Available online at www.sciencedirect.com
SCIENCE DIRECT®

Comput. Methods Appl. Mech. Engrg. 194 (2005) 605–624

Computer methods in applied mechanics and engineering

www.elsevier.com/locate/cma

Fully automatic *hp*-adaptivity for Maxwell's equations

L. Demkowicz [*]

Institute for Computational Engineering and Sciences, The University of Texas at Austin, SHC 304, Campus Code C0200, 105 W. 26th Street, Austin, TX 78712, USA

Received 18 May 2004; accepted 18 May 2004

Abstract

I report on the development of a fully automatic *hp*-adaptive strategy for the solution of time-harmonic Maxwell equations. The strategy produces a sequence of grids that deliver exponential convergence for both regular and singular solutions. Given a (coarse) mesh, we refine it first globally in both h and p, and solve the problem on the resulting fine mesh. We consider then the projection-based interpolants of the fine mesh solution with respect to both current and next (to be determined) coarse grid, and introduce the interpolation error decrease rate equal to the difference of the old and new (coarse) mesh interpolation errors vs. number of degrees-of-freedom added. The optimal *hp*-refinements leading to the next coarse grid are then determined by maximizing the interpolation error decrease rate.

Keywords: *hp* edge finite elements; *hp*-adaptivity

1. Introduction

The paper presents a progress report on the development of a fully automatic *hp*-adaptive algorithm for both elliptic and Maxwell problems, the work is an extension and continuation of the ideas presented in [5,7]. The ultimate goal of the presented work is to combine *hp*-adaptivity with goal-oriented adaptivity, and apply it to radar scattering problems [10].

The mesh optimization algorithm builds on the idea of the projection-based interpolation [3,4]. The interpolation error exhibits the same convergence rates in both element size h and order of approximation p as the actual approximation error but, contrary to the approximation error, it depends upon the mesh only *locally*. Changes in the discretization over an element affect the value of the interpolation error for the element only. This local dependence enables the construction of the *hp* optimization algorithm.

[*] Tel.: +1 512 471 3312; fax: +1 512 471 8694.
E-mail address: leszek@ticam.utexas.edu

0045-7825/$ - see front matter © 2004 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2004.05.023

Obviously, we cannot interpolate the unknown exact solution, and we have to replace it with some approximation that is significantly more accurate that the Finite Element (FE) solution itself. The additional "scales" must allow for making the choice between *h* and *p*-refinements, so the idea of using the global *hp*-refinement, and the corresponding fine grid solution suggests itself.

Introducing the fine grid solution practically eliminates the need for an a posteriori error estimation—the difference between the fine and coarse grid solution provides an excellent approximation to the error function.

In 3D, the global *hp*-refinement typically increases the problem size by one order of magnitude, and the only feasible way to obtain the fine grid solution is to use multigrid techniques [2]. Our results for elliptic problems [8] indicate that it is sufficient to work with a partially converged fine grid solution only. The experiments indicate that it is essential to reduce the iteration error below 10% of the current error estimate. For elliptic problems, this typically requires 3–5 iterations only. In this context, the partially converged fine grid solution may be seen as the result of a special postprocessing applied to the current solution. On the other side, the fine and not the coarse grid solution may be seen as a final product of the mesh optimization. It is, therefore, important to be able to converge to the fine grid solution with a machine accuracy, if desired.

The presented work focuses on details of the algorithm and the corresponding numerical implementation. There is no proof that the algorithm delivers exponential convergence, in fact we cannot show even that it must converge at all. All claims made here are based on numerical experiments only, with the number of test cases being quite limited. It is in this context that we strive at least to achieve a certain generality of the algorithm—the same logic should apply to the 1D version and to both 2D (elliptic or Maxwell) versions of the algorithm. In the process of achieving this coherence we had to slightly modify our original version of the algorithm.

The outline of the paper is as follows. We will begin by recalling briefly the 1D version of the algorithm and discuss the case of a "missing scale". Next we present the 2D algorithm, the logic for both elliptic and Maxwell case being identical, only the norms will change. We then enter the discussion of "dirty details", and argue why we believe that our choices are optimal, illustrating the discussion with numerical results.

