DIMENSIONAL SPLITTING WITH FRONT TRACKING AND ADAPTIVE GRID REFINEMENT

K.–A. LIE, V. HAUGSE, AND K. HVISTENDAHL KARLSEN

Abstract. Front tracking in combination with dimensional splitting is analysed as a numerical method for scalar conservation laws in two space dimensions. An analytic error bound is derived, and convergence rates based on numerical experiments are presented. Numerical experiments indicate that large CFL numbers can be used without loss of accuracy for a wide range of problems. A new method for grid refinement is introduced. The method easily allows for dynamical changes in the grid, using, for instance, the total variation in each grid cell as a criterion for introducing new or removing existing refinements. Several numerical examples are included, highlighting the features of the numerical method. A comparison with a high-resolution method confirms that dimensional splitting with front tracking is a highly viable numerical method for practical computations.

1. Introduction

Dimensional splitting, or fractional steps, has been widely used to extend one-dimensional numerical methods to multi-dimensional problems. The method was first introduced by Godunov [5] in connection with gas dynamics and later extended by Strang [25]. The idea is quite simple. Consider the two-dimensional scalar conservation law

$$u_t + f(u)_x + g(u)_y = 0, \qquad u(x,y,0) = u_0(x,y). \tag{1}$$

Let $u(t) = \mathcal{S}(t)u_0$ denote the entropy solution of (1), and $v(t) = \mathcal{S}^f(t)v_0$ and $w(t) = \mathcal{S}^g(t)w_0$ the entropy solutions of the one-dimensional equations

$$\begin{aligned} v_t + f(v)_x = 0, &\qquad v(x,y,0) = v_0(x), \\ w_t + g(w)_y = 0, &\qquad w(x,y,0) = w_0(y). \end{aligned}$$

Then the first order Godunov splitting is defined as

$$\mathcal{S}(n\Delta t)u_0(x,y) \cong \left[\mathcal{S}^g(\Delta t)\mathcal{S}^f(\Delta t)\right]^n u_0(x,y), \tag{2}$$

while for Strang's second order method the approximation is

$$\mathcal{S}(n\Delta t)u_0(x,y) \cong \left[\mathcal{S}^f(\Delta t/2)\mathcal{S}^g(\Delta t)\mathcal{S}^f(\Delta t/2)\right]^n u_0(x,y). \tag{3}$$

The order of the operators $\mathcal{S}^f$ and $\mathcal{S}^g$ could be reversed, of course.

When used as a numerical method, the one-dimensional solution operators are approximated by some numerical scheme. Crandall and Majda [3] proved that both splitting methods produce an approximation that converges to the unique entropy solution as temporal

Key words and phrases. Scalar conservation laws, dimensional splitting, front tracking, error bounds, adaptive grid refinement.

Lie has been supported by the Research Council of Norway under grant 100555/410. Karlsen has been supported by VISTA, a research cooperation between the Norwegian Academy of Science and Letters and Den norske stats oljeselskap a.s. (Statoil).

and spatial discretization tend to zero, when using either monotone schemes, a modified Lax–Wendroff scheme, or Glimm's method in each step.

Teng [26] proved that the accuracy of the splitting methods (2) and (3) are of order $\mathcal{O}(\Delta t^{1/2})$. We will derive a similar error bound for the method (2) when the one-dimensional solution operators are approximated by *front tracking* [8]. This unconditionally stable numerical method was proposed by Holden and Risebro [9]. We give some numerical examples with accuracy close to $\mathcal{O}(\Delta t)$, indicating that the error bounds are "worst case".

Numerical methods based on dimensional splitting are very computer efficient and easy to implement, but there are also some objections to such methods [24]. Higher-dimensional effects, which are important locally, may be poorly represented, and grid effects due to the introduced directional bias may be strong. However, in many applications the inherent dynamics in the scalar equation *can* be resolved accurately by only a few splitting steps. An example is water injection in oil reservoirs, where resolving sharp changes in water saturation due to non-mixing of water and oil has great impact on the quality of the computed solution.

Another objection to dimensional splitting is that complicated boundary conditions may be difficult to include properly. This is not the case for outflow boundaries, which are very simple to implement. Outflow conditions occur in many problems, for instance when the computational domain is a truncation of a larger physical domain, where it is important that waves leave the domain without non-physical reflections.

The front tracking method has no stability restriction on the time steps. Numerical experiments in §4 show that the accuracy is improved by increasing the CFL number moderately above unity. Similar observations were made by LeVeque [17] for his related shock-capturing method. Compared to first order Godunov's method, discontinuous parts are resolved remarkably better. We also observe that the CFL number can be increased even further without *significant* loss in accuracy. Using large time steps means increasing the efficiency. So, despite the objections to dimensional splitting, the front tracking method may be a preferred choice for practical computations on scalar equations due to its efficiency and ability to correctly track discontinuities.

To improve the spatial accuracy, we incorporate local grid refinement in the front tracking method. The nontrivial part is how to treat the interface between ordinary and refined grid blocks. Having settled this, we discuss a strategy for adaptive grid refinement. An intuitive method for dimensional splitting is to use properties of the solution after one step to generate a grid for the next step. The total variation in each grid block is a natural measure when determining changes in the grid.

The outline of the paper is as follows. In §2 we review the one-dimensional front tracking method and its extension to multi-dimensional problems by means of dimensional splitting and derive an error estimate for this numerical method. Then in §3, local grid refinement is incorporated into the method, and a strategy for adaptive refinement is discussed. Finally in §4, we present convergence rates based on numerical examples. Furthermore, we show the gain in efficiency by using large CFL numbers and highlight several features of the new strategy for adaptive grid refinement.

2. Dimensional Splitting with Front Tracking

We begin by briefly describing the front tracking method that was first introduced by Dafermos [4], and later proved well-defined and developed into a numerical method for one-dimensional scalar conservation laws by Holden, Holden, and Høegh–Krohn [8].

Assume that v_0 is piecewise constant and that f is piecewise linear. Using front tracking, the solution of

$$v_t + f(v)_x = 0, \qquad v(x,0) = v_0(x)$$

is constructed as follows: Each jump in the initial data v_0 defines a Riemann problem, which is solved exactly. Since the flux function is piecewise linear, each Riemann problem leads to a series of discontinuities propagating in the (x,t)–plane. The global solution is obtained by connecting the solutions of the local Riemann problems. This solution is well-defined until two or more discontinuities interact at some point. Then we have what is called a *shock collision*. A shock collision defines a new Riemann problem with left and right states given by the values immediately to the left and to the right of the colliding discontinuities. By solving this Riemann problem, the global solution is determined until the next shock collision, and so on. We may continue in this fashion and thereby advance the (exact) solution up to any positive time.

In the general case (arbitrary f and v_0), the front tracking method consists in replacing the flux function f by a continuous, piecewise linear approximation and the initial data v_0 by a piecewise constant approximation. Then this perturbed problem is solved according to the procedure outlined above. For a more detailed description we refer the reader to Holden *et al.* [8] or Langseth [14].

The front tracking idea can be extended to multi-dimensional problems by the means of dimensional splitting [9]. For simplicity we consider the two-dimensional case, since a generalization to higher dimensions is straightforward. Assume that the piecewise constant initial function is given on an underlying rectangular grid with n columns and m rows. Each row or column defines a sequence of Riemann problems. (A single row or column will in the following be referred to as a *tube*). In the first step the solution is computed up to time Δt for all tubes in one direction, say x, using the procedure outlined above. The solution is piecewise constant in each tube, but not necessarily in each grid block. Accordingly, the solution must be projected back onto the grid to give proper initial data for the next step. Afterwards, the solution can be computed for each tube in the other direction, and so on.

Now we formalize our discussion. Consider a uniform Cartesian grid defined by the nodes $\{(i\Delta x, j\Delta y)\}$, where $\Delta x, \Delta y$ are given positive numbers and $i, j \in \mathbf{Z}$. Let π be the usual grid block averaging operator defined on this grid, that is,

$$\pi u(x,y) = \frac{1}{\Delta x \Delta y} \int_{z_{i,j}} u(\tilde{x}, \tilde{y})\, d\tilde{x} d\tilde{y}, \qquad \forall (x,y) \in z_{i,j}, \tag{4}$$

where $z_{i,j}$ is grid block number (i,j) with lower left-hand corner in $(i\Delta x, j\Delta y)$. Furthermore, let f_δ and g_δ be piecewise linear approximations to f and g, respectively, and $\mathcal{S}^{f_\delta}(t)$ and $\mathcal{S}^{g_\delta}(t)$ the corresponding one-dimensional solution operators. Using Godunov splitting, the dimensional splitting solution $\{u_\eta^n\}_{n=1}^N$ is defined as

$$u_\eta^n(x,y) = [\pi \mathcal{S}^{g_\delta}(\Delta t) \pi \mathcal{S}^{f_\delta}(\Delta t)]^n u_\eta^0(x,y), \tag{5}$$

where $u_\eta^0(x,y) = \pi u_0(x,y)$ and $\eta = (\Delta x, \Delta y, \delta)$. The approximate solution is defined analogously for the Strang splitting.

