INTERNATIONAL JOURNAL OF
NUMERICAL ANALYSIS AND MODELING, *SERIES B*
Volume 3, Number 4, Pages 408–428

A MESH ADAPTATION METHOD FOR 1D-BOUNDARY LAYER PROBLEMS

ANDRÉ FORTIN, JOSÉ M. URQUIZA, AND RICHARD BOIS

Abstract. We present a one-dimensional version of a general mesh adaptation technique developed in [1, 2] which is valid for two and three-dimensional problems. The simplicity of the one-dimensional case allows to detail all the necessary steps with very simple computations. We show how the error can be estimated on a piecewise finite element of degree k and how this information can be used to modify the grid using local mesh operations: element division, node elimination and node displacement. Finally, we apply the whole strategy to many challenging singularly perturbed boundary value problems where the one-dimensional setting allows to push the adaptation method to its limits.

Key words. hierarchical error estimation, mesh adaptation, singular perturbation, boundary layer problems.

1. Introduction

Adaptive numerical methods have now a long history in efficiently computing approximations to ordinary or partial differential equations. Their goal is to reach a given level of precision at a minimal cost which often means a minimal number of degrees of freedom (DOFs). This also results in minimizing the size of the resulting algebraic systems of equations to be solved. In the case of partial differential equations (PDEs) and the finite element method – one of the most important approximation methods for PDEs – the two key ingredients are an *a posteriori* error estimator and a mesh adaptation procedure. The adaptation method uses the information from the error estimator to modify the mesh in order to reach the desired error level. For each of these two steps, there exist a large variety of possibilities. Our goal here is not to review all or even several of these methods and we refer the interested reader to [3, 4, 5, 6, 7] for starting points.

In this paper, we present a one-dimensional version of a general technique developed in [1, 2] for two and three-dimensional problems. The presented method does not depend on the PDE itself (unlike some other error estimations, based on residuals, as in [8]) and can be applied to finite element solutions of any degree.

The adaptation method is based on an error estimator that can be easily calculated on each element. As we shall see, our error estimator has a nice interpretation linking the finite element error to the classical Lagrange interpolation error. Once the error has been estimated, local operations are used to modify the mesh: element division, node elimination and node displacement. This indirectly means that the node positions are also unknown and must therefore be determined through an iterative method.

The first main goal of the present paper is to take advantage of the one-dimensional setting to give all the details of the adaptive strategy: the explicit construction of

Received by the editors May, 2012 and, in revised form, September 2012.
2000 *Mathematics Subject Classification.* 65G20 65N12 65N50.
This research was supported by NSERC.

the error estimator, the computation of the local errors on a patch of adjacent elements and the decision process for the local mesh modifying operations. This is done for two-point boundary value problems.

The second main goal is to illustrate the efficiency of the method through a variety of problems, some of them having proved to be particularly challenging steady convection-diffusion-reaction equations. When the diffusion coefficient is small compared to the convection velocity, solutions present steep variations localized in so-called *boundary layers*. Finite element approximations then present unphysical oscillations in these regions unless the mesh size is very small and thus the resulting algebraic system to be solved is considerably large. Part of this problem can be circumvented by modifying the variational formulation and adding stabilization terms but oscillations may remain or overshooting (or undershooting) phenomena may appear. In [9], stabilization methods are combined with graded meshes adapted to the solution to improve the numerical solution but using such hand-tailored meshes necessitates a precise knowledge of the solution. This is highly unrealistic for more general applications such as fluid flow problems in several space dimensions. In our numerical examples, we show that the automatic adaptation strategy presented here, which does not use an *a priori* knowledge of the solution (nor of the PDEs being solved) compares favorably with these hand-made meshes. Finally, we also test our strategy for approximating solutions presenting discontinuities, an even more challenging problem.

The major advantage of the one-dimensional case is that it can be pushed to the limit without unduly increasing the computational burden. As we shall see in the numerical results, we end up with elements of length as small as 10^{-11} in a unit domain. This would hardly be possible in 2D and even less in 3D.

This article is organized as follows. In Section 2 we introduce the one-dimensional convection-diffusion equation and its standard Galerkin and stabilized Petrov-Galerkin formulations. The error estimator and its interpretation as an interpolation error are presented in details in Section 3. In Section 4, each step and the whole methodology of the adaptation strategy is presented in the one-dimensional setting. In Section 5 we present our numerical tests, which compare our mesh adaptation strategy to uniform meshes or to hand-made graded meshes for convection-diffusion problems presenting boundary layers or even discontinuous solutions.

2. Position of the problem

As a test problem, we consider the classical convection-diffusion-reaction equation of the general form

$$-\frac{d}{dx}\left(d(x)\frac{du}{dx}(x)\right) + b(x)\frac{du}{dx}(x) + c(x)u(x) = f(x) \tag{1}$$

on the domain $\Omega = [\alpha, \beta]$ with Dirichlet boundary conditions $u(\alpha) = u_\alpha$ and $u(\beta) = u_\beta$. More general boundary conditions can also be considered. Convection-diffusion problems have been studied for a long time since it is well known that standard Galerkin finite element formulations (or centered finite differences) are unable to produce appropriate solutions when advection is dominant, unless very fine meshes are used. Some form of stabilization is needed, such as the Streamline-Upwind-Petrov-Galerkin (SUPG) formulation which goes back to the work of Hughes and Brooks [10, 11].

We will solve equation (1) by the finite element method. This requires a mesh $\mathcal{T}_h$ which is a partition of the interval $[\alpha, \beta]$ with N elements denoted $K = [x_i, x_{i+1}]$ with length $h_K = x_{i+1} - x_i$. Let V_h the space of Lagrange continuous functions

whose restriction to any element K of $\mathcal{T}_h$ belongs to the space $P^{(k)}(K)$ of polynomials of degree k over the element K. We also introduce the space V_{h0} of the functions of V_h vanishing at $x = \alpha$ and $x = \beta$. Multiplying equation (1) by $v_h^{(k)}(x) + \tau_K b(x)\frac{dv_h^{(k)}}{dx}$ with $v_h^{(k)}(x) \in V_{h0}$, we get

$$\int_\alpha^\beta \left(-\frac{d}{dx}\left(d(x)\frac{du_h^{(k)}}{dx}\right) + b(x)\frac{du_h^{(k)}}{dx} + c(x)u_h^{(k)}(x) - f(x)\right) v_h^{(k)}(x)\, dx +$$

$$\sum_{K\in\mathcal{T}_h} \tau_K \int_K \left(-\frac{d}{dx}\left(d(x)\frac{du_h^{(k)}}{dx}\right) + b(x)\frac{du_h^{(k)}}{dx} + c(x)u_h^{(k)} - f(x)\right) b(x)\frac{dv_h^{(k)}}{dx}\, dx = 0.$$

The first diffusion term is integrated by parts

$$\int_\alpha^\beta -\frac{d}{dx}\left(d(x)\frac{du_h^{(k)}}{dx}\right) v_h^{(k)}(x)dx = \int_\alpha^\beta d(x)\frac{du_h^{(k)}}{dx}\frac{dv_h^{(k)}}{dx}dx$$

and the boundary terms vanish due to the Dirichlet boundary conditions ($v_h^{(k)}(\alpha) = v_h^{(k)}(\beta) = 0$). The diffusion stabilization term

$$-\sum_K \tau_K \int_K \frac{d}{dx}\left(d(x)\frac{du_h^{(k)}}{dx}\right) b(x)\frac{dv_h^{(k)}}{dx}dx \tag{2}$$

remains as it is, though it vanishes in the linear case ($k = 1$) when $d(x)$ is constant. This is probably why it is sometimes neglected in the quadratic case but this may have undesirable consequences. The SUPG-stabilized variational formulation takes the form: find $u_h^{(k)} \in V_h$ such that for all $v_h^{(k)} \in V_{h0}$

(3)

$$\int_\alpha^\beta d(x)\frac{du_h^{(k)}}{dx}\frac{dv_h^{(k)}}{dx} + \left(b(x)\frac{du_h^{(k)}}{dx} + c(x)u_h^{(k)}(x) - f(x)\right) v_h^{(k)}(x)\, dx +$$

$$\sum_{K\in\mathcal{T}_h} \tau_K \int_K \left(-\frac{d}{dx}\left(d(x)\frac{du_h^{(k)}}{dx}\right) + b(x)\frac{du_h^{(k)}}{dx} + c(x)u_h^{(k)} - f(x)\right) b(x)\frac{dv_h^{(k)}}{dx}\, dx = 0.$$

Determining an optimal choice for the parameter τ_K has been the object of a vast literature. We will use the formula proposed by Roos [12]

$$\tau_K = \begin{cases} \dfrac{h_K}{2|b_K|}\left(1 - \dfrac{1}{P_K}\right) & \text{if} \quad P_K = \dfrac{|b_K|h_K}{2d_K} > 1, \\ 0 & \text{if} \quad P_K \le 1, \end{cases}$$

where P_K is the local Peclet number and b_K and d_K are the values of $b(x)$ and $d(x)$ at the center of the element. Other similar forms could be used. To obtain the classical Galerkin formulation, we simply put $\tau_K = 0$.

