Subdivision Schemes for Fluid Flow

Henrik Weimer[*] Joe Warren

Rice University

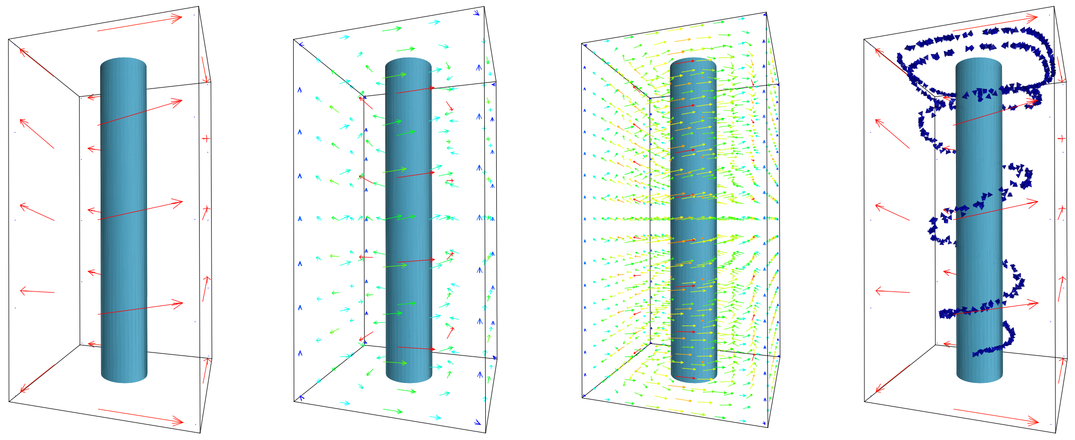

Figure 1: Modeling a helical flow using subdivision.

Abstract

The motion of fluids has been a topic of study for hundreds of years. In its most general setting, fluid flow is governed by a system of non-linear partial differential equations known as the Navier-Stokes equations. However, in several important settings, these equations degenerate into simpler systems of linear partial differential equations. This paper will show that flows corresponding to these linear equations can be modeled using subdivision schemes for vector fields. Given an initial, coarse vector field, these schemes generate an increasingly dense sequence of vector fields. The limit of this sequence is a continuous vector field defining a flow that follows the initial vector field. The beauty of this approach is that realistic flows can now be modeled and manipulated in real time using their associated subdivision schemes.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling— Physically based modeling; I.6.5 [Simulation and Modeling]: Model Development—Modeling Methodologies

Keywords: CAD, subdivision, multi-grid, fluid simulations, fractals, physically based animation, physically based modeling

[*]Rice University, Department of Computer Science, P.O. Box 1892, Houston, TX 77251-1892, {henrik,jwarren}@rice.edu

1 Introduction

This paper describes an application of an increasingly popular modeling technique, subdivision, to a traditional problem in computer graphics, generating realistic fluid flows. Given that the physical behavior of a flow is governed by a system of partial differential equations (PDEs), one standard approach is to use some type of multi-level solver to compute solutions for these PDEs. One of the most efficient multi-level methods currently in use is multi-grid. Multi-grid recursively generates increasingly dense approximations to the exact continuous flow using a nested sequence of domain grids. This method provides exceptionally good convergence rates and has found many successful applications [2].

While multi-grid solvers are very efficient compared to standard iterative solvers, they remain too costly to permit interactive modeling in many situations. One alternate technique that has proven very useful in surface modeling is subdivision. Subdivision schemes have been used to model a variety of surface types [3, 8, 7, 14, 24]. More recent work has applied subdivision to the modeling of surfaces with variational definitions [18, 19, 27, 28]. Due to their simplicity, subdivision schemes are easy to implement and permit interactive modeling of the resulting surfaces.

This paper has two goals, one practical and one theoretical: The practical goal of this paper is to demonstrate that subdivision can be extended beyond the domain of surface modeling to that of modeling flows. Modeling fluid flow through subdivision provides the user with an intuitive modeling metaphor. She defines a coarse initial vector field as a small set of vectors. The subdivision process produces an arbitrarily dense discrete vector field which follows the initial field and satisfies the underlying partial differential equations. Application of the subdivision scheme is conceptually simple, easy to implement and efficient because subdivision is in essence a local weighted averaging process. In particular, implementation of a subdivider based on these schemes is much simpler than implementation of any flow solver with comparable stability. Furthermore, should a denser solution be required than has previously been computed, the subdivision scheme can simply be applied a few extra rounds with no need to re-run the entire solving process.

The theoretical goal of this paper is to expose an intrinsic link between linear systems of PDEs and subdivision. Given a linear system of PDEs, there exists an associated subdivision scheme that produces solutions to these equations. In particular, this scheme is a variant of a more general multi-grid process. The next section of the paper briefly reviews multi-grid and derives the basic relationship between multi-grid and subdivision. The following section then re-derives a known subdivision scheme for polynomial splines as an illustrative example. After introducing the partial differential equations for fluid flow, the paper uses the same methodology to derive subdivision schemes for flow. The resulting subdivision schemes are finally used to construct an interactive modeling system for fluid flow and for real time fluid flow animation.

2 Multi-Resolution Methods

Multi-resolution methods typically compute a sequence of discrete approximations to some continuous limit shape. In this paper, this shape is characterized as the solution to a system of partial differential equations. At an abstract level, the method computes a continuous shape $\mathbf{u}$ that satisfies the equation $\mathbf{D}\ \mathbf{u} = 0$, where $\mathbf{D}$ is a continuous differential operator. Since $\mathbf{u} = 0$ is a trivial solution to this equation, extra *boundary conditions* are added to disallow this trivial solution. Instead, the continuous problem consists of finding the shape $\mathbf{u}$ that satisfies

$$\mathbf{D}\ \mathbf{u} = \mathbf{b} \tag{1}$$

where $\mathbf{b}$ is determined by the boundary conditions.

Because computation with continuous quantities is cumbersome, a common approach is to discretize the continuous shape $\mathbf{u}$ over some type of domain grid T yielding a discrete representation u. [1] Similarly, the continuous differential operator $\mathbf{D}$ is represented by a discrete counterpart D and the boundary conditions $\mathbf{b}$ are replaced by their discrete analog b. Now, the discrete analog of equation 1 is

$$D\,u = b. \tag{2}$$

If the discretization is chosen appropriately, the discrete solution u of equation 2 approximates the continuous solution $\mathbf{u}$ of equation 1.

Equation 2 can be solved using a direct method such as Gauß elimination. However, this approach is impractical for large problems. An alternate approach is to exploit the local structure of the discrete operator D and to solve equation 2 using an iterative method such as Jacobi smoothing. Varga [26] and Young [30] provide background material on several iterative smoothing methods. Unfortunately, these iterative methods tend to converge very slowly for large problems. Multi-grid solvers converge much more rapidly and provide an efficient method for recursively solving large systems of linear equations [2].

In multi-grid, the domain grid T is replaced by a sequence of nested grids $T_0 \subseteq T_1 \subseteq \ldots \subseteq T_n$. Similarly, the discrete entities D, u and b are replaced by corresponding entities D_k, u_k and b_k, defined over the associated domain grid T_k. Consequently, the problem now consists of solving the sequence of linear systems

$$D_k\, u_k = b_k. \tag{3}$$

Again, if the discretizations are chosen appropriately, the limit of these discrete solutions u_k converges to the desired continuous solution $\mathbf{u}$.

[1] Note that we follow the convention of using bold roman notation such as $\mathbf{u}$ for continuous entities and italics such as u for their discrete counterparts.

Multi-grid is a recursive method for computing u_0, u_1, u_2, ... Given the solution u_{k-1}, multi-grid computes u_k using the following three steps:

1. **Prediction:** Compute an initial guess of the solution u_k using a prediction operator S_k,

   $$u_k = S_{k-1} u_{k-1}.$$

   In multi-grid terminology, this step is also referred to as *prolongation*. A common choice for the prediction operator S_k is piecewise linear interpolation.

2. **Smoothing:** Use a traditional iterative method such as Jacobi or Gauß-Seidel smoothing to improve the current solution for u_k.

3. **Coarse grid correction:** Restrict the current residual $D_k\, u_k - b_k$ to the next coarser grid T_{k-1}. There, solve for an error correction term e_{k-1} that is then added back into the solution using $u_k = u_k + S_{k-1} e_{k-1}$.

Note that both steps 2 and 3 serve to improve the accuracy of the solution u_k. If the operator S_{k-1} used in the prediction phase of multi-grid produces an exact initial guess, i.e. $D_k S_{k-1} u_{k-1} = b_k$, then both the smoothing steps and coarse grid correction steps can be omitted. Such an operator S_{k-1} is a *perfect predictor*. A multi-grid scheme based on perfect predictors consists of repeated applications of S_k to produce finer and finer solutions that converge to the exact, continuous solution. Such a multi-grid scheme is exactly a subdivision scheme, where the perfect predictor takes the role of the subdivision operator! The remainder of this section focuses on characterizing perfect predictors in terms of the discrete operators D_k.

To be useful in practice, application of the perfect predictor should not be too costly. Therefore, the perfect predictors should have a local structure. To this end, we suggest a specific choice for the right-hand side b_k of equation 3 based on upsampling. Let R_{k-1} be the discrete *upsampling (replication) operator* that converts data defined over the coarse grid T_{k-1} into data defined over the next finer grid T_k. R_{k-1}'s action is very simple: data associated with grid points in T_{k-1} are replicated; the remaining data associated with grid points in $T_k - T_{k-1}$ are set to zero. Choosing the right-hand side b_k to have the form

$$b_k = R_{k-1} R_{k-2} \ldots R_0 D_0 u_0$$

results in equation 3 having the form

$$D_k u_k = R_{k-1} R_{k-2} \ldots R_0 D_0 u_0. \tag{4}$$

This particular choice of b_k can be interpreted as follows: u_k is determined so that the discrete differences $D_0 u_0$ over the coarsest grid T_0 are replicated onto the fine grids T_k; all the remaining entries in $D_k u_k$ are forced to zero. Intuitively, the limit of this process, as $k \to \infty$, is an object $\mathbf{u}$ that satisfies the continuous partial differential equation $\mathbf{D}\ \mathbf{u} = 0$ everywhere except at the coarsest grid points. At these coarsest grid points $\mathbf{D}\ \mathbf{u}$ exactly reproduces the differences $D_0\ u_0$.