2.1. **An Error Bound.** We shall derive an error bound for the splitting method (5) using the classical approximation theory of Kuznetsov [13]. This theory has already been used to estimate the accuracy of a large class of approximate methods: the method of vanishing viscosity, the method of smoothing, and several widely used finite difference methods [13]; the methods of Glimm, Godunov, and LeVeque [19]; and the one-dimensional front tracking method [20]. See also the recent work of Cockburn and Gremaud [2] and the historical overview of connected works therein.

In the analysis we have to work with approximate solutions as functions of t, and not merely defined at discrete times $t_n = n\Delta t$. Motivated by Crandall and Majda [3], we define the interpolation

$$u_\eta(x,y,t) = \begin{cases} \mathcal{S}^{f_\delta}\big(2(t-t_n)\big)u_\eta^n(x,y), & \text{for } t \in [t_n, t_{n+1/2}), \\ \mathcal{S}^{g_\delta}\big(2(t-t_{n+1/2})\big)u_\eta^{n+\frac12}(x,y), & \text{for } t \in [t_{n+1/2}, t_{n+1}). \end{cases}$$

Recall that the splitting solutions are defined by

$$\begin{aligned} u_\eta^n &= \left[\pi\mathcal{S}^{g_\delta}(\Delta t)\pi\mathcal{S}^{f_\delta}(\Delta t)\right]^n u_\eta^0(x,y), \\ u_\eta^{n+\frac12} &= \pi\mathcal{S}^{f_\delta}(\Delta t)\left[\pi\mathcal{S}^{g_\delta}(\Delta t)\pi\mathcal{S}^{f_\delta}(\Delta t)\right]^n u_\eta^0(x,y). \end{aligned}$$

The solution operators $\mathcal{S}^{f_\delta}(t)$ and $\mathcal{S}^{g_\delta}(t)$ are L^1-contractions, which implies that the spatial variation of the splitting solutions is non-increasing in time. Moreover, if A denotes the common Lipschitz constant associated with the flux functions f and g, the splitting solutions satisfy the uniform temporal estimate

$$\|u_\eta(\cdot,\cdot,t_2) - u_\eta(\cdot,\cdot,t_1)\|_1 \le \left[A|t_2-t_1| + C(t_1,t_2)(\Delta x + \Delta y)\right]|u_0|_{BV(\mathbf{R}^2)}, \tag{6}$$

where $C(t_1,t_2)$ is the smallest integer greater than or equal to $2|t_2-t_1|/\Delta t$.

Let $\omega_\varepsilon(x) = \frac{1}{\varepsilon}\omega(\frac{x}{\varepsilon})$ be a usual $C_0^\infty(-\varepsilon,\varepsilon)$ unit mass mollifier, and let $\Omega^{\varepsilon,\varepsilon_0}(x,y,t)$ be a smooth approximation to $\delta(x)\delta(y)\delta(t)$ defined by

$$\Omega^{\varepsilon,\varepsilon_0}(x,y,t) = \omega_\varepsilon(x)\,\omega_\varepsilon(y)\,\omega_{\varepsilon_0}(t).$$

Let $D_T = \mathbf{R}^2\times[0,T]$ and $u,v : D_T \to \mathbf{R}$, and introduce the so-called Krŭzkov form

$$\begin{aligned} \mathcal{D}(\phi,u,a) = &\iint_{D_T} \left[U(u,a)\phi_t + F(u,a)\phi_x + G(u,a)\phi_y\right] dt dx dy \\ &+ \int_{\mathbf{R}^2} U(u,a)\phi\big|_{t=0}\, dx dy - \int_{\mathbf{R}^2} U(u,a)\phi\big|_{t=T^-}\, dx dy, \end{aligned}$$

and the integrated form

$$\mathcal{D}^{\varepsilon,\varepsilon_0}(u,v) = \iint_{D_T} \mathcal{D}(\Omega^{\varepsilon,\varepsilon_0}(\cdot-\xi,\cdot-\eta,\cdot-\tau), u(\cdot,\cdot,\cdot), v(\xi,\eta,\tau))\, d\tau d\xi d\eta.$$

Here (U,F,G) is the usual convex entropy associated with (1); that is, $U(u,a) = |u-a|$, $F(u,a) = \operatorname{sign}(u-a)[f(u)-f(a)]$, and $G(u,a) = \operatorname{sign}(u-a)[g(u)-g(a)]$. Krŭzkov [12] defines $u(t)$ as the (unique) entropy solution of (1), provided $\mathcal{D}(\phi,u,a) \ge 0$ for all suitable positive test functions ϕ and real numbers a.

For later use, let $u_\delta(t) = \mathcal{S}_\delta(t)u_0$ be the entropy solution of

$$u_t + f_\delta(u)_x + g_\delta(u)_y = 0, \qquad u(x,y,0) = u_0(x,y).$$

By a straightforward two-dimensional extension of Lucier's stability result [19], it follows that

$$\|\mathcal{S}(T)u_0 - \mathcal{S}_\delta(T)u_0\|_1 \le T\max\left(\|f-f_\delta\|_{\mathrm{Lip}}, \|g-g_\delta\|_{\mathrm{Lip}}\right)|u_0|_{BV(\mathbf{R}^2)}. \tag{7}$$

According to Kuznetsov's lemma [13] the following bound is valid,

$$\|\mathcal{S}_\delta(T)u_0 - u_\eta^N\|_1 \leq \|u_0 - u_\eta^0\|_1 + \left[2(\varepsilon + A\varepsilon_0) + C(\Delta x + \Delta y)\right]|u_0|_{BV(\mathbf{R}^2)} - \mathcal{D}^{\varepsilon,\varepsilon_0}(u_\eta, u_\delta), \tag{8}$$

where C is the smallest integer greater than or equal to $2\varepsilon_0/\Delta t$. Consequently, we are in a position to prove the theorem:

Theorem 1 (Error Bound)**.** *If $u_0 \in L^1 \cap L^\infty \cap BV$, and f and g are Lipschitz continuous functions with constant A, then*

$$\begin{aligned} \|\mathcal{S}(T)u_0 - u_\eta^N\|_1 \leq \|u_0 - u_\eta^0\|_1 + T\max\left(\|f - f_\delta\|_{Lip}, \|g - g_\delta\|_{Lip}\right)|u_0|_{BV(\mathbf{R}^2)} \\ + |u_0|_{BV(\mathbf{R}^2)} \min_{\varepsilon_0,\varepsilon} \Lambda(\varepsilon_0, \varepsilon; \Delta x, \Delta y, \Delta t), \end{aligned} \tag{9}$$

where Λ is given by (11)*. In particular, if $\Delta x = K_1\Delta y = K_2\Delta t$ for some constants K_1, K_2, then $\|\mathcal{S}(T)u_0 - u_\eta^N\|_1 = \mathcal{O}(\sqrt{\Delta t} + \sqrt{\Delta x} + \delta)$.*

Proof. Having the preceding discussion in mind, it remains to bound the term $\|\mathcal{S}_\delta(T)u_0 - u_\eta^N\|_1$, that is, the discrepancy $\mathcal{D}^{\varepsilon,\varepsilon_0}(u_\eta, u_\delta)$. Taking advantage of the fact that the splitting solution $u_\eta(t)$ satisfies "local" one-dimensional Krŭzkov forms (one in each space direction) we obtain

$$-\mathcal{D}(\phi, u_\eta, a) \leq \mathcal{I}_1 + \mathcal{I}_2 + \mathcal{I}_3 + \mathcal{I}_4, \tag{10}$$

where $\mathcal{I}_i = \mathcal{I}_i(\phi, u_\eta, a)$ are defined by

$$\begin{aligned} \mathcal{I}_1 &= 2\sum_{n=0}^{N-1} \int_{\mathbf{R}^2}\int_{t_n}^{t_{n+1/2}} F(u_\eta, a)\phi_x dt dx dy - \iint_{D_T} F(u_\eta, a)\phi_x \, dt dx dy \\ \mathcal{I}_2 &= 2\sum_{n=0}^{N-1} \int_{\mathbf{R}^2}\int_{t_{n+1/2}}^{t_{n+1}} G(u_\eta, a)\phi_y \, dt dx dy - \iint_{D_T} G(u_\eta, a)\phi_y \, dt dx dy \\ \mathcal{I}_3 &= -\sum_{n=1}^{N-1} \int_{\mathbf{R}^2} \left[U(u_\eta, a)\big|_{t=t_n^-} - U(u_\eta, a)\big|_{t=t_n^+}\right]\phi(x, y, t_n)\, dx dy \\ \mathcal{I}_4 &= -\sum_{n=0}^{N-1} \int_{\mathbf{R}^2} \left[U(u_\eta, a)\big|_{t=t_{n+1/2}^-} - U(u_\eta, a)\big|_{t=t_{n+1/2}^+}\right]\phi(x, y, t_{n+1/2})\, dx dy. \end{aligned}$$

