Journal of Computational Physics **159,** 246–273 (2000)
doi:10.1006/jcph.2000.6435, available online at http://www.idealibrary.com on IDEAL®

An Iterative Grid Redistribution Method for Singular Problems in Multiple Dimensions

Weiqing Ren and Xiao-Ping Wang

Department of Mathematics, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong
E-mail: mawang@ust.hk

Received September 1, 1999; revised December 13, 1999

We introduce an iterative grid redistribution method based on the variational approach. The iterative procedure enables us to gain more precise control of the grid distribution near the regions of large solution variations. The method is particularly effective for solving partial differential equations with singular solutions (e.g., blowup solutions). Our method requires little prior information of the singular solutions and can handle multiple singularities. The method is successfully applied to the nonlinear Schrödinger equation and the Keller–Segal equations where solutions with multiple blowup points can be solved up to times very close to the blowup time.

1. INTRODUCTION

Numerical solution of many problems from areas such as fluid dynamics, combustion, and heat transfer requires small node separations over a portion of the physical domain to resolve large solution variations. Using uniform meshes for these problems is formidable when the system involves two or more spatial dimensions. It is now widely recognized that an adaptive computation mesh increases the accuracy and decreases the cost of numerical calculations. Adaptivity is achieved in various ways by using, for example, local adaptive mesh refinement [2], moving finite elements mesh refinement [8, 13], adaptive node movement (see, e.g., [3, 9, and references therein]), or methods based on attraction and repulsion pseudoforces between nodes [14].

In this paper, we are mainly interested in adaptive mesh redistribution methods. When such an adaptive method is used, the mesh point locations change while the solution of the physical problem is computed. Both the grid point locations and the original physical unknowns evolve as part of the problem solution. The grid point movement is usually achieved in two different ways, static and dynamic. In a static method, mesh points are redistributed at fixed time levels according to a mesh generating rule. In a dynamic or moving

0021-9991/00 $35.00

mesh method, the mesh points move continuously in the space–time domain according to a moving mesh equation. In both cases, the grid points are moved during the calculation in order to improve the quality of the result. This is usually assumed to mean that some measure of the total error in the solution has been reduced or physical resolution has been improved in regions where large changes in dependent variables occur.

The key ingredient of such adaptive methods is the grid generation rule. In grid generation, one seeks a change of independent variables (or a mapping from the physical domain to the computational domain), such that, in the new variables, the variation of the concerned quantity (defined by the monitor function) is reduced. In the discrete level, this means that a uniform grid in the computational domain is mapped into the physical domain so that more grid points are concentrated in the regions of large variations. The question that we are concerned with in this paper is whether it is possible (and how) to determine an optimal coordinate mapping in certain sense, so that we can achieve the best possible behavior in the computation variables. As we will see later, this question is particularly important if the problem that we are dealing with is singular.

In two (or higher) spatial dimensions, the commonly used mesh generation techniques are based on a variational approach, many of which are derivations of a technique first proposed by Winslow. The functional is chosen so that the minimum is suitably influenced by the solution of the partial differential equation (PDE). Specifically, given a solution $u(x)$ of the PDE at a fixed time, a mapping from the physical domain Ω_p to the computational domain Ω_c, which usually is taken to be the same as Ω_p,

$$\xi(\mathbf{x}) : \Omega_p \to \Omega_c, \tag{1.1}$$

is determined by minimizing a functional of the form

$$E(\xi) = \int_\Omega \frac{1}{w} |\nabla \xi|^2 \, d\mathbf{x}, \tag{1.2}$$

where $w(\mathbf{x})$ is a weight function (or monitor function) depending on the physical solution $u(\mathbf{x})$ to be adapted. The monitor function should be chosen so that the function in the computational domain given by

$$v(\xi) = u(x(\xi)) \tag{1.3}$$

is better behaved. With (1.2), the coordinate transform (1.1) is determined from the Euler–Lagrange equation

$$\nabla \cdot \left(\frac{1}{w} \nabla \xi \right) = 0. \tag{1.4}$$

In one dimension, (1.4) is simplified to

$$x_\xi w = C, \tag{1.5}$$

which is the differential form of the equidistribution rule used in many one-dimensional adaptive methods [9].

Understanding how the monitor function influences the resulting mesh properties is crucial for the success of a mesh adaption method. In one dimension, such a relation is

given by the equidistribution rule, but only in an average sense. In multiple dimensions, it is even more difficult to predict the overall resulting mesh behavior from the monitor function itself. In other words, it is difficult to have a precise control of the resulting grid distribution from (1.4) or (1.5). Our numerical experiments using the above methods in various forms have shown that adaptivity is achieved only when the solution is moderate. However, grid points stop moving toward (or even turn away from) the singular region when singularity is approached; i.e., adaptivity is lost when it is most needed. Part of the reason is because the solution is so concentrated in a small region that it has little effect on the grids far away from the singularity. An explicit reason for the failure is given in Section 3 for the one-dimensional problem. In a particular application, such a problem may be fixed partially by choosing a carefully designed monitor function when the structure of the singularity is available. But a method that works in general is desirable. Such a method must be based on a better grid redistribution procedure than those given above.

In Section 4, we introduce an iterative grid redistribution method. The fundamental idea is that one realizes that a successive application of the mapping (the Winslow mapping) obtained from (1.1)–(1.4) (or the generalization of them) improves the adaptivity of the grids. In fact, our results indicate that the iteration converges to an optimal coordinate mapping in the sense that the monitor function converges to a constant function almost everywhere. With the iteration procedure, we gain control of the mesh distribution near the singular point; i.e., we may achieve desirable grid adaption by controlling the number of mesh iterations. In Section 5, we show how our iterative grid redistribution procedure can be easily implemented in a static adaptive grid redistribution procedure for PDEs.

In Section 6, we show several numerical examples. We mainly apply our method to the nonlinear Schrödinger equation [11, 12]. The dynamic rescaling method has been a very successful method for solving the blowup solutions of the nonlinear Schrödinger equation (NLS). Not only it can integrate the equation very close to blowup time, the method also provides the blowup profile as well as the rate of blowup. However, the method is restricted to the case in which the solution blows up at only a single point and it also cannot resolve the solution outside the self-similarity region. In our numerical examples, we will demonstrate the capability of our method for tracking multiple singularities. First, we solve the nonlinear Schrödinger equation with a single blowup point and the results are shown to match the results obtained by the dynamic rescaling method. More importantly, in the second example we solve a solution that blows up at two points. To the best of our knowledge, such calculations have never been done before, at least not with such high resolutions. We also show a calculation in a supercritical case where the boundary of the region of self-similarity can be determined by our method. This shows that not only our method can resolve the singular regions, it also takes care of the the regions away from the singularities. Finally, we will show a numerical solution for the Keller–Segal model for bacterial pattern formation where multiple singularities arise from an initial uniform bacterial density distribution.

Note that various improvements of Winslow's method have been considered by many authors [1, 3, 4, 10, 14, 17]. For example, Brackbill [4] incorporates an efficient directional control as well as orthogonality of the mesh into the mesh adaption by adding to (1.2) more functionals measuring those effects and thereby improves both accuracy and efficiency in certain physical problems. Further improvement can also be obtained by introducing a "grid inertia" term into the formulation (see [17]). It is expected that our iterative procedure can also be applied in a similar way to problems where directional control and orthogonality of

the mesh are important. We also note that, in this paper, we do not consider the effect of grid smoothing which is important in many applications, especially when solution (singular) structure is complex.

2. VARIOUS GRID GENERATION RULES

2.1. *Grid Distribution Based on the Equidistribution Principle in One Dimension*

The equidistribution principle was introduced by de Boor [7] for solving boundary value problems for ordinary differential equations. It involves selecting mesh points such that some measure of the solution error is equalized over each subinterval. Based on this principle, many moving mesh methods have been developed.

Let x and ξ denote the physical and computational coordinates, respectively, on the unit interval [0, 1]. A one-to-one coordinate transformation between these domains is denoted by

$$\begin{cases} x = x(\xi), & \xi \in [0, 1] \\ x(0) = 0, & x(1) = 1. \end{cases} \tag{2.1}$$

Suppose that a uniform mesh is given on the computational domain by

$$\xi_i = \frac{i}{n}, \qquad i = 0, 1, \ldots, n,$$

where n is a certain positive integer, and denote the corresponding mesh in x by $\{x_0, x_1, \ldots, x_n\}$. For a chosen monitor function $w(x)$ (>0), which provides some measure of the computational error in the solution $u(x)$ of the underlying physical PDE, the (one-dimensional) equidistribution principle can be expressed in its integral form as

$$\int_0^x w(s)\, ds = \xi C, \tag{2.2}$$

where

$$C = \int_0^1 w(s)\, ds, \tag{2.3}$$

or equivalently, in the discrete form,

$$\int_{x_i}^{x_{i+1}} w(x)\, dx = \int_{x_{i-1}}^{x_i} w(x)\, dx \qquad \text{for} \qquad i = 1, 2, \ldots, n-1.$$

Differentiating (2.2) once, we obtain a differential form,

$$w(x(\xi))\frac{\partial}{\partial \xi} x(\xi) = C. \tag{2.4}$$

When $w(x) = \sqrt{1 + u_x^2}$, the above method is known as the "arclength method."

2.2. *Grid Distribution Based on the Variational Principle*

The above equidistribution principles cannot be generalized directly to two or higher dimensions. In fact, equidistribution can be achieved locally in only a certain way [9]. In two (or higher) spatial dimensions, mesh adaption is commonly done using the variational approach, specifically by minimizing a functional of the coordinate mapping between the physical domain and the computational domain. The functional is chosen so that the minimum is suitably influenced by the desired properties of the solution of the PDE itself.

Again, let $\mathbf{x}$ and ξ denote the physical and computational coordinates, respectively, on a domain $\Omega \in \mathbf{R}^d$. A one-to-one coordinate transformation on Ω is denoted by

$$\mathbf{x} = \mathbf{x}(\xi), \qquad \xi \in \Omega. \tag{2.5}$$

The functionals used in existing variational approaches for mesh generation and adaptation can usually be expressed in the form

$$E(\xi) = \int_\Omega \sum_{i,j,\alpha,\beta} g^{i,j} \frac{\partial \xi^\alpha}{\partial x^i} \frac{\partial \xi^\beta}{\partial x^j}\, d\mathbf{x}, \tag{2.6}$$

where $G = (g_{i,j})$, $G^{-1} = (g^{i,j})$ are symmetric positive definite matrices that are monitor functions in a matrix form. The coordinate transformation and the mesh are determined from the Euler–Lagrange equation,

$$\nabla \cdot (G^{-1} \nabla \xi) = 0. \tag{2.7}$$

Equations (2.6) and (2.7) are related to the theory of the harmonic map, where the Hamilton–Schoen–Yau theorem guarantees the existence and uniqueness of the mapping with nonvanishing Jacobian.

