SIAM J. SCI. COMPUT. © 2011 Society for Industrial and Applied Mathematics
Vol. 33, No. 2, pp. 588–611

VORTICITY PRESERVING FINITE VOLUME SCHEMES FOR THE SHALLOW WATER EQUATIONS[*]

ULRIK S. FJORDHOLM[†] AND SIDDHARTHA MISHRA[†]

Abstract. We propose a finite volume method for the shallow water equations that accurately approximates the transport of vorticity. The algorithm is based on a predictor–corrector-type projection method. *Any* consistent finite volume scheme may be used in the prediction step of the algorithm. An elliptic equation is solved and the momentum field is corrected to obtain the correct evolution of vorticity. We describe this projection algorithm for the wave equation and the shallow water equations. The crucial role played by the *pseudovorticity* transport is highlighted. Numerical experiments demonstrating a considerable gain in computational efficiency with the vorticity projection algorithm are presented.

Key words. finite volume methods, shallow water equations, vorticity preservation

AMS subject classifications. 65M06, 35L65

DOI. 10.1137/090775956

1. Introduction.

In many interesting flows like those in rivers, in the near shore ocean and irrigation channels, the vertical scale (depth) of the flow is much smaller than the horizontal scales. This scale separation can be employed to simplify the incompressible Euler equations and obtain the so-called *shallow water equations*:

$$\begin{aligned} h_t + (hu)_x + (hv)_y &= 0,\\ (hu)_t + \left(hu^2 + \frac{1}{2}gh^2\right)_x + (huv)_y &= 0,\\ (hv)_t + (huv)_x + \left(hv^2 + \frac{1}{2}gh^2\right)_y &= 0, \end{aligned} \tag{1.1}$$

where h is the height of the fluid column and (u,v) is the velocity field. The constant g is the acceleration due to gravity.

The shallow water equations are an example of a system of conservation laws,

$$U_t + f(U)_x + g(U)_y = 0, \tag{1.2}$$

where $U = [h, hu, hv]^\top$ is the vector of unknowns and $f = [hu, hu^2 + \frac{1}{2}gh^2, huv]^\top$ and $g = [hv, huv, hv^2 + \frac{1}{2}gh^2]^\top$ are the flux vectors. A straightforward calculation yields that the eigenvalues of the Jacobians $f'(U)$ and $g'(U)$, respectively, are

$$\begin{aligned} &\lambda_1 = u - \sqrt{gh}, && \lambda_2 = u, && \lambda_3 = u + \sqrt{gh},\\ &\mu_1 = v - \sqrt{gh}, && \mu_2 = v, && \mu_3 = v + \sqrt{gh}. \end{aligned}$$

Hence, the shallow water system is a strictly hyperbolic system. Eigenvectors can be calculated similarly [16].

[*]Submitted to the journal's Methods and Algorithms for Scientific Computing section November 3, 2009; accepted for publication (in revised form) December 6, 2010; published electronically March 1, 2011.

http://www.siam.org/journals/sisc/33-2/77595.html

[†]Seminar for Applied Mathematics, D-MATH, ETH Zürich, Rämistrasse 101, 8092 Zürich, Switzerland (ulrikf@sam.math.ethz.ch, smishra@sam.math.ethz.ch).

Hyperbolic conservation laws (1.2) are characterized by the formation of finite time discontinuities, even for smooth initial data. These discontinuities or *shock waves* are a very important object of study. The presence of shock waves implies that solutions of (1.2) are sought in the weak sense [6]. It is also well known that weak solutions are not necessarily unique. Additional admissibility criteria or *entropy conditions* need to be imposed in order to recover uniqueness. For the shallow water system (1.1), the total energy

$$E = \frac{1}{2}\left(hu^2 + hv^2 + gh^2\right)$$

serves as the entropy function, with corresponding entropy flux functions

$$H(U) = \frac{1}{2}\left(hu^3 + huv^2\right) + gh^2u \qquad \text{and} \qquad K(U) = \frac{1}{2}\left(hu^2v + hv^3\right) + gh^2v.$$

Thus, the entropy condition imposed on weak solutions is

$$E(U)_t + H(U)_x + K(U)_y \leq 0 \tag{1.3}$$

(in the sense of distributions). The energy inequality leads to bounds on the solutions of the shallow water system. In addition to the total energy, another object of interest is the *vorticity*,

$$\omega = v_x - u_y. \tag{1.4}$$

This quantity measures the rotation of the flow and is very important in practical applications. A detailed account of the role played by vorticity in meteorological models is available in [1, 2] and other references therein. By a simple calculation, one finds that smooth solutions of the shallow water equations satisfy the vorticity transport equation

$$\omega_t + (u\omega)_x + (v\omega)_y = 0. \tag{1.5}$$

Thus, vorticity is advected with the flow. In particular, the total circulation

$$C^*(t) = \int_{\mathbb{R}^2} \omega(x, y, t)dxdy$$

is conserved in time. However, this conservation is valid only for smooth solutions, and for discontinuous solutions, vorticity may be generated over shocks [12].

A related object of interest is the *potential vorticity*,

$$\zeta = \frac{\omega^2}{h}.$$

Smooth solutions of (1.1) also satisfy the potential vorticity transport equation

$$\zeta_t + (u\zeta)_x + (v\zeta)_y = 0. \tag{1.6}$$

Integrating (1.6) in time and utilizing the positivity of height, we obtain bounds on the vorticity in L^2.

A similar situation is encountered with the Euler equations of gas dynamics [16]. The physical entropy satisfies an inequality of the form (1.3), and the vorticity of the flow is transported as in (1.5). Preserving quantities of interest like the energy (entropy) and vorticity in numerical approximations of hyperbolic conservation laws is a major computational challenge [19].

1.1. Finite volume schemes. In the absence of explicit formulas for the solution of (1.1), numerical methods are the main tools for studying the shallow water equations. Among the most popular methods are the so-called finite volume methods [16]. For simplicity, we consider a uniform Cartesian mesh $\{(x_i, y_j)\}$ in $\mathbb{R}^2$ with mesh sizes Δx and Δy. The domain is partitioned into rectangular cells $I_{i,j} = [x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}] \times [y_{j-\frac{1}{2}}, y_{j+\frac{1}{2}}]$. A standard cell-centered finite volume method consists of updating the cell averages

$$U_{i,j}(t) = \frac{1}{\Delta x \Delta y} \int_{I_{i,j}} U(x, y, t) dx dy$$

at each time level. For simplicity, we drop the time dependence of every quantity and write a standard finite volume scheme for (1.2) in the semidiscrete form as

$$\frac{d}{dt} U_{i,j} = \mathcal{L}(U_{i,j}) := -\frac{1}{\Delta x}\left(F_{i+\frac{1}{2},j} - F_{i-\frac{1}{2},j}\right) - \frac{1}{\Delta y}\left(G_{i,j+\frac{1}{2}} - G_{i,j-\frac{1}{2}}\right), \tag{1.7}$$

where F and G are numerical fluxes at the cell edges that are consistent with the fluxes f and g, respectively. The numerical fluxes are computed by using either exact or approximate solutions of Riemann problems across the cell edges in the normal directions. They take the form

$$F_{i+\frac{1}{2},j} = F(U_{i,j}, U_{i+1,j}), \qquad G_{i,j+\frac{1}{2}} = G(U_{i,j}, U_{i,j+1}). \tag{1.8}$$

The scheme (1.7), based on two-point numerical fluxes (1.8), is only first-order accurate, but it can be extended to higher-order accuracy by employing numerical fluxes based on wider $(2p+1)$-point stencils,

$$F_{i+\frac{1}{2},j} = F\left(U_{i-p+1,j}, \ldots, U_{i+p,j}\right), \qquad G_{i,j+\frac{1}{2}} = G\left(U_{i,j-p+1}, \ldots, U_{i,j+p}\right). \tag{1.9}$$

The building blocks for such extensions are still the 2-point numerical fluxes $F(\cdot,\cdot)$ and $G(\cdot,\cdot)$. As a prototype example, we recall the class of second-order schemes based on piecewise bilinear MUSCL reconstruction [14],

$$p_{i,j}(x, y) := U_{i,j} + U'_{i,j}\frac{x - x_i}{\Delta x} + U^{\backprime}_{i,j}\frac{y - y_j}{\Delta y}. \tag{1.10a}$$

Here, U' and $U^{\backprime}$ denote the *numerical derivatives*

$$\begin{aligned} U'_{i,j} &= \operatorname{minmod}\left(U_{i+1,j} - U_{i,j},\ U_{i,j} - U_{i-1,j}\right), \\ U^{\backprime}_{i,j} &= \operatorname{minmod}\left(U_{i,j+1} - U_{i,j},\ U_{i,j} - U_{i,j-1}\right), \end{aligned} \tag{1.10b}$$

which utilize the minmod limiter

$$\operatorname{minmod}(a, b) = \begin{cases} \operatorname{sgn}(a)\min\{|a|, |b|\} & \text{if } \operatorname{sgn}(a) = \operatorname{sgn}(b), \\ 0 & \text{otherwise.} \end{cases} \tag{1.10c}$$

(For systems of equations in which case U is an n-vector, the reconstruction is done componentwise.) In this manner, one can reconstruct in each cell $I_{i,j}$ the point values

$$\begin{aligned} U^E_{i,j} &:= p_{i,j}(x_{i+\frac{1}{2}}, y_j), & U^W_{i,j} &:= p_{i,j}(x_{i-\frac{1}{2}}, y_j), \\ U^N_{i,j} &:= p_{i,j}(x_i, y_{j+\frac{1}{2}}), & U^S_{i,j} &:= p_{i,j}(x_i, y_{j-\frac{1}{2}}) \end{aligned} \tag{1.11a}$$

from the neighboring cell values $U_{i,j}$, $U_{i\pm 1,j}$, and $U_{i,j\pm 1}$. The resulting second-order fluxes are then given by

$$F_{i+\frac{1}{2},j} = F\left(U^E_{i,j}, U^W_{i+1,j}\right), \qquad G_{i,j+\frac{1}{2}} = G\left(U^N_{i,j}, U^S_{i,j+1}\right). \tag{1.11b}$$

The use of the minmod limiter ensures the nonoscillatory behavior of the second-order schemes (1.7), (1.10). Similar reconstructions together with upwind or central averaging yield a large class of high-resolution finite volume semidiscrete schemes; see, e.g., [10, 13, 21],

For time integration, we use a standard forward Euler method for first-order schemes and for second-order schemes the strong stability preserving (SSP) Runge–Kutta method of [9]. Given a solution $U^n_{i,j}$ at time step t_n, the solution $U^{n+1}_{i,j}$ is computed by

$$\begin{aligned} U^*_{i,j} &= U^n_{i,j} + \Delta t_n \mathcal{L}(U^n_{i,j}), \\ U^{**}_{i,j} &= U^*_{i,j} + \Delta t_n \mathcal{L}(U^*_{i,j}), \\ U^{n+1}_{i,j} &= \frac{1}{2}(U^n_{i,j} + U^{**}_{i,j}), \end{aligned} \tag{1.12}$$

where $\mathcal{L}$ is defined in (1.7). The time step Δt_n is determined by a standard CFL condition. In all simulations, we use a CFL number of 0.9.

