Centrum voor Wiskunde en Informatica
Centre for Mathematics and Computer Science

F.W. Wubs

Performance evaluation of explicit shallow-water equations solver on the cyber 205

Department of Numerical Mathematics Report NM-R8511 June

Performance Evaluation of Explicit Shallow-Water Equations Solvers on the Cyber 205

F.W. Wubs
Centre for Mathematics and Computer Science, Amsterdam

The performance of an explicit method and an ADI method for the shallow-water equations is compared on a CYBER 205.
Furthermore, a stabilization technique is discussed, which stabilizes the explicit method in such a way that any desired time step is possible without the development of instabilities.
Comparing the codes for two test models, we found that the explicit methods are attractive on the CYBER 205.
Finally, some proposals are made for the handling of irregular geometries.

1980 Mathematics Subject Classification: Primary: 65M10, Secondary: 76D99.
Key Words & Phrases: stabilization, hyperbolic equations, method of lines, residual averaging, shallow-water equations.
Note: These investigations were supported by the Netherlands Foundation for Technical Research (STW), futureTechnical Science Branch Division of the Netherlands Organization for the Advancement of Pure Research (ZWO). The experiments were done on a CYBER 750 at the expence of the Centre for Mathematics and Computer Science (CWI).
This paper is submitted for publication elsewhere.

INTRODUCTION

Designing a solver for the hyperbolic shallow-water equations for use on a CYBER 205, we would like to have some insight in the performance of explicit and implicit methods on such a computer. Therefore, we implemented an explicit method on the CYBER 205 and compared the performance of this method with an existing ADI method. However, the available code of the ADI method was not vectorized. Currently this method is in the final stage of vectorization and the computation speed is already known.
For some problems the time step restriction of an explicit method is severe. In order to overcome this restriction, we constructed an explicit stabilization technique, which allows us to use any desired time step.

THE SHALLOW-WATER EQUATIONS

In hydraulic engineering, the shallow-water equations (SWE's) are used to describe flows in shallow seas, estuaries and rivers. Output from the numerical models based on these SWE's can be used as input for models describing the influence of infrastructural works, salt intrusion, the effect of waste discharges, the water quality, cooling water recirculation and sediment transports. An important application, in the Netherlands, is the storm surge barrier under construction in the mouth of the Oosterschelde Estuary, by which this estuary can be closed off during storms. In this case, the numerical model, based on the SWE's, provides guide lines for the operation of the barrier in order to preserve the delicate ecological balance in the estuary, which has an important fish nursery as well as oyster and mussel cultures. Therefore, the construction of efficient

Report NM-R8511
Centre for Mathematics and Computer Science
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands

numerical models for the SWE's is relevant.
The nature of the applications is such that strong gradients are common, though shocks do not appear. As a consequence, it is not neccesary to satisfy numerical conservation of momentum or energy. However, the conservation of mass is important as the depth determines largely the prolongation of the waves.
In Figure 1, we have drawn a cross section of a sea.

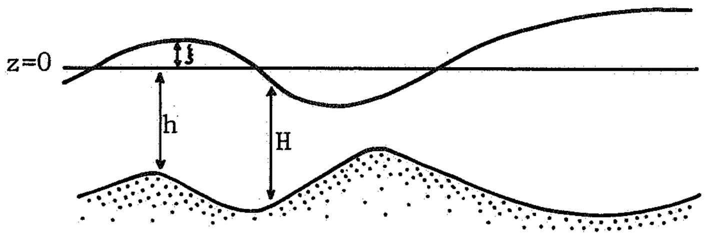

Figure 1 Schematisation of a cross section of the sea.