We note that more terms can be added to the functional (2.6) to control other properties of the mesh, such as orthogonality of the mesh and the alignment of the mesh lines with a prescribed vector field [4].

2.3. *Winslow's Variable Diffusion Equation*

A special case of (2.6) is Winslow's variable diffusion method. Winslow [16] suggested a functional of the form

$$E(\xi) = \int_\Omega \sum_j \frac{1}{w} |\nabla \xi^j|^2\, d\mathbf{x}, \tag{2.8}$$

where $w > 0$ is a weight function depending on the physical solution to be adapted, and this corresponds to (2.6) with the monitor function,

$$G = wI.$$

The Euler–Lagrange equations whose solution minimizes E are

$$\nabla \cdot \frac{1}{w} \nabla \xi^j = 0, \tag{2.9}$$

which are diffusion equations.

The diffusion coefficient $D = \frac{1}{w}$ can be directly related to the cell area or volume [1]. We denote the Jacobian of the mapping as defined as

$$J^* = \xi_x \eta_y - \xi_y \eta_x$$

for the two-dimensional case. Let $\bar{D} = \ln D$, $\bar{J} = -\ln J^*$. A straightforward calculation gives

$$\nabla^2(\bar{J} - \bar{D}) - \nabla \bar{J} \cdot \nabla(\bar{J} - \bar{D}) = -RJ^{*-1},$$

where

$$R = 2\left[\nabla\left(\frac{\partial \xi}{\partial x}\right) \cdot \nabla\left(\frac{\partial \eta}{\partial y}\right) - \nabla\left(\frac{\partial \xi}{\partial y}\right) \cdot \nabla\left(\frac{\partial \eta}{\partial x}\right)\right].$$

If we assume the term on the right side, RJ^{*-1}, to be small, then we have

$$\nabla^2(\bar{J} - \bar{D}) - \nabla \bar{J} \cdot \nabla(\bar{J} - \bar{D}) = 0$$

and a solution may readily be written as

$$\nabla(\bar{J} - \bar{D}) = 0$$

or

$$(\bar{J} - \bar{D}) = \text{constant} \tag{2.10}$$

and

$$DJ^* = C,$$

where C is a constant. This shows that D is proportional to the J^{*-1}. Therefore, equidistribution is approximately satisfied in the regions where RJ^{*-1} is small.

2.4. *Solving the Generator Equation by Heat Flow*

In our applications, the solutions of the elliptic system (2.9) are obtained as a steady state of the heat flow equations [10],

$$\begin{cases} \frac{\partial \xi}{\partial t} - \nabla \cdot \left(\frac{1}{w} \nabla \xi\right) = 0 \\ \frac{\partial \eta}{\partial t} - \nabla \cdot \left(\frac{1}{w} \nabla \eta\right) = 0. \end{cases} \tag{2.11}$$

That is, we solve the evolution Eq. (2.11) for long time until the solution becomes almost steady. This steady state is used as the solution to (2.9). Note that t in (2.11) should not be confused with the time variable t used in the underline nonlinear PDE (5.1). For actual computation, the dependent and independent variables in (2.11) are interchanged and we have

$$\begin{aligned} \mathbf{x}_t = &-\frac{\mathbf{x}_\xi}{J}\left\{\frac{\partial}{\partial \xi}\left(\mathbf{x}_\eta^T \frac{1}{Jw} \mathbf{x}_\eta\right) - \frac{\partial}{\partial \eta}\left(\mathbf{x}_\xi^T \frac{1}{Jw} \mathbf{x}_\eta\right)\right\} \\ &-\frac{\mathbf{x}_\eta}{J}\left\{\frac{\partial}{\partial \xi}\left(\mathbf{x}_\eta^T \frac{1}{Jw} \mathbf{x}_\xi\right) + \frac{\partial}{\partial \eta}\left(\mathbf{x}_\xi^T \frac{1}{Jw} \mathbf{x}_\xi\right)\right\}, \end{aligned} \tag{2.12}$$

where the Jacobian $J = x_\xi y_\eta - x_\eta y_\xi$. The steady-state solution of (2.12) serves as the desired solution of (2.9). We can start from a uniform mesh, i.e., $x(\xi, \eta, 0) = \xi$, $y(\xi, \eta, 0) = \eta$, or from an appropriate initial guess provided in the problem.

It is obvious that, to improve efficiency, more sophisticated elliptic solvers, such as the multigrid method, can be used to solve the elliptic system. In our method, grid redistribution were needed only once in a while. Computational cost for this part is not an issue (at least in two dimensions).

3. PROPERTIES OF THE RESULTING GRID DISTRIBUTION IN THE EXTREME CASE

From equidistribution property in one dimension and the properties of the solution of the Winslow equations, we expect more grid points will be concentrated at the regions of rapid variation of the solution. However, we will show that when the solution becomes singular, such grid adaptivity is actually lost with the above methods.

3.1. *Numerical Tests*

We first consider a one-dimensional example. Let

$$u(x) = \frac{1}{\varepsilon} e^{-(\frac{x-0.5}{\varepsilon})^2}$$

and the monitor function be $w = \sqrt{1+u}$. We show the grid distribution for various ε in Fig. 1. It is clear from the graph that as one decreases ε, the grid points move toward the center first, but most of the points then move away from the center as ε continues to decrease.

A two-dimensional example for function

$$u(x, y) = ce^{-c^2(x^2+y^2)}$$

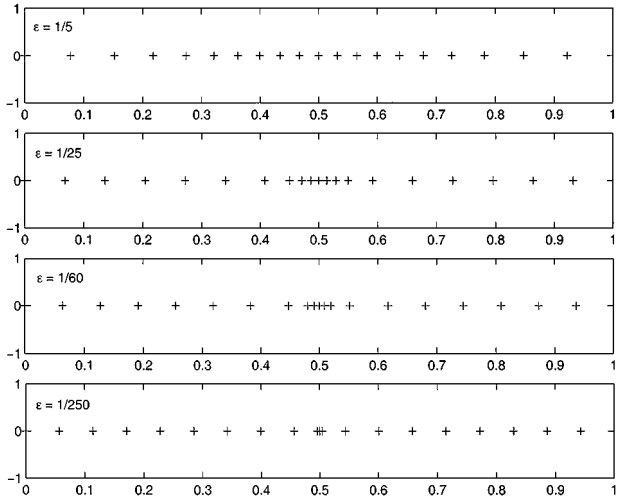

FIG. 1. Grid behavior as ε decreases.

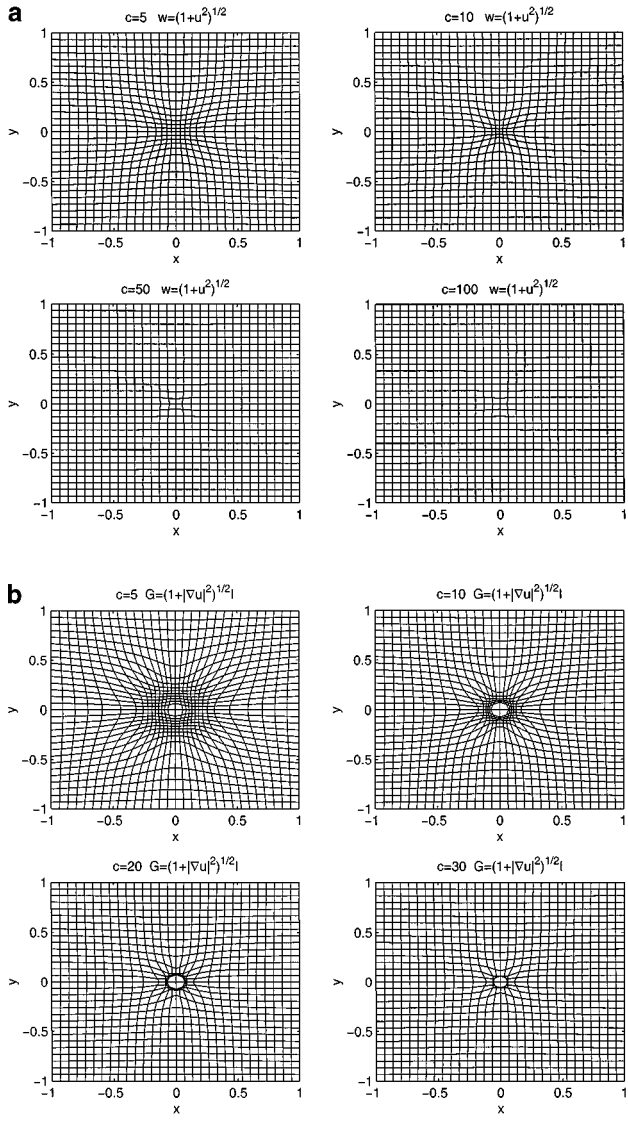

FIG. 2. Grid behavior for $u = ce^{-c^2(x^2+y^2)}$ in 2-d as c increases.

shows similar grid behavior. Figure 2 shows the grid distributions obtained from (2.9) for two different monitor functions

$$\text{(a) } w(x, y) = \sqrt{1 + |u|^2}, \quad \text{(b) } w(x, y) = \sqrt{1 + |\nabla u|^2}.$$

In both cases, we see that, just like in one dimension, grid adaption is achieved for smaller c and then lost when c is large.

Other examples with various choices of monitor functions exhibit similar phenomena. Such behavior seems to be generic when the monitor function becomes singular.