2. One-dimensional algorithm

We shall focus on the standard model problem—one-dimensional Poisson equation,

$$\begin{cases} u \in \tilde{u}_D + H^1_0(0,1), \\ \displaystyle\int_0^1 \frac{\mathrm{d}u}{\mathrm{d}x}\frac{\mathrm{d}v}{\mathrm{d}x}\,\mathrm{d}x = \int_0^1 fv\,\mathrm{d}x, \quad \forall v \in H^1_0(0,1). \end{cases}$$

Here $\tilde{u}_D \in H^1(0,1)$ denotes a lift of Dirichlet boundary data at $x = 0, 1$. The problem is solved using the standard Galerkin method on an *hp* mesh. Given an interval (a,b), a space $X_{hp}(a,b)$ of polynomials or piecewise polynomials defined on the interval, and a function $u \in H^1(a,b)$, we define the projection-based interpolant $w_{hp} = \Pi_{hp}u$ of function u, as the solution of the local approximate Dirichlet problem,

$$\begin{cases} w_{hp} \in X_{hp}(a,b), \\ w_{hp}(a) = u(a), \quad w_{hp}(b) = u(b), \\ \displaystyle\int_a^b 7\frac{\mathrm{d}(w_{hp}-u)}{\mathrm{d}x}\frac{\mathrm{d}v}{\mathrm{d}x}\,\mathrm{d}x = 0 \quad \forall v \in V_{hp}(a,b). \end{cases}$$

Here $V_{hp}(a,b)$ denotes the space of FE test functions on interval (a,b),

$$V_{hp}(a,b) = \{v_{hp} \in X_{hp}(a,b) : v_{hp}(a) = v_{hp}(b) = 0\}.$$

We shall interpolate either over a single element of order p, or an h-refined element with sons of orders p_1, p_2.

The one-dimensional example is very special. It is well known that the Galerkin solution of the original problem matches the exact solution at vertex nodes. Consequently, the projection-based interpolant coincides simply with the FE solution. The (projection-based) interpolation error and the approximation error are the same.

Given a (coarse) *hp* mesh, we first refine it globally in both h and p, i.e. each element is refined into two element sons, and the order of approximation is raised uniformly by one. Both the coarse and the fine mesh may be quite non-uniform but the refinement itself is uniform. This eliminates the need for maintaining two separate data structures.

We solve on the fine mesh to obtain the fine mesh solution $u_{h/2,p+1}$.

The determination of the optimal *hp* refinements involves now three steps.

Step 1: *For each element in the coarse mesh*, *choose between p and h refinement*, *and determine the* guaranteed *element rate.* The optimal refinement is found by staging a *competition* between the p-refinement and competitive h-refinements. The competitive h-refinements are those that result in the same increase in the number of degrees-of-freedom (d.o.f.) as the p-refinement—just one d.o.f. added. If the order of element sons is p_1, p_2, the condition is simply that $p_1 + p_2 = p + 1$. For an element of order p, the number of h competitors is p. We determine the interpolant corresponding to the original element, and next the interpolants for the p-refined element, and for each of the competitive h-refined meshes consisting of two elements. This enables computation of the corresponding *error decrease rates*,

$$\frac{\|u_{h/2,p+1} - w_{hp}\| - \|u_{h/2,p+1} - w_{hp}^{\text{new}}\|}{1}.$$

Here w_{hp} denotes the interpolant on the current mesh, and w_{hp}^{new} is the interpolant corresponding to any of the competing meshes. The errors are measured in the H_0^1 norms.

Next we determine the *guaranteed rate* with which the interpolation error must decrease over the element. In order to explain the concept, suppose for a moment that we interpolate the exact solution, and that we have infinite resources to check all possible refinements of the element. This would include checking all possible h-refinements into two *or more* elements, with arbitrary distribution of polynomial orders p. For each of those refinements, we would compute the corresponding decrease rate of the interpolation error, and take the supremum over all refinements. This would be the element *maximum rate*.

