Journal of Computational Physics 228 (2009) 2354–2365

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A grid redistribution method for singular problems

Adi Ditkowski *, Nir Gavish

School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, Israel

ARTICLE INFO

Article history:
Received 30 January 2008
Received in revised form 2 November 2008
Accepted 27 November 2008
Available online 24 December 2008

Keywords:
Iterative grid redistribution (IGR)
Nonlinear Schrödinger (NLS)
Singular problems
Grid generation

ABSTRACT

Many physical phenomena develop singular, or nearly singular behavior in localized regions, e.g. boundary layers or blowup solutions. Using uniform grids for such problems becomes computationally prohibitive as the solution approaches singularity. Ren and Wang developed a semi-static adaptive grid method [W. Ren, X.P. Wang, An iterative grid redistribution method for singular problems in multiple dimensions, J. Comput. Phys. 159 (2000) 246–273] for the solution of these problems, known as the iterative grid redistribution (IGR) method. In this study we develop a theoretical basis for semi-static adaptive grid method for singular problems. Based on this theory, we obtain the key result of this study – a methodology for designing robust weight functionals which ensures grid resolution in the singular region, as well as control of the maximal grid spacing in the outer region. Using this methodology, we introduce a semi-static adaptive grid method, which does not involve an iterative procedure for grid redistribution, as in the IGR method. We demonstrate the efficacy of this method with numerical examples of solutions which localize by more than nine orders of magnitude.

1. Introduction

Many physical phenomena develop singular or nearly singular behavior in localized regions, e.g. boundary layers or blow-up solutions. Using uniform grids for the numerical solution of these problems require extremely fine grids for accurately resolving the solution in those small regions. Hence, the use of uniform grids becomes computationally prohibitive as the solution approaches singularity.

Various methods had been developed for integration of nearly singular solutions. Among them are local grid refinement methods and adaptive grid methods. Local grid refinement methods add grid points to 'singular' regions of the solution, i.e. regions of large solution variations, such that the scheme is solved on a piecewise uniform grid (see, e.g. [2]). In contrast, adaptive grid methods maintain a fixed number of grid points, but allow them to move towards singular regions, see Fig. 1(A). In the case of time-dependent differential equations, adaptive grid methods can be roughly divided into two categories, dynamic and semi-static. For dynamic methods, or moving grid methods, grid point locations are *updated at every time step of the numerical scheme* (see, e.g. [5,6,11]). For semi-static adaptive grid methods grid point locations are held stationary as long as the solution is well-resolved and are *updated only when necessary*, i.e. only when the solution starts to become under-resolved. The first semi-static methods where developed for non-singular problems such as moving solitons or fronts, see, e.g. [16–18]. In 2000, Ren and Wang presented the first semi-static adaptive grid method [15] for singular (blow-up) problems known as the iterative grid redistribution (IGR) method.

* Corresponding author. Tel.: +972 3 640 5987.
E-mail addresses: adid@tau.ac.il (A. Ditkowski), nirgvsh@tau.ac.il (N. Gavish).
URLs: http://www.math.tau.ac.il/~adid (A. Ditkowski), http://www.tau.ac.il/~nirgvsh (N. Gavish).

0021-9991/$ - see front matter © 2008 Published by Elsevier Inc.
doi:10.1016/j.jcp.2008.11.035

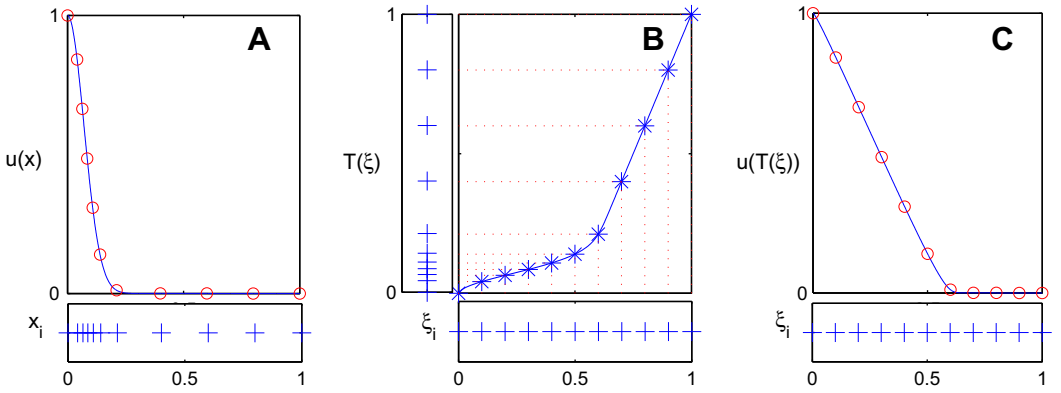

Fig. 1. Illustration of (A) $u(x)$ (solid), and the projection of u on non-uniform grid $\{x_i\}$ ($\circ$). Bottom lying graph is the location of grid points x_i. (B) Mapping $x = T(\xi)$ (solid), and the projection of T on uniform grid $\{\xi_i\}$ (*). Bottom lying graph is the location of grid points $\{\xi_i\}$ and left standing graph is the location of grid points x_i. (C) $u(T(\xi))$ (solid), and the projection of u on $\{\xi_i\}$ ($\circ$). Bottom lying graph is the location of grid points ξ_i.

In this paper, we focus on semi-static adaptive grid methods for singular problems. We consider only the one-dimensional case and focus on blowup problems characterized by a fixed or slowly moving singular region. These types of problems arise in collapsing ring solutions of the nonlinear Schrödinger equation, see [9,7], which motivated this study. The study of this case also serves as a basis for the development of similar methods in higher-dimensions and for different singular behaviors.

A key component of any adaptive method is how to find a grid distribution that is adapted to the solution. A common approach to find the new locations of the grid points is to use DeBoor's equidistribution principle [3]. According to this principle the grid points $\{x_i\}$ are distributed such that some measure $w[u]$ of the solution, denoted as the weight functional, is uniformly spread over the whole domain, i.e.

$$\int_{x_i}^{x_{i+1}} w[u(x)]dx \equiv \text{Const}, \quad w > 0. \tag{1}$$

For example, if w is chosen to be the arclength, i.e. $w[u(x)] = \sqrt{1 + (u')^2}$, the grid points $\{x_i\}$ are distributed such that the arclength of u at all intervals $[x_i, x_{i+1}]$ is equal. Therefore, the larger the gradients, the shorter the intervals. Hence, large values of $w[u(x)]$ leads to high concentration of grid points.

It turns out that equidistribution does not always ensure that the grid distribution adapts to the solution. Extensive study has been dedicated to designing robust weight functionals for which the grid distribution adapts to the solution, see, e.g. [1,20]. For the case of nearly singular solutions, Ren and Wang [15] concluded that, unless the specific solution dynamics are a priori known, equidistribution only moderately improves the grid distribution. Based on this conclusion, Ren and Wang developed the iterative grid redistribution (IGR) method. This method involves successive applications of the grid redistribution process. As a result, the moderate improvements in grid distribution achieved at each grid redistribution accumulate to a significant improvement in the overall grid distribution. This approach led to ground breaking results; it was the first time a semi-static adaptive grid method was developed and used to solve hard problems in one and two dimensions, see, e.g. [8,10]. The IGR method was later extended to the three-dimensional case [21].

