JOURNAL OF COMPUTATIONAL PHYSICS **70**, 397–438 (1987)

A Lagrangian Fractional Step Method for the Incompressible Navier–Stokes Equations on a Periodic Domain

CHRISTOPH BÖRGERS

Lawrence Berkeley Laboratory and Department of Mathematics, University of California, Berkeley, California 94720

AND

CHARLES S. PESKIN

Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, New York 10012

Received January 29, 1986; revised August 21, 1986

In the Lagrangian fractional step method introduced in this paper, the fluid velocity and pressure are defined on a collection of N fluid markers. At each time step, these markers are used to generate a Voronoi diagram, and this diagram is used to construct finite-difference operators corresponding to the divergence, gradient, and Laplacian. The splitting of the Navier–Stokes equations leads to discrete Helmholtz and Poisson problems, which we solve using a two-grid method. The nonlinear convection terms are modeled simply by the displacement of the fluid markers. We have implemented this method on a periodic domain in the plane. We describe an efficient algorithm for the numerical construction of periodic Voronoi diagrams, and we report on numerical results which indicate that the fractional step method is convergent of first order. The overall work per time step is proportional to $N \log N$. © 1987 Academic Press, Inc.

1. INTRODUCTION

Most of the finite-difference methods presently used in computational fluid dynamics use the Eulerian rather than the Lagrangian representation. An advantage of Lagrangian methods for the Navier–Stokes equations is the natural and simple way in which the nonlinear convection part of the equations is treated, by moving the grid in the flow. However, this movement of the grid causes crucial problems, since the discretization of the differential operators as well as the numerical solution of the resulting algebraic equations are more difficult on irregular grids than on the regular grids used for Eulerian computations.

Recently, there has been progress in overcoming these problems. Finite-difference discretizations of differential operators on irregular grids can be computed with increased efficiency with the aid of new computational geometry algorithms. The solution of the resulting discrete problems has become easier due to new numerical techniques such as multigrid methods. We briefly discuss these two developments in turn.

In order to discretize a differential operator on a general set S of points in the plane, one often uses a mesh associated with S, for example, a set of polygons each containing exactly one of the points in S, or a triangulation whose vertices are the points in S. The Voronoi diagram associated with S can serve as a polygonal mesh or as a tool for the construction of a triangular mesh. The Voronoi polygon associated with $\mathbf{X} \in S$ is the set of points at least as close to $\mathbf{X}$ as to any other point in S; see Voronoi [25]. Recently many algorithms for the construction of Voronoi diagrams have been developed; see Shamos and Hoey [21] for a very fast sequential algorithm, and Aggarwal *et al.* [1] for a parallel algorithm. Voronoi diagrams have been used in Lagrangian fluid dynamics codes by several authors; see, e.g., Augenbaum [2, 3] and Trease [24].

In addition, efficient iterative reconnection algorithms for triangular meshes have been used in Lagrangian fluid dynamics; see, e.g., Crowley [13], Dukowicz [14], Fritts [15], and Fritts and Boris [17]. The triangulations thus generated are closely related to Voronoi diagrams and can in fact be used to construct Voronoi diagrams. The generalization of these algorithms to three dimensions is possible, but complicated; see Fritts [16].

We turn to the solution of equations on Lagrangian grids. Many numerical algorithms for incompressible fluid dynamics require the solution of Poisson equations for the pressure. An example is the projection method introduced by Chorin [11]. We shall consider a Lagrangian version of the projection method which requires the solution of a Poisson equation for the pressure and a Helmholtz equation for each velocity component.

Multigrid algorithms applicable to (finite element) discretizations of elliptic problems on irregular triangular meshes have been studied extensively; see, e.g., Bank [5]. The algorithm in [5] constructs a hierarchy of nested triangulations, starting with a given coarse triangulation. This refinement procedure generates a grid adapted to the problem and, at the same time, the grid hierarchy needed for the solution of the discrete problem on the finest grid.

A different approach is required if the finest grid is part of the input to the multigrid algorithm. This is the case in the present context, since the grid is determined by the flow. In such a case, the hierarchy of auxiliary grids cannot be obtained by *refinement*; instead, a *coarsening* procedure, or something similar, is needed. We propose a two-grid algorithm which uses auxiliary rectangular grids. To smooth the error on the Lagrangian grid, our method uses a modified Gauss–Seidel iteration. We have implemented this algorithm on a square, with periodicity conditions at the boundaries. For simplicity, we use only one rectangular grid, solving the problems on it by Fast Fourier Transform. Our algorithm performs well even on uniformly distributed random grids.

A different successful multigrid algorithm applicable on Lagrangian grids has recently been introduced by Löhner *et al.* [18]. These authors use a Jacobi iteration scheme to smooth the error and non-rectangular auxiliary grids. We have not carried out a detailed comparison between the two approaches.

Most of the recent work on Lagrangian methods concentrates on inviscid flow. In this paper, however, we shall study a method for the viscous, incompressible Navier–Stokes equations, using primitive variables. We now give an introduction to this method.

The method is based on a collection of N moving points (fluid markers) which serve as the grid. The fluid velocity and pressure are defined at these points. For each configuration of the points, i.e., at each time step, a Voronoi diagram is generated. This diagram serves two purposes: It identifies the region of space that we assign to each grid point, and it establishes a discrete topology on the grid, i.e., it determines which pairs of grid points are to be treated as neighbors.

The algorithm that we use for the construction of the Voronoi diagrams is particularly fast in the present context of a time-dependent calculation. The reason for this is that the algorithm exploits information that was saved from the previous time step. We describe the algorithm here for the case of a periodic domain in the plane, and we emphasize those features of the algorithm associated with the periodicity. The reader should note, however, that the algorithm described here is adaptable to many different geometries. It has been used, for example, on a square domain in the plane [20] and also on the surface of a sphere [4]. We see no fundamental obstacle to its use in three dimensions.

We use the Voronoi diagram in this work as an aid in the construction of finite-difference operators corresponding to the divergence, gradient, and Laplace operators, and we prove that these operators are weakly consistent of first order with their continuous counterparts.

The finite-difference operators constructed in this way are used to define an implicit fractional step method for the time-dependent Stokes equations, i.e., the Navier–Stokes equations without the nonlinear terms. This fractional step method involves the solution of a Helmholtz equation for a preliminary velocity field and then the projection of this preliminary velocity field onto the space of discretely divergence-free vector fields. The projection step involves the solution of Poisson's equation for the pressure. The fractional steps corresponding to the Stokes equations are Eulerian, i.e., they involve ordinary space derivatives rather than derivatives with respect to the initial particle positions. The time step is completed by the actual motion of each grid point at its own velocity for one time step, simulating the convective terms in the Navier–Stokes equations. This movement of the grid is the Lagrangian part of the method. The Lagrangian form of the Navier–Stokes equations does not appear explicitly in our work.

An alternative to the above procedure would be to solve the discrete Stokes equations directly, without using two fractional steps. This was indeed tried [20], but no efficient method was found for the numerical solution of the discrete Stokes equations, which form a symmetric but indefinite system. In the fractional step method introduced here, however, all systems are at least semidefinite.

In Section 6, we report on numerical experiments for several simple test problems. The results suggest that the method is of first order in the sense that the discretization error is roughly reduced by $\frac{1}{2}$ if the number N of grid points is multiplied by 4 and the time step is reduced by $\frac{1}{2}$. The work per time step is $O(N \log N)$. There is no stability condition on the size of the time step.

We regard the present work as an initial investigation of a natural idea rather than a paper introducing a competitive practical method. In fact, the method in its present form is rather inaccurate in comparison with other methods for the incompressible Navier–Stokes equations. See, e.g., references [10, 11], which contain results of Eulerian calculations for test problems similar to ours. Further work will be required to increase the accuracy of the method and develop it into a practical tool.

A preliminary version of the present paper has appeared in [7]; see also [6, 20].

2. Construction of Periodic Voronoi Diagrams

Let S be a set of points in R^2, where R denotes the real numbers. For $\mathbf{X} \in S$, we define

$$P(\mathbf{X}, S) := \{\mathbf{Y} \in R^2 \colon |\mathbf{Y}-\mathbf{X}| \leqslant |\mathbf{Y}-\tilde{\mathbf{X}}| \text{ for all } \tilde{\mathbf{X}} \in S \text{ with } \tilde{\mathbf{X}} \neq \mathbf{X}\}. \tag{2.1}$$

$P(\mathbf{X}, S)$ is an intersection of half planes, i.e., a convex polygon. It is called the Voronoi polygon [25] or Dirichlet region of $\mathbf{X}$ with respect to S. The collection of all $P(\mathbf{X}, S)$, $\mathbf{X} \in S$, is called the Voronoi diagram associated with S and is denoted by $V(S)$; see Fig. 1. An edge in $V(S)$ is generated by two points, i.e., it belongs to two polygons. A corner is generated by three or more points, i.e., it belongs to three or more polygons. We always assume that every corner is generated by exactly three points. The resolution of multiple corners into sets of simple corners linked by edges of length zero, as shown in Fig. 2, is always possible. It is, however, never unique. This fact is of some importance for our construction algorithm described below.

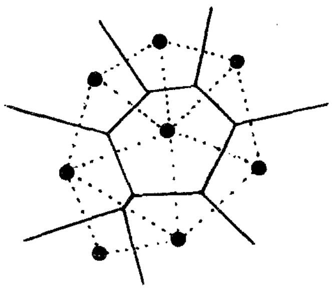


FIG. 1. Voronoi diagram (———) and Delaunay triangulation (···).

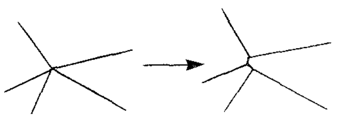


FIG. 2. Resolution of a multiple corner into a set of simple corners.

It is well known that the Voronoi diagram for a set of N points in the plane can be constructed in $O(N \log N)$ operations and that this operation count is best possible; see Shamos and Hoey [21]. However, if additional information on S is available, better operation counts can be achieved. The algorithm that we shall describe here makes use of the assumption that the Voronoi diagram has already been constructed for a set $\tilde{S}$ from which S can be obtained by slightly displacing each point, and that certain pieces of information about this construction have been saved. Similarly, the algorithm used by Fritts [15] can be viewed as an algorithm which uses $V(\tilde{S})$ to compute $V(S)$. Experiments indicate that both algorithms construct $V(S)$ in linear time. If S is a random set, there may be algorithms for which the expected value of the number of operations is smaller than $O(N \log N)$. This is particularly plausible if some information is given about the probability distribution of S. Numerical experiments supporting this statement will be presented later in this section.

We now consider the case

$$S = S_N := \{\mathbf{X}_j + \mathbf{k} : 1 \leqslant j \leqslant N, \mathbf{k} \in Z^2\}, \text{ with } \mathbf{X}_j \in [0, 1)^2, \tag{2.2}$$

where Z denotes the integers. In our fractional step method, the $\mathbf{X}_j$ will be fluid markers. The corresponding Voronoi diagram can be viewed as a periodic Voronoi diagram in the plane or as a Voronoi diagram on the torus

$$T := R^2/Z^2 \tag{2.3}$$

with the metric

$$d(\mathbf{X}, \mathbf{Y}) := \min\{|\mathbf{X} - \mathbf{Y} + \mathbf{k}| : \mathbf{k} \in Z^2\}. \tag{2.4}$$

Using our previous convention of resolving multiple corners into sets of simple corners, we have

PROPOSITION 2.1. *$V(S_N)$ has exactly $M = 2N$ corners in $[0; 1)^2$.*

Proof. We shall use Euler's formula,

$$v - e + f = 2,$$

for the numbers v, e, and f of vertices, edges and faces in any polygonal net covering the sphere. Consider m by m copies of the points $\mathbf{X}_1, \ldots, \mathbf{X}_N$,

$$\{\mathbf{X}_j + \mathbf{k} : \mathbf{k} = (k_1, k_2) \in Z^2, 1 \leqslant k_1, k_2 \leqslant m\}.$$

Consider the corresponding Voronoi diagram. Intersect it with $[0; m]^2$. The result is a polygonal mesh. We extend this mesh to one on the sphere $R^2 \cup \{\infty\}$ by introducing edges which connect the four corners of $[0; m]^2$ with ∞. This introduces a degenerate corner at ∞ which we resolve into two non-degenerate corners connected by an edge of length zero. The mesh on the sphere has m^2N+4 faces. Since every edge joins 2 corners and every corner belongs to 3 edges, the number of corners is $\frac{2}{3}$ times the number of edges. From Euler's formula, we now conclude that the total number of corners, including those at ∞, is $2m^2N+4$.

The $(m-2)\times(m-2)$ inner squares are identical up to translation and contain $(m-2)^2M$ corners. For fixed N, the number of corners in the $2m-4$ boundary squares is $O(m)$. Therefore we have

$$(m-2)^2M + O(m) = 2m^2N + 2.$$

Letting $m \to \infty$, we obtain $M = 2N$. ▌

In each period, the number of corners is $\frac{2}{3}$ times the number of edges. Hence there are exactly $3N$ edges (modulo Z^2) in $V(S)$. Note that the edges of length zero needed to resolve degenerate corners are counted here. The number of edges is of interest for our method because edges in $V(S)$ correspond to nonzero couplings in the finite-difference operators presented in Section 3.

Next we describe our storage scheme for the Voronoi diagram $V(S_N)$. We store information about the $2N$ corners which lie in $[0; 1)^2$. In the following, we shall call these corners the stored corners and call the points $\mathbf{X}_1, ..., \mathbf{X}_N$ the stored points. For the jth stored corner, we store its cartesian coordinates $(XC(j), YC(j))$ and the square RAD 2(j) of its radius, which is defined as its distance from its three generating points, i.e., the three points whose Voronoi polygons meet in the corner. We introduce the notation rad($\mathbf{c}$) for the radius of the corner $\mathbf{c}$. In addition, we store the three generating points of the jth stored corner and the three corners linked to it by edges in $V(S_N)$. This information is encoded in the following way.

The cartesian coordinates of a corner in $V(S)$ can be written in the form

$$XC(IC) + KX, \qquad YC(IC) + KY$$

with uniquely determined integer indices IC, KX, KY. We call IC the basic index and KX, KY the shift indices. We call the corner with basic index IC and shift indices KX, KY the corner (KX, KY, IC). We use analogous conventions for the points in S, i.e., the fluid markers and their translations. We store integer indices

$$IPT(i, n, j), \qquad i = 1, 2, 3, n = 0, 1, 2$$

and

$$ICR(i, n, j), \qquad i = 1, 2, 3, n = 0, 1, 2,$$

which have the following meaning. The point

$$(IPT(i, 1, j), IPT(i, 2, j), IPT(i, 0, j))$$

is the ith generating point of the jth stored corner, and the corner

$$(ICR(i, 1, j), ICR(i, 2, j), ICR(i, 0, j))$$

is the ith neighbor corner of the jth stored corner. This completes the description of our storage scheme.