Obviously, we have neither the exact solution nor infinite resources. However, our fine grid solution carries more information that we have used so far. The *hp*-refinement of an element of order p, is adding $p + 2$ d.o.f. and we can check at least the rates corresponding to the h-refined element with orders p_1, p_2 for the sons resulting with more than one d.o.f. added,

$$\frac{\|u_{h/2,p+1} - w_{hp}\| - \|u_{h/2,p+1} - w_{hp}^{\text{new}}\|}{(p_1 + p_2 - 1) - (p - 1)}.$$

The motivation for checking the h-refinements beyond the competitive h-refinements comes from the *case of a missing scale*. A typical situation is depicted in Fig. 1. The error (function) in the second element is approximately an odd function, and raising order of the element from $p = 1$ to $p = 2$ (which results in adding an even shape function) does not decrease it. The only competitive h-refinement in this case—breaking the linear element into two linear elements adds also an even function (piecewise linear ''hat function''), and results in no significant decrease of the interpolation error either. The p-refinement wins but the rates corresponding to both competing refinements are practically zero. Based on these rates alone, the element will

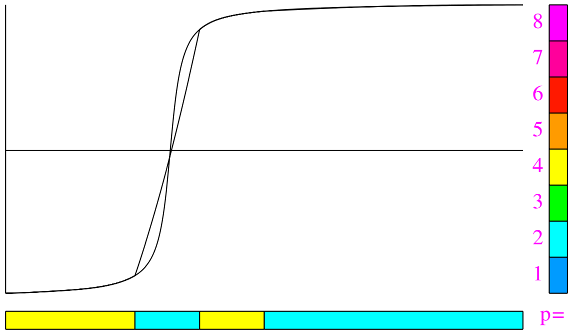

Fig. 1. Solution with a "missing scale".

never be selected for any refinement (see step 2 of the algorithm below) at all. Checking the h-refined element with orders $p_1, p_2 > 1$, reveals immediately the large error in the element, represented by large error decrease rates. The situation is not limited to linear elements and may occur for higher p as well.

In practice, we do not check all possible combinations of values p_1, p_2. This would be prohibitively expensive, especially in 2D and 3D. Instead, we start with the best competitive h-refinement (i.e. the one that has produced the biggest drop in the interpolation error) and continue increasing p following the *biggest element error refinement path*. Given the interpolant for a h-refined element with orders p_1, p_2, we determine the son contributions to the error and raise the order for the sons whose error is within 70% of the maximum error. The procedure, reflecting a typical p-refinement strategy, is illustrated in Fig. 2. The 70% factor is arbitrary and can be replaced with another value, typically between 50% and 90%. Ideally, one should increase the order only in the element contributing with the largest error, until the max value of $p + 1$ is reached and only then continue with the p-enrichment of the other element son. The 50–90% factor reflects a common practice in adaptivity (including p-adaptivity) aimed at minimizing the number of intermediate meshes considered. Increasing order in more than one element son when the error is almost equidistributed, has little effect on efficiency of the 1D algorithm, but it accelerates significantly the analogous procedure in 2D and 3D implementations.

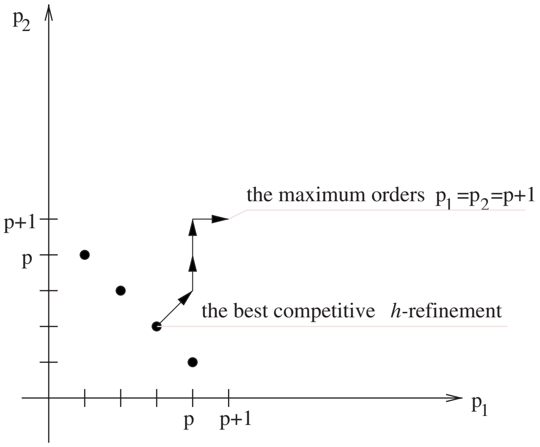
p2
p1
p+1
p
p
p+1
the maximum orders p1 =p2=p+1
the best competitive h-refinement

Fig. 2. Biggest element error refinement path.

We emphasize that obtaining a bigger error decrease rate with a non-competitive h-refinement does not mean that it represents an optimal refinement. We simply have not checked p-refinements corresponding to raising order p by more than one. Consequently, the choice between the p- and an h-refinement is made by comparing *only* the competitive refinements, but the rate is determined by checking all h-refinements possible. It is for this reason that we call the corresponding maximum rate the *guaranteed rate*. If the guaranteed rate is high, we know that we have to invest into the element but our profits eventually may actually be higher than predicted.

Step 2: *Determine which elements to refine.* Given the guaranteed rate for each element in the mesh, we determine the maximum rate for all elements,

$$\text{rate}_{\max} = \max_{K} \text{ (element } K \text{ rate)}.$$

All elements that produce a rate within 1/3 of the maximum rate, are selected for a refinement. The factor 1/3 is again somehow arbitrary.