We omit the derivation of inequality (10), since Crandall and Majda [3] have derived an analogous inequality in the case of exact one-dimensional solution operators. With $\mathcal{I}_i^{\varepsilon,\varepsilon_0} = \mathcal{I}_i^{\varepsilon,\varepsilon_0}(u_\eta, u_\delta)$ defined analogously to $\mathcal{D}^{\varepsilon,\varepsilon_0} = \mathcal{D}^{\varepsilon,\varepsilon_0}(u_\eta, u_\delta)$, we observe that $-\mathcal{D}^{\varepsilon,\varepsilon_0} \leq \sum_{i=1}^4 \mathcal{I}_i^{\varepsilon,\varepsilon_0}$. We start by estimating the error terms $\mathcal{I}_1^{\varepsilon,\varepsilon_0}$ and $\mathcal{I}_2^{\varepsilon,\varepsilon_0}$, which are due to the operator splitting. Substituting

$$F(u_\eta(t), u_\delta(\tau)) = F(u_\eta(t_{n+1/2}), u_\delta(\tau)) + \left[F(u_\eta(t), u_\delta(\tau)) - F(u_\eta(t_{n+1/2}), u_\delta(\tau))\right]$$

into $\mathcal{I}_1^{\varepsilon,\varepsilon_0}$ yields

$$\mathcal{I}_1^{\varepsilon,\varepsilon_0} = \sum_{n=0}^{N-1}\left(\left[J_1^n(t_n, t_{n+1/2}) - J_1^n(t_{n+1/2}, t_{n+1})\right] + \left[J_2^n(t_n, t_{n+1/2}) - J_2^n(t_{n+1/2}, t_{n+1})\right]\right),$$

where

$$J_1^n(t_1,t_2) = \iint_{D_T}\int_{\mathbf{R}^2}\int_{t_1}^{t_2} F(u_\eta(t_{n+1/2}), u_\delta(\tau))\,\Omega_x^{\varepsilon,\varepsilon_0} dt,$$

$$J_2^n(t_1,t_2) = \iint_{D_T}\int_{\mathbf{R}^2}\int_{t_1}^{t_2} \big[F(u_\eta(t), u_\delta(\tau)) - F(u_\eta(t_{n+1/2}), u_\delta(\tau))\big]\,\Omega_x^{\varepsilon,\varepsilon_0} dt.$$

We recall that F is Lipschitz continuous with constant A and write $\omega_{\varepsilon_0}(t-\tau)$ as

$$\omega_{\varepsilon_0}(t-\tau) = \omega_{\varepsilon_0}(t_{n+1/2}-\tau) + \int_{t_{n+1/2}}^{t} \omega'_{\varepsilon_0}(z-\tau)\,dz.$$

Applying standard arguments and the well-known properties of the mollifier function (see [13] and e.g., [16]), it is not difficult to see that the x-derivative of $F(u_\eta(t_{n+1/2}), u_\delta(\tau))$ satisfies

$$|J_1^n(t_n, t_{n+1/2}) - J_1^n(t_{n+1/2}, t_{n+1})| \le \frac{AM_\omega}{2}\frac{\Delta t^2}{\varepsilon_0}\left[\int_{\mathbf{R}} |u_\eta^{n+\frac{1}{2}}(\cdot,y)|_{BV(\mathbf{R})}\,dy\right],$$

where M_ω is an upper bound on ω. Next, by exploiting the fact that u_η is Lipschitz continuous in the time variable (see (6)), we deduce the estimate

$$|J_2^n(t_n, t_{n+1/2}) - J_2^n(t_{n+1/2}, t_{n+1})| \le \left[\frac{A^2 M_\omega}{2}\frac{\Delta t^2}{\varepsilon} + AM_\omega \frac{\Delta t \Delta x}{\varepsilon}\right]|u_\eta^n|_{BV(\mathbf{R}^2)}.$$

The last term inside the brackets comes from the projection step at $t = t_{n+1/2}$. The analysis and the estimates are similar for $\mathcal{I}_2^{\varepsilon,\varepsilon_0}$, and from to the spatial estimate $|u_\eta(t)|_{BV(\mathbf{R}^2)} \le |u_0|_{BV(\mathbf{R}^2)}$, it follows that

$$|\mathcal{I}_1^{\varepsilon,\varepsilon_0}| + |\mathcal{I}_2^{\varepsilon,\varepsilon_0}| \le \frac{AM_\omega}{2}\frac{T\Delta t}{\varepsilon_0}|u_0|_{BV(\mathbf{R}^2)} + AM_\omega \frac{T\big[A\Delta t + \Delta x + \Delta y\big]}{\varepsilon}|u_0|_{BV(\mathbf{R}^2)}.$$

It now remains to estimate the error terms $|\mathcal{I}_3^{\varepsilon,\varepsilon_0}|$ and $|\mathcal{I}_4^{\varepsilon,\varepsilon_0}|$ due to the projection operator π. Recall that $\mathcal{I}_3^{\varepsilon,\varepsilon_0}$ takes the form (similar formula for $\mathcal{I}_4^{\varepsilon,\varepsilon_0}$)

$$\mathcal{I}_3^{\varepsilon,\varepsilon_0} = \sum_{n=1}^{N-1} \iint_{D_T}\int_{\mathbf{R}^2} \big\{|u_\eta(x,y,t_n^+) - u(\xi,\eta,\tau)| - |u_\eta(x,y,t_n^-) - u(\xi,\eta,\tau)|\big\}\Omega^{\varepsilon,\varepsilon_0},$$

where $u_\eta(t_n^+)$ is the projection of $u_\eta(t_n^-)$ defined in (4). Lucier [19, p. 1076] proposed a technique for estimating the one-dimensional versions of $\mathcal{I}_3^{\varepsilon,\varepsilon_0}$ and $\mathcal{I}_4^{\varepsilon,\varepsilon_0}$. By extending this technique to several space dimensions, the following estimate can be derived,

$$|\mathcal{I}_3^{\varepsilon,\varepsilon_0}| + |\mathcal{I}_4^{\varepsilon,\varepsilon_0}| \le \frac{(\Delta x + \Delta y)^2}{6}\frac{T}{\Delta t}\frac{\|\omega'\|_1}{\varepsilon}|u_0|_{BV(\mathbf{R}^2)}.$$

By letting $\omega \to \frac{1}{2}(1+x)\chi_{[-1,0)} + \frac{1}{2}(1-x)\chi_{[0,1]}$, the constant $\|\omega'\|_1$ may be chosen arbitrarily close to 1 and M_ω close to $\frac{1}{2}$. We have thereby obtained a proper bound on $|\mathcal{D}^{\varepsilon,\varepsilon_0}(\Omega^{\varepsilon,\varepsilon_0}, u_\eta, u)|$. Taking (8) into account, all terms involving ε_0 and ε can be summarized in

$$\begin{aligned}\Lambda(\varepsilon_0, \varepsilon; \Delta x, \Delta y, \Delta t) =& \frac{1}{4}AT\frac{\Delta t}{\varepsilon_0} + \frac{1}{2}AT\frac{A\Delta t + \Delta x + \Delta y}{\varepsilon} \\ &+ 2(A\varepsilon_0 + \varepsilon) + \left\lceil \frac{2\varepsilon_0}{\Delta t}\right\rceil(\Delta x + \Delta y) + \frac{1}{6}\frac{T}{\Delta t}\frac{(\Delta x + \Delta y)^2}{\varepsilon},\end{aligned} \tag{11}$$

and we have proved (9). Minimizing Λ over ε_0 and ε gives some function $\Gamma(\Delta x, \Delta y, \Delta t)$. If $\Delta x = K_1 \Delta y = K_2 \Delta t$, and we choose for instance $\varepsilon_0 = \sqrt{\Delta t}$ and $\varepsilon = \sqrt{\Delta x}$, then $\Gamma \leq C_1\sqrt{\Delta t} + C_2\sqrt{\Delta x}$ for some constants C_1 and C_2. □

Remark 1. The error bound suggests a convergence rate of order $\frac{1}{2}$ in both time and space. The method of estimation is the best available in the literature for this type of schemes. Unfortunately it gives very crude bounds, which are not particularly useful as *a priori* error estimates. We observed a tendency to overestimate the actual error by several orders of magnitude, and for the test cases we have run (also others not reported here) the observed rates were much closer to 1 than $\frac{1}{2}$, see §4.

