Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Journal of Computational Physics 193 (2003) 159–179

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Variational grid adaptation based on the minimization of local truncation error: time-independent problems

Giovanni Lapenta [*]

Theoretical Division, Los Alamos National Laboratory, Los Alamos, NM 87544, USA

Received 27 September 2002; received in revised form 13 August 2003; accepted 13 August 2003

Abstract

A new approach to grid adaptation is presented. The method is based on two established foundations. First, the method is based upon variational grid adaptation, retaining all the well-known properties of robustness and regularity. Second, the adaptation method presented here is based on a general definition of the error detector obtained from the moving finite element (MFE) method. The error detector is general, applicable to any given problem, and does not require any a priori knowledge of the solution or of the physical behaviour of the system under investigation. The primary theoretical contribution of the present work is in establishing a link between various adaptation methods previously regarded as different and unrelated. We show that they all derive from the same approach and are all equivalent in the sense that the same grid is generated by all of them for the same problem, once the monitor functions are chosen according to our approach. The primary practical contribution of the present work is in prescribing a rigorous monitor function for previously published adaptation strategies. The choice proposed here is shown to outperform previous heuristic choices. The method is tested in a series of elliptic problems, where the adaptation strategy presented here can improve the accuracy by orders of magnitude.

1. Introduction

Grid adaptation is a powerful strategy for increasing the efficiency of numerical schemes by focusing the computational effort where needed. A reliable and useful adaptation strategy must be founded on two ingredients.

First, grid adaptation must be based on a robust algorithm with a proven ability to avoid singularities (e.g., folded grids). One such method is the variational grid adaptation method [5] that is based on the solution of elliptic Euler–Lagrange equations. Based on properties of harmonic functions, it has

[*] Tel.: +1-505-667-5493; fax: +1-505-665-3107.
E-mail address: lapenta@lanl.gov (G. Lapenta).

0021-9991/$ - see front matter © 2003 Elsevier B.V. All rights reserved.
doi:10.1016/j.jcp.2003.08.004

been shown that under general conditions, elliptic grid adaptation methods never produce folded grids [6]. Note that this conclusion is true in the continuum and does not always hold once the equations are discretized.

Second, the strategy to adapt grids must be based on reliable detectors to locate the regions where finer grid spacing is required. A desirable property in this regard is that such detectors be automatic and not require any user intervention. In a seminal paper by Dukowicz [8], it was suggested that the grid adaptation strategy typical of the moving finite element (MFE) method [2] could be isolated from the rest of the MFE approach to create a stand alone grid generation tool. Inspired by this suggestion, we have attempted to use the error detector typical of the MFE method, within the variational grid adaptation method.

The result of this attempt is a new variational grid generation method based on the now classical approach by Brackbill and Saltzmann [5] and on a MFE-like computation of the local truncation error. Typically, variational grid adaptation methods rely on monitor functions to guide the adaptation strategy. Such monitor functions are usually chosen heuristically, e.g., as the gradient of a field known to the user to be of particular physical importance for the specific case under investigation. The novelty of the present work resides in replacing heuristic definitions of the monitor functions with the more rigorous error detector provided by the MFE method. A direct link between error minimization and grid adaptation has been proposed before as potentially the optimal approach to grid generation [7,10], but a practical way of implementing it in general is still needed. The approach proposed here can be beneficial in the direction of finding such an approach.

Four contributions are produced by our approach.

First, the approach followed here shows that a link can be established between error minimization and error equidistribution. This result is of theoretical importance since generally it is easier to devise methods to equidistribute the error than it is to directly minimize it.

Second, our approach establishes a rigorous method to adapt computational grids. We reach the condition where the computational error is minimum for a given grid topology. It is shown in practice that previous methods based on heuristic assumptions can fail when our method performs well. It will be shown that the improvement in accuracy is of orders of magnitude and is obtained at no additional cost over previously published and commonly used methods.

Third, we show that a number of previous variational grid adaptation methods can be derived from our theoretical approach as specific examples of its formulation. This ability to summarize within a single formalism methods previously regarded as unrelated not only is of theoretical relevance, but also has a very desirable practical consequence. Our approach can teach what monitor function ought to be chosen in methods previously published to obtain the optimal result. Indeed, all previous methods give the same optimal answer provided that the monitor function is chosen according to our new approach.

Four, a practical way to compute the local truncation error is derived and applied in practice. Although based on ideas typical of the MFE literature, the specific implementation proposed here and its formulation in general are not known to the author to have been published before.

The work is organized as follows.

Section 2 provides the theoretical foundations of variational grid adaptation based on the minimization of the local truncation error. Section 2 is organized into three subsections. Section 2.1 presents the theoretical derivations in 1D to allow a more intuitive understanding. Section 2.2 generalizes the derivation to any number of dimensions and shows how the present formalism can be used to obtain published methods previously though of as independent. Section 2.3 describes the implementation of the grid adaptation methods. Section 3 describes how the truncation error, the main ingredient required to adapt the grid, can be computed in practice. Section 4 summarizes the algorithm used in practice to adapt the grid. Section 5 presents the application of our grid adaptation method to paradigmatic elliptic problems. Conclusions are drawn in Section 6.

2. Adaptation strategy

In the present work we are concerned with strategies to adapt general 3D structured grids. We will adopt a common dimensional-independent notation. Our grid points will be labeled with just one latin letter index, $\mathbf{x}_i$; each cell i has a volume V_i.

The key goal of our method is to define a strategy for locating the positions of the grid points in order to solve a time-independent problem with the minimum error for a given grid topology.

The first point to clarify is the definition of the error to be used to base the grid adaptation strategy upon. Should we use the local truncation error or the actual local error in the solution (i.e., the global truncation error)?

A recent paper [14] has analyzed the issue, considering grid adaptation toward the two errors mentioned above. The answer is clear: the local truncation error is the source of error which then propagates throughout the domain to produce the global truncation error in the solution. Adapting to the local truncation error removes the source of error resulting in the most effective adaptation strategy. Adapting to the solution error (global truncation error) is ineffective as it acts on the effect not on the cause.

Given the discussion above, the task becomes clear. We need to choose the grid points $\mathbf{x}_i$ so that the norm of the local truncation error is minimized. In the present section we shall describe how, given a local truncation error, a grid can be generated to adapt to it using the tools provided by variational grid adaptation. In the next section we shall consider the practical issue of how to compute the local truncation error on a given grid.

The present section is organized in three parts. First, we will consider grid adaptation in a simple 1D geometry where the derivations are simpler and immediately understandable. Second, we will generalize the derivation to any number of dimensions. Third, we will discuss practical issues of implementation.

The 1D section serves as a learning and understanding tool that facilitates the derivations in ND.

2.1. 1D derivation

In 1D, the problem of minimization can be stated simply, starting from the usual assumption that the numerical method used to discretize the equation under consideration is of order m. Based on this hypothesis, the leading term of the local truncation error can be expressed as

$$e = \psi\Delta^m, \tag{1}$$

where the local grid spacing is Δ and the factor ψ depends on the local value of the solution of the problem under consideration (typically ψ is a derivative of some order).

The statement of error minimization in L_1-norm in a 1D domain extended from 0 to L can be written as

$$\int_0^L |e|\,\mathrm{d}x. \tag{2}$$

We will use the L_1-norm definition since it leads to the most convenient derivation. The conclusions reached depend on this choice but the general procedure can be extended to some other norms (e.g., L_2-norm) but not to all (e.g., L_∞).

We attack the task of determining the grid by using the variational approach and following ideas outlined in Chapters 7 and 8 of [10]. To this end, two coordinate systems are introduced: the physical coordinate system x and the logical coordinate system ξ that maps the grid on the physical domain to a uniform and unitary grid in the logical domain [10]. The domain $x \in [0, L]$ is mapped in the domain $\xi \in [0, N]$ where N is the number of cells. The task of defining the grid is then restated into defining the mapping from the logical coordinate to the physical coordinate. We recall that the grid spacing can be written in terms of the mapping as

$$\Delta = \frac{\mathrm{d}x}{\mathrm{d}\xi}, \tag{3}$$

where the derivatives in 1D are total, not partial, being present only one coordinate. Using this notation, the statement of error minimization (2) becomes

$$\int_0^L |\psi| \left| \frac{\mathrm{d}x}{\mathrm{d}\xi} \right|^m \mathrm{d}x, \tag{4}$$

where the expression (1) for the local truncation error has been substituted.