A wide variety of finite volume methods are available for the shallow water equations [16]. However, standard numerical methods may not faithfully discretize objects of interest like energy and vorticity. In particular, the numerical approximations may not respect the energy inequality (1.3) or the vorticity transport (1.5). Energy errors are highly problematic for long time integration of the shallow water system [2]. In the recent papers [7, 8], the authors present energy preserving and energy stable schemes for the shallow water system and demonstrate a major gain in accuracy as well as stability when the energy preserving numerical schemes are used.

Standard finite volume schemes may produce large errors in the vorticity transport [19, 12], which may result in incorrect qualitative behavior of, for instance, meteorological flows [2]. Similar issues arise in the simulation of the Euler equations of gas dynamics. It is well known [11] that standard finite volume schemes produce large vorticity errors, which results in incorrect approximations for strongly vortical flows.

1.2. Vorticity preservation.

1.2. Vorticity preservation. Our goal is to design *vorticity preserving schemes* —numerical schemes that transport vorticity accurately—for the shallow water equations (1.1). Recent papers like [12, 19] have proposed a simple model problem, the *wave equation*

$$\begin{bmatrix} \hbar \\ m_1 \\ m_2 \end{bmatrix}_t + \begin{bmatrix} 0 & c & 0 \\ c & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \hbar \\ m_1 \\ m_2 \end{bmatrix}_x + \begin{bmatrix} 0 & 0 & c \\ 0 & 0 & 0 \\ c & 0 & 0 \end{bmatrix} \begin{bmatrix} \hbar \\ m_1 \\ m_2 \end{bmatrix}_y = 0, \tag{1.13}$$

to act as a testbed for the design of such schemes. The wave equation may be viewed as a linearization of (1.1) around a stationary background state $h = h_0$, $u = v = 0$. Letting $c = \sqrt{gh_0}$, $\hbar = ch$, $m_1 = hu$, and $m_2 = hv$ and ignoring terms of order two or more in (1.1), we get (1.13) precisely. The eigenvalues of the flux matrices in (1.13) are $-c$, 0, and c, and so the wave equation is strictly hyperbolic.

The relevant measure of vorticity for the wave equation is the curl of the momentum field,

$$\Gamma := (m_2)_x - (m_1)_y. \tag{1.14}$$

It is easily seen that this quantity is constant in time:

$$\begin{aligned}\Gamma_t &= -\big((c\hbar_y)_x - (c\hbar_x)_y\big)\\ &= 0.\end{aligned} \tag{1.15}$$

There are two approaches for design vorticity preserving schemes. The first, explored in recent papers like [12, 18, 19], is based on rewriting the standard finite volume scheme (1.7) on staggered meshes [19], in terms of flux distributions [12], or in a genuinely multidimensional manner [18]. The resulting schemes preserve a discrete version of the vorticity for the linear wave equation (1.13). The task of extending these schemes to obtain schemes that transport vorticity correctly (discrete version of (1.5)) for the shallow water equations is highly nontrivial. This issue has been addressed and successfully resolved for transporting vorticity accurately for the nonlinear Euler equations in a recent paper [15]. The authors build on the staggered mesh method of [19] and devise residual based schemes based on novel numerical diffusion operators.

The second approach for designing schemes with correct vorticity transport was recently proposed by Ismail and Roe [11] in the context of the Euler equations of gas dynamics. Their algorithm is based on a modified projection method. The projection method relies upon the *Hodge decomposition* of any vector field into a divergence free part and a curl free part and was first used for incompressible flow calculations by Chorin in [5]. Extensions of the projection method for incompressible flow were proposed in [3]. Another application of the projection method is in numerical methods for the magnetohydrodynamics (MHD) equations [22]. The MHD equations are a nonlinear system of conservation laws, equipped with the constraint that the magnetic field remain divergence-free during the evolution. Standard numerical schemes like the finite volume scheme (1.7) may not respect this constraint and may lead to oscillations and other instabilities [22]. In [4], the authors apply the projection method to clean divergence. This approach is quite popular in MHD codes.

The projection method leads to an elliptic equation at every time step. This equation can be solved by standard fast elliptic solvers, and the resulting solution satisfies the appropriate constraint. However, the projection method of [3, 4, 5] cannot be directly used to design a scheme that transports vorticity correctly for the shallow water system (1.1). Instead, as in [11], an estimate of the vorticity is provided by the *pseudovorticity*, defined as the curl of the momentum field $(m_1, m_2) = (hu, hv)$:

$$\Omega = (m_2)_x - (m_1)_y. \tag{1.16}$$

As momentum, and not velocity, is a conserved variable, it will be easier to impose a correct flow in pseudovorticity rather than in vorticity. Smooth solutions of the shallow water system (1.1) satisfy the pseudovorticity transport equation

$$\begin{aligned}&\Omega_t + (u\Omega)_x + (v\Omega)_y + \big(m_2(u_x+v_y)\big)_x - \big(m_1(u_x+v_y)\big)_y\\ &\quad + \left(\frac{u^2+v^2}{2}h_y\right)_x - \left(\frac{u^2+v^2}{2}h_x\right)_y = 0.\end{aligned} \tag{1.17}$$

The pseudovorticity transport is more complicated than the vorticity transport (1.5), as second derivative terms appear in the equation explicitly. Furthermore, the pseudovorticity transport (1.17) accounts for compressibility. Note that if the flow is incompressible, i.e, the velocity field is divergence-free, and the height h is constant in space, then (1.17) reduces to the vorticity transport equation (1.5). Thus, nonzero divergence of velocity and height variations contribute to the additional terms in (1.17). The approach of [11] relied on controlling the pseudovorticity for the Euler equations, which satisfies an equation analogous to (1.17). However, because of an incorrect vector identity in equation 25 of [11], the computation of the pseudovorticity transport equation (equation 30 in [11]) is incorrect, particularly when the height (density for the Euler equations) varies in space. This error is not reflected in the numerical experiments reported in [11], as the authors present only test cases with constant density. The projection method of [11] is based on a relaxation-type elliptic solver.

In this paper we propose a projection method for the shallow water equations. Our *predictor-corrector method* involves a prediction step in which the solution is evolved by *any* consistent finite volume method (1.7). In particular, high-order accurate finite volume schemes (1.10) can be used in this step. The essential step involves controlling pseudovorticity by a suitable discretization of the transport equation (1.17). The correction step is based on a Hodge decomposition and results in an elliptic equation that needs to be solved at every time step. We utilize fast solvers to solve the elliptic equation: a direct method for Neumann boundary conditions and a suitable conjugate gradient method for periodic boundary conditions. Some stability estimates are derived for the projection method for the system wave equation. Numerical experiments are presented to demonstrate the robustness and computational efficiency of this modified projection method for both the linear wave equation (1.13) and the shallow water system (1.1).

The rest of this paper is organized as follows. We present the projection method for the system wave equation in section 2 and for the shallow water equations in section 3. Conclusions are drawn in section 4.

2. Vorticity projection for the system wave equation.

Before presenting the vorticity projection method for the shallow water equations, we will as a motivation consider the simpler context of the linear system wave equation (1.13). The simplicity of this system allows us to prove stability estimates for the projection method in this simpler context.

We design a projection method for approximating the system wave equation (1.13) that preserves a discrete version of the vorticity (1.14). Recall that the vorticity of the wave equation is constant in time (cf. (1.15)). The projection method, which we describe presently, will respect a discrete version of this property.

We employ the following discretization of vorticity:

$$\Gamma_{i,j} = D_x(m_2)_{i,j} - D_y(m_1)_{i,j}, \tag{2.1}$$

where D_x and D_y are the central differences

$$D_x U_{i,j} = \frac{U_{i+1,j} - U_{i-1,j}}{2\Delta x} \qquad \text{and} \qquad D_y U_{i,j} = \frac{U_{i,j+1} - U_{i,j-1}}{2\Delta y}. \tag{2.2}$$

We seek to modify the finite volume scheme (1.7) such that this discrete version of vorticity remains constant in time. We do so by the following discrete projection algorithm.

Step 1: *Prediction.* Compute a candidate solution $\widetilde{U}^{n+1}_{i,j}$ for (1.13) at time t_{n+1} for all mesh points x_i, y_j with *any* consistent and conservative finite volume scheme (1.7). Let $\widetilde{\Gamma}^{n+1}_{i,j}$ be the discrete vorticity of $\widetilde{U}^{n+1}_{i,j}$, computed as in (2.1).

Step 2: *Elliptic solve.* Find the solution $\psi = (\psi_{i,j})$ of the discrete Poisson problem

$$\begin{cases} -\left(D_x^2 + D_y^2\right)\psi_{i,j} = \widetilde{\Gamma}_{i,j}^{n+1} - \Gamma_{i,j}^0 & \text{in the interior,} \\ \psi_{i,j} = 0 & \text{at the boundary.} \end{cases} \tag{2.3}$$

Here, $\Gamma_{i,j}^0$ denotes the initial vorticity.