Having a reference plane z=0, which is for example the mean level of the sea, we define the local bottom profile by h(x,y) and the local elevation by $\xi(x,y,t)$. Now, the total depth is given by $H=h+\xi$. Furthermore, averaging the velocities over the depth and assuming hydrostatic pressure and incompressibility of water, the SWE's read:

$$\begin{aligned}
u_t &= -uu_x - vu_y - g\xi_x + fv - Cz\sqrt{u^2+v^2}\,u/H + \upsilon\Delta u,\\
v_t &= -uv_x - vv_y - g\xi_y - fu - Cz\sqrt{u^2+v^2}\,v/H + \upsilon\Delta v, \qquad (1)\\
\xi_t &= -(Hu)_x - (Hv)_y.
\end{aligned}$$

The first two equations are momentum equations describing, in this incompressible case, the change in time of the averaged velocities u and v. The third one is a continuity equation. In the momentum equation appear the Coriolis force parametrized by f, which is due to the rotation of the earth, and the bottom friction parametrized by Cz. Furthermore, g and υ denote the acceleration due to gravity and the diffusion coefficient for horizontal momentum, respectively. In practice, even more terms are introduced in the equations, but we have restricted ourselves to the most important of them. In practical numerical calculations the grid sizes are so large, of order 100m, that the significance of the diffusion terms is small with respect to both accuracy and stability. As a consequence, the equations behave numerically as hyperbolic equations.

EXPLICIT VERSUS IMPLICIT METHODS

On a vector computer, it is not a priori clear whether implicit or explicit methods should be used. Therefore we have collected some arguments to help us in that decision:

Table 1 Arguments pro and contra explicit and implicit methods.

EXPLICIT	IMPLICIT
simple implementation	difficult implementation
no recursion	recursion
long vectors	short vectors(ADI)
simple to decompose the domain	difficult to decompose the domain
time step restriction	no time step restriction

One could also take into account the need for workspace of the methods. However, as the vector computer CYBER 205 uses dynamically storage, this is not easy to determine. For example, a temporary variable, occurring in a "do loop", to which an array element is assigned, transforms to an array of the length of the "do loop" on a vector machine. Taking this into account it is questionable whether an explicit method uses less storage than an implicit method on a vector computer. But, as the SWE's are a 2-D problem, the storage is not the main problem.
From the above arguments, we decided to use an explicit method. The consequence of this choice is the restriction on the time step, which may be severe for problems, slowly varying in time. Because, in that case we would like to use a large time step. Fortunately, by using a stabilization technique, we were able to overcome this drawback.

DISCRETIZATIONS

In this section, we will shortly describe the ADI method and the explicit method we used in our comparison.

An ADI method

The method given here was designed by Stelling [4] in 1983. It is claimed that this method has good stability properties with respect to the advection terms, which are in general quite troublesome. Furthermore, on a sequential computer the storage requirements are very low. For the tests we performed, we had not a vectorized version of this program. Currently, the program is in the final stage of vectorization. The main problem for the vectorization is the solution process for the system of equations occurring after discretization. Here, this is handled by using cyclic reduction which vectorizes to a good degree. The execution time of the vectorized version for long vectors is about $2.\ 10^{-5}$seconds per grid point per time step on a one-pipe CYBER 205 in half precision, whereas this number is $1.5\ 10^{-4}$ for the scalar version.
For the introduction of the ADI method we use the method of lines approach[3]. In this approach, first (1) is semi-discretized with respect to space, giving the system of ordinary differential equations

$$\frac{d}{dt}\vec{W} = \vec{F}(\vec{W}), \tag{2}$$

where $\vec{W}=(\vec{U},\vec{V},\vec{\xi})$ and $\vec{F}(\vec{W})$ is a short form for the discretized right-hand side of (1). Then, for the time integration of (2), an appropriate time integrator is used. In this case, the time integrator is given in the so-called split notation [1]:

$$\begin{aligned} \vec{W}^{n+\frac{1}{2}} &= \vec{W}^{n} + 1/2\Delta t\ \vec{G}(\vec{W}^{n+\frac{1}{2}},\vec{W}^{n}), \\ \vec{W}^{n+1} &= \vec{W}^{n+\frac{1}{2}} + 1/2\Delta t\ \vec{G}(\vec{W}^{n+\frac{1}{2}},\vec{W}^{n+1}), \end{aligned} \tag{3}$$

where the splitting function $\vec{G}$ satisfies $\vec{G}(\vec{W},\vec{W})=\vec{F}(\vec{W})$. It is straight-forward to show that (3) is second-order in time. Furthermore, the components of $\vec{G}$ are described by