Remark 2. When the method is used as a computational tool, the time step must somehow be related to the space step. As seen in the error analysis, there are two factors contributing to the total error; the errors introduced by the splitting and by the projection. The error bound gives little information on how to choose the time step to balance these two error sources at a minimum. Dimensional splitting is unconditionally stable in the sense that there is no restriction on the discretization parameters, which means that large time steps can be taken. Although this is favourable in terms of computer efficiency, the splitting error will certainly increase with increasing time step. On the other hand, increasing time steps means fewer projections and hence less projection error. The question on how to choose the time step will therefore be investigated numerically in §4.

Remark 3. Finally, note that if the solution is smooth, the calculations above together with integration by parts implies first order convergence.

3. Local Grid Refinement

An underlying grid is introduced to obtain piecewise constant data in one direction from the solution computed in the other. When projecting the solution back onto the grid, the fine structures in the approximate solution is lost and errors introduced; large shocks are smeared, and sequences of small discontinuities (approximating rarefactions) are made coarser. By introducing some kind of grid refinement, we can hope to improve the projection, retaining more of the fine structure, and thus reduce the error.

The commercial reservoir simulator Frontsim [1] has a refinement strategy where tubes follow the position of major discontinuities. With this approach, the projection operator defines the density of tubes in one direction based on the solution computed in the other. Accordingly, the position of major discontinuities are kept.

The strategy in Frontsim is not local, and computational effort may be spent in regions where the physical process is of little interest. In view of this problem, we consider another approach, where the grid is refined by local partition of blocks in a fixed (coarse) grid. This partition can remain fixed throughout the computation, or be determined by for instance the projection algorithm. An example of a refined grid is shown in Figure 1.

The proposed grid structure requires a reformulation of the front tracking algorithm, because a tube of coarse grid blocks now may contain several parallel tubes of fine grid blocks. *Inside* each tube, refined or coarse, the front tracking algorithm can be applied without modifications, but the interface between coarse and refined grid blocks must be treated separately. This will be discussed later. First, however, it is natural to introduce a suitable data structure to use in the front tracking algorithm.

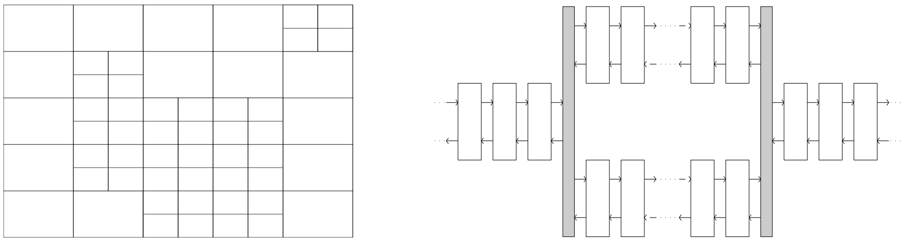

FIGURE 1. (Left) Grid refinement by local partitioning of a regular Cartesian grid. (Right) Part of a front list containing two parallel refined tubes. The shaded boxes represents the static fronts.

3.1. **Data Structure.** Risebro and Tveito [22, 23] proposed a data structure for ordinary front tracking, in which discontinuities are represented by two linked lists of objects (referred to as fronts). A front object contains the left and right state of the front, the starting point, the speed of propagation, and the point where the front possibly collides with its right hand neighbour. All fronts are organized in a spatial list (X-list) from left to right. To keep track of possible collisions, the leftmost of two colliding fronts is kept in a time list (T-list) sorted in ascending order. Note that all fronts are stored only once; the X- and T-list refer to the same objects.

In each coarse section of a tube there will be an X-list, and similarly an X-list for each tube in refined sections. These lists must be connected at the interface between coarse and refined blocks. For this purpose we introduce an artificial *static front*; that is, a front with zero speed of propagation containing an array, where each element again is a pointer to the first or last front in an adjacent refined tube. A schematic representation of the data structure is given in Figure 1.

It is possible to have several levels of refinement; that is, a refined block may contain refined grid blocks. The refinement factor can also differ from block to block. For the rest of the discussion we assume only one level of refinements and a uniform refinement factor in each direction. Having the data structure, we focus on how to solve Riemann problems at an interface and how to resolve possible collisions involving static fronts.

3.2. **The Riemann Problem at an Interface.** At an interface, a Riemann problem is in general two-dimensional, but since the flux is nonzero only along the tube, the two-dimensional problem may be decomposed into a series of simple Riemann problems, one for each refined tube. Suppose that the refined part is to the left of the interface and the coarse part to the right. Let u^c denote the initial state immediately inside the coarse tube and u^i the state immediately inside fine tube number i. Each of these Riemann problems, defined by u^c and u^i, will in general give waves propagating in both directions, into the coarse tube and into the refined tubes. Waves with a negative speed can simply be inserted into the corresponding X-list. Waves with positive speed propagate into the coarse part and must be treated differently.

A simple approach would be to compute some representative average $\bar{u}$ of all states u^i immediately inside the refined tubes, solve the Riemann problem defined by $\bar{u}$ and u^c, and insert fronts with positive speed into the coarse tube (fronts with negative speed are ignored). The average could be a spatial average or a flux average. Although both approaches would work for monotone flux functions, it is easy to construct examples for non-monotone flux

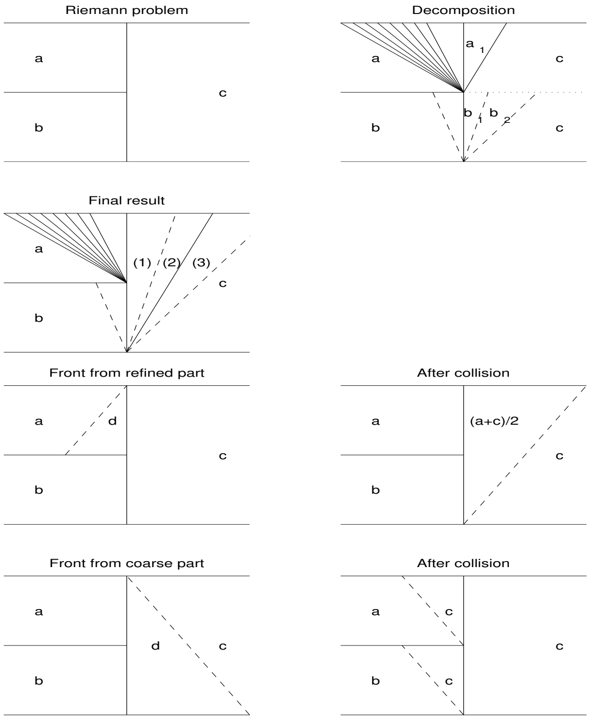

FIGURE 2. (Top) A Riemann problem at an interface between a coarse tube and two refined tubes is decomposed into two simple Riemann problems: solid lines represents (a, c) and dashed lines (b, c). Fronts propagating into coarse part are assigned new states by cross-sectional average; (1): $(a_1+b_1)/2$, (2): $(a_1+b_2)/2$, and (3): $(b_2+c)/2$. (Bottom) The static front is transparent during tracking: Colliding fronts from the refined part are assigned new states. Colliding fronts from the coarse part are copied.

functions where conservation of mass is violated or where the state corresponding to the average flux value is not unique.

For general flux functions, we therefore propose an approximation that will retain the structure from each individual Riemann problem (u^i, u^c) in a spatially averaged sense inside the coarse tube. All fronts going into the coarse tube (possibly from different Riemann problems) are assigned new states according to the spatial average across the coarse tube, collected in a list of increasing wave speeds, and inserted into the local X-list for the coarse tube. Afterwards, these fronts will generally not satisfy the Rankine–Hugoniot condition, but their states will form a monotone sequence, and the mass is conserved, see Figure 2. However, note that this will lead to discontinuities across the interface.