However, some issues concerning the IGR method were left open. The method is sensitive to the choice of weight functional. Certain weight functionals may give excellent results for some problems, but may fail under a different setting, e.g. a change of the initial condition (see Section 5 for examples). Moreover, typically all grid points are attracted to the singular region. Therefore, the adaptivity of the grid distribution is achieved at the cost of losing grid resolution in the outer region. These issues make the method hard to implement as it requires customizing many small details for specific problems (e.g. fine-tuning of the weight functional or handling problems that occur due to grid under-resolution in the tail of the singular region).

The goal of our study is to address these issues. In Section 2, we analyze the stationary grid redistribution (SGR) process for blowup problems. A key result of this analysis is finding a design criterion for weight functionals. Using this criterion, we then show that it is possible to construct robust weight functionals such that the stationary grid redistribution (SGR) process adapts the grid distribution to the solution, even when the solution is close to singular. As a result, the iterative application of the SGR process becomes unnecessary. In Section 3, we show how to control the maximal grid spacing, so that grid adaptivity can be achieved without loss of grid resolution in the outer region. This achievement is our second key result. In Section 4, we review the steps to implement the adaptive method for a time-dependent problem. These implementation details are significantly simplified compared to the original semi-static adaptive method. For example, the grid distribution is now found by only three Matlab lines that implement simple integration, while in previous methods it required solving a nonlinear ODE. The efficiency and robustness of this method is demonstrated in Section 5 by applying our method to the nonlinear Schrödinger equation. Final remarks are given in Section 6.

2. The stationary grid redistribution (SGR) process

In this section, we analyze the grid redistribution process for finding a grid distribution for a *given* function.

As was noted, a common approach to find the grid points distribution is to use DeBoor's equidistribution principle (1). However, when using the equidistribution principle, the grid distribution may not adapt to functions which are close to singular. For example, as was demonstrated by Ren and Wang [15], in the case of the function

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

and the weight functional $w = \sqrt{1+u}$, the grid points distribution tends to a uniform distribution as $\varepsilon \to 0$, see column A of Fig. 2.

In fact, the equidistribution principle leads to failure due to an improper choice of the weight functional, which is the only parameter of (1). We now want to find a criterion for the design of the weight functional. To do so let us first distinguish between three possible cases for the behavior of the grid points distribution in the extreme case, i.e. when the function is close to singular,

(1) *Sub-adaptive case* – no grid points are attracted into the singular region in the extreme case, as in column A of Fig. 2. Clearly, this is undesired.
(2) *Over-adaptive case* – all grid points are attracted into the singular region in the extreme case, see column C of Fig. 2. Here the primary goal of concentrating grid points into the singular region is achieved at the expense of losing resolution in the outer region.
(3) *Adaptive case* – a specified portion of the grid points is mapped into the singular region and the rest of the grid points are mapped to the regular region. For example, in column B of Fig. 2 half the grid points are mapped into the singular region and half of the grid points are mapped to the outer region.

We are, therefore, interested in designing a weight functional such that the resulting grid distribution is adaptive.

2.1. Analysis – finding a criterion for weight functional design

Let us, first, note that the equidistribution principle (1) determines the discrete locations of the grid points and therefore hard to analyze. The first step of the analysis is, therefore, to find an equivalent expression for the (continuous) distribution of the grid points location which is easier to analyze. The original approach to do so was to take the limit of infinite grid points in (1). This approach leads to a nonlinear ODE for the grid points distribution [15, Eq. (2.4)], which, due to the form of its nonlinear term, is hard to analyze. We take a different approach and consider a continuous version of the equidistribution principle (1). As will be shown, an explicit expression for the (inverse of the) grid point locations naturally arises from this continuous version and leads the way for the analysis. We note that both approaches are analytically equivalent.

To consider a continuous version of the equidistribution principle, we first refer to the distribution of grid points at the discrete level as a mapping T from an initial uniform grid $\{\xi_i = \frac{i}{n}\}$ to a new grid $\{x_i\}$, where $i = 0, \ldots, n$, see Fig. 1(B). To find a continuous version of the equidistribution principle (1), we note that when the constant in (1) is determined the equidistribution principle becomes

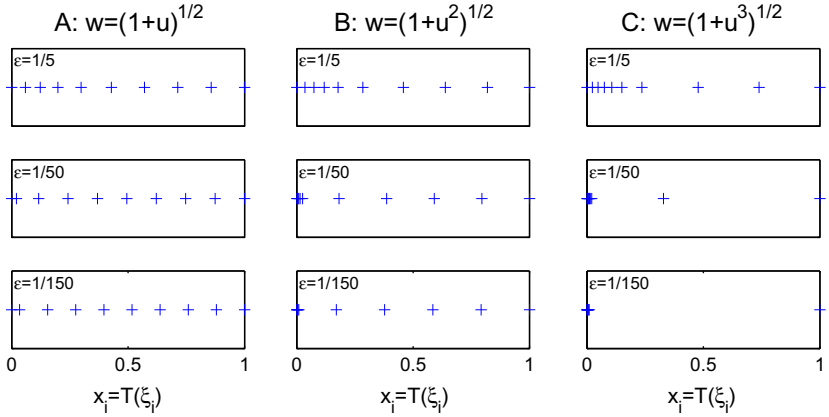

Fig. 2. Location of grid points $x_i = T(\xi_i)$ for $\xi_i = 0, 0.1, \ldots, 1$. Grid distribution was determined by the equidistribution principle (3) for function (2) with $\varepsilon = 1/5$, $1/50$ and $1/150$ (top to bottom, respectively) and for the following weight functionals – column A: $w = \sqrt{1+u}$; column B: $w = \sqrt{1+u^2}$; column C: $w = \sqrt{1+u^3}$.

$$\int_{x_i}^{x_{i+1}} w[u(x)]dx = \frac{1}{n}\int_{x_0}^{x_n} w[u(x)]dx = (\xi_{i+1}-\xi_i)\int_{x_0}^{x_n} w[u(x)]dx, \quad w>0.$$

Therefore, the continuous version of the equidistribution principle (1) for any $\xi_\alpha,\ \xi_\beta \in \Omega_c$ is

$$\int_{T(\xi_\alpha)}^{T(\xi_\beta)} w[u(x)]dx = (\xi_\beta-\xi_\alpha)\int_0^1 w[u(x)]dx, \quad w>0. \tag{3}$$

This equation implies an explicit expression for the grid points locations. Indeed, substituting $\xi_\beta = T^{-1}(x)$ and $\xi_\alpha = 0$ in (3) gives

$$T^{-1}(x) = \frac{\int_0^x w[u(s)]ds}{\int_0^1 w[u(s)]ds}. \tag{4}$$

Eq. (4) opens the way for the analysis. Here, for bravity, we assume that the singular region of u is $[0,\varepsilon]$ and that the weight functional is chosen such that its L^1 norm is bounded outside the singular region,[1] i.e.

$$\lim_{\varepsilon\to 0}\int_\varepsilon^1 w[u(s)]ds \leqslant C, \quad 0<C<\infty. \tag{5}$$

Then, by (4), the relative number of grid points that are mapped by T into the singular region of u

$$\ell_\varepsilon = \frac{|T^{-1}([0,\varepsilon])|}{|T^{-1}(\Omega_p)|} = \frac{\int_0^\varepsilon w[u(s)]ds}{\int_0^1 w[u(s)]ds}. \tag{6}$$

Eq. (6) directly implies the following proposition which defines the conditions under which the mapping T is sub-adaptive, over-adaptive or adaptive.