Step 3: *Projection.* Define the solution $U_{i,j}^{n+1} = \left[\hbar_{i,j}^{n+1}, (m_1)_{i,j}^{n+1}, (m_2)_{i,j}^{n+1}\right]^\top$ at the next time step as

$$\begin{aligned} \hbar_{i,j}^{n+1} = \widetilde{\hbar}_{i,j}^{n+1}, \qquad (m_1)_{i,j}^{n+1} &= (\widetilde{m}_1)_{i,j}^{n+1} - D_y\psi_{i,j}, \\ (m_2)_{i,j}^{n+1} &= (\widetilde{m}_2)_{i,j}^{n+1} + D_x\psi_{i,j}. \end{aligned} \tag{2.4}$$

Any finite volume scheme can be used in Step 1 of the algorithm. Similarly, any linear system solver can be used to solve the discrete Poisson equation in Step 2. Hence, the algorithm is very general. The key feature of the projection method is stated in the following lemma.

LEMMA 2.1. *The solution* $U^{n+1} = (\hbar^{n+1}, m_1^{n+1}, m_2^{n+1})$ *computed by the projection method satisfies for all* $n \in \mathbb{N}_0$

$$\Gamma_{i,j}^{n+1} = \Gamma_{i,j}^0,$$

with $\Gamma_{i,j}^0$ *being the initial vorticity. Hence, the discrete vorticity remains constant in time.*

Proof. Simply inserting the definition of m_1^{n+1} and m_2^{n+1} (cf. (2.4)) gives

$$\begin{aligned} D_x(m_2)_{i,j}^{n+1} - D_y(m_1)_{i,j}^{n+1} &= D_x(\widetilde{m}_2)_{i,j}^{n+1} - D_y(\widetilde{m}_1)_{i,j}^{n+1} + \left(D_x^2 + D_y^2\right)\psi_{i,j} \\ &= \Gamma_{i,j}^0. \qquad \square \end{aligned}$$

Remark 2.2. The following properties hold:

(i) All quantities, including Γ, ψ, $D_x\psi$, *and* $D_y\psi$, are cell-centered.

(ii) The discrete Laplace operator $D_x^2 + D_y^2$ in (2.3) uses cell values two grid cells apart. The purpose of using such a wide stencil is to avoid the need to stagger the vorticity variable Γ. It may seem like this introduces an odd/even decoupling of grid cells; however, such effects are removed through the tight coupling of neighboring cells in the prediction solver. No signs of odd/even decoupling were observed in any of the numerical experiments.

(iii) The overall scheme (prediction + correction) is conservative, as the correction term $D_x\psi_{i,j}$ can be written as a flux difference

$$D_x\psi_{i,j} = \frac{1}{\Delta x}\left(\frac{\psi_{i,j} + \psi_{i+1,j}}{2} - \frac{\psi_{i-1,j} + \psi_{i,j}}{2}\right)$$

and likewise for $D_y\psi_{i,j}$. It is straightforward to check that this rewriting as a flux difference, together with a conservative scheme in the prediction step, leads to conservation of momentum.

In addition to vorticity preservation, another desirable feature of the projection algorithm is its stability. The system wave equation (1.13) is equipped with the energy

$$E = \frac{1}{2}\left(\hbar^2 + m_1^2 + m_2^2\right). \tag{2.5}$$

A straightforward calculation shows that the energy satisfies the identity

$$E_t + (c\hbar m_1)_x + (c\hbar m_2)_y = 0. \tag{2.6}$$

Hence, energy is conserved in time. Many standard finite volume schemes like the Rusanov scheme and the Roe scheme (detailed descriptions are provided later in this section) are energy stable—the total energy dissipates in time. Hence, using such a scheme in Step 1 of the discrete projection algorithm results in a decrease in energy. Next, we show that Steps 2 and 3 of the projection algorithm also dissipate energy. We have dropped the time dependence of all quantities for notational convenience.

LEMMA 2.3. *Let E be the energy of a solution computed by the projection method, and let $\widetilde{E} = \frac{1}{2}\big(\widetilde{\hbar}^2 + \widetilde{m}_1^2 + \widetilde{m}_2^2\big)$ be the energy of the solution obtained in the prediction step. We have*

$$\sum_{i,j} E_{i,j} \leq \sum_{i,j} \widetilde{E}_{i,j} - 2\sum_{i,j} \Gamma_{i,j}\psi_{i,j}.$$

In particular, if the initial vorticity is zero, then

$$\sum_{i,j} E_{i,j} \leq \sum_{i,j} \widetilde{E}_{i,j}.$$

Proof. We use summation by parts extensively. As $\psi = 0$ at the boundary, all boundary terms will drop out. Then

$$\begin{aligned}\sum_{i,j} E_{i,j} &= \sum_{i,j} \big(\hbar_{i,j}^2 + (\widetilde{m}_{1,i,j} - D_y\psi_{i,j})^2 + (\widetilde{m}_{2,i,j} + D_x\psi_{i,j})^2\big)\\ &= \sum_{i,j} \widetilde{E}_{i,j} - 2\sum_{i,j}\big(\widetilde{m}_{1,i,j}D_y\psi_{i,j} - \widetilde{m}_{2,i,j}D_x\psi_{i,j}\big) + \sum_{i,j}\big((D_x\psi_{i,j})^2 + (D_y\psi_{i,j})^2\big).\end{aligned}$$

The second term is

$$\begin{aligned}2\sum_{i,j}\big(\widetilde{m}_{1,i,j}D_y\psi_{i,j} - \widetilde{m}_{2,i,j}D_x\psi_{i,j}\big) &= -2\sum_{i,j}\psi_{i,j}(D_y\widetilde{m}_{1,i,j} - D_x\widetilde{m}_{2,i,j})\\ &= 2\sum_{i,j}\psi_{i,j}\big(\Gamma_{i,j} - (D_x^2 + D_y^2)\psi_{i,j}\big)\\ &= 2\sum_{i,j}\Gamma_{i,j}\psi_{i,j} - 2\sum_{i,j}\big((D_x\psi_{i,j})^2 + (D_y\psi_{i,j})^2\big).\end{aligned}$$

Hence,

$$\begin{aligned}\sum_{i,j} E &= \sum_{i,j}\widetilde{E}_{i,j} - 2\sum_{i,j}\Gamma_{i,j}\psi_{i,j} - \sum_{i,j}\big((D_x\psi_{i,j})^2 + (D_y\psi_{i,j})^2\big)\\ &\leq \sum_{i,j}\widetilde{E}_{i,j} - 2\sum_{i,j}\Gamma_{i,j}\psi_{i,j}.\end{aligned}$$

If $\Gamma_{i,j} \equiv 0$, then the last term drops out, leading to energy dissipation. ☐

2.1. Prediction step. In order to complete the description of the discrete projection algorithm, we need to specify the numerical fluxes for the prediction step (Step 1 of the algorithm). As stated earlier, *any* consistent numerical flux can be used in the finite volume scheme (1.7). We use the Rusanov flux, which takes the form

$$\begin{aligned} F^{\mathrm{Rus}}_{i+\frac{1}{2},j} &= \frac{1}{2}\big(f(U_{i,j}) + f(U_{i+1,j})\big) - \frac{c}{2}(U_{i+1,j} - U_{i,j}), \\ G^{\mathrm{Rus}}_{i,j+\frac{1}{2}} &= \frac{1}{2}\big(g(U_{i,j}) + g(U_{i,j+1})\big) - \frac{c}{2}(U_{i,j+1} - U_{i,j}), \end{aligned} \tag{2.7}$$

where f and g are fluxes for the system wave equation (1.13) in the x- and y- directions, respectively. The Rusanov flux is very simple to implement and requires minimal characteristic information. However, it can be dissipative, particularly at discontinuities. A more accurate numerical flux is the Roe flux

$$\begin{aligned} F^{\mathrm{Roe}}_{i+\frac{1}{2},j} &= \frac{1}{2}\big(f(U_{i,j}) + f(U_{i+1,j})\big) - \frac{1}{2}R^x|\Lambda^x|(R^x)^{-1}(U_{i+1,j} - U_{i,j}), \\ G^{\mathrm{Roe}}_{i,j+\frac{1}{2}} &= \frac{1}{2}\big(g(U_{i,j}) + g(U_{i,j+1})\big) - \frac{1}{2}R^y|\Lambda^y|(R^y)^{-1}(U_{i,j+1} - U_{i,j}), \end{aligned} \tag{2.8}$$

where Λ^x, Λ^y are the diagonal matrices of eigenvalues and R^x, R^y the matrices of eigenvectors for the Jacobians of the wave equation (1.13).