We now outline the algorithm for the construction of $V(S_N)$.

ALGORITHM 2.1.

```
Step 1. Construct $V(S_1)$.
Step 2. For $j = 2,..., N$:
        Step 2.1. Construct $V(S_{j-1} \cup \{\mathbf{X}_j\})$ from $V(S_{j-1})$.
        Step 2.2. Construct $V(S_j)$ from $V(S_{j-1} \cup \{\mathbf{X}_j\})$.
```

Thus in step 2.1, we add the stored point $\mathbf{X}_j$ without its periodic images to the otherwise periodic Voronoi diagram. In step 2.2, we then add all the periodic images of $\mathbf{X}_j$.

Step 1 is trivial. The central part of the algorithm is a procedure for the construction of $V(S \cup \{\mathbf{Y}\})$ if $\mathbf{Y} \notin S$ and $V(S)$ is known, i.e., step 2.1. This procedure was described by Peskin [20] and Bowyer [8] independently. For completeness, we present it here.

A corner $\mathbf{c}$ in $V(S)$ is called broken by $\mathbf{Y}$ if it is not a corner in $V(S \cup \{\mathbf{Y}\})$. $\mathbf{Y}$ breaks $\mathbf{c}$ if and only if $\mathbf{Y}$ lies in the open disk around $\mathbf{c}$ with radius rad($\mathbf{c}$). For a general discrete set S of points in the plane, $\mathbf{Y}$ breaks at least one corner of $V(S)$ if it lies in the convex hull of S. To see this, note that the convex hull of S is exactly the union of the Delaunay triangles, i.e., the triangles which are obtained by joining all pairs of points in S with adjacent Voronoi polygons; see Fig. 1. $\mathbf{Y}$ lies in one of the Delaunay triangles. Consider the circumscribing circle of the Delaunay triangle. Its center is a corner $\mathbf{c}$ in $V(S)$. Since $\mathbf{Y}$ lies in the interior of this circle, it breaks $\mathbf{c}$. In our case, the convex hull of $S = S_{j-1}$ is R^2. Thus, Y is guaranteed to break a corner.

PROPOSITION 2.2. *The set of broken corners is connected in $V(S)$.*

Proof. This proof is due to D. Goldfarb (unpublished). The corners of a fixed polygon in $V(S)$ which are broken by $\mathbf{Y}$ clearly form a connected set. Walking along the boundary of $P(\mathbf{Y}, S \cup \{\mathbf{Y}\})$, one visits all polygons which have corners broken by $\mathbf{Y}$. Whenever $\partial P(\mathbf{Y}, S \cup \{\mathbf{Y}\})$ crosses an edge of $V(S)$, this edge has exactly one endpoint which is broken by $\mathbf{Y}$. This endpoint is a common corner of the two polygons bordered by the edge. Therefore, a walk through all broken corners is constructed by following $\partial P(\mathbf{Y}, S \cup \{\mathbf{Y}\})$. ▮

We can therefore find all broken corners in $O(1)$ operations, once we know one of them. We can find a first broken corner in the following way. We first choose some point $\mathbf{X} \in S$ close to $\mathbf{Y}$. Good ways of choosing $\mathbf{X}$ will be discussed below, since the efficiency of the algorithm depends mainly on this choice. We determine a corner $\mathbf{c}$ of $P(\mathbf{X}, S)$. For this purpose, we use an array *IPTCR* with the following meaning. The corner

$$(IPTCR(1, k), IPTCR(2, k), IPTCR(0, k))$$

is a corner of the kth polygon. Notice that the pointers *IPT* and *ICR* introduced above point from corners to points and corners and are therefore different from *IPTCR*. Beginning with $\mathbf{c}$, we then conduct a breadth first search through the graph of corners of $V(S)$ until a broken corner is found.

Once we know all corners broken by $\mathbf{Y}$, we can find all corners of the new polygon $P(\mathbf{Y}, S \cup \{\mathbf{Y}\}))$ by using the facts that each such corner lies on an edge between a broken and an unbroken corner in $V(S)$ and that on each such edge there is a new corner.

It remains to describe step 2.2. To simplify the notation, we take $j = N$ and define

$$\hat{P}_N := P(\mathbf{X}_N, S_{N-1} \cup \{\mathbf{X}_N\}) \tag{2.5}$$

and

$$P_N := P(\mathbf{X}_N, S_N). \tag{2.6}$$

It is clear how to get P_N from $\hat{P}_N$,

$$P_N = \hat{P}_N \cap ([X_{N,1} - 0.5; X_{N,1} + 0.5] \times [X_{N,2} - 0.5; X_{N,2} + 0.5]), \tag{2.7}$$

where $X_{N,1}$ and $X_{N,2}$ are the coordinates of $\mathbf{X}_N$.

We first assume that the decision which corners of $\hat{P}_N$ to cut off can simply be made by using (2.7), by checking the coordinates. Once this decision has been made, it is not difficult to find the radii of the new corners in $V(S_N)$, their neighbor corners and their neighbor points. Therefore the description of step 2.2 seems to be completed.

However, while the indicated procedure works correctly in most cases, it occasionally breaks down because of rounding errors. In Fig. 3, we show a typical situation which may lead to a breakdown. Here $N = 2$, $\mathbf{X}_1 = (0.0, 0.0)$, $\mathbf{X}_2 = (0.0, 0.5)$. P_N should have exactly 6 corners, two of which are identical up to translation with two others. In any finitely accurate arithmetic, the computed number of corners of P_N may be 4, 5, 6, 7, or 8.

The situations which cause difficulties are cases in which a polygon is adjacent to its own translation. This is most likely to happen at early stages of the construction.

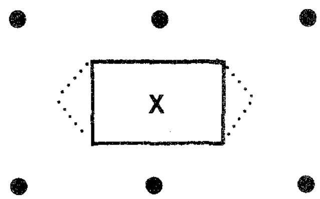


FIG. 3. A situation in which the straightforward use of (2.7) may lead to a breakdown. $\partial \hat{P}_N$ (···) and ∂P_N (———).

It may, however, even happen in $V(S_N)$ with arbitrarily large N, as when all points $\mathbf{X}_1, ..., \mathbf{X}_N$ lie on a line parallel to one of the coordinate axes. Even if a polygon is adjacent to its own translation, difficulties such as the one illustrated above occur only with a probability which tends to zero with the machine epsilon. It is desirable, though, to have an algorithm which will work, for any given set of points, if the machine epsilon is sufficiently small but positive.

We therefore have to cut $\hat{P}_N$ in a more careful way to obtain P_N. Let $\mathbf{c}$ be a corner of $\hat{P}_N$, generated by $\boldsymbol{\xi}$, $\boldsymbol{\eta}$, $\mathbf{X}_N$. Let $\tilde{\mathbf{c}}$ be a corner of $\hat{P}_N$, generated by $\boldsymbol{\xi} + (1, 0)$, $\boldsymbol{\eta} + (1, 0)$, $\mathbf{X}_N$. Fig. 4 shows two such situations. For simplicity, we use examples in which the points lie on a square grid. In practice, breakdowns do occur even in irregular configurations, even though with small probability.

Case (a) in Fig. 4 is the one discussed above as an example for a breakdown. A situation such as case (b) occurs, for example, when constructing the Voronoi diagram associated with the set

$$\{\mathbf{X}_1 = (0, 0), \mathbf{X}_2 = (0, \tfrac{1}{2}), \mathbf{X}_3 = (\tfrac{1}{2}, \tfrac{1}{2})\}.$$

Here the point being introduced is $\mathbf{X}_N = \mathbf{X}_3 = (\tfrac{1}{2}, \tfrac{1}{2})$.

To decide whether $\mathbf{c}$ is broken by $\mathbf{X}_N - (1, 0)$ in situation (a), we consider the Voronoi diagram $V(\{\boldsymbol{\xi}; \boldsymbol{\eta}; \mathbf{X}_N; \mathbf{X}_N - (1, 0)\})$; see Fig. 5. If we draw the diagram as in Fig. 5, we conclude that $\mathbf{c}$ is not broken by $\mathbf{X}_N - (1, 0)$, since it is a corner in the four-point Voronoi diagram. Clearly, we can use the same diagram to decide whether $\tilde{\mathbf{c}}$ is broken by $\mathbf{X}_N + (1, 0)$. We conclude that it is. If we had drawn the

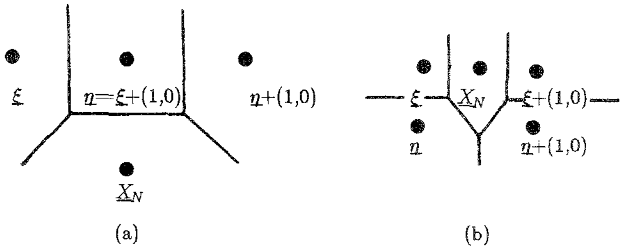


FIG. 4. Typical situations illustrating the use of (2.8).

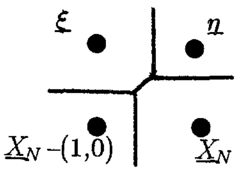


FIG. 5. The four-point Voronoi diagram needed for Fig. 4, situation (a).

short edge of length zero as in Fig. 6, we would have concluded that $\mathbf{c}$ is broken whereas $\tilde{\mathbf{c}}$ is not. In any case, we make opposite decisions for $\mathbf{c}$ and $\tilde{\mathbf{c}}$, which is correct here. We consider case (b) now. In this case, it is *not* correct to make opposite decisions for $\mathbf{c}$ and $\tilde{\mathbf{c}}$: Neither $\mathbf{c}$ nor $\tilde{\mathbf{c}}$ are broken here. Again, we can easily read this off from the Voronoi diagram $V(\{\boldsymbol{\xi}; \boldsymbol{\eta}; \mathbf{X}_N; \mathbf{X}_N-(1,0)\})$.

Our considerations motivate the following algorithm for cutting:

ALGORITHM 2.2. For each corner $\mathbf{c}$ of $\hat{P}_N$, decide whether or not it lies in $[X_{N,1}-0.5;\ X_{N,1}+0.5]\times[X_{N,2}-0.5;\ X_{N,2}+0.5]$, using the following procedure. Decide whether $\mathbf{c}$ is closer to $\mathbf{X}_N$ than to $\mathbf{X}_N+\mathbf{k}_0$, $\mathbf{k}_0=\pm(1,0)$ or $\mathbf{k}_0=\pm(0,1)$ by computing

$$V(\{\boldsymbol{\xi}; \boldsymbol{\eta}; \mathbf{X}_N; \mathbf{X}_N+\mathbf{k}_0\}), \tag{2.8}$$

where $\mathbf{X}_N$, $\boldsymbol{\xi}$, $\boldsymbol{\eta}$ are the generating points of $\mathbf{c}$. If $\mathbf{c}$ is a corner in this diagram, it is at least as close to $\mathbf{X}_N$ as to $\mathbf{X}_N+\mathbf{k}_0$, otherwise it is broken by $\mathbf{X}_N+\mathbf{k}_0$. If $\tilde{\mathbf{c}}$ is a corner of $\hat{P}_N$ generated by $\mathbf{X}_N$, $\boldsymbol{\xi}-\mathbf{k}_0$, $\boldsymbol{\eta}-\mathbf{k}_0$, use the *same* diagram to decide whether or not $\tilde{\mathbf{c}}$ is broken by $\mathbf{X}_N-\mathbf{k}_0$.

In our code, this procedure is simplified. The main simplification is a test whether cutting is necessary at all. This can safely be done by cheking the coordinates of the corners of $\hat{P}_N$. In the vast majority of cases, no cutting is necessary.

Computing the topological structure of the Voronoi diagram is an ill-conditioned problem in the sense that the solution, the topological structure, changes discontinuously with the data, the given points. Therefore the difficulty which we have described and overcome should not be called numerical instability. We rejected the first version of the algorithm not because it computes the wrong topological struc-

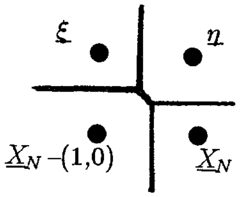


FIG. 6. Another way of drawing the diagram in Fig. 5.

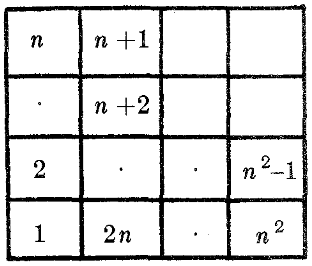


FIG. 7. Ordering of cells, determining the order in which the fluid markers are introduced at time zero.

ture, but because it computes two different structures at the same point on the torus. The improved version of the algorithm may still, due to rounding errors, compute the wrong topological structure; compare Figs. 5 and 6. This is unavoidable, but it fortunately does not matter in our application.

We remark that the *areas* of the Voronoi polygons depend on the given points in a continuous, in fact in a differentiable, way; see Section 3.

We finally describe two ways of finding a good starting point $\mathbf{X}$ for the search for the first broken corner. If the points in S are known to be roughly uniformly distributed in $[0; 1]^2$, we can divide $[0; 1]^2$ into n^2 square cells of equal size with $n^2 \approx N$ and create pointers from those cells to the points in S which they contain. A linked list is a good data structure for this purpose. In each cell, a pointer to one point in the cell is stored. Then each point in the cell points to another in the cell, with the last point containing a stop code. We introduce the points cell by cell in the order specified in Fig. 7. When introducing $\mathbf{X}_{k_0}$, a good choice of $\mathbf{X}$ is then $\mathbf{X}_{k_0-1}$. Using this method on random grids in $[0; 1]^2$, we obtain the CPU-times for algorithm 2.2 displayed in Table I. The measured CPU-times support our assertion that the CPU-time per point is bounded independently of N.

In our fractional step method, we use this method at time 0. At all later times, we use the following, slightly faster method. Assume that $V(\tilde{S})$ has been constructed for

TABLE I

Construction of Periodic Voronoi Diagrams

N	Worst case	Best case	Average
400	16	13	14
800	15	13	14
1200	15	14	14
1600	16	14	15
2000	16	15	15

Note. CPU-times per point in msec on a VAX 11/780, 20 uniformly distributed random grids for each N.

a set $\tilde{S}=\{\tilde{\mathbf{X}}_j+\mathbf{k}\colon j\in\{1,\dots,N\},\ \mathbf{k}\in Z^2\}$, where the distances $d(\mathbf{X}_j, \tilde{\mathbf{X}}_j)$ are small, and $\tilde{\mathbf{X}}_j\in[0;1)^2$. Here d is defined as in (2.4). Let $j(k_0)$ be the index of a point in $\tilde{S}$ whose polygon had a broken corner when $\tilde{\mathbf{X}}_{k_0}$ was inserted during the construction of $V(\tilde{S})$. A suitable translation of the point $\mathbf{X}_{j(k_0)}$ is then close to $\mathbf{X}_{k_0}$. The indices $j(k_0)\in\{1;\dots;k_0\}$ are therefore saved while constructing $V(\tilde{S})$. In our application, S is the set of fluid markers and their periodic images at a given time step, and $\tilde{S}$ is the corresponding set at the preceding time step.