Based on equation 4, a predictor S_{k-1} is perfect if it satisfies the *fundamental multi-grid relation*

$$D_k S_{k-1} = R_{k-1} D_{k-1}. \tag{5}$$

To verify this assertion, let u_{k-1} be a solution to equation 4, i.e. $D_{k-1}\ u_{k-1} = R_{k-2}\ R_{k-3} \ldots R_0\ D_0\ u_0$. Multiplying both sides of this equation by R_{k-1} yields that $R_{k-1}\ D_{k-1}\ u_{k-1} = R_{k-1}\ R_{k-2}\ R_{k-3} \ldots R_0\ D_0\ u_0$. Replacing $R_{k-1} D_{k-1}$ by $D_k S_{k-1}$ (due to equation 5) yields $D_k S_{k-1}\ u_{k-1} = R_{k-1}\ R_{k-2}\ R_{k-3}...R_0\ D_0\ u_0$. Finally, since $u_k = S_{k-1}u_{k-1}$, u_k also satisfies equation 4.

At a high level, our approach to building subdivision schemes for linear systems of PDEs can be outlined as follows: First, we construct discrete analogs of the continuous differential operators involved in the problem. Next, we determine the relationship between the grids of different levels, and define the upsampling and concrete discrete difference operators for these grids. Finally, we solve for the perfect predictor using the fundamental relation 5. The result of this process is a perfect predictor, or subdivision scheme, which converges to solutions of the original, continuous problem.

The next section fills in more of the details of this process. In particular, we derive a subdivision scheme for the simple differential equation that characterizes piecewise polynomial splines. Later in the paper, the same methodology is used to construct subdivision schemes for the PDEs that characterize fluid flow.

3 Subdivision for Polynomial Splines

A polynomial spline $\mathbf{u}[\mathbf{x}]$ of order m is a piecewise polynomial function of degree $m-1$ in a real variable $\mathbf{x}$. For uniform grids (i.e. T is the integer grid, $\mathbb{Z}$), the spline $\mathbf{u}[\mathbf{x}]$ satisfies the differential equation

$$\mathbf{D}[\mathbf{x}]^m \mathbf{u}[\mathbf{x}] = 0 \qquad (6)$$

for all $\mathbf{x} \notin \mathbb{Z}$. Here, the differential operator $\mathbf{D}[\mathbf{x}]$ computes the derivative of $\mathbf{u}[\mathbf{x}]$ with respect to $\mathbf{x}$. $\mathbf{D}[\mathbf{x}]^m$ takes the mth derivative of $\mathbf{u}[\mathbf{x}]$ with respect to $\mathbf{x}$. Our task is to derive the subdivision rules for these polynomial splines using equation 5.

3.1 A multi-grid scheme

Moving to the multi-grid setting, we consider a sequence of domain grids T_k of the form $T_k = \frac{1}{2^k}\mathbb{Z}$. Note that the grids T_k grow dense in the domain $\mathbb{R}$ as $k \to \infty$. Given the continuous spline $\mathbf{u}[\mathbf{x}]$, let u_k be a vector whose ith entry approximates $\mathbf{u}\left[\frac{i}{2^k}\right]$. The vectors u_k are discrete approximations to $\mathbf{u}[\mathbf{x}]$. As $k \to \infty$, these vectors u_k should converge to the continuous function $\mathbf{u}[\mathbf{x}]$.

In the uniform case, generating functions provide a particularly concise and powerful representation for the vectors u_k. Let $u_k[x]$ denote the generating function of the form

$$u_k[x] = \sum_i (u_k)_i x^i$$

where $(u_k)_i$ is the coefficient of the vector u_k with grid point i. For example, the vector $u_0 = \{4, 12, 10, 5, 8\}$ over the integer grid points $\{0, 1, 2, 3, 4\}$ can be represented by the generating function $u_0[x] = 4 + 12x + 10x^2 + 5x^3 + 8x^4$. (Note that the generating functions used in this paper may posses negative and even fractional exponents.)

To discretize the left-hand side of equation 6, we next derive the discrete counterpart of the continuous mth derivative operator $\mathbf{D}[\mathbf{x}]^m$. On the initial grid $T_0 = \mathbb{Z}$, the discrete analog of $\mathbf{D}[\mathbf{x}]$ is the generating function

$$\mathrm{D}[\mathrm{x}] = \frac{1-\mathrm{x}}{\mathrm{x}^{1/2}}.$$

(Note the power $x^{-1/2}$ centers the difference operator D[x] over the origin.) On finer grids $T_k = \frac{1}{2^k}\mathbb{Z}$, this generating function D[x] is scaled by a factor of 2^k. Application of the differential operator $\mathbf{D}[\mathbf{x}]$ to $\mathbf{u}[\mathbf{x}]$ has a simple discrete analog: multiplication of $u[x]$ by D[x]. Repeated application of $\mathbf{D}[\mathbf{x}]$ is equivalent to repeated multiplication by D[x]. Thus, on the grid T_k, the discrete analog of the left-hand side of equation 6 is

$$(2^k \mathrm{D}[\mathrm{x}])^m u_k[x].$$

This expression forms the left-hand side of the fundamental multi-grid relation of equation 4. Our next task is to build a generating function that models the right-hand side of these equations. Recall that the upsampling operator R_k replicated data on T_k and inserted zero on $T_{k+1} - T_k$. If $u_k[x]$ is the generating function associated with the vector u_k, then the expression $R_k u_k$ corresponds to the generating function $u_k[x^2]$. Likewise, the action of the combined operator $R_{k-1}...R_0$ can be modeled by replacing x by x^{2^k}. Thus, equation 4 can be expressed in terms of generating functions as

$$(2^k \mathrm{D}[\mathrm{x}])^m u_k[x] = 2^k \mathrm{D}[\mathrm{x}^{2^k}]^m u_0[x^{2^k}]. \qquad (7)$$

Note that an extra factor of 2^k appears on the right-hand side of equation 7. This factor reflects the effect of the varying grid spacing on R_k. By scaling each upsampling operator R_k by 2^d (where d is the dimension of the underlying domain), the resulting normalized equation defines a sequence of solutions u_k that converges to the desired polynomial spline $\mathbf{u}[\mathbf{x}]$.

3.2 The associated subdivision scheme

Given this multi-grid framework, we can now apply equation 5 and derive a perfect predictor for this scheme. In the uniform setting, the predictor (subdivision matrix) S is a 2-slanted matrix whose columns are all shifts of a single vector. If we let $s[x]$ denote the generating function associated with this vector, then equation 5 has the form:

$$(2^k \mathrm{D}[\mathrm{x}])^m s[x] = 2\left(2^{k-1}\mathrm{D}[\mathrm{x}^2]\right)^m. \qquad (8)$$

The beauty of equation 8 is that we can now directly solve for the subdivision mask $s[x]$. In particular, $s[x]$ is independent of k and has the form:

$$s[x] = 2\left(\frac{\mathrm{D}[\mathrm{x}^2]}{2\mathrm{D}[\mathrm{x}]}\right)^m = \frac{1}{2^{m-1}}\frac{(1+x)^m}{x^{m/2}}. \qquad (9)$$

To verify this fact, recall that $\mathrm{D}[\mathrm{x}] = \frac{1-\mathrm{x}}{\mathrm{x}^{1/2}}$. Therefore, $\mathrm{D}[\mathrm{x}^2] = \frac{1-\mathrm{x}^2}{\mathrm{x}}$ and $\frac{\mathrm{D}[\mathrm{x}^2]}{\mathrm{D}[\mathrm{x}]} = \frac{1+x}{x^{1/2}}$. Substituting this expression into equation 9 yields a mask of the form $s[x] = \frac{1}{2^{m-1}}\frac{(1+x)^m}{x^{m/2}}$. This mask exactly captures the subdivision rule for polynomial splines of order m. In the case of cubic splines, i.e. $m = 4$, equation 9 correctly yields the subdivision mask $s[x] = \frac{(1+x)^4}{8x^2} = \frac{1}{8}\left(x^{-2} + 4x^{-1} + 6 + 4x + x^2\right)$ due to Lane and Riesenfeld [21].

Given the mask $s[x]$, subdivision can be expressed very concisely in terms of generating functions. The coefficient vectors u_{k-1} and u_k are related by

$$u_k[x] = s[x]u_{k-1}[x^2].$$

As $k \to \infty$, the coefficients of $u_k[x]$ form increasingly dense approximations to a continuous limit function $\mathbf{u}[\mathbf{x}]$. In particular, the coefficient of x^i in $u_k[x]$ acts as a discrete approximation to a continuous function value $\mathbf{u}\left[\frac{i}{2^k}\right]$. Figure 2 shows three rounds of subdivision for a cubic spline.