Under fairly general conditions, one can expect the error to satisfy

$$|u - u_h^{(k)}|_{1,\Omega} = \left(\int_\alpha^\beta \left(\frac{du}{dx}(x) - \frac{du_h^{(k)}}{dx}(x)\right)^2 dx\right)^{1/2} \le C_1 h^k |u|_{k+1,\Omega}$$

for the H^1-seminorm while for the L^2-norm, one should observe

$$\|u - u_h^{(k)}\|_{0,\Omega} = \left(\int_\alpha^\beta (u(x) - u_h^{(k)}(x))^2\, dx\right)^{1/2} \le C_0 h^{k+1} |u|_{k+1,\Omega}$$

for the Galerkin formulation and

$$\|u - u_h^{(k)}\|_{0,\Omega} \leq C_0 h^{k+1/2} |u|_{k+1,\Omega}$$

for the SUPG formulation (see Roos-Stynes-Tobiska [13]). Note that, in the latter case, $O(h^{k+1})$ convergence is often observed in practice but it is shown in Zhou [14] that there exists two-dimensional examples where the convergence rate is only $k + 1/2$.

In the above error estimates, C_0 and C_1 are constants, independent of h and u. Moreover, h is the mesh size but this definition makes sense only on quasi uniform meshes. For the adapted meshes that will be considered in this paper, it is more appropriate to express these convergence rates as functions of the number of degrees of freedom (DOFs). For instance, for a linear solution, the number of DOFs is $N+1$ while it is $2N + 1$ for a quadratic solution and so forth. We thus have $h \sim N^{-1}$ and consequently

$$\|u - u_h^{(k)}\|_{0,\Omega} \leq C_0 N^{-(k+1)} |u|_{k+1,\Omega} \text{ and } |u - u_h^{(k)}|_{1,\Omega} \leq C_1 N^{-k} |u|_{k+1,\Omega}. \tag{4}$$

3. Error estimator

Let u be the solution of Equation (1), usually unknown. We therefore suppose that $u_h^{(k)} \in V_h$ is a finite element approximation of degree k to the exact solution u. Our objective now is to estimate the error $\|u - u_h^{(k)}\|$ where $\|\cdot\|$ is some appropriate norm.

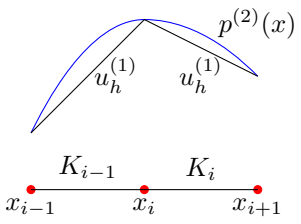
Our error estimator is based on the hypothesis that a more accurate approximation $\tilde{u}_h^{(k+1)}$ of degree $k+1$ of the solution can be built from $u_h^{(k)}$ in a post-processing step. The precise construction of $\tilde{u}_h^{(k+1)}$ will soon be described but for now we suppose its existence. The error is then simply approximated using

$$||u - u_h^{(k)}|| \simeq ||\tilde{u}_h^{(k+1)} - u_h^{(k)}||. \tag{5}$$

Our construction of $\tilde{u}_h^{(k+1)}$ requires not only the finite element solution $u_h^{(k)}$ but it also necessitates a continuous approximation of degree k of its derivative that will be denoted $g_h^{(k)}$ and which will also belong to V_h.

3.1. Approximating the derivatives at nodes.

Since $u_h^{(k)}$ is of Lagrange type, its derivative is discontinuous at element interfaces. Therefore $\frac{du_h^{(k)}}{dx} \notin V_h$ and some form of projection is needed. We thus want to evaluate the derivatives at each node of the mesh, including those at the boundary between two adjacent elements where $\frac{du_h^{(k)}}{dx}$ is not defined. These nodal values will be denoted $g_i^{(k)}$. In the linear case, we consider for each node, the polynomial of degree 2 ($p^{(2)}(x)$) interpolating $u_h^{(1)}(x)$ at the three nodes of the two adjacent elements:



and the value of the recovered derivative at this node is simply $g_i^{(1)} = \frac{dp^{(2)}}{dx}(x_i)$. Once this is done for each node, a piecewise linear approximation (denoted $g_h^{(1)}(x)$) of the derivative $u'(x)$ is obtained. In the quadratic case, we determine for each

node x_i (including mid-element nodes) of the mesh, a polynomial of degree 3 using the 4 closest nodes and we set $g_i^{(2)} = \frac{dp^{(3)}}{dx}(x_i)$. In the general case, a similar procedure can be applied to obtain the nodal values $g_i^{(k)}$ of the recovered derivative $g_h^{(k)}(x)$.

The above procedure for the recovery of derivatives at nodes works well but for the two and three-dimensional cases, we use a more sophisticated method proposed by Zhang and Naga [15] which requires the solution of a least square problem around each node. The method is shown to be superconvergent on more or less regular meshes.

3.2. Construction of the estimator. The construction of $\hat{u}_h^{(k+1)}$ is performed on an element by element basis. Indeed, on each element of the mesh, we look for an approximation of the form:

$$\hat{u}_h^{(k+1)}(x)\Big|_K = u_h^{(k)}(x)\Big|_K + c_K^{(k+1)}\phi_K^{(k+1)}(x),$$

where $c_K^{(k+1)}$ is an unknown coefficient and $\phi_K^{(k+1)}(x)$ is a basis function of degree $k+1$. The new solution is thus written as the sum of the finite element solution of degree k and a correction term of degree $k+1$. Fort this reason, we say that the estimator is *hierarchical* as in Bank and Smith [16]. The function $\phi_K^{(k+1)}(x)$ vanishes at all the nodes of the element. For instance, in the linear and quadratic cases, let

$$\phi^{(2)}(\xi) = 1 - \xi^2 \text{ and } \phi^{(3)}(\xi) = \xi(1-\xi^2)$$

be the quadratic and cubic hierarchical basis functions defined on the reference element $[-1, 1]$. Transformed on the element $K = [x_i, x_{i+1}]$, we get

$$\phi_K^{(2)}(x) = \frac{4}{h_K^2}(x - x_i)(x_{i+1} - x), \quad \phi_K^{(3)}(x) = \frac{8}{h_K^3}(x - x_i)(x - x_m)(x_{i+1} - x), \tag{6}$$

where $x_m = (x_i + x_{i+1})/2$ is the mid-element node for a quadratic element. These functions are illustrated in Figure 1.

The idea for the construction of $\hat{u}_h^{(k+1)}$ is then very simple. Since $\hat{u}_h^{(k+1)}$ is an approximation of u and $g_h^{(k)}$ is an approximation of its derivative, then the derivative of order $k+1$ of $\hat{u}_h^{(k+1)}$ should coincide with the derivative of order k of $g_h^{(k)}$

$$\frac{d^{k+1}}{dx^{k+1}}\hat{u}_h^{(k+1)} = \frac{d^k}{dx^k}g_h^{(k)}.$$

Since $u_h^{(k)}$ is a polynomial of order k, it disappears from the left-hand side and we get

$$\frac{d^{k+1}}{dx^{k+1}}\hat{u}_h^{(k+1)} = c_K^{(k+1)}\frac{d^{k+1}}{dx^{k+1}}\phi_K^{(k+1)}.$$

The unknown coefficient can now be evaluated as

$$c_K^{(k+1)} = \frac{\frac{d^k}{dx^k}g_h^{(k)}}{\frac{d^{k+1}}{dx^{k+1}}\phi_K^{(k+1)}}.$$

In the linear and the quadratic cases, simple computations give

$$c_K^{(2)} = -\frac{h_K}{8}(g_{i+1}^{(1)} - g_i^{(1)}) \text{ and } c_K^{(3)} = -\frac{h_K}{12}(g_{i+1}^{(2)} - 2g_m^{(2)} + g_i^{(2)}), \tag{7}$$

where $g_i^{(k)}$ denotes the nodal values of the recovered derivative at node i and $g_m^{(k)}$ is the estimated derivative at the mid-element node x_m. We now have our error estimator.