In order to establish a notation easy to extend to any number of dimensions, we rewrite Eq. (4) using the metric tensor. In 1D the metric tensor reduces to a simple scalar

$$g \equiv g_{11} = \left| \frac{\mathrm{d}x}{\mathrm{d}\xi} \right|^2 \tag{5}$$

and the minimization principle can be rewritten as

$$\int_0^L w(x) g^k \, \mathrm{d}x, \tag{6}$$

where $w = |\psi|$ (semi-positive definite) and $k = m/2$. Written in logical coordinates, Eq. (6) can also be written as

$$\int_0^N w(\xi) g(\xi)^k \frac{\partial x}{\partial \xi} \, \mathrm{d}\xi, \tag{7}$$

where a change of variable is made in the integral (and in all the functions in it). Recalling the definition of g, we need to minimize the following integral:

$$\int_0^N w(\xi) g(\xi)^{k+1/2} \, \mathrm{d}\xi. \tag{8}$$

The answer to the minimization problem (8) is now obtained straightforwardly using the Euler–Lagrange approach [10]. The resulting Euler–Lagrange equation is

$$\frac{\mathrm{d}}{\mathrm{d}\xi} \left(w g^{k+1/2} \right) = 0 \tag{9}$$

that can be solved numerically to generate the grid [5,6].

Note that the solution of Eq. (9) has a simple geometrical interpretation

$$w g^{k+1/2} = \mathrm{const.} \tag{10}$$

obtained integrating the outermost derivative in Eq. (9).

Recalling the expression for k, w and the metric tensor, it follows that

$$|\psi| \left| \frac{\mathrm{d}x}{\mathrm{d}\xi} \right|^m \left| \frac{\mathrm{d}x}{\mathrm{d}\xi} \right| = \mathrm{const.} \tag{11}$$

Recalling the definition of local truncation error, this statement requires the local truncation error to be *equidistributed*

$$|e_i|V_i = \text{const.} \tag{12}$$

Eq. (12) states that the average local truncation error $|e_i|$ in a given cell i multiplied by the cell size ($V_i = |\mathrm{d}x/\mathrm{d}\xi|$) should be constant for the grid to be optimal.

The conclusion of the 1D derivation is two-fold.

First, the optimal grid can be computed solving the non-linear Euler–Lagrange equation (9). Indeed, one can observe that Eq. (9) obtained here has been long used in a different form in grid adaptation. If we define a monitor function

$$D = |\psi|\left|\frac{\mathrm{d}x}{\mathrm{d}\xi}\right|^m \tag{13}$$

the Euler–Lagrange equation (9) can be rewritten as

$$\frac{\mathrm{d}}{\mathrm{d}\xi}\left(D\frac{\mathrm{d}x}{\mathrm{d}\xi}\right) = 0 \tag{14}$$

that can be immediately recognized as the 1D version of the Euler–Lagrange equation obtained for the Winslow [12] and Brackbill–Saltzman method [5]. The importance of our approach is in the rigorous determination of the monitor function D, defined in Eq. (13). Instead, traditionally a heuristic definition of the monitor function is used.

Second, the grid generated by the method proposed here has the practical property of equidistributing the local truncation error. The implication of this result is that error minimization and error equidistribution are identical and equivalent tasks. This conclusion provides justification to the common practice of basing grid adaptation on the equidistribution of an error measure [10].

It is interesting to observe that a similar conclusion can be reached in a different way by considering the following algebraic equivalence [4]:

$$N\left(\sum_i E_i^2\right) = \frac{1}{2}\sum_i\sum_j (E_i - E_j)^2 + \left(\sum_i E_i\right)^2 \tag{15}$$

valid for any set of N real numbers E_i. When E_i is the cell error, the following equivalence can be used to construct grid adaptation strategies.

Theorem 1 (cf. [4]). *The two following statements*:

$$\begin{aligned} &\min\left[\left(\sum_i E_i^2\right)\right], \\ &\min\sum_i\sum_j (E_i - E_j)^2 \end{aligned} \tag{16}$$

with $\forall E_i \in \mathbb{R}$, *subject to the condition*

$$\sum_i E_i = \text{const.} \tag{17}$$

are satisfied by the same set E_i.

The minimizations discussed above involve the most general change of grid subdivision (with the constraint that the total number of cells N is constant and that the total volume is constant): we search the minimum of the two expressions above, while changing concurrently the cell volume, shape and location and the corresponding error E_i suitably defined in the next section.

The first of Eq. (16) represents the standard minimization of a given cell quantity E_i. The second of Eq. (16) is clearly minimum when all E_i are equal. When E_i is chosen as the total local truncation error in a cell $E_i = V_i e_i$, the conclusion of Baines' theorem is that the minimum error corresponds to an equidistribution of the error among the cells. Clearly the same conclusion that we reached above.

Note that Baines' derivation requires the total error to remain constant, Eq. (17), as it is in some implementations of conservative difference schemes where the total error is zero. In the present section, we have generalized the equivalence of minimization and equidistribution previously obtained by Baines to the case where the total error is not necessarily constant. Still our derivation is limited to the 1D case; next we generalize it to any number of dimensions.

2.2. Generalization to any number of dimensions

The extension of the procedure outlined above to any number of dimensions (ND) is trivial in theory but leads to the practical difficulty of its implementation. In this section we minimize directly the local truncation error and not its leading term. In the 1D derivation, we used the leading term for simplicity, but in the general case we prefer to use the complete truncation error that provides a more exact and complete error definition.

The theoretical starting point is generalized easily. The goal is now to determine the mapping $\mathbf{x}(\boldsymbol{\xi})$ between the ND logical coordinate $\boldsymbol{\xi}$ and the ND physical coordinate $\mathbf{x}$ that minimizes the L_1-norm of the local truncation error

$$\int_V e \, \mathrm{d}^N x, \tag{18}$$

where the error is assumed to be positive definite (if not one simply takes the absolute value as in the previous section).

The form of the Euler–Lagrange equation for the minimization of the integral definition (18) depends on how the local truncation error e is formally assumed to depend upon the mapping from the logical to the physical coordinates.

The actual local truncation error depends upon the scheme used and on the definition of the error. However, to derive a specific form of the Euler–Lagrange equation, we can assume a specific form that is not necessarily related to the actual form of the local truncation error for a specific scheme. For example in the most straightforward extension of Eq. (1), the local truncation error could be written in the form [10]

$$e(\mathbf{x}) = w(\mathbf{x}) g^k, \tag{19}$$

where w is a factor depending upon $\mathbf{x}(\boldsymbol{\xi})$ and g is the determinant of the metric tensor g_{ij} of the transformation. Clearly, we can always write any function of $\mathbf{x}$ in that manner, choosing w appropriately. As noted above, this functional form of e is chosen for convenience of derivation, but other forms could work equally well. As an example, Table 1 reports some alternative choices that could be used. We note that the discussion above is purely focused on the form of the equations. In practice the form used can be very important. If the actual dependence of the local truncation error on the mapping is exactly the one chosen, the function w that factors out in the error definition would be a smooth or even a constant function of the mapping, simplifying considerably the solution of the Euler–Lagrange equation and resulting in a efficient grid adaptation scheme. If instead, the form used for the local truncation error is totally unrelated to the actual dependence, the factor function w would be a strong function of the mapping and the solution of the Euler–Lagrange equation would be much more difficult and strongly non-linear. In the present work, we

Table 1
Formulation of varational grid generation methods previously published [10] within the new formalism presented here

$e(\mathbf{x}) = w(\mathbf{x})g^k$	$\frac{\partial}{\partial x^j}\left(wg^k g_{im}\frac{\partial \xi^m}{\partial x^j}\right) = 0$
$e(\mathbf{x}) = w_k(\mathbf{x})g^{kk}$	$\frac{\partial}{\partial x^j}\left(w_i\frac{\partial \xi^i}{\partial x^j}\right) = 0$
$e(\mathbf{x}) = w_{ij}(\mathbf{x})g^{ij}$	$\frac{\partial}{\partial x^j}\left(w_{ik}\frac{\partial \xi^k}{\partial x^j}\right) = 0$

The first column presents the link of the monitor functions with the local truncation error and the second presents the corresponding Euler–Lagrange equation. Each Euler–Lagrange is a system of three equations (in 3D) for $i = 1, 2, 3$, where the index i is not summed. All other repeated indexes are summed.

are just assuming the simplest formal dependence, Eq. (19), without claiming it to lead to the ideal Euler–Lagrange equation effective for all schemes. Future work will be focused on determining under what circumstances a given formal dependence might lead to the most effective scheme. However, this issue is closely related to the scheme used to solve the non-linearity in the Euler–Lagrange equation, an issue beyond the scope of the present work. It should be emphasized that the issue just discussed only affects the effectiveness of the solution and the difficulty of solving numerically the Euler–Lagrange equation. Whatever the formula we use to relate the error to the mapping and whatever the final form of the Euler–Lagrange equation, we are still always solving the minimization statement (18) and the final grid that we obtain (provided that we can get one and that the solution is unique) is the same.