The upper left curve in figure 2 depicts the coefficients $u_0 = \{4, 12, 10, 5, 8\}$ plotted over the integer grid points $\{0, 1, 2, 3, 4\}$. This coefficient sequence can be represented by the generating function $u_0[x] = 4 + 12x + 10x^2 + 5x^3 + 8x^4$. The upper right curve shows the coefficients of $u_1[x] = s[x]u_0[x^2]$,

$$u_1[x] = \frac{1}{8}\left(\frac{1}{x^2} + \frac{4}{x} + 6 + 4x + x^2\right)\left(4 + 12\,x^2 + 10\,x^4 + 5\,x^6 + 8x^8\right),$$

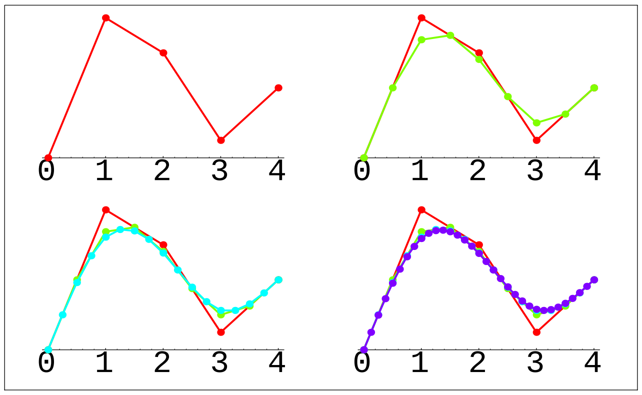

Figure 2: Subdivision for cubic splines.

plotted over the half-integer sequence, $\left\{0, \frac{1}{2}, 1, \frac{3}{2}, 2, \frac{5}{2}, 3, \frac{7}{2}, 4\right\}$. (Remember, the coefficients of x^i in $u_k[x]$ is plotted at $\mathbf{x} = \frac{i}{2^k}$.) The lower figures show the coefficients of $u_2[x] = s[x]u_1[x^2]$ and $u_3[x] = s[x]u_2[x^2]$ plotted over the quarter and one eighth integers, respectively. Observe that this process quickly converges to the desired polynomial spline.

Figure 3 shows plots of the discrete differences $\mathrm{D}[x]^4 u_k[x]$ for $k = 0, 1, 2, 3$. Note that these differences are identically zero except at the original integer grid points, $\{0, 1, 2, 3, 4\}$.

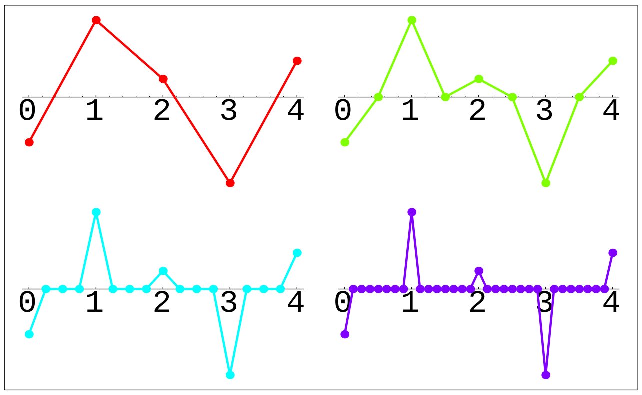

Figure 3: Fourth order discrete differences for cubic splines.

This univariate subdivision scheme can be extended to two-dimensional meshes in a variety of ways. Much of subdivision's popularity as a modeling technique is due to the landmark papers by Doo and Sabin [8] and Catmull and Clark [3]. Hoppe, DeRose and co-workers [14],[7] and Sederberg et al. [24] developed subdivision schemes that can incorporate curvature discontinuities such as creases. Zorin and Schroeder [31] and Kobbelt et al. [20] show that subdivision is well suited for interactive multi-resolution editing of smooth models. Finally, recent work by Kobbelt [18, 19] and the authors [28] investigates the link between subdivision and variational problems.

4 A Brief Introduction to Fluid Mechanics

Fluid mechanics is a field of study that can occupy an entire career. Our goal in this section is to give a brief introduction suitable for the mathematically savvy non-specialist. The source for most of the material in this paper is *Fluid Mechanics* by Liggett [17]. Our approach is to develop most of the theory for flows in two-dimensions. Later in the paper, we extend the theory to three dimensions.

A flow in two dimensions is a vector-valued function $(\mathbf{u}[\mathbf{x},\mathbf{y}], \mathbf{v}[\mathbf{x},\mathbf{y}])^T$ in two parameters $\mathbf{x}$ and $\mathbf{y}$. Flows can be visualized in a number of ways. Perhaps the simplest method is to evaluate $(\mathbf{u}[\mathbf{x},\mathbf{y}], \mathbf{v}[\mathbf{x},\mathbf{y}])^T$ at a grid of parameter values $(\mathbf{x}_i, \mathbf{y}_j)$. Each resulting vector $(\mathbf{u}[\mathbf{x}_i,\mathbf{y}_j], \mathbf{v}[\mathbf{x}_i,\mathbf{y}_j])^T$ is then plotted with its tail placed at $(\mathbf{x}_i, \mathbf{y}_j)$.

The behavior of flows is governed by a set of PDEs known as the *Navier-Stokes equations*. In the most general setting, this set of equations is non-linear. As a result, the behavior of many types of flows is very hard to predict. However, in several important settings, the Navier-Stokes equations reduce to a much simpler set of linear PDEs. In the next sections, we consider two such cases: perfect flow and slow flow. Later, we derive subdivision schemes for slow flow, noting that the exact same methodology can be used to build schemes for perfect flow.

4.1 Perfect flows

Perfect flows are characterized by two properties: *incompressibility* and *zero viscosity*. A flow $(\mathbf{u}[\mathbf{x},\mathbf{y}], \mathbf{v}[\mathbf{x},\mathbf{y}])^T$ is *incompressible* if it satisfies the partial differential equation:

$$\mathbf{u}^{(1,0)}[\mathbf{x},\mathbf{y}] + \mathbf{v}^{(0,1)}[\mathbf{x},\mathbf{y}] = 0.$$

The superscript (i, j) denotes the ith derivative with respect to $\mathbf{x}$ and the jth derivative with respect to $\mathbf{y}$. Flows $(\mathbf{u}[\mathbf{x},\mathbf{y}], \mathbf{v}[\mathbf{x},\mathbf{y}])^T$ satisfying this equation are said to be *divergence-free*. In most flows, viscosity induces rotation in the flow. However, if the fluid has zero viscosity, then its flows are free of rotation. Such flows are often referred to as *irrotational*. Irrotational flows are characterized by the partial differential equation:

$$\mathbf{u}^{(0,1)}[\mathbf{x},\mathbf{y}] = \mathbf{v}^{(1,0)}[\mathbf{x},\mathbf{y}].$$

Together, these two equations in two functions $\mathbf{u}$ and $\mathbf{v}$ uniquely characterize perfect flow:

$$\begin{aligned} \mathbf{u}^{(1,0)}[\mathbf{x},\mathbf{y}] + \mathbf{v}^{(0,1)}[\mathbf{x},\mathbf{y}] = 0, \\ \mathbf{u}^{(0,1)}[\mathbf{x},\mathbf{y}] - \mathbf{v}^{(1,0)}[\mathbf{x},\mathbf{y}] = 0. \end{aligned} \tag{10}$$

To facilitate manipulations involving such systems of PDEs, we use two important tools. As in the univariate case, we express various derivatives using the differential operators $\mathbf{D}[\mathbf{x}]$ and $\mathbf{D}[\mathbf{y}]$. Left multiplication by the differential operator $\mathbf{D}[\mathbf{z}]$ is simply a shorthand for taking a continuous derivative in the $\mathbf{z}$ direction. For example, $\mathbf{D}[\mathbf{x}]\mathbf{u}[\mathbf{x},\mathbf{y}] = \mathbf{u}^{(1,0)}[\mathbf{x},\mathbf{y}]$ and $\mathbf{D}[\mathbf{y}]\mathbf{v}[\mathbf{x},\mathbf{y}] = \mathbf{v}^{(0,1)}[\mathbf{x},\mathbf{y}]$. Our second tool is matrix notation. After replacing the derivatives in equation 10 by their equivalent differential operators, the two linear equations in two unknowns can be written in matrix form as:

$$\begin{pmatrix} \mathbf{D}[\mathbf{x}] & \mathbf{D}[\mathbf{y}] \\ \mathbf{D}[\mathbf{y}] & -\mathbf{D}[\mathbf{x}] \end{pmatrix} \begin{pmatrix} \mathbf{u}[\mathbf{x},\mathbf{y}] \\ \mathbf{v}[\mathbf{x},\mathbf{y}] \end{pmatrix} = 0. \tag{11}$$

In conjunction, these techniques allow generation of subdivision schemes from systems of PDEs using the same strategy as was employed for polynomial splines in the previous section.

4.2 Slow flows

As we saw in the previous section, perfect flows are governed by a simple set of linear PDEs. Another important class of linear flow is *slow flow*. A slow flow is an incompressible flow in which the viscous behavior of the flow dominates any inertial component of the flow. For example, the flow of viscous fluids such as asphalt, sewage sludge or molasses is governed almost entirely by its viscous nature. Other examples of slow flow include the movement of small particles through water or air and the swimming of micro organisms [17].

Slow flow may also be viewed as a minimum energy elastic deformation applied to an incompressible material. This observation is based on the fact that the flow of an extremely viscous fluid is essentially equivalent to an elastic deformation of a solid. One byproduct of this view is that any technique for creating slow flows can also be applied to create minimum energy deformations.