$$\begin{aligned} \vec{G}_1(\vec{W},\tilde{\vec{W}}) &= -[\ U\tilde{U}_x + V\tilde{U}_y + g\xi_x \ldots.], \\ \vec{G}_2(\vec{W},\tilde{\vec{W}}) &= -[\ \tilde{U}V_x + \tilde{V}V_y + g\tilde{\xi}_y \ldots.], \\ \vec{G}_3(\vec{W},\tilde{\vec{W}}) &= -[\ (UH)_x + (\tilde{V}\tilde{H})_y\ ], \end{aligned} \tag{4}$$

where only the most important terms are given. The brackets [] denote the space discretization of the terms within it. This discretization, which is done on a space staggered grid, is second order and central for all terms except for the cross terms VU_y and UV_x which are treated third order. This third order treatment is chosen because of its damping properties for high frequency components in the numerical solution.
In the time discretization, the treatment of the advection terms is different from what is done usually in ADI methods. For example, in the first stage, the momentum equation for V is treated explicitly in the elevation (pressure) term and implicitly in the advection terms. Usually the advection terms are treated also explicitly in this stage. For a more detailed discussion of this discretization and its properties we refer to the author's thesis [4].
The method has the important property that the implicit systems contain only tridiagonal matrices, which allow a fast solution process.

An explicit method

The method we implemented is straight-forward. The time integration is given by:

$$\vec{W}^{n+1} = \vec{W}^{n} + \Delta t\ (\vec{K}_1 + 2\vec{K}_2 + 2\vec{K}_3 + \vec{K}_4)/6, \tag{5}$$

where

$$\begin{aligned} \vec{K}_1 &= \vec{F}(\vec{W}^{n}), \\ \vec{K}_2 &= \vec{F}(\vec{W}^{n} + 1/2\Delta t\ \vec{K}_1), \\ \vec{K}_3 &= \vec{F}(\vec{W}^{n} + 1/2\Delta t\ \vec{K}_2), \\ \vec{K}_4 &= \vec{F}(\vec{W}^{n} + \Delta t\ \vec{K}_3). \end{aligned} \tag{6}$$

This is the classical fourth-order Runge-Kutta method [3]. The imaginary stability boundary of this method is $C=2\sqrt{2}$. The space-discretization is

also treated fourth-order consistent. However, we have added a fourth-order diffusion term with an $O(\Delta x^3)$ term (Δx is the mesh size in space directions) in front of it, which has the same effect as a third order treatment of the advection terms. This term can be incorporated, without costs, into the physical diffusion, because this only involves a change of constants.
The execution time for long vectors for this method is 6.10^{-6} seconds per grid point per time step for half precision calculation on a CYBER 205, which is approximately three times faster than the vectorized ADI-implementation.

STABILIZATION OF THE TIME INTEGRATION

For some problems the stability condition of the explicit method is much to restrictive. Hence, we tried to stabilize the explicit method in order to be able to use larger step sizes. Here, we shortly give the basic idea. For details we refer to the reports [5] and [6].
Consider the hyperbolic equation

$$\vec{w}_t = \vec{f}(\vec{w},\vec{w}_x,\vec{w}_y,x,y,t), \quad (x,y)\in R^2,\ t>0, \tag{7}$$

where $\vec{w}=(\vec{u},\vec{v},\vec{\xi})$. Now, under certain conditions, we can show that, if the time derivatives of $\vec{w}$ are small then the space derivatives of $\vec{f}$ are also small, where $\vec{f}$, after substitution of the exact solution, is only a function of x,y and t.
This property means that the right-hand side function is quite smooth in the space directions if the solution varies slowly in time. Note that this does not imply that $\vec{w}$ itself is smooth in space directions. Hence, if the solution varies slowly in time, then this property of a smooth right-hand side allows us to smooth the right-hand side of the discretized equations.
Using the method of lines approach[3], semi-discretization of (7) results in the system of ordinary differential equations

$$\frac{d}{dt}\vec{W} = \vec{F}(\vec{W}). \tag{8}$$

Instead of (8), we propose to solve

$$\frac{d}{dt}\vec{W} = S\,\vec{F}(\vec{W}), \tag{9}$$

where S is a smoothing operator satisfying

$$S = I + O(h^2). \tag{10}$$

We give here two examples of smoothing operators. First an implicit one:

$$-\mu(S\vec{F})_{j-1} + (1+2\mu)\,(S\vec{F})_j - \mu(S\vec{F})_{j+1} = F_j. \tag{11}$$