Step 3: *Determine optimal order of approximation for elements to be refined.* We revisit now all elements selected for a refinement. If the p-refinement won, the element is simply p-refined. In the case of a winning h-refinement however, we set to determine the optimal distribution of orders p_1, p_2. Equipped with rate$_{\max}$ that decides about our investments, we follow the *biggest element error refinement path* again. This time, however, we exit the path at the moment the element error decrease rate drops below 1/3 of rate$_{\max}$, consistently with our global investment policy. Implicit in the procedure is the assumption that the element rate decreases monotonically with the number of d.o.f. added. The procedure is illustrated in Fig. 3.

This completes determining the next optimal coarse grid. We perform the next global hp-refinement, and continue until a desired error tolerance level is met. The error is estimated by comparing simply the coarse and fine grid solutions,

$$\frac{\|u_{hp} - u_{h/2,p+1}\|}{\|u_{h/2,p+1}\|} \times 100\% \leqslant \text{requested tolerance in percent.}$$

While the fine grid solution is always used for the stopping criterion, in the case when the exact solution is known, we use it in place of the fine grid solution to report the convergence history.

The described procedure represents a slight modification of the original hp algorithm presented in [5]. Results obtained with the modified algorithm for the 1D model problems presented in [5] are practically identical with those in [5], and we will not reproduce them here.

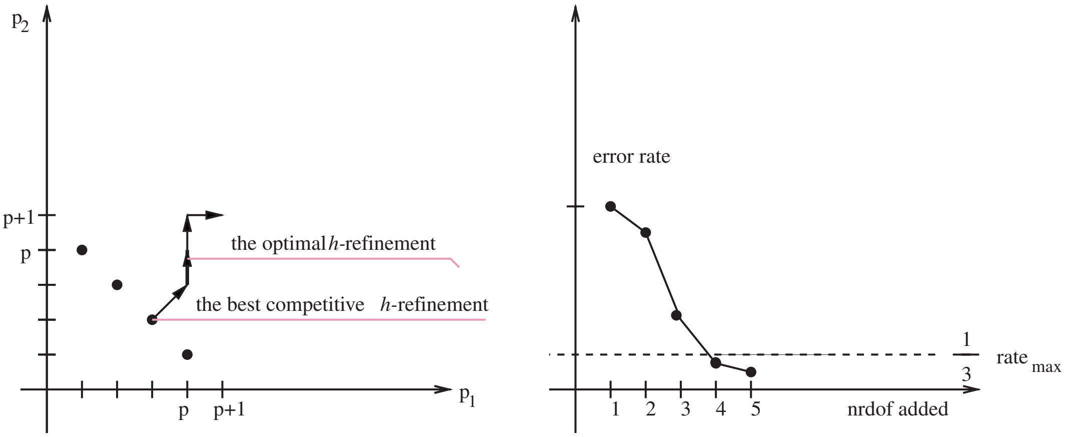

Fig. 3. Investing into an h-refined element.

3. The 2D algorithm

We shall focus on two classes of problems.

3.1. A 2D elliptic problem

$$\begin{cases} u \in \tilde{u}_D + V, \\ \int_\Omega \left(a_{ij} \frac{\partial u}{\partial x_i} \frac{\partial v}{\partial x_j} + b_j \frac{\partial u}{\partial x_j} v + cuv \right) \mathrm{d}\boldsymbol{x} + \int_{\Gamma_\mathrm{C}} \beta uv \, \mathrm{d}s = \int_\Omega fv \, \mathrm{d}\boldsymbol{x} + \int_{\Gamma_\mathrm{N} \cup \Gamma_\mathrm{C}} gv \, \mathrm{d}s, \quad \forall v \in V. \end{cases}$$

Here Ω is a bounded domain in $\mathbb{R}^2$, Γ_D, Γ_N, Γ_C denote the three disjoint parts of its boundary where Dirichlet, Neumann and Cauchy boundary conditions are imposed, $a_{ij}(\boldsymbol{x})$, $b_j(\boldsymbol{x})$, $c(\boldsymbol{x})$, $\beta(\boldsymbol{x})$ are the material data, and $f(\boldsymbol{x})$, $g(\boldsymbol{x})$ are the load data. Function $\tilde{u}_D \in H^1(\Omega)$ denotes a lift of Dirichlet boundary condition data, and we use the summation convention. Finally, V denotes the space of test functions,

$$V = \{v \in H^1(\Omega) : v = 0 \text{ on } \Gamma_\mathrm{D}\}.$$

The material data satisfy the standard regularity and ellipticity assumptions. The problem can easily be generalized to a general elliptic system of second order equations, with the linear elasticity problem being a prime example. For typical wave propagation problems, the data and solution are complex-valued.