The next step is to resolve collisions involving static fronts. This is done by making the interface transparent. Fronts coming in from a coarse part are copied and inserted into each connected refined X-list. Fronts coming in from refined tubes are collected, assigned new states according to the spatial average, and inserted into the connected coarse list, see Figure 2. In the special case of at least two fronts from the same refined tube colliding exactly at the interface, the resulting simple Riemann problem must be solved first. Then the fronts moving back into the refined part are inserted into the corresponding X-list, and the fronts moving out of the refined part are collected, assigned new states, and inserted into the coarse X-list.

3.3. **Adaptive Grid Refinement.** By the nature of the splitting, possible refinements should be determined in the projection step. Before the projection, the solution in the tube is known as a sequence of fronts with arbitrary spatial distribution. The total variation inside each grid block could therefore be used to determine of the block should be refined.

Thus the strategy is as follows: When the total variation in a coarse grid cell exceeds an upper limit, the grid cell is refined. Similarly, when the total variation in a refined block is less than a lower limit, the solution is projected onto a coarse block and the refinement removed. Thus, the grid cells will be coarse in regions with small variation and refined in regions with large variation. By tuning the two threshold parameters, the fraction of refined grid blocks varies. Total variation is quickly computed as part of the projection, and the refinement strategy can easily be extended to systems.

This strategy improves the projection operator and thus provides better initial data for the next step in the other direction. The error will be reduced, but there is still room for improvement. It is very likely that some of the structure in the refined part of a tube will move into the coarse part during the next time step. Although Riemann problems are approximated quite well at the interface, the computation is expensive and some of the structure will be lost. It would therefore be desirable if one was able to predict where the refined structure will move in the next step and place refinements accordingly in advance. We only need the fastest wave in each direction and can therefore use an approximate Riemann solver that is certain to overestimate the wave speeds. For a monotone, convex flux function this could be the Rankine–Hugoniot velocity multiplied by a factor greater than one. This part obviously requires problem dependent code, and we will not go into further details.

In a practical implementation, a smoothing step should be included, ruling out small sections of unrefined blocks scattered in refined regions, since solving Riemann problems at an interface between coarse and fine grid blocks is expensive computationally.

4. Numerical Experiments

The dimensional splitting methods, from now referred to as DimSplit and GridRef, were coded in C and compiled using `gcc` with optimization `-O3`. For simplicity we have used a uniform Cartesian grid for both methods, and one level of refinements with a uniform refinement factor in each direction for GridRef. All experiments were conducted on a Sun Sparc 10 with 4 CPUs running Solaris. Unless stated otherwise, the spatial discretization parameters are equal in both directions, we use a Riemann solver designed for general non-convex flux functions, Strang splitting, and a distance 0.01 between interpolation points in the flux function. All reported runtimes are in CPU seconds.

4.1. **Burgers' Equation.** This example is taken from Teng [26] and is a two-dimensional Riemann problem for which the analytical solution is easily determined. Consider Burgers' equation

$$u_t + \big(\frac{1}{2}u^2\big)_x + \big(\frac{1}{2}u^2\big)_y = 0$$

CFL	rate	st.dev
1	0.83	0.004
4	0.82	0.003
16	0.81	0.003
*	0.71	0.06

TABLE 1. Numerical convergence rates for DIMSPLIT at different CFL numbers. The star (*) denotes that the time step is given by $\Delta t = 0.95\sqrt{\Delta x}$. The rates were determined by linear regression from runs with $\Delta x = \frac{a}{50}$, $a = 1, \ldots, 2^{-5}$.

with initial data

$$u_0(x,y) = \begin{cases} 1, & xy > 0 \\ 0, & xy < 0. \end{cases}$$

Applying a 45° rotation of the coordinates, this becomes a family of one-dimensional scalar problems. The analytical solution is

$$u(x,y,t) = \begin{cases} 0, & x/t < 0,\ y/t \geq \frac{1}{2}, \text{or } x/t \geq \frac{1}{2},\ y/t < 0 \\ x/t, & 0 \leq x/t < 1,\ y/t > ((x/t)^2+1)/2 \\ y/t, & 0 \leq y/t < 1,\ x/t > ((y/t)^2+1)/2 \\ 1, & \text{otherwise}, \end{cases}$$

and consists of both rarefaction and shock waves. In all computations we will use final time $t = 1.0$ and domain $[-0.2, 1.0] \times [-0.2, 1.0]$.

Convergence Rates. It is customary to relate the time step Δt to the spatial discretization Δx through the CFL number, $\nu = (\Delta t/\Delta x) \sup_{u<\|u\|_\infty} |f'(u)|$. Note that $\nu = \frac{1}{2}$ corresponds to a first order Godunov method for the approximate equation with piecewise linear flux functions. We now assume the error to be of form $E \approx C\Delta x^r$. In Table 1 the numerical rates are shown for CFL numbers 1, 4, and 16. We have also included the result of a different choice, $\Delta t = 0.95\sqrt{\Delta x}$. We see that the rates for constant CFL numbers are closer to 1 than to $\frac{1}{2}$.

Choosing the Time Step. As pointed out in Remark 2, it is not obvious how to choose the time step. Although CFL number $\nu = 1$ gives the highest convergence rate, the actual numerical error is least for $\nu = 8$. Choosing $\Delta t = 0.95\sqrt{\Delta x}$ on the other hand, gave favourable time steps for larger values of Δx, but too large steps for small Δx.

We determined the number of time steps giving least numerical error for $\Delta x = 1/50n$, $n = \frac{1}{2}, 1, 2, 3, \ldots, 16$. We assumed the time step to be given as $\Delta t = C\Delta x^\beta$. Fitting this model by linear regression for gave $C = 8.7$ and $\beta = 0.94$

When choosing the time step, efficiency should also be taken into account. We observed that the error $E(\Delta t)$ for a fixed Δx is slowly varying around the minimum point, and by choosing the largest time with an error not exceeding the minimum by more than 5%, the runtime can be greatly reduced (by at least 25%). Fitting the model once more gave $C = 11.6$ and $\beta = 0.93$, which we will denote as the "optimal" choice of time steps for this particular problem. Repeating the convergence tests above with these two choices of time steps, gave rates 0.81 and 0.79, respectively. In Figure 3 we have plotted the numerical error versus

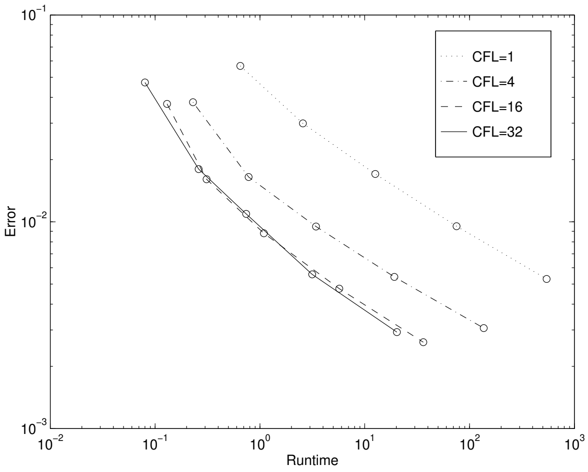

FIGURE 3. Numerical error versus runtime for DIMSPLIT at different CFL numbers for runs with $\Delta x = \frac{a}{25}$, $a = 1, \ldots, 2^{-4}$.

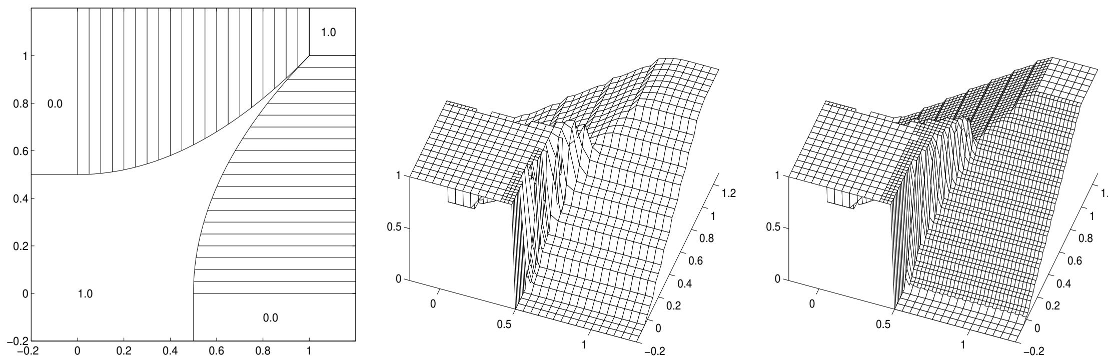

FIGURE 4. The analytical solution is given to the left. The other two plots show a typical grid after the last time step for two different sets of threshold values; (0.1, 0.5) in the middle and (0.005, 0.01) to the right. The grid consists of 30×30 coarse grid blocks on domain $[-0.2, 1.3] \times [-0.2, 1.3]$ and three time steps were used.

runtime for CFL numbers 1, 4, 16, and 32. The curve corresponding to the "optimal" time step, is approximately equal that of CFL numbers 16 and 32.