This operator is applied first in x-direction and thereafter in y-direction. The stability condition is now given by

$$\Delta t \leq C\,\Delta x\sqrt{\mu}\ \ 1/(7/6\sqrt{2\,gH}), \tag{12}$$

where C is the imaginary stability boundary of the time integrator. The same operator is used by Jameson[2]. He uses it to stabilize explicit methods for boundary-value problems. However, using a vector computer, we

are more interested in explicit smoothing operators.
An explicit smoothing operator is given by

$$S := \prod_{k=1}^{n} S_k, \tag{13}$$

where

$$(S_k\vec{F})_j = \mu_k F_{j-2^{k-1}} + (1-2\mu_k)F_j + \mu_k F_{j+2^{k-1}}, \; \mu_k < 1/4. \tag{14}$$

Again the operator is applied in x- and y-direction, successively. Using this operator, the stability condition is, for $\mu_k = 1/4$,

$$\Delta t < C\Delta x \; 2^n \; 3\sqrt{3}/4 \quad 1/(7/6\sqrt{2\;gH}). \tag{15}$$

In practical computations we choose μ_k smaller than 1/4 such that the constant $3\sqrt{3}/4$ can be replaced by 1. This choice assures that the smoothing operator (14) is diagonal dominant for all k, which appeared to be essential for initial-boundary-value problems. With both methods arbitrary step sizes are possible; in the implicit method μ should be chosen such that (12) is satisfied and in the explicit method n should be chosen such that (15) is satisfied.
The advantage of the second operator is that it vectorizes much better than the implicit operator. We found that one smoothing operator of the form (14) is 25 times faster than the implicit smoother in a half precision calculation. In this experiment, the decomposition of the implicit smoothing operator was performed beforehand as far as possible. As for our purpose an increase of the time step by a factor 8 or 16 (obtained by 3 and 4 smoothings respectively) is usually sufficient, the explicit smoothing is about 8 or 6 times as fast as implicit smoothing. This factor will become even larger if more vector pipes are used.
Furthermore, the cost of one explicit smoothing is about one nineth of the $\vec{F}$ evaluation (in case of the SWE's). As the $\vec{F}$ evaluations determine the costs of the whole integration, one smoothing increases also the costs of the whole integration with this factor.

RESULTS

We have tested both methods on two simple geometries: a one-dimensional problem and a two-dimensional symmetric problem. The tests were performed on a one-pipe CYBER 205 in half precision, which means a machine precision of approximately 7 digits.

A one-dimensional problem

The geometry of the first problem is drawn in Figure 2. It consists of a basin of length 50km with a bump in the middle.

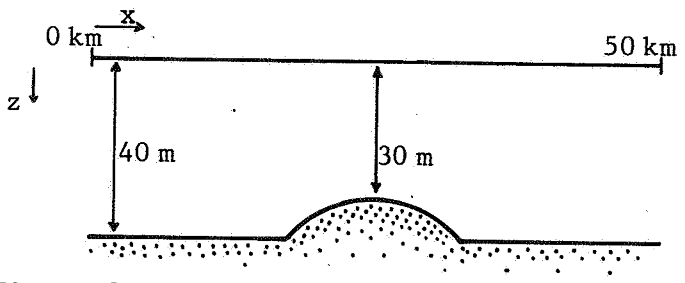

Figure 2 Geometry of a one-dimensional problem.