3.2. Time-harmonic 2D Maxwell problem

$$\begin{cases} \boldsymbol{E} \in \tilde{\boldsymbol{E}}_D + \boldsymbol{W}, \\ \int_\Omega \left(\frac{1}{\mu} (\boldsymbol{\nabla} \times \boldsymbol{E})(\boldsymbol{\nabla} \times \boldsymbol{F}) - (\omega^2 \epsilon - \omega \sigma) \boldsymbol{E}\boldsymbol{F} \right) \mathrm{d}\boldsymbol{x} - \mathrm{i}\omega \int_{\Gamma_\mathrm{C}} \beta (\boldsymbol{n} \times \boldsymbol{E}) \boldsymbol{F} \, \mathrm{d}s \\ \quad = -\mathrm{i}\omega \int_\Omega \boldsymbol{J}^{\mathrm{imp}} \boldsymbol{F} \, \mathrm{d}\boldsymbol{x} + \mathrm{i}\omega \int_\Omega \boldsymbol{J}_S^{\mathrm{imp}} \boldsymbol{F} \, \mathrm{d}s, \quad \forall \boldsymbol{F} \in \boldsymbol{W}. \end{cases}$$

Here again, Ω is a bounded domain in $\mathbb{R}^2$, [1] with the Dirichlet, Neumann and Cauchy parts of the boundary, material data include permeability μ, permittivity ϵ, conductivity σ, and impedance constant β, and the load data consist of volume impressed current $\boldsymbol{J}^{\mathrm{imp}}$, and surface impressed current $\boldsymbol{J}_S^{\mathrm{imp}}$. Function $\tilde{\boldsymbol{E}}_D$ denotes a lift of Dirichlet data $\boldsymbol{E}_D$,

$$\tilde{\boldsymbol{E}}_D \in \boldsymbol{H}(\mathrm{curl}, \Omega), \quad \boldsymbol{n} \times \tilde{\boldsymbol{E}}_D = \boldsymbol{n} \times \boldsymbol{E}_D,$$

and the space of test functions is defined as,

$$\boldsymbol{W} = \{\boldsymbol{F} \in \boldsymbol{H}(\mathrm{curl}, \Omega) : \boldsymbol{n} \times \boldsymbol{F} = 0 \text{ on } \Gamma_\mathrm{D}\}.$$

Finally, ω is the angular frequency, and $\boldsymbol{n}$ denotes the outward normal unit vector on the domain boundary.

We shall also use the *stabilized formulation*.

$$\begin{cases} \boldsymbol{E} \in \tilde{\boldsymbol{E}}_D + \boldsymbol{W}, \quad p \in V, \\ \int_\Omega \left(\frac{1}{\mu} (\boldsymbol{\nabla} \times \boldsymbol{E})(\boldsymbol{\nabla} \times \boldsymbol{F}) - (\omega^2 \epsilon - \omega \sigma) \boldsymbol{E}\boldsymbol{F} \right) \mathrm{d}\boldsymbol{x} - \mathrm{i}\omega \int_{\Gamma_\mathrm{C}} \beta (\boldsymbol{n} \times \boldsymbol{E}) \boldsymbol{F} \, \mathrm{d}s, \\ - \int_\Omega (\omega \epsilon - \sigma) \boldsymbol{\nabla} p \boldsymbol{F} \, \mathrm{d}\boldsymbol{x} - \mathrm{i} \int_{\Gamma_\mathrm{C}} \beta (\boldsymbol{n} \times \boldsymbol{\nabla} p) \boldsymbol{F} \, \mathrm{d}s = -\mathrm{i}\omega \int_\Omega \boldsymbol{J}^{\mathrm{imp}} \boldsymbol{F} \, \mathrm{d}\boldsymbol{x} + \mathrm{i}\omega \int_\Omega \boldsymbol{J}_S^{\mathrm{imp}} \boldsymbol{F} \, \mathrm{d}s, \quad \forall \boldsymbol{F} \in \boldsymbol{W}, \\ - \int_\Omega (\omega \epsilon - \sigma) \boldsymbol{E} \boldsymbol{\nabla} q \, \mathrm{d}\boldsymbol{x} - \mathrm{i} \int_{\Gamma_\mathrm{C}} \beta (\boldsymbol{n} \times \boldsymbol{E}) \boldsymbol{\nabla} q \, \mathrm{d}s = -\mathrm{i} \int_\Omega \boldsymbol{J}^{\mathrm{imp}} \boldsymbol{\nabla} q \, \mathrm{d}\boldsymbol{x} + \mathrm{i} \int_\Omega \boldsymbol{J}_S^{\mathrm{imp}} \boldsymbol{\nabla} q \, \mathrm{d}s, \quad \forall q \in V. \end{cases}$$