Slow flows in two dimensions are also governed by two partial differential equations. The first partial differential equation corresponds to incompressibility. However, due to their viscosity, slow flows have a rotational component. This rotational component is governed by the partial differential equation:

$$\mathbf{u}^{(2,1)}[\mathbf{x},\mathbf{y}]+\mathbf{u}^{(0,3)}[\mathbf{x},\mathbf{y}]-\mathbf{v}^{(3,0)}[\mathbf{x},\mathbf{y}]-\mathbf{v}^{(1,2)}[\mathbf{x},\mathbf{y}]=0$$

For a complete derivation of this equation, see Liggett [17], p.161. If $\mathbf{L}[\mathbf{x},\mathbf{y}]=\mathbf{D}[\mathbf{x}]^2+\mathbf{D}[\mathbf{y}]^2$ denotes the continuous Laplacian, then these two partial differential equations can now be written together in matrix form as:

$$\begin{pmatrix} \mathbf{D}[\mathbf{x}] & \mathbf{D}[\mathbf{y}] \\ \mathbf{L}[\mathbf{x},\mathbf{y}]\mathbf{D}[\mathbf{y}] & -\mathbf{L}[\mathbf{x},\mathbf{y}]\mathbf{D}[\mathbf{x}] \end{pmatrix} \begin{pmatrix} \mathbf{u}[\mathbf{x},\mathbf{y}] \\ \mathbf{v}[\mathbf{x},\mathbf{y}] \end{pmatrix} = 0. \tag{12}$$

In this form, the similarity between the governing equations for perfect flow and slow flow is striking: The second equation in 12 corresponds to the equation for irrotational flow multiplied by the Laplacian $\mathbf{L}[\mathbf{x},\mathbf{y}]$. The effect of this extra factor of $\mathbf{L}[\mathbf{x},\mathbf{y}]$ on the rotational component of slow flows is easy to explain: Perfect flows exhibit infinite velocities at sources of rotational flow. The extra factor of $\mathbf{L}[\mathbf{x},\mathbf{y}]$ smoothes the velocity field at rotational sources and causes slow flows to be continuous everywhere.

Due to limited space, our main focus in this paper is on developing subdivision schemes for slow flows based on equation 12. Slow flows define smooth vector fields and relate to other interesting problems such as minimum energy deformations. However, we remind the reader that all of the techniques described here are equally applicable to perfect flows.

4.3 Previous work on flows

Traditionally, fluid flow has been modeled either through explicit or numerical solutions of the associated PDEs. Explicit solutions for perfect flow are known for a number of primitives such as sources, sinks and rotors. These can be combined into more complex fields using simple linear combinations (see [17] for more details).

Numerical solutions for the modeling and simulation of flow is a very active research area. Commonly, such solutions involve approximation using either finite difference or finite element schemes. Brezzi and Fortin [1] provide an introduction to some of the standard numerical techniques used for modeling flow. Recently, cellular automata have been applied with some success for the discrete modeling of physical problems, including gas and fluid flow [6, 23].

On the computer graphics side, Kass and Miller [16] simulate surface waves in water by approximately solving a two-dimensional shallow water problem. Chen and Lobo [4] solve the Navier Stokes equations in two dimensions and use the resulting pressure field to define a fluid surface. Chiba et al. [5] simulate water currents using a particle based behavioral model. Miller and Pearce [22] model viscous fluids through a connected particle system. Wejchert and Haumann [29] introduce notions from aerodynamics to the graphics community. Stam and Fiume [25] model turbulent wind fields for animation based on solving the underlying PDEs. Foster and Metaxas [11] suggest solving the Navier-Stokes equations on a coarse grid in three dimensions using a finite difference approach and then interpolating the coarse solution locally as needed. They also extended their method to handle turbulent steam [12].

5 A Subdivision Scheme for Slow Flow

Following the same approach as in the polynomial case, we first build discrete analogs of the continuous PDEs for slow flow and then employ the fundamental multi-grid relation of equation 5 to solve for the subdivision scheme. Given a coarse vector field $(u_0,v_0)^T$, the resulting subdivision scheme defines a sequence of increasingly dense vector fields $(u_k,\ v_k)^T$. These fields converge to a continuous vector field $(\mathbf{u}[\mathbf{x},\mathbf{y}],\mathbf{v}[\mathbf{x},\mathbf{y}])^T$ that follows the original coarse vector field $(u_0,v_0)^T$.

As for polynomial splines, we represent a discrete vector field by a 2-vector of generating functions:

$$\begin{pmatrix} u_k[x,y] \\ v_k[x,y] \end{pmatrix} = \sum\nolimits_{i,j} \begin{pmatrix} u_k \\ v_k \end{pmatrix}_{ij} x^i y^j.$$

Again, the vectors comprising $(u_k,v_k)^T$ act as discrete approximations to the continuous limit field $(\mathbf{u}[\mathbf{x},\mathbf{y}],\mathbf{v}[\mathbf{x},\mathbf{y}])^T$. In particular, the ijth entry of $(u_k,v_k)^T$ acts at $(\mathbf{x},\mathbf{y})=\left(\frac{i}{2^k},\frac{j}{2^k}\right)$. The subdivision scheme $s[x,y]$ is the linear operator that relates $(u_{k-1}[x],v_{k-1}[x])^T$ and $(u_k[x],v_k[x])^T$ according to

$$\begin{pmatrix} u_k[x,y] \\ v_k[x,y] \end{pmatrix} = s[x,y] \begin{pmatrix} u_{k-1}[x^2,y^2] \\ v_{k-1}[x^2,y^2] \end{pmatrix}.$$

Thus, $s[x,y]$ is a 2×2 matrix of generating functions.

In practice, interesting flows depend on boundary conditions which drive the flow. Our approach to boundary conditions is similar to that for polynomial splines. In particular, we enforce the flow PDEs everywhere except on the initial integer grid, $\mathbb{Z}^2$. The coarse, initial vector field encoded by $(u_0[x,y],v_0[x,y])^T$ determines the boundary conditions at $\mathbb{Z}^2$. Denser vector fields are represented by the generating functions $(u_k[x],v_k[x])^T$ and determined so as to satisfy the PDEs for flow.

5.1 A multi-grid scheme

Following the polynomial case, we consider slow flows $(\mathbf{u}[\mathbf{x},\mathbf{y}],\mathbf{v}[\mathbf{x},\mathbf{y}])^T$ that satisfy the differential equations (in matrix form)

$$\begin{pmatrix} \mathbf{D}[\mathbf{x}] & \mathbf{D}[\mathbf{y}] \\ \mathbf{L}[\mathbf{x},\mathbf{y}]\mathbf{D}[\mathbf{y}] & -\mathbf{L}[\mathbf{x},\mathbf{y}]\mathbf{D}[\mathbf{x}] \end{pmatrix} \begin{pmatrix} \mathbf{u}[\mathbf{x},\mathbf{y}] \\ \mathbf{v}[\mathbf{x},\mathbf{y}] \end{pmatrix} = 0$$

for all $(\mathbf{x},\mathbf{y})\notin\mathbb{Z}^2$. On the grid $T_k=\frac{1}{2^k}\mathbb{Z}^2$, the discrete version of this equation (left-hand side only) has the form:

$$\begin{pmatrix} 2^k\mathrm{D}[\mathrm{x}] & 2^k\mathrm{D}[\mathrm{y}] \\ 8^k L[x,y]\mathrm{D}[\mathrm{y}] & -8^k L[x,y]\mathrm{D}[\mathrm{x}] \end{pmatrix} \begin{pmatrix} u_k[x,y] \\ v_k[x,y] \end{pmatrix}$$

where $L[x,y]=\mathrm{D}[\mathrm{x}]^2+\mathrm{D}[\mathrm{y}]^2$ is the discrete version of the Laplacian. The various powers of two are necessary for the discrete differences to converge to the appropriate continuous derivatives as $k\to\infty$. Based on this observation, the fundamental multi-grid equation (equation 4) can be expressed in terms of generating functions as:

$$\begin{pmatrix} 2^k\mathrm{D}[\mathrm{x}] & 2^k\mathrm{D}[\mathrm{y}] \\ 8^k L[x,y]\mathrm{D}[\mathrm{y}] & -8^k L[x,y]\mathrm{D}[\mathrm{x}] \end{pmatrix} \begin{pmatrix} u_k[x,y] \\ v_k[x,y] \end{pmatrix} = \\ 4^k \begin{pmatrix} 0 & 0 \\ L\left[x^{2^k},y^{2^k}\right]\mathrm{D}[\mathrm{y}^{2^k}] & -L\left[x^{2^k},y^{2^k}\right]\mathrm{D}[\mathrm{x}^{2^k}] \end{pmatrix} \begin{pmatrix} u_0\left[x^{2^k},y^{2^k}\right] \\ v_0\left[x^{2^k},y^{2^k}\right] \end{pmatrix}. \tag{13}$$

As before, the left-hand side of equation 13 is a discrete approximation to the PDEs for slow flow on T_k. On the right-hand side, the top row of the difference matrix, corresponding to divergence, is zero. The bottom row of the difference matrix computes a discrete approximation to the rotational component of the flow on $\mathbb{Z}^2$. This rotational component is then upsampled (via $x \to x^{2^k}$ and $y \to y^{2^k}$) to the grid $\frac{1}{2^k}\mathbb{Z}^2$. (The extra factor of 4^k on the right-hand side is due to the scaling of the upsampling matrices.)

As $k \to \infty$, the discrete flows $(u_k[x], v_k[x])^T$ satisfying equation 13 converge to a continuous slow flow $(\mathbf{u}[\mathbf{x},\mathbf{y}], \mathbf{v}[\mathbf{x},\mathbf{y}])^T$ satisfying equation 12. The appendix demonstrates this convergence by deriving an analytic representation for the limit flow directly from equation 13. This analytic representation for $(\mathbf{u}[\mathbf{x},\mathbf{y}], \mathbf{v}[\mathbf{x},\mathbf{y}])^T$ is especially useful in understanding the behavior of the resulting flows. For example, the resulting slow flows are divergence-free everywhere and rotation-free everywhere except at the initial grid points $\mathbb{Z}^2$. The flow is driven by rotational sources that are positioned to drive a flow that follows the initial vector field $(u_0, v_0)^T$. Figure 4 depicts two flows, each generated by a single unit vector in the $\mathbf{x}$ and $\mathbf{y}$ direction, respectively. Note that each flow is driven by a pair of rotational source located on grid points in $\mathbb{Z}^2$. More complex flows are formed by taking linear combinations of translates of these flows on $\mathbb{Z}^2$.