Under the assumptions described above, the minimization principle is written as

$$\int w(\mathbf{x})g^k \, \mathrm{d}\mathbf{x}. \tag{20}$$

The expression above corresponds to Eq. (8.58) of [10] and the system of Euler–Lagrange equations is readily obtained from [10] as

$$g^{-1/2}\frac{\partial}{\partial \xi^i}\left(wg^{k+1/2}\right) = 0. \tag{21}$$

Note that in the derivaiton of the Euler–Lagrange equation, as shown in [10], one needs to exchange the physical and logical coordinates in the integral. Indeed, the 1/2 exponent in Eq. (21) comes from the Jacobian of the transformation from physical to logical coordinates.

In the traditional implementation of variational grid adaptation methods, w is the undetermined monitor function that the user chooses heuristically based upon the properties of the solution. Our derivation, instead, determines the monitor function w uniquely in terms of Eq. (19)

$$w = \frac{e}{g^k}. \tag{22}$$

Once the local truncation error is known, using the approach described in the next section, the monitor function is uniquely defined.

As in the 1D example, Eq. (21) leads to the conclusion that the error is equidistributed. By integrating Eq. (21) on cell i of the logical space, it follows that

$$wg^{k+1/2}\big|_i = \text{const.} \tag{23}$$

Recalling Eq. (22), we obtain the following equidistribution property.

Theorem 2. *In a optimal grid, defined as a grid that minimizes the local truncation error according to the minimization principle* (18)*, the product of the local truncation error in any cell i by the cell volume V_i* (*given by the Jacobian $J = \sqrt{g}$*) *is constant*

$$e_i V_i = \text{const.} \tag{24}$$

It is important to observe that the choice (19) of the form for the local truncation error is arbitrary and was chosen to simplify the derivation of the equidistribution property stated above. Arbitrarily many others can be devised, obtaining a different form for the Euler–Lagrange equations. Other possible choices can be obtained by comparing the functionals used in published variational approaches with the present formulation. For example, Table 1 casts some previously published [10] methods in the present formalism.

The present formalism leads to three main conclusions.

First, from a theoretical point of view, it creates a class of equivalent variational grid adaptation methods. Many approaches previously published and considered to be independent can now be regarded as different implementations of the same method. Once their monitor functions are linked to the local truncation error by our formalism (as done in Table 1 for a few examples), the previously published methods become equivalent and generate the same grid.

Second, the new approach presented here allows one to continue to use existing methods and existing codes, simply prescribing rigorously the previously undetermined monitor functions. Traditionally the choice of the monitor function has been based on heuristic intuition and has been very problem dependent. With the approach outlined above, the choice becomes rigorous, problem independent and automatic.

Third, the Euler–Lagrange equations for the mapping are highly non-linear. The practical solution within a computer code can be dependent upon the precise choice of implementation. Therefore, while all variational methods that can be derived within the formalism above can be considered an equivalence class that leads to the same grid, in practice some methods might perform better than others. As noted above, if the formal expression (19) reflects accurately the actual form of the dependence of the local truncation error upon the grid spacing, the monitor function w will depend weakly upon the mapping, leading to a non-linear equation easier to solve.

2.3. Implementation

In the present work, we limit the scope to 2D examples. As noted in the previous section, many possible implementations of the method proposed above are possible. Table 1 shows a number of possible examples. All Euler–Lagrange equations in Table 1 minimize the same functional, Eq. (18). Clearly, since all problems are derived from the same minimization principle, if the solution is unique, all Euler–Lagrange equations must give the same solution. However, two considerations are in order.

First, no guarantee exists that the solution is unique, and if the solution is not unique different formulations in Table 1 can give different answers even when the same truncation error is minimized as prescribed in the formulation in Eq. (18). In the examples below this never happens, but in principle it can unless a proof on uniqueness can be derived, a task beyond the scope of the present work.

Second, even when the position of the problem, Eq. (18), results in a unique solution, the various cases in Table 1 can still provide a different practical solution. In fact, when the solution is computed, a numerical method has to be employed. The actual form of the Euler–Lagrange equation in each of the cases in Table 1 is very different. Each case comes from assuming a formal dependence of the error e being minimized. All cases still minimize the same error e, according to Eq. (18), but the form of the equation is different. Once the various equations are discretized with different numerical methods, different solutions can in principle be obtained in practice.

In the present work, we do not investigate the best choice of the minimization procedure and we will not investigate which of the various choices in Table 1 (or of the many more possible but not listed in Table 1) leads to the best method. We note that this task is a challenging scientific endeavor of its own right. The issue involves the existence and uniqueness of the solution of each equation, the formulation of an appropriate numerical discretization for each equation and its numerical solution. Of particular importance is to note that all equations are very non-linear and the choice of the minimization method would also depend on the choice of the non-linear solver. Future work will be devoted to this issue.

In the present work we consider only the first two choices in Table 1 and we apply them in practice testing that they indeed result in the same grid.

In the first case one chooses to minimize the error, as prescribed by Eq. (18), by formally writing the error as

$$e(\mathbf{x}) = w(\mathbf{x})g^k. \tag{25}$$

As noted above this formal rewriting is always allowed since the determinant of the metric tensor is never singular on regular grids. This choice lead to the Euler–Lagrange equation given in the first row of Table 1. For practical purposes, it is required to write the equation in a form where x^i are the independent variables and ξ^i are the independent variables. With simple algebraic manipulations, the Euler–Lagrange for the first case in Table 1 becomes [10]

$$\frac{\partial}{\partial \xi^i}\left(eg^{1/2}\right) = 0, \tag{26}$$

where the merit function w has chosen as prescribed by the choice of the formal relationship with the error being minimized and shown in Table 1.

In the second case one chooses to minimize the error, as prescribed by Eq. (18), by formally writing the error as

$$e(\mathbf{x}) = \sum_i w_i(\mathbf{x})g^{ii}. \tag{27}$$

In the present case, we have 2 merit functions (w_l, for $l = 1, 2$ in the present 2D scenario) to choose. The choice can affect the performance of the method. In the present work, we choose the two merit functions to be equal

$$w_i(\mathbf{x}) = \frac{e(\mathbf{x})}{g^{ll}}, \tag{28}$$

where the repeated index l is summed.

Note that this specific choice leads to the classic variable diffusion method proposed by Brackbill and Saltzman [5] and by Winslow [12] and widely used in the literature. In this case, the Euler–Lagrange equation, written exchanging the role of x^i and ξ^i as dependent and independent variables, is [5,6]:

$$g^{ij}\frac{\partial}{\partial \xi^i}\left(\frac{g^{ll}}{e}\frac{\partial x^k}{\partial \xi^j}\right) = 0 \tag{29}$$

that reduces to the classic variable diffusion method choosing $D(\mathbf{x}) = g^{ll}/e$. In the example sections below, we solve both minimization problems, Eqs. (29) and (26), to verify in practice the validity of Theorem 2 and the consequent equivalence of the different Euler–Lagrange equations listed in Table 1.