We note that this method does not require any renumbering of the points.

We conclude this section with some remarks about other possible ways of updating periodic Voronoi diagrams in a time-dependent calculation.

It is, of course, possible to reduce the problem to nonperiodic Voronoi diagrams in the plane. Consider

$$\hat{S}:=\{\mathbf{X}_j+\mathbf{k}\colon 1\leqslant j\leqslant N,\ \mathbf{k}\in Z^2,\ |k_1|\leqslant 1 \text{ and } |k_2|\leqslant 1\}. \tag{2.9}$$

It is easy to see that

$$P(\mathbf{X}_j,\hat{S})=P(\mathbf{X}_j,S) \qquad \text{for all } j. \tag{2.10}$$

Since $\hat{S}$ contains $9N$ points, a procedure based on $\hat{S}$ would be quite inefficient. The definition of $\hat{S}$ can be modified to reduce the number of auxiliary points, but the algorithm then becomes complicated. It becomes impossible to search for the first broken corner using the second strategy, since the set of auxiliary points becomes dependent on the given configuration and therefore time-dependent.

It may also be possible to work with the torus (2.3) and the metric (2.4) directly. The point insertion algorithm, however, cannot be used in literally the same way as in the plane. To see this, consider the step in which the coordinates of a new corner are computed after its three generating points $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ have been found. In the plane, this is done by computing the midpoint of the circle through $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$. On the torus T with metric d, such a unique midpoint need not exist. A counterexample is given by $\mathbf{X}=(0,0)$, $\mathbf{Y}=(\frac{1}{2},0)$, $\mathbf{Z}=(0,\frac{1}{2})$. Here

$$d(\mathbf{X},\mathbf{A})=d(\mathbf{Y},\mathbf{A})=d(\mathbf{Z},\mathbf{A})=1/\sqrt{8} \qquad \text{for} \quad \mathbf{A}\in\{(\tfrac{1}{4},\tfrac{1}{4}),(\tfrac{1}{4},\tfrac{3}{4}),(\tfrac{3}{4},\tfrac{1}{4}),(\tfrac{3}{4},\tfrac{3}{4})\}.$$

A periodic version of the algorithm used by Fritts [15] appears to be an excellent alternative. This algorithm constructs the Delaunay triangulation rather than the Voronoi diagram and seems more efficient than our method. However, we believe that an approach like the one described here may be more generally applicable because it makes use of the metric only and does not require an inner product or angles. A generalization to three-dimensional periodic Voronoi diagrams seems straightforward. For a version on the surface of a sphere see Augenbaum and Peskin [4].

3. Finite-Difference Operators on Irregular Grids

In this section, we define discrete Laplace, divergence and gradient operators, and we discuss their properties. Even though we use a terminology appropriate for 2 dimensions, the definitions and statements in this section are dimension independent.

We introduce the following notations. For $\mathbf{X} \in S_N$, let $V[\mathbf{X}]$ be the area of the Voronoi polygon $P(\mathbf{X}, S_N) =: P[\mathbf{X}]$. For $\mathbf{Y} \in S_N$, $\mathbf{Y} \neq \mathbf{X}$, let $A[\mathbf{X}, \mathbf{Y}]$ be the length of the common edge of $P(\mathbf{X}, S_N)$ and $P(\mathbf{Y}, S_N)$. $A[\mathbf{X}, \mathbf{Y}] = 0$ if $P(\mathbf{X}, S_N)$ and $P(\mathbf{Y}, S_N)$ are not adjacent to each other. Our discrete operators couple two distinct points $\mathbf{X}, \mathbf{Y}$ to each other only if they are neighbors, i.e., if $A[\mathbf{X}, \mathbf{Y}] \neq 0$. In the following, we let $Nb[\mathbf{X}]$ be the set of neighbors of $\mathbf{X}$. By definition, $\mathbf{X} \notin Nb[\mathbf{X}]$.

3.1. *Discrete Laplace Operator*

We define a discrete Laplace operator L by

$$V[\mathbf{X}](L\phi)(\mathbf{X}) := \sum_{\mathbf{Y} \neq \mathbf{X}} A[\mathbf{X}, \mathbf{Y}] \frac{\phi(\mathbf{Y}) - \phi(\mathbf{X})}{|\mathbf{Y} - \mathbf{X}|} \qquad \text{for} \quad \mathbf{X} \in S_N. \tag{3.1}$$

This definition is morivated by the formula

$$\int_P \Delta\phi(\mathbf{x})\, d\mathbf{x} = \int_{\partial P} \frac{\partial \phi}{\partial \mathbf{n}}(\mathbf{x})\, ds \tag{3.2}$$

for smooth functions ϕ. The sum in (3.1) is formally infinite but contains only finitely many non-zero terms. Similar discretizations of the Laplace operator have been used in numerical computations for a long time; see, for example, MacNeal [19]. The following proposition was pointed out to us by B. Mercier (unpublished communication).

Proposition 3.1. *The operator in (3.1) seen as a matrix is the stiffness matrix* [illegible]

Proof. Both matrices have zero row sums. Therefore we need to consider the non-diagonal entries only. The matrices have the same non-zero structure and are both symmetric. It therefore makes sense to talk about the coupling between $\mathbf{X}$ and $\mathbf{Y}$ ($\mathbf{X}, \mathbf{Y} \in S_N$, $\mathbf{X} \neq \mathbf{Y}$). We compare the couplings in the matrix in (3.1) with those in the finite element stiffness matrix.

Without loss of generality, we assume that $\mathbf{X} = (0, 0)$ and $\mathbf{Y} = (0, 1)$. Let $\mathbf{A}, \mathbf{B}$ be the points in S_N such that $\mathbf{A}, \mathbf{X}, \mathbf{Y}$ and $\mathbf{B}, \mathbf{X}, \mathbf{Y}$ are triangles in the Delaunay triangulation. Let $\mathbf{C}$ be the center of the circle through $\mathbf{A}, \mathbf{X}, \mathbf{Y}$, and let $\mathbf{D}$ be the center of the circle through $\mathbf{B}, \mathbf{X}, \mathbf{Y}$. $\mathbf{C}$ and $\mathbf{D}$ are corners in $V(S_N)$. Our claim is reduced to the statement that the coupling between $\mathbf{X}$ and $\mathbf{Y}$ obtained with piecewise linear finite elements is $|\mathbf{C} - \mathbf{D}|$.

To show this, we first express this coupling in terms of the coordinates of **A** and **B**,

$$-\frac{A_2^2-A_2}{2A_1}-\frac{A_1}{2}+\frac{B_2^2-B_2}{2B_1}+\frac{B_1}{2}. \tag{3.3}$$

(Without loss of generality, we have assumed that $A_1<0$ and $B_1>0$.) Next we compute $|\mathbf{C}-\mathbf{D}|$,

$$\mathbf{C}=\left(\frac{1}{2}A_1+\frac{A_2(A_2-1)}{2A_1}, 0.5\right) \tag{3.4}$$

$$\mathbf{D}=\left(\frac{1}{2}B_1+\frac{B_2(B_2-1)}{2B_1}, 0.5\right) \tag{3.5}$$

Therefore the square of (3.3) equals the square of $|\mathbf{C}-\mathbf{D}|$.

To conclude the proof of our assertion, it remains to be shown that (3.3) is positive. We note that the foregoing computations do not make any use of the fact that we are using Delaunay triangulations and Voronoi diagrams. However, all off-diagonal elements of the finite element Laplacian obtained with a triangulation τ are positive if and only if τ is a Delaunay triangulation.

We outline the proof of this well-known result.

A triangulation τ is called locally equiangular if it has the following property. If T_1 and T_2 are adjacent triangles in τ whose union Q is a convex quadrilateral, the sum of the angles in Q which are cut by the common edge of T_1 and T_2 is larger than 180°.

It is quite obvious that Delaunay triangulations are locally equiangular. Sibson [22] showed that the converse is also true.

A simple calculation shows that all off-diagonal elements of the finite element Laplacian obtained with a triangulation τ are positive if and only if τ is locally equiangular; see Fritts [15]. This concludes the proof of Proposition 3.1. ∎

In spite of Proposition 3.1, a discretization of the Poisson equation based on (3.1) is not a common finite element discretization. The difference lies in the right-hand sides. Let f be a given continuous right-hand side. In the finite element method, the discrete right-hand side consists of integrals of the form $\int \tilde{f}(\mathbf{x})\,\beta(\mathbf{x})\,d\mathbf{x}$, where $\tilde{f}$ is an approximation of f and β is a basis function of the finite element space. When using piecewise linear elements, the common choice of $\tilde{f}$ is the piecewise linear interpolant of f. The value in $\mathbf{X}$ of the discrete right-hand side is then a weighted sum of the values of f in the neighbors of $\mathbf{X}$. The sum $M[\mathbf{X}]$ of the weights (elements of the mass matrix) is one third the sum of the areas of the triangles to which $\mathbf{X}$ belongs. Using (3.1), one is lead to using $V[\mathbf{X}]f(\mathbf{X})$ as the value of the discrete right-hand side in $\mathbf{X}$. $V[\mathbf{X}]$ and $M[\mathbf{X}]$ differ from each other in general, but they are, of course, equal on the average. $M[\mathbf{X}]$ is the area of a cell suggested by Dukowicz [14] and others, which is obtained by joining the centroids of the triangles with the midpoints of their sides.

We now prove that the operator L is weakly consistent to first order with the continuous Laplacian Δ. We define

$$h := \max_{\mathbf{X}} \operatorname{diam}(P[\mathbf{X}]). \tag{3.6}$$

PROPOSITION 3.2. *If*

$$N \leqslant O\left(\frac{1}{h^2}\right), \tag{3.7}$$

then

$$\sum_k \psi(\mathbf{X}_k)\, L\phi(\mathbf{X}_k)\, V[\mathbf{X}_k] = \int_{[0;1]^2} \psi(\mathbf{x})\, \Delta\phi(\mathbf{x})\, d\mathbf{x} + O(h) \tag{3.8}$$

for arbitrary smooth, periodic functions ϕ and ψ.

Proof. We approximate the difference to be estimated by

$$\sum_k \psi(\mathbf{X}_k)(L\phi)(\mathbf{X}_k)\, V[\mathbf{X}_k] - \sum_k \psi(\mathbf{X}_k) \int_{P[\mathbf{X}_k]} \Delta\phi(\mathbf{x})\, d\mathbf{x}. \tag{3.9}$$

Since ψ is uniformly continuous, the error which we have introduced is only $O(h)$. We next apply Green's formula (3.2) and its discrete counterpart (3.1) and obtain

$$\sum_k \sum_{\mathbf{Y} \neq \mathbf{X}_k} \psi(\mathbf{X}_k) \left[A[\mathbf{X}_k, \mathbf{Y}] \frac{\phi(\mathbf{Y}) - \phi(\mathbf{X}_k)}{|\mathbf{Y} - \mathbf{X}_k|} - \int_{P[\mathbf{X}_k] \cap P[\mathbf{Y}]} \frac{\partial\phi}{\partial\mathbf{n}}(\mathbf{x})\, ds \right]. \tag{3.10}$$

Since the expression in the bracket is of order $O(h^2)$, and because of (3.7), we seem to obtain an upper bound of size $O(1)$ now. However, observe that the roles of $\mathbf{X}_k$ and $\mathbf{Y}$ in (3.10) may be reversed, therefore (3.10) is equal to

$$-\sum_k \sum_{\mathbf{Y} \neq \mathbf{X}_k} \psi(\mathbf{Y}) \left[A[\mathbf{X}_k, \mathbf{Y}] \frac{\phi(\mathbf{Y}) - \phi(\mathbf{X}_k)}{|\mathbf{Y} - \mathbf{X}_k|} - \int_{P[\mathbf{X}_k] \cap P[\mathbf{Y}]} \frac{\partial\phi}{\partial\mathbf{n}}(\mathbf{x})\, ds \right]. \tag{3.11}$$

Here $\mathbf{n}$ always denotes the normal on $P[\mathbf{X}_k] \cap P[\mathbf{Y}]$ which is exterior with respect to $P[\mathbf{X}_k]$. Since (3.10) and (3.11) are equal, they are also both equal to the average of the two expressions,

$$\frac{1}{2} \sum_k \sum_{\mathbf{Y} \neq \mathbf{X}_k} (\psi(\mathbf{X}_k) - \psi(\mathbf{Y})) \left[A[\mathbf{X}_k, \mathbf{Y}] \frac{\phi(\mathbf{Y}) - \phi(\mathbf{X}_k)}{|\mathbf{Y} - \mathbf{X}_k|} - \int_{P[\mathbf{X}_k] \cap P[\mathbf{Y}]} \frac{\partial\phi}{\partial\mathbf{n}}(\mathbf{x})\, ds \right], \tag{3.12}$$

which is $O(h)$. ▮

Assumption (3.7) is appropriate for two space dimensions. In d dimensions, it is replaced by $N \leqslant O(1/h^d)$. (3.7) essentially states that the fluid markers should be spread out uniformly. Consider, for example, a case in which all $\mathbf{X}_k$ lie on a vertical or horizontal straight line. Then $h = O(1)$, independently of N.

We give a second, shorter proof of (3.8). Because of Proposition 3.1, (3.8) is equivalent to

$$a(\phi_h, \psi_h) = a(\phi, \psi) + O(h), \tag{3.13}$$

where $a(\cdot, \cdot)$ is the bilinear form

$$a(\phi, \psi) := \int_{[0;1]^2} \nabla\phi(\mathbf{x}) \cdot \nabla\psi(\mathbf{x})\, d\mathbf{x} \tag{3.14}$$

associated with the Laplace operator, and ϕ_h, ψ_h are the piecewise linear interpolants of ϕ, ψ with respect to the Delaunay triangles. Equation (3.13) is well known from the theory of interpolation in Sobolev spaces needed in the finite element convergence theory; see Ciarlet [12]. The non-degeneracy assumption on the triangulation which is needed for this argument is, however, neither necessary nor sufficient for (3.7) to hold.

We gave the first argument for two reasons. First, it is independent of the connection with the finite element method and therefore applicable even if the control areas are not Voronoi polygons but areas defined in some different way. Second, it is applicable to discretizations of differential operators other than the Laplace operator; see below for the case of the divergence operator.

PROPOSITION 3.3. *L is* not *pointwise consistent with the Laplace operator.*

Proof. Counterexamples can be constructed easily; see [6]. ▌

L is, however, pointwise consistent of order 0 in the following sense.