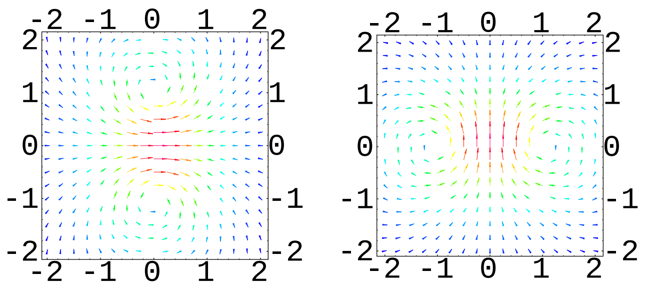

Figure 4: Vector basis functions for slow flow

5.2 The associated subdivision scheme

In analogy with the case of polynomial splines, we next apply our characterization for perfect predictors (equation 5) and derive the subdivision scheme associated with this multi-grid process.

$$\begin{pmatrix} \frac{1}{2}\mathrm{D}[\mathrm{x}] & \frac{1}{2}\mathrm{D}[\mathrm{y}] \\ 2L[x,y]\mathrm{D}[\mathrm{y}] & -2L[x,y]\mathrm{D}[\mathrm{x}] \end{pmatrix} s[x,y] = \begin{pmatrix} 0 & 0 \\ L[x^2,y^2]\mathrm{D}[\mathrm{y}^2] & -L[x^2,y^2]\mathrm{D}[\mathrm{x}^2] \end{pmatrix}. \qquad (14)$$

Solving for $s[x,y]$ in equation 14 yields a two by two matrix of generating functions of the form

$$s[x,y] = \frac{L[x^2,y^2]}{2L[x,y]^2} \begin{pmatrix} \mathrm{D}[\mathrm{y}]\ \mathrm{D}[\mathrm{y}^2] & -\mathrm{D}[\mathrm{x}^2]\ \mathrm{D}[\mathrm{y}] \\ -\mathrm{D}[\mathrm{x}]\ \mathrm{D}[\mathrm{y}^2] & \mathrm{D}[\mathrm{x}]\ \mathrm{D}[\mathrm{x}^2] \end{pmatrix}.$$

The matrix $s[x,y]$ encodes the subdivision scheme for slow flow. Note that this subdivision scheme is a true vector scheme: $u_k[x,y]$ depends on both $u_{k-1}[x,y]$ and $v_{k-1}[x,y]$. Such schemes, while rare, have been the object of some theoretical study. Dyn [10] investigates some of the properties of such vector schemes.

Unfortunately, our analogy with polynomials is not complete. The generating function $L[x,y]^2$ fails to divide into the numerators of $s[x,y]$. In reality, this difficulty is to be expected. Polynomial splines are known to have locally supported bases (B-splines). This observation is consistent with the subdivision scheme $s[x]$ having finitely supported subdivision rules. On the other hand, the vector basis functions in figure 4 do not have local support. Therefore, the generating function $s[x,y]$ can not be expected to divide out and yield finite rules.

Fortunately, all is not lost. The vector basis functions depicted in figure 4 do have two important properties: they are highly localized (i.e. decay rapidly towards zero as $\mathbf{x},\mathbf{y} \to \infty$) and form a partition of unity. As a result, these vector basis functions can be approximated to any desired degree of accuracy via a locally supported vector subdivision scheme. The next section describes an algorithm for finding the subdivision scheme with fixed support that nearly satisfies equation 14.

6 Locally Supported Approximations

Before computing approximations to the matrix mask $s[x,y]$, we make an important adjustment to the subdivision scheme for slow flow. Uniform subdivision schemes for univariate splines come in two types: primal and dual. *Primal* schemes have subdivision masks of odd length that position new coefficients both over old coefficients and between adjacent pairs of old coefficients. *Dual* schemes have masks of even length that position two new coefficients between each adjacent pair of old coefficients. In the bivariate setting, each dimension of a uniform scalar subdivision scheme can be either primal or dual. For vector subdivision schemes, the situation can be even more complex. Each dimension of each component can be primal or dual. In particular, distinct components of a vector scheme need not share the same primal/dual structure.

Indeed, this situation is the case for slow flow. Its vector subdivision scheme has the remarkable property that the $\mathbf{u}$ component is primal in $\mathbf{x}$ and dual in $\mathbf{y}$, and the $\mathbf{v}$ component is dual in $\mathbf{x}$ and primal in $\mathbf{y}$. This observation can be verified by checking the size of the masks in the numerator of $s[x,y]$. In practice, this property makes computing with the given vector subdivision scheme unpleasant. The indexing for schemes with such mixed structure is difficult and prone to errors. In practice, we desire a subdivision scheme for slow flows that is purely primal.

To this end, we adjust equation 14 slightly. In particular, we approximate the differential operators $\mathbf{D}[\mathbf{x}]$ and $\mathbf{D}[\mathbf{y}]$ in equation 12 by $\mathrm{D}[\mathrm{x}^2]$ and $\mathrm{D}[\mathrm{y}^2]$ instead of $\mathrm{D}[\mathrm{x}]$ and $\mathrm{D}[\mathrm{y}]$. The resulting subdivision scheme $s[x,y]$ has the form

$$\begin{pmatrix} \frac{1}{2}\mathrm{D}[\mathrm{x}^2] & \frac{1}{2}\mathrm{D}[\mathrm{y}^2] \\ 2L[x,y]\mathrm{D}[\mathrm{y}^2] & -2L[x,y]\mathrm{D}[\mathrm{x}^2] \end{pmatrix} s[x,y] = \begin{pmatrix} 0 & 0 \\ L[x^2,y^2]\mathrm{D}[\mathrm{y}^4] & -L[x^2,y^2]\mathrm{D}[\mathrm{x}^4] \end{pmatrix}. \qquad (15)$$

Note that the scheme still converges to slow flows. However, the associated subdivision scheme is now primal with a matrix generating function of the form:

$$\frac{1}{2L[x,y]} \begin{pmatrix} \mathrm{D}[\mathrm{y}^2]\ \mathrm{D}[\mathrm{y}^4] & -\mathrm{D}[\mathrm{x}^4]\ \mathrm{D}[\mathrm{y}^2] \\ -\mathrm{D}[\mathrm{x}^2]\ \mathrm{D}[\mathrm{y}^4] & \mathrm{D}[\mathrm{x}^2]\ \mathrm{D}[\mathrm{x}^4] \end{pmatrix}. \qquad (16)$$

Our final goal is to compute a matrix of finitely supported generating functions

$$\begin{pmatrix} s_{11}[x,y] & s_{12}[x,y] \\ s_{21}[x,y] & s_{22}[x,y] \end{pmatrix}$$

that nearly satisfy equation 15. By treating each coefficient of the masks $s_{ij}[x,y]$ as an unknown, the values for these unknowns can be computed as the solution to an optimization problem. In particular, the matrix of approximations $s_{ij}[x,y]$ should nearly satisfy equation 15:

$$\begin{pmatrix} \frac{1}{2}\mathrm{D}[\mathrm{x}^2] & \frac{1}{2}\mathrm{D}[\mathrm{y}^2] \\ 2L[x,y]\mathrm{D}[\mathrm{y}^2] & -2L[x,y]\mathrm{D}[\mathrm{x}^2] \end{pmatrix} \begin{pmatrix} s_{11}[x,y] & s_{12}[x,y] \\ s_{21}[x,y] & s_{22}[x,y] \end{pmatrix} \approx \begin{pmatrix} 0 & 0 \\ L[x^2,y^2]\mathrm{D}[\mathrm{y}^4] & -L[x^2,y^2]\mathrm{D}[\mathrm{x}^4] \end{pmatrix} \qquad (17)$$

The maximum of the matrix difference between the two sides of equation 17 can easily be expressed as a linear program. In practice, the size of this program can be reduced by noting that $s_{11}[x,y] = s_{22}[y,x]$ and $s_{21}[x,y] = s_{12}[y,x]$ due to structure of expression 16. Likewise, due to expression 16, the coefficients of $s_{11}[x,y]$ are symmetric in x and y while the coefficients of $s_{21}[x,y]$ are anti-symmetric in x and y. As a final modification, we constrain the $s_{ij}[x,y]$ to define a vector scheme with constant precision, i.e. $s[x,y]$ applied to a coarse, constant field should produce the same constant on the finer grid. This extra constraint is necessary for the scheme to be convergent (see Dyn [10]) and is consistent with the observation that the vector basis functions for slow flow have constant precision.

Using a standard linear programming code (*CPLEX*, http://www.cplex.com/), we have precomputed approximations $s_{ij}[x,y]$ for matrix masks of size 5×5 to size 15×15. Figure 5 shows a plot of the maximum differences in equation 17 for masks of various sizes. Note the rapid convergence of this residual to zero. This convergence is due to the fact that the basis functions for slow flow decay to zero as $\mathbf{x}, \mathbf{y} \to \infty$.

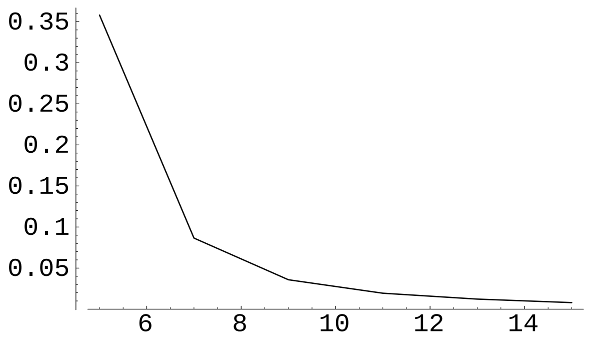

Figure 5: Residuals for finite matrix masks of various sizes.