For the solution of the non-linear Euler–Lagrange equations (26) and (29) we use a simple time-like approach. The time-like iteration strategy generates a succession of grids, each assuming the monitor function and the metric tensor calculated on the previous grid. The approach has been used extensively in the literature [6] and has been shown in practice to converge to the solution of the complete non-linear equation. For example, the Euler–Lagrange equation (29) is modified into a fictitious time-dependent problem

$$\frac{\partial \mathbf{x}(\tau)}{\partial \tau} = g^{ij} \frac{\partial}{\partial \xi^i} \left(D(\mathbf{x}(\tau)) \frac{\partial \mathbf{x}(\tau)}{\partial \xi^j} \right). \tag{30}$$

By using a semi-implicit time discretization method, a spatial discretization and appropriate boundary conditions [6], the grid data required for the definition of the monitor function $D(\mathbf{x})$ and of the metric tensor is obtained using the previous fictitious time step. Upon convergence to a steady solution of Eq. (30), the time dependence drops out and the solution coincides with the solution of the original Euler–Lagrange equation.

3. Operator recovery computation of the local truncation error

In the present section, we will describe the operator recovery method to estimate the local truncation error. Our aim is to derive an error detector, not a mathematically rigorous a posteriori error estimator (i.e., an error definition that converges to the actual truncation error). Error detectors are considered starting from the formalism typical of the MFE research [2]. In summary, the MFE method defines the local truncation error as the norm of the difference between the linearly interpolated discretized operators and the exact differential operators applied to the linear interpolation of the discretized fields.

The idea can be recast in a formalism suitable for application to our grid adaptation method. We consider a general multi-dimensional non-linear partial differential equation (PDE)

$$\mathbb{A}(q) = \mathbb{X}(q). \tag{31}$$

Eq. (31) summarizes the most general PDE for an unknown function $q(\mathbf{x})$ defined on the multidimensional space $\mathbf{x}$. The extension to multiple fields will be discussed below. The operator $\mathbb{X}$ summarizes all the spatial operators, of any degree and possibly non-linear; the operator $\mathbb{A}$ summarizes all other terms, such as source terms or linear and non-linear homogeneous terms. In the present paper the attention will be limited to time-independent operators but the formalism can be extended to any PDE [9].

Eq. (31) is discretized in space on a grid with N nodes $\mathbf{x}_i$

$$\mathbb{A}_i = X_i(q_1, \ldots, q_N). \tag{32}$$

The two sides of the equation are evaluated at $\mathbf{x}_i$; to achieve this goal, the operator $\mathbb{A}$ is simply evaluated in $\mathbf{x}_i$, while the operator $\mathbb{X}$, differential in space, is discretized using a suitably chosen discretization scheme to obtain X_i.

From the discretized field q_i and from the discretized operator X_i applied to q_i defined only on the grid points, it is possible to reconstruct two functions defined everywhere in the continuum space $\mathbf{x}$:

$$\begin{aligned} \widetilde{q}(\mathbf{x}) &= \mathbf{L}\{q_i\}, \\ \widetilde{X_q}(\mathbf{x}) &= \mathbf{L}\{X_i(q_1, \ldots, q_N)\}, \end{aligned} \tag{33}$$

where $\mathbf{L}$ is the operator representing the multilinear (e.g., linear in 1D, bilinear in 2D) interpolation from the grid points.

The local truncation error is defined as the residual of the application of the exact operators upon the interpolation reconstruction of the solution of the discretized equation:

$$e = \mathbb{A}(\widetilde{q}(\mathbf{x})) - \mathbb{X}\widetilde{q}(\mathbf{x}). \tag{34}$$

This expression can be simplified further when the operator $\mathbb{A}$ commutes with the linear interpolation operator $\mathbf{L}$ and it follows:

$$\mathbb{A}(\widetilde{q}(\mathbf{x})) = \widetilde{X_q}(\mathbf{x}). \tag{35}$$

Using Eq. (35), the definition of the local truncation error becomes

$$e = \widetilde{X_q}(\mathbf{x}) - \mathbb{X}\widetilde{q}(\mathbf{x}), \tag{36}$$

which defines the local truncation error as the difference between the linear interpolation of the discretized operator applied to the discretized field $\widetilde{X_q}(\mathbf{x})$ and the exact differential operator applied to the linear interpolation of the discretized field $\widetilde{q}(\mathbf{x})$. The expression (36) is valid only if the commuting condition is satisfied; in all other cases (non-linear operators generally do not satisfy it) the more general expression (34) needs to be used.

Continuing to follow ideas inspired by the MFE literature, two possible definition of the local truncation error in each cell are considered. One is borrowed from the least squares moving finite elements method (LSMFE) [3]. In that approach, the average local truncation error on any given cell i is defined as the L_2-norm:

$$e_i = \left(\frac{1}{V_i}\int_{V_i} e^2 \,\mathrm{d}V\right)^{1/2}, \tag{37}$$

where e_i is the average local truncation error over cell i and V_i is the cell volume.

Another approach (least square moving finite volumes – LSMFV) again proposed by Baines [3] requires to consider the simple average

$$e_i = \frac{1}{V_i}\int_{V_i} e \,\mathrm{d}V. \tag{38}$$

The two definitions above are different. In the first case, the local truncation error is squared and then averaged over the cell, therefore negative or positive values do not cancel and both contribute. In the second case, the local truncation error is simply averaged over each cell: if in a given cell negative and positive values co-exist, a cancellation would occur, leaving only the net value as a result. Of course, the two definitions (37) and (38) are equivalent only when the local truncation error (36) is constant in each cell. However, in practice this instance rarely occurs and the application of LSMFV can result in unwanted error cancellations when a cell has large positive and negative errors that cancel each other and are not detected by the LSMFV error measure. A safer, although somewhat more involved, choice is to adhere to the LSMFE definition.

The method presented above is based on the reconstruction of information available on a discrete set to the continuum using linear interpolation. In analogy with the gradient recovery error estimator [1] used in the finite element community and based on a similar reconstruction, we name the method described here *operator recovery* error detector. It generalizes the gradient recovery error measure by considering the actual operators that appear in the equation under consideration.

The procedure described above represents a terse but complete operational definition of the operator recovery error detector. To render the description more intuitive, Fig. 1 represents pictorially the procedure for the simple case of the gradient operator in 1D.

In the example we consider the derivative of a function $q(x)$ in 1D. The continuum is discretized on a uniform grid with spacing δ, where the nodes are labeled by n and the cells are labeled by $c = n + 1/2$ with cell c between node n and node $n + 1$. The nodal q_n values are shown as dots in Fig. 1(a). The reconstructed

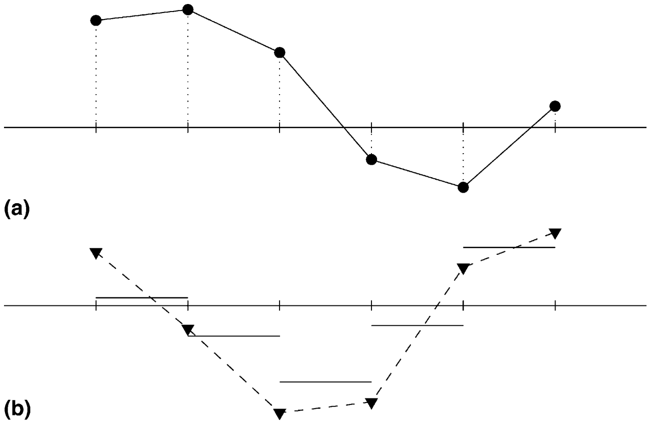

Fig. 1. Pictorial example of the application of the operator recovery error detector. The operator $\mathbb{X}(q) = \mathrm{d}q/\mathrm{d}x$ is the first order derivative in 1D. (a) The discrete function q defined at the nodes of the grid and its linear reconstruction $\widetilde{q}(x)$ (solid line). (b) The discretized operator defined at the nodes (triangles) and, from it, its linear interpolation (dashed line); the exact operator applied to the reconstructed function is shown by the solid line. The error is given by the norm over each cell of the difference between dashed line and solid line.

function $\widetilde{q}(x)$ in the continuum is obtained by linear interpolation and is shown as a solid line in Fig. 1(a). From the discrete function q_n we can define the second order accurate node centered derivative as:

$$\left.\frac{\mathrm{d}q}{\mathrm{d}x}\right|_n = \frac{q_{n+1} - q_{n-1}}{2\delta}. \tag{39}$$

The discrete approximation to the derivative operator is shown in Fig. 1(b) by triangles. From the triangles using linear interpolation, we can reconstruct the operator in the continuum $\widetilde{X}(x)$, shown as a dashed line. To compute the error we compare the reconstructed operator with the exact operator applied to the reconstructed function $\mathbb{X}\widetilde{q}$ (solid line in Fig. 1(b)). The difference between the solid and dotted line in Fig. 1(b) is the truncation error e defined in Eq. (36). The LSMFE definition of e in each cell provides the error detector e_c defined in Eq. (37).