Proposition 1. *Let $u(x;\varepsilon):\Omega_p\to\mathbb{C}$ be a given function and let $w[u]>0$ be a weight functional. Assume that the singular region of u is $[0,\varepsilon]$ and that (5) holds. Let $T(\xi)$ be the grid mapping determined by (3). Then,*

1. *If $\lim_{\varepsilon\to 0}\int_0^\varepsilon w[u(s)]ds = 0$, then the mapping T is sub-adaptive.*
2. *If $\lim_{\varepsilon\to 0}\int_0^\varepsilon w[u(s)]ds = \infty$, then T is over-adaptive.*
3. *If $0<\lim_{\varepsilon\to 0}\int_0^\varepsilon w[u(s)]ds<\text{Const}<\infty$, then T is adaptive.*

Proof. The relative number of grid points mapped to the singular region in the extreme case ($\varepsilon\to 0$), is explicitly given by $\ell_0 = \lim_{\varepsilon\to 0}\ell_\varepsilon$. In terms of ℓ_0, the following cases are possible,

$$\begin{cases} \ell_0 = 0 & \text{sub-adaptive} \\ 0<\ell_0<1 & \text{adaptive} \\ \ell_0 = 1 & \text{over-adaptive} \end{cases}$$

The denominator of ℓ_ε is $\int_0^1 w ds = \int_0^\varepsilon w ds + \int_\varepsilon^1 w ds$ and from (5) it follows that $\lim_{\varepsilon\to 0}\int_\varepsilon^1 w ds$ is a positive constant. Hence, $\ell_0 = \lim_{\varepsilon\to 0}\int_0^\varepsilon w ds/(\int_0^\varepsilon w ds + \text{Const})$, from which the result follows. □

Proposition 1 characterizes the mapping T in terms of the weight functional and by that enables us to find a simple criterion for designing an 'adaptive' weight functional.

Corollary 2. *The mapping T is adaptive if the weight functional w scales reciprocally to the width of the singular region, i.e.*

$$w \sim \frac{1}{\varepsilon}.$$

Proof. By Proposition 1 the mapping T is adaptive if and only if $0<\lim_{\varepsilon\to 0}\int_0^\varepsilon w[u(s)]ds<\text{Const}<\infty$. This is achieved when $w\sim\frac{1}{\varepsilon}$. □

By the same reasoning, if the weight functional is asymptotically smaller than the reciprocal of the width of the singular region, i.e. $w\ll 1/\varepsilon$ then the mapping is sub-adaptive and if $w\gg 1/\varepsilon$ then the mapping is over-adaptive.

In column A of Fig. 2, $w\sim 1/\sqrt{\varepsilon}$ while the width of the singular region is $\mathcal{O}(\varepsilon)$, hence the mapping is sub-adaptive. By Corollary 2, to achieve adaptiveness for this example, we need to modify the weight functional such that it scales as $1/\varepsilon$. For example, we can choose $w=\sqrt{1+u^2}$. In this case, $\lim_{\varepsilon\to 0}\int_0^\varepsilon w[u(s)]ds\approx 0.39$ hence the mapping is adaptive, as is confirmed in column B of Fig. 2. As a final confirmation of this argument, let us consider the weight functional $w=\sqrt{1+u^3}$. In this case,

[1] As noted, these assumptions were made to allow us to present the main ideas of the analysis with minimal treatment of technical issues. Similar results, however, can be derived without these assumptions, e.g. when the singular region is inside the domain, when it is slowly moving or where there is more than one region.

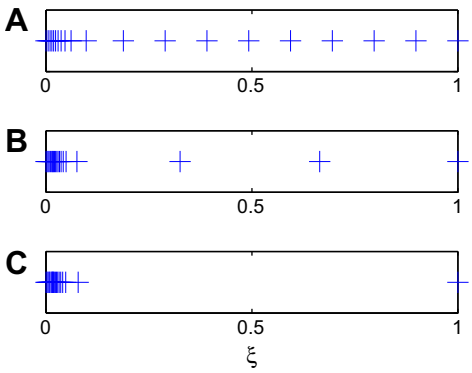

Fig. 3. Plot of $u = e^{-(x/\varepsilon)^2}$ with $\varepsilon = 0.025$ on a distributed grid determined by successive applications SGR processes with weight functional (7). (A)–(C): Result of first, second and third application of SGR process, respectively.

$w \sim 1/\varepsilon^{1.5}$, i.e. it is larger in scale than $1/\varepsilon$. Therefore, we expect the mapping to be over-adaptive, as, indeed, confirmed, in column C of Fig. 2.

We note that the design of the adaptive weight functional for column B of Fig. 2 relied not only on Corollary 2, but also on a priori knowledge of the relation between the amplitude and the width of the singular region in u. As a result the resulting weight functional is not robust. In contrast, the weight functional

$$w_{solution} = \sqrt{1 + \frac{|u'(x)|^2}{\|u\|^2} + \frac{|u''(x)|}{\|u\|}} \tag{7}$$

gives a direct measure to the gradients of the solution and does incorporate any a priori knowledge of the solution, hence it is robust. We note that the purpose of the second derivative is to avoid grid under-resolution around local extremum points of the solution where $u'(x) = 0$.

The key result of this section is a simple criterion for the design of adaptive weight functions. When using such a properly designed weight functional, a single application of the SGR process is sufficient to successfully adapt the grid distribution to the solution.

3. Semi-static adaptive grid method

In this section, we present a semi-static adaptive grid method for singular problems. As noted, the general outline of such a method is as following: Initially, the PDE is solved on a uniform grid until at some time, t_1, the solution fails to meet some smoothness criterion. Then the grid is redistributed, i.e. a mapping T_1 from the uniform computational grid $\{\xi\}$ to a non-uniform physical grid is found and applied to the solution. Note that as a result, the solution becomes a non-singular (i.e. not localized) function $u(t_1, \xi) = u(t_1, T_1(\xi))$, see Fig. 1(C). Therefore, it is possible to continue the simulation with a sufficiently smooth solution. This process is repeated such that the kth iteration is as following:

(1) The PDE is solved for $u(t, \overline{T}_k(\xi))$, where $\overline{T}_k(\xi) = T_1(T_2(\cdots T_k(\xi)))$, until at some time t_k the solution fails to meet some smoothness criterion.
(2) A mapping T_{k+1} is found such that $u(t_k, \overline{T}_k(T_{k+1}(\xi)))$ is smooth enough.
(3) Return to (1).