Given these precomputed masks $s_{ij}[x,y]$, we can apply the vector subdivision scheme to an initial, coarse vector field $(u_0, v_0)^T$ and generate increasingly dense vector fields $(u_k, v_k)^T$ via the matrix relation

$$\begin{pmatrix} u_k[x,y] \\ v_k[x,y] \end{pmatrix} = \begin{pmatrix} s_{11}[x,y] & s_{12}[x,y] \\ s_{12}[y,x] & s_{11}[y,x] \end{pmatrix} \begin{pmatrix} u_{k-1}[x^2,y^2] \\ v_{k-1}[x^2,y^2] \end{pmatrix}.$$

If the vector field $(u_{k-1}, v_{k-1})^T$ is represented as a pair of arrays, then the polynomial multiplications in this expression can be implemented very efficiently using discrete convolution. The resulting implementation allows us to model and manipulate flows in real time. Although matrix masks as small as 5×5 yield nicely behaved vector fields, we recommend the use of 9×9 or larger matrix masks if visually realistic approximations to slow flows are desired. Figure 6 shows a plot of several rounds of subdivision applied to a vector basis function defined in the $\mathbf{x}$-direction (using a matrix mask of size 9×9). Note the similarity to the plot of the exact analytic representation in the left-hand side of figure 4.

The proceedings video tape contains a demonstration of an interactive flow modeler based on these subdivision schemes and discusses some further issues. More material, including the actual local subdivision masks of various sizes, can be found on the conference CDROM as well as on our web page under http://www.cs.rice.edu/CS/Graphics/S99.

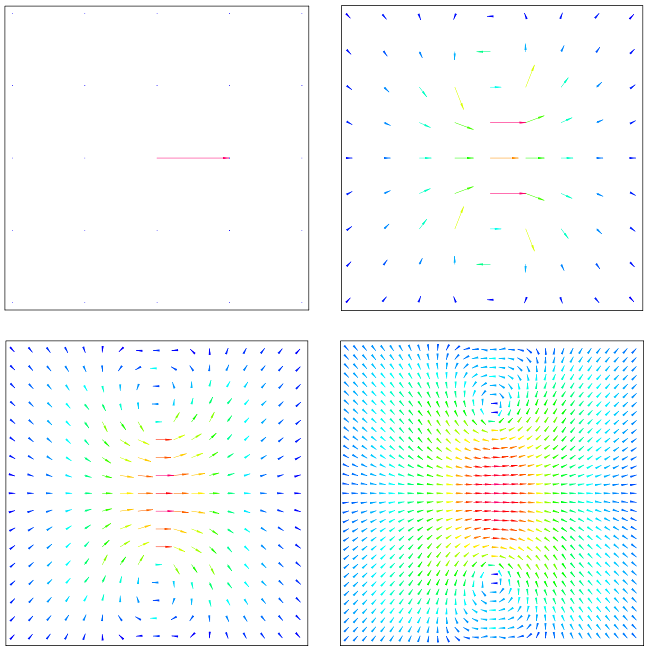

Figure 6: Three rounds of subdivision for a basis vector.

7 Extension to Three Dimensions

Three-dimensional flow is a vector-valued function $(\mathbf{u}[\mathbf{x},\mathbf{y},\mathbf{z}], \mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}], \mathbf{w}[\mathbf{x},\mathbf{y},\mathbf{z}])^T$ in three variables $\mathbf{x},\mathbf{y},\mathbf{z}$. Three-dimensional slow flow is characterized by two sets of partial differential equations modeling the same conditions as in the two-dimensional case. The first partial differential equation forces the flow to be incompressible. The incompressibility condition is given by

$$\mathbf{D}[\mathbf{x}]\mathbf{u}[\mathbf{x},\mathbf{y},\mathbf{z}] + \mathbf{D}[\mathbf{y}]\mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}] + \mathbf{D}[\mathbf{z}]\mathbf{w}[\mathbf{x},\mathbf{y},\mathbf{z}] = 0.$$

The second set of partial differential equations characterizes the rotational behavior of the flow. These conditions are slightly more complicated in three dimensions. For perfect flow these conditions are:

$$\begin{aligned} \mathbf{D}[\mathbf{y}]\mathbf{u}[\mathbf{x},\mathbf{y},\mathbf{z}] &= \mathbf{D}[\mathbf{x}]\mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}], \\ \mathbf{D}[\mathbf{z}]\mathbf{u}[\mathbf{x},\mathbf{y},\mathbf{z}] &= \mathbf{D}[\mathbf{x}]\mathbf{w}[\mathbf{x},\mathbf{y},\mathbf{z}], \\ \mathbf{D}[\mathbf{z}]\mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}] &= \mathbf{D}[\mathbf{y}]\mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}]. \end{aligned} \tag{18}$$

For slow flows, each equation is multiplied by a factor of $\mathbf{L}[\mathbf{x},\mathbf{y},\mathbf{z}]$. Notice that these three continuous equations are linearly dependent. Thus, a continuous solution to these four equations in three unknowns is possible. Unfortunately, the multi-grid approach used in equation 14 fails to yield a subdivision scheme for this set of equations. However, modifying the PDEs for the rotational component of the flow makes subdivision possible. Multiplying each equation of 18 by $\mathbf{D}[\mathbf{z}]$, $\mathbf{D}[\mathbf{y}]$, and $\mathbf{D}[\mathbf{x}]$, respectively yields an new system of partial differential equations:

$$\begin{aligned} \mathbf{D}[\mathbf{z}]\mathbf{D}[\mathbf{y}]\mathbf{u}[\mathbf{x},\mathbf{y},\mathbf{z}] &= \mathbf{D}[\mathbf{x}]\mathbf{D}[\mathbf{z}]\mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}], \\ \mathbf{D}[\mathbf{y}]\mathbf{D}[\mathbf{z}]\mathbf{u}[\mathbf{x},\mathbf{y},\mathbf{z}] &= \mathbf{D}[\mathbf{x}]\mathbf{D}[\mathbf{y}]\mathbf{w}[\mathbf{x},\mathbf{y},\mathbf{z}], \\ \mathbf{D}[\mathbf{x}]\mathbf{D}[\mathbf{z}]\mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}] &= \mathbf{D}[\mathbf{x}]\mathbf{D}[\mathbf{y}]\mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}]. \end{aligned}$$

Note that any solution to these equations is also a solution to equations 18. Given these new equations, we can apply the same

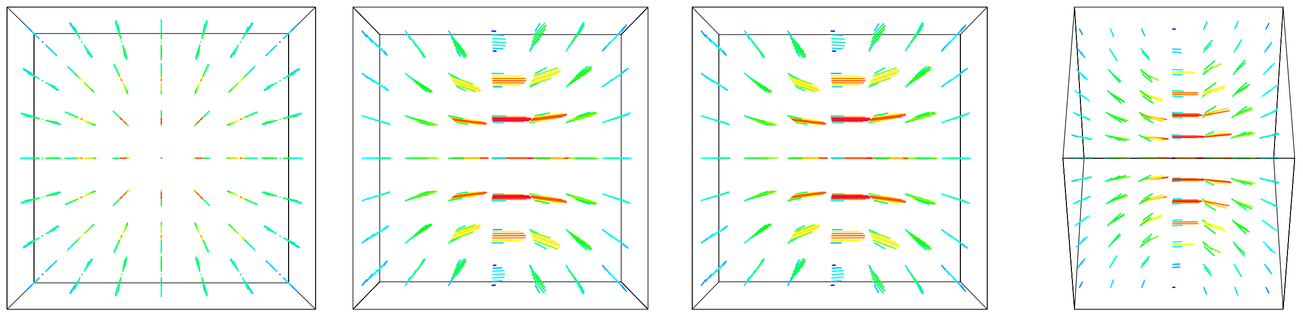

Figure 7: Four views of a 3D vector basis function for slow flow.

derivation used in the two-dimensional case. First, we set up a discrete version of this continuous system of PDEs. Next, we derive the associated subdivision scheme and its analytic bases from the fundamental equation 5. Finally, we compute approximations to the matrix subdivision mask for the scheme using linear programming. Again, the actual subdivision masks can be found on the conference CDROM and on our web page under http://www.cs.rice.edu/CS/Graphics/S99.

The resulting subdivision scheme shares all of the nice properties of the two-dimensional scheme. Given a coarse, initial vector field $(u_0, v_0, w_0)^T$, the limit of the scheme is a continuous vector field $(\mathbf{u}[\mathbf{x},\mathbf{y},\mathbf{z}], \mathbf{v}[\mathbf{x},\mathbf{y},\mathbf{z}], \mathbf{w}[\mathbf{x},\mathbf{y},\mathbf{z}])^T$ that follows $(u_0, v_0, w_0)^T$. Figure 7 shows four plots of a vector basis function in the **x**-direction. The respective plots show views of the vector basis along the **x**-axis, the **y**-axis, the **z**-axis and the vector $(0,1,1)^T$. The basis defines a localized slow flow driven by a toroidal flow around a unit square orthogonal to the **x**-axis.

We conclude with a snapshot from the video proceedings. Figure 8 depicts a stream of particles flowing down a drain. This flow was computed by applying several rounds of subdivision to an initial $3 \times 3 \times 3$ grid of vectors.

8 Conclusion and Future Work

This paper developed vector subdivision schemes for slow flows. These subdivision schemes were special instances of more general multi-grid schemes for solving the PDEs that characterize slow flow. Just as subdivision has revolutionized the process of polyhedral modeling, we feel that these schemes have the potential to revolutionize the process of modeling flow. We have several strong reasons for this belief. First and foremost, the speed of these subdivision schemes allows examination and manipulation of interesting flow fields at interactive rates. Next, the simplicity of these subdivision schemes allow graphics enthusiasts who are not specialist in flow to experiment with creating realistic flows. For example, many of the stability problems that arise in conventional solvers are not an issue here. Subdivision also provides a modeling paradigm even if other simple methods such as potential fields fail (i.e. flows in three dimensions). Finally, subdivision can also be incorporated as an add-on to an existing flow package with the subdivision rules used to refine an existing solution and provide more local detail. Because the subdivision scheme models solutions to the Navier-Stokes PDEs, this approach is far superior to polynomial interpolation, which is usually used to locally refine the solutions of a flow package.