2.2. Elliptic solver. The discrete Poisson equation (2.3) must be solved at every time step, so a fast and efficient solver is essential to the computational efficiency of the scheme. The subject of efficient linear solvers for the Poisson equation is treated in many text books. We describe the methods used in our computations very briefly. For simplicity, we compute on an $N \times M$ uniform Cartesian grid. If we write $C_{i,j} = \widetilde{\Gamma}^{n+1}_{i,j} - \Gamma^0_{i,j}$, then the matrix equation (2.3) can be rewritten as

$$\frac{1}{4\Delta x^2} T_N \Psi + \frac{1}{4\Delta y^2} \Psi T_M = C, \tag{2.9}$$

where $\Psi = (\psi_{i,j})$ and T_N is the $N \times N$ symmetric, positive definite matrix

$$T_N = \begin{bmatrix} 2 & 0 & -1 & & & & \\ 0 & 2 & 0 & -1 & & & \\ -1 & 0 & 2 & 0 & -1 & & \\ & \ddots & & \ddots & & \ddots & \\ & & -1 & 0 & 2 & 0 & -1 \\ & & & -1 & 0 & 2 & 0 \\ & & & & -1 & 0 & 2 \end{bmatrix}. \tag{2.10}$$

We solve this equation with a direct method from [17]. Let R_N be the matrix of eigenvectors and $D_N = \mathrm{diag}(\lambda^1_N, \dots, \lambda^N_N)$ be the matrix of eigenvalues of T_N. R_N may be chosen such that $R_N^2 = I_N$, the identity matrix in $\mathbb{R}^{N\times N}$. Multiplying (2.9) by R_N on the left and R_M on the right, we find that

$$\frac{1}{4\Delta x^2} D_N X + \frac{1}{4\Delta y^2} X D_M = R_N C R_M,$$

where $X = R_N \Psi R_M$. The (i, j) entry of the left-hand side of this equation is

$$\frac{\lambda^i_N}{4\Delta x^2} X_{i,j} + \frac{\lambda^j_M}{4\Delta y^2} X_{i,j}.$$

Hence, $X = (R_N C R_M)./S$, where

$$S_{i,j} = \frac{\lambda_N^i}{4\Delta x^2} + \frac{\lambda_M^j}{4\Delta y^2}$$

and ./ denotes componentwise division. The solution of (2.9) is therefore

$$\Psi = R_N X R_M.$$

2.2.1. Periodic boundary conditions. The above discussion did not rely on the specifics of the boundary condition on U. In the special case of periodic boundary conditions,

$$U_{0,j} = U_{N,j}, \qquad U_{-1,j} = U_{N-1,j},$$

and similarly in the y-direction we can improve the method by applying the same condition on ψ in (2.3). This will result in the matrix equation

$$\frac{1}{4\Delta x^2} P_N \Psi + \frac{1}{4\Delta y^2} \Psi P_M = C, \tag{2.11}$$

with

$$P_N = \begin{bmatrix} 2 & 0 & -1 & & & -1 & \\ 0 & 2 & 0 & -1 & & & -1 \\ -1 & 0 & 2 & 0 & -1 & & \\ & \ddots & & \ddots & & \ddots & \\ & & -1 & 0 & 2 & 0 & -1 \\ -1 & & & -1 & 0 & 2 & 0 \\ & -1 & & & -1 & 0 & 2 \end{bmatrix}. \tag{2.12}$$

The direct method of the previous section is unsuitable for this case. Hence, we will employ the conjugate gradient method [17]. The conjugate gradient method searches for the solution of a matrix equation $Az = b$ along orthogonal search paths $s_k \in \mathbb{R}^{NM}$, and it finds the exact solution after at most NM iterations. As this number grows large quadratically, we will use the method as an iterative method, halting the process when $||s_k||_{\ell^2}$ is less than some $\varepsilon > 0$. Given a right-hand side b, we set $\varepsilon = \alpha ||b||_{\ell^2}$ for an $\alpha > 0$, so that the allowed error in the solution is proportional to b. We choose $\alpha = 10^{-8}$ in this paper. Numerical evidence indicates that the error in vorticity is proportional to α. Lowering α has very little impact on runtime, indicating that the conjugate gradient method converges fast. Using higher values of α leads to a slight increase in the error. Hence, the above choice is a suitable one.

The matrix-matrix equation (2.11) can be rewritten as a matrix-vector equation $Az = b$, with

$$z = \mathrm{vec}(\Psi), \quad b = \mathrm{vec}(C), \quad \text{and} \quad A = \frac{1}{4\Delta x^2} P_N \otimes I_M + \frac{1}{4\Delta y^2} I_N \otimes P_M,$$

with $\otimes$ denoting the Kronecker product and vec(Ψ) the column-first vectorized version of Ψ [17]. Contrary to T_N, the matrix P_N is only positive semidefinite; its kernel is spanned by the vectors

$$r_N^1 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix}, \quad r_N^2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ \vdots \end{bmatrix} \in \mathbb{R}^N$$

if N is even and by

$$r_N^1 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \end{bmatrix} \in \mathbb{R}^N$$

if N is odd. Hence, the matrix A will be positive semidefinite with a kernel spanned by the vector(s)

$$r_N^k \otimes r_M^l.$$

As a consequence, the equation $Az = b$ does not have a solution whenever b has a nonzero component in $\ker A$. However, the equation $Az = \widehat{b}$, where

$$\widehat{b} = \operatorname{proj}_{(\ker A)^\perp}(b) \ \ (= \operatorname{proj}_{\operatorname{Im} A}(b)),$$

does have a unique solution in $(\ker A)^\perp$. The conjugate gradient method is well defined and converges for this modified equation. Therefore, we propose using the solution Ψ of this modified equation. Note that the above discussion provides a sort of preconditioner for the problem: We have modified the ill-conditioned problem $Az = b$ by multiplying on both sides by the projection matrix of $(\ker A)^\perp$, thus obtaining $Az = \widehat{b}$.

Remark 2.4. The linear solvers presented above are not necessarily optimal. We can employ more efficient *scalable* iterative solvers like variants of the preconditioned conjugate gradient and other Krylov space methods for solving the linear system (2.3) and increase the efficiency of our algorithm. However, we choose to keep the description of the projection algorithm simple and present the simplest linear solvers.

Remark 2.5. The projection method described in this section is based on a simple Cartesian mesh. In real-life applications, one has to deal with unstructured meshes. It is relatively simple to extend the projection algorithm to unstructured meshes; finite volume schemes on unstructured meshes are readily available, and a suitable discrete form of vorticity is easily obtained [19]. Solving the resulting discrete Poisson equation on an unstructured mesh is standard numerical linear algebra. All these ingredients need to be combined to yield a vorticity projection algorithm on unstructured meshes. We will describe this method in detail in a forthcoming paper.

2.3. Numerical experiments. We test the vorticity preserving (VP) schemes presented in the last section on a couple of numerical experiments for the wave equation (1.13). The following schemes are tested:

Rusanov	First-order finite volume scheme (1.7) with Rusanov flux (2.7).
Roe	First-order finite volume scheme (1.7) with Roe flux (2.8).
VPRus	Vorticity preserving scheme with Rusanov flux in the prediction step.
VPRoe	Vorticity preserving scheme with Roe flux in the prediction step.

Numerical experiment: Periodic waves. The first experiment features a periodic boundary and a nonzero initial vorticity. The initial conditions are given by

$$\hbar \equiv 0, \qquad m_1(x,y) = m_2(x,y) = \cos(\pi(x+y)) - \cos(\pi(x-y)).$$

It is readily checked that

$$\hbar(x,y,t) = \sqrt{2}\sin(\pi(x+y))\sin(\sqrt{2}\pi t),$$
$$m_1(x,y,t) = m_2(x,y,t) = \cos(\pi(x+y))\cos(\sqrt{2}\pi t) - \cos(\pi(x-y))$$

is the solution of this initial value problem. The corresponding vorticity is

$$\Gamma(x,y,t) = (m_2)_x - (m_1)_y = 2\pi\sin(\pi(x-y)).$$

As expected, the expression for Γ is constant in time.

We compute for $(x,y) \in [-2,2]\times[-2,2]$ up to time $t = 2$. The solution at the final time step computed with the Rusanov, VPRus, Roe, and VPRoe schemes is plotted in Figure 1, along with the exact solution. While $\hbar$ is left untouched by the VP schemes, the vorticity field is resolved much more sharply than in the predictor schemes. This is verified in Tables 1 and 2, where we show relative errors

$$\frac{\|\Gamma - \Gamma_{\text{exact}}\|_{L^2}}{\|\Gamma_{\text{exact}}\|_{L^2}}$$

for Γ and $\hbar$ on a sequence of meshes. The VP schemes preserve vorticity up to an error of the order of 10^{-12}, while the Rusanov and Roe schemes have errors in vorticity of the order of discretization error.

Momentum is shown in the third column of Figure 1. Clearly, the projection methods have a positive effect on the accuracy, preventing too much diffusion in m. Indeed, as shown in Table 3, the error in momentum is about 20–40 percent lower than in the prediction solvers.

The conjugate gradient method used in the elliptic solver converges rapidly, with only 5 to 10 iterations needed to get below the error threshold. Thus, the overhead is low, and the VP schemes take only about 1.5 to 3 times more time to run than the standard schemes. The runtime for the VP schemes can be improved by employing a more efficient iterative solver for the elliptic equation (2.3).

TABLE 1
Relative error in Γ.

	Rusanov	VPRus	Roe	VPRoe
40	8.65×10^{-1}	9.07×10^{-12}	6.28×10^{-1}	6.02×10^{-11}
80	6.29×10^{-1}	6.27×10^{-11}	3.90×10^{-1}	4.40×10^{-11}
160	3.90×10^{-1}	1.70×10^{-11}	2.19×10^{-1}	1.04×10^{-11}
320	2.19×10^{-1}	3.28×10^{-12}	1.16×10^{-1}	1.73×10^{-12}

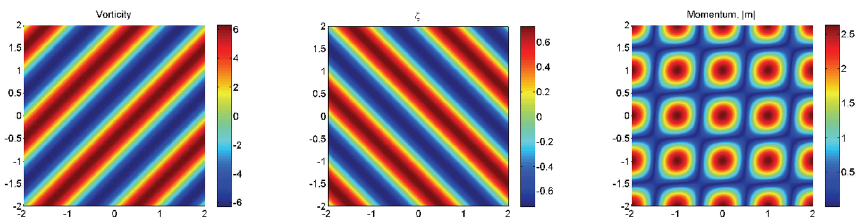
TABLE 2
Relative error in $\hbar$.