Note, that the analysis of the semi-static adaptive grid method is problem-dependent, as it is affected by the dynamics of u. Hence, we cannot follow the analysis of Section 2, which is for stationary problems, to analyze the semi-static adaptive grid method. However, we can study a stationary version of the semi-static adaptive grid method, which involves successive applications of the stationary grid redistribution (SGR) process, for a *given* function.[2] Following the analysis of Section 2 for this stationary problem shows that as each SGR process is applied more grid points are attracted into the singular region and that eventually all grid points are attracted into the singular region. In Fig. 3, we observe this phenomena numerically and observe that after only three iterations, almost all grid points are mapped into the singular region. In particular, we observe that grid resolution outside the singular region is lost.

[2] This stationary version arises in the IGR method.

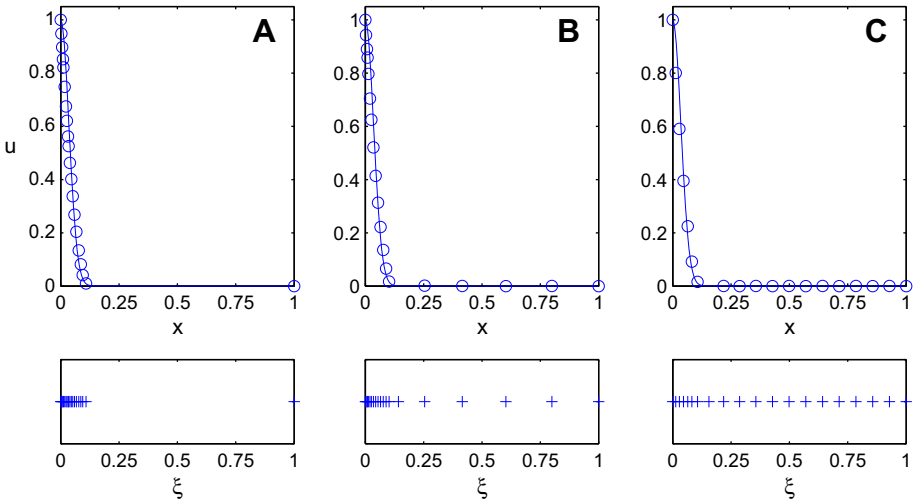

Fig. 4. Plot of $u = e^{-(x/\varepsilon)^2}$ with $\varepsilon = 0.025$ on a distributed grid. Grid was repetitively distributed six times with $\varepsilon = 0.025$ with weight functional (8). (A) $\gamma = 1$; (B) $\gamma = 0.75$; (C) $\gamma = 0.25$.

3.1. Control of grid resolution at non-singular regions

To the best of our knowledge, no semi-static adaptive grid method controls the maximal grid spacing. Our approach for doing so is as follows: We add to the weight functional a penalty $w_{penalty}$ for the low grid resolution, i.e. we increase the value of the weight functional in regions of large grid spacing. As a result, grid points are attracted into regions of high weight values which correspond to large grid spacing. The resulting weight function is

$$w[u(x)] = \gamma \frac{w_{solution}[u(x)]}{\int_0^1 w_{solution}[u(x)]dx} + (1-\gamma)\frac{w_{penalty}[T(\xi)]}{\int_0^1 w_{penalty}[T(\xi)]d\xi}, \tag{8}$$

where $w_{solution}$ is given by (7),

$$w_{penalty}[T(\xi)] = \sqrt{1 + |T'(\xi)|^2},$$

and γ is a new parameter for the grid resolution.

To demonstrate the control of grid spacing, we consider the singular function $u = e^{-(x/\varepsilon)^2}$ where $\varepsilon = 0.025$. We first find the grid mapping by repetitively redistributing the grid six times with no control of grid spacing, i.e. using the weight functional (8) with $\gamma = 1$. In this case, it can be seen that all 20 grid points are mapped into the singular region, see Fig. 4(A). To control the grid spacing, we set $\gamma = 0.75$ and show that for this value approximately 15 out of the 20 points are mapped into the singular region, see Fig. 4(B). Finally, we set $\gamma = 0.25$ and show that 6 out of the 20 grid points are mapped into the singular region, see Fig. 4(C). Therefore, the weight functional (8) provides a simple and efficient way to avoid loss of grid resolution in the outer region. Moreover, it provides approximate control of the spread of grid points between the singular region and the outer region using the parameter γ.

4. Simple steps to the implementation of the SGR method

We now describe the steps to implement the SGR method for the numerical computation of time evolution problems of the form

$$u_t(x) = f(t, x, u, u_x, u_{xx}), \tag{9}$$

where x is the physical variable.

The implementation consists of three simple steps:

(1) Transform the PDE for $u(x)$ to a PDE for $u(\xi)$.
(2) Add a monitoring mechanism for the smoothness of the solution to determine when grid redistribution is needed.
(3) Incorporate the SGR process into the scheme:
   (a) Find a new grid mapping T when necessary.
   (b) Project the solution on the new grid and update operators.

Note that the difference between the implementation of the SGR method and other adaptive method is only in step (3a). However, to make it easier for the reader we also briefly describe steps (1), (2) and (3b) and refer the reader to Ceniceros and Hou [6] and Ren and Wang [15] for more details.

Note that this section includes examples of Matlab code. In this context, we use the notation (x,u) to refer to vectors x and u that represent the function $u(x)$.

4.1. Step (1) – transforming the PDE for $u(x)$

Transforming the PDE for $u(x)$ to a PDE for $u(\xi)$ using the mapping $x = T(\xi)$ is by expressing PDE (9) in terms of ξ, i.e. derivatives of x are expressed as derivatives of ξ. For example, $u_x = u_\xi \xi_T = u_\xi / T_\xi$. Using this procedure, the PDE (9) becomes

$$u_t(\xi) = u_t(T(\xi)) = f\left(t, T(\xi), u(T(\xi)), \frac{u_\xi}{T_\xi}, \frac{u_{\xi\xi}}{T_\xi^2} - \frac{u_\xi T_{\xi\xi}}{T_\xi^3}\right). \tag{10}$$

4.2. Step (2) – add a monitoring mechanism for the solution smoothness

As noted in Section 3, grid redistribution is applied when the solution fails to meet some smoothness criterion.

We use the smoothness criterion $\|u'(\xi)\|_\infty < TOL$. Note that often $u'(\xi)$ is already calculated for solving the PDE, hence the monitoring overhead is minimal.