3.2. *Asymptotic Analysis*

To understand the above phenomena, let's assume that we have a monitor function that is close to a self-similar singularity characterized by a parameter ε,

$$w_\varepsilon(x) = \sqrt{1 + \frac{1}{\varepsilon^k} g\left(\frac{x}{\varepsilon}\right)},$$

where $g(y)$ is positive, has maximum at $x = 0$, and decays rapidly (say, faster than $\frac{1}{y^n}$ for a large enough n) as $y \to \infty$. We want to study the behavior of the grid as $\varepsilon \to 0$. From (2.4), we have

$$x_\xi = \frac{C(\varepsilon)}{w} = \frac{C(\varepsilon)}{\sqrt{1 + \frac{1}{\varepsilon^k} g\left(\frac{x}{\varepsilon}\right)}}, \tag{3.1}$$

where

$$C(\varepsilon) = \int_0^1 \sqrt{1 + \frac{1}{\varepsilon^k} g\left(\frac{x}{\varepsilon}\right)}\, dx.$$

Integration by parts, we have

$$\begin{aligned} C(\varepsilon) &= x\sqrt{1 + \frac{1}{\varepsilon^k} g\left(\frac{x}{\varepsilon}\right)}\Bigg|_0^1 - \int_0^1 \frac{x \frac{1}{\varepsilon^k} g'(x)}{2\sqrt{1 + \frac{1}{\varepsilon^k} g\left(\frac{x}{\varepsilon}\right)}}\, dx \\ &= \sqrt{1 + \frac{1}{\varepsilon^k} g\left(\frac{x}{\varepsilon}\right)} - \varepsilon^{1-\frac{k}{2}} \int_0^{\frac{1}{\varepsilon}} \frac{y g'(y)}{2\sqrt{\varepsilon^k + g(y)}}\, dy. \end{aligned}$$

Since $g(y)$ decays rapidly as $y \to \infty$, we have, as $\varepsilon \to 0$,

$$C(\varepsilon) \sim 1 + A\varepsilon^{1-\frac{k}{2}}, \tag{3.2}$$

where

$$A = -\int_0^\infty \frac{y g'(y)}{2\sqrt{g(y)}}\, dy = \int_0^\infty \sqrt{g(y)}\, dy.$$

We thus have for $x \neq 0$ (i.e., away from the singularity) and $\varepsilon \to 0$

$$\begin{aligned} x_\xi &\to 1 \quad && \text{when} \quad && k < 2 \\ x_\xi &\to 1 + A \quad && \text{when} \quad && k = 2 \\ x_\xi &\to \infty \quad && \text{when} \quad && k > 2. \end{aligned} \tag{3.3}$$

This implies that, in the case $k < 2$, we have $\Delta X = \Delta\xi$ away from the singularity. Therefore, as $\varepsilon \to 0$, only grid points very close to 0 are allowed to move; i.e., the grid adaption is very limited near the place that function g is large. This also shows that one has to know the solution behavior in some detail in order to design the monitor function which can generate the desired grid distribution near the singularity.

4. AN ITERATIVE REMESHING PROCEDURE

To improve the mesh adaption, we introduce an iterative remeshing procedure. Let us first define the Winslow mapping:

$$\mathbf{T}\colon (\mathbf{x}, u(\mathbf{x})) \to (\xi, v(\xi)).$$

Here $\mathbf{x} = \mathbf{x}(\xi)$ is determined from (2.9), where the monitor function $w(x)$ depends on $u(x)$; $v(\xi) = u(\mathbf{x}(\xi))$.

If the monitor function w is chosen properly, the resulting mesh should concentrate more grid points in the regions with large variations. This also means that $v(\xi)$ should be better behaved than the original function $u(x)$ in the sense that the variation of the monitor function in the new variables is reduced. However, the examples in the previous section show that in some cases, such improvement is very limited. A natural idea to improve further is to repeat the same procedure for $v(\xi)$. In fact, this process can be repeated until a satisfactory $v(\xi)$ is achieved. Based on this intuition, an iterative remeshing procedure is introduced by applying the Winslow mapping $\mathbf{T}$ iteratively:

- Let $u^k(\mathbf{x})$ be the function after k iterations.
- Determine the mapping $\mathbf{x}^{k+1}(\xi)$ from $u^k(\mathbf{x})$ according to (1.4) or (1.5), where monitor function w^k is defined using $u^k(x)$.
- Define $u^{k+1}(\xi) := u^k(\mathbf{x}^{k+1}(\xi))$.

The results of the iteration is to flatten out the monitor function gradually. In fact, if $u^k(\mathbf{x})$ and $\mathbf{x}^k(\xi)$ converge, then we must have $\mathbf{x}^k \to \mathbf{x}^*(\xi) = \xi$ and $u^k \to u^*(\mathbf{x})$.

Claim. w^k converges to a constant function almost everywhere.

This shows that we achieve the maximum adaption for the monitor function. The above claim is verified by many of our numerical examples below, although so far, rigorous proof can only be obtained in some special cases.

EXAMPLE 1. Let $u(x) = \exp(-20(x - 0.5)^2)$ on $[0, 1]$. We apply the above iteration using three different monitor functions:

$$\text{(a) } w = \sqrt{1 + u_x^2}, \quad \text{(b) } w = \sqrt{1 + u^2}, \quad \text{(c) } w = \sqrt{1 + 0.1u_x^2 + u^2}.$$

In Fig. 3, we show $u^k(x)$ for different k. Figure 3a shows that $u^k(x)$ converges to two straight lines, i.e., $|u_x^k(x)|$ converges to constants a.e. Figure 3b shows that $|u^k(x)|$ converges to 1. Figure 3c shows that a combination of (a) and (b) gives a limiting function with much better behavior. In all cases, we have $w \to$ Constant a.e. as $k \to \infty$.

EXAMPLE 2. Let $u(x, y) = e^{(-5(x^2+y^2))}$. We use three different monitor functions:

$$\text{(a) } w = \sqrt{1 + |\nabla u|^2}, \quad \text{(b) } w = \sqrt{1 + u^2}, \quad \text{(c) } w = \sqrt{1 + |\nabla u|^2 + u^2}.$$

Again, in case (a), we see that u^k converges to a cone so that $|\nabla u|^2$ converges to a constant. In case (b), $|u^k|$ converges to 1. In case (c), the limiting function has much better behavior. In all cases, we again have $w \to$ Constant a.e. as $k \to \infty$. Only pictures for case (a) are shown in Fig. 4.

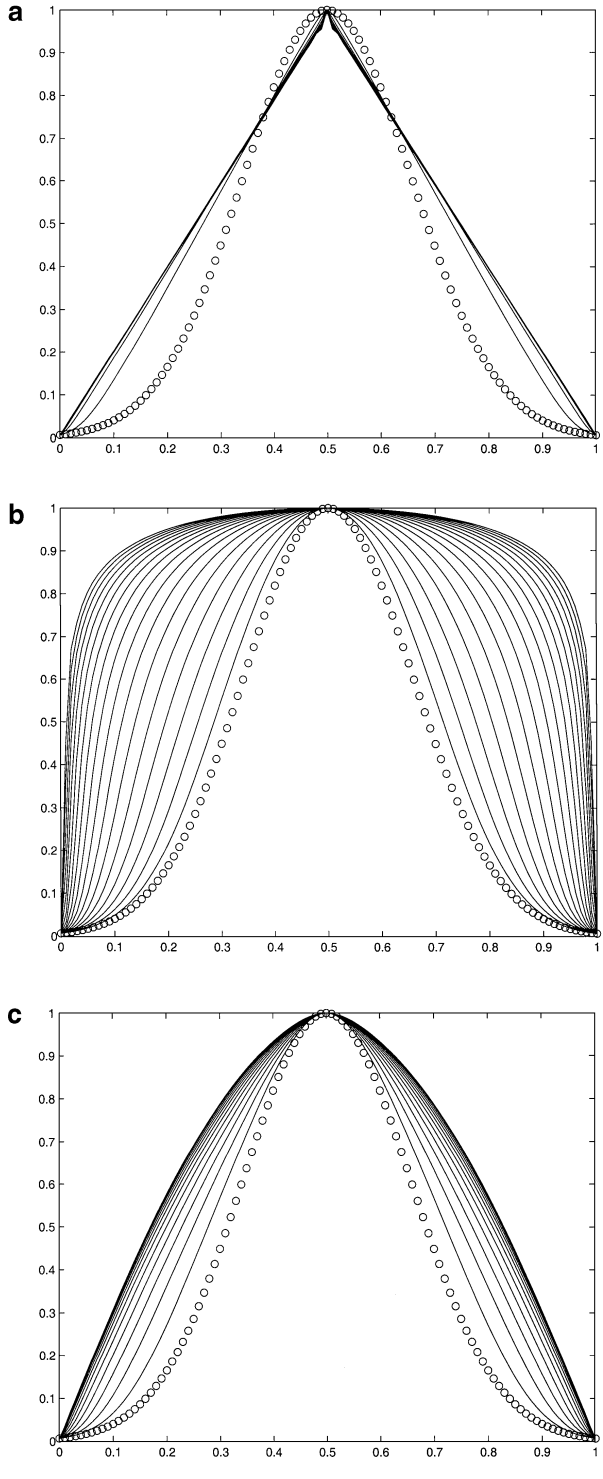

FIG. 3. Iterative remeshing for three different monitor functions. Circles represent $u(x)$ in physical variable.

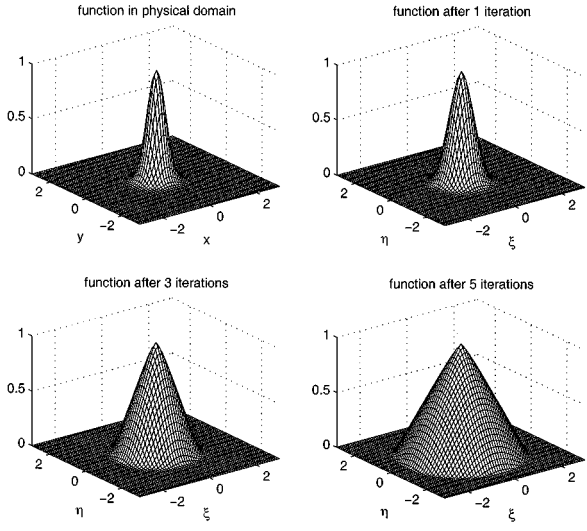

FIG. 4. Iterative remeshing in two dimensions with monitor function $w = \sqrt{1 + |\nabla u|^2}$.

The above examples also suggest that to achieve better limiting behavior in the computational domain, the monitor function should include both $|u|$ and $|\nabla u|$. Theoretically, the more higher order derivatives are added (although difficult to implement in practice), the better limiting behavior of the function in the computational domain is expected. To see this, let the monitor function be $w = \sqrt{1 + |u| + |Du| + |D^2u|}$. Since in the iteration limit, the monitor function w tends to a constant, which is the integral average of w, we have that, in the computational variables, $\max |Du|$, $\max |D^2u|$ are all bounded by the same constant.

EXAMPLE 3. Next, we show the capability of improving the mesh adaption by the iterative remeshing in Fig. 5. Again for function $u(x, y) = ce^{-c^2(x^2+y^2)}$ with $c = 50$, we see

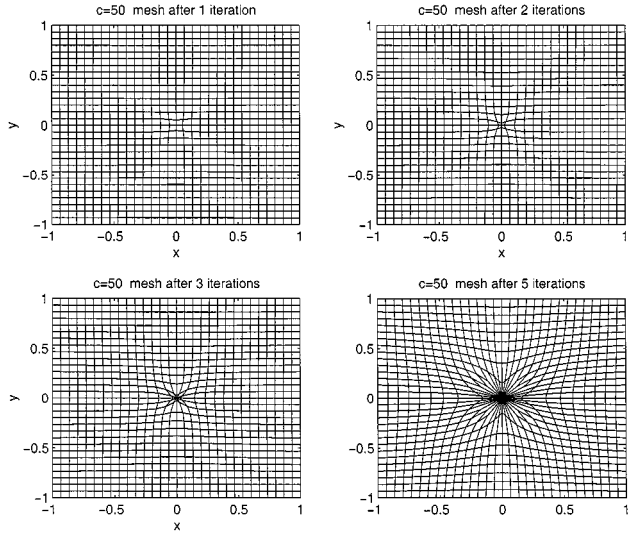

FIG. 5. Improved mesh distribution (compare to Fig. 2) for $u = ce^{-c^2(x^2+y^2)} (c = 50)$ after several iterations.