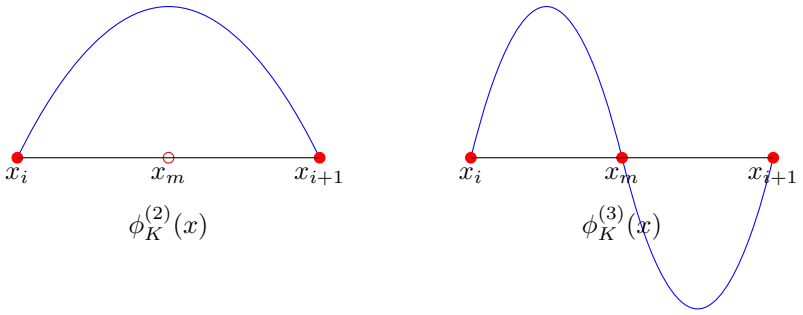


FIGURE 1. Quadratic (left) and cubic (right) basis functions $\phi_k^{(k)}(x)$ on the element $[x_i, x_{i+1}]$

In the adaptation process, the L^2-norm and the H^1-seminorm of the estimated error will be needed. A simple calculation gives

$$||\tilde{u}_h^{(2)} - u_h^{(1)}||_{0,K}^2 = \int_{x_i}^{x_{i+1}} \left(c_K^{(2)}\phi_K^{(2)}(x)\right)^2 dx = \frac{h_K^3}{120}(g_{i+1}^{(1)} - g_i^{(1)})^2 \tag{8}$$

and

$$|\tilde{u}_h^{(2)} - u_h^{(1)}|_{1,K}^2 = \int_{x_i}^{x_{i+1}} \left(c_K^{(2)}\frac{d}{dx}\phi_K^{(2)}(x)\right)^2 dx = \frac{h_K}{12}(g_{i+1}^{(1)} - g_i^{(1)})^2 \tag{9}$$

for the linear case while one gets

$$||\tilde{u}_h^{(3)} - u_h^{(2)}||_{0,K}^2 = \int_{x_i}^{x_{i+1}} \left(c_K^{(3)}\phi_K^{(3)}(x)\right)^2 dx = \frac{h_K^3}{1890}(g_{i+1}^{(2)} - 2g_m^{(2)} + g_i^{(2)})^2 \tag{10}$$

and

$$|\tilde{u}_h^{(3)} - u_h^{(2)}|_{1,K}^2 = \int_{x_i}^{x_{i+1}} \left(c_K^{(3)}\frac{d}{dx}\phi_K^{(3)}(x)\right)^2 dx = \frac{h_K}{45}(g_{i+1}^{(2)} - 2g_m^{(2)} + g_i^{(2)})^2 \tag{11}$$

in the quadratic case. Note that $u_h^{(k)}$ is not needed in the estimation of the error.

3.3. Interpretation of the error estimator. The error is thus estimated as

$$u - u_h^{(k)} \simeq \tilde{u}_h^{(k+1)} - u_h^{(k)} = c_K^{(k+1)}\phi_K^{(k+1)}(x).$$

In the linear case, recalling that $g_h^{(1)}$ is a piecewise linear approximation of $u'(x)$, we have from (7) and (6)

$$\begin{aligned} u(x) - u_h^{(1)}(x) &\simeq c_K^{(2)}\phi_K^{(2)}(x) = -\frac{1}{2h_K}(g_{i+1}^{(1)} - g_i^{(1)})(x - x_i)(x_{i+1} - x), \\ &= \frac{1}{2}\frac{(g_{i+1}^{(1)} - g_i^{(1)})}{h_K}(x - x_i)(x - x_{i+1}), \\ &\simeq \frac{1}{2}u''(x_m)(x - x_i)(x - x_{i+1}) + O(h_K^2). \end{aligned}$$

Similarly, $g_h^{(2)}(x)$ is now a piecewise quadratic approximation of $u'(x)$ and

$$u(x) - u_h^{(2)}(x) \simeq c_K^{(3)}\phi_K^{(3)}(x) \simeq \frac{1}{3!}u'''(x_m)(x - x_i)(x - x_m)(x - x_{i+1}) + O(h_K^3)$$

in the quadratic case. The error estimator on the element is nothing but an approximation of the classical Lagrange interpolation error.

4. Mesh adaptation

The global strategy driving our mesh adaptation is to try to reach a target level e_Ω of error in L^2-norm. Other norms can also be considered but our experience shows that the L^2-norm provides the best results. More specifically, we would like to have

$$\int_\alpha^\beta (u - u_h^{(k)})^2 dx = e_\Omega^2.$$

We would also like to obtain some form of equidistribution of this error in the sense that it should be constant $|u - u_h^{(k)}| = c$ everywhere in the domain. Using the L^2-norm, this means that

$$e_\Omega^2 = \int_\alpha^\beta |u - u_h^{(k)}|^2 dx = c^2(\beta - \alpha) \text{ and thus } c^2 = \frac{e_\Omega^2}{(\beta - \alpha)}.$$

This also implies that on each element $K = [x_i, x_{i+1}]$, the local error e_K should satisfy

$$e_K^2 = \int_{x_i}^{x_{i+1}} |u - u_h^{(k)}|^2 dx = \frac{e_\Omega^2 h_K}{(b - a)}.$$

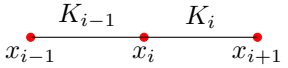
If this target value can be reached on each element, then the global error e_Ω will be attained on the whole interval $[\alpha, \beta]$. The mesh will therefore be modified in order to achieve this goal.

To obtain a new mesh, only local operations on the mesh (node displacement, node elimination and element refinement) are used. Element refinement and node elimination are used to reach the target level of error e_Ω. Node displacement is used to minimize the L^2-norm of the derivative of the error (the energy norm). Minimizing the norm of the derivative leads to an equidistribution of the error in the sense that if the norm of the derivative of the error is small, this means that the error is essentially constant. Moreover, this also leads to optimal position of the nodes as described in D'Azevedo and Simpson [17]. We end up with a mesh minimizing the error per unit length.

In practice, the true error is obviously replaced by the estimated error described in Section 3.2. We now describe the local operations on the mesh. We will present the situation in the linear case ($k = 1$) and we will therefore use formula (8) and (9) for the L^2-norm and H^1-seminorm of the estimated errors. In the quadratic case, these formulas are replaced by (10) and (11) respectively.

4.1. Node elimination. Node elimination is applied only to nodes at the boundary between two elements. It is not applied to nodes inside an element such as the mid-element node in a quadratic discretization. The same is true for node displacement.

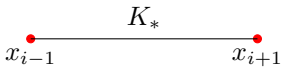
For each node of the mesh, we consider the two adjacent elements:



If the sum of the estimated errors is smaller than the sum of the local target errors

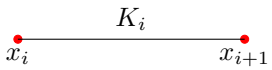
$$\frac{h_{K_{i-1}}^3}{120} \mid g_i^{(1)} - g_{i-1}^{(1)} \mid^2 + \frac{h_{K_i}^3}{120} \mid g_{i+1}^{(1)} - g_i^{(1)} \mid^2 < \frac{e_\Omega^2}{\beta - \alpha}(h_{K_{i-1}} + h_{K_i}),$$

the node is removed and a new element K_* is created (the elements should be renumbered)



Otherwise, nothing is done and we move on to the next node. In practice, the nodes of the mesh are swept many times until no node is eliminated or until the maximum number of iterations is attained.

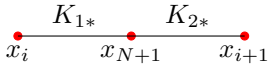
4.2. Element division. On an element K_i:



if the estimated error is larger than the local target error

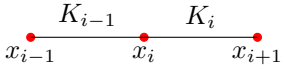
$$\frac{h_{K_i}^3}{120} \mid g_{i+1}^{(1)} - g_i^{(1)} \mid^2 > \frac{e_\Omega^2}{\beta - \alpha} h_{K_i},$$

then a new node is created and thus the element is divided into two new elements K_{1*} and K_{2*} (the elements should be renumbered):



Otherwise, nothing is done and we move on to the next element. At the new node, the nodal value $g_{N+1}^{(k)}$ is obtained by interpolation using $g_h^{(k)}(x)$. In practice, the elements of the mesh are swept many times until no node is created or until the maximum number of iterations is attained.