4.3. Step (3a) – finding the grid distribution

To find the grid mapping T for a given function u and for a chosen weight functional $w[u]$, we first calculate $T^{-1}(x)$ from (4) and then find its inverse. As noted in Section 2, this method is easier than finding T directly, i.e. without first finding T^{-1}, as done in existing methods [15], since it does not involve solving a nonlinear ODE.

A possible Matlab code for calculating the inverse mapping (x, invT) for a given weight functional (x,w) is

```
0001 invT = cumsum(w) − w(1);
0002 invT = invT/invT(end);
```

This code implements the integration in (4) using the rectangular rule. Note, that there is no need to apply a high order rule for the numerical integration of $\int_0^x w(s)ds$, since the exact location of each grid point is less important as long as the grid points concentrate in singular regions.

To find the inverse of T^{-1}, we note that, from a computational point of view, the inverse of $T^{-1} = (x, invT)$ is simply $T = (invT, x)$. However, (invT,x) is defined on a non-uniform grid invT. It is more convenient to define $T = (invT, x)$ on the uniform grid ξ, to do so, we interpolate (invT,x),

```
0003 T = interpl(invT,x,xi,'pchip');
```

We use the pchip option which ensures that the interpolation is monotonic on the expense of its smoothness and accuracy. The reason is that it is vital for T to be monotonic, i.e. that the order of grid points is kept, while it is less important to calculate T very accurately.

4.4. Step (3b) – after the grid distribution is found

After the SGR process is applied and a new mapping T is found, it is necessary to calculate the solution[3] on the new computational grid, i.e. to find $u(\xi) = u(T(\xi))$. A possible Matlab command for doing so by interpolation is

```
0004 u = interpl(xi,u,T,'spline');
```

In addition, it is highly recommended to calculate and store the accumulated mapping $\overline{T}_k$ after the grid redistribution. This is required in order to recover the solution $u(x)$ on the physical domain. A possible Matlab command for calculating $\overline{T}_{k+1}(\xi) = \overline{T}_k(T(\xi))$ is

```
0005 accumulatedT = interpl(xi,accumulatedT,T,'pchip');
```

[3] In case the numerical integration depends on the solution in previous time steps (e.g. leap frog or predictor–corrector schemes), the solution in those previous times should also be projected to the new grid.

The physical solution is therefore represented by the vectors (`accumulatedT,u`).

Finally, it is necessary to transform the PDE to the new computational' coordinates, i.e. find the PDE for $u(\xi) = u(\overline{T}(\xi))$ where $\overline{T}$ is the new accumulated mapping, see Section 4.1 for more details.

4.5. Additional considerations for implementation – noise

Any numerical scheme introduces numerical noise to the solution of the scheme. Since noisy regions are characterized by large derivatives, the grid redistribution process tends to move grid points toward noisy regions instead of moving them toward singular regions of the solution. Therefore, the grid redistribution process becomes less efficient, or may even fail, in the presence of noise.

To minimize the phenomenon of grid points concentration in noisy regions, it is possible to construct the weight functional based on a smoothed solution. That is to calculate $w = w[u_{smoothed}]$ instead of calculating $w = w[u]$. After the new grid transformation is calculated, the simulation is continued with the original non-smoothed solution. In this way, the grid redistribution process is 'blind' to noise in the solution.

5. Example – NLS equation

We now demonstrate the efficiency and robustness of the stationary grid redistribution (SGR) method, described in Section 3, and compare it to the iterative grid redistribution (IGR) method of Ren and Wang [15]. To do so, we solve the radially symmetric nonlinear Schrödinger equation (NLS),

$$iu_t(t,r) + \Delta u + |u|^{2\sigma}u = 0, \quad \Delta = \partial_{rr} + \frac{1}{r}\partial_r, \quad \sigma \geqslant 1, \tag{11a}$$

in the domain $r \in \Omega_p = [0, r_{bound}]$ and $t > 0$ with boundary conditions

$$u(0,r) = u_0(r), \quad u_r(t,0) = 0, \quad u(t,r)|_{r=r_{bound}} = 0. \tag{11b}$$

It is well-known that the NLS admits blowup solutions, i.e. solutions that become singular in a finite time (see [19] for a review). Various methods had been developed for solving the equation very close to blowup time. The dynamic rescaling method [13] with approximate boundary conditions [12] has been a very successful method for solving the blowup solutions of the radially symmetric NLS. This method exploits the known self-similar structure of the collapsing part of the solution near the singularity, which is of the form

$$|u| \sim \frac{1}{\varepsilon^{\frac{1}{\sigma}}} R\left(\frac{r}{\varepsilon}\right), \quad 0 < \varepsilon \ll 1.$$

In particular, the method relates the shrinking transverse width of the solution to the increase in its norm [13]. Therefore, the solution is computed on a fixed computational grid, which in physical space corresponds to a solution on a domain that shrinks uniformly toward the singularity. More general methods, which do not assume an a priori knowledge of the asymptotic behavior of the solution are the adaptive methods. A moving (dynamic) adaptive mesh method was developed by Budd et al. [5]. Later the semi-static iterative grid redistribution (IGR) method was introduced in [15] and further improved in [10,21]. Both methods were successfully used to solve the NLS with various blowup behaviors [4,8–10] as well as cases of multiple blowup points [15].

The computational domain Ω_c is chosen to be $[0,1]$. Let $\overline{T} = r_{bound} \cdot T(\xi)$ be a transformation from the computational domain Ω_c to the physical domain $\Omega_p = [0, r_{bound}]$. Then, the NLS in the computational domain becomes

$$iu_t(t,\xi) + \Delta_\xi u + |u|^{2\sigma}u = 0, \quad \Delta_\xi = \frac{1}{r_{bound}^2 \cdot T_\xi^2}\partial_{\xi\xi} + \frac{1}{r_{bound}^2}\left[\frac{1}{TT_\xi} - \frac{T_{\xi\xi}}{T_\xi^3}\right]\partial_\xi. \tag{12}$$

The transformed NLS (12) is solved using a standard Crank–Nicholson predictor–corrector scheme on a uniform grid with a variable time step. Based on the rescaled time variable used in the method of dynamic rescaling,

$$\tau = \int_0^t \frac{ds}{\varepsilon^2(s)}, \quad \varepsilon = 1/\|u\|_\infty^{2\sigma},$$

the variable time step is set to be $dt = \varepsilon^2 d\tau$, where $d\tau$ is a fixed time step. We note that in the examples presented in this section, ε also serves a measure for the width of the singular region of the solution.

The weight functional is taken to be (8) where, unless otherwise specified, the parameter γ of the weight functional (8) is set to 0.6. For this choice of γ, roughly 60% of the grid points are expected to be attracted to the singular region. The smoothness criterion is set so that the maximum amplitude of the gradient of u is smaller than some tolerance, i.e. $\|u_\xi\|_\infty < TOL$. In addition, unless otherwise specified, the number of grid points in the computational grid is taken to be 500.