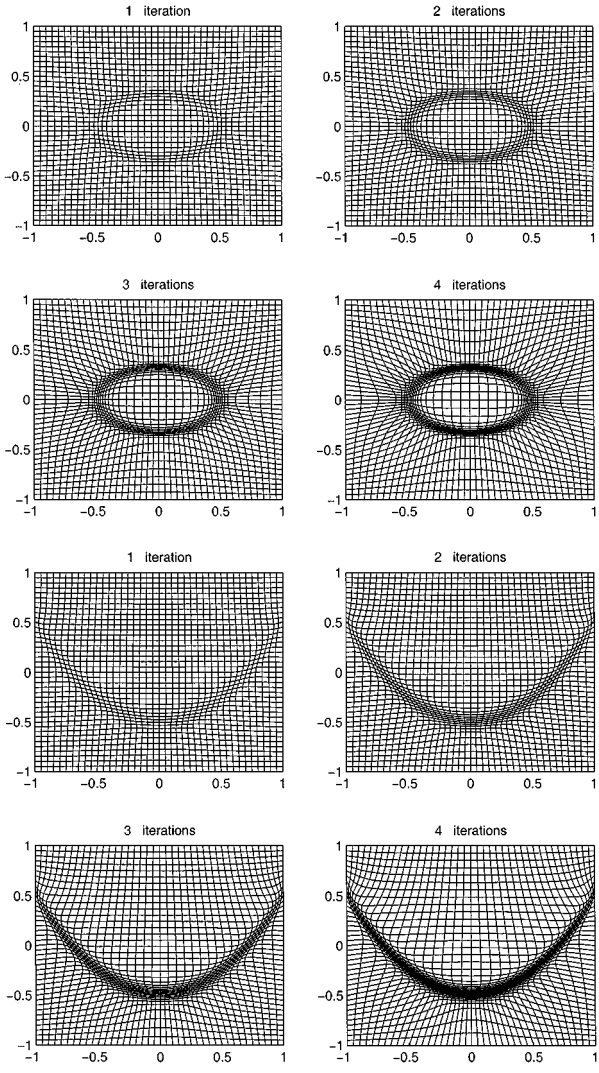

FIG. 6. Iterative remeshing for $u = \exp(-8(4x^2 + 9y^2 - 1)^2)$ (top) and $u = \exp(-100(y - x^2 + 0.5)^2)$ (bottom). In both cases, we choose $w = 1 + u$.

significant improvement (compared to Fig. 2) in mesh concentration at the origin as the number of iterations increases.

EXAMPLE 4. Figure 6 shows the effect of the iterative remeshing for function $u = \exp(-8(4x^2 + 9y^2 - 1)^2)$ and $u = \exp(-100(y - x^2 + 0.5)^2)$, respectively. The maximum points of the above two functions are on the curves. This example shows the capability of our method for handling possible line singularities.

5. ADAPTIVE PROCEDURE FOR SOLVING PDEs

In this section, we design a numerical scheme that incorporates the iterative remeshing into a static adaptive method for solutions of PDEs. For simplicity, we will only formulate

the scheme in two dimensions. Generalization to three dimensions is straightforward. Our scheme is static; i.e., we use a fixed grid to solve the PDE in the computational variables until a certain criterion is violated that indicates that the adaption is needed. Then iterative remeshing is used to generate a new grid.

Let's assume that we solve a PDE or system of PDEs of the form

$$u_t = F(\mathbf{x}, u, Du, D^2u), \mathbf{x} \in \Omega_p, \tag{5.1}$$

supplemented with initial and boundary conditions. D is the first-order differential operator. The one level grid generation is based on (2.9),

$$\begin{cases} \nabla \cdot \left(\frac{1}{w}\nabla\xi\right) = 0 \\ \nabla \cdot \left(\frac{1}{w}\nabla\eta\right) = 0, \end{cases} \tag{5.2}$$

which determines the transformation $(x(\xi, \eta), y(\xi, \eta))$ from the computational domain Ω_c to the physical domain Ω_p. The choice of the monitor function w is problem dependent. In most cases, $w = \sqrt{1 + \alpha|\nabla\vec{u}|^2 + \beta|\vec{u}|^2}$ is good enough for certain constants $\alpha > 0, \beta > 0$.

The procedure is described in the following:

(0) Given an initial condition $\vec{u}(x, y, 0)$, the initial grid transforms $x(\xi, \eta), y(\xi, \eta)$ are determined from the iterative remeshing, which in turn gives an initial condition in the computational domain $\vec{u}(x(\xi, \eta), y(\xi, \eta), 0)$.

(1) Solve the PDE in the computational variables ξ, η with the grid transformation $x(\xi, \eta), y(\xi, \eta)$ being fixed, until some time t^* when the solution $\vec{u}(\xi, \eta, t^*)$ cannot meet a certain criterion.

(2) Generate a new mesh by the iterative remeshing, starting with $\vec{u}(\xi, \eta, t^*)$. The remeshing iteration stops if the criterion in (1) is satisfied. The interpolation is used to move the solution on the new grids.

(3) Go to (1) to continue the integration.

Remark 1. The stopping criterion in step (1) depends on the specific problem. In our numerical examples below, the criterion is set so that the maximum amplitude of the gradient is smaller than a given value TOL. The choice of TOL is flexible. However, it has to be larger than the integral average of the gradient of the solution over the domain. It is easy to see that a smaller TOL requires more remeshing iterations. In actual applications, one should include quantities of interests near the singularity in both the monitor function and the stopping criterion. For example, in the fluid problem, vorticity might be the desired quantity to be included in the monitor function.

Remark 2. In most of the cases, only one remeshing iteration is needed when we start the iteration with $\vec{u}(\xi, \eta, t^*)$ in Step 2. Since we always start the iteration from the most recent solution in the computational domain, when the cycle (1)–(3) is repeated k times, effectively we have at least k remeshing iterations at t^* from the solution in the original physical variables.

Remark 3. Our grid movement method is static; i.e., the grids are held stationary during the evolution of PDEs until the stopping criterion is violated and are shifted to their new positions by our iterative procedure. The solution values are moved from the old grid to the new grid by interpolation. The interpolation is carried out on the uniform mesh and using cubic polynomial interpolations. As pointed out in Remark 2, our iterative procedure was carried

out gradually as the solution evolves toward the singularity and solution behavior in the computational domain is always controlled by the stopping criterion (e.g., $\max |\nabla \mathbf{u}| \leq \mathrm{TOL}$). Therefore interpolation errors are also controlled.

6. NUMERICAL EXAMPLES

We first solve the nonlinear Schrödinger equation (NLS),

$$\begin{cases} i\psi_t + \Delta\psi + |\psi|^{2\sigma}\psi = 0, & (x, y) \in \Omega_{\mathrm{p}},\ t > 0, \\ \psi(x, y, t)|_{\partial\Omega_{\mathrm{p}}} = 0, \end{cases} \tag{6.1}$$

in two dimensions. The singular solutions with one blowup point were successfully computed using the dynamic rescaling method first developed in [12] for the radially symmetric case and generalized in two and three dimensions in [11]. We shall first reproduce the result for the single blowup point. Then we will present an example with multiple blowup points as well as an example in the supercritical case.

Let $(x(\xi, \eta), y(\xi, \eta))$ be the spatial coordinate transformations. Both Ω_{p} (the physical domain) and Ω_{c} (the computational domain) are chosen to be $[-1, 1] \times [-1, 1]$. As in the dynamic rescaling, we also rescale the time from t to τ as

$$\frac{\mathrm{d}\tau}{\mathrm{d}t} = \frac{1}{\lambda^2(t)}. \tag{6.2}$$

Let

$$\psi = \frac{1}{L(t)}\phi,$$

where $L(t)$ is a scaling factor chosen to be

$$L(t) = \frac{1}{\max_{(x,y)\in\Omega_{\mathrm{p}}} |\psi(x, y, t)|}. \tag{6.3}$$

To balance the coefficients in the transformed equation, we choose $\lambda = L^{\sigma}$. In the coordinate system (ξ, η, τ), the NLS becomes

$$\phi_\tau - \frac{L_\tau}{L}\phi - i\left(\lambda^2 \Delta_{\mathrm{B}}\phi + |\phi|^2\phi\right) = 0, \tag{6.4}$$

together with

$$L_\tau = L^{2\sigma+1}\,\mathrm{Im}(\phi^* \Delta_{\mathrm{B}}\phi)|_{(\xi_0,\eta_0)}, \tag{6.5}$$

where $(\xi_0(t), \eta_0(t))$ is the maximum point of $|\phi(\xi, \eta, t)|$, and

$$\Delta_{\mathrm{B}}\phi = \frac{1}{J}\left\{\frac{\partial}{\partial\xi}\left(\frac{b_{22}\phi_\xi - b_{12}\phi_\eta}{J}\right) + \frac{\partial}{\partial\eta}\left(\frac{b_{11}\phi_\eta - b_{12}\phi_\xi}{J}\right)\right\},$$

$$b_{11} = x_\xi^2 + y_\xi^2, \qquad b_{12} = x_\xi x_\eta + y_\xi y_\eta, \qquad b_{22} = x_\eta^2 + y_\eta^2.$$

J is the Jacobian of the coordinate transformation.

In the grid redistribution, the monitor function is taken to be $w(\xi, \eta) = (1 + 2|\phi(\xi, \eta)|^2 + |\nabla\phi(\xi, \eta)|^2)$. The criterion is set so that the maximum amplitude of the gradient is smaller than a given value of TOL. Two values of TOL, 5 and 7, are used for our computations.