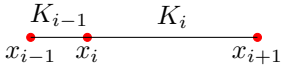
4.3. Node displacement. For each interior node x_i, we consider once again the patch consisting of two adjacent elements:



The node is moved in order to minimize the H^1-seminorm of the error on the patch

$$\min_{x_i} \left(\frac{h_{K_{i-1}}}{12} \mid g_i^{(1)} - g_{i-1}^{(1)} \mid^2 + \frac{h_{K_i}}{12} \mid g_{i+1}^{(1)} - g_i^{(1)} \mid^2 \right),$$

using an optimal descent or a Fibonacci method for instance.



At the new position, the nodal value of $g_i^{(k)}$ is simply interpolated using $g_h^{(k)}(x)$. Here again, the nodes of the mesh are swept many times until no node is moved or until the maximum number of iterations is reached.

The above local operations on the mesh are systematically used in the heuristic Algorithm 4.1. Step 3, in particular, can take many forms depending on the order chosen to perform the different local operations on the mesh. In practice, we start with element division followed by node displacement and then, node elimination, followed by another node displacement. For each operation, the nodes (or the elements) are swept many times until there is no more change in the mesh. The starting point is, unless otherwise specified, a more or less refined uniform mesh. Obviously, starting from two different initial meshes will result in very similar meshes but not exactly the same ones. Despite this, the algorithm seems very robust as will be demonstrated in the following section.

Algorithm 4.1 GLOBAL MESH ADAPTATION

INPUT: Initial mesh $\mathcal{M}_0$, maximum number of iterations max_it, e_Ω.
OUTPUT: Final mesh and computed solution.
$i = 0$;
while Adaptation process modified the mesh and $i \neq max\_it$ **do**
STEP 1 : Solve 1 to obtain a solution $u_h^{(k)}$ on mesh $\mathcal{M}_i$;
STEP 2 : Use the derivative recovery method to compute $g_h^{(k)}$ on mesh $\mathcal{M}_i$;
STEP 3 : Adapt the mesh $\mathcal{M}_i$ using local operations to obtain $\mathcal{M}_{i+1}$;
→ Minimize the H^1-seminorm of the error using node displacement (error equidistribution and element optimality);
→ Control the L^2-norm of the error (or the number of elements) using element refinement and node elimination;
STEP 4 : $i \leftarrow i + 1$;
end while

5. Numerical experiments

We conducted numerical experiments in order to assess the performance of our adaptive method. The global strategy is presented in Algorithm 4.1. First, we show that optimal convergence rates can be obtained as a function of the number of nodes (see Equation 4). More difficult problems such as singularly perturbed problems as described in Franz and Roos [9] are then tested.

5.1. Convergence rates. We start with a rather simple problem in order to verify that our adaptive method results in a significant gain in accuracy with respect to uniform meshes and that the convergence rates are those expected from the theory. We consider the problem

$$-u''(x) = -\frac{8ar^4x^2}{(1+r^2x^2)^3} + \frac{2ar^2}{(1+r^2x^2)^2}, \tag{12}$$

in the interval $[-5, 5]$, with $a = 1$ and $r = 10$. The analytical solution (see Figure 2) is

$$u(x) = \frac{a}{(1+r^2x^2)},$$

where the parameter a gives the maximum value of u while r controls the steepness of u in the neighborhood of the origin. A large value of r gives a steeper variation of u in a narrower interval around $x = 0$. In the numerical results, we set $a = 1$ and $r = 10$. Appropriate Dirichlet boundary conditions are applied at both ends of the interval. A standard Galerkin formulation was used in that case (no SUPG).

We first use uniform meshes and the results are summarized in Table 1 for different norms as a function of the number of DOFs. The L^2-norm and H^1-seminorm are computed as usual while for the L^∞-norm, each element is subdivided into 10 subintervals and the maximum error is evaluated on this partition i.e. not only at the computational nodes. The number of DOFs is also indicated. Recall that for quadratic elements, this number includes mid-element nodes and is basically twice the number of elements.

For the linear approximation $u_h^{(1)}$, quadratic and linear convergence rates are observed in L^2-norm and H^1-seminorm respectively while cubic and quadratic rates are observed for the quadratic solution $u_h^{(2)}$. We have also indicated the L^∞-norm to present a more complete picture.

TABLE 1. Equation 12: Error in different norms for uniform meshes

Linear solution ($k = 1$)			
Exact L^2norm	Exact H^1-seminorm	L^∞-norm	Number of DOFs
1.0×10^{-2}	6.7×10^{-1}	4.2×10^{-2}	200
2.7×10^{-3}	3.5×10^{-1}	1.4×10^{-2}	400
6.9×10^{-4}	1.8×10^{-1}	3.8×10^{-3}	800
1.7×10^{-4}	8.8×10^{-2}	9.6×10^{-4}	1600
4.3×10^{-5}	4.3×10^{-2}	2.4×10^{-4}	3200
1.1×10^{-5}	2.2×10^{-2}	6.1×10^{-5}	6400
Quadratic solution ($k = 2$)			
8.5×10^{-3}	5.5×10^{-1}	2.8×10^{-2}	200
9.8×10^{-4}	1.3×10^{-1}	4.1×10^{-3}	400
1.2×10^{-4}	3.1×10^{-2}	5.6×10^{-4}	800
1.5×10^{-5}	7.7×10^{-3}	7.3×10^{-5}	1600
1.9×10^{-6}	1.9×10^{-3}	9.1×10^{-6}	3200
2.3×10^{-7}	4.8×10^{-4}	1.1×10^{-6}	6400

For our adaptive method, according to Algorithm 4.1, we give as input a target error e_Ω and a maximum number of iterations $max\_it = 10$. The initial mesh $\mathcal{M}_0$ is a 100 element uniform mesh.

Table 2 presents the evolution of the errors for the adapted meshes. The target error e_Ω and the final estimated error in L^2-norm are also indicated. Clearly, as the number of DOFs increases, the estimated and exact errors become very close showing the efficiency of the estimator.

Figure 3 compares these L^2-errors for both uniform and adapted meshes on a log-log scale. Clearly, the adapted meshes presents the same (optimal) rates of convergence as the uniform meshes but with an error level almost two orders of magnitude smaller in the linear case and three in the quadratic case. Important gains are also observed for the H^1-seminorm. These differences can be made even more spectacular by increasing the value of the parameter r in the problem definition.

As an example, the numerical solutions (linear and quadratic) are illustrated in Figure 4 for a target error $e_\Omega = 10^{-4}$. As expected, for the same target error, the quadratic solution requires a much smaller number of DOFs (81 instead of 171), that is 40 quadratic elements versus 170 linear elements.

5.2. Finding a needle in a haystack. We consider the problem

$$-u''(x) = \sin(x) - \frac{8ar^4(x-\frac{\pi}{2})^2}{(1+r^2(x-\frac{\pi}{2})^2)^3} + \frac{2ar^2}{(1+r^2(x-\frac{\pi}{2})^2)^2}, \tag{13}$$

whose solution is

$$u(x) = \sin(x) + \frac{a}{(1+r^2(x-\frac{\pi}{2})^2)}. \tag{14}$$

Here again, a standard Galerkin method is used. We will work in the interval $[0, \pi]$ with Dirichlet boundary conditions provided by (14). We will also set $a = 0.001$ and $r = 10\,000$ such that the solution is a sine function plus a very small amplitude