We first use the SGR method to solve the NLS (11) in the domain $\Omega_p = [0,1]$ with the initial condition

$$u_0 = 16e^{-(4r)^2}. \tag{13}$$

The solution in this case maintains a peak-type (Townes) profile, i.e. it attains its global maximum at the origin, and becomes narrower in width and higher in amplitude as it collapses, see Fig. 5(A)–(C). In Fig. 5(D), we plot the absolute value of the solution after it focused by nine orders of magnitude. It can be seen that the solution is extremely localized around the origin. However, the solution is still well-resolved in the singular region which size is approximately 10^{-9}, see Fig. 5(E). We note that approximately 65% are attracted to the singular region $[0, 10^{-9}]$ and the rest of the grid points are uniformly spread in outer region. As expected, the solution profile is in excellent agreement with the asymptotic (Townes) profile of the NLS, see Fig. 6.

In Fig. 7, we plot the smoothness criterion $\|u_\xi(t,\cdot)\|_\infty$ as a function of time. The sharp falls in $\|u_\xi\|_\infty$ are due to grid redistributions. It can be seen that $\|u_\xi(t,\cdot)\|_\infty < TOL = 15$ during all the simulation. Initially, the solution is smooth, but as it becomes localized, its gradient increases until it reaches the tolerance value. At this point, the grid is redistributed, and $\|u_\xi\|_\infty$ decreases by a factor of ≈ 3.

To show that our numerical results are insensitive to the choice of the smoothness criterion, we also run the code with the smoothness criterion $\|w[u]\|_\infty < 30$. The solution of this NLS simulation is in agreement with the solution obtained using the smoothness criterion $\|u_\xi\|_\infty < TOL = 15$, see Fig. 6.

To check for convergence, we also run the code with 50 grid points and observe that the solution profiles are in agreement, see Fig. 6. Note that even with such a small number of grid points, the solution is well-resolved in all the domain after it focused by a factor of 10^9, see Fig. 8.

Our computations were preformed on an Intel Core2 1.86 GHz PC running Matlab 2007a. The elapsed time for the simulations with 500 grid points with the smoothness criterion $\|u_\xi\|_\infty < TOL$ was only 9.2 s. The time for the simulation using the smoothness criterion $\|w[u]\|_\infty < TOL_w$ increases by 16% to 10.67 s. The simulations required 12 grid redistributions so that the 'overhead' of grid redistribution, i.e. the cpu time used by the SGR process, is less than 2% of the total cpu time.

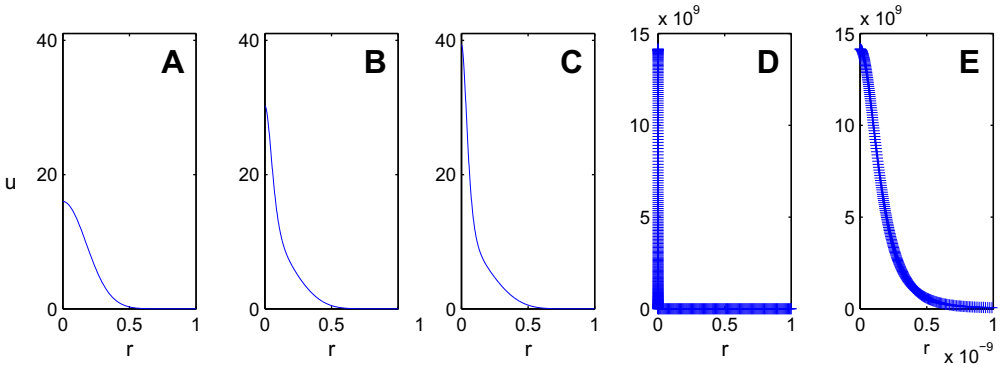

Fig. 5. Solution of NLS with initial condition u_0 given by (13) at times (A) $t = 0$, (B) $t = 0.1$, (C) $t = 0.12$ and (D) $t = 0.145$ ($\varepsilon \approx 1.12 \times 10 - 9$), markers are the projection of grid points. (E) Same data as in D in singular region. The simulation is conducted with 500 grid points using the SGR method.

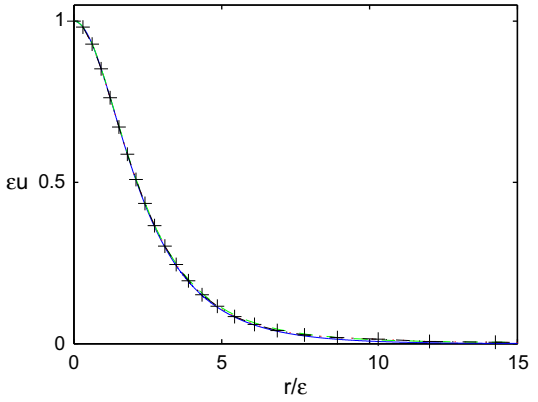

Fig. 6. Rescaled solutions $\varepsilon u(r/\varepsilon)$ at the ending time of the simulation of Fig. 5 with smoothness criterion $\|u_\xi\|_\infty \leqslant 15$ with 500 grid points (dotted). A simulation with smoothness criterion $\|w[u]\|_\infty \leqslant 30$ with 500 grid points (dashed), and a simulation with smoothness criterion $\|u_\xi\|_\infty \leqslant 15$ with 50 grid (+markers). Solid curve is the rescaled Townes profile. The four curves are nearly indistinguishable.

We now consider an example of a moving singularity region. To do so we solve the NLS (11) with the initial condition [8, Eq. (28)]

$$u_0 = G(r)e^{-i\frac{\alpha^2}{8}r^2}, \quad \alpha = 0.357, \tag{14}$$

where $G(r)$ is a ring profile, approximated [8] by $\mathrm{sech}(r - r_{max})$, i.e. it attains its global maximum at some point $r_{max} > 0.$, see Fig. 9(A). Fig. 9(A)–(C) shows the initial stages of the ring collapse. In particular, the singular region of the solution is located around the ring peak $r_{max}(t)$ which moves as the ring radius changes. We now continue this simulation for higher focusing levels using the SGR method. As expected, even at a focusing level of 10^{10} the solution is well-resolved in both singular and outer regions, see Fig. 9(D) and (E). An overall of 85 grid redistributions were applied during the simulation. Note, that the only adjustment of the SGR method made for this simulation, is to increase the tolerance of the smoothness criterion from 15 to 35. This is done to reduce the frequency of the grid redistributions, which is relatively high due to the movement of the singular region.

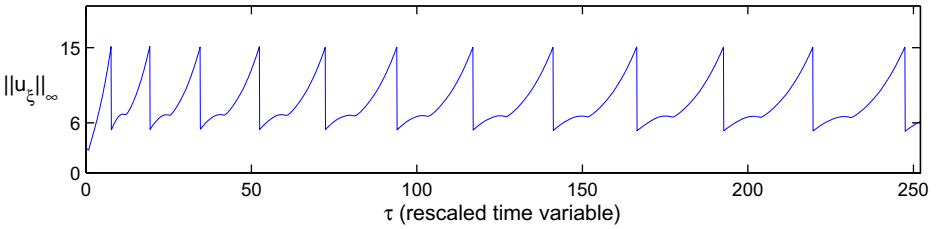

Fig. 7. Smoothness criterion as function of rescaled time variable τ for simulation with initial condition (13) and for the cases A: Smoothness criterion $\|u_\xi\|_\infty \leqslant 15$ with 500 grid points.