PROPOSITION 3.4. *For a fixed smooth periodic function ϕ, $L\phi$ is bounded independently of the fluid marker configuration.*

Proof.

$$\begin{aligned}
|L\phi(\mathbf{X})| &= \left| \frac{1}{V[\mathbf{X}]} \sum_{\mathbf{Y} \neq \mathbf{X}} \frac{\phi(\mathbf{Y}) - \phi(\mathbf{X})}{|\mathbf{Y}-\mathbf{X}|} A[\mathbf{X}, \mathbf{Y}] \right| \\
&= \left| \frac{1}{V[\mathbf{X}]} \sum_{\mathbf{Y} \neq \mathbf{X}} \frac{\phi(\mathbf{Y}) - \phi(\mathbf{X})}{|\mathbf{Y}-\mathbf{X}|} A[\mathbf{X}, \mathbf{Y}] - \frac{1}{V[\mathbf{X}]} \sum_{\mathbf{Y} \neq \mathbf{X}} \nabla\phi(\mathbf{X}) \cdot \frac{\mathbf{Y}-\mathbf{X}}{|\mathbf{Y}-\mathbf{X}|} A[\mathbf{X}, \mathbf{Y}] \right| \\
&\leqslant \frac{C}{V[\mathbf{X}]} \sum_{\mathbf{Y} \neq \mathbf{X}} |\mathbf{Y}-\mathbf{X}|\, A[\mathbf{X}, \mathbf{Y}]
\end{aligned}$$

for some constant C depending on ϕ only. It can easily be shown that

$$\sum_{\mathbf{Y} \neq \mathbf{X}} A[\mathbf{X}, \mathbf{Y}]\, |\mathbf{Y}-\mathbf{X}| = 4V[\mathbf{X}].$$

This concludes the proof of Proposition 3.4. ▌

We define the discrete L^2-product by

$$(\phi, \psi) := \sum_k \phi(\mathbf{X}_k)\, \psi(\mathbf{X}_k)\, V[\mathbf{X}_k] \tag{3.15}$$

for scalar functions and

$$(\mathbf{u}, \mathbf{v}) := \sum_k \mathbf{u}(\mathbf{X}_k) \cdot \mathbf{v}(\mathbf{X}_k)\, V[\mathbf{X}_k]. \tag{3.16}$$

for vector fields.

PROPOSITION 3.5. *L is symmetric and negative semidefinite in the inner product (3.15). The kernel of L is the set of constants.*

Proof.

$$(L\phi, \psi) = \sum_k \sum_{\mathbf{X} \neq \mathbf{X}_k} A[\mathbf{X}_k, \mathbf{Y}]\, \frac{\phi(\mathbf{Y}) - \phi(\mathbf{X}_k)}{|\mathbf{Y} - \mathbf{X}_k|}\, \psi(\mathbf{X}_k). \tag{3.17}$$

Reversing the roles of $\mathbf{X}_k$ and $\mathbf{Y}$ and averaging the resulting expression with (3.17), we obtain

$$(L\phi, \psi) = \frac{1}{2} \sum_k \sum_{\mathbf{Y} \neq \mathbf{X}_k} A[\mathbf{X}_k, \mathbf{Y}]\, \frac{\phi(\mathbf{Y}) - \phi(\mathbf{X}_k)}{|\mathbf{Y} - \mathbf{X}_k|}\, (\psi(\mathbf{Y}) - \psi(\mathbf{X}_k)), \tag{3.18}$$

from which the assertions follow. ∎

We conclude this subsection with a brief discussion of the changes required if Dirichlet or Neumann boundary conditions rather than periodicity conditions are imposed on ϕ. One can still base the definition of $L\phi$ on (3.2), taking the boundary conditions into account in a straightforward way. The case of Neumann boundary conditions is particularly simple, since (3.2) involves $\partial\phi/\partial\mathbf{n}$. In the case of Dirichlet boundary conditions, the simplest definition involves discretizing $\partial\phi/\partial\mathbf{n}$ on $\partial(0;1)^2$ by one-sided difference quotients, using auxiliary grid points on $\partial(0;1)^2$.

Without loss of generality, we assume now that the Dirichlet or Neumann conditions imposed on ϕ are homogeneous. As in the periodic case, L is then a linear operator acting on functions defined on $\{\mathbf{X}_1, ..., \mathbf{X}_N\}$. Proposition 3.1 cannot be formulated as in the periodic case, since the Delaunay triangles do not completely cover the unit square. Propositions 3.2 to 3.5 remain virtually unchanged. In Eq. (3.8), we now require ϕ, ψ to be smooth on $[0;1]^2$. In addition, ϕ must satisfy [illegible] case, L is then negative definite.

3.2. *Discrete Divergence Operator*

We turn to the discrete incompressibility condition. In terms of the partition of the plane into Voronoi polygons, the constraint of incompressibility can be formulated as follows: The motions of the grid points should be such that the area of each polygon is conserved. At a given configuration of the generating points, this leads to a linear system of constraints on the velocities which is a natural discretization of $\nabla \cdot \mathbf{u} = 0$. Thus we define

$$V[\mathbf{X}]\, D\mathbf{u}(\mathbf{X}) := \sum_{\mathbf{Y}} \mathbf{u}(\mathbf{Y}) \cdot \frac{\partial V[\mathbf{X}]}{\partial \mathbf{Y}} \quad \text{for} \quad \mathbf{X} \in S_N. \tag{3.19}$$

Here $\partial V[\mathbf{X}]/\partial \mathbf{Y}$is the gradient of $V[\mathbf{X}]$ with respect to $\mathbf{Y}$, keeping all other points fixed; see Proposition 3.7.

If $(\mathbf{X}_1,..., \mathbf{X}_N) = (\mathbf{X}_1(t),..., \mathbf{X}_N(t))$ are points moving at velocity $\mathbf{u}$, then

$$\frac{d}{dt} V[\mathbf{X}_j(t)] = V[\mathbf{X}_j(t)]\, D\mathbf{u}(\mathbf{X}_j(t)) \tag{3.20}$$

for all j, in particular:

PROPOSITION 3.6. *If $\mathbf{X}_1(t),..., \mathbf{X}_N(t)$ are points in space which move in a discretely divergence-free velocity field, the Voronoi polygons maintain their areas exactly.*

However, the Voronoi polygons do change their areas, usually even quite drastically, if the points are moved in a *continuously* divergence-free field. Numerical experiments show that this is even true for large numbers of fluid markers and small diameters of the Voronoi polygons. We give an example which confirms and generalizes this observation. Consider the divergence-free flow

$$\mathbf{u} = A\mathbf{x} \tag{3.21}$$

with

$$A = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}.$$

Let a set of fluid markers move in this flow and consider the corresponding Voronoi diagrams. If one of the markers is at the origin $\mathbf{x} = \mathbf{0}$, it does not move. Its Voronoi polygon at later times depends on markers which are far away from it at time 0 and will therefore usually not maintain its area. If the coordinates of all fluid markers at time 0 are multiplied by ε, then the resulting Voronoi diagrams are just scaled by the same factor ε.

In summary, this example shows that Voronoi polygons of a set of points moving in a continuously divergence-free flow may change their areas over a fixed time by a factor which does not converge to 1 when the set is made finer in the sense that h, as in (3.6), tends to zero. Furthermore, the example shows that the same statement is true for all control areas which are just scaled by ε when the given set of points is scaled by ε.

We now derive an explicit formula for $\partial V[\mathbf{X}]/\partial \mathbf{Y}$. For an alternative derivation see [7, 20].

PROPOSITION 3.7. *The mapping from* $\{(\mathbf{X}_1,..., \mathbf{X}_N) \in (0; 1)^{2N} : \mathbf{X}_i \neq \mathbf{X}_j$ *for* $i \neq j\}$ *into* R^N *which maps the cartesian coordinates of* $\mathbf{X}_1,..., \mathbf{X}_N$ *onto the areas* $V[\mathbf{X}_1],..., V[\mathbf{X}_N]$ *is once continuously differentiable. The partial derivatives are given by*

$$\frac{\partial V[\mathbf{X}]}{\partial \mathbf{Y}} = \frac{\mathbf{Y} - \mathbf{C}[\mathbf{X}, \mathbf{Y}]}{|\mathbf{Y} - \mathbf{X}|} A[\mathbf{X}, \mathbf{Y}] \tag{3.23}$$

if $\mathbf{X} \neq \mathbf{Y}$. *Furthermore,*

$$\frac{\partial V[\mathbf{X}]}{\partial \mathbf{X}} = -\sum_{\mathbf{Y} \neq \mathbf{X}} \frac{\partial V[\mathbf{Y}]}{\partial \mathbf{X}} \tag{3.24}$$

and

$$\frac{\partial V[\mathbf{X}]}{\partial \mathbf{X}} = -\sum_{\mathbf{Y} \neq \mathbf{X}} \frac{\partial V[\mathbf{X}]}{\partial \mathbf{Y}}. \tag{3.25}$$

Proof. We distinguish the following three cases for a given $\mathbf{Y}$:

(i) $\mathbf{X} = \mathbf{Y}$,

(ii) $\mathbf{X} \in Nb[\mathbf{Y}]$ (so $\mathbf{X} \neq \mathbf{Y}$),

(iii) $\mathbf{X} \neq \mathbf{Y}$ and $\mathbf{X} \notin Nb[\mathbf{Y}]$.

We first observe

$$\frac{\partial V[\mathbf{X}]}{\partial \mathbf{Y}} = 0 \qquad \text{in case (iii)}. \tag{3.26}$$

Case (i) may be reduced to case (ii) using (3.26) and one of the two formulae, (3.24), (3.25).

Equation (3.24) may be derived in the following way. Equation (3.26) ensures that the sum in (3.24) is finite. The summation can be restricted to the points in a finite number m of periods. $\sum_{\mathbf{X}} \partial V[\mathbf{X}]/\partial \mathbf{Y}$ is the rate of change of the total area of the Voronoi polygons associated with those points. This total area equals m. Therefore (3.24) follows.

To see (3.25), let the points $\mathbf{X}_1,..., \mathbf{X}_N$ all move at the same constant velocity $\mathbf{u}$. Then

$$\sum_{\mathbf{Y}} \frac{\partial V[\mathbf{X}]}{\partial \mathbf{Y}} \cdot \mathbf{u} \tag{3.27}$$

is the rate of change of $V[\mathbf{X}]$. But $V[\mathbf{X}]$ clearly does not change at all, $P[\mathbf{X}]$ is just translated. Since $\mathbf{u}$ is arbitrary, (3.25) follows.

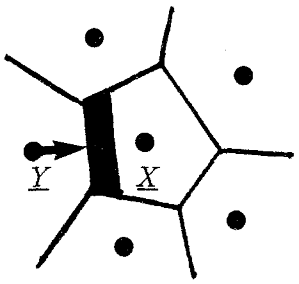


FIG. 8. The loss of area of $P[\mathbf{X}]$ occurring when $\mathbf{Y} \in Nb(\mathbf{X})$ moves towards $\mathbf{X}$. The displacement of $\mathbf{Y}$ is $\varepsilon\mathbf{n}$; $\varepsilon < 0$ here. The area of the shaded strip is $\frac{1}{2}|\varepsilon|\ A[\mathbf{X}, \mathbf{Y}] + O(\varepsilon^2)$.

We turn to case (ii) now. We first consider the derivative of $V[\mathbf{X}]$ in the direction $(\mathbf{Y}-\mathbf{X})/|\mathbf{Y}-\mathbf{X}|$. This derivative is

$$\tfrac{1}{2}A[\mathbf{X}, \mathbf{Y}] \tag{3.28}$$

as can be seen from Fig. 8.

We next consider the derivative in the direction parallel to the face $P[\mathbf{X}] \cap P[\mathbf{Y}]$; see Fig. 9. Up to corrections of area $O(\varepsilon^2)$, both the gained triangle and the lost triangle are similar to the triangle formed by $\mathbf{X}$, $\mathbf{Y}$, $\mathbf{Y} + \varepsilon\boldsymbol{\tau}$. The scaling factors are

$$(-|\tfrac{1}{2}(\mathbf{X}+\mathbf{Y}) - \mathbf{C}[\mathbf{X}, \mathbf{Y}]| + \tfrac{1}{2}A[\mathbf{X}, \mathbf{Y}])/|\mathbf{X}-\mathbf{Y}| \tag{3.29}$$

for the gained triangle and

$$(|\tfrac{1}{2}(\mathbf{X}+\mathbf{Y}) - C[\mathbf{X}, \mathbf{Y}]| + \tfrac{1}{2}A[\mathbf{X}, \mathbf{Y}])/|\mathbf{X}-\mathbf{Y}| \tag{3.30}$$

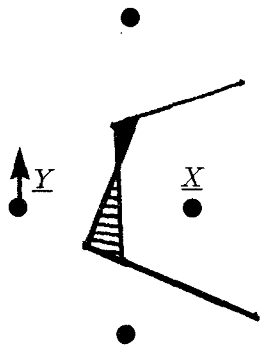


FIG. 9. The change of $P[\mathbf{X}]$ occurring when $\mathbf{Y} \in Nb(\mathbf{X})$ moves parallel to $A[\mathbf{X}, \mathbf{Y}]$. The displacement of $\mathbf{Y}$ is $\varepsilon\boldsymbol{\tau}$. $\boldsymbol{\tau}$ is of length 1. The solidly shaded triangle is the area lost by $P[\mathbf{X}]$ through the motion of $\mathbf{Y}$. The lightly shaded triangle is the gained area.

for the lost triangle. Therefore the total area change is

$$(\varepsilon/2)\,|\mathbf{X}-\mathbf{Y}|\,[(\tfrac{1}{2}A[\mathbf{X},\mathbf{Y}]-|\tfrac{1}{2}(\mathbf{X}+\mathbf{Y})-\mathbf{C}[\mathbf{X},\mathbf{Y}]|)^2 \\ -(\tfrac{1}{2}A[\mathbf{X},\mathbf{Y}]+|\tfrac{1}{2}(\mathbf{X}+\mathbf{Y})-\mathbf{C}[\mathbf{X},\mathbf{Y}]|)^2]/|\mathbf{X}-\mathbf{Y}|^2, \tag{3.31}$$

which is

$$-\varepsilon A[\mathbf{X},\mathbf{Y}]|\,\tfrac{1}{2}(\mathbf{X}+\mathbf{Y})-\mathbf{C}[\mathbf{X},\mathbf{Y}]|/|\mathbf{X}-\mathbf{Y}|. \tag{3.32}$$

Area is lost moving in the direction of the center $\mathbf{C}[\mathbf{X},\mathbf{Y}]$, and area is gained moving away from $\mathbf{C}[\mathbf{X},\mathbf{Y}]$. From what we have shown, it follows that

$$\frac{\partial V[\mathbf{X}]}{\partial \mathbf{Y}}=\frac{1}{2}A[\mathbf{X},\mathbf{Y}]\frac{\mathbf{Y}-\mathbf{X}}{|\mathbf{Y}-\mathbf{X}|}+\frac{A[\mathbf{X},\mathbf{Y}]}{|\mathbf{X}-\mathbf{Y}|}\left(\frac{1}{2}(\mathbf{X}+\mathbf{Y})-\mathbf{C}[\mathbf{X},\mathbf{Y}]\right), \tag{3.33}$$

which is equivalent with (3.23). ∎

We gave a proof for Proposition 3.7 in two space dimensions. However, Eq. (3.23) is correct in all dimensions, if $\mathbf{C}[\mathbf{X},\mathbf{Y}]$ is defined as the center of mass of the face between $\mathbf{X}$ and $\mathbf{Y}$. The derivation of (3.23) given in [20, 7] shows this.