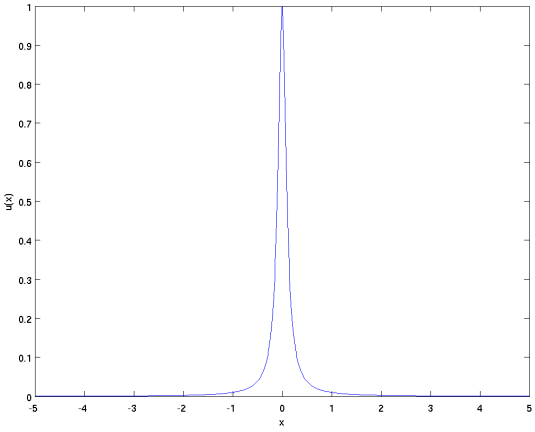


FIGURE 2. Equation 12: Analytical solution $u(x) = \frac{1}{1+100x^2}$

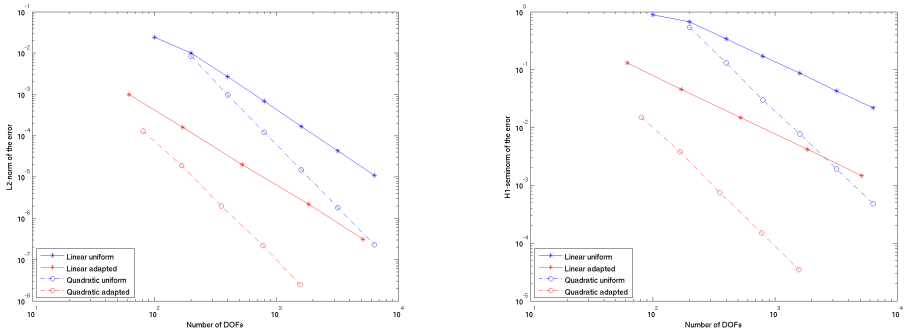


FIGURE 3. Equation 12: Convergence rates for uniform and adapted meshes

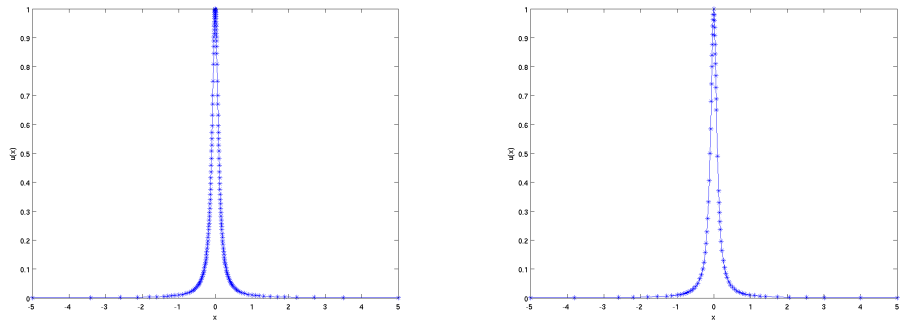


FIGURE 4. Equation 12: Numerical solutions with linear and quadratic adapted meshes ($e_\Omega = 10^{-4}$)

TABLE 2. Equation 12: Error in different norms for adapted meshes

Linear adaptation ($k = 1$)					
Target L^2-norm	Estimated L^2-norm	Exact L^2norm	Exact H^1-seminorm	L^∞-norm	Number of DOFs
1.0×10^{-3}	2.0×10^{-3}	1.1×10^{-3}	1.3×10^{-1}	2.2×10^{-3}	62
1.0×10^{-4}	2.1×10^{-4}	1.7×10^{-4}	4.6×10^{-2}	1.9×10^{-4}	171
1.0×10^{-5}	2.0×10^{-5}	2.0×10^{-5}	1.5×10^{-2}	2.5×10^{-5}	527
1.0×10^{-6}	2.3×10^{-6}	2.3×10^{-6}	4.2×10^{-3}	2.5×10^{-6}	1844
1.0×10^{-7}	3.1×10^{-7}	3.1×10^{-7}	1.5×10^{-3}	2.5×10^{-7}	5163
Quadratic adaptation ($k = 2$)					
1.0×10^{-4}	1.5×10^{-4}	1.3×10^{-4}	1.5×10^{-2}	2.6×10^{-4}	81
1.0×10^{-5}	1.7×10^{-5}	1.9×10^{-5}	3.8×10^{-3}	8.0×10^{-5}	167
1.0×10^{-6}	2.0×10^{-6}	2.0×10^{-6}	7.4×10^{-4}	3.5×10^{-6}	355
1.0×10^{-7}	2.2×10^{-7}	2.2×10^{-7}	1.5×10^{-4}	3.5×10^{-7}	777
1.0×10^{-8}	2.5×10^{-8}	2.5×10^{-8}	3.5×10^{-5}	3.5×10^{-8}	1653

perturbation centered at $x = \pi/2$ (barely visible) of the exact same form as in Section 5.1. Taking $r = 10\,000$ forces the perturbation to take the form of a needle.

As can be expected, uniform meshes have some difficulties capturing this solution. The particular form of the right-hand side in Equation 13 introduces a very strong perturbation that cannot be properly captured unless the mesh is very fine in the neighborhood of $x = \pi/2$. Figure 5 presents different intermediate solutions in comparison with the analytical solution together with the respective number of DOFs. Note that the perturbation in the solution (see the second term in Equation 14) is visible only on the finest mesh. It therefore takes 32 000 uniform quadratic elements (64 001 DOFs) to correctly represent the solution. Some of the intermediate solutions are completely wrong.

This problem can be seen as having two scales since the L^2-norm of the solution is 1.25 while the L^2-norm of the perturbation is around 10^{-5}. It is therefore necessary to use a target value e_Ω for the L^2-norm of the error much smaller than 10^{-5}. The iterative process was started with a 100 element uniform mesh and thus the size of the elements ($h = \pi/100$) is much larger than the width of the perturbation. Using $e_\Omega = 10^{-7}$ produces a solution where the perturbation is clearly visible with only 223 DOFs (not illustrated). Figure 6 presents the results for $e_\Omega = 10^{-9}$ resulting in a 501 element mesh (1003 DOFs). The L^2-norm of the error on this adapted mesh is similar to the most refined uniform mesh. For a similar accuracy, the adapted mesh requires 64 times less DOFs.

It is worth mentioning that our adaptive method is also an iterative method whose unknowns are the nodes of the mesh. Convergence is therefore not guaranteed. Interestingly, in this example the intermediate adapted meshes also passes through completely erroneous intermediate solutions similar to those in Figure 5 but ends up with a very good solution. This shows some robustness in the iterative process.

5.3. Failure of the SUPG method. We consider the following problem (see [9])

$$-\epsilon u''(x) + \left(x - \frac{1}{2}\right) u'(x) = x - \frac{1}{2}, \tag{15}$$

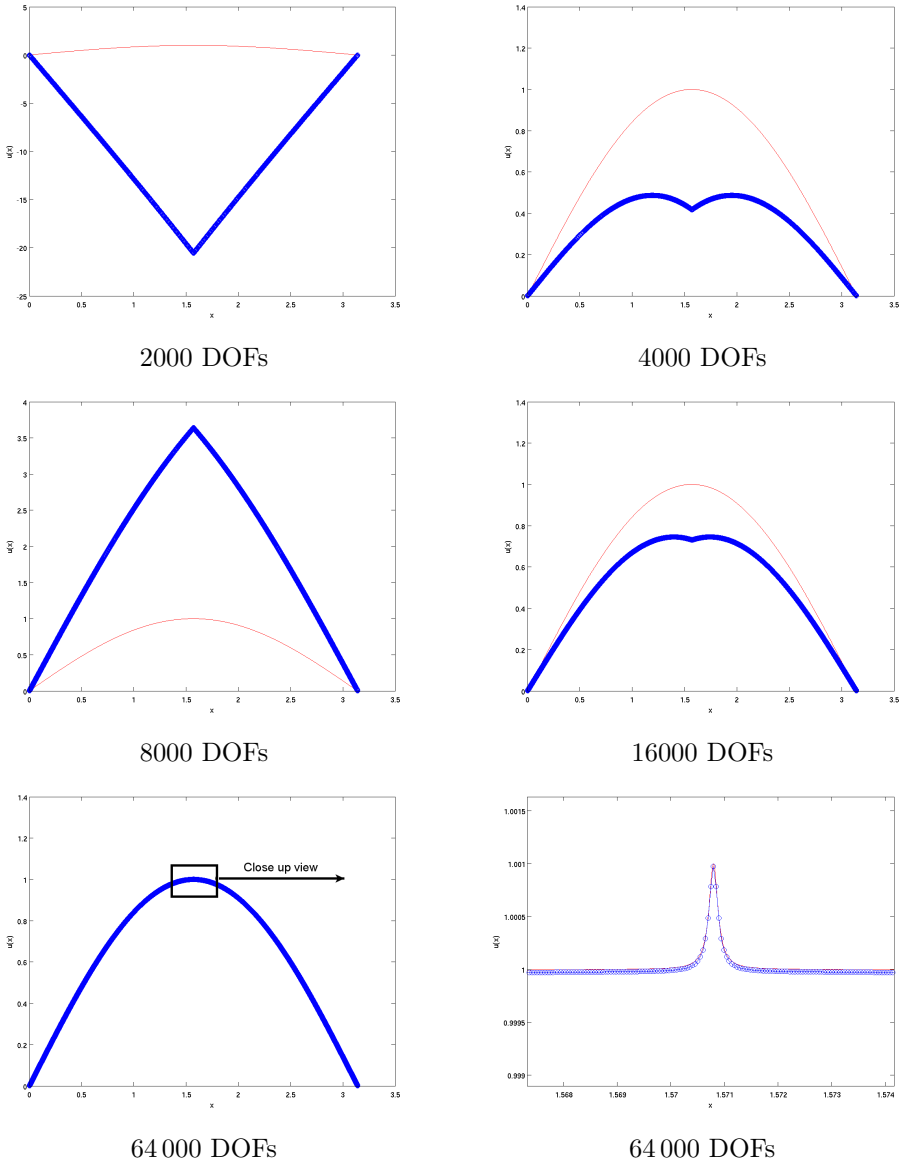


FIGURE 5. Equation 13: Analytical (red) and quadratic numerical solutions (blue) on uniform meshes

with $u(0) = -1/2$ and $u(1) = 1/2$. The solution is simply $u(x) = x - \frac{1}{2}$. Note that the exact solution belongs to the discrete finite element space and therefore, we should expect to obtain the exact solution when solving. However, the change of sign of the coefficient of $u'(x)$ (the convection term) forces the development of artificial boundary layers at both ends of the domain.