On the other hand, there are limitations to the modeling of fluid flow with the methodology developed here. First, subdivision is an intrinsically linear process. Therefore, subdivision can only be applied to modeling solutions to linear systems of PDEs. In the case of flow, this restriction forces the focus onto modeling linear flows such as slow flows and perfect flows. For now, modeling non-linear flows seems beyond the reach of this method. Second, the system of linear PDEs used here describes steady, time independent flows. Thus, the resulting subdivision scheme models steady, time independent flows. However, there is no fundamental limitation to applying the methodology to derive subdivision schemes for time dependent problems as long as the associated PDEs remain linear. Finally, the derivations in this paper are based on regular grids, modeled through generating functions. Hence, the subdivision schemes only model flows on regular grids. However, the same methodology using linear algebra in place of generating functions can be employed to derive subdivision schemes for irregular grids. The main obstacle to this approach is the enumeration and computation of subdivision rules for each distinct geometric configuration in the underlying grid.

Our future plans focus on two problems. The first problem is to use these schemes as the building block for an interactive system for designing flows. Ideally, the user would sketch approximate streamlines and obstacles for the flow. The modeler would then use subdivision to create a physically accurate flow that approximates the sketch. Our second task is to consider an extension of this work to time varying flows. One simple approach would be to modify the initial, coarsest vector field smoothly over time. Since the underlying basis functions are highly localized and smooth, the resulting limit flow should change smoothly. A more physically realistic approach for time-dependent flows would be to add an extra dimension corresponding to time to the discrete vector fields. Time slices of the initial vector field $(u_0, v_0)^T$ would be viewed as key flow configurations (analogous to key frames). The vector subdivision scheme would then define a sequence of increasingly dense (in both time and space) vector fields $(u_k, v_k)^T$ that converge to the true time-dependent flow.

Acknowledgments

We would like to thank the anonymous reviewers of the paper for the many constructive suggestions. This work was supported in part under National Science Foundation grants CCR-9500572 and CCR-9732344.

References

[1] F. Brezzi, M. Fortin: *Mixed and Hybrid Finite Element Methods*, Springer, 1991.

[2] W. L. Briggs: *A Multigrid Tutorial.* Society for Industrial and Applied Mathematics, 1987.

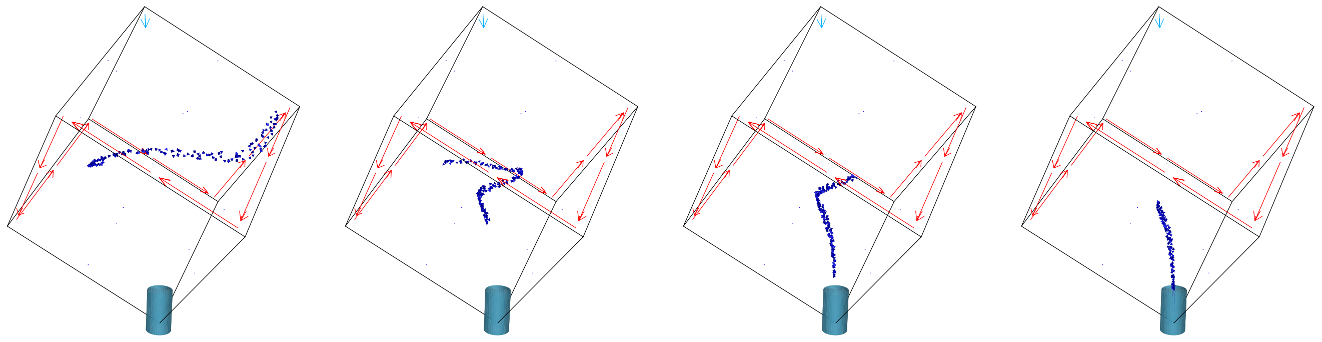

Figure 8: Particles flowing down a drain.

[3] E. Catmull, J. Clark: *Recursive generated B-spline surfaces on arbitrary topological meshes.* Computer Aided Design 10(6), pp. 350-355, 1978.

[4] J. Chen, N. Lobo: *Toward interactive-rate simulation of fluid with moving obstacles using Navier-Stokes equations.* Graphical Models and Image Processing, 57(2), pp. 107-116, 1995.

[5] N. Chiba et al.: *Visual simulation of water currents using a particle-based behavioral model.* Journal of Visualization and Computer Animation 6, pp. 155-171, 1995.

[6] B. Chopard, M. Droz: *Cellular Automata Modeling of Physical Systems*. Cambridge University Press, 1998.

[7] T. DeRose, M. Kass, T. Truong: *Subdivision surfaces in character animation*. Proceedings of SIGGRAPH 98. In *Computer Graphics* Proceedings, Annual Conference Series, pp. 85-94, 1998.

[8] D. Doo, M. Sabin: *Behavior of recursive division surfaces near extraordinary points.* Computer Aided Design 10(6), pp. 356-360, 1978.

[9] N. Dyn: *Interpolation of scattered data by radial functions*. In C. K. Chui et al. (eds.): *Topics in Multivariate Approximation*, Academic Press, pp. 47-61, 1987.

[10] N. Dyn: *Subdivision schemes in computer aided geometric design*. In W. Light (ed.): *Advances in Numerical Analysis II*, Oxford University Press, , pp. 36-104, 1992.

[11] N. Foster, D. Metaxas: *Realistic animation of liquids.* Graphical Models and Image Processing, 58 (5), pp. 471-483, 1996.

[12] N. Foster, D. Metaxas: *Modeling the motion of a hot, turbulent gas*. Proceedings of SIGGRAPH 97. In *Computer Graphics* Proceedings, Annual Conference Series, pp. 181-188, 1997.

[13] J. Hosckek, D. Lasser: *Fundamentals of Computer Aided Geometric Design*. A K Peters, 1993.

[14] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, W. Stuetzle: *Piecewise smooth surface reconstruction*. Proceedings of SIGGRAPH 94. In *Computer Graphics* Proceedings, Annual Conference Series, pp. 295-302, 1994.

[15] Z. Kadi, A. Rockwood: *Conformal maps defined about polynomial curves.* Computer Aided Geometric Design 15, pp. 323-337, 1998.

[16] M. Kass, G. Miller: *Rapid, stable fluid dynamics for computer graphics.* Proceedings of SIGGRAPH 89. In Computer Graphics Proceedings, Annual Conference Series, pp. 49-57, 1989.

[17] J. Liggett: *Fluid Mechanics.* McGraw Hill, 1994.

[18] L. Kobbelt: *A variational approach to subdivision*. Computer Aided Geometric Design 13, pp. 743-761, 1996.

[19] L. Kobbelt: *Variational design with parametric meshes of arbitrary topology.* To appear, http://www9.informatik.uni-erlangen.de/Persons/Kobbelt/papers/design.ps.gz, 1998.

[20] L. Kobbelt, S. Campagna, J. Vorsatz, H.-P. Seidel: *Interactive multi-resolution modeling on arbitrary meshes.* Proceedings of SIGGRAPH 98. In *Computer Graphics* Proceedings, Annual Conference Series, pp. 105-114, 1998.

[21] J. Lane, R. Riesenfeld. *A theoretical development for the computer generation and display of piecewise polynomial surfaces*. IEEE Transactions on Pattern Analysis and Machine Intelligence 2, 1, pp. 35-46, 1980.

[22] G. Miller, A. Pearce: *Globular dynamics: A connected particle system for animating viscous fluids.* Computers and Graphics 13 (3), pp. 305-309, 1989.

[23] D. H. Rothman, S. Zaleski: *Lattice-Gas Cellular Automata*. Cambridge University Press, 1997.

[24] T. W. Sederberg, J. Zheng, D. Sewell, M. Sabin: *Non-uniform recursive subdivision surfaces.* Proceedings of SIGGRAPH 98. In *Computer Graphics* Proceedings, Annual Conference Series, pp. 387-394, 1998.

[25] J. Stam, E. Fiume: *Turbulent Wind Fields for Gaseous Phenomena*. Proceedings of SIGGRAPH 93. In *Computer Graphics* Proceedings, Annual Conference Series, pp. 369-376, 1993.

[26] R. S. Varga: *Matrix Iterative Analysis.* Prentice-Hall, 1962.

[27] J. Warren, H. Weimer: *A cookbook for variational subdivision*. in D. Zorin (ed.): *Subdivision for Modeling and Animation*, SIGGRAPH 99 course-notes number 37, 1999.

[28] H. Weimer, J. Warren: *Subdivision schemes for thin plate splines*. Computer Graphics Forum 17, 3, pp. 303-313 & 392, 1998.

[29] J. Wejchert, D. Haumann: *Animation Aerodynamics*. Proceedings of SIGGRAPH 91. Computer Graphics 25 (3), pp. 19-22, 1991.

[30] D. M. Young: *Iterative Solution of Large Linear Systems*. Academic Press, 1971.

[31] D. Zorin, P. Schröder, W. Sweldens: *Interactive multiresolution mesh editing.* Proceedings of SIGGRAPH 97. In *Computer Graphics* Proceedings, Annual Conference Series, pp. 259-168, 1997.