The bottom profile is given by

$$h(x,y)=\begin{cases} 40 - 10\sin((r+1)\pi/2)\ [m] & \text{for} |r| < 1, \\ 40\ [m] & \text{elsewhere,} \end{cases} \tag{16}$$

where

$$r = (x-2.5\ 10^4)/\ 10^4. \tag{17}$$

At the boundaries the elevation is prescribed by

$$\begin{aligned} \xi(0,t) &= -\sin(wt)\ [m], \\ \xi(50.10^3,t) &= -\sin(wt-\phi)\ [m], \end{aligned} \tag{18}$$

where

$$\begin{aligned} w &= 2\pi/(12*3600)\ [s^{-1}], \\ \phi &= 2\pi\ 5/60. \end{aligned} \tag{19}$$

Furthermore, the constants in the equations (1) are chosen

$$\begin{aligned} f &= 0\ [s^{-1}], \\ \upsilon &= 0\ [m^2/s], \\ Cz &= 4.\ 10^3. \end{aligned} \tag{20}$$

We integrated 28 hours physically with various time steps (see Table 2). The calculations were performed on a 48x48 grid. At the end of each calculation we compared the solution with a reference solution computed on a finer grid (96x96). The results are given in significant digits of the ξ-component, defined by

$$Sdm = -{}^{10}\log\ (\ \max|\xi - \xi_{ref}|\ /\ \max|\xi_{ref} - \bar{\xi}_{ref}|) \tag{21}$$

where the subsript ref denotes the reference solution and the bar above $\bar{\xi}_{ref}$ denotes the average over the whole computational domain. The results are given in Table 2

Table 2 Significant digits for the one-dimensional problem

Δt[s]	ADI method			Explicit method		
	Computation time [s]		Sdm	Computation time [s]	Sdm	Number of smoothings
	scalar	vector				
21				78	2.3	0
42				39	2.3	0
84	454	65	1.5	22	2.1	1
168	227	32	1.5	13	1.5	2
336	114	16	1.5	8	.9	3
672	56	8	1.1			

In the first column, the time step is given; it increases downward with a factor two. In the second and fifth column the computation times are given. In the third column, we have added the expected computation times of the vectorized version of the ADI method. In the fourth and sixth column, the significant digits are given. In the last column, we have given the number of smoothings used in the product sequence (13). The time step of 42 seconds is close to the maximum time step allowed without smoothing.
With respect to accuracy, we see for both methods the same effect when the time step increases. At first, the number of significant digits changes slightly; then, when the time step is larger than 84 and 336 seconds for the explicit and implicit method, respectively, the number of significant digits decreases rapidly. This can be understood by the following reasoning. At first the error due to the space discretization, largely induced by the bump, is larger than the error due to the time discretization. This time discretization error is largely determined by the time-dependent boundary conditions (18). Then, when the time step increases the error due to the time discretization becomes more and more significant and even larger than the error due to space discretization.
With this in mind, we did not performe calculations with the implicit method for the time steps 21 and 42 seconds. The calculations would become rather expensive for these time steps, but the error would be the same as for the time step 84 seconds.
From the results we see that after vectorization of the ADI method the explicit method is generally faster even if a rather modest accuracy is required.

A symmetric two-dimensional problem

Now, we choose a geometry as drawn in Figure 3. In the middle of the basin, which has in this case sides of length 5km, we have placed a round bump.