To prove the weak consistency of D with the continuous divergence operator, we first observe that the argument which was used to prove (3.8) can be extended in the following way.

Let A be a linear differential operator. Assume that a formula of the form

$$\int_P A\phi(\mathbf{x})\,d\mathbf{x}=\int_{\partial P}\sigma\phi(\mathbf{x})\,ds \tag{3.34}$$

holds for all smooth functions ϕ, where σ is a linear differential operator on ∂P, possibly of order zero. In the case where A is the Laplace operator, $\sigma\phi=\partial\phi/\partial\mathbf{n}$. Since σ depends on P, we also use the notation

$$\sigma=\sigma_P.$$

We discretize A by

$$V[\mathbf{X}_k]\,L\phi(\mathbf{X}_k):=\sum_{\mathbf{Y}\neq\mathbf{X}_k} I[\mathbf{X}_k,\mathbf{Y}], \tag{3.35}$$

where

$$I[\mathbf{X}_k,\mathbf{Y}]=\int_{P[\mathbf{X}_k]\cap P[\mathbf{Y}]}\sigma\phi(\mathbf{x})\,ds+O(h^2). \tag{3.36}$$

(If we replace $O(h^2)$ by $O(h^3)$ here, pointwise consistency of first order of L and Λ follows immediately.) We assume in addition that

$$\int_{P[\mathbf{X}_k] \cap P[\mathbf{Y}]} \sigma_{P[\mathbf{X}_k]} \phi(\mathbf{x})\, ds = -\int_{P[\mathbf{X}_k] \cap P[\mathbf{Y}]} \sigma_{P[\mathbf{Y}]} \phi(\mathbf{x})\, ds \tag{3.37}$$

$$I[\mathbf{X}, \mathbf{Y}] = -I[\mathbf{Y}, \mathbf{X}] + O(h^3). \tag{3.38}$$

PROPOSITION 3.8. *Equations* (3.34)–(3.38) *together with the non-degeneracy assumption* (3.7) *imply the weak consistency of L with Λ.*

The proof exactly duplicates that of Proposition 3.2. Analogous propositions hold if Λ and L act on vectors rather than scalars, or if the $P[\mathbf{X}]$ are not the Voronoi polygons but some different control areas with mutually disjoint interiors, covering the area of interest.

From Proposition 3.8, we now conclude the weak consistency of D.

PROPOSITION 3.9. *Let the non-degeneracy assumption* (3.7) *be satisfied. Then the operator D is weakly consistent with the continuous divergence operator, i.e.,*

$$\sum_k \phi(\mathbf{X}_k)\, D\mathbf{u}(\mathbf{X}_k)\, V[\mathbf{X}_k] = \int_{[0;1]^2} \phi(\mathbf{x})\, \nabla \cdot \mathbf{u}(\mathbf{x})\, d\mathbf{x} + O(h) \tag{3.39}$$

for any smooth periodic (scalar and vector) functions ϕ and $\mathbf{u}$.

Proof.

$$\int_P \nabla \cdot \mathbf{u}\, d\mathbf{x} = \int_{\partial P} \mathbf{u}(\mathbf{x}) \cdot \mathbf{n}\, ds, \tag{3.34b}$$

i.e.,

$$\sigma \mathbf{u} = \mathbf{u} \cdot \mathbf{n},$$

$$I[\mathbf{X}, \mathbf{Y}] = \left[\frac{1}{2}(\mathbf{u}(\mathbf{Y}))\, \Lambda[\mathbf{X}, \mathbf{Y}] \frac{\mathbf{Y} - \mathbf{X}}{|\mathbf{Y} - \mathbf{X}|}\right] + \left[\frac{\mathbf{u}(\mathbf{Y}) - \mathbf{u}(\mathbf{X})}{|\mathbf{Y} - \mathbf{X}|} \Lambda[\mathbf{X}, \mathbf{Y}] \left(\frac{1}{2}(\mathbf{X} + \mathbf{Y}) - \mathbf{C}[\mathbf{X}, \mathbf{Y}]\right)\right]. \tag{3.35b}$$

The first summand in (3.35b) originates from the normal component of (3.33), the second from the tangential component. In the first summand, one obtains $-\mathbf{u}(\mathbf{X})$ rather than $\mathbf{u}(\mathbf{X})$ from (3.33) and (3.26). To change the sign is correct because the integral over $\partial P[\mathbf{X}]$ of the exterior unit normal is zero. We obtain

$$I[\mathbf{X}_k, \mathbf{Y}] = \int_{P[\mathbf{X}_k] \cap P[\mathbf{Y}]} \mathbf{u}(\mathbf{x}) \cdot \mathbf{n}\, ds + O(h^2). \tag{3.36b}$$

This relation would remain true if we dropped the second summand in (3.35b), which is itself of order $O(h^2)$. We thus obtain an alternative discretization $\tilde{D}$ of ∇,

$$V[\mathbf{X}]\,\tilde{D}\mathbf{u}(\mathbf{X}) := \sum_{\mathbf{Y}\neq\mathbf{X}} \frac{1}{2}(\mathbf{u}(\mathbf{X})+\mathbf{u}(\mathbf{Y}))\,A[\mathbf{X},\mathbf{Y}]\,\frac{\mathbf{Y}-\mathbf{X}}{|\mathbf{Y}-\mathbf{X}|}. \tag{3.40}$$

$\tilde{D}$ can also be derived by discretizing the divergence theorem on the Voronoi polygons in the most obvious way.

Furthermore we obtain

$$\int_{P[\mathbf{X}_k]\cap P[\mathbf{Y}]} \sigma_{P[\mathbf{X}_k]}\mathbf{u}(\mathbf{x})\,ds = -\int_{P[\mathbf{X}_k]\cap P[\mathbf{Y}]} \sigma_{P[\mathbf{Y}]}\mathbf{u}(\mathbf{x})\,ds, \tag{3.37b}$$

$$I[\mathbf{X},\mathbf{Y}] = -I[\mathbf{Y},\mathbf{X}] \qquad \text{(exactly)}. \tag{3.38b}$$

So (3.39) follows. ▮

We have introduced the operator $\tilde{D}$ as an additional illustration of Proposition 3.8, but shall not use it in later sections. Numerical results show that the accuracy of the fractional step method introduced in Section 6 deteriorates when the projection onto the kernel of D is replaced by the projection onto the kernel of $\tilde{D}$.

PROPOSITION 3.10. *D and $\tilde{D}$ are pointwise inconsistent.*

Proof. Counterexamples can be constructed easily; see [6]. The inconsistency of D was first shown by M. McCracken (unpublished).

We give here a counterexample which proves the inconsistency of D. The argument is a reformulation of our previous considerations concerning the example (3.21), (3.22) and shows a statement slightly more general than Proposition 3.10. As mentioned above, the area of the Voronoi polygon of a marker at (0, 0) will, in general, not be constant. This implies that there are fluid marker configurations for which $D\mathbf{u}(0,0)\neq 0$, if $\mathbf{u}(\mathbf{x}) = \binom{x_2}{0}$. Take such a configuration and scale it by ε. This causes scaling of the gradients $\partial V[(0,0)]/\partial\mathbf{Y}$ by ε and scaling of $V[(0,0)]$ by ε^2. Because of $\mathbf{u}(\varepsilon\mathbf{Y}) = \varepsilon\mathbf{u}(\mathbf{Y})$, $Du(0,0)$ remains unchanged. The same argument shows the pointwise inconsistency of any discrete divergence defined as the relative rate of change of the area of a cell which is scaled by ε if the marker configuration is scaled by ε. ▮

Again, we briefly discuss the case of non-periodic boundary conditions. If we impose the condition $\mathbf{u}\cdot\mathbf{n}=0$ on $\partial(0;1)^2$, then the foregoing results remain virtually unchanged. In this case, the summation in (3.19) extends over $\mathbf{Y}=\mathbf{X}_j$, $j=1,\ldots,N$. In (3.39), we now assume that $\mathbf{u}$ and ϕ are smooth on $[0;1]^2$, $\mathbf{u}\cdot\mathbf{n}=0$ on $\partial(0;1)^2$.

3.3. *Discrete Gradient Operator*

We define $\mathbf{G}$ as the negative adjoint of D with respect to the discrete L^2 inner product (3.15). We also define $\tilde{\mathbf{G}}$ as the negative adjoint of $\tilde{D}$ with respect to (3.15).

PROPOSITION 3.11. *$\mathbf{G}$ and $\tilde{\mathbf{G}}$ are weakly consistent with the continuous gradient operator.*

Proof. This follows from the weak consistency of $D, \tilde{D}$ by integrating by parts. ▌

From (3.25), it follows that the kernel of $\mathbf{G}$ contains the constants. (Similarly, it follows from (3.26) that the kernel of D contains the constant vector fields.) Therefore $(D\mathbf{u}, 1) = 0$ for all $\mathbf{u}$, and $L\phi = D\mathbf{u}$ always has a solution ϕ.

The kernel of $\mathbf{G}$ may contain non-constant functions. If, for example, $\mathbf{X}_1, ..., \mathbf{X}_N$ lie on a square grid, and if N is the square of an even integer, then the kernel of $\mathbf{G}$ is four-dimensional, for $\mathbf{G}$ is then the discretization of $\boldsymbol{\nabla}$ with central difference quotients. We have been unable to find a general estimate of the dimension of the kernel of $\mathbf{G}$.

In subsequent sections, $\mathbf{G}$ is used as a tool for orthogonally projecting discrete vector fields onto the kernel of D. It is for this reason that we have defined the discrete gradient as the negative adjoint of the discrete divergence; see Eqs. (5.1)–(5.3), which describe the orthogonal projection onto the kernel of D if D is any arbitrary linear operator and $\mathbf{G}$ is its negative adjoint.

3.4. *Additional Remarks on Consistency*

The lack of pointwise consistency does not imply lack to convergence. This is demonstrated numerically in Section 6. We note that a similar behavior is typical for finite element discretizations of elliptic equations, whose pointwise consistency is in general entirely unrelated to the convergence and accuracy of the resulting approximations. As an example, consider the discretization of Poisson's equation based on triangles of type (2) (Ciarlet [12]), using a triangulation constructed by cutting the cells of a regular square mesh into triangles. This discretization can easily be shown to be pointwise inconsistent. Nevertheless, the L^2-norm of the difference between the solution of the continuous problem and the solution of the discrete problem is of size $O(h^3)$, where h is the mesh size of the square grid; see [12].

In the present context, the weak consistency serves only as a heuristic justification of our discrete operators. It might, however, prove useful in a convergence analysis of the fractional step method presented in Section 6.

4. SOLUTION OF DISCRETE HELMHOLTZ EQUATIONS ON IRREGULAR GRIDS

In this section, we consider the fast numerical solution of

$$-L\phi + c\phi = f \qquad (c \geqslant 0). \tag{4.1}$$

We use a two-grid iteration which is identical with the well-known multigrid correction cycle, replacing the coarsening of the grid by regularization.

Consider a regular grid of the form

$$\Gamma^h = \{((i+0.5)h, (j+0.5)h) \colon i, j \in Z\} \tag{4.2}$$

with $h = 1/n$ and $n \approx N^{1/2}$. Let R be a linear interpolation operator which maps functions defined on S_N onto functions defined on Γ^h. Let P be a linear inter- [illegible]

We introduce the abbreviations

$$A := -L + cI \tag{4.3}$$

and

$$B := \text{discretization of } -\Delta + cI \text{ on } \Gamma^h \text{, using the standard 5-point operator.} \tag{4.4}$$

A and B are operators which act on periodic grid functions.

A simple iterative method for (4.1) is defined by

$$\phi^{n+1} := \phi^n + PB^{-1}R(f - A\phi^n). \tag{4.5}$$

If B is singular, i.e., if $c = 0$, B^{-1} denotes the pseudoinverse of B.

If P and R are chosen in a simple, straightforward way, the iteration (4.5) converges very poorly if at all. The reason is that P and R introduce smoothing, which prevents highly oscillatory errors from converging fast. Therefore (4.5) has to be supplemented by a method which is efficient on such errors. We choose a suitable relaxation method for this, obtaining the two-grid cycle

Step 1. ν_1 *relaxation sweeps,*

Step 2. (4.5), (4.6)

Step 3. ν_2 *relaxation sweeps.*

ν_1 and ν_2 are small integers, typically 1 or 2. Definition (4.5) requires the solution of problems involving B. In order to exhibit as clearly as possible the properties of the basic method (4.6), we solve these problems exactly, using the Fast Fourier Transform. A very fast approximate solver, for example, a multigrid cycle requiring only $O(N)$ operations, could be used instead. This would result in a slight reduction of the computational work. The convergence factors could be expected to deteriorate only insignificantly if a sufficiently efficient cycle were used; see, e.g., Stüben and Trottenberg [23].

We remark that the cycle (4.6) costs $O(N \log N)$ operations. Since the fractional step method discussed in Section 6 requires a fixed number of cycles, independent of N, in every time step, the total amount of work per time step required for the solution of Poisson and Helmholtz problems is $O(N \log N)$.

We have to choose P, R and the relaxation scheme. Our objective is to choose these components such that the resulting cycle is efficient, assuming that the grid points are roughly uniformly distributed. This assumption is satisfied in the examples in Section 6.

We discuss the choice of P and R first. We list some desirable properties:

(CP) P preserves constants: $P(1)(\mathbf{X}_j) = 1$ for all j.

(CR) R preserves constants: $R(1)(\mathbf{x}) = 1$ for all $\mathbf{x} \in \Gamma^h$.

($I0P$) P preserves zero mean values:

$$\text{If } \sum_{\mathbf{x}} \phi(\mathbf{x}) h^2 = 0, \text{ then } \sum_k (P\phi)(\mathbf{X}_k)\, V[\mathbf{X}_k] = 0.$$

($I0R$) R preserves zero mean values:

$$\text{If } \sum_k \Phi(\mathbf{X}_k)\, V[\mathbf{X}_k] = 0, \text{ then } \sum_{\mathbf{x}} (R\Phi)(\mathbf{x}) h^2 = 0.$$

Here $\sum_{\mathbf{x}}$ stands for $\sum_{\mathbf{x} \in \Gamma^h \cap [0;1]^2}$. ($I0P$) and ($I0R$) state that the compatibility conditions are preserved if $c = 0$. For our two-grid algorithm, ($I0R$) is particularly useful, since it ensures solvability of the problems on Γ^h if $c = 0$.