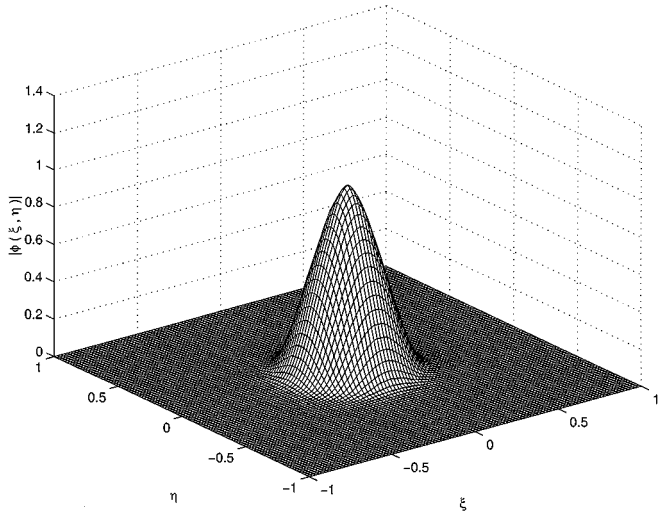

FIG. 7. $\phi(\xi, \eta, \tau)$ in Ω_c (computational domain) at $\tau = 95.82$ ($t = 0.024317104$), $1/\lambda = 2.9514e + 05$ (NLS with initial value (i)).

1. In the first example, we solve the NLS in the critical case ($\sigma = 1$) with the initial condition

i. $\psi(x, y, 0) = 10e^{-4x^2-9y^2}$.

Equations (6.4) and (6.5) are solved simultaneously on Ω_c with 100×100 grid points and TOL $= 5$. About 20 remeshing steps are conducted before the computation reaches $\tau = 95.82$ (or $t = 0.024317104$) when the maximum value of the solution is 2.9514×10^5. Figures 7 and 8 show solutions in the computational and the physical variables, respectively, at $\tau = 95.82$. The mesh distribution is shown in Figs. 9 and 10, where one can see that the minimum mesh size is about 10^{-6}. A cross section of the solution in both the computational

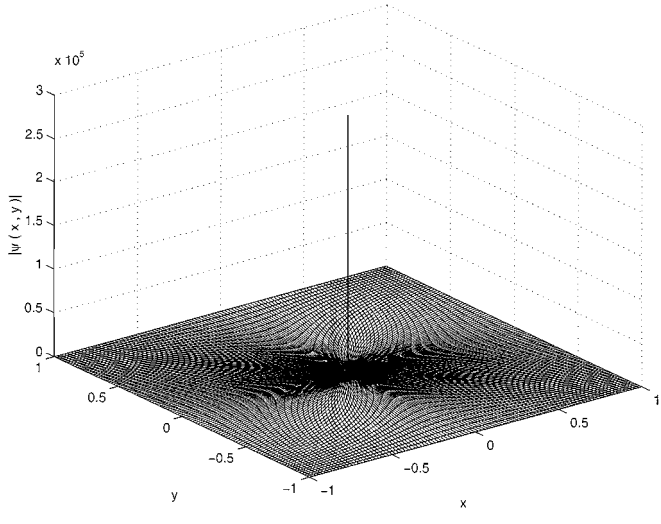

FIG. 8. $\psi(x, y, t)$ in Ω_p (physical domain) at $t = 0.024317104$ corresponding to Fig. 7 (NLS with initial value (i)).

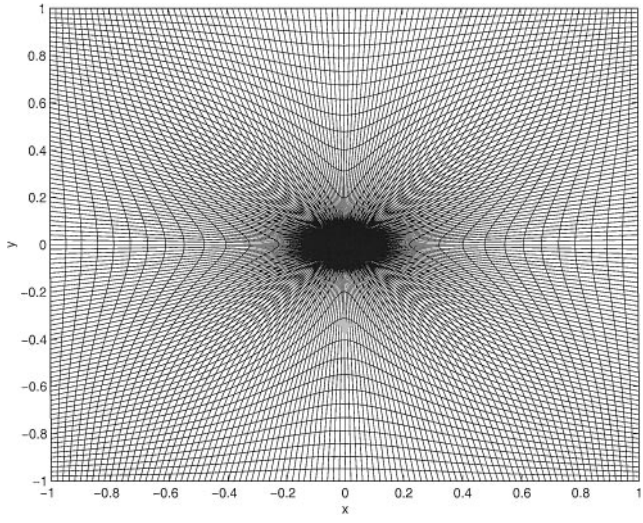

FIG. 9. Mesh in Ω_p at $t = 0.024317104$ ($\tau = 95.82$) corresponding to Fig. 8 (NLS with initial value (i)).

and the physical variables are shown in Figs. 11 and 12. The computation is well resolved with more than 40 grid points within the interval of size 10^{-5}.

To verify the stable property of the isotropic singularity, a result first shown numerically by the dynamic rescaling method [11], we define two scaling factors in the x and y directions:

$$L_1 = \sqrt{\frac{\int_{\Omega_p} |\psi_x|^2 \, dx \, dy}{\int_{\Omega_p} |\psi|^2 \, dx \, dy}}$$

$$L_2 = \sqrt{\frac{\int_{\Omega_p} |\psi_y|^2 \, dx \, dy}{\int_{\Omega_p} |\psi|^2 \, dx \, dy}}$$

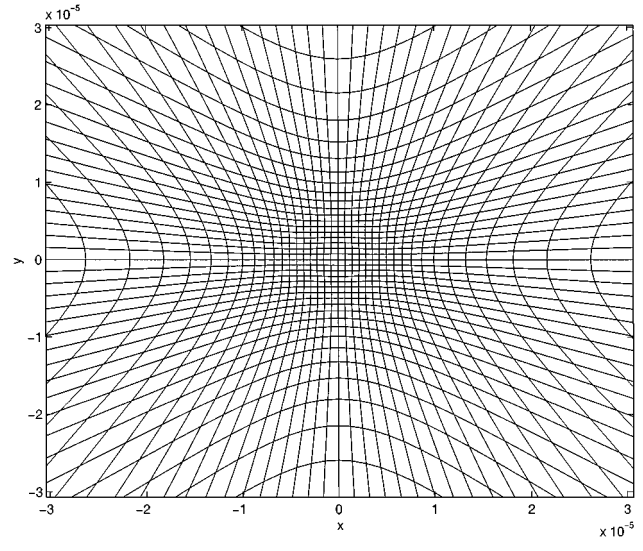

FIG. 10. An enlarged picture of the above mesh around the center (NLS with initial value (i)).

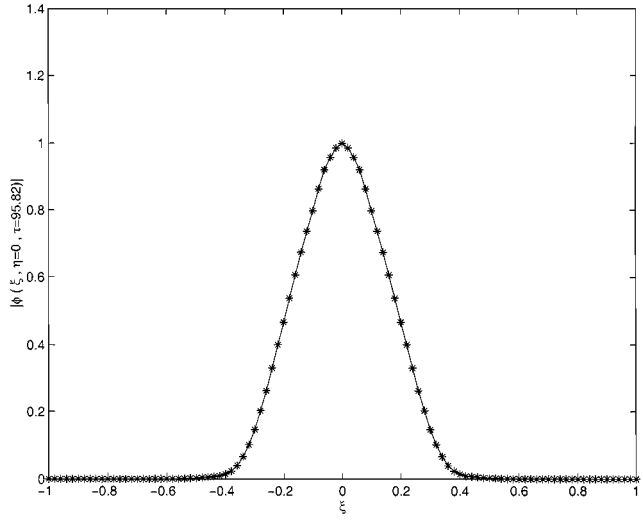

FIG. 11. A cross section of Fig. 7 at $\eta = 0$ (NLS with initial value (i)).

Figure 13 shows the ratio of $\frac{L_1}{L_2}$ as a function of τ and the limit converges to 1 which shows that the solution converges to an isotropic singularity. Figure 14 shows $\frac{\lambda_\tau}{\lambda}(=\sigma\frac{L_\tau}{L})$ as a function of τ, which approaches 0 slowly, which leads to the blowup rate.

All the results above are consistent with the results obtained from the dynamic rescaling method. To check the convergence, we run our code with 100×100 and 160×160 grid points, respectively, and the results are compared in Fig. 15. The solution profiles are in excellent agreement. To show that our numerical results are insensitive to the remeshing criterion (i.e., the value of TOL), we also run the code with TOL $= 7$. The result is also in good agreement with the result obtained with TOL $= 5$ (Fig. 16). Our computations were performed on a single node SGI O2000 machine. The total CPU time for this run is 5590 s,

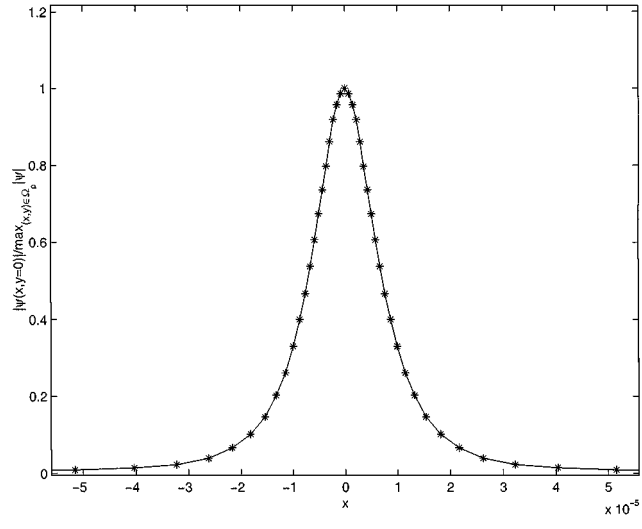

FIG. 12. A cross section of Fig. 8 at $y = 0$ (NLS with initial value (i)).

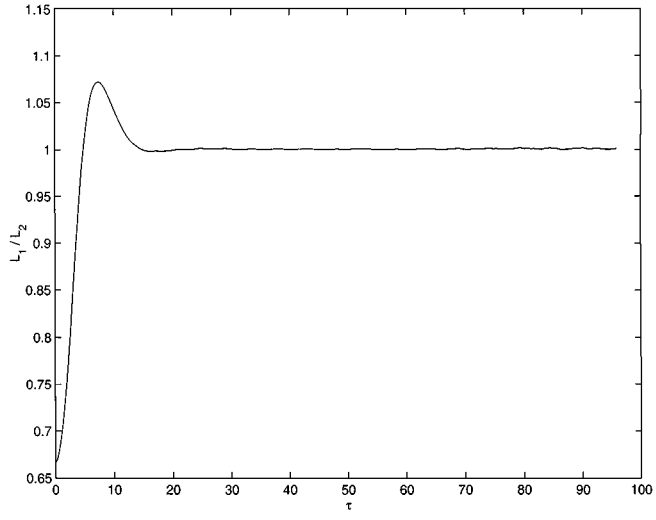

FIG. 13. Ratio $\frac{L_1}{L_2}$ as a function of τ (NLS with initial value (i)).

in which 1650 s are used in the remeshing (Step 2). If one uses a uniform grid with a resolution of grid size 10^{-6} and an explicit scheme in time, to reach the same maximum value, the estimated CPU time would be 1.2665×10^4 h!