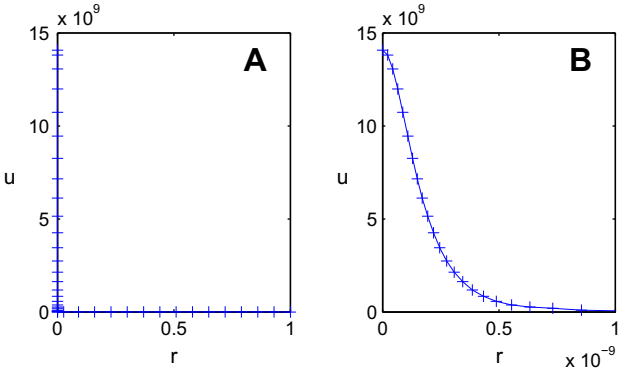

Fig. 8. Solution of NLS with initial condition u_0 given by (13) at time $t = 0.145$ $(\varepsilon \approx 1.12 \times 10^{-9})$. Simulation is conducted with 50 grid points using the SGR method. (A) Physical solution $u(r)$. (B) Same data as in A in singular region.

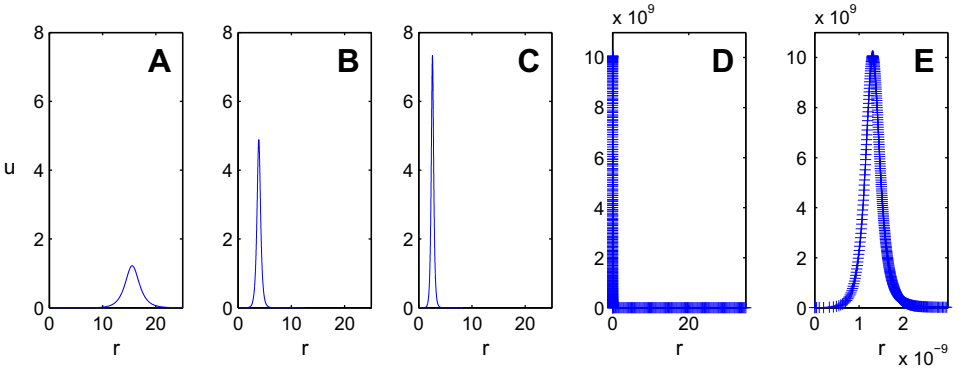

Fig. 9. Solution of NLS with initial condition u_0 given by (14) at times (A) $t = 0$, (B) $t = 0.58$, (C) $t = 7.52$ and (D) $t = 7.82$ $(\varepsilon \approx 9.9 \times 10^{-11})$. (E) Same data as in D in singular region.

Finally, we demonstrate the ability of our method to solve problems with multiple singular regions. To do so, we solve the NLS (11) with $\sigma = 2$ in the domain $[0,1]$, and with the initial condition

$$u_0 = P(r-0.5;0.01) + P(r-0.75;0.088), \quad P(r;\varepsilon) = \sqrt[4]{3}\,\mathrm{sech}^{\frac{1}{2}}(2r/\varepsilon)/\sqrt{\varepsilon}. \tag{15}$$

The solution in this case collapses as two standing rings [14], i.e. each of the rings becomes narrower in width and higher in amplitude, while their radii remains fixed, see Fig. 10.

In Fig. 11(A), we plot the absolute value of the solution after it focused (i.e. localized) by nine orders of magnitude, it can be seen that the solution is localized around two points, $r^{(1)}_{max} \approx 0.49$ and $r^{(2)}_{max} \approx 0.72$. Note that the inner ring collapsed slightly faster than the outer ring. In Figs. 11(B) and 11(C), we plot $|u(x)|$ around each of the ring peaks and show that the solution is well-resolved in those regions. To the best of our knowledge, it is the first time a double standing ring collapse in the NLS is numerically observed.

In this section, we presented three different simulations which were successfully conducted using the SGR method. The only adjustment of the SGR method in the different cases was in the tolerance of smoothness criteria.

We also ran these simulations using the IGR method. In this case, each problem required adjustment of the weight functional, and even for successful choices of weight functionals, grid resolution in the outer region was lost. For example, the following, slightly over-adaptive, weight functional

$$w_1 = \sqrt{1 + 1.5\frac{|u'|^2}{\|u\|^2_\infty} + 0.1\frac{|u''|^2}{\|u\|^2_\infty}},$$

is used in NLS simulations [22]. Using it for the simulation of the collapsing Townes profile, see Fig. 5, was successful but resulted in lose of grid resolution in the outer region (data not shown). However, using this same weight functional for the simulation of the collapsing ring, see Fig. 9, lead to failure of the simulation after a focusing factor of ~300 (data not shown).

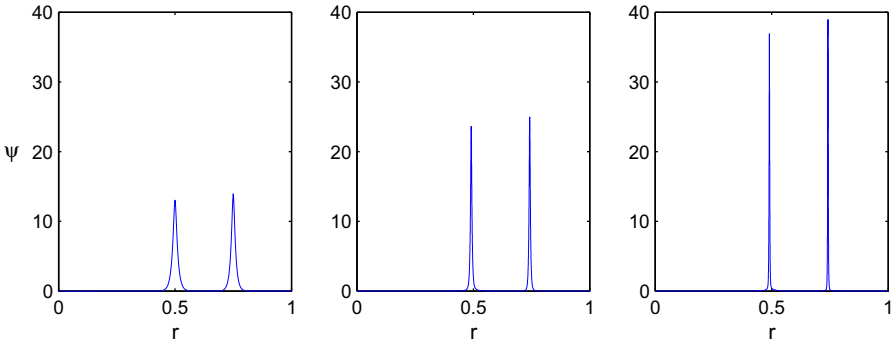

Fig. 10. Solution of NLS with initial condition u_0 given by (15) at times (left to right): $t = 0$, $t = 6.46 \times 10^{-4}$ and $t = 6.59 \times 10^{-4}$.