[1] Formulation in $\mathbb{R}^3$ is identical.

Here p is the Lagrange multiplier, identically equal zero, with the corresponding space of test functions V,

$$V = \{q \in H^1(\Omega) : q = 0 \text{ on } \Gamma_{\mathrm{D}}\}.$$

We assume that domain Ω is simply connected and Dirichlet boundary Γ_{D} is connected as well. The gradient operator is then a linear isomorphism from space V onto a subspace of $\boldsymbol{W}$ of curl-free vector fields. The stabilized formulation has improved stability properties as $\omega \to 0$. In the case of non-conductive materials, and divergence free impressed currents, it is possible to ''build into'' the Lagrange multiplier p and test functions q the remaining ω, in which case the stabilized formulation remains uniformly stable as $\omega \to 0$.

3.3. Projection-based interpolation for elliptic problems

Given an element K in the coarse mesh, the corresponding FE space $X_{hp}(K)$, and a function u defined on the element, we define now the (projection-based) interpolant $\Pi_{hp}u = w_{hp}$. If element K has not been h-refined, the FE space (on the master element) consists of polynomials, otherwise we speak of piecewise polynomials of specific degree corresponding to the order of approximation for the four element sons. We introduce the space of element test functions,

$$V(K) = \{v \in X_{hp}(K) : v = 0 \text{ on } \partial K\},$$

and for each edge e, the space of edge test functions,

$$V(e) = \{v|_e : v \in X_{hp}(K), \quad v = 0 \text{ on } \partial K - e\}.$$

The edge test functions vanish at the edge endpoints and ''live'' in the finite element space. The element projection-based interpolant is now defined by requesting the following three conditions,

$$\begin{cases} w_{hp}(\boldsymbol{a}) = u(\boldsymbol{a}), & \forall \text{ vertex } \boldsymbol{a}, \\ \displaystyle\int_e \frac{\mathrm{d}}{\mathrm{d}s}(w_{hp} - u)\frac{\mathrm{d}v}{\mathrm{d}s}\frac{\mathrm{d}s}{\mathrm{d}\xi}\mathrm{d}s = 0, \quad \forall v \in V(e), & \forall \text{ edge } e, \\ \displaystyle\int_K \boldsymbol{\nabla}(w_{hp} - u)\boldsymbol{\nabla}v\,\mathrm{d}\boldsymbol{x} = 0, \quad \forall v \in V(K). \end{cases}$$

Here $x(\xi)$ denotes the element map transforming the master element into the physical element. For affine elements, derivative $\mathrm{d}s/\mathrm{d}\xi$ present in the definition of the edge projection reduces simply to the length of the element edge.