Grid Refinement. The total variation threshold parameters will obviously determine the fraction of grid blocks being refined and hence the accuracy of the grid refinement method. In Figure 4 we show the result of two computations with different threshold values together with the analytical solution. For threshold (0.1, 0.5) we observe that the grid is refined in regions along the shock paths. The numerical error is 0.037. Using a smaller threshold (0.005, 0.01) improves both the resolution of the converging shocks and the rarefaction areas. The numerical error is now reduced to 0.025. When the technique of predicting the behaviour of refined structures was included, we were able to reproduce the same error as for ordinary dimensional splitting with the same number of time steps and grid size equal that of the refined blocks.

CFL	rate	st.dev
1	0.81	0.005
4	0.70	0.004
16	0.85	0.001
*	0.85	0.001

CFL	rate	st.dev
1	0.80	0.004
4	0.77	0.003
16	0.87	0.020
*	0.85	0.003

TABLE 2. Numerical convergence rates for dimensional splitting with grid refinement. (Left) Refinement factor 2 and grid $\Delta x = \frac{a}{25}$, $a = 1, \dots, 2^{-4}$. (Right) Refinement factor 2 and grid $\Delta x = \frac{a}{25}$, $a = 2, \dots, 2^{-3}$. The star (*) denotes that the time step is given by $\Delta t = 11.6 \Delta x^{0.93}$.

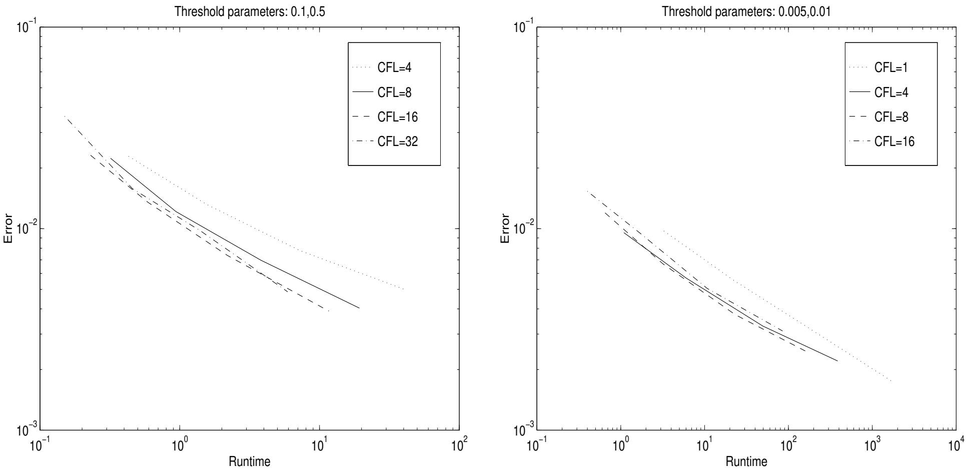

FIGURE 5. Numerical error versus runtime for two different sets of threshold parameters with four different CFL numbers. The error was measured for grid sizes $\Delta x = \frac{a}{25}$, $a = 1, \dots, 2^{-3}$ with a refinement factor 4.

We repeated the above convergence rate tests with local grid refinement for CFL numbers 1, 4, and 16, and the optimal time step determined above. In all runs we used threshold (0.005, 0.01). The numerical rates obtained for refinement factor 2 and factor 4 are shown in Table 2. We see that the observed convergence rates are well above $\frac{1}{2}$. Note however, that with fixed threshold parameters the fraction of refined grid blocks will decrease with decreasing Δx.

The next question is how to choose the time step. In Figure 5 we show the numerical error plotted versus runtime for a large and a small threshold. We see that for the smaller choice of parameters (0.005, 0.01) the CFL number should be chosen in the range 4–8 in order to maximize efficiency. In this case a large fraction of the grid blocks are refined, so the typical grid size is that of the refined blocks. Relating the CFL number to the typical grid size, means multiplying it by the refinement factor. For the larger choice of parameters (0.1, 0.5) the CFL number should be chosen in the range 16–32. In this case the fraction of refined grid blocks is small, and the typical grid size is that of the coarse blocks. Thus, we conclude that a large CFL number (10–20) related to the typical grid size should be used to achieve highest efficiency.

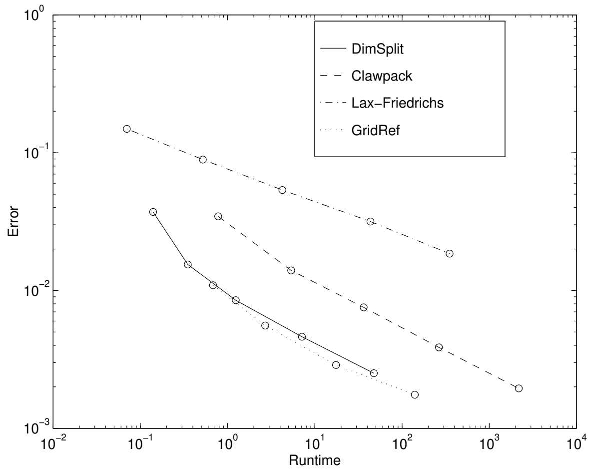

FIGURE 6. Comparison of the efficiency of Lax–Friedrichs, CLAWPACK, DIMSPLIT, and GRIDREF measured on grids $\Delta x = \frac{a}{25}$, $a = 1, \ldots, 2^{-4}$ for the first three methods and $a = 1, \ldots, 2^{-4}$ for GRIDREF. The threshold parameters for GRIDREF are $(0.01a; 0.1a)$

Comparison With Other Methods. To evaluate the efficiency of dimensional splitting, we made a comparison with two methods; a simple "low" resolution scheme, Lax–Friedrichs, and a modern high resolution scheme, the two-dimensional routine in the software package CLAWPACK Version 2.1 by LeVeque [18], which uses a multi-dimensional wave-propagation algorithm with flux delimiters. For those familiar with the package, we can describe the configuration as `method=(1,2,2)` and `mthlim=2`; that is, varying time steps with preferred CFL number 0.9, second order corrections with the superbee flux delimiter, and transverse propagation of both increment and correction waves. For DIMSPLIT we used CFL number 12.5, and for GRIDREF we used refinement factor 4, CFL number 8, and a TV threshold decreasing with the mesh parameter.

In Figure 6 runtime is plotted versus the observed error for the Lax–Friedrichs scheme, the CLAWPACK routine, DIMSPLIT, and GRIDREF. We observe that the dimensional splitting codes are superior compared to CLAWPACK for this example. For Burgers' equation, the Riemann solver is very simple, and by replacing our general solver by another tailored for this specific problem, even greater differences in performance were observed. Note also that GRIDREF is only slightly better than DIMSPLIT. This problem is quite demanding for the grid refinement, because of the large rarefaction areas contributing to the error in L^1-norm. In the last step the fraction of refined cells was approximately 50% in all runs.

4.2. **Non-Convex Flux Function.** For a more difficult example, we consider the following flux functions

$$
\begin{aligned}
f(u) &= 0.5u + 8u(u - 0.75)^2, \\
g(u) &= u(u + 1).
\end{aligned}
$$

Here f is non-convex. For non-convex flux functions, the entropy fix needed by the Roe solver in CLAWPACK is difficult to implement (see Harten and Hyman [7]). On the other hand, the non-convexity of the flux function will not cause any problems in a front tracking setting.