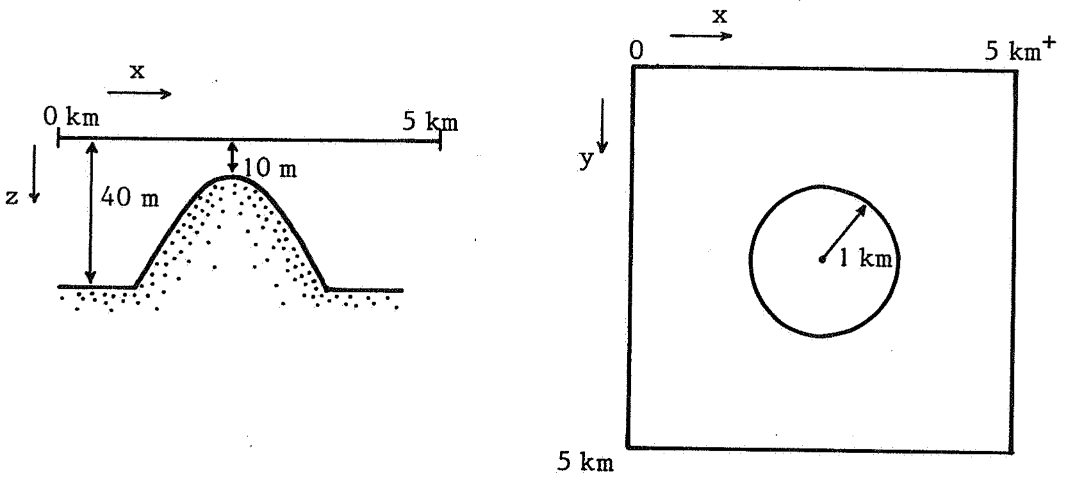

Figure 3 Geometry of the symmetric two-dimensional problem.

The bottom profile is in this case given by

$$h(x,y) = \begin{cases} 40 - 30\cos(\pi * r/2) \text{ [m] for } r<1, \\ 40 \text{ [m]} \quad \text{elsewhere,} \end{cases} \tag{22}$$

where

$$r = \sqrt{(x - 2.5\ 10^3)^2 + (y - 2.5\ 10^3)^2} / 1000. \tag{23}$$

At the left and right boundary, again the elevation is prescribed

$$\begin{aligned} &\xi(0,y,t) = - \sin(wt), \\ &\xi(5.0\ 10^3,y,t) = - \sin(wt-\phi),\ 0<y<5.0\ 10^3 \text{[m]}, \end{aligned} \tag{24}$$

where

$$\begin{aligned} w &= 2\pi/(12*3600) \text{ [s}^{-1}\text{]}, \\ \phi &= 2\pi\ 5/600. \end{aligned} \tag{25}$$

At the upper and lower boundary, at $y=0$ and $y=5.0\ 10^3$ [m] respectively, the normal velocity component v is zero. Furthermore, the constants in the equations (1) are chosen

$$\begin{aligned} f &= 0 \text{ [s}^{-1}\text{]}, \\ \upsilon &= 10 \text{ [m}^2\text{/s]}, \\ Cz &= 4.\ 10^{-3}. \end{aligned} \tag{26}$$

We integrated 15 hours physically with various time steps (see Table 3). The calculations were performed on a 24x24 grid. At the end of each calculation we compared the solution with a reference solution computed on a finer grid (96x96). The results are now given in significant digits of the v-component. In this case we have added also a root-mean-squares error given by

$$\begin{aligned} &Sd2 = -{}^{10}\log\ (\ |v - v_{ref}|_2 / |v_{ref} - \bar{v}_{ref}|_2), \\ &|v|_2 = \sqrt{\textstyle\sum_i v_i^2} / (24X24), \end{aligned} \tag{27}$$

where the summation is over all grid points. The results are given in Table 3.

Table 3 Significant digits for the symmetric two-dimensional problem

Δt[s]	ADI method				Explicit method			
	Computation time [s]		Sd2	Sdm	Computation time [s]	Sd2	Sdm	Number of smoothings
	scalar	vector						
8					45	2.1	1.3	0
16					26	2.1	1.3	1
32	181	40	1.6	1.1	15	2.1	1.3	2
64	91	20	1.4	.8	9	1.6	1.0	3
128	46	10	.8	.2				

Globally, we observe the same effect with the errors as in the previous case. At first, the number of significant digits remains constant as the time step increases and then, when the time step becomes larger than 32 seconds with both methods, the error due to the time step becomes dominant. Again we see that to obtain a certain accuracy the explicit method is faster than the ADI method.
On this 24X24 grid, we have lost already a factor 3 with respect to the asymptotical speed on a very fine grid due to the start up times of the vector instructions. The asymptotical speed is for the explicit method almost reached for the 96X96 grid.
It is clear that without the stabilization technique the explicit method would become to expensive with respect to a vectorized version of the ADI method.

IRREGULAR GEOMETRIES

In the near future, we want to start calculations for irregular geometries. When in such a case the computational domain is covered with a rectangle, on which a fast computation can be performed, a lot of idle computations on "land points" are done. To minimize this number of land points, we have two proposals. The first is to decompose the domain and the second is the use of two numberings.