The interpolation here is defined on the physical element. Switching to the integration over the master element, we obtain,

$$\begin{cases} \hat{w}_{hp}(\hat{\boldsymbol{a}}) = \hat{u}(\hat{\boldsymbol{a}}), & \forall \text{ vertex } \hat{\boldsymbol{a}}, \\ \displaystyle\int_{\hat{e}} \frac{\mathrm{d}}{\mathrm{d}\xi}(\hat{w}_{hp} - \hat{u})\frac{\mathrm{d}\hat{v}}{\mathrm{d}\xi}\mathrm{d}\xi = 0, \quad \forall \hat{v} \in V(\hat{e}), & \forall \text{ edge } \hat{e}, \\ \displaystyle\int_{\hat{K}} \bar{a}_{ij}\frac{\partial}{\partial \xi_i}(\hat{w}_{hp} - \hat{u})\frac{\partial \hat{v}}{\partial \xi_j}\mathrm{d}\boldsymbol{\xi} = 0, \quad \forall \hat{v} \in V(\hat{K}). \end{cases}$$

Metric $\bar{a}_{ij}$ is dictated by the transformation from the master to physical coordinates,

$$\bar{a}_{ij} = \frac{\partial \xi_i}{\partial x_k}\frac{\partial \xi_j}{\partial x_k}\mathrm{jac}(\boldsymbol{\xi}),$$

with $\mathrm{jac}(\boldsymbol{\xi}) = \det(\partial x_i/\partial \xi_j)$ denoting the jacobian.

Thus only in the case of a simple scaling $\boldsymbol{x} = \boldsymbol{a} + h\boldsymbol{\xi}$, the interpolation on the master element and the physical elements yield identical results. More precisely, we have the commutativity property typical for standard interpolation procedures,

$$(\Pi u)^{\wedge} = \hat{\Pi}\hat{u}.$$

In the case of quad elements and anisotropic scaling,

$$x_1 = a_1 + h_1\xi_1, \quad x_2 = a_2 + h_2\xi_2,$$

the last step in the interpolation procedure reduces to,

$$\int_{\hat{K}} \left(\frac{h_2}{h_1}\right)\frac{\partial}{\partial\xi_1}(\hat{w}_{hp} - \hat{u})\frac{\partial\hat{v}}{\partial\xi_1} + \left(\frac{h_1}{h_2}\right)\frac{\partial}{\partial\xi_2}(\hat{w}_{hp} - \hat{u})\frac{\partial\hat{v}}{\partial\xi_2}\,\mathrm{d}\xi_1\,\mathrm{d}\xi_2 = 0, \quad \forall\hat{v}\in V(\hat{K}). \tag{3.1}$$

3.4. Projection-based interpolation for Maxwell problems

Consider the same scenario as before. Let $\boldsymbol{E}$ be a (vector-valued) function defined on element K, $\boldsymbol{Y}_{hp}(K)$ the element space consisting of either polynomials or piecewise polynomials, the corresponding space of element test functions,

$$\boldsymbol{W}(K) = \{\boldsymbol{F}\in\boldsymbol{Y}_{hp}(K) : \boldsymbol{n}\times\boldsymbol{F} = 0 \text{ on } \partial K\},$$

and the space of edge test functions,

$$Y_{hp}(e) = \{(\boldsymbol{n}\times\boldsymbol{E})|_e : \boldsymbol{E}\in\boldsymbol{Y}_{hp}(K)\}.$$

Replacing the H_0^1-projection over element K with $\boldsymbol{H}$(curl)-projection, and the weighted H_0^1-projection over edge e with weighted $L^2(e)$ projection, we obtain the interpolation procedure for $\boldsymbol{H}$(curl) spaces.

$$\begin{cases} \displaystyle\int_e \boldsymbol{n}\times(\boldsymbol{E}_{hp} - \boldsymbol{E})F\frac{\mathrm{d}s}{\mathrm{d}\xi}\,\mathrm{d}s = 0, \quad \forall F\in W(e), \quad \forall \text{ edge } e,\\ \displaystyle\int_K \{\boldsymbol{\nabla}\times(\boldsymbol{E}_{hp} - \boldsymbol{E})\boldsymbol{\nabla}\times\boldsymbol{F} + (\boldsymbol{E}_{hp} - \boldsymbol{E})\boldsymbol{F}\}\mathrm{d}\boldsymbol{x} = 0, \quad \forall\boldsymbol{F}\in\boldsymbol{W}(K). \end{cases}$$