Fig. 1 provides just a simple example but it is a complete and easy to visualize description of the general procedure.

The definition of the operator recovery error detector above is valid for any number of dimensions. However, the formulation is limited to one single equation. The extension of the method to a set of equations is considered next. The issue is how to provide an error detector for the case of a system of equations, that we can summarize as

$$\mathbb{A}_l(\mathbf{q}) = \mathbb{X}_l(\mathbf{q}), \tag{40}$$

where l labels the R equations and the M unknown fields are $\mathbf{q} \equiv \{q_f\}_{f=1,M}$.

The definition of the error detector for each equation is still provided by Eq. (38), but its relative counterpart is considered. The error detector for the system of equations (40) is then obtained summing the various relative errors for all the equations

$$e_i = \sum_l \frac{e_{l,i}}{\|\mathbb{A}_l\|_{L_2} + \varepsilon}, \tag{41}$$

where $e_{l,i}$ is the error detector for the single equation l defined above. A few considerations are in order. First, the definition (41) is well posed. When the denominator $\|\mathbb{A}_l\|_{L_2}$ vanishes, the truncation error vanishes also and the relative truncation error vanishes (thanks to the small parameter ε). Second, the error detector is dimensionless. Third, if a single field is used, Eq. (41) leads to the same adaptive grid as the previous definition since a multiplying factor does not change the solution of the Euler–Lagrange equation. Finally, multiple fields can now be readily accommodated because the various relative truncation errors are dimensionless.

Note that Eq. (41) represents a simple prescription to combine various fields with different physical units, but it is by no means unique. It is conceivable that specific circumstances might require to weigh the various fields differently.

4. Summary of the grid adaptation strategy

The fundamental innovation of the approach presented here is to connect a practical definition of the error detector with the variational formalism for grid adaptation.

The procedure described in the two previous sections is summarized below, for a generic problem in any number of dimensions.

1. We consider a general time-independent non-linear PDE for $q(\mathbf{x})$ as in Eq. (31).
2. We assume a given initial guess for the grid, for example uniform, but any given grid can be used.
3. We introduce a suitable discretization of the spatial operators, Eq. (32).
4. We solve the dicretized Eq. (32) for q_i.
5. We define the *operator recovery error detector* as in Eq. (37) based on the grid assumed.
6. We obtain an improved grid by advancing the discretized version of Eq. (30) (or the equivalent for other Euler–Lagrange equations in Table 1) one fictitious time step.
7. We iterate between 4 and 6 until convergence of the time-like grid generator problem (30).

The algorithm outlined above corresponds to a simple iteration scheme for the original non-linear problem. In a future work we will discuss a more efficient Newton–Krylov approach. Below, we apply the grid adaptation algorithm to a series of paradigmatic elliptic problems.

5. Application to a paradigmatic elliptic problem

In the present section, we consider the following elliptic problem in any number of dimensions:

$$\operatorname{div}\mathbf{E} = \rho, \tag{42}$$

where the vector field can be expressed in terms of a scalar potential as

$$\mathbf{E} = -\operatorname{grad}\Phi. \tag{43}$$

Eqs. (42) and (43) form the classic Poisson equation. In the electrostatic analogy, using units where $\epsilon_0 = 1$, $\mathbf{E}$ is the electric field and Φ is the electrostatic potential. Eqs. (42) and (43) constitute a system of equations where the second specifies that the vector field is irrotational. The derivations below are general and hold for any number of dimensions but the examples are limited to 2D.

The present section is organized as follows.

First, the operator recovery error detector defined above is computed for the specific problem considered here.

Second, the adaptation strategy and the error detector proposed above are tested for the Poisson equation. Such test will probe the performance of the adaptation method when smooth solutions are involved. By its nature the Poisson equation has smooth solutions, even in response to singular sources.

Third, the adaptation strategy and the error detector are tested in connection with quasi-singular solutions of the Debye–Huckel equation. The solution of the Debye–Huckel equation can have arbitrarily sharp transitions, depending on the choice of the source and Debye length.

5.1. Poisson equation: error detector

The computational domain is discretized on a logically cubic grid, where the fields are discretized: Φ_i and $\mathbf{E}_i$. The differential operators are discretized; the discretized divergence operator is DIV and the discretized gradient operator is GRAD:

$$\begin{aligned}&(\mathrm{DIV}\,\mathbf{E})_i = \rho_i,\\ &(\mathrm{GRAD}\,\Phi)_i = \mathbf{E}_i.\end{aligned} \tag{44}$$

For each equation the error detector can be obtained following the method described in Section 2; the errors are then combined according to Eq. (41).

For the first Eq. (43), the operators are $\mathbb{A} = \rho$ and $\mathbb{X} = \mathrm{div}\,\mathbf{E}$ and the field q is $\mathbf{E}$. The LSMFE version of the operator recovery error detector, Eq. (36), becomes

$$e^2_{\mathbf{E},i} = \frac{1}{V_i}\int_{V_i}\left(\widetilde{\mathrm{DIV}}\,\mathbf{E} - \mathrm{div}\,\widetilde{\mathbf{E}}\right)^2 \mathrm{d}V, \tag{45}$$

where $\widetilde{\mathrm{DIV}}(\mathbf{x})$ and $\widetilde{\mathbf{E}}(\mathbf{x})$ are the continuous functions obtained by interpolation, as defined in Section 3. In practice, the first term in Eq. (45) is expressed recalling that $(\mathrm{DIV}\,\mathbf{E})_i = \rho_i$

$$e^2_{\mathbf{E},i} = \frac{1}{V_i}\int_{V_i}\left(\widetilde{\rho} - \mathrm{div}\,\widetilde{\mathbf{E}}\right)^2 \mathrm{d}V. \tag{46}$$

The definition of the operator recovery error detector (46) requires the analytical integration of simple polynomial functions, a tedious but straightforward task. Such computation is performed once with pencil and paper (or more likely with MATHEMATICA [13]) and the final expression can be coded once for a given equation. All runs of the computer code will use the same analytical expression.

Similarly, we can derive the error detector for the potential equation (43)

$$\mathbf{e}^2_{\Phi,i} = \frac{1}{V_i}\int_{V_i}\left(\widetilde{\mathbf{E}} - \widetilde{\mathrm{GRAD}}\,\Phi\right)^2 \mathrm{d}V \tag{47}$$

that is computed again analytically integrating the polynomial functions present in the integral definition. It is interesting to note that in the present case, the operator to reconstruct is the gradient. The operator recovery error detector becomes the gradient recovery error estimator [1] often used in the finite element community. For this particular case, it can be showed that the error detector is an actual error estimator, that converges to the exact error, in H_1-norm [1].

The two error detectors are then combined according to Eq. (41)

$$e_i = \frac{e_{\mathbf{E},i}}{\|\rho\|_{L_2}+\varepsilon} + \frac{e_{x,\Phi,i}}{\|E_x\|_{L_2}+\varepsilon} + \frac{e_{y,\Phi,i}}{\|E_y\|_{L_2}+\varepsilon} + \frac{e_{z,\Phi,i}}{\|E_z\|_{L_2}+\varepsilon} \tag{48}$$

valid for the general 3D case.

5.2. Adaptation: 2D Poisson equation

The Poisson equation (42) and (43) is solved on a 2D unit square with Dirichelet conditions on all boundaries: $\Phi|_{\partial V} = 0$. We use the finite volume method developed by Sulsky and Brackbill [11]. The grid is adapted as described in Section 4.

We consider two different sources:

$$\begin{aligned}&\text{CASE 1}: \quad \rho_1 = \sin(2\pi x)\sin(2\pi y),\\ &\text{CASE 2}: \quad \rho_2 = 4\frac{\sigma^2 - r^2}{\sigma^4}\mathrm{e}^{-r^2/\sigma^2},\end{aligned} \tag{49}$$

where $\sigma = .1$ and $r = \sqrt{(x-.5)^2 + (y-.5)^2}$ is the radial distance from the center of the domain.

Fig. 2 shows the grid obtained at convergence of the time-like iterative procedure formulated in Eq. (30) for the two sources above. For both cases we tested the convergence procedure starting from two different initial guesses shown in Fig. 3: in both cases the same converged grid is obtained, regardless of the initial guess.

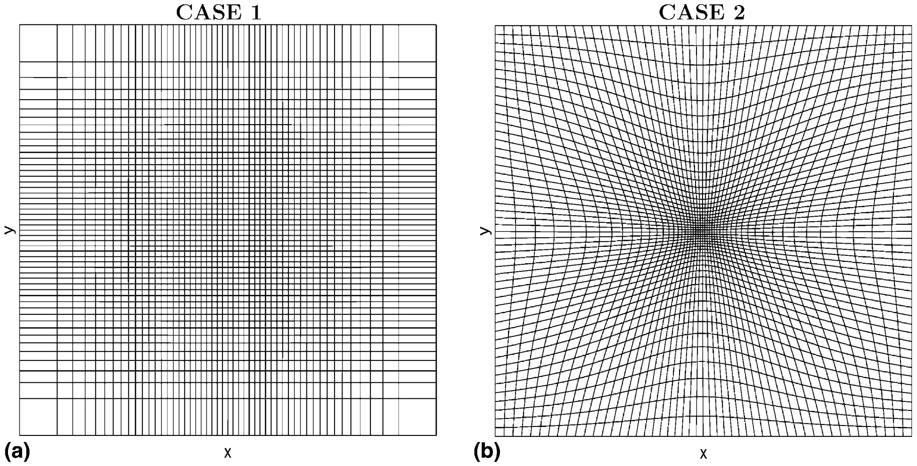

Fig. 2. Poisson equation: optimal 50 × 50 adaptive grid generated using the operator recovery error detector for CASE 1 (a) and CASE 2 (b). The same grids are obtained regardless of the initial condition (see Fig. 3) and of choice of the Euler–Lagrange in Table 1 (only the first two rows were actually tested).

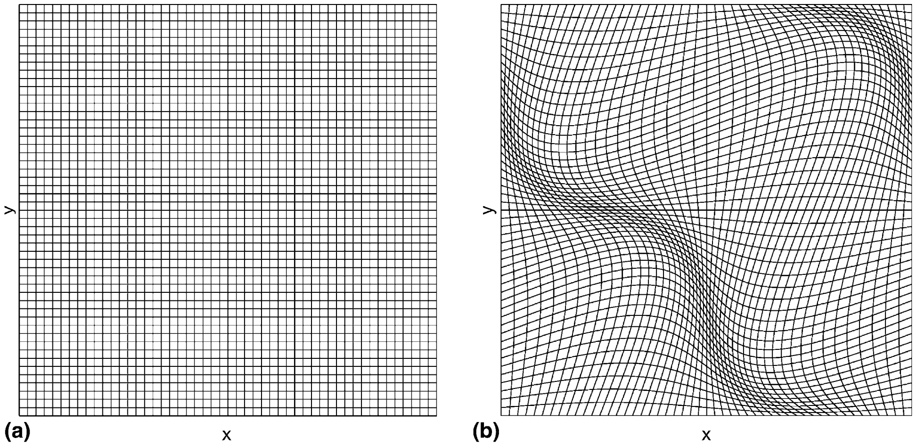

Fig. 3. Poisson equation, 50 × 50 grids used as initial guess for generating the optimal grid. Grid (a) and grid (b) are used in CASE 1 and CASE 2.

Note that in CASE 1, the source is separable in its x and y dependence, consequently the optimal grid is simply the tensor product of the two 1D optimal grids in x and y. In CASE 2, instead, the symmetry of the source is cylindrical, and the converged grid attempts to satisfy cylindrical symmetry.

Note that the optimal grids shown in Fig. 2 have two significant properties of relevance to the theoretical considerations described above. First, the estimate of the local truncation error is exactly uniform over the grid: $e_i V_i$ is constant (within a standard deviation of 10^{-3} due to the convergence criteria used in stopping the convergence iteration), as prescribed by the equidistribution theorem derived in Section 2. Furthermore, the solution obtained using Eq. (29) is compared with the solution of Eq. (26). We obtained visually the same grids (and for that reason we do not not report the grids again) using both methods, with quantitative differences less than then the convergence criteria ($|\Delta \mathbf{x}_i| < 10^{-3}$). Second, the ratio of the exact error ($|\Phi_i - \Phi_{\text{exact}}|$) and the operator recovery error detector varies only within 1.3%, proving that the operator recovery method provides a good approximation of the error.

Even for the smooth sinusoidal source of CASE 1, the use of adaptive grids can reduce the discretization error. Fig. 4 shows the actual error (in max-norm) in the numerical solution (the analytical solution is known) as a function of the number of grid points for different square grids, using uniform grids and two different grid adaptation methods. In all cases the correct scaling of the error typical of the scheme used is recovered. The use of the adaptation method considered in Section 2 provides a reduction of the error by a factor of three, without altering the convergence pattern. It is interesting to consider a heuristic monitor function often used in the variational grid approach [6,5,10]: $D = |\nabla\Phi|/|\Phi|$. In that case, the adaptive grid is not only inferior to the grid generated with the method proposed here, but even to the uniform grid.

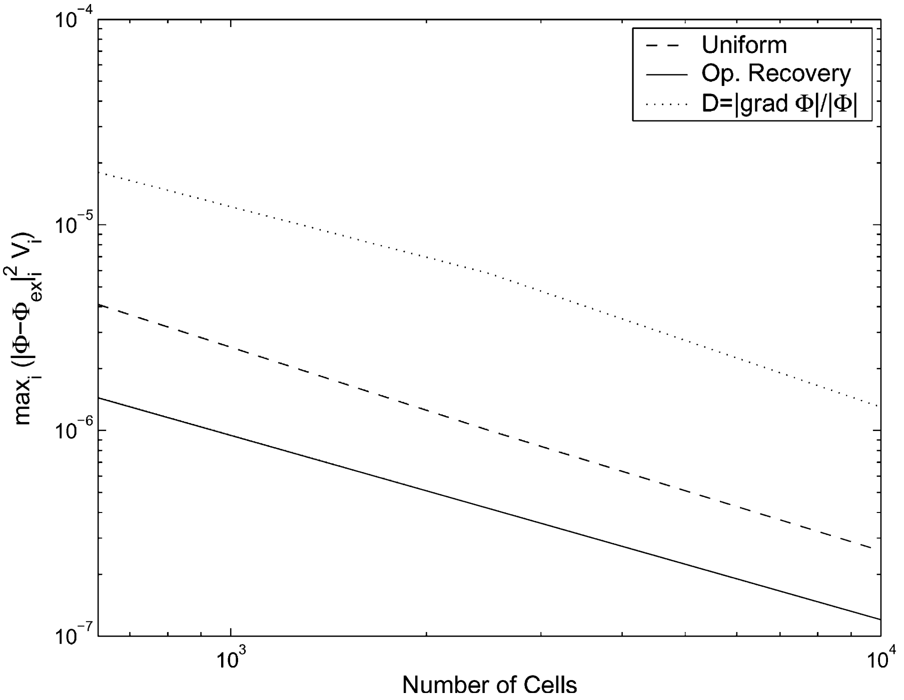

Fig. 4. Poisson equation. CASE 1: Convergence on a uniform (dashed), and two types of adaptive grids: one uses the operator recovery error detector (solid line) and the other uses a variational grid adaptation method based on the heuristic weight $|\nabla\Phi|/\Phi$ (dotted). All grids are squared with equal number of subdivisions in x and y.

The reason can be observed in Fig. 5: the error in the numerical solution is larger in the central region (where the second derivative is largest), while the gradients are largest towards the boundaries. Fig. 5 compares the actual error on a uniform 50 × 50 grid with the grid spacing produced by the operator recovery error detector and by the heuristic monitor function. Clearly the heuristic monitor function $D = |\nabla\Phi|/|\Phi|$ is completely inadequate in the present case. The use of the operator recovery error detector leads to grids with finer spacing in the central region, thereby reducing the largest error. The heuristic weight leads to grids with finer spacing near the boundary, reducing the error where it is already small, and not attempting to reduce the larger error in the centre of the system. As an example Fig. 6 shows the adaptive grid generated by the heuristic weight. The spacing is coarse in the center where more accuracy would be needed.

Similar conclusions are reached using the Gaussian profile of CASE 2. Fig. 7 shows the convergence study (using both the max-norm and the L_2-norm). Again, in all cases the correct second order scaling of the error typical of the scheme used is recovered. The same conclusions are reached regarding the superiority of the operator recovery error detector proposed above when compared with the heuristic monitor function. In the present case, the solution is more localized and the improvement gained by the use of an adaptive grid is more notable: more than an order of magnitude is gained.

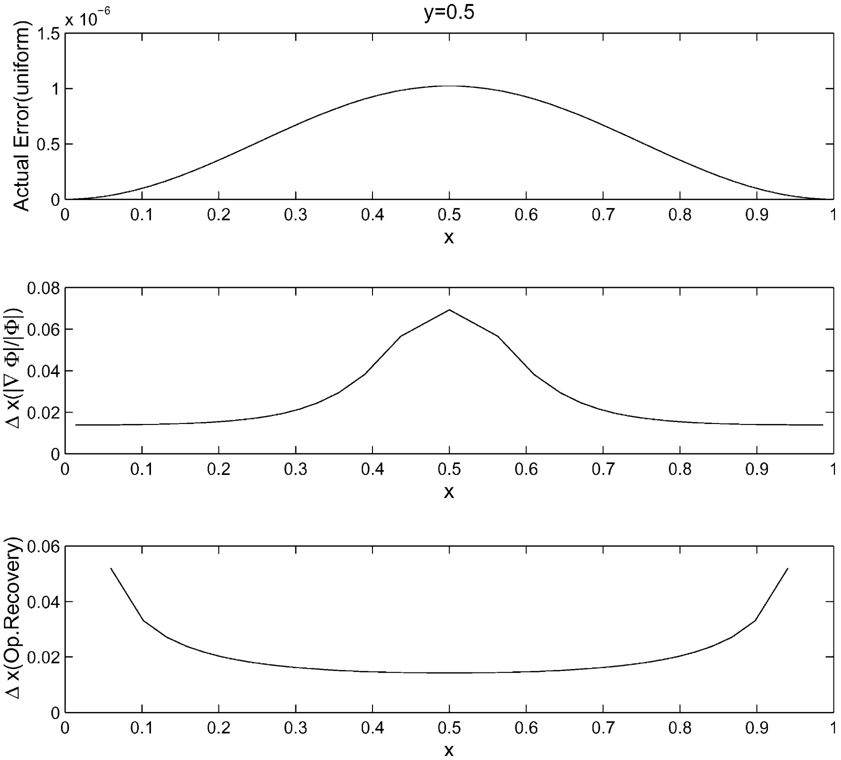

Fig. 5. Poisson equation, CASE 1: On a uniform 50 × 50 grid, the top panel shows the actual error; the center panel shows the grid spacing on a adaptive grid generated using the operator recovery error detector; the bottom panel shows the grid spacing on a adaptive grid generated using the heuristic weight $|\nabla\Phi|/\Phi$. All plots are cuts along the line $y = .5$.

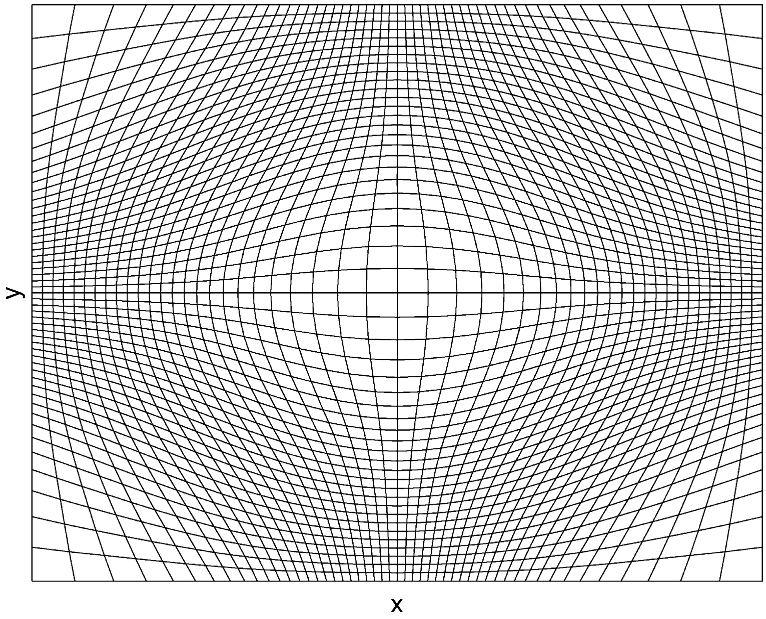

Fig. 6. Poisson equation, CASE 1: 50 × 50 adaptive grid generated using the heuristic error detector.

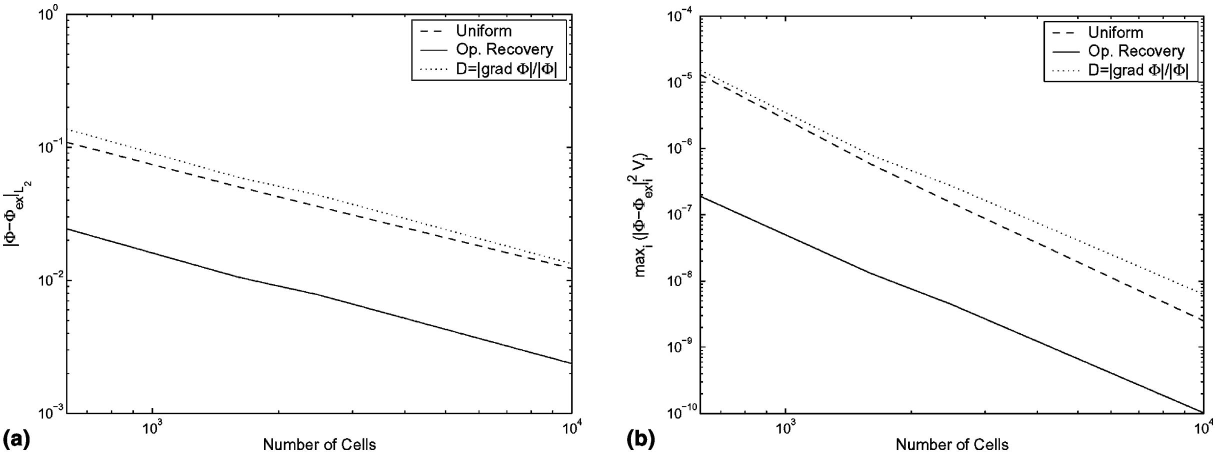

Fig. 7. Poisson equation. CASE 2: Convergence on a uniform (dashed), and two types of adaptive grids: one uses the operator recovery error detector (solid line) and the other uses a variational grid adaptation method based on the heuristic weight $|\nabla\Phi|/\Phi$ (dotted). All grids are squared with equal number of subdivisions in x and y. Two error norms are used: L_2-norm (a) and max-norm (b).

5.3. Adaptation: 2D Debye equation

As a further example, we consider the Debye–Huckel equation

$$\nabla^2\Phi = -\frac{\Phi}{\lambda_D^2} - \rho, \tag{50}$$

where λ_D is the Debye length that controls the spatial decay rate of the solution in response to a localized source. The interest of the Debye–Huckel equation in testing our method is that it allows localized solutions (almost singular) while the Poisson equation, by its nature, has smooth solutions. The solution of the Debye–Huckel equation and the adaptive grid are obtained using the same approach used for the Poisson equation.

Fig. 8 shows the 50 × 50 adaptive grid obtained for a case where

$$\rho = \left(4\frac{\sigma^2 - r^2}{\sigma^4} - \frac{1}{\lambda_D^2} \right) e^{-r^2/\sigma^2} \tag{51}$$

with $\lambda_D = 0.1$ and $\sigma = .01$. Note that in the present case smoothing and rescaling of the merit function D is used to impose a maximum ratio of largest to smallest cell of 100 [5].

The adaptive grid provides a great advantage over a uniform grid. Fig. 9 shows the convergence of the actual error (using both the max-norm and the L_2-norm) as the number of grid points is increased on a uniform and on a adaptive grid. More than four orders of magnitude are gained.

In the present case the source and the solution are very localized, nearly singular. The heuristic monitor function $D = |\nabla \Phi|/|\Phi|$ has been originally proposed specifically for such cases [5]. Indeed, Fig. 9 shows that the use of the heuristic weight is nearly as good as the use of the operator recovery error detector.

Note that the present implementation of the solver of the grid evolution Eq. (30) does not allow sufficient grid condensation around the singularity. For this reason, we have been forced to use smoothing and rescaling of D. As a consequence, the grid cannot condense enough to provide the ideal convergence scaling on non-uniform grids (while it does on uniform grids). The reason for this limitation is technical, the iteration of the non-linearity and the conjugate residual solver of the resulting linear equations fail when strong condensation is required. This is not a failure of the formulation of the grid adaptation method.

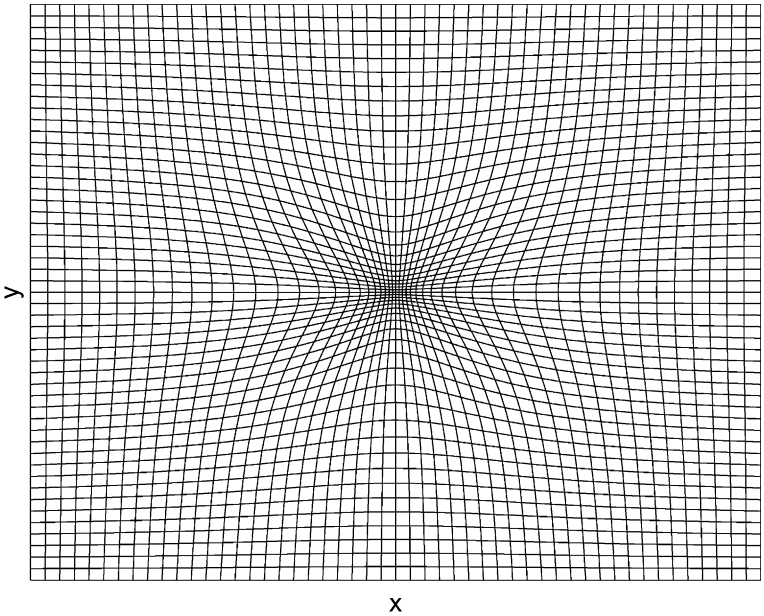

Fig. 8. Debye–Huckel: Example of a 50 × 50 adaptive grid generated with the operator recovery error detector.

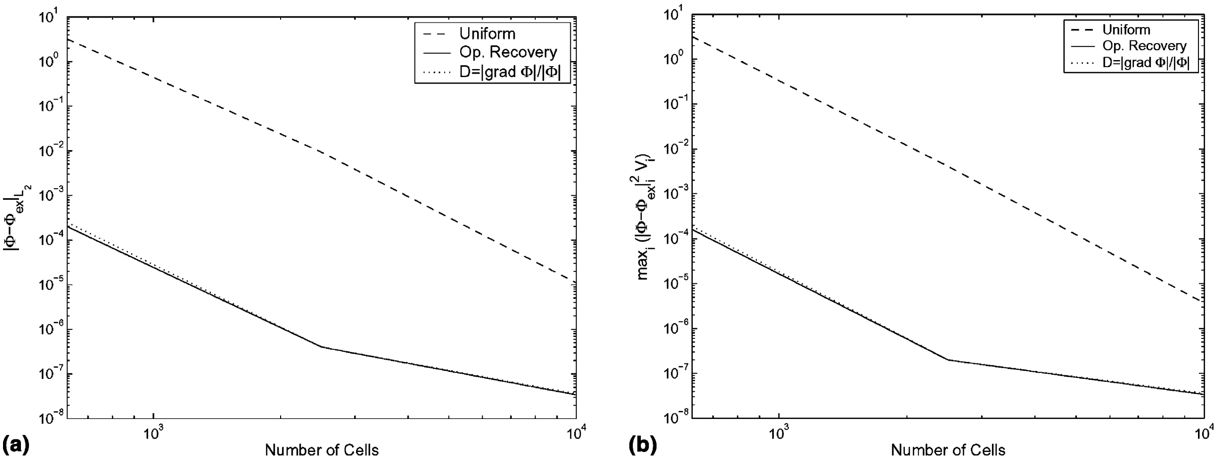

Fig. 9. Debye–Huckel: Convergence on a uniform (dashed), and two types of adaptive grids: one uses the operator recovery error detector (solid line) and the other uses a variational grid adaptation method based on the heuristic weight $|\nabla\Phi|/\Phi$ (dotted). All grids are squared with equal number of subdivisions in x and y. Two error norms are used: L_2-norm (a) and max-norm (b).

Work is currently undergoing on designing Newton–Krylov solvers of the non-linear grid adaptation equations that should remove the limitation on strong grid condensation.

The importance of the conclusion reached above is that only the operator recovery detector can be proposed as a general weight function for grid generation applicable in all cases without user intervention. The heuristic monitor function that fails when smooth solutions are considered, can only be used in specific cases and relies on the user's knowledge of the physics of the problem under consideration.

6. Conclusions

We have derived a new approach to grid adaptation for time-independent problems. It is based on the use of the variational grid adaptation method and of a reliable error detector.

We have derived the method and summarized its implementation. We have then applied it to classic elliptic problems. Two conclusions are reached.

First, the heuristic weight $D = |\nabla\Phi|/|\Phi|$ often used in adaptive grid strategies is only appropriate for nearly singular solutions, while the method proposed here and based on the operator recovery error detector never fails.

Second, the various tests performed show that the adaptation strategy described in the present paper can be the basis for an automatic adaptation tool.

Future work will be devoted to the application of the present work. First, the issue of the practical efficiency of the various formulations listed in Table 1 will be considered. Second, the method will be applied to a larger class of problems, including parabolic and hyperbolic problems where grid anisotropy might become important [6].

Acknowledgements

The author is very grateful to Jerry Brackbill for his invaluable help and for providing computer codes for grid adaptation. The author wishes to thank Robert Anderson, Andy Barlow, Luis Chacón, Doug

Kothe, Alex Kurganov, Len Margolin, Rick Pember, Misha Shashkov and Alex Shestakov for many useful discussions on the topics reported in the present paper. The author is also very grateful to Mike Baines for the illuminating suggestions during his visit to Los Alamos in summer 1999. This research is supported by the National Nuclear Security Administration of the United States Department of Energy, under the Accelerated Strategic Computing Initiative (ASCI).

References

[1] M. Ainsworth, J.T. Oden, A Posteriori Error Estimation in Finite Element Analysis, Wiley, New York, 2000.
[2] M.J. Baines, Moving Finite Elements, Oxford University Press, Oxford, 1994.
[3] M.J. Baines, Appl. Numer. Math. 26 (1998) 77.
[4] M.J. Baines, Numeric. Methods Partial Diff. Eq. 15 (1999) 605.
[5] J.U. Brackbill, J.S. Saltzman, J. Comput. Phys. 46 (1982) 342.
[6] J.U. Brackbill, J. Comput. Phys. 108 (1993) 38.
[7] C. deBoor, B. Swartz, SIAM J. Numer. Anal. 10 (1973) 582.
[8] J.K. Dukowicz, J. Comput. Phys. 56 (1984) 324.
[9] G. Lapenta, A recipe to detect the error in discretization schemes, Int. J. Numer. Methods Engrg. (to appear).
[10] V.D. Liseikin, Grid Generation Methods, Springer, Berlin, 1999.
[11] D. Sulsky, J.U. Brackbill, J. Comput. Phys. 96 (1991) 339.
[12] A.M. Winslow, UCID-19062, Lawrence Livermore National Laboratory, 1981.
[13] S. Wolfram, The Mathematica Book, Cambridge University Press, Cambridge, 1999.
[14] X.D. Zhang, J.-Y. Trépanier, R. Camarero, Comput. Methods Appl. Mech. Engrg. 185 (2000) 1.