2. In the second example, we use the initial condition

ii. $\psi(x, y, 0) = 20(e^{-20((x+0.5)^2+y^2)} + e^{-20((x-0.5)^2+y^2)})$,

with two maximum points. Again, we take $\sigma = 1$. Our results show that the solution blows up at two points. Such a calculation cannot be done by the dynamic rescaling method because one can rescale only around one point. We have not seen other successful calculations on such problems. We solve the equation with 160×160 grid points in the computational domain and TOL $= 7$. Figures 17 and 18 show solutions in computational and physical variables, respectively, at $\tau = 76.40$ ($t = 0.005651$) when the maximum of the solution

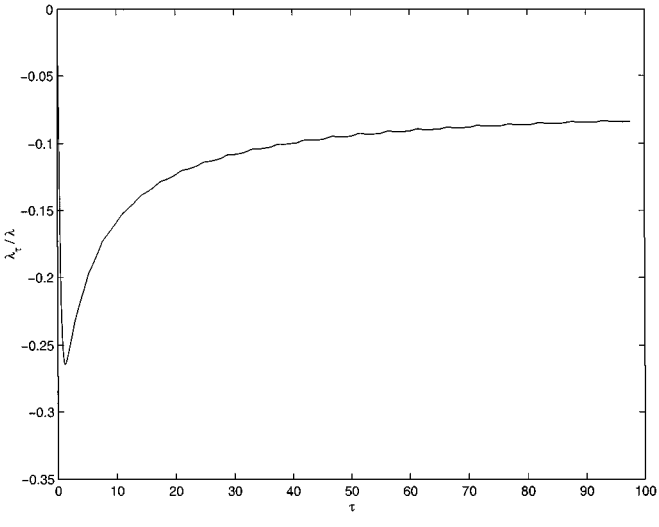

FIG. 14. $a(\tau) = \frac{\lambda_\tau}{\lambda}$ as a function of τ (NLS with initial value (i)).

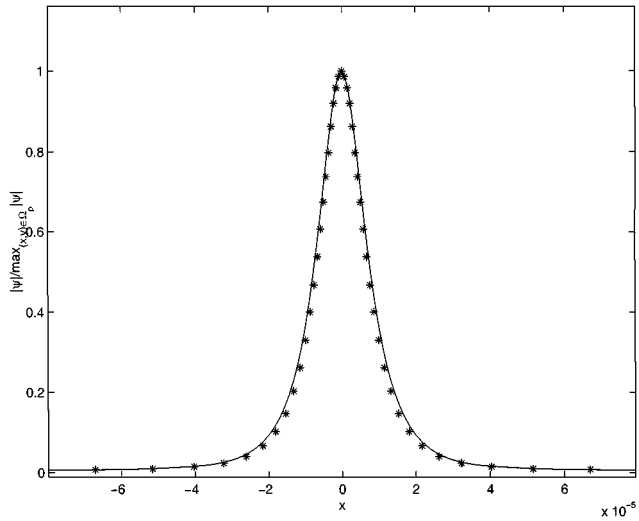

FIG. 15. A cross section of $|\psi(x, y, t)|$ at $y = 0$, $\tau = 95.82$. 100×100 (160×160) grid points are used for $*$ (—), and the maximum value at this time is $2.9514e + 05$ ($2.7568e + 05$). TOL $= 5$ (NLS with initial value (i)).

reaches 1.488×10^5. The grid distribution at the same time is shown in Figs. 19 and 20. The blowup structure of each singularity is the same as that of the solution with single blowup point.

3. In the third example, we solve the NLS in the supercritical case ($\sigma = 2$) with initial condition

iii. $\psi(x, y, 0) = 10e^{-9x^2-9y^2}$.

The dynamics is much faster in the rescaled time variable τ than in the critical case. We plot the solution at $\tau = 34.3$ in both computational variables and physical variables in Figs. 21 and 22. A distinct property of the supercritical collapse is that there seems to be a finite

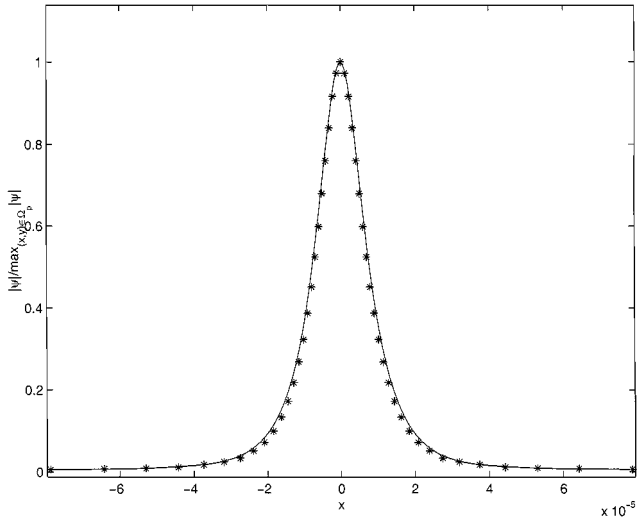

FIG. 16. A cross section of $|\psi(x, y, t)|$ at $y = 0$, $\tau = 95.82$. TOL $= 7$ (TOL $= 5$) is used for $*$ (—), and the maximum value at this time is $2.9279e + 05$ ($2.7568e + 05$). 160×160 (NLS with initial value (i)).

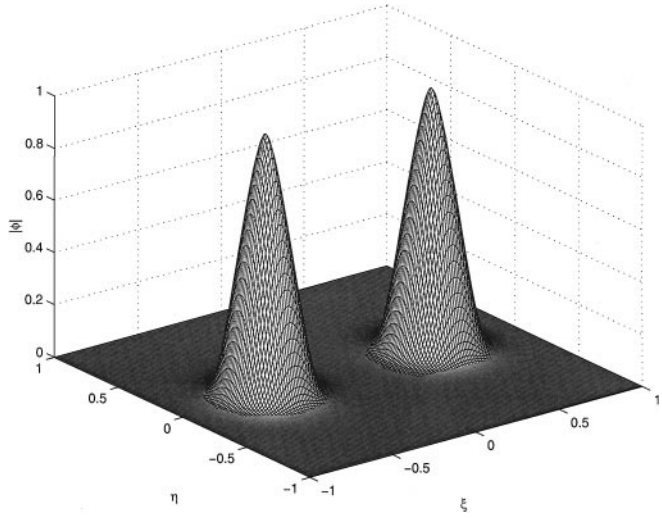

FIG. 17. $\phi(\xi, \eta, \tau)$ in Ω_c at $\tau = 76.40$ ($t = 0.005651$), $1/\lambda = 1.4880e + 05$ (NLS with initial value (ii)).

region of self-similarity which is time independent. In a paper of Shvets *et al.* [15], this behavior was shown by an asymptotic analysis and supported by numerical experiments based on an improved dynamic rescaling method in the radially symmetric case. Here we show that our method also captures the phenomenon. In Fig. 23 we show a cross section of the solution in the computational variable and an enlarged picture (Fig. 24) clearly shows the transition from the algebraic decay of the self-similar profile to the exponential decay of the tail. In Fig. 25, we show $|\psi| \cdot |x|^{\frac{1}{\sigma}}$ against $\log(|x|)$ at several time steps τ near the blowup time. It shows a transition region (or boundary of self-similarity) near $\log(|x|) = -6$ or $|x| = 0.025$ which seems to be independent of time.

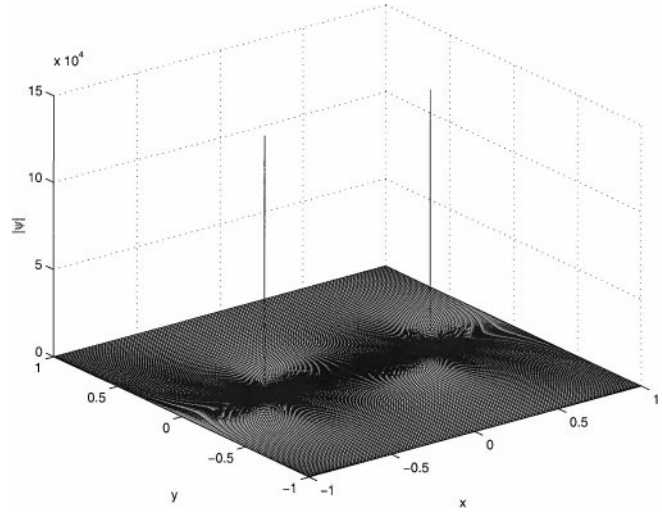

FIG. 18. $\psi(x, y, t)$ in Ω_p at $t = 0.005651$ corresponding to Fig. 17 (NLS with initial value (ii)).

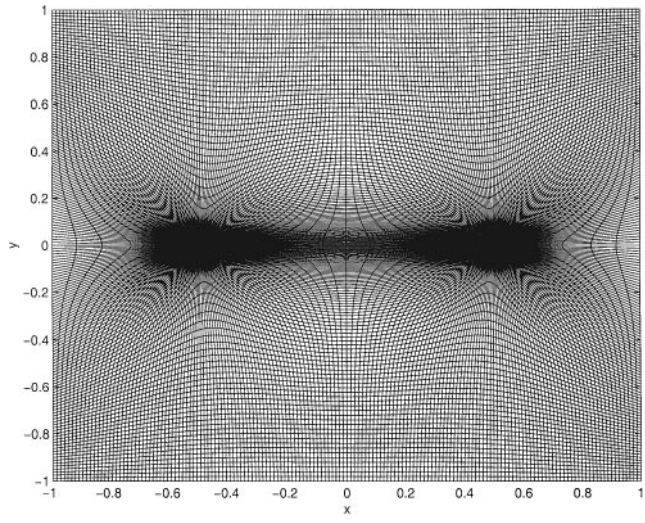

FIG. 19. Mesh in Ω_p at $t = 0.005651$ ($\tau = 76.40$) corresponding to Fig. 18 (NLS with initial value (ii)).

4. Our next example is the Keller–Segal (KS) model for bacterial pattern formation:

$$\begin{cases} \rho_t = \epsilon \Delta \rho - \nabla \cdot (\rho \nabla c), & (x, y) \in \Omega_p, \quad t > 0, \\ c_t = \Delta c + \rho. \end{cases} \tag{6.6}$$

Here ρ is the bacterial density and c is the attractant field. In some cases, the concentration of the attractant c draws the bacteria together and they achieve an infinite density at some finite time [5, 6].