	Rusanov	VPRus	Roe	VPRoe
40	7.39×10^{-1}	7.39×10^{-1}	4.66×10^{-1}	4.66×10^{-1}
80	4.51×10^{-1}	4.51×10^{-1}	2.62×10^{-1}	2.62×10^{-1}
160	2.48×10^{-1}	2.48×10^{-1}	1.39×10^{-1}	1.39×10^{-1}
320	1.30×10^{-1}	1.30×10^{-1}	7.18×10^{-2}	7.18×10^{-2}

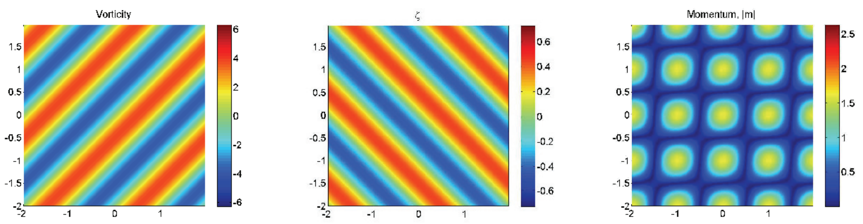
Numerical experiment: Expanding wave. This experiment features a smooth solution with an open- (Neumann-) type boundary condition. The initial data is given by

$$\hbar(x,y) = c\exp\left(-15(x^2+y^2)\right), \qquad m_1 = m_2 = 0. \tag{2.13}$$

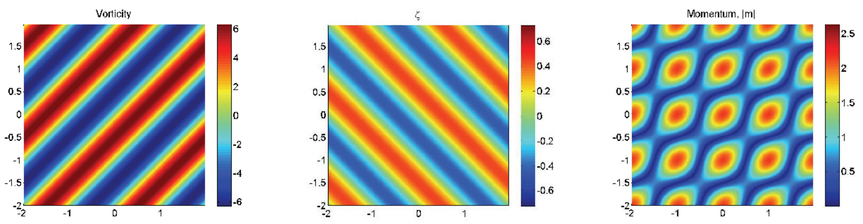
As the initial vorticity is zero, it should stay zero at all later times.



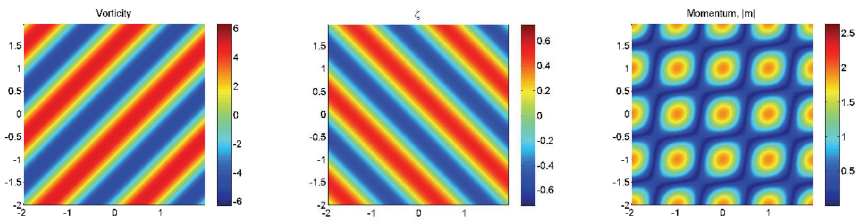
(a) Exact solution



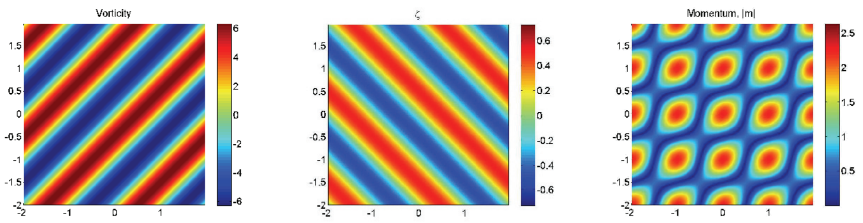
(b) Rusanov



(c) VPRus



(d) Roe

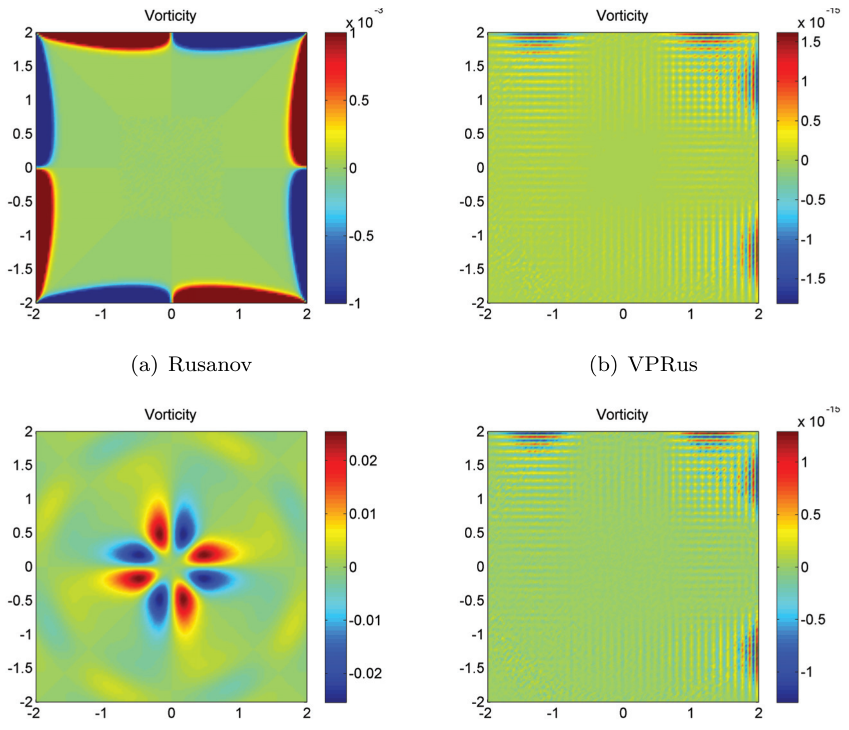


(e) VPRoe

FIG. 1. *Solutions at* $t = 2$ *computed by the four schemes on a mesh of* 160×160 *grid points.*

TABLE 3
Relative error in m, the momentum.

	Rusanov	VPRus	Roe	VPRoe
40	7.77×10^{-1}	4.16×10^{-1}	5.64×10^{-1}	3.02×10^{-1}
80	5.46×10^{-1}	2.66×10^{-1}	3.43×10^{-1}	1.74×10^{-1}
160	3.33×10^{-1}	1.52×10^{-1}	1.91×10^{-1}	9.41×10^{-2}
320	1.85×10^{-1}	8.20×10^{-2}	1.01×10^{-1}	4.89×10^{-2}



(a) Rusanov (b) VPRus

(c) Roe (d) VPRoe

FIG. 2. *Vorticity at $t = 2$ for the four schemes, computed on a mesh of 150×150 grid points. Note the scaling of each figure.*

We solve for $(x, y) \in [-2, 2] \times [-2, 2]$ up to time $t = 2$. The spatial domain was discretized with $N = M = 50, 100, 150$, and 200 grid points in each direction. Vorticity at the final time step is shown in Figure 2. The Rusanov scheme preserves the initial (zero) vorticity exactly in the middle of the domain, but there is a large amount of vorticity production near the boundaries. This vorticity propagates into the domain at a speed of one grid cell per time step. As the ratio $\frac{\Delta t}{\Delta x}$ is kept constant, this speed is invariant with respect to grid size. This is clear in Table 4, where we show vorticity errors for the four schemes. The error for the Rusanov scheme is about $3 \cdot 10^{-2}$, irrespective of grid size. The VPRus scheme clears out these errors, and only noise of the order of machine precision is left (note the scaling of the figures).

The vorticity of the Roe scheme is shown in Figure 2(c). The figure shows that the initial constant vorticity is not preserved. Again, the projection method clears out these errors completely.

The projection method gives no gain in accuracy for the conserved variables in this experiment. As the runtime of the method lies between 2 and 4 times that of the predictor solver, there is little incentive for using the vorticity projection when the main interest is an accurate solution of conserved variables. However, the success of the method in clearing out vorticity errors motivates its use in computing vortex dominated flows.

Table 4
$\|\Gamma\|_{L^1}$ *in the expanding wave problem.*

	Rusanov	VPRus	Roe	VPRoe
50	$2.43 \cdot 10^{-2}$	$3.64 \cdot 10^{-16}$	$8.10 \cdot 10^{-2}$	$2.61 \cdot 10^{-16}$
100	$2.83 \cdot 10^{-2}$	$7.96 \cdot 10^{-16}$	$5.83 \cdot 10^{-2}$	$6.71 \cdot 10^{-16}$
150	$2.88 \cdot 10^{-2}$	$1.54 \cdot 10^{-15}$	$4.68 \cdot 10^{-2}$	$1.18 \cdot 10^{-15}$
200	$2.84 \cdot 10^{-2}$	$1.75 \cdot 10^{-15}$	$3.94 \cdot 10^{-2}$	$1.54 \cdot 10^{-15}$

Remark 2.6. We would like to point out that the staggered grid approach of [15, 19] works quite well for the system wave equation on Cartesian meshes. The vorticity transport method for the wave equation, presented in this section, acts only as a motivation of the analogous method for the shallow water system, which we consider in the next section.

3. Vorticity projection for the shallow water system. The vorticity projection algorithm for the shallow water equations (1.1) is a modification of the vorticity projection algorithm for the wave equation. The prediction step is exactly the same: The solution at the next time step is estimated using any finite volume method. We use the Rusanov and Roe schemes, although any consistent numerical flux is applicable.

As stated earlier, we use pseudovorticity as a measure of vorticity. Unlike the wave equation, where vorticity is constant in time, the pseudovorticity of the shallow water system exhibits the more complicated evolution equation (1.17). Hence, we must approximate the exact pseudovorticity Ω^{n+1} at the next time step as dictated by this equation. For this purpose we use the second-order central Nessyahu–Tadmor (NT) scheme [20]. As this is a central scheme, we avoid having to deal with the complicated wave structure of the evolution equation (1.17).