We focus our attention on operators defined as convolutions of the form

$$(P\phi)(\mathbf{X}) := \sum_{\mathbf{x} \in \Gamma^h} \phi(\mathbf{x})\, \delta_h(\mathbf{X} - \mathbf{x}) h^2, \tag{4.7}$$

$$(R\Phi)(\mathbf{x}) := \sum_{\mathbf{X} \in S_N} \Phi(\mathbf{X})\, \delta_h(\mathbf{x} - \mathbf{X})\, V[\mathbf{X}], \tag{4.8}$$

where δ_h is a spread-out model of the delta distribution with support of linear size $O(h)$. (4.7) and (4.8) imply $P = R^*$. It is clear that (CR) cannot be satisfied by an operator R of this form, for it is possible that no point in S_N lies close to $\mathbf{x} \in \Gamma^h$, in which case $R\Phi(\mathbf{x})$ is 0, no matter what Φ is. (CP), on the other hand, is easily satisfied, for example, with

$$\delta_h(\mathbf{x}) = \delta_h(x_1)\, \delta_h(x_2), \tag{4.9}$$

with

$$\delta_h(x) = \begin{cases} \dfrac{1}{h}\left(1 - \dfrac{|x|}{h}\right) & \text{if } |x| \leqslant h, \\ 0 & \text{otherwise.} \end{cases} \tag{4.10}$$

Note that (4.7), (4.9), (4.10) is just piecewise bilinear interpolation. It is easy to conclude that ($I0R$) is satisfied while ($I0P$) is not. In fact, R preserves discrete integrals in general. These conclusions depend only on the fact that $P = R^*$.

Equations (4.7)–(4.10) specify our choice of P and R.

It remains to choose a relaxation scheme. Using Gauss–Seidel relaxation, we obtain a performance which is disappointing in comparison with many multigrid methods on regular grids. This is illustrated by Table II. To explain Table II, we first define the discrete L^2-norm of a function ϕ defined on the grid points $\mathbf{X}_k$ by

$$|\phi|_{L^2} := \sqrt{\sum_{k=1}^{N} V[\mathbf{X}_k]\, \phi(\mathbf{X}_k)^2}.$$

We measure the factors by which the discrete L^2-norm of the residual is reduced per two-grid cycle. Corresponding to each N, Table II has four rows, which have the following meaning. The first three rows show the reduction factors for the first three cycles. Here the continuous problem being solved has the solution

$$\phi(\mathbf{x}) = \sin(2\pi(x_1 - 2x_2)). \tag{4.11}$$

The fourth row shows the reduction factor in the 30th cycle, which is in general a good approximation to the spectral radius of the iteration matrix. To avoid rounding error influence, we take the right-hand side to be identically zero in this case, start with a non-zero initial approximation and rescale the approximation by the convergence factor after each cycle.

The order in which the unknowns are relaxed is determined by the numbering of the grid points, which is random.

Note that the convergence factors in Table II deteriorate rapidly when N increases. The results can be improved significantly by modifying the relaxation

TABLE II

Residual Reduction per Two-Level Cycle, in the 1st, 2nd, 3rd, and 30th Cycle, Measured as Described in the Text

N	Worst case	Best case	Average
100	0.18	0.090	0.14
	0.26	0.083	0.14
	0.30	0.11	0.17
	0.31	0.12	0.18
400	0.34	0.17	0.22
	0.27	0.098	0.17
	0.37	0.13	0.24
	0.42	0.17	0.28
1600	0.34	0.23	0.27
	0.25	0.13	0.17
	0.41	0.21	0.29
	0.52	0.24	0.40

Note. Helmholtz equation, $c = 10$, $\nu_1 = 2$, $\nu_2 = 1$, Gauss–Seidel relaxation. Ten random, uniformly distributed grids for each N. Best results, worst results, and geometric means over all 10 experiments.

method. This modification is an application of a general rule due to Brandt [9] on block relaxation in multigrid methods. The improved relaxation scheme is defined as follows. Introduce the notation

$$V[\mathbf{X}_k](L\phi)(\mathbf{X}_k) =: \sum_{\mathbf{Y}} L[\mathbf{X}_k, \mathbf{Y}]\, \phi(\mathbf{Y}). \tag{4.12}$$

When relaxing $\mathbf{X}_k$, determine the set of all neighbors $\mathbf{Y}$ of $\mathbf{X}_k$ for which

$$L[\mathbf{X}_k, \mathbf{Y}] \geqslant \delta(-L[\mathbf{X}_k, \mathbf{X}_k] + cV[\mathbf{X}_k]), \tag{4.13}$$

where $\delta \in [0; 1]$ remains to be chosen. Change the values in all these neighbors and in $\mathbf{X}_k$ simultaneously so that their equations becomes satisfied. As a result, some of the $\phi(\mathbf{X}_k)$ may be modified more than once during one relaxation sweep. We note that our criterion (4.13) for relaxing points as a block is not equivalent to an analogous criterion based on the distances between the grid points. We have conducted experiments which indicate that such a criterion leads to a less efficient algorithm than ours.

If δ is at least 0.5 and $c \geqslant 0$, the blocks which are relaxed simultaneously are at most of size 2. To see this, recall that the diagonal elements of L are negative, that the off-diagonal elements are positive or zero with at least 3 positive elements in each row, and that the row sums of L are zero. Thus there is at most one neighbor $\mathbf{Y}$ of any point $\mathbf{X}_k$ for which (4.13) is satisfied when $\delta \geqslant 0.5$. With $\delta = 1.0$, we obtain the usual Gauss–Seidel iteration, by a similar argument. We choose $\delta = 0.5$.

We repeat the above experiments with the modified relaxation scheme. The results are displayed in Table III.

TABLE III

Experiments as in Table II, with Modified Relaxation Scheme

N	Worst case	Best case	Average
100	0.083	0.027	0.041
	0.13	0.018	0.035
	0.14	0.020	0.039
	0.15	0.030	0.049
400	0.13	0.087	0.11
	0.084	0.040	0.054
	0.12	0.046	0.072
	0.14	0.054	0.086
1600	0.16	0.13	0.14
	0.048	0.030	0.042
	0.089	0.048	0.069
	0.17	0.074	0.11

The results for $c = 0.0$ are hardly different. Of course, we must project the discrete right-hand side onto its constant-free part to ensure that there is a solution in this case. It is unnecessary to impose any condition such as a prescribed mean or point value to ensure uniqueness of the constant component in the computed solution. The non-uniqueness of the solution may simply be ignored.

We still obtain convergence factors which increase with growing N. Good multigrid methods have convergence factors which are bounded independent of the number of unknowns, with a bound far below 1; see Stüben and Trottenberg [23]. We do not know whether our method has such a property. It is, in any case, satisfactory when used within our fractional step method; see Section 6.

An alternative to the algorithm described here would be to construct coarse grids as nested subsets of the Lagrangian grid $\{\mathbf{X}_1, ..., \mathbf{X}_N\}$. Even though the resulting algorithm would be more complicated than ours, it might have the advantage of performing well even on strongly non-uniform grids. We have not yet tried such a method.

5. Projection of a Vector Field onto the Kernel of the Discrete Divergence Operator

The last tool needed for our fractional step method is an algorithm which projects a given vector field $\mathbf{u}$ on S_N orthogonally onto the kernel of the discrete divergence operator D. This means that we want to find $\mathbf{Pu}$ and q such that

$$\mathbf{u} = \mathbf{Pu} + \mathbf{G}q \tag{5.1}$$

and

$$D\mathbf{Pu} = 0. \tag{5.2}$$

For this purpose, we have to solve

$$D\mathbf{G}q = D\mathbf{u}. \tag{5.3}$$

We have $D\mathbf{G}p = 0 \Rightarrow (D\mathbf{G}p, p) = 0 \Rightarrow -(\mathbf{G}p, \mathbf{G}p) = 0 \Rightarrow \mathbf{G}p = \mathbf{0} \Rightarrow D\mathbf{G}p = 0$. Therefore $\ker(D\mathbf{G}) = \ker(\mathbf{G})$.

This implies that $D\mathbf{u}$ is orthogonal to ker $(D\mathbf{G})$, for if $D\mathbf{G}p = 0$, then $\mathbf{G}p = \mathbf{0}$ and thus $(D\mathbf{u}, p) = (\mathbf{u}, \mathbf{G}p) = (\mathbf{u}, \mathbf{0}) = 0$. Therefore (5.3) has a solution q, and $\mathbf{G}q$ is uniquely determined.

On a square grid with meshwidth h, $D\mathbf{G}$ is the standard 5-point discretization of the Laplace operator with meshwidth $2h$. We ignore the fact that the system can be decoupled into 4 smaller systems which can be solved easily, since this will not be the case on irregular grids. Simple relaxation schemes have no smoothing effect for this discretization, as can immediately be verified by Fourier analysis. The difficulty is related to lack of discrete ellipticity. Problems of this kind have been investigated much more generally by Brandt and Dinar [10].

One might consider solving (5.3) by a conjugate gradient iteration with preconditioning. On a square mesh, it is useless to precondition $D\mathbf{G}$ with L. This can immediately be shown by discrete Fourier analysis. Even if a good preconditioner for $D\mathbf{G}$ could be found, the fact that we do not know much about the kernel of $D\mathbf{G}$, i.e., the kernel of $\mathbf{G}$, would present a problem. A non-trivial null-space does not affect the performance of the conjugate gradient method in exact arithmetic. In floating point arithmetic, however, the method often does not converge when applied to a positive semidefinite matrix with a non-trivial null-space unless the computed residuals are projected into the orthogonal complement of the null-space.

We therefore replace the discrete projection operator

$$I-\mathbf{G}(D\mathbf{G})^{-1}D \tag{5.4}$$

by

$$I-\mathbf{G}L^{-1}D. \tag{5.5}$$

$D\mathbf{G}$ and L are singular. Nevertheless $\mathbf{G}(D\mathbf{G})^{-1}D$ and $\mathbf{G}L^{-1}D$ have well-defined meanings. To apply, for example, $\mathbf{G}L^{-1}D$ to a vector field $\mathbf{u}$, solve $Lq=D\mathbf{u}$ first. Note that $D\mathbf{u}$ is orthogonal to the kernel of L, which is the set of constants. q is unique up to an additive constant, and thus $\mathbf{G}q=\mathbf{G}L^{-1}D\mathbf{u}$ is uniquely determined. $\mathbf{G}(D\mathbf{G})^{-1}D\mathbf{u}$ is obtained analogously.

(5.5) is not a projection operator. In fact, the discrete L^2-norm

$$|I-\mathbf{G}L^{-1}D|=\rho(I-\mathbf{G}L^{-1}D) \tag{5.6}$$

is often, i.e., on many fluid marker configurations, larger than 1. Of course, the eigenvalues whose absolute values are larger than 1 must be negative.

If all eigenvalues of $I-\mathbf{G}L^{-1}D$ lie in $(-1;1]$, then

$$\lim_{j\to\infty}(I-\mathbf{G}L^{-1}D)^j=I-\mathbf{G}(D\mathbf{G})^{-1}D. \tag{5.7}$$

To see this, note that the limit in (5.7) exists if and only if all eigenvalues of $I-\mathbf{G}L^{-1}D$ lie in $(-1;1]$. Furthermore, if $\mathbf{u}^*=\lim_{j\to\infty}(I-\mathbf{G}L^{-1}D)^j\mathbf{u}$, then $(I-\mathbf{G}L^{-1}D)\mathbf{u}^*=\mathbf{u}^*$, hence $\mathbf{G}L^{-1}D\mathbf{u}^*=\mathbf{0}$ and thus $D\mathbf{u}^*=0$. Since $\mathbf{u}-(I-\mathbf{G}L^{-1}D)^j\mathbf{u}$ is a gradient for all j, $\mathbf{u}-\mathbf{u}^*$ is a gradient, and therefore $\mathbf{u}^*$ is the orthogonal projection of $\mathbf{u}$ onto the kernel of D, i.e., $\mathbf{u}^*=\mathbf{G}(D\mathbf{G})^{-1}D\mathbf{u}$.

Since (5.5) is a discretization of the continuous projection operator, it is to be expected that at least those eigenvalues which correspond to smooth eigenfunctions are larger than -1. This motivates the following modification of (5.5),

$$I-\mathbf{G}(I+\omega L)^kL^{-1}D, \tag{5.8}$$

with $\omega\approx 1/\rho(L)$ but $\omega\leqslant 1/\rho(L)$, k integer. In our experiments, we always take

$$\omega:=\frac{1}{\tilde{\rho}}, \tag{5.9}$$

where $\tilde{\rho}$ is the maximum row sum norm of the matrix L, i.e.,

$$\omega = \left(\max_k \frac{2}{V[\mathbf{X}_k]} \sum_{\mathbf{Y} \neq \mathbf{X}_k} \frac{A[\mathbf{X}_k, \mathbf{Y}]}{|\mathbf{Y} - \mathbf{X}_k|} \right)^{-1}. \tag{5.10}$$

One can easily show that ω is at most of order $O(h^2)$. $I + \omega L$ is therefore pointwise consistent with the identity, since $L\phi$ is bounded for $h \to 0$ if ϕ is a fixed smooth periodic function; see Proposition 3.4.

For fixed k, (5.8) can still be considered a discretization of the continuous projection operator. For sufficiently large k, the powers of (5.8) converge to the exact discrete projection operator $I - \mathbf{G}(D\mathbf{G})^{-1}D$.

This motivates the following algorithm, which chooses the smallest possible k itself:

ALGORITHM 5.1.

```
Given $\mathbf{u}$, generate $\mathbf{u}^0, \mathbf{u}^1, \mathbf{u}^2, ...$ with $\mathbf{u}^j \to \mathbf{P}\mathbf{u}$ as follows:
$\mathbf{u}^0 := \mathbf{u}$.
Given $\mathbf{u}^j$, define
$k := 0$; $\mathbf{u}^{j,0} := \mathbf{u}^j$, $q^{j,1} := L^{-1}D\mathbf{u}^{j,0}$; $\mathbf{u}^{j,1} := \mathbf{u}^{j,0} - \mathbf{G}q^{j,1}$.
While $|\mathbf{u}^{j,k+1}| > |\mathbf{u}^j|$:
  $k := k+1$; $q^{j,k+1} := \omega L q^{j,k}$; $\mathbf{u}^{j,k+1} := \mathbf{u}^{j,k} - \mathbf{G}q^{j,k+1}$.
$\mathbf{u}^{j+1} := \mathbf{u}^{j,k+1}$.
```

In our experiments, this algorithm usually requires $k = 0$ when computing $\mathbf{u}^1$ from $\mathbf{u}^0$, $k > 0$ for $j > 1$, sometimes unfortunately $k \gg 0$.

Note that $|\mathbf{u}^{j+1}| \leqslant |\mathbf{u}^j|$. This property ensures the linear stability of the fractional step method introduced in Section 6, even if only a few iterations of the projection Algorithm 5.1 are used.