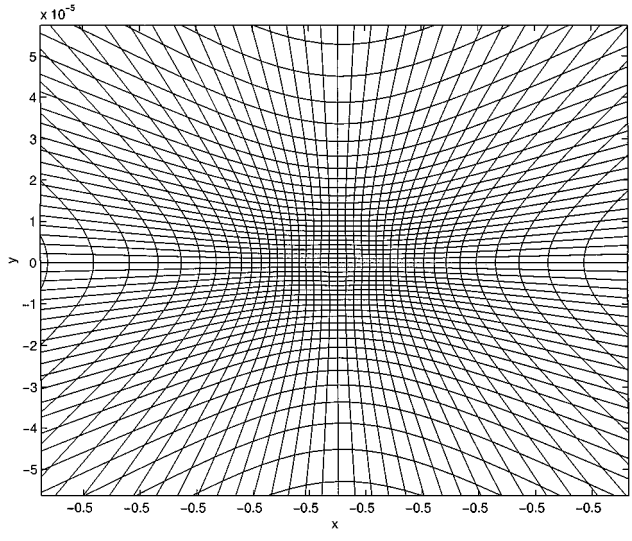

FIG. 20. Enlarged view of the above mesh in Ω_p around one blowup point at $t = 0.005651$ ($\tau = 76.40$) corresponding to Fig. 18 (NLS with initial value (ii)).

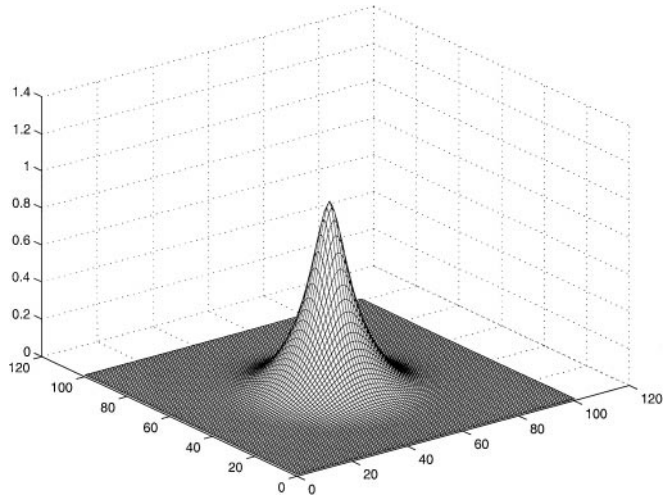

FIG. 21. Solution in Ω_c at $\tau = 34.3$ (NLS with $\sigma = 2$ and initial value (iii)).

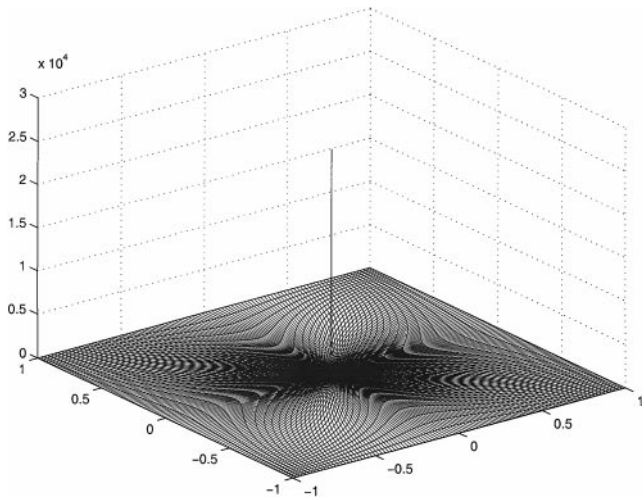

FIG. 22. Solution in Ω_p at $\tau = 34.3$ (NLS with $\sigma = 2$ and initial value (iii)).

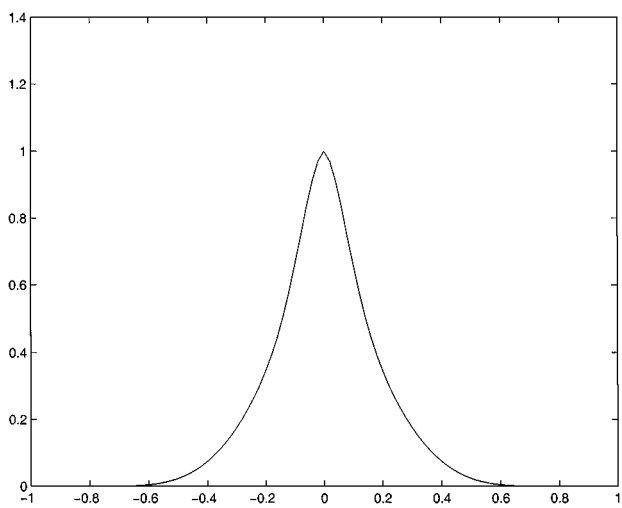

FIG. 23. A cross section of Fig. 21 at $\eta = 0$ (NLS with $\sigma = 2$ and initial value (iii)).

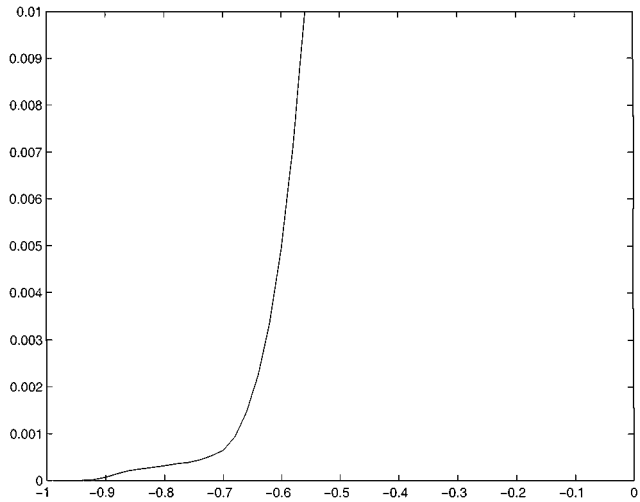

FIG. 24. An enlarged view of Fig. 23 near the boundary.

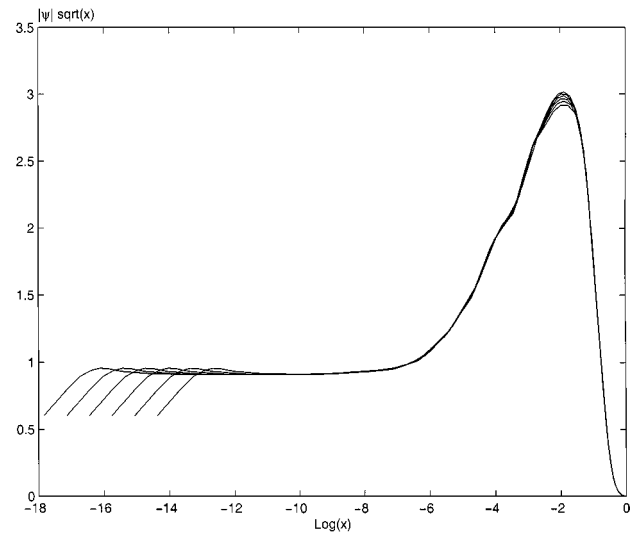

FIG. 25. $|\psi| \cdot \sqrt{|x|}$ are plotted against $\log(|x|)$ for several time steps closed to blowup time.

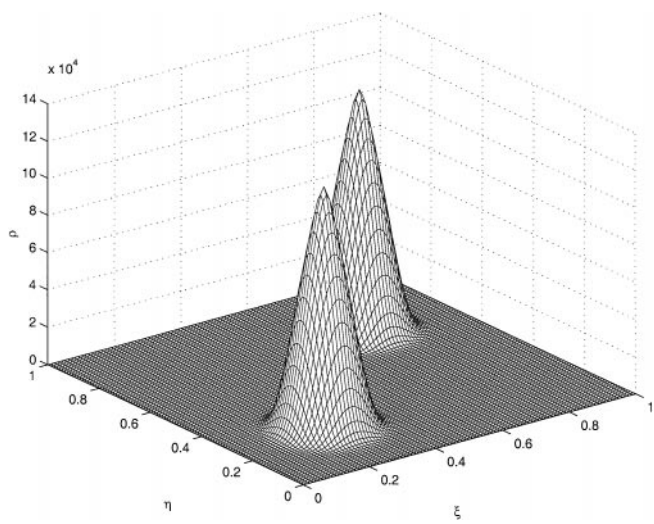

FIG. 26. ρ in Ω_ϵ at $t = 2.7775$ (KS).

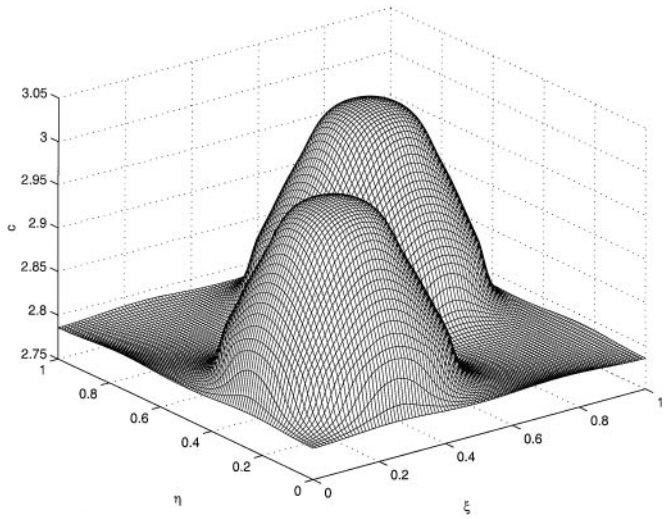

FIG. 27. c in Ω_c at $t = 2.7775$ (KS).

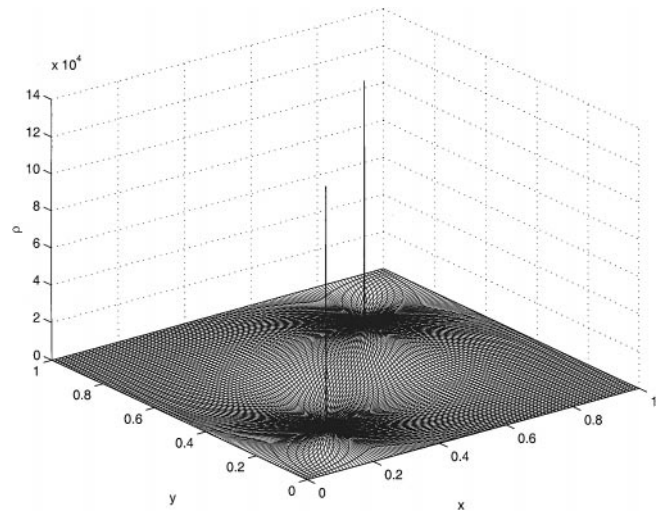

FIG. 28. ρ in Ω_ρ at $t = 2.7775$ corresponding to Fig. 26 (KS).

