, Chichester.

33. K.W. MORTON (1977). Initial-value problems by finite difference and other methods. D. JACOBS (ed.). *The State of the Art in Numerical Analysis,* Academic Press, London, New York, San Francisco.

34. I.M. NAVON (1978). Application of a new partly implicit time differencing scheme for solving the shallow-water equations. *Contrib. Atmospheric Phys. 51,* 281-305.

35. N. PRAAGMAN (1979). *Numerical Solution of the Shallow Water Equations by a Finite Element Method,* Thesis, Delft.

36. R.D. RICHTMYER, K.W. MORTON (1967). *Difference Methods for Initial Value Problems,* Interscience Publishers, New York.

37. P. SONNEVELD, B. VAN LEER (1985). A minimax problem along the imaginary axis. *Nieuw Arch. Wiskunde, Ser (4) 31,* 19-22.

38. G.S. STELLING (1983). *On the Construction of Computational Methods for Shallow Water Flow Problems,* Thesis, Delft.

39. H.J. STETTER (1973). *Analysis of Discretization Methods for Ordinary Differential Equations,* Springer-Verlag, Berlin.

40. R. VICHNEVETSKY (1983). New stability theorems concerning one-step numerical methods for ordinary differential equations. *Mathematics and Computers in Simulation 25,* 199-205.

41. C.B. VREUGDENHIL (1979). Two-layer shallow-water flow in two dimensions, a numerical study. *J. of Comp. Phys. 33,* 169-184.

42. T.J. WEARE (1976). Finite element or finite difference methods for the two-dimensional shallow water equations. *Computer Methods appl. Mech. Engin. 7,* 351-357.

43. T.J. WEARE (1976). Instability in tidal flow computational schemes. *Journal of the Hydraulics Division, ASCE, 102,* 569-580.

44. P. WESSELING (1982). Theoretical and practical aspects of a multigrid method. *SIAM J. Sci. Stat. Comp. 3,* 387-407.