When the number of elements is small (< 20) both Galerkin and SUPG methods give more or less the exact solution. The error is however larger for the SUPG method. Increasing the number of elements, one gets erroneous solutions such as

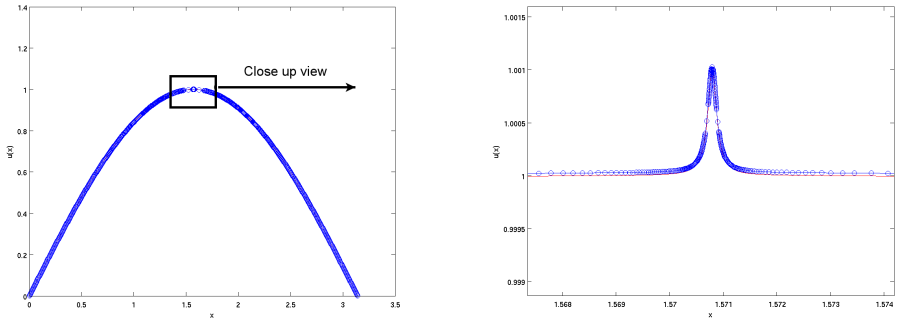


FIGURE 6. Equation 13: Analytical (red) and quadratic numerical solutions (blue) on the adapted mesh

TABLE 3. Equation 15: L^2-norm of the error on uniform meshes

	$\lvert u - u_h \rvert_0$	
DOFs	Galerkin	SUPG
32	3.9×10^{-15}	2.1×10^{-11}
64	1.5×10^{-8}	6.8×10^{-2}
128	1.5×10^{-2}	1.5×10^{-2}
256	1.4×10^{-3}	1.4×10^{-3}
512	3.6×10^{-3}	3.6×10^{-3}
1024	2.6×10^{-3}	2.6×10^{-3}
2048	1.8×10^{-3}	1.8×10^{-3}
4096	2.0×10^{-3}	2.0×10^{-3}
8192	1.4×10^{-3}	1.4×10^{-3}

the one presented in Figure 7 with 32 elements (65 DOFs) obtained using the SUPG method. The numerical solution is essentially linear but artificial boundary layers appears at both extremities. Both methods (Galerkin and SUPG) hardly converge as can be seen in Table 3.

This is therefore a case where both Galerkin and SUPG methods fail. As explained in [9], this behavior is due to the ill-conditioning of the corresponding linear systems. For a small number of elements, the conditioning of the Galerkin system is slightly better but as the number of elements increases, both systems are equally ill-conditioned and they give similar results.

Fortunately, our mesh adaptation corrects the situation. Starting from the erroneous solution illustrated in Figure 7, it first removes all unnecessary nodes where the numerical solution is linear thus decreasing the number of elements and the condition number of the matrix. It then provides the exact solution (with only one element!) in one adaptation step. In this specific case, mesh adaptation serves as a handrail for the numerical method.

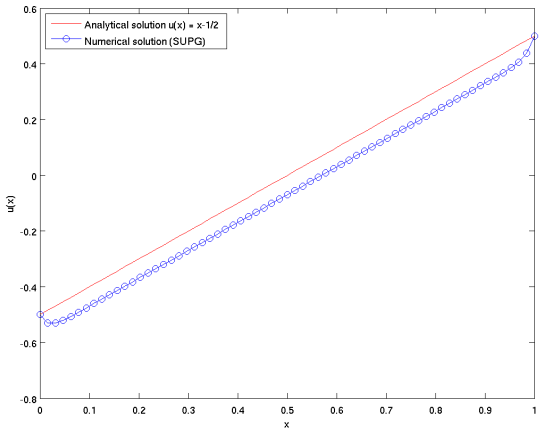


FIGURE 7. Equation 15: Analytical (red) and quadratic numerical solution (blue) with 32 quadratic elements (65 DOFs)

5.4. Singularly perturbed problem. Our next example was also proposed in [9] and takes the form

$$-\epsilon u''(x) - u'(x) = -2\epsilon - 2x, \tag{16}$$

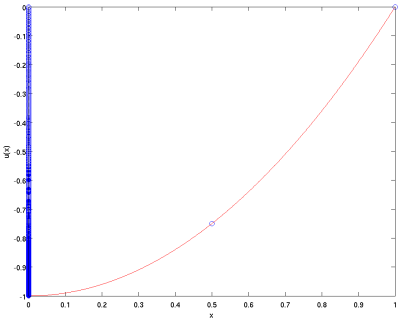
with $u(0) = u(1) = 0$. The solution is

$$u(x) = x^2 - 1 + \frac{\exp(-x/\epsilon) - \exp(-1/\epsilon)}{1 - \exp(-1/\epsilon)}.$$

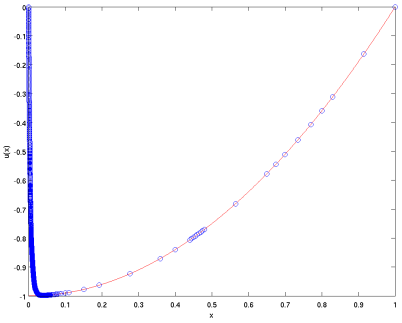
We will consider the case where $\epsilon = 10^{-8}$ for which the solution presents a very steep variation at $x = 0$. This is a classical problem with a boundary layer and where the Galerkin formulation fails and some stabilization method such as SUPG is necessary.

Figure 8 (top) presents a solution on a quadratically adapted mesh with 393 elements (787 DOFs) obtained using a target error $e_\Omega = 10^{-9}$. The exact L^2-norm of the error in this particular case is 1.16×10^{-11} and the L^∞-norm of the error is 1.11×10^{-8}. To obtain a comparable accuracy, "hand-adapted" meshes such as those proposed in Shishkin [18] with more than 10^5 nodes was necessary in [9]. In the boundary layer, the length of the elements varies from 6×10^{-11} to 6×10^{-9}. It then rapidly increases up to the last element whose length is 0.992. Such a large element is made possible by the fact that the analytical solution is essentially parabolic apart from the boundary layer and our quadratic finite element approximation can represent it almost exactly with a single element. We have also illustrated in Figure 8, the first and third adapted meshes. The first adapted mesh does not capture the boundary layer correctly and presents a large number of nodes in the parabolic region. The third intermediate mesh is already very satisfactory and afterward, the adaptation method does not change the mesh significantly.

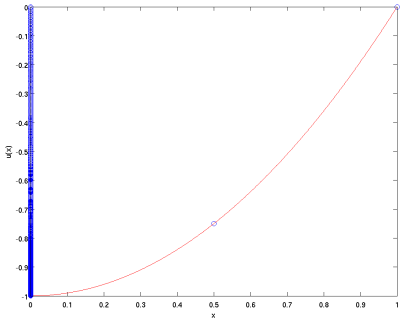
Our adaptive method automatically provides a better mesh than those proposed by Shishkin [18] and this, without any *a priori* knowledge of the solution or the position of the boundary layer. The convergence rates in the different norms are presented in Figures 9 and 10 showing optimal rates of convergence in all cases. Similar results are obtained (but not shown) for linear adaptation.