For completeness we include a description of the NT scheme. Given data $\Omega^n_{i,j}$ at time t_n, we solve for the staggered value $\Omega^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}$ at time t_{n+1},

$$\Omega^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} \approx \frac{1}{\Delta x \Delta y} \int_{J_{i,j}} \Omega(x,y,t_{n+1})\; dxdy,$$

where $J_{i,j} = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$. To go from a vertex-centered solution $\Omega^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}$ to a cell-centered solution $\Omega^{n+1}_{i,j}$, we perform a piecewise linear reconstruction of $\Omega^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}$,

$$\Omega^{n+1}(x,y) = \Omega^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} + \sigma_{i,j}(x - x_{i+\frac{1}{2}}) + \gamma_{i,j}(y - y_{j+\frac{1}{2}}) \qquad \text{for } (x,y) \in J_{i,j}.$$

This function is then averaged over $I_{i,j} = [x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}] \times [y_{j-\frac{1}{2}}, y_{j+\frac{1}{2}}]$ to obtain the final solution

$$\Omega^{n+1}_{i,j} = \frac{1}{\Delta x \Delta y} \int_{I_{i,j}} \Omega^{n+1}(x,y)\; dxdy.$$

To solve for $\Omega^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}}$, we first write the vorticity flux functions as $f(\Omega, U) = u\Omega + dm_2 + sh_y$ and $g(\Omega, U) = v\Omega - dm_1 - sh_x$, where $d = u_x + v_y$ and $s = \frac{u^2+v^2}{2}$ (cf. (1.17)). We then have

$$\begin{aligned}\Omega^{n+1}_{i+\frac{1}{2},j+\frac{1}{2}} &= \frac{1}{\Delta x \Delta y}\int_{J_{i,j}} \Omega^n(x,y)\, dxdy \\ &\quad - \frac{1}{\Delta x \Delta y}\int_{t_n}^{t_{n+1}}\int_{J_{i,j}} f(\Omega,U)_x + g(\Omega,U)_y \, dxdydt \\ &\approx \frac{1}{4}\left(\Omega^n_{i,j} + \Omega^n_{i+1,j} + \Omega^n_{i,j+1} + \Omega^n_{i+1,j+1}\right) \\ &\quad - \frac{\Delta t}{\Delta x \Delta y}\int_{J_{i,j}} f\left(\Omega^{n+\frac{1}{2}}, U^{n+\frac{1}{2}}\right)_x + g\left(\Omega^{n+\frac{1}{2}}, U^{n+\frac{1}{2}}\right)_y \, dxdy,\end{aligned}$$

where we have used a quadrature rule for the spatial integral. $\Omega^{n+\frac{1}{2}}$ and $U^{n+\frac{1}{2}}$ are approximations of Ω and U at time $t_{n+\frac{1}{2}}$. We select $U^{n+\frac{1}{2}} = \frac{1}{2}(U^n + \widetilde{U}^{n+1})$ and $\Omega^{n+\frac{1}{2}} = \Omega^n - \frac{\Delta t}{2}(f^n_x + g^n_y)$, where f^n_x and g^n_y are the gradients of the flux at time t_n.

All gradients ($\sigma_{i,j}, \gamma_{i,j}, f^n_x$ and g^n_y) used in this method are obtained using the monotonized central limiter

$$\sigma_{i,j} = \operatorname{minmod}\left(2\frac{\Omega_{i,j} - \Omega_{i-1,j}}{\Delta x},\ 2\frac{\Omega_{i+1,j} - \Omega_{i,j}}{\Delta x},\ \frac{\Omega_{i+1,j} - \Omega_{i-1,j}}{2\Delta x}\right),$$

with minmod as in (1.10c).

We are now in a position to state the vorticity projection algorithm for the shallow water system.

Step 1: *Prediction.* Compute a candidate solution $\widetilde{U}^{n+1}_{i,j}$ of (1.1) at time t_{n+1} with *any* consistent finite volume scheme (1.7). Let $\widetilde{\Omega}^{n+1}_{i,j}$ be the discrete pseudovorticity computed by

$$\widetilde{\Omega}^{n+1}_{i,j} = D_x(\widetilde{m}_2)^{n+1}_{i,j} - D_y(\widetilde{m}_1)^{n+1}_{i,j},$$

where D_x and D_y are the discrete derivatives (2.2).

Step 2: *Vorticity estimate.* Use the NT scheme to update the pseudovorticity and obtain an estimate of the pseudovorticity $\Omega^{n+1}_{i,j}$.

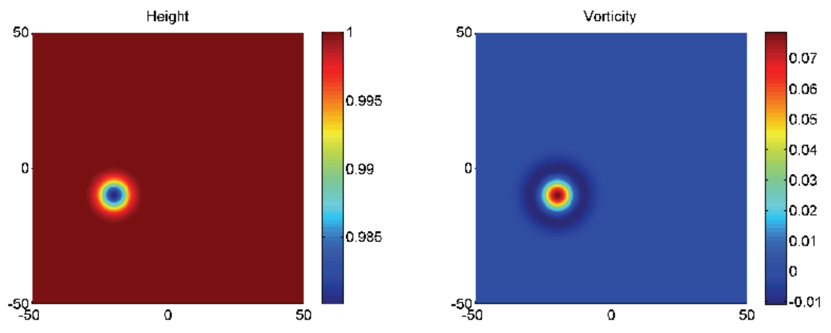
Step 3: *Elliptic solve.* Find the solution $\psi = (\psi_{i,j})$ of the discrete Poisson problem

$$\begin{cases} -\left(D_x^2 + D_y^2\right)\psi_{i,j} = \widetilde{\Omega}^{n+1}_{i,j} - \Omega^{n+1}_{i,j} & \text{in the interior,} \\ \psi_{i,j} = 0 & \text{at the boundary.}\end{cases} \tag{3.1}$$

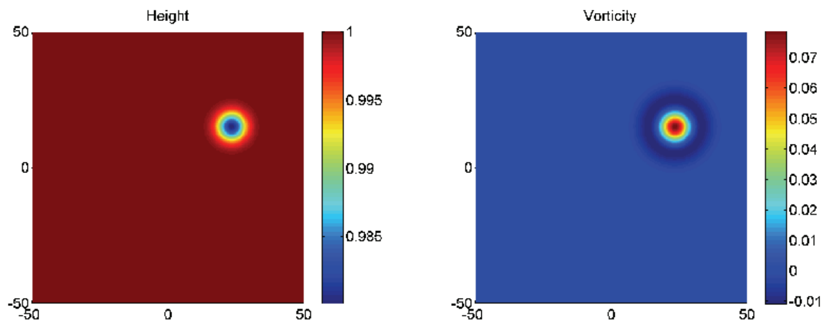
Step 4: *Projection.* Define the solution $U^{n+1}_{i,j} = \left[h^{n+1}_{i,j}, (m_1)^{n+1}_{i,j}, (m_2)^{n+1}_{i,j}\right]^\top$ at the next time step as

$$\begin{aligned} h^{n+1}_{i,j} = \widetilde{h}^{n+1}_{i,j}, \qquad (m_1)^{n+1}_{i,j} &= (\widetilde{m}_1)^{n+1}_{i,j} - D_y\psi_{i,j}, \\ (m_2)^{n+1}_{i,j} &= (\widetilde{m}_2)^{n+1}_{i,j} + D_x\psi_{i,j}. \end{aligned} \tag{3.2}$$

The vorticity projection algorithm ensures that the pseudovorticity at the next time level is equal to the estimated pseudovorticity Ω^{n+1}. The proof of this fact is identical to the proof of Lemma 2.1.



(a) $t = 0$



(b) $t = 100$

FIG. 3. *Exact solution of the vorticity advection problem at* $t = 0$ *and* $t = 100$.

3.1. Numerical experiment: Vorticity advection. We test the projection method on a problem where the exact solution is known. It can be readily checked that

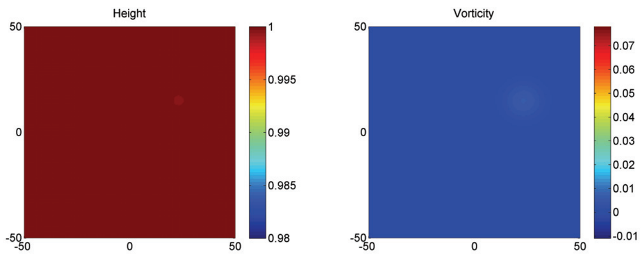
$$\begin{aligned} h(x,y,t) &= 1 - \frac{c_1^2}{4c_2 g} e^{2f}, \\ u(x,y,t) &= M\cos(\alpha) + c_1(y - y_0 - Mt\sin(\alpha))e^f, \\ v(x,y,t) &= M\sin(\alpha) - c_1(x - x_0 - Mt\cos(\alpha))e^f, \end{aligned} \tag{3.3}$$

where

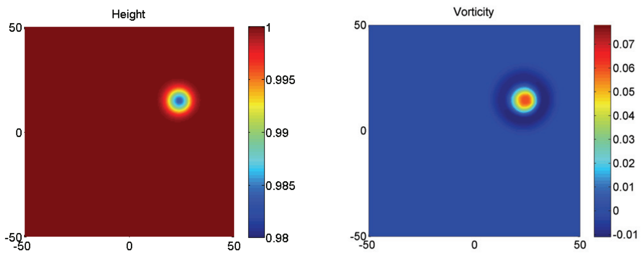
$$f = f(x,y,t) = -c_2\left((x - x_0 - Mt\cos(\alpha))^2 + (y - y_0 - Mt\sin(\alpha))^2\right),$$

gives a smooth solution $U = [h, hu, hv]^\top$ of the shallow water equations (1.1) for any choice of constants M, g, c_1, c_2, α, x_0, or y_0. The solution consists of a vortex traveling at a constant velocity M in a direction specified by the angle α. We let $M = 1/2$, $g = 1$, $c_1 = 0.04$, $c_2 = 0.02$, and $(x_0, y_0) = (-20, -10)$. To test the schemes' ability to resolve flows that are not aligned with the computational grid, we let $\alpha = \frac{\pi}{6}$. We compute for $(x, y) \in [-50, 50] \times [-50, 50]$ up to time $t = 100$. The exact solution is shown in Figure 3.