We give a numerical example. Consider

$$\mathbf{u}(\mathbf{x}) = (\sin(2\pi x_1)\cos(2\pi x_2), 0). \tag{5.11}$$

The divergence-free part of $\mathbf{u}$ is

$$0.5(\sin(2\pi x_1)\cos(2\pi x_2), \ -\cos(2\pi x_1)\sin(2\pi x_2)). \tag{5.12}$$

We compute the divergence-free part of $\mathbf{u}$ numerically, compute the discrete L^2-norms of the discretization errors in the two components, and compute the square [illegible]

iteration or larger than 2 in the second iteration.

One iteration generates an approximation to the continuous solution which is as good as or even better than the approximation obtained after many iterations.

TABLE IV

Discretization Errors Obtained with Algorithm 5.1 for Example (5.11)

N	Iteration	Worst case	Best case	N	Iteration	Worst case	Best case
25	1	0.30	0.22	400	1	0.081	0.070
	2	0.30	0.20		2	0.078	0.061
	3	0.28	0.20		3	0.083	0.063
	4	0.29	0.19		4	0.081	0.064
	5	0.29	0.20		5	0.086	0.064
	10	0.29	0.20		10	0.084	0.062
	15	0.30	0.20		15	0.089	0.062
100	1	0.17	0.13	1600	1	0.041	0.035
	2	0.16	0.11		2	0.040	0.031
	3	0.16	0.11		3	*	0.033
	4	0.16	0.11		4	*	0.034
	5	0.16	0.11		5	*	0.035
	10	0.16	0.11		10	*	0.037
	15	0.16	0.11		15	*	0.037

* $k=50$ is not sufficient.

Note. Twenty random, uniformly distributed grids for each N.

What we have gained by replacing (5.5) with (5.8) are guaranteed stability and convergence to the solution of the discrete projection problem, not higher accuracy.

Note that the truncation error does decrease with growing N here, even though D is not pointwise consistent with the divergence operator. We also observe that the variance of the truncation error seems to decrease with growing N. These two statements are based on numerical results. We have not been able to prove them.

In the periodic case, the projection algorithm presented here is useful in solving the Navier–Stokes equations; see Section 6. The question how to modify our algorithm for different boundary conditions requires further study.

6. A Fractional Step Method for the Navier–Stokes Equations

The two-dimensional incompressible Navier–Stokes equations have the form

$$\begin{gathered} \frac{d\mathbf{u}}{dt} - \nu\,\Delta\mathbf{u} + \nabla p = \mathbf{f} \\ \nabla\cdot\mathbf{u} = 0, \end{gathered} \tag{6.1}$$

where $\mathbf{u}=\mathbf{u}(\mathbf{x},t)\in R^2$, $p=p(\mathbf{x},t)\in R$, $\mathbf{f}=\mathbf{f}(\mathbf{x},t)\in R^2$, $\mathbf{x}\in R^2$, $t\in R$, $t>0$, and $d\mathbf{u}/dt=\mathbf{u}_t+(\mathbf{u}\cdot\nabla)\mathbf{u}$ is the convective derivative of $\mathbf{u}$. We impose an initial condition

$$\mathbf{u}(\mathbf{x},0)=\mathbf{u}_0(\mathbf{x}) \tag{6.2}$$

and the periodicity conditions

$$\mathbf{u}(\mathbf{x}+\mathbf{k}, t)=\mathbf{u}(\mathbf{x}, t), \qquad p(\mathbf{x}+\mathbf{k}, t)=p(\mathbf{x}, t), \quad \mathbf{k}\in Z^2. \tag{6.3}$$

We consider the following fractional step method for (6.1)–(6.3),

$$\begin{aligned}\frac{\tilde{\mathbf{u}}^{n+1}-\mathbf{u}^n}{\Delta t}-\nu L^n\tilde{\mathbf{u}}^{n+1}&=\mathbf{f}^{n+1}\\ \mathbf{u}^{n+1}&=\mathbf{P}^n\tilde{\mathbf{u}}^{n+1}\\ \mathbf{X}^{n+1}&=\mathbf{X}^n+\Delta t\mathbf{u}^{n+1}.\end{aligned} \tag{6.4}$$

Here $\mathbf{u}^n$, $\tilde{\mathbf{u}}^{n+1}$, $\mathbf{u}^{n+1}$ and $\mathbf{f}^{n+1}$ denote functions defined on S_N^n, the set S_N at time $n\,\Delta t$. $\mathbf{f}^{n+1}(\mathbf{X}_j^n+\mathbf{k})=\mathbf{f}(\mathbf{X}_j^n, (n+1)\,\Delta t)$. L^n denotes the discrete Laplacian on S_N^n. $\mathbf{P}^n$ is the orthogonal projection onto the kernel of D^n, where D^n is the discrete divergence operator on S_N^n. The points $\mathbf{X}_j^n+\mathbf{k}$ $(1\leqslant j\leqslant N,\ \mathbf{k}\in Z^2)$ in S_N^n form the infinite vector $\mathbf{X}^n$.

Writing $\mathbf{P}^n\tilde{\mathbf{u}}^{n+1}=: \tilde{\mathbf{u}}^{n+1}-\mathbf{G}^n q^{n+1}$, we obtain

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t}-\nu L^n\tilde{\mathbf{u}}^{n+1}+\frac{\mathbf{G}^n q^{n+1}}{\Delta t}=\mathbf{f}^{n+1} \quad \text{and} \quad D^n\mathbf{u}^{n+1}=0. \tag{6.5}$$

Therefore $p^{n+1}:=q^{n+1}/\Delta t$ is an approximation of the pressure. We note that the formulas (6.5) with $L^n\mathbf{u}^{n+1}$ instead of $L^n\tilde{\mathbf{u}}^{n+1}$, together with the third equation of (6.4), define the method introduced in [20].

Method (6.4) is unconditionally linearly stable in the sense that

$$|\mathbf{u}^{n+1}|_{(n)}\leqslant|\mathbf{u}^n|_{(n)}+\Delta t\,|\mathbf{f}^{n+1}|_{(n)}, \tag{6.6}$$

where $|\cdot|_{(n)}$ denotes the discrete L^2-norm on S_N^n. This follows from Proposition 3.5. Using the projection algorithm introduced in Section 5, linear stability is guaranteed even though we do not apply the operator $\mathbf{P}^n$ exactly.

We also consider the following modification of (6.4).

$$\begin{aligned}\frac{\tilde{\mathbf{u}}^{n+1}-\mathbf{P}^n\mathbf{u}^n}{\Delta t}-\nu L^n\tilde{\mathbf{u}}^{n+1}&=\mathbf{P}^n\mathbf{f}^{n+1}\\ \mathbf{u}^{n+1}&=\mathbf{P}^n\tilde{\mathbf{u}}^{n+1}\\ \mathbf{X}^{n+1}&=\mathbf{X}^n+\Delta t\mathbf{u}^{n+1}.\end{aligned} \tag{6.4b}$$

Note that this method requires two, not three applications of $\mathbf{P}^n$ per time step.

Writing, as before, $\mathbf{P}^n\tilde{\mathbf{u}}^{n+1}=: \tilde{\mathbf{u}}^{n+1}-\mathbf{G}^n q^{n+1}$, and writing

$$\mathbf{P}^n(\mathbf{u}^n+\Delta t\mathbf{f}^{n+1})=: \mathbf{u}^n+\Delta t\mathbf{f}^{n+1}-\mathbf{G}^n r^{n+1},$$

we obtain

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t} - \nu L^n \tilde{\mathbf{u}}^{n+1} + \mathbf{G}^n\left(\frac{q^{n+1}+r^{n+1}}{\Delta t}\right) = \mathbf{f}^{n+1} \quad \text{and} \quad D^n \mathbf{u}^{n+1} = 0. \tag{6.5b}$$

Therefore $p^{n+1} := (q^{n+1}+r^{n+1})/\Delta t$ is an approximation of the pressure.

We use method (6.4) for Test Problems 1 and 2. For Test Problem 3, method (6.4b) appears to be somewhat more accurate.

TEST PROBLEM 1. We construct an exterior force density such that the flow becomes

$$\begin{aligned} u_1(\mathbf{x}, t) &= \sin\left(\frac{\pi}{2} t\right) \cos(2\pi x_1) \sin(2\pi x_2) \\ u_2(\mathbf{x}, t) &= -\sin\left(\frac{\pi}{2} t\right) \sin(2\pi x_1) \cos(2\pi x_2) \\ p(\mathbf{x}, t) &= \sin\left(\frac{\pi}{2} t\right) \cos(2\pi x_1) \cos(2\pi x_2). \end{aligned} \tag{6.7}$$

We measure the discrete L^2-norms of the errors in velocity and pressure at time 1. In every time step, we use only one step of Algorithm 5.1. Initially, the $\mathbf{X}_j$ are at the positions $((k_1+0.5)h, (k_2+0.5)h)$, with $0 \leqslant k_1 \leqslant N^{1/2}-1$, $0 \leqslant k_2 \leqslant N^{1/2}-1$. We expect that the solution of the scalar Helmholtz and Poisson problems up to rounding error accuracy is necessary only at time 0. At a later time step, the values from the previous time step should provide an excellent initial guess, and little work should be required to improve this guess such that the truncation error level is reached. We use only one two-grid cycle per scalar problem, with $\nu_1 = 2$, $\nu_2 = 1$. Table V shows results of some of our experiments. We use the abbreviations $h_N := 1/N^{1/2}$, $e_1 :=$ error in u_1, $e_2 :=$ error in u_2, and $e_p :=$ error in p. Since the average of the exact pressure is zero, we project the computed approximation for the pressure onto its constant free part with respect to the discrete L^2-product before measuring e_p. Note that the error in the velocity components is reduced faster than the error in the pressure. We have no explanation for this observation.

TABLE V

Errors in Velocity and Pressure, Test Problem 1, $\Delta t = h_N$, $\nu = 0.1$, Starting on Square Grids

N	e_1	e_2	e_p
100	0.35	0.35	0.11
400	0.12	0.12	0.12
1600	0.064	0.065	0.082

Table VI contains results obtained with random initial grids. As before, we use one two-grid cycle per scalar problem. Note that the errors in the velocity components do not differ very much from those displayed in Table V. As in Section 5, we observe that the variance of the error appears to decrease for increasing N.

Solving all scalar problems up to rounding error accuracy, but still applying only one step of Algorithm 5.1, one obtains results which differ only insignificantly from those displayed in Table V. This supports our claim that one two-grid cycle per scalar problem suffices.

Using very small viscosity coefficients, for example $\nu = 10^{-4}$, we still obtain reasonable results for this problem. However, some of our results for Test Problem 3 indicate that the method may not be generally applicable for such viscosity coefficients.

TEST PROBLEM 2. We consider a shear flow

$$\begin{aligned} u_1(\mathbf{x}, t) &= \sin(2\pi x_2) \\ u_2(\mathbf{x}, t) &= 0 \\ p(\mathbf{x}, t) &= 0. \end{aligned} \tag{6.8}$$

PROPOSITION 6.1. *If the initial marker configuration is a rectangular grid with meshwidths Δx_1 and Δx_2,*

$$\Delta x_2 \geqslant \tfrac{1}{2} \Delta x_1, \tag{6.9}$$

and if the initial velocity and the exterior force are independent of x_1 and parallel to the x_1 axis, then the discrete approximation for the second velocity component obtained by (6.4) *or* (6.4b) *is* 0.

Proof. The statement would be obvious if $\mathbf{P}^n$ were replaced by the identity operator. Therefore, it suffices to prove the following statement.

TABLE VI

Experiments as in Table V, Starting on 20 Uniformly Distributed, Random Grids

	e_1			e_2			e_p		
N	Worst	Best	Average	Worst	Best	Average	Worst	Best	Average
100	0.51	0.31	0.38	0.48	0.29	0.36	0.37	0.20	0.27
400	0.17	0.13	0.15	0.18	0.13	0.15	0.14	0.086	0.12
1600	0.064	0.059	0.061	0.062	0.059	0.061	0.076	0.069	0.072

Assume that

$$\{\mathbf{X}_1,..., \mathbf{X}_N\} = \left\{\left(\left(i+\frac{1}{2}\right)\Delta x_1+\delta_j, \left(j+\frac{1}{2}\right)\Delta x_2\right): i=0,..., \frac{1}{\Delta x_1}-1, j=0,..., \frac{1}{\Delta x_2}-1\right\} \tag{6.10}$$

with numbers

$$\delta_j \in [-\tfrac{1}{2}\Delta x_1, \tfrac{1}{2}\Delta x_1] \tag{6.11}$$

independent of i, and assume that (6.9) holds. Consider a vector field

$$\mathbf{u}(\mathbf{x}) = \begin{pmatrix} u_1(x_2) \\ 0 \end{pmatrix}. \tag{6.12}$$

Then

$$D\mathbf{u}(\mathbf{X}_k) = 0 \tag{6.13}$$

for all k.

To prove this, first consider two rows of fluid markers and the associated Voronoi diagram in the infinite horizontal strip $\{\mathbf{x} \in R^2: x_1 \in R,\ x_2 \in [0; 1]\}$. If the locations of the fluid markers are

$$((i+\tfrac{1}{2})\,\Delta x_1+\delta_u, \tfrac{1}{2}+\delta_y), \qquad i \in Z \tag{6.14}$$

in the upper row, and

$$((i+\tfrac{1}{2})\,\Delta x_1+\delta_l, \tfrac{1}{2}-\delta_y), \qquad i \in Z \tag{6.15}$$

in the lower row, with

$$\delta_u \in [-\tfrac{1}{2}\Delta x_1, \tfrac{1}{2}\Delta x_1], \tag{6.16}$$

$$\delta_l \in [-\tfrac{1}{2}\Delta x_1, \tfrac{1}{2}\Delta x_1], \tag{6.17}$$

and

$$\delta_y \in (0; \tfrac{1}{2}), \tag{6.18}$$

then all Voronoi polygons have the same shape and therefore the same area.

Next we consider the case of more than two rows of markers. Inequality (6.9) ensures that the Voronoi polygons in row $j+1$ are not adjacent to the Voronoi polygons in row $j-1$. The Voronoi polygons in row j are similar to each other. Our considerations regarding two rows show that the total area which the polgons in row j occupy within one period is constant. Therefore each individual polygon

TABLE VII

Errors in Velocity and Pressure, Test Problem 2,
$\Delta t = h_N$, $\nu = 1.0$, Starting on 10 Random, Uniformly Distributed Grids

	e_1			e_2			e_p		
N	Worst	Best	Average	Worst	Best	Average	Worst	Best	Average
100	0.075	0.045	0.060	0.071	0.041	0.056	0.061	0.041	0.053
400	0.035	0.025	0.030	0.035	0.024	0.029	0.033	0.022	0.027
1600	0.016	0.014	0.015	0.015	0.013	0.014	0.015	0.013	0.014

We start the calculation on random grids. We obtain the results displayed in Table VII, suggesting first order convergence as the previous results. Figure 10 shows the exact velocity profile and a computed velocity profile at time 1. The parameters are as in Table VII, with $N = 1600$. Here the first 10 markers have the initial positions $(0.5, (j-0.5)/10)$, $j = 1, \ldots, 10$, the remaining 1590 markers have random initial positions. The figure shows the computed velocity vectors and the computed x_2-coordinates of the first 10 markers at time 1. To facilitate the comparison of the two profiles, the computed x_1-coordinates at time 1 have been reset to 0.5, and the lengths of the vectors at time 1 have been scaled such that the longest vector has the same length as at time 0.