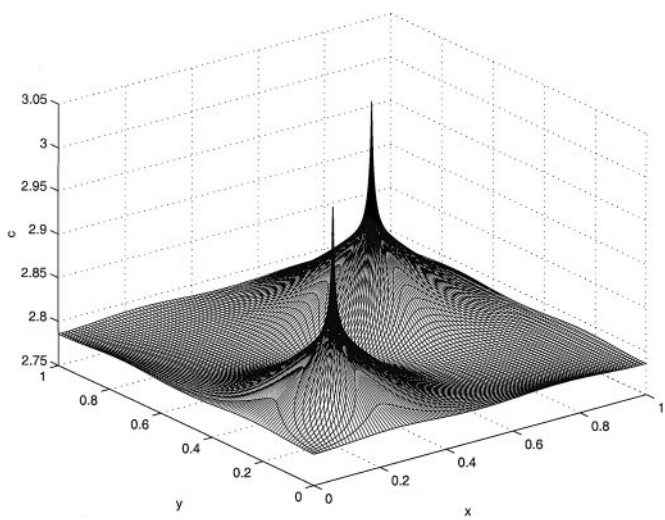

FIG. 29. c in Ω_ρ at $t = 2.7775$ corresponding to Fig. 27 (KS).

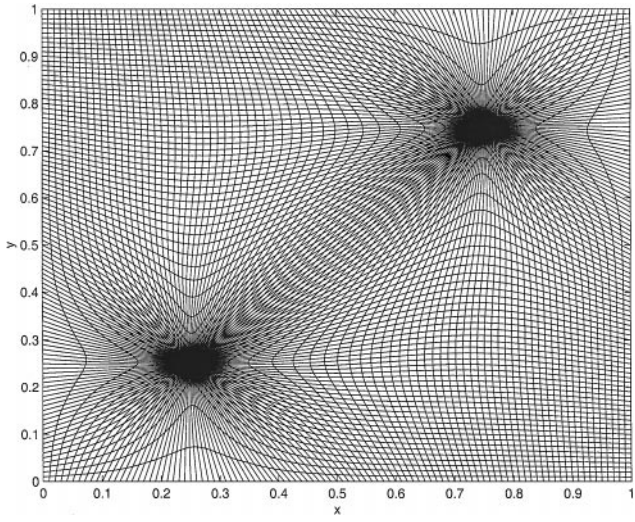

FIG. 30. Mesh in Ω_p at $t = 2.7775$ corresponding to Figs. 28 and 29 (KS).

We start with a uniform density distribution and a perturbation in the attractant field on a periodic domain $[0, 1] \times [0, 1]$:

$$\begin{cases} \rho(x, y, 0) = 1, \\ c(x, y, 0) = \sin 2\pi x \sin 2\pi y. \end{cases} \tag{6.7}$$

The ϵ is taken to be 0.01. On the computational domain, the PDEs have the form of

$$\begin{cases} \rho_t = \epsilon \Delta_B \rho - \rho \Delta_B c - \nabla' \rho \cdot \nabla' c, \\ c_t = \Delta_B c + \rho, \end{cases} \tag{6.8}$$

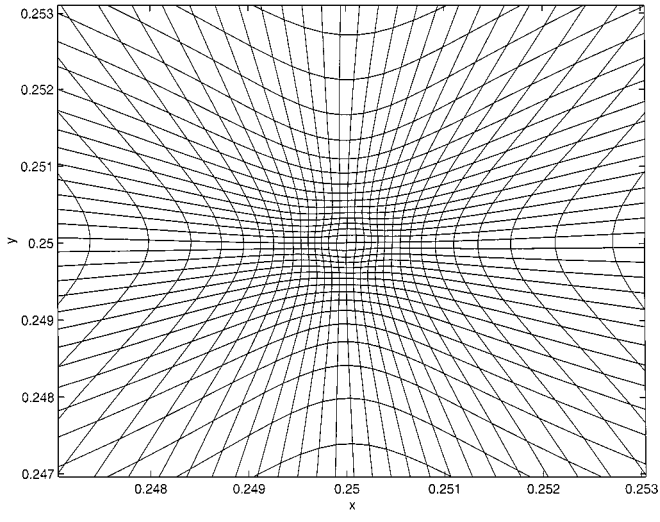

FIG. 31. Mesh in Ω_p around one blowup point at $t = 2.7775$ corresponding to Figs. 28 and 29 (KS).

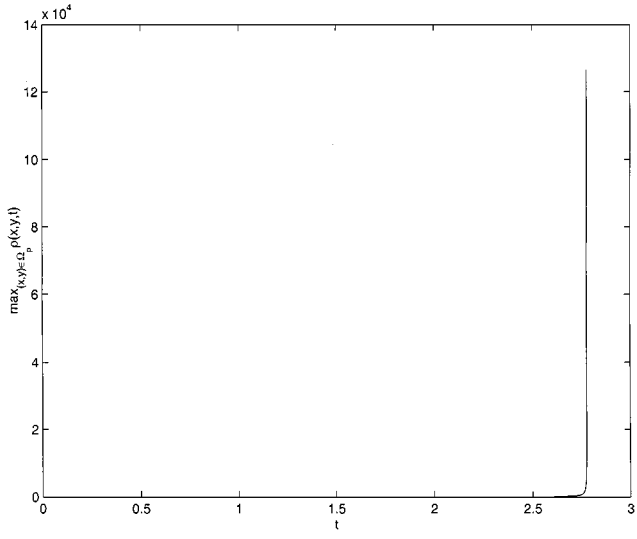

FIG. 32. $\mathrm{Max}_{(x,y)\in\Omega_p}\ \rho(x, y, t)$ versus t (KS).

where

$$\nabla' = \frac{1}{J}\begin{pmatrix} y_\eta & -y_\xi \\ -x_\eta & x_\xi \end{pmatrix}\begin{pmatrix} \frac{\partial}{\partial\xi} \\ \frac{\partial}{\partial\eta} \end{pmatrix}.$$

Equation (6.8) is solved on a uniform mesh in Ω_c with 100×100 grid points. The TOL is chosen to be 10. The monitor function is $w(\xi, \eta) = (1 + 2|\rho(\xi, \eta)|^2 + |\nabla\rho(\xi, \eta)|^2)$. The computation is continued until $t = 2.7775$ when the maximum density reaches about 10^5. About 10 remeshing iterations are conducted prior to that time. The density ρ and the attractant c are shown in the computation domain in Figs. 26 and 27 and in the physical domain in Figs. 28 and 29. Mesh distributions at the same t are shown in Figs. 30 and 31. The maximum of the density as a function of t is shown in Fig. 32. Again we see a multiple blowup solution captured by our adaptive method.

7. CONCLUSIONS

We have introduced an iterative grid redistribution method for computing singularities in multiple dimensions. The major improvement of our method over the conventional grid redistribution method is that we gain control of the mesh distribution around the singularity, which the usual mesh generation procedure based on equidistribution or variational approach is unable to achieve. Our method is also rather general: it is capable of handling multiple singularities and requires little information in advance about the locations and structure of the singularities. Moreover, it is relatively easy to implement compared to many grid refinement methods. Although our examples in this paper are in two dimensions, it is straightforward to generalize the method to the three dimensions. Many three-dimensional problems exhibit richer behavior in singularities such as line singularities and transitions from line singularities to point singularities. Our method would be more desirable in such cases. Research on the three-dimensional problems is currently being conducted.

ACKNOWLEDGMENTS

X.P.W. thanks George Papanicolaou for his constant encouragement and for some useful discussions. We thank Qiang Du, Weinan E, Gadi Fibich, and Zhouping Xin for some helpful suggestions and discussions. We also thank Michael Brenner for introducing us to the Keller–Segal model. This work is supported in part through the Research Grant Council of Hong Kong by Grant HKUST 6165/97P.

REFERENCES

1. D. A. Anderson, Grid cell volume control with an adaptive generator, *Appl. Math. Comp.* **35**, 209 (1990).
2. M. Berger and P. Colella, Local adaptive mesh refinement for shock hydrodynamics, *J. Comput. Phys.* **82**, 64 (1989).
3. J. Brackbill and J. Saltzman, Adaptive zoning for singular problems in two dimensions, *J. Comput. Phys.* **46**, 342 (1982).
4. J. Brackbill, A adaptive grid with directional control, *J. Comput. Phys.* **108**, 38 (1993).
5. M. P. Brenner, L. S. Levitov, and E. O. Budrene, Physical mechanisms for chemotactic pattern formation by bacteria, *Biophys. J.* **74**, 1677 (1998).
6. M. P. Brenner, P. Constantin, L. P. Kadanoff, A. Schenkel, and S. C. Venkataramani, *Diffusion, Attraction and Collapse*. [preprint]
7. C. De Boor, *Good Approximation by Splines with Variable Knots II* (Springer-Verlag, Berlin, 1973), Springer Lecture Notes Series 363.
8. P. K. Moore and J. E. Flaherty, *J. Comput. Phys.* **98**, 54 (1992).
9. W. Z. Huang and D. M. Sloan, A simple adaptive grid method in two dimensions, *SIAM J. Sci. Comput.* **15**(4), 776 (1994).
10. W. Huang and R. D. Russell, Moving mesh strategy based upon gradient flow equation for two dimensional problems, *SIAM J. Sci. Comput.*, in press.
11. M. Landman, G. C. Papanicolaou, P. L. Sulem, C. Sulem, and X. P. Wang, Stability of isotropic singularities for the nonlinear schrödinger equation. *Physica D* **47**, 393 (1991).
12. D. W. McLaughlin, G. Papanicolaou, C. Sulem, and P. L. Sulem, *Phys. Rev. A* **34**, 1200 (1986).
13. K. Miller and R. N. Miller, Moving finite elements I, *SIAM J. Numer. Anal.* **18**, 1019 (1981).
14. M. M. Rai and D. A. Anderson, Grid evolution in time asymptotic problems, *J. Comput. Phys.* **43**, 327 (1981).
15. V. Shvets, N. E. Kosmatov, and B. J. LeMesurier, On collapsing solutions of the nonlinear Schrödinger equation in the supercritical case, in Singularities in Fluids, Plasmas and Optics, Edited by R. Caflisch and G. Papanicolaou, NATO ASI Series Vol. 404, Kluwer Academic Publishers.
16. A. Winslow, Numerical solution of the quasi-linear Poisson equation, *J. Comput. Phys.* **1**, 149 (1996).
17. H. Zhang and M. K. Moallemi, MAGG—A multizone adaptive grid generation technique for simulation of moving and free boundary problems, *Numer. Heat Transfer* B**27**, 255 (1995).