Decomposition of the domain

In Figure 4, we have drawn an irregular geometry, which is covered by a number of rectangles.

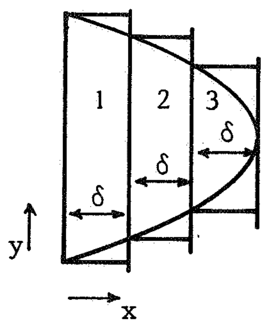

Figure 4 Irregular geometry covered with rectangles.

These rectangles have been chosen such that they all have equal width δ. This allows us to put them together into one large rectangle as drawn in Figure 5.

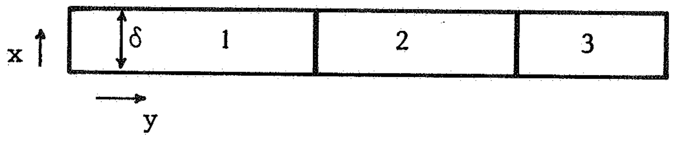

Figure 5 Composition of the rectangles into one large rectangle.

On this rectangle again a fast calculation can be performed. It will be clear that the rectangles in Figure 4 should have a small overlap in order to calculate differences near the interfaces. This involves some copying of data each time step.

Two numberings

Another possibility is to use two numberings, one in x-direction and one in y-direction as drawn in Figure 6.

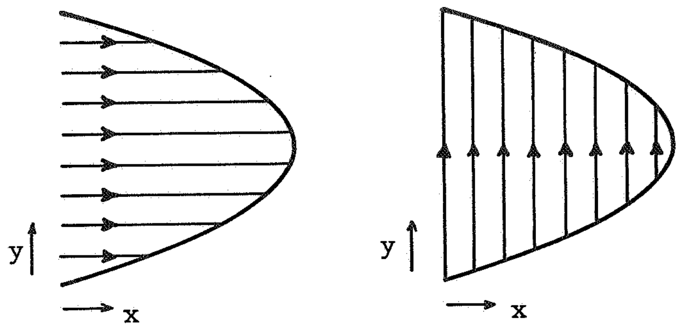

Figure 6 A numbering in x- and y-direction.

Now, $\vec{F}$ is split up into two parts

$$\vec{F} = \vec{F}^x + \vec{F}^y, \qquad (28)$$

where $\vec{F}^x$ and $\vec{F}^y$ contain only differences with respect to x and y, respectively. During the F evaluation, the numbering of the variables has to be changed in order to have a fast evaluation of both $\vec{F}^x$ and $\vec{F}^y$. The renumbering can be done with a gather or scatter vector instruction.

CONCLUSIONS

In this contribution, we have considered the performance of an explicit method and an implicit method for the SWE's. Implementing an explicit method on a vector computer is quite straight-forward, whereas with implicit methods, one has to reconsider the algebraic solution process in order to solve the occurring set of equations as effective as possible.
The stabilization technique presented in this contribution works well for the SWE's, i.e., the explicit Runge-Kutta method with stabilization is competitive with the ADI method with respect to both accuracy and computational costs. Furthermore, it can be easily added to an existing explicit time integration process based on the method of lines approach.

REFERENCES

[1] Houwen, P.J. van der, and J.G. Verwer, One-Step Splitting Methods for Semi-Discrete Parabolic Equations, Computing 22, pp 291-309, 1979.

[2] Jameson, A., and D. Mavriplis, Finite Volume Solution of the Two-Dimensional Euler Equations on a Regular Triangular Mesh, AIAA 23rd Aerospace Sciences Meeting, AIAA-85-0435, Nevada, 1985.

[3] Lambert, J.D., Computational Methods in Ordinary Differential Equations, Wiley, London-New York, 1973.

[4] Stelling, G.S., On the Construction of Computational Methods for Shallow-Water Flow Problems, Thesis, TH Delft, 1983.

[5] Wubs, F.W., Stabilization of Explicit Methods for Hyperbolic Initial-Value Problems, in preparation, C.W.I., Amsterdam 1985.

[6] Wubs, F.W., Smoothing Techniques for Initial-Boundary-Value Problems, in preparation.