Appendix: Analytic Bases for Slow Flow

This appendix derives an analytic representation for the limiting flows of equation 13. Due to the linearity of the process, the limiting vector field defined by the multi-grid scheme can be written as a linear combination of translates of two vector basis functions $(\phi_{\mathbf{x}}[\mathbf{x},\mathbf{y}], \phi_{\mathbf{y}}[\mathbf{x},\mathbf{y}])$ (shown in figure 4) times the vectors of the initial field $(u_0, v_0)^T$. Specifically, the limiting field $(\mathbf{u}[\mathbf{x},\mathbf{y}], \mathbf{v}[\mathbf{x},\mathbf{y}])^T$ has the form:

$$\begin{pmatrix} \mathbf{u}[\mathbf{x},\mathbf{y}] \\ \mathbf{v}[\mathbf{x},\mathbf{y}] \end{pmatrix} = \sum_{i,j=-\infty}^{\infty} \left(\phi_{\mathbf{x}}[\mathbf{x}-i, \mathbf{y}-j] \ , \ \phi_{\mathbf{y}}[\mathbf{x}-i, \mathbf{y}-j] \right) \begin{pmatrix} u_0 \\ v_0 \end{pmatrix}_{ij}$$

Note again that both $\phi_{\mathbf{x}}[\mathbf{x},\mathbf{y}]$ and $\phi_y[\mathbf{x},\mathbf{y}]$ are vector-valued functions. Our goal is to find a simple, closed-form expression for these functions.

A rotational generator of slow flow

Recall that the continuous field $(\mathbf{u}[\mathbf{x},\mathbf{y}], \mathbf{v}[\mathbf{x},\mathbf{y}])^T$ is the limit of the discrete fields$(u_k, v_k)^T$ as $k \to \infty$. By solving equation 13 we can express $(u_k, v_k)^T$ directly in terms of $(u_0, v_0)^T$,

$$\begin{pmatrix} u_k[x,y] \\ v_k[x,y] \end{pmatrix} = \frac{L\left[x^{2^k}, y^{2^k}\right]}{2^k L[x,y]^2} \begin{pmatrix} \mathrm{D}[y] \\ -\mathrm{D}[x] \end{pmatrix} \left(\mathrm{D}[y^{2^k}], \ -\mathrm{D}[x^{2^k}] \right) \begin{pmatrix} u_0[x,y] \\ v_0[x,y] \end{pmatrix}.$$

This matrix of generating functions relating $(u_0, v_0)^T$ and $(u_k, v_k)^T$ can be re-written in the form:

$$\frac{1}{4^k L[x,y]^2} L\left[x^{2^k}, y^{2^k}\right] \begin{pmatrix} 2^k \mathrm{D}[y] \\ -2^k \mathrm{D}[x] \end{pmatrix} \left(\mathrm{D}[y^{2^k}], \ -\mathrm{D}[x^{2^k}] \right). \qquad (19)$$

Our next task is to find continuous analogs of various parts of this matrix expression as $k \to \infty$. We first analyze the most difficult part of the expression, $\frac{1}{4^k L[x,y]^2}$. Consider a sequence of generating functions $r_k[x,y]$ such that $4^k L[x,y]^2 r_k[x,y] = 1$ for all k. The coefficients of the $r_k[x,y]$ are discrete approximations to a continuous function $\mathbf{r}[\mathbf{x},\mathbf{y}]$ satisfying the differential equation,

$$\mathbf{L}[\mathbf{x},\mathbf{y}]^2 \mathbf{r}[\mathbf{x},\mathbf{y}] = 0$$

everywhere except at $(\mathbf{x},\mathbf{y}) = 0$. Fortunately, a closed form solution to this partial differential equation is known. The function $\mathbf{r}[\mathbf{x},\mathbf{y}]$ is a radial basis function of the form:

$$\mathbf{r}[\mathbf{x},\mathbf{y}] = \frac{1}{16\pi}(\mathbf{x}^2 + \mathbf{y}^2)\log[\mathbf{x}^2 + \mathbf{y}^2].$$

Hoschek and Lasser [13] and Dyn [9] give a brief introduction to radial basis functions and discuss some of their properties. The factor of 4^k normalizes the coefficients of the sequence $r_k[x,y]$ so that their limit $\mathbf{r}[\mathbf{x},\mathbf{y}]$ satisfies $\int \mathbf{L}[\mathbf{x},\mathbf{y}]^2 \mathbf{r}[\mathbf{x},\mathbf{y}] = 1$.

The other components of expression 19 are easier to interpret. Differences of the form $2^k \mathrm{D}[z]$ taken on $\frac{1}{2^k}\mathbb{Z}^2$ converge to the continuous derivative $\mathbf{D}[\mathbf{z}]$. Differences of the form $\mathrm{D}[z^{2^k}]$ taken on $\frac{1}{2^k}\mathbb{Z}^2$ correspond to unit differences taken on $\mathbb{Z}^2$. Based on these two observations, we can convert successively larger portions of expression 19.

Let $g_k[x,y]$ denote $r_k[x,y] L\left[x^{2^k}, y^{2^k}\right] \left(2^k \mathrm{D}[y] \ , \ -2^k \mathrm{D}[x] \right)^T$. The continuous analog of $L\left[x^{2^k}, y^{2^k}\right]$ is a sequence of differences on $\mathbb{Z}^2$ corresponding to the discrete difference mask $L[x,y]$. The continuous analog of the vector of discrete operators $\left(2^k \mathrm{D}[y], \ -2^k \mathrm{D}[x] \right)^T$ is the vector of differential operators $\left(\mathbf{D}[\mathbf{y}], \ -\mathbf{D}[\mathbf{x}] \right)^T$. Based on these observations, the corresponding vector function $\mathbf{g}[\mathbf{x},\mathbf{y}]$ has the form:

$$\begin{aligned} \mathbf{g}[\mathbf{x},\mathbf{y}] \ &= \ \begin{pmatrix} \mathbf{d}[\mathbf{y}] \\ -\mathbf{d}[\mathbf{x}] \end{pmatrix} \Big(-4\mathbf{r}[\mathbf{x},\mathbf{y}] + \mathbf{r}[\mathbf{x}+1,\mathbf{y}] + \\ &\quad \mathbf{r}[\mathbf{x}-1,\mathbf{y}] + \mathbf{r}[\mathbf{x},\mathbf{y}+1] + \mathbf{r}[\mathbf{x},\mathbf{y}-1] \Big). \end{aligned}$$

The behavior of $\mathbf{g}[\mathbf{x},\mathbf{y}]$ gives us our first insight into the structure of slow flow. In fact, $\mathbf{g}[\mathbf{x},\mathbf{y}]$ is a generator for a localized, rotational slow flow. We can verify that $\mathbf{g}[\mathbf{x},\mathbf{y}]$ is truly a slow flow by substituting the definition of $\mathbf{g}[\mathbf{x},\mathbf{y}]$ into equation 12. Likewise, we can verify that this flow is highly localized (i.e. decays to zero very fast away from the origin) based on the analytic representation. Figure 9 shows a plot of $\mathbf{g}[\mathbf{x},\mathbf{y}]$ on the domain $[-2,2]^2$.

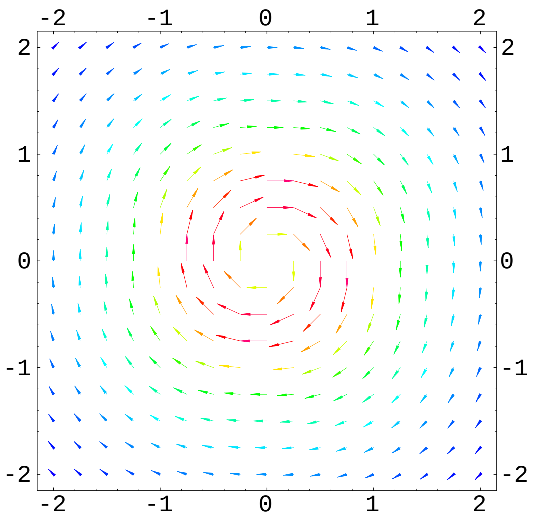

Figure 9: A rotational generator for slow flow

Vector basis functions for slow flow

Finally, expression 19 can be written as $g_k[x,y] \left(\mathrm{D}[y^{2^k}] \ , \ -\mathrm{D}[x^{2^k}] \right)$. The coefficients of this sequence converge to differences of the rotational generator $\mathbf{g}[\mathbf{x},\mathbf{y}]$ taken in the $\mathbf{y}$-direction and $\mathbf{x}$-direction, respectively. The resulting functions are the vector basis functions for our subdivision scheme:

$$\begin{aligned} \phi_{\mathbf{x}}[\mathbf{x},\mathbf{y}] \ &= \ \mathbf{g}\left[\mathbf{x}, \mathbf{y}+\tfrac{1}{2}\right] - \mathbf{g}\left[\mathbf{x}, \mathbf{y}-\tfrac{1}{2}\right], \\ \phi_{\mathbf{y}}[\mathbf{x},\mathbf{y}] \ &= \ -\mathbf{g}\left[\mathbf{x}+\tfrac{1}{2}, \mathbf{y}\right] + \mathbf{g}\left[\mathbf{x}-\tfrac{1}{2}, \mathbf{y}\right]. \end{aligned}$$

Figure 4 contains plots of these basis functions in the $\mathbf{x}$ and $\mathbf{y}$ directions, respectively. Note that the vector basis consists of a pair of rotational sources positioned so as to drive a flow along the appropriate axis. Again, this flow is localized in the sense that it decays rapidly to zero away from the origin. More complex flows can be constructed by taking linear combinations of these vector basis functions. Due to the normalizing constant of $\frac{1}{16\pi}$ used in defining the original radial basis function $\mathbf{r}[\mathbf{x},\mathbf{y}]$, the scheme has constant precision. In particular, choosing the coarse vector field $(u_0, v_0)^T$ to be translates of a constant vector defines a constant flow of the same magnitude in the same direction.