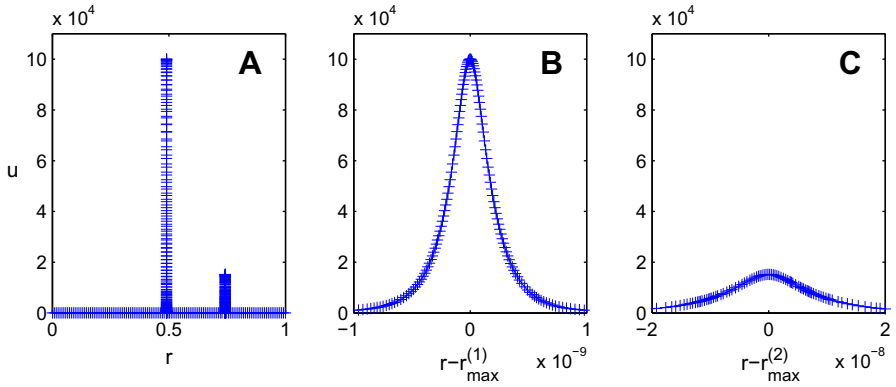

Fig. 11. Solution of NLS with initial condition u_0 given by (15) at time $t = 6.62 \times 10^{-4}$ ($\varepsilon \approx 9.9 \times 10^{-11}$). (A) Physical solution $u(r)$. (B) Same data as in A in singular region around $r^{(1)}_{max} \approx 0.49$. (C) Same data as in A in singular region around $r^{(2)}_{max} \approx 0.72$.

6. Concluding remarks

We have introduced a semi-static adaptive grid redistribution method for solving singular problems in one dimension, which we denote as the stationary grid redistribution (SGR) method. This method provides a solution for some of the open issues concerned with the existing iterative grid redistribution (IGR) method [15]. In particular, we provide a theoretical basis for such semi-static adaptive grid redistribution methods. The key result of this study is developing a methodology for designing robust weight functionals which ensures grid resolution in the singular region, as well as control of the maximal grid spacing in the outer region. This is demonstrated by numerical simulations of the NLS which include various scenarios such as a moving singularity region and multiple singular regions. All simulations were successfully solved by the SGR method, in the sense that the solution remained well-resolved during the simulation, and that grid resolution was also maintained in the outer region. In particular, those different simulations were solved with no need to change the numerical parameters (e.g. weight functional) of the SGR method, demonstrating the robustness of the method.

Much of this work was devoted to the analytical study of grid redistribution for a *given* function and the way to incorporate it into an *existing* scheme. Since the results of this study are independent of the specific scheme used, we did not consider the parameters of the scheme used in our simulations. e.g. order of convergence and stability.

We have considered only the 'simple' one-dimensional case and limited ourselves to blowup problems in which the singular region does not move rapidly. As this study shows this case is far from trivial and allows us to examine the fundamental issues of semi-static adaptive methods. A thorough understanding of these issues is essential for developing adaptive methods for problems in multiple dimensions or for other types of singularities.

We note that the SGR method can also be used to solve non-singular problems, but with a very large domain, e.g. long-time simulations of moving solitons. For these problems, the solution is localized with respect to the size of the domain, hence it cannot be efficiently solved on a uniform grid. In such cases, the control of maximal grid spacing in the outer region becomes crucial.

Acknowledgments

Special thanks to Xiao-Ping Wang who introduced us to this subject, for sharing his insights and for his guidance in the implementation of the IGR method. We thank Gadi Fibich and Yonatan Sivan for useful discussions. The research of Adi Ditkowski was supported by the Israel Science Foundation (grant no. 1364/04), and the United States–Israel Binational Science Foundation (grant no. 2004-099). The research of Nir Gavish was partially supported by grant number 123/08 from the Israel Science Foundation (ISF) and by the Israel Ministry of Science Culture and Sports.

References

[1] G. Beckett, J.A. Mackenzie, A. Ramage, D.M. Sloan, On the numerical solution of one-dimensional pdes using adaptive methods based on equidistribution, J. Comput. Phys. 167 (2001) 372–392.
[2] M.J. Berger, P. Collela, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1989) 64–84.
[3] C. De Boor, Good Approximation by Splines with Variable Knots II (Springer Lecture Notes Series 363), Springer-Verlag, Berlin, 1973.
[4] C.J. Budd, S. Chen, R.D. Russell, New self-similar solutions of the nonlinear Schrödinger equation with moving mesh computations, J. Comput. Phys. 152 (1999) 756.
[5] C.J. Budd, W. Huang, R.D. Russell, Moving mesh methods for problems with blow-up, SIAM J. Sci. Comput. 17 (1996) 305.
[6] H.D. Ceniceros, T.Y. Hou, An efficient dynamically adaptive mesh for potentially singular solutions, J. Comput. Phys. 172 (2001) 609639.
[7] G. Fibich, N. Gavish, Theory of singular vortex solutions of the nonlinear Schrödinger equation, Physica D (2008).
[8] G. Fibich, N. Gavish, X.P. Wang, New singular solutions of the nonlinear Schrödinger equation, Physica D 211 (2005) 193–220.
[9] G. Fibich, N. Gavish, X.P. Wang, Singular ring solutions of critical and supercritical nonlinear Schrödinger equations, Physica D 231 (2007) 55–86.
[10] G. Fibich, W. Ren, X.P. Wang, Numerical simulations of self focusing of ultrafast laser pulses, Phys. Rev. E 67 (2003) 056603.
[11] W. Huang, Y. Ren, R.D. Russell, Moving mesh methods based on moving mesh partial differential equations, SIAM J. Comput. Phys. 113 (1994) 279–290.
[12] N.E. Kosmatov, V.F. Shvets, V.E. Zakharov, Computer simulation of wave collapses in the nonlinear Schrödinger equation, Physica D 52 (1991) 16–35.
[13] D.W. McLaughlin, G.C. Papanicolaou, C. Sulem, P.L. Sulem, Focusing singularity of the cubic Schrödinger equation, Phys. Rev. A 34 (1986) 1200–1210.
[14] P. Raphael, Existence and stability of a solution blowing up on a sphere for a L^2 supercritical nonlinear Schrödinger equation, Duke Math. J. 134 (2) (2006) 199–258.
[15] W. Ren, X.P. Wang, An iterative grid redistribution method for singular problems in multiple dimensions, J. Comput. Phys. 159 (2000) 246–273.
[16] J. M Sanz-Serna, I. Christie, A simple adaptive technique for nonlinear wave problems, J. Comput. Phys. 128 (1996) 274–288.
[17] P. Saucez, A. Vande Wouwer, W.E. Schiesser, Some observations on a static spatial remeshing method based on equidistribution principles, J. Comput. Phys. 128 (1996) 274–288.
[18] P. Saucez, A. Vande Wouwer, W.E. Schiesser, An adaptive method of lines solution of the Korteweg–de Vries equation, Comput. Math. Appl. 35 (1998) 13–25.
[19] C. Sulem, P.L. Sulem, The Nonlinear Schrödinger Equation, Springer, New York, 1999.
[20] A. van Dam, P.A. Zegeling, A robust moving mesh finite volume method applied to 1D hyperbolic conservation laws from magnetohydrodynamics, J. Comput. Phys. 216 (2006) 526–546.
[21] D. Wang, X.P. Wang, A three-dimensional adaptive method based on the iterative grid redistribution, J. Comput. Phys. 199 (2004) 423–436.
[22] X.P. Wang, Personal communications.