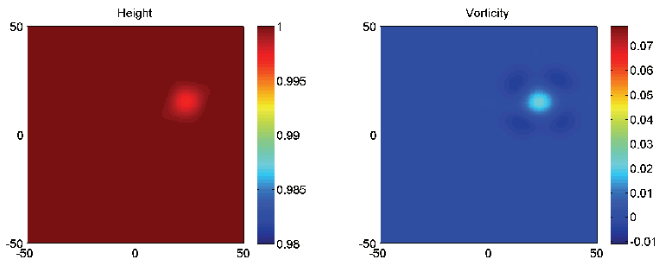
The computed solutions with the Rusanov, Roe, VPRus, and VPRoe schemes on a uniform 200×200 mesh are plotted in Figure 4. The Rusanov scheme dissipates the solution by a large amount, and the vortex is barely visible in the plot. The VPRus



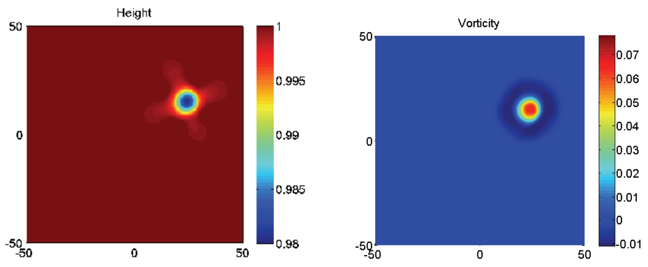
(a) Rusanov



(b) VPRus



(c) Roe



(d) VPRoe

FIG. 4. *Height and vorticity at $t = 100$ computed by the four schemes on a mesh of 200×200 grid points.*

scheme, on the other hand, preserves both the magnitude and the symmetry of the vorticity quite well, and the solution resembles the exact solution closely. The Roe scheme is less diffusive than the Rusanov scheme. Yet, the vorticity of the solution is diffused considerably. The VPRoe scheme is clearly the most accurate in this example.

The VP schemes take about twice as long to run as the prediction solver on the same grid, and, as such, a comparison of error versus grid size between the schemes

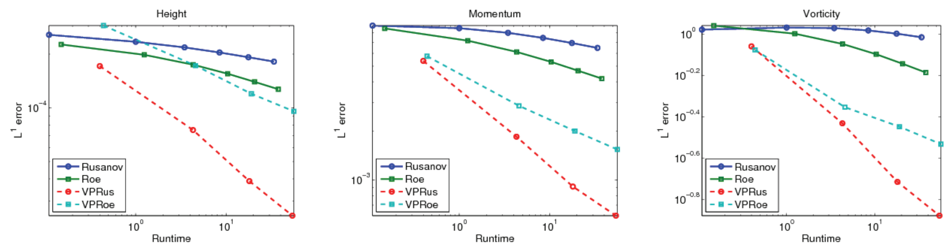


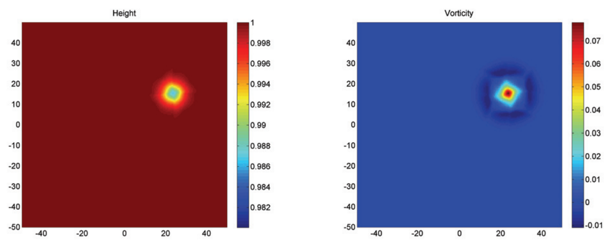
FIG. 5. *Runtime (x-axis) versus relative L^1 errors in height, momentum, and vorticity (y-axis).*

would be unfair. Instead, we compute error versus runtime over a sequence of meshes, from 50×50 to 250×250 grid points. These are plotted in Figure 5. The plots show that the VPRus and VPRoe schemes are consistently more computationally efficient than the Roe and Rusanov schemes. In fact, the VPRus scheme has the most efficient error versus runtime performance of all the schemes for the height, momentum, and vorticity variables. This experiment serves to illustrate the *considerable* gain in accuracy that results by using the vortex projection method.

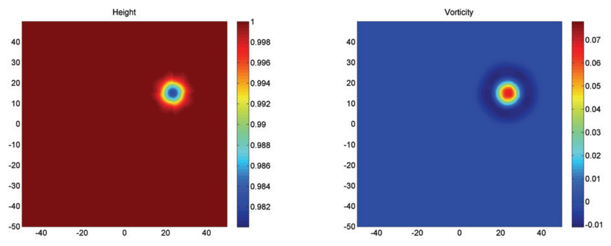
3.1.1. Second-order schemes. We repeat the vorticity advection experiment using second-order versions of the Rusanov (Rusanov2) and Roe (Roe2) schemes in the prediction step. The corresponding VP schemes are termed VPRus2 and VPRoe2, respectively. The results are shown in Figure 6. Even though the Rusanov2 scheme is more accurate than the first-order Rusanov scheme, the vortex is still dissipated. The VPRus2 scheme is much more accurate in resolving the height as well as the vorticity than the Rusanov2 scheme. The Roe2 scheme is considerably more accurate than the first-order Roe scheme, particularly in resolving the height variable. However, there are inaccuracies in resolving the vorticity variable with the Roe2 scheme. The VPRoe2 scheme is clearly more accurate than the Roe2 scheme is in resolving vorticity, thus justifying its design. There are minor differences in resolution between the VPRus2 and VPRoe2 schemes, with the VPRoe2 scheme being slightly more accurate. The computational efficiency plots are presented in Figure 7 and show error versus runtime for all the schemes. There is a difference in the behavior of computational efficiency between the first- and second-order cases. As the standard second-order schemes are more accurate, the Roe2 scheme has a slightly better asymptotic error versus runtime performance for the height variable. On the other hand, the VPRus2 and VPRoe2 schemes are more efficient (asymptotically) when the momentum and vorticity variables are taken into account. This numerical experiment clearly indicates that the VP schemes are the most computationally efficient when strongly vortical flows need to be approximated.

3.2. Shock-vortex interaction. The previous numerical experiment considered a smooth solution. However, solutions of (1.1) contain discontinuities in the form of shock waves. The interaction between shocks and vortices is quite significant in many applications. It is interesting to study the performance of the VP schemes in the presence of shocks.

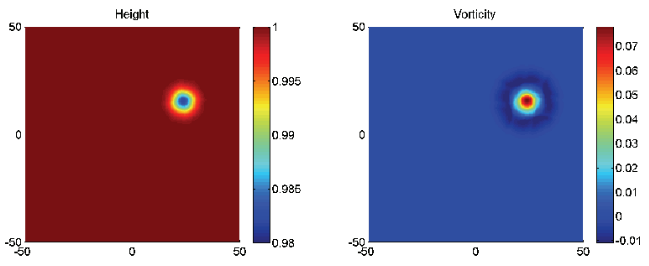
Furthermore, the presence of shocks implies that the pseudovorticity transport equation (1.17) and the vorticity transport equation (1.5) no longer hold pointwise and need to be interpreted in a weak sense and justified using a vanishing viscosity limit. It is natural to examine how the VP methods function in this case.



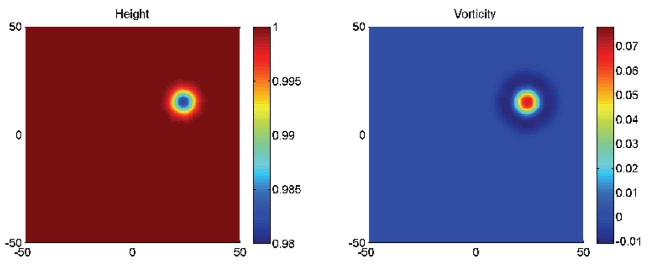
(a) Rusanov2



(b) VPRus2



(c) Roe2



(d) VPRoe2

FIG. 6. *Height and vorticity at* $t = 100$ *computed by second-order versions of the four schemes on a mesh of* 200×200 *grid points.*

We add a shock to the vortex advection problem by letting

$$h(x,y) = \begin{cases} \widetilde{h}(x,y) & \text{if } x < 20, \\ \widetilde{h}(x,y)(1+\xi) & \text{else,} \end{cases}$$

$$u(x,y) = \begin{cases} \widetilde{u}(x,y) & \text{if } x < 20, \\ \widetilde{u}(x,y) - \sqrt{\frac{1}{2} g \widetilde{h}(x,y) \xi^2 \frac{2+\xi}{1+\xi}} & \text{else,} \end{cases}$$

$$v(x,y) = \widetilde{v}(x,y),$$

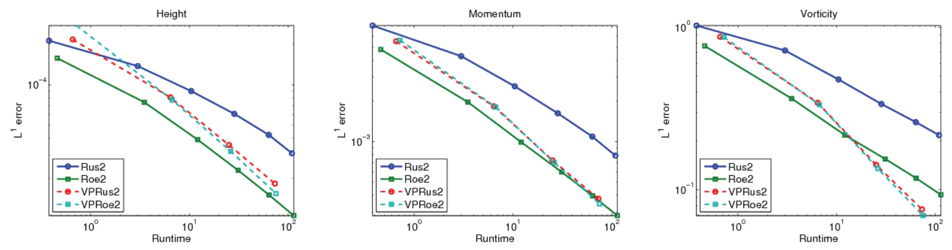


FIG. 7. *Runtime (x-axis) versus relative L^1 errors in height, momentum, and vorticity (y-axis) for second-order schemes. Note the change of axes from Figure* 5.

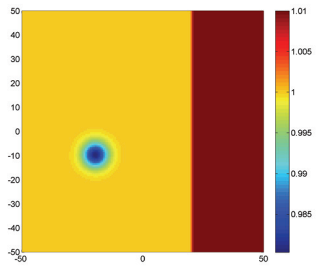


FIG. 8. *Initial height of the shock-vortex interaction problem.*

where $\widetilde{h}, \widetilde{u}$, and $\widetilde{v}$ are the initial data (3.3) of the previous experiment at $t = 0$. The initial data is displayed in Figure 8. The solution will consist of an advecting vortex, as in the previous example, moving through a left-going shock. The vortex interacts with the shock, distorts it, and emerges out of it to continue its motion. The shock continues to move to the left. The parameter $\xi > 0$ determines the strength of the shock; we set $\xi = 10^{-2}$.

We run the Rusanov, VPRus, Rusanov2, and VPRus2 schemes on a 150×150 mesh and display the results in Figure 9. Clearly, the resolution of the vortex in the VPRus scheme is by far superior to the Rusanov scheme, while the shock is resolved up to the accuracy of the predictor scheme. Similarly, the VPRus2 scheme is more accurate than the Rusanov2 scheme. The vortex is resolved to a greater degree of accuracy, and the shock is captured as accurately as the underlying high-resolution prediction scheme. This experiment clearly demonstrates that the VP schemes perform quite well, even in the presence of discontinuities and complex shock-vortex interactions.