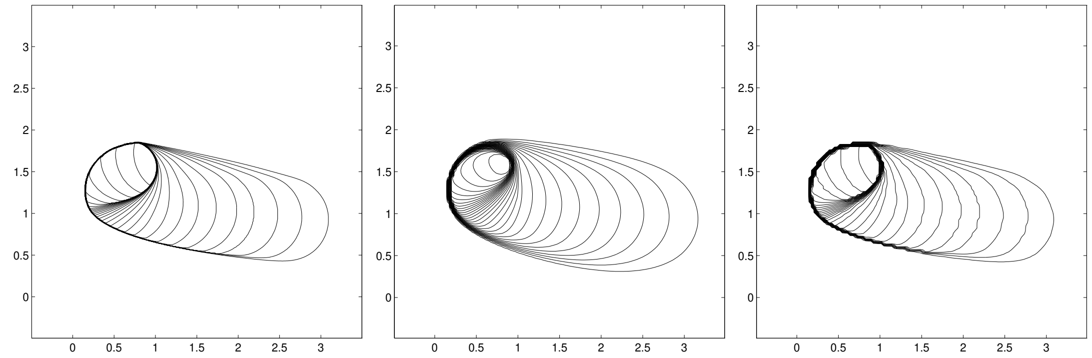

FIGURE 7. (Left) Approximate solution generated by DIMSPLIT with $\Delta x = 2^{-7}$ and CFL number 4. (Middle) DIMSPLIT with $\Delta x = 2^{-5}$ and CFL number $\frac{1}{2}$, i.e. a first order Godunov method. (Right) DIMSPLIT with $\Delta x = 2^{-5}$ and CFL number 8.

CFL	rate	st.dev
1	0.77	0.003
2	0.82	0.008
4	0.95	0.008
8	0.98	0.02
16	1.01	0.009

CFL	rate	st.dev
1	0.80	0.01
2	0.92	0.005
4	0.97	0.01
8	1.02	0.006
16	0.96	0.05

CFL	rate	st.dev
1	1.03	0.04
2	1.06	0.04
4	1.04	0.01
8	1.04	0.02
16	0.92	0.1

TABLE 3. Numerical convergence rates for different CFL numbers. (Left) DIMSPLIT with grid $\Delta x = 2^{-3}, \ldots, 2^{-7}$. (Middle) GRIDREF with refinement factor 2, and factor 4 (right) with grid $\Delta x = 2^{-2}, \ldots, 2^{-6}$ and threshold (0.005; 0.01)

We use initial data given by

$$u_0(x, y) = \begin{cases} 1, & \text{for } (x-0.4)^2 + (y-0.4)^2 < 0.25 \\ 0, & \text{otherwise.} \end{cases}$$

The analytical solution is unknown, and we used DIMSPLIT with CFL number 4 to generate a reference solution on a fine grid $\Delta x = 2^{-9}$. The solution is computed at time $t = 0.5$ on domain $[-0.5, 3.5] \times [-0.5, 3.5]$; see Figure 7.

Convergence Rates. In Table 3 we show the numerical convergence rates obtained with CFL numbers 1, 2, 4, 8, and 16 for DIMSPLIT and GRIDREF with factor 2 and 4. The rates are higher than for Burgers' equation, but are probably overestimated a bit, since we are now using a reference solution generated by the same numerical method. Nevertheless, we can conclude that the true rates are closer to 1 than to $\frac{1}{2}$. Also for this example we observe that the lowest numerical error for a fixed grid is obtained for CFL numbers well above 1, typically in the range 4–8.

Choosing the Time Step. Figure 7 shows that DIMSPLIT with CFL number 8 resolves the discontinuities noteably better than a first order Godunov method. The runtime is also much lower, 1.5 CPU seconds for DIMSPLIT and 16.1 for Godunov.

For the runs reported in Table 3, the numerical error was plotted versus the observed runtime (or versus the number of calls to the Riemann solver) to determine the CFL number

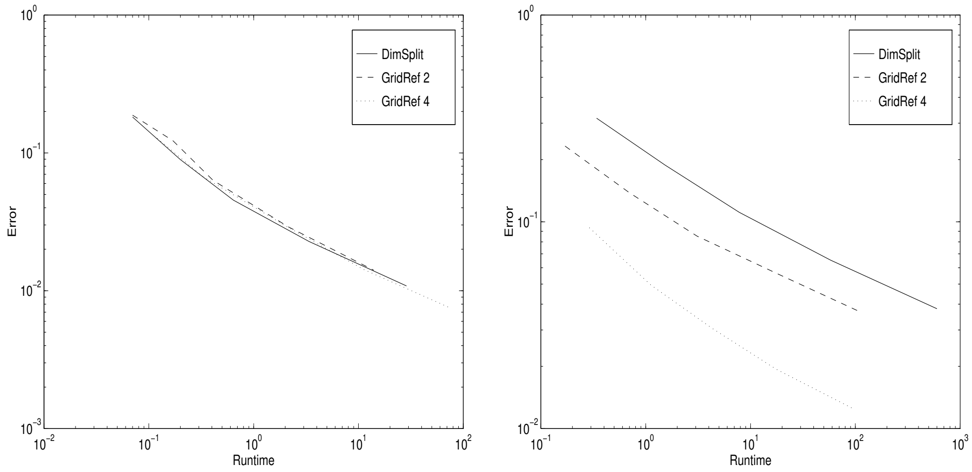

FIGURE 8. Comparison of the efficiency of DIMSPLIT and GRIDREF factor 2 and 4 at their best performance (left) and at CFL number 1 (right).

leading to the highest efficiency. For DIMSPLIT, GRIDREF factor 2, and GRIDREF factor 4 the CFL numbers were 16, 16, and 8, respectively. Figure 8 shows a plot of the efficiency of the three configurations at 'optimal' CFL number and at CFL number 1. We see that GRIDREF represents a great improvement in the efficiency of dimensional splitting at CFL number 1 (the same was observed for Burgers' equation and other examples not reported here). On the other hand, the CFL number can be increased more in DIMSPLIT than in GRIDREF before the loss in accuracy becomes significant, and at the optimal performance the efficiency of the three configurations are approximately equal (with a slight preference to DIMSPLIT for this particular problem).

4.3. **General Initial Data.** In the experiments so far we have considered initial data with a very simple structure. It is reasonable to see whether large CFL numbers are feasible for more general initial data. We therefore consider the following flux functions

$$f(u) = u(1 - u^2),$$
$$g(u) = \frac{1}{2}u^2,$$

and initial data

$$u_0(x, y) = \begin{cases} \sin(3\pi x)\sin(2\pi y), & \text{for } 0 \le x \le 1,\ 0 \le y \le 1 \\ 0, & \text{otherwise.} \end{cases}$$

The analytical solution is unknown, and we used DIMSPLIT with CFL number 2 to generate a reference solution on a fine grid $\Delta x = 2^{-10}$. The solution is computed at time $t = 0.5$ on domain $[-0.1, 1.775] \times [-0.1, 1.15]$; see Figure 9.

Again the numerical convergence rates were estimated for DIMSPLIT with CFL numbers 1, 2, 4, 8, and 16 and grid $\Delta x = 2^{-4}, \ldots, 2^{-8}$. The rates were 0.77, 0.97, 0.98, 1.01, and 1.04, respectively. For each grid, the lowest error was observed for CFL number 8, and the highest efficiency with respect to runtime (or the number of calls to the Riemann solver) was observed for CFL number 16. This confirms once more that large CFL numbers can be used without

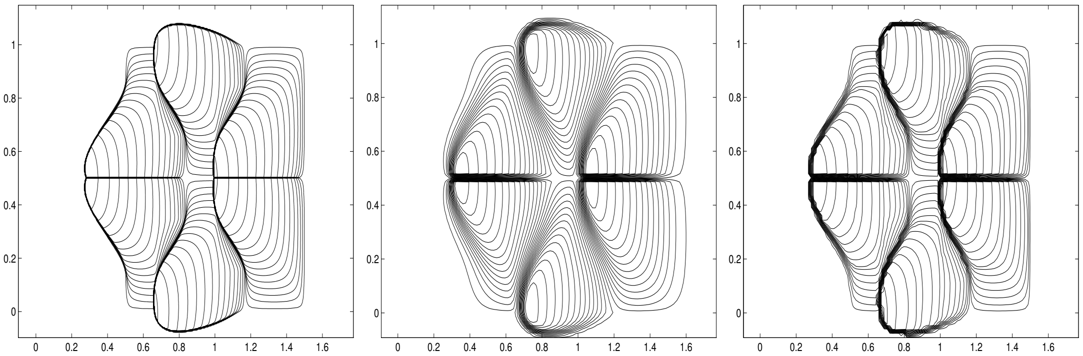

FIGURE 9. (Left) Approximate solution generated by DIMSPLIT with $\Delta x = 2^{-9}$ and CFL number 4. (Middle) DIMSPLIT with $\Delta x = 2^{-6}$ and CFL number $\frac{1}{2}$, i.e. a first order Godunov method. (Right) DIMSPLIT with $\Delta x = 2^{-6}$ and CFL number 8.

loss of accuracy. Figure 9 shows that also for this problem DIMSPLIT with CFL number 8 gives much better resolution than the first order Godunov method.