In the case of the anisotropically scaled quad element we get,

$$\begin{cases} \displaystyle\int_{\hat{e}} \boldsymbol{n}\times(\hat{\boldsymbol{E}}_{hp} - \hat{\boldsymbol{E}})\hat{F}\,\mathrm{d}\xi = 0, \quad \forall\hat{F}\in W(\hat{e}), \quad \forall \text{ edge } \hat{e},\\ \displaystyle\int_{\hat{K}} \{\boldsymbol{\nabla}\times(\hat{\boldsymbol{E}}_{hp} - \hat{\boldsymbol{E}})\boldsymbol{\nabla}\times\boldsymbol{F} + h_2^2(\hat{E}_{1,hp} - \hat{E}_1)\hat{F}_1 + h_1^2(\hat{E}_{2,hp} - \hat{E}_2)\hat{F}_2\}\mathrm{d}\xi_1\,\mathrm{d}\xi_2 = 0, \quad \forall\hat{\boldsymbol{F}}\in\boldsymbol{W}(\hat{K}). \end{cases} \tag{3.2}$$

For small elements ($h_1, h_2 \to 0$) the problem degenerates and looses its stability in the same way as the standard variational formulation for Maxwell equations looses its stability for small frequency ω. A remedy to the problem is again to introduce the Lagrange multiplier and stabilized variational formulation,

$$\begin{cases} \displaystyle\int_{\hat{e}} \boldsymbol{n}\times(\hat{\boldsymbol{E}}_{hp} - \hat{\boldsymbol{E}})\hat{F}\,\mathrm{d}\xi = 0, \quad \forall\hat{F}\in W(\hat{e}), \quad \forall \text{ edge } \hat{e},\\ \displaystyle\int_{\hat{K}} \{\boldsymbol{\nabla}\times(\hat{\boldsymbol{E}}_{hp} - \hat{\boldsymbol{E}})\boldsymbol{\nabla}\times\boldsymbol{F} + h_2^2(\hat{E}_{1,hp} - \hat{E}_1)\hat{F}_1 + h_1^2(\hat{E}_{2,hp} - \hat{E}_2)\hat{F}_2\}\mathrm{d}\xi_1\,\mathrm{d}\xi_2\\ \displaystyle\quad + \int_{\hat{K}} \left\{\left(\frac{h_2}{h_1}\right)\frac{\partial\hat{p}}{\partial\xi_1}F_1 + \left(\frac{h_1}{h_2}\right)\frac{\partial\hat{p}}{\partial\xi_2}F_2\right\}\mathrm{d}\xi_1\,\mathrm{d}\xi_2 = 0, \quad \forall\hat{\boldsymbol{F}}\in\boldsymbol{W}(\hat{K}),\\ \displaystyle\int_{\hat{K}} \left\{\left(\frac{h_2}{h_1}\right)(\hat{E}_{1,hp} - \hat{E}_1)\frac{\partial\hat{v}}{\partial\xi_1} + \left(\frac{h_1}{h_2}\right)(\hat{E}_{2,hp} - \hat{E}_2)\frac{\partial\hat{v}}{\partial\xi_2}\right\}\mathrm{d}\xi_1\,\mathrm{d}\xi_2 = 0, \quad \forall\hat{v}\in V(\hat{K}). \end{cases} \tag{3.3}$$

Finally, dropping the L^2-term in the element projection altogether, we recover our definition of the projection-based interpolation [4],

$$\begin{cases} \int_{\hat{e}} \boldsymbol{n} \times (\hat{\boldsymbol{E}}_{hp} - \hat{\boldsymbol{E}})\hat{F}\,\mathrm{d}\xi = 0, \quad \forall \hat{F} \in W(\hat{e}), \quad \forall \text{ edge } \hat{e}, \\ \int_{\hat{K}} \left\{ \boldsymbol{\nabla} \times (\hat{\boldsymbol{E}}_{hp} - \hat{\boldsymbol{E}})\boldsymbol{\nabla} \times \boldsymbol{F} + \left(\frac{h_2}{h_1}\right)\frac{\partial \hat{p}}{\partial \xi_1}F_1 + \left(\frac{h_1}{h_2}\right)\frac{\partial \hat{p}}{\partial \xi_2}F_2 \right\} \mathrm{d}\xi_1\,\mathrm{d}\xi_2 = 0, \quad \forall \hat{\boldsymbol{F}} \in \boldsymbol{W}(\hat{K}), \\ \int_{\hat{K}} \left\{ \left(\frac{h_2}{h_1}\right)(\hat{E}_{1,hp} - \hat{E}_1)\frac{\partial \hat{v}}{\partial \xi_1} + \left(\frac{h_1