4. Conclusion.

The shallow water equations (1.1) model a number of interesting flows in meteorology and oceanography. The system is nonlinear, and solutions contain discontinuities like shocks and interesting smooth regions like vortices. The vorticity for smooth solutions is advected with the flow (1.5). Standard finite volume schemes may not transport vorticity in the correct manner and may thus result in large vorticity errors. We address this issue and devise vorticity preserving methods. These methods are based on a projection algorithm.

We illustrate the projection method by considering the system wave equation (1.13), a linearization of the shallow water system. The equation is linear, and vorticity is constant in time. The projection method consists of computing a candidate solution with any finite volume method, which is then corrected through a projec-

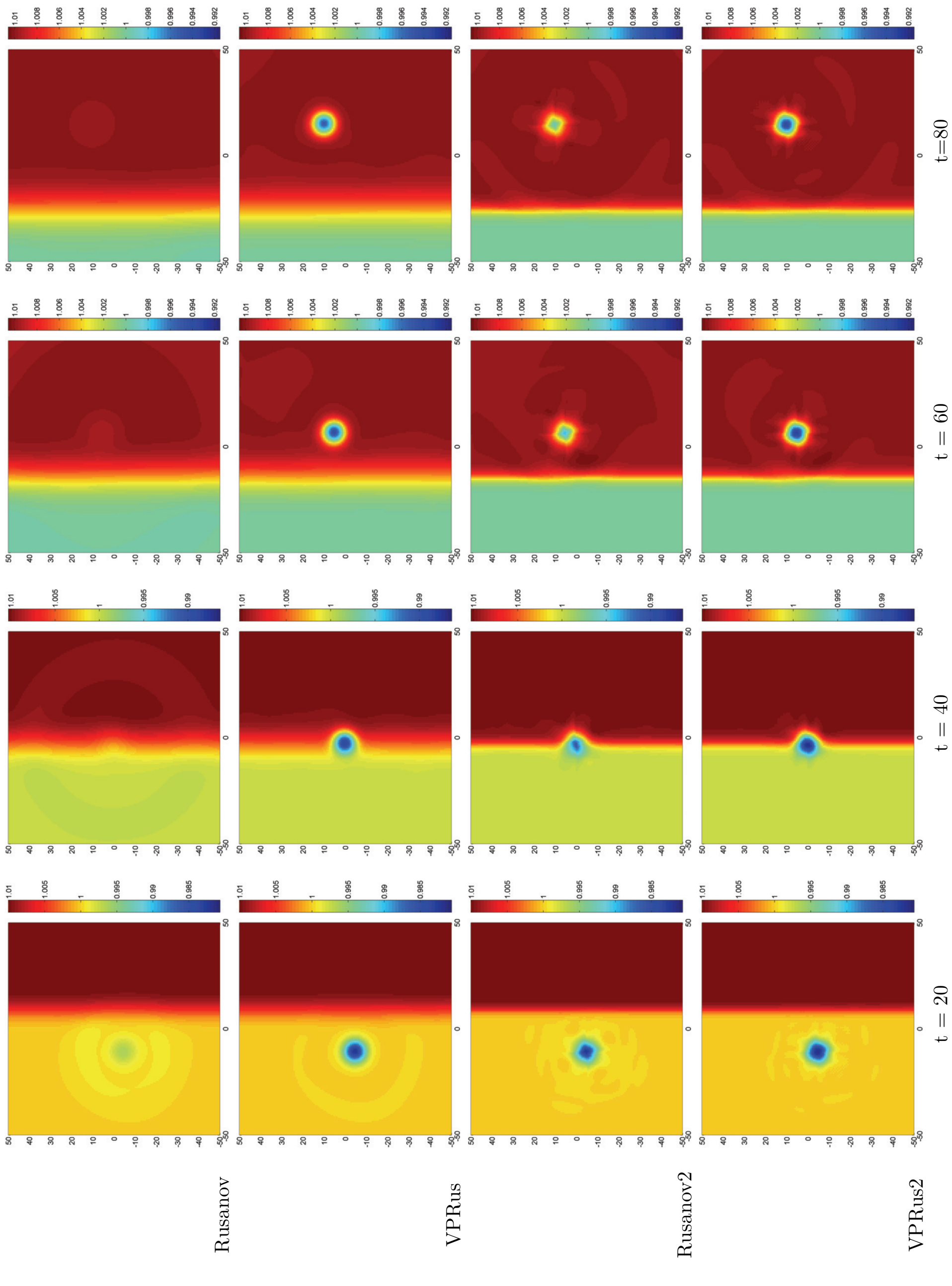


Fig. 9. *Height variable in the shock-vortex interaction problem at times* $t = 20, 40, 60, 80$.

tion step. The result is an approximate solution with an exact resolution of the vorticity field. We prove vorticity preservation and a stability estimate. Numerical experiments for the wave equation demonstrate the computational efficiency of this projection method.

The vorticity projection method is extended to the shallow water equations. The prediction step involves computing approximate solutions with a finite volume scheme. However, the vorticity may no longer be constant in time but is advected with the flow. To easily correct vorticity errors in the prediction scheme, we consider *pseudovorticity*, the curl of momentum, as a measure of vorticity. The pseudovorticity is evolved via a complicated transport equation (1.17). This equation is solved numerically at every time step with an NT central scheme. The resulting quantity serves as a good estimate for the pseudovorticity at the next time step. The momentum field is corrected by an elliptic solver, and the resulting method yields the correct pseudovorticity. The vorticity projection algorithm is tested on challenging vortex advection and shock-vortex interaction problems. The numerical results demonstrate a *considerable* gain in accuracy and computational efficiency with the vorticity projection method.

REFERENCES

[1] A. Arakawa, *Computational design for long-term numerical integration of the equations of fluid motion: Two-dimensional incompressible flow*, J. Comput. Phys., 1 (1966), pp. 119–143.
[2] A. Arakawa and V. R. Lamb, *Computational design of the basic dynamical process of the UCLA general circulation model*, Meth. Comput. Phys., 17 (1977), pp. 173–265.
[3] J. B. Bell, P. Colella, and H. M. Glaz, *A second-order projection method for the incompressible Navier-Stokes equations*, J. Comput. Phys., 85 (1989), pp. 257–283.
[4] J. U. Brackbill and D. C. Barnes, *The effect of nonzero* Div$B$ *on the numerical solution of the magnetohydrodynamic equations*, J. Comput. Phys., 35 (1980), pp. 426–430.
[5] A. J. Chorin, *Numerical solutions of the Navier-Stokes equations*, Math. Comp., 22 (1968), pp. 745–762.
[6] C. Dafermos, *Hyperbolic Conservation Laws in Continuum Physics*, Springer, Berlin, 2000.
[7] U. S. Fjordholm, S. Mishra, and E. Tadmor, *Energy preserving and energy stable schemes for the shallow water equations*, in Foundations of Computational Mathematics, London Math. Soc. Lecture Note Ser. 363, F. Cucker, A. Pinkus, and M. Todd, eds., Cambridge University Press, Cambridge, UK, 2009, pp. 93–139.
[8] U. S. Fjordholm, S. Mishra, and E. Tadmor, *Energy Preserving and Energy Stable Schemes for the Shallow Water Equations with Topography*, preprint, 2009, J. Comput. Phys., submitted.
[9] S. Gottlieb, C.-W. Shu, and E. Tadmor, *Strong stability-preserving high-order time discretization methods*, SIAM Rev., 43 (2001), pp. 89–112.
[10] A. Harten, B. Engquist, S. Osher, and S. R. Chakravarty, *Uniformly high-order accurate essentially nonoscillatory schemes*. III, J. Comput. Phys., 71 (1987), pp. 231–303.
[11] F. Ismail and P. L. Roe, *Towards a vorticity preserving second order finite volume scheme solving the Euler equations*, in Proceedings of the 17th AIAA CFD Conference, Toronto, Canada, 2005.
[12] R. Jeltsch and M. Torrilhon, *On curl-preserving finite volume discretizations for shallow water equations*, BIT, 46 (2006), pp. S35–S53.
[13] A. Kurganov and E. Tadmor, *New high resolution central schemes for non-linear conservation laws and convection-diffusion equations*, J. Comput. Phys., 160 (2000), pp. 241–282.
[14] B. van Leer, *Towards the ultimate conservative difference scheme*, V. *A second order sequel to Godunov's method*, J. Comput. Phys., 32 (1979), pp. 101–136.
[15] A. Lerat, F. Falissard, and J. Sides, *Vorticity-preserving schemes for the compressible Euler equations*, J. Comput. Phys., 225 (2007), pp. 635–651.
[16] R. J. LeVeque, *Finite Volume Methods for Hyperbolic Problems*, Cambridge University Press, Cambridge, UK, 2002.
[17] T. Lyche, *Lecture Notes for Numerical Linear Algebra*, http://www.uio.no/studier/emner/matnat/ifi/INF-MAT4350/h10/book2010.pdf.

[18] S. Mishra and E. Tadmor, *Constraint Preserving Schemes Using Potential-Based Fluxes. II. Genuinely Multi-dimensional Central Schemes for Systems of Conservation Laws*, preprint, 2009, SIAM J. Numer. Anal., submitted.

[19] K. W. Morton and P. L. Roe, *Vorticity-preserving Lax–Wendroff-type schemes for the system wave equation*, SIAM. J. Sci. Comput., 23 (2001), pp. 170–192.

[20] H. Nessyahu and E. Tadmor, *Non-oscillatory central difference schemes for hyperbolic conservation laws*, J. Comput. Phys., 87 (1990), pp. 408–463.

[21] C. W. Shu and S. Osher, *Efficient implementation of essentially non-oscillatory schemes*—II, J. Comput. Phys., 83 (1989), pp. 32–78.

[22] G. Toth, *The* Div$B = 0$ *constraint in shock capturing magnetohydrodynamics codes*, J. Comput. Phys., 161 (2000), pp. 605–652.