Final mesh with 393 elements (787 DOFs)



First intermediate adapted mesh



Third intermediate adapted mesh

FIGURE 8. Equation 16: Analytical (red) and quadratic numerical solutions (blue) on different meshes

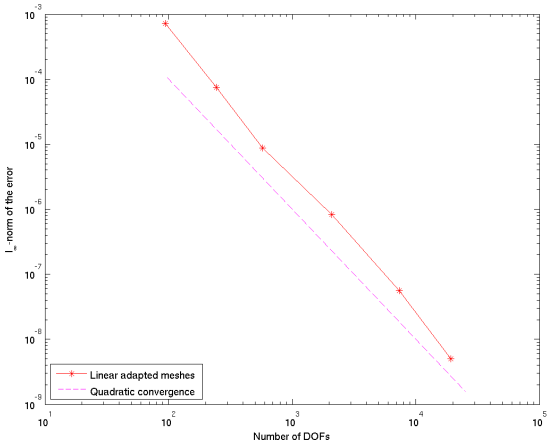


FIGURE 9. Equation 16: Convergence in L^∞-norm.

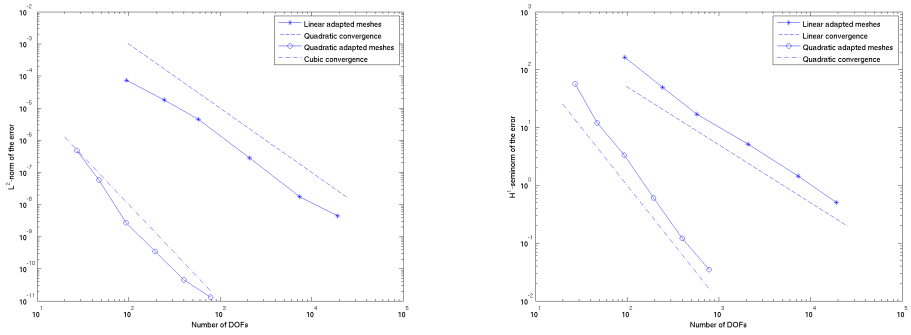


FIGURE 10. Equation 16: Convergence in L^2-norm and H^1-seminorm.

5.5. A discontinuous solution. We now consider a first order equation:

$$-xu'(x) = xe^x, \tag{17}$$

with $u(-1) = 0$ and $u(1) = 0$. The solution is characterized by a discontinuity (a shock) at $x = 0$:

$$u(x) = \begin{cases} e^{-1} - e^x & \text{if} \quad x < 0, \\ e^1 - e^x & \text{if} \quad x > 0. \end{cases}$$

Using a SUPG formulation, we have obtained the piecewise quadratic solution illustrated on top of Fig 11 with 691 elements (1383 DOFs). The solution is not bad at all but small amplitude oscillations are still visible at the discontinuity. The L^2 norm of the error is around 1.8×10^{-6} since we used a target error $e_\Omega = 10^{-6}$.

The fact that our mesh adaptation method does not give entirely satisfactory results should not be a surprise. Recall that the method is based on a reconstruction of the derivative of the numerical solution, which in this particular case, does not exist at the discontinuity. The results are nevertheless surprisingly good.

We now consider a slightly perturbed problem of the form

$$-\epsilon u''(x) - xu'(x) = xe^x, \tag{18}$$

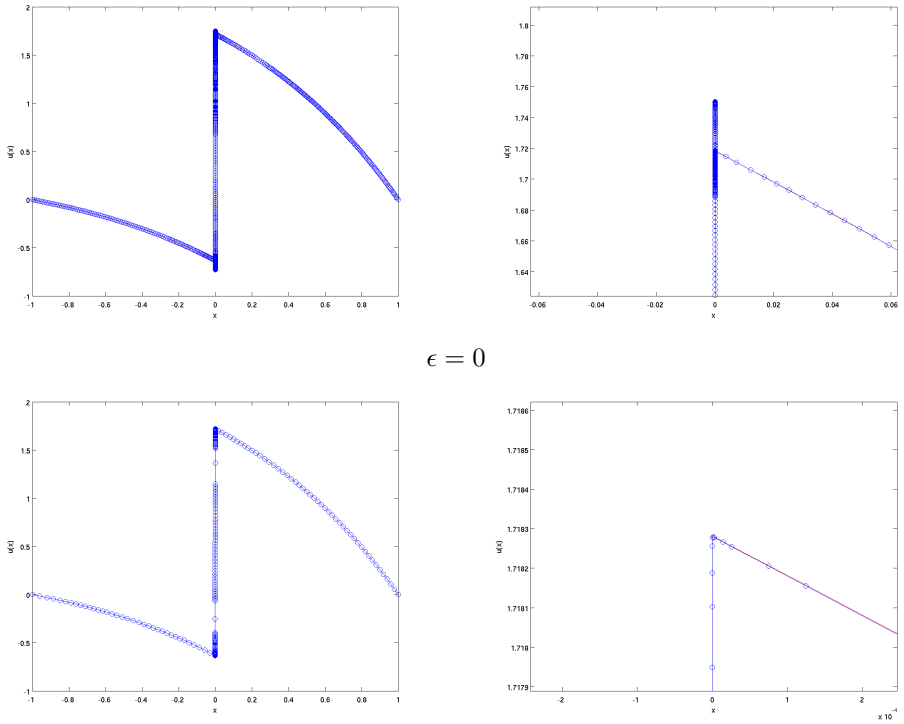
with a very small value for ϵ (10^{-15}). The solution of this perturbed problem is now continuous and differentiable, though a very steep variation is present at $x = 0$. The results are presented in Figure 11. Here again, these results are very satisfactory taking into account that computing a discontinuous solution within a space of continuous functions is a difficult challenge. As stated in [9], a discontinuous Galerkin method is more likely to give a better solution at the express condition that a node is provided at the discontinuity. This again requires an *a priori* knowledge of the solution. Our adaptive method provides a very good compromise and does not necessitate such information.

5.6. Exponentially ill-conditioned problem. Our last problem is a very difficult test case:

$$-\epsilon u''(x) + xu'(x) = 0, \tag{19}$$

with $u(-1) = -3$ and $u(1) = 1$. The exact solution is

$$u(x) = -1 + \frac{2\int_0^x e^{s^2/2\epsilon}\, ds}{\int_0^1 e^{s^2/2\epsilon}\, ds}.$$



$\epsilon = 0$

$\epsilon = 10^{-15}$

FIGURE 11. Equation 18: Approximation of a discontinuous solution

The simplicity of the differential equation is misleading. As can be seen, even evaluating with sufficient accuracy the integrals appearing in the exact solution is not a trivial task. For this reason, we will present the H^1-seminorm of the error which requires only the computation of the derivative

$$u'(x) = \frac{2e^{x^2/2\epsilon}}{\int_0^1 e^{s^2/2\epsilon}\, ds}.$$

The difficulty of the problem was explained in O'Malley and Ward [19] where it is shown that the continuous problem has an asymptotically exponentially small leading eigenvalue behaving as

$$\lambda_0 \sim \sqrt{\frac{2}{\pi\epsilon}} e^{-1/2\epsilon}.$$

To give an idea, for $\epsilon = 0.04, 0.03, 0.02$ and 0.01, this gives respectively $\lambda_0 \sim 10^{-5}$, 10^{-7}, 10^{-11} and 10^{-21}. Obviously, the associated discrete problem inherits this property making its numerical solution extremely ill-conditioned.

We first start with uniform meshes. The analytical solution presents boundary layers at both $x = -1$ and $x = 1$ and uniform meshes are not likely to perform well. As can be seen in Table 4, quadratic convergence is clearly obtained for $\epsilon = 0.04$ and for $\epsilon = 0.03$ up to 3200 DOFs. For $\epsilon = 0.02$ we get non optimal convergence and only for small numbers of DOFs. Finally, we were not able to obtain a valid

TABLE 4. Equation 19: Error in different norms for uniform meshes

	$\|u - u_h\|_1$		
Number of DOFs	$\epsilon = 0.04$	$\epsilon = 0.03$	$\epsilon = 0.02$
200	8.8×10^{-2}	1.8×10^{-1}	4.9×10^{-1}
400	2.2×10^{-2}	4.6×10^{-2}	1.3×10^{-1}
800	5.6×10^{-3}	1.1×10^{-2}	5.2×10^{-2}
1600	1.4×10^{-3}	2.9×10^{-3}	1.8×10^{-2}
3200	4.5×10^{-4}	9.5×10^{-4}	8.5×10^{-1}
6400	8.8×10^{-5}	2.4×10^{-3}	

solution in the case $\epsilon = 0.01$. These results are similar to what was observed in Sun [20].