TEST PROBLEM 3. We consider an initial vortex blob centered at (0.5, 0.5) and compute the flow which results in the absence of any exterior force. To obtain the

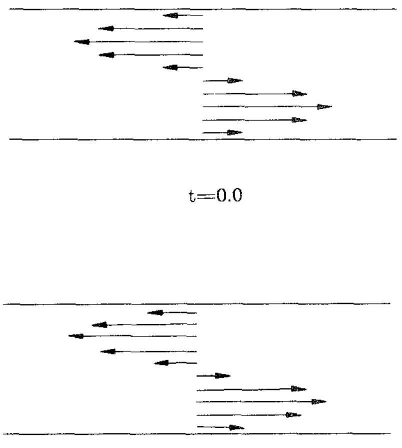


FIG. 10. Test Problem 2, computed velocity profiles at $t = 0$ and $t = 1$.

initial velocity field, we solve the discrete Poisson problem with periodic boundary conditions and the right-hand side

$$\begin{cases} 10.0\cdot\left(1+\cos\left(\frac{\pi(x_1-0.5)}{0.2}\right)\right)\left(1+\cos\left(\frac{\pi(x_2-0.5)}{0.2}\right)\right)+c & \\ & \text{if } |x_1-0.5|\leqslant 0.2 \text{ and } |x_2-0.5|\leqslant 0.2 \\ c & \text{otherwise,} \end{cases} \tag{6.19}$$

where the constant c is chosen such that the discrete compatibility condition is satisfied. From the resulting discrete stream function ψ^h, we obtain the velocity using central differencing.

Our numerical results for this example show that the centrifugal force tends to push the markers away from the center, leaving a region in which there are no markers at all. To counteract this tendency, we use the modified method (6.4b), in which the velocity field is projected onto its divergence-free component not only before but also after moving the markers. Again we use only one step of Algorithm 5.1 in each projection.

We perform the following convergence test. We perform the computations with $N=100$, 400, and 1600, each time starting on

$$\{(k_1h_N, k_2h_N) \colon 1\leqslant k_1, k_2\leqslant N^{1/2}\}.$$

Note that the initial marker sets for $N=400$, $N=1600$ contain the initial marker set for $N=100$ in this case. Let $\mathbf{X}_1^{100},..., \mathbf{X}_{100}^{100}$ be the marker positions for $N=100$ at time 0.5. Let $\mathbf{X}_i^{400}$ and $\mathbf{X}_i^{1600}$ be the positions of the *same* markers at time 0.5 computed with $N=400$ and $N=1600$. We measure

$$e_{100,400} := \frac{\sum_{i=1}^{100} |\mathbf{X}_i^{100}-\mathbf{X}_i^{400}|}{100}$$

and

$$e_{400,1600} := \frac{\sum_{i=1}^{100} |\mathbf{X}_i^{400}-\mathbf{X}_i^{1600}|}{100}.$$

For a first order convergent method, one would expect

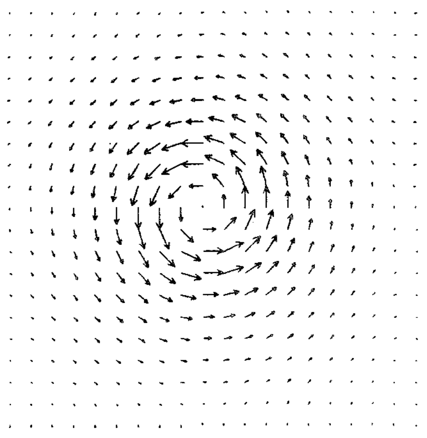
$$\frac{e_{100,400}}{e_{400,1600}}\approx 2.$$

The computed values are shown in Table VIII. We apparently obtain convergence even for very small viscosity coefficients. Note, however, that the size of the error increases substantially for $\nu\to 0$ when N is fixed. Figure 11 shows the velocity fields obtained at $t=0.0$ and $t=0.5$, using $N=6400$. Here the viscosity coefficient is

TABLE VIII

Test Problem 3, $e_{100,400}$ and $e_{400,1600}$, $\Delta t = h_N/2$

ν	$e_{100,400}$	$e_{400,1600}$	$\dfrac{e_{100,400}}{e_{400,1600}}$
0.1	0.0036	0.0019	1.9
0.01	0.022	0.011	2.0
0.001	0.033	0.016	2.1
0.0001	0.036	0.017	2.1



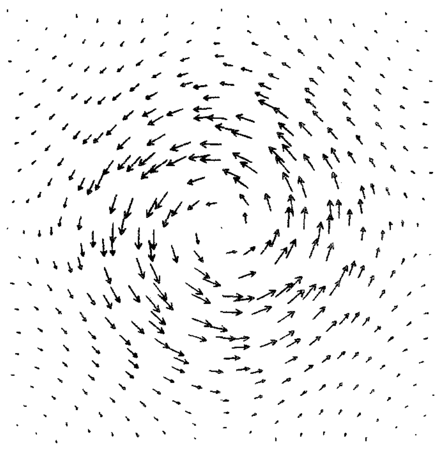
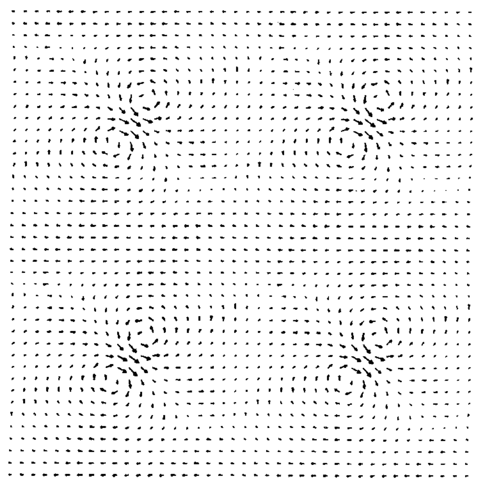


FIG. 11. Test Problem 3, computed velocity fields at $t=0$ and $t=0.5$; $N=6400$ fluid markers, $\nu=0.01$, $\Delta t=h_N/2$; 400 markers are shown.

$\nu = 0.01$. The figure shows the one period which is computed by the method. The positions and velocities of 400 markers are shown. The viscosity causes the flow to slow down. This is not visible in the figure. The vectors are scaled such that the longest vectors in both plots are of equal lengths.



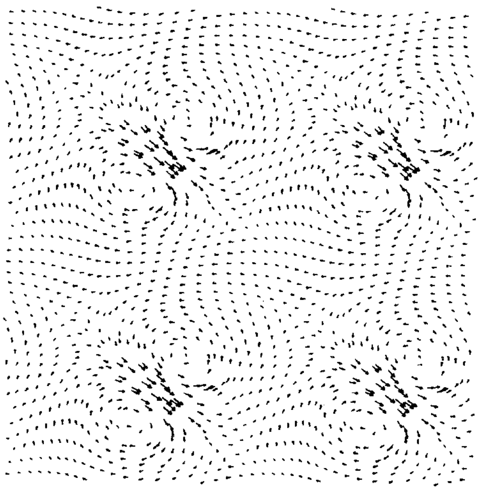


FIG. 12. Test Problem 4, computed velocity fields at $t=0$ and $t=0.5$; $N=6400$ fluid markers, $\nu=0.01$, $\Delta t = h_N/2$; four periods are shown, and in each period, 400 markers are shown.

TEST PROBLEM 4. We compute the motion of a pair of vortex blobs of different signs. At time zero, the blobs are centered at

$$\mathbf{x}^{(1)} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} + \frac{1}{\sqrt{300}} \begin{pmatrix} 1 \\ \sqrt{2} \end{pmatrix}$$

and

$$\mathbf{x}^{(2)} = \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} - \frac{1}{\sqrt{300}} \begin{pmatrix} 1 \\ \sqrt{2} \end{pmatrix}.$$

In addition to these two blobs, the initial vorticity function includes all the periodic images. The vorticity of the blob at $\mathbf{x}^{(i)}$, $i = 1, 2$, is

$$\omega^{(i)}(\mathbf{x}) = \begin{cases} (-1)^{i+1} \cdot 50 \cdot \left(1 + \cos\left(\dfrac{\pi\, |\mathbf{x} - \mathbf{x}^{(i)}|}{0.1}\right)\right) & \text{if } |\mathbf{x} - \mathbf{x}^{(i)}| \leqslant 0.1, \quad (6.20a) \\ 0 & \text{otherwise.} \quad (6.20b) \end{cases}$$

We use $N = 6400$ fluid markers, initially positioned as in Test Problems 1 and 2. The viscosity coefficient is $\nu = 0.01$. The time step and the splitting are as for Test Problem 3. Figure 12 shows positions and normalized velocity vectors for 400 of the 6400 fluid markers at $t = 0.0$ and $t = 0.5$.

We do not know the analytic solution of this problem. Neglecting the viscosity and the periodic images, and replacing the vortex blobs by point vortices with the same total vorticity, one obtains a translational motion in the direction perpendicular to the straight line joining the centers of the two vortices at the constant speed $\frac{5}{4} - (5/\pi^2) \approx 0.74$. The flow in Fig. 12 therefore appears to be qualitatively correct. The computed speed of the vortex blobs is smaller than 0.74 because of the viscosity. A quantitative test, conducted as for Test Problem 3, again indicates first order convergence.

ACKNOWLEDGMENTS

The computations were carried out at the Courant Mathematics and Computing Laboratory of New York University under Contract DE-AC02-76ER03077 with the U.S. Department of Energy, and at the Lawrence Berkeley Laboratory under contract DE-AC03-76SF00098. The authors are indebted to Olof Widlund for his careful reading of the thesis [6] in which much of this work first appeared.

REFERENCES

1. A. AGGARWAL, B. CHAZELLE, L. GUIBAS, C. O'DÚNLAING, AND C. YAP, in *Proc. 26th Annual IEEE Symposium on Foundations of Computer Science, Portland, 1985*, p. 468.
2. J. M. AUGENBAUM, "A New Lagrangian Method for the Shallow Water Equations," Ph. D. thesis, New York University, New York, 1982.

3. J. M. Augenbaum, "A Lagrangian Method for the Shallow Water Equations Based on a Voronoi Mesh-Flows on a Rotating Sphere," *The Free Lagrange Method*, Lecture Notes in Physics, Vol. 238, edited by M. J. Fritts, W. P. Crowley, and H. Trease (Springer-Verlag, Berlin, 1985), p. 54.
4. J. M. Augenbaum and C. S. Peskin, *J. Comput. Phys.* **59**, 177 (1985).
5. R. Bank, "A Multi-Level Iterative Method for Nonlinear Elliptic Equations," *Elliptic Problem Solvers*, edited by M. H. Schultz (Academic Press, New York, 1981), p. 1.
6. C. Börgers, "A Lagrangian Fractional Step Method for the Incompressible Navier–Stokes Equations," Ph. D. thesis, New York University, New York, 1985.
7. C. Börgers and C. S. Peskin, "A Lagrangian Method Based on the Voronoi Diagram for the Incompressible Navier–Stokes Equations on a Periodic Domain," *The Free Lagrange Method*, Lecture Notes in Physics, Vol. 238, edited by M. J. Fritts, W. P. Crowley, and H. Trease (Springer-Verlag, Berlin, 1985), p. 87.
8. A. Bowyer, *Comput. J.* **24**, 162 (1981).
9. A. Brandt, "Guide to Multigrid Development," *Multigrid Methods*, Lecture Notes in Mathematics, Vol. 960, edited by W. Hackbusch and U. Trottenberg (Springer-Verlag, Berlin, 1982), p. 220.
10. A. Brandt and N. Dinar, "Multigrid Solutions to Elliptic Flow Problems," *Numerical Methods for Partial Differential Equations*, edited by S. V. Parer (Academic Press, New York/London, 1979), p. 53.
11. A. J. Chorin, *Math. Comput.* **22**, 745 (1968).
12. P. G. Ciarlet, *The Finite Element Method for Elliptic Problems* (North-Holland, Amseterdam, 1978).
13. W. P. Crowley, in *Proceedings of the Second International Conference on Numerical Methods in Fluid Dynamics, Berkeley, 1970*, edited by M. Holt (Springer-Verlag, Berlin, 1971), p. 37.
14. J. Dukowicz, "Lagrangian Fluid Dynamics Using the Voronoi–Delaunay Mesh," *Numerical Methods for Coupled Problems* (Pineridge Press, Swansea, U.K., 1981).
15. M. J. Fritts, "Two-Dimensional Lagrangian Fluid Dynamics Using Triangular Grids," *Finite-Difference Techniques for Vectorized Fluid Dynamics Calculations*, edited by D. L. Book (Springer-Verlag, Berlin, 1981), p. 98.
16. M. J. Fritts, "Three-Dimensional Algorithms for Grid Restructuring in Free-Lagrangian Calculations," *The Free Lagrange Method*, Lecture Notes in Physics, Vol. 238, edited by M. J. Fritts, W. P. Crowley, and H. Trease (Springer-Verlag, Berlin, 1985), p. 122.
17. M. J. Fritts and J. P. Boris, *J. Comput. Phys.* **31**, 173 (1979).
18. R. Löhner, K. Morgan, J. Peraire, and O. C. Zienkiewicz, "Recent Deevelopments in FEM-CFD," *The Free Lagrange Method*, Lecture Notes in Physics, Vol. 238, edited by M. J. Fritts, W. P. Crowley, and H. Trease (Springer-Verlag, Berlin, 1985), p. 236.
19. R. H. MacNeal, *Quart. Appl. Math.* **11**, 295 (1953).
20. C. S. Peskin, A Lagrangian Method for the Navier–Stokes Equations with Large Deformations, 1979 (unpublished).
21. M. I. Shamos and D. Hoey, in *Proc. 16th Annual IEEE Symposium on Foundations of Computer Science, Berkeley, 1975.*
22. R. Sibson, *Comput. J.* **21**, 243 (1978).
23. K. Stüben and U. Trottenberg, "Multigrid Methods: Fundamental Algorithms, Model Problem Analysis and Applications," *Multigrid Methods*, Lecture Notes in Mathematics, Vol. 960, edited by W. Hackbusch and U. Trottenberg (Springer-Verlag, Berlin, 1982).
24. H. E. Trease, "A Two Dimensional Free Lagrangian Hydrodynamics Model," Ph. D. Thesis, University of Illinois at Urbana-Champaign, 1981.
25. G. Voronoi, *J. Reine Angew. Math.* **134**, 198 (1908).