5. CONCLUDING REMARKS

The convergence rates confirm that the analytic error bound giving rate $\frac{1}{2}$ is a worst case estimate and that better performance with rates closer to 1 can be observed in actual computations.

Although the method is only first order in smooth regions, it resolves discontinuities very sharply and is thus best suited for problems where they are important. We observed that surprisingly large time steps can be used without loss in accuracy. We even observed that the numerical error was *decreased* by increasing the CFL number ν beyond 1 and up to 4–8. Compared to a first order Godunov method, which corresponds to ν less than $\frac{1}{2}$, there is a notable improvement in areas containing discontinuities. This indicates that the smearing error introduced by the projection operator is more dominant than the splitting error for low CFL numbers, and that these two balance at a CFL number about 4–8. By increasing ν further, the splitting error becomes dominant and the total error starts to increase. On the other hand, this leads to a decrease in the runtime, and a significant improvement in the efficiency can be observed by choosing ν in the range 10–20.

For the type of problems considered here, the dynamics can be resolved by a few time steps. Scalar problems with similar dynamics typically arise when simulating water injection in a oil reservoir. The front tracking and dimensional splitting method presented here constitutes an important part of a novel methodology for accurate computations of advection dominated displacement of oil by water [10, 11]. Similar observations are made concerning the size of the time steps for test cases closer to real-life applications.

Adaptive grid refinement means great improvement in the performance of dimensional splitting at low CFL numbers for the test cases reported here. However, at high CFL numbers the gain in accuracy is not enough to compensate for the computational overhead in our code, and the performance is equal to DIMSPLIT at its optimum. Still there are cases where GRIDREF performs better also at high CFL numbers, for instance the 2-D Riemann problem used in [6, 17], where the error in L^1-norm is dominated by the representation of shocks and not by large rarefaction areas as here. Since most effort has been spent on the front tracking part of GridRef we believe that its performance can be improved by optimizing the management of the adaptive grid. Also both DimSplit and GridRef can be further enhanced by introducing adaptive time steps.

The front tracking method can be generalized to systems [21], for instance the Euler equations [14, 15, 23] and the non-strictly hyperbolic polymer system [22]. Preliminary experiments for the 2-D Euler equations show that DimSplit performs well at moderate (but not at high) CFL numbers, and that GridRef has improved performance compared to DimSplit.

Only outflow boundary conditions have been applied here, but in a forthcoming paper both Dirichlet and "no flow" type boundary conditions will be considered.

Acknowledgements. We would like to thank Helge Holden and Nils Henrik Risebro for introducing us to this problem and for guidance along the way.

References

[1] F. Bratvedt, K. Bratvedt, C. F. Buchholz, T. Gimse, H. Holden, L. Holden, and N. H. Risebro. Frontline and Frontsim, two full scale, two-phase, black oil reservoir simulators based on front tracking. *Surv. Math. Indu.*, 3:185–215, 1993.

[2] B. Cockburn and P.-A. Gremaud. A priori error estimates for numerical methods for scalar conservation laws. Part I: The general approach. Supercomputer Institute Research Report UMSI 94/152, University of Minnesota, 1200 Washington Avenue South, Minneapolis, Minnesota 55415, Aug. 1994.

[3] M. G. Crandall and A. Majda. The method of fractional steps for conservation laws. *Numer. Math.*, 34:285–314, 1980.

[4] C. M. Dafermos. Polygonal approximation of solutions of the initial value problem for a conservation law. *J. Math. Anal. Appl.*, 38:33–41, 1972.

[5] S. K. Godunov. Finite difference methods for numerical computation of discontinous solutions of the equations of fluid dynamics. *Mat. Sbornik*, 47:271–295, 1959. (Russian).

[6] W. D. Gropp. Local uniform mesh refinement with moving grids. *SIAM J. Sci. Stat. Comput.*, 8(3):292–304, May 1987.

[7] A. Harten and J. M. Hyman. Self adjusting grid methods for one-dimensional hyperbolic conservation laws. *J. Comp. Phys.*, 50:235–269, 1983.

[8] H. Holden, L. Holden, and R. Høegh-Krohn. A numerical method for first order nonlinear scalar conservation laws in one-dimension. *Comput. Math. Applic.*, 15(6–8):595–602, 1988.

[9] H. Holden and N. H. Risebro. A method for fractional steps for scalar conservation laws without the CFL condition. *Math. Comput.*, 60(201):221–232, Jan. 1993.

[10] K. H. Karlsen, K. Brusdal, H. K. Dahle, S. Evje, and K.-A. Lie. The corrected operator splitting approach applied to a nonlinear advection–diffusion problem. Preprint 104, Department of Applied Mathematics. University of Bergen, N-5007 Bergen, Norway, Jan. 1997.

[11] K. H. Karlsen, K.-A. Lie, N. H. Risebro, and J. Frøyen. A front tracking approach to a two-phase fluid flow model with capillary forces. Preprint, Department of Applied Mathematics. University of Bergen, N-5007 Bergen, Norway, 1997.

[12] S. N. Kružkov. First order quasi-linear equations in several independent variables. *Math. USSR Sbornik*, 10(2):217–243, 1970.

[13] N. N. Kuznetsov. Accuracy of some approximative methods for computing the weak solutions of a first-order quasi-linear equation. *USSR Comput. Math. and Math. Phys. Dokl.*, 16(6):105–119, 1976.

[14] J. O. Langseth. On an implementation of a front tracking method for hyperbolic conservation laws. *Advances in Engineering Software*, 26(1):45–63, 1996.

[15] J. O. Langseth, N. H. Risebro, and A. Tveito. A conservative front tracking scheme for 1D hyperbolic conservation laws. In *Proc. 4th Internat. Conf. Hyperbolic Problems*, pages 385–392, Taormina-Italy, 1992.

[16] J. O. Langseth, A. Tveito, and R. Winther. On the convergence of operator splitting applied to conservation laws with source terms. Preprint Series 1, University of Oslo, July 1993.

[17] R. J. LeVeque. Large time step shock-capturing techniques for scalar conservation laws. *SIAM J. Numer. Anal.*, 19(6):1091–1109, Dec. 1982.

[18] R. J. Leveque. *CLAWPACK User Notes*. Applied Mathematics, Box 352420, University of Washington, Seattle, WA 98195–2420, Nov. 1995. Available at the URL `http://www.amath.washington.edu/~rjl/clawpack.html`.

[19] B. J. Lucier. Error bounds for the methods of Glimm, Godunov and LeVeque. *SIAM J. Numer. Anal.*, 22:1074–1081, 1985.

[20] B. J. Lucier. A moving mesh numerical method for hyperbolic conservation laws. *Math. Comput.*, 46(173):59–69, Jan. 1986.

[21] N. H. Risebro. A front–tracking alternative to the random choice method. *Proc. Amer. Math. Soc*, 117(4):1125–1129, Apr. 1993.

[22] N. H. Risebro and A. Tveito. Front tracking applied to a nonstrictly hyperbolic system of conservation laws. *SIAM J. Sci. Stat. Comput.*, 12(6):1401–1419, Nov. 1991.

[23] N. H. Risebro and A. Tveito. A front tracking method for conservation laws in one dimension. *J. Comp. Phys.*, 101(1):130–139, July 1992.

[24] P. L. Roe. Discontinuous solutions to hyperbolic systems under operator splitting. *Numer. Meth. for PDEs*, 7:277–297, 1991.

[25] G. Strang. On the construction and comparison of difference schemes. *SIAM J. Numer. Anal.*, 5(3):506–517, Sept. 1968.

[26] Z.-H. Teng. On the accuracy of fractional step methods for conservation laws in two dimensions. *SIAM J. Numer. Anal.*, 31(1):43–63, Feb. 1994.

(Knut–Andreas Lie)
Department of Mathematical Sciences,
Norwegian University of Science and Technology,
N-7034 Trondheim, Norway
E-mail address: `andreas@math.ntnu.no`

(Vidar Haugse)
Department of Petroleum Engineering and Applied Geophysics,
Norwegian University of Science and Technology,
N-7034 Trondheim, Norway
E-mail address: `vidarh@ipt.unit.no`

(Kenneth Hvistendahl Karlsen)
Department of Mathematics,
University of Bergen,
Johs. Brunsgt. 12,
N-5008 Bergen, Norway
E-mail address: `kennethk@mi.uib.no`