This problem was really a test for our mesh adaptation method. Contrarily to all the preceeding tests, convergence was not easily obtained and was linked to the smallest eigenvalue of the problem. Indeed, the target value e_Ω given as an input to our method had to be chosen smaller than λ_0 to obtain convergence. For example, in the case $\epsilon = 0.04$ ($\lambda_0 \sim 10^{-4}$), no convergence was possible unless e_Ω was chosen smaller than 10^{-5}. Optimal convergence rate was then obtained as can be seen in Figure 12. For a target error as small as 10^{-12}, the L^2-norm of the error is so small on each element that we are scratching the limits of machine precision available.

An optimal convergence rate was also observed for $\epsilon = 0.03$ ($\lambda_0 \sim 10^{-7}$) but for a narrower range of values of the number of DOFs using a target error between 10^{-8} and 10^{-12} .

Finally, the situation is not very clear for $\epsilon = 0.02$ ($\lambda_0 \sim 10^{-11}$) where we had to use a target error between 10^{-11} and 10^{-12} to get a solution. We are already flirting with the limits of double digits accuracy. Finally, despite numerous efforts, we were not able to get a correct solution for $\epsilon = 0.01$. Numerical solutions could be obtained but were far from the exact solution and were simply discarded. If our hypothesis is correct, quadruple precision would be necessary to get a correct behaviour. This was also the conclusion in [20]. We have definitely reached the limits of our adaptive method with respect to machine precision at our disposal.

6. Conclusions

In this work, we have introduced a novel mesh adaptive strategy in the one-dimensional case. The method was tested on various problems to illustrate its potential but also to underline some of its limitations. In particular, the method was shown to perform very well for boundary layer problems. The overall iterative method whose unknowns are the mesh nodes also shows a certain robustness in the sense that the final adapted mesh does not strongly depend on the initial mesh and that, even when starting with a clearly inappropriate mesh, we can get a satisfactory adapted mesh.

The proposed adaptive method is also implemented in two and three dimensions (see [1, 2]) and we are currently investigating if similar results can be obtained for higher dimensional boundary layer problems.

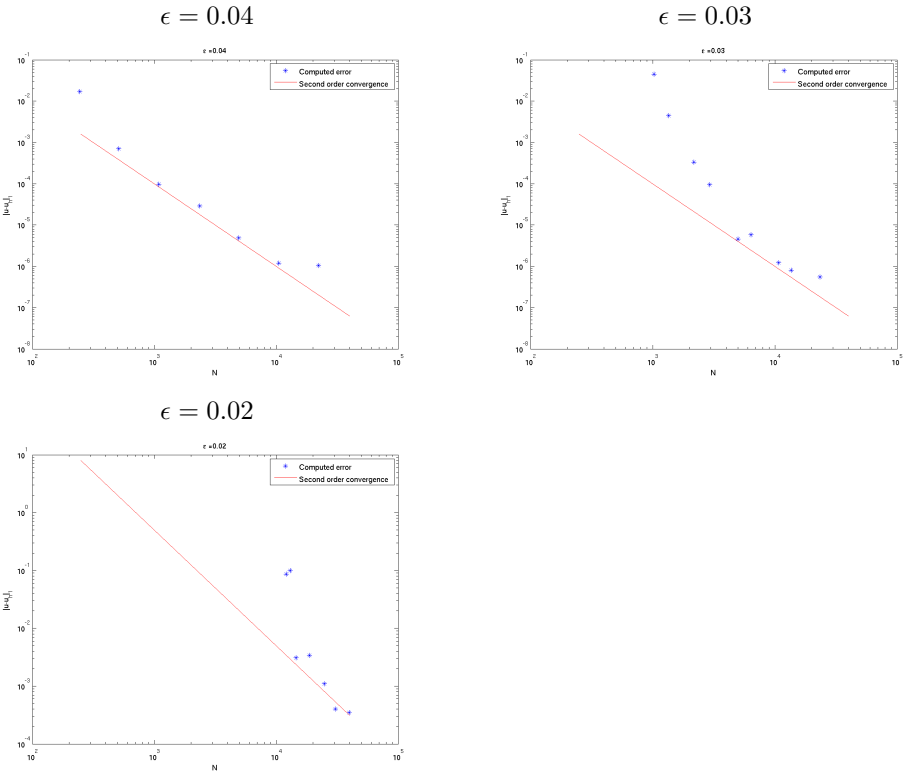


FIGURE 12. Equation 19: Convergence in H^1-seminorm for the adapted meshes.

Acknowledgments

The authors wish to acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] R. Bois, M. Fortin, and A. Fortin. A fully optimal anisotropic mesh adaptation method based on a hierarchical error estimator. *Computer Methods in Applied Mechanics and Engineering*, 209-212:12–27, february 2012.

[2] R. Bois, M. Fortin, A. Fortin, and A. Couët. High order optimal anisotropic mesh adaptation using hierarchical elements. *European Journal of Computational Mechanics/Revue Européenne de Mécanique Numérique*, 21(1-2):72–91, 2012.

[3] F. Alauzet. High-order methods and mesh adaptation for euler equations. *International Journal for Numerical Methods in Fluids*, 56(8):1069–1076, 2008.

[4] Y. Belhamadia, A. Fortin, and É. Chamberland. Anisotropic mesh adaptation for the solution of the Stefan problem. *Journal of Computational Physics*, 194(1):233–255, 2004.

[5] Y. Belhamadia, A. Fortin, and É Chamberland. Three-dimensional anisotropic mesh adaptation for phase change problems. *Journal of Computational Physics*, 201(2):753–770, 2004.

[6] W. G. Habashi, J. Dompierre, Y. Bourgault, D. Ait Ali Yahia, M. Fortin, and M.-G. Vallet. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part I: General principles. *International Journal for Numerical Methods in Fluids*, 32:725–744, 2000.

[7] M. Hecht and B. Mohammadi. Mesh adaptation by metric control for multi-scale phenomena and turbulence. In *35th Aerospace Sciences Meeting & Exhibit*, number 97–0859, Reno, USA, 1997.

[8] I. Babuska and W.C. Rheinboldt. A-posteriori error estimates for the finite element method. *International Journal for Numerical Methods in Engineering*, 12:1597–1615, 1978.

[9] S. Franz and H.-G. Roos. The capriciousness of numerical methods for singular perturbations. *SIAM review*, 53(1):157–173, 2011.

[10] T. J. R. Hughes and A. N. Brooks. *A multidimensional upwind scheme with no crosswind diffusion*, volume 34 of *Finite Element Methods for Convection Dominated Flows*, pages 19–35. Amer. Soc. of Mech. Eng., New York, 1979.

[11] T. J. R. Hughes and A. N. Brooks. A theoretical framework for Petrov-Galerkin methods with discontinuous weighting functions. Applications to the streamline upwind procedure. In R. H. Gallagher, editor, *Finite element in fluids*, volume IV. Wiley, London, 1982.

[12] H.-G. Roos. Stabilized FEM for convection-diffusion problems on layer-adapted meshes. *Journal of Computational Mathematics*, 27:266–279, 2009.

[13] H.-G. Roos, M. Stynes, and L. Tobiska. *Robust numerical methods for singularly perturbed differential equations*, volume 24 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, second edition, 2008. Convection-diffusion-reaction and flow problems.

[14] G. Zhou. How accurate is the streamline diffusion finite element method? *Mathematics of Computation*, 66(217):31–44, 1997.

[15] Z. Zhang and A. Naga. A new finite element gradient recovery method: Superconvergence property. *SIAM Journal for Scientific Computing*, 26(4):1192–1213, 2005.

[16] R. E. Bank and K. R. Smith. A posteriori error estimates based on hierarchical bases. *SIAM Journal on Numerical Analysis*, 30(4):921–935, 1993.

[17] E.F. D'Azevedo and R.B. Simpson. On optimal triangular meshes for minimizing the gradient error. *Numerische Mathematik*, 59(1):321–348, 1991.

[18] G. Shishkin and L. Shishkina. *Difference methods for singularly perturbed problems*. Chapman and Hall, Boca Raton, FL, 2009.

[19] R.E. O'Malley and M. J. Ward. Exponential asymptotics, boundary layer resonance, and dynamic metastability. In L. P. Cook et al. and V. Roytburd & M. Tulin, editors, *Mathematics is for solving problems*, pages 189–203. SIAM publication, 1996.

[20] X. Sun. Numerical analysis of an exponentially ill-conditioned boundary value problem with applications to metastable problems. *IMA Journal of Numerical Analysis*, 21:817–842, 2001.

Département de mathématiques et de statistique, 1045 avenue de la médecine, Université Laval, Québec, Canada, G1V 0A6,

E-mail: `afortin@giref.ulaval.ca`

URL: `http://www.giref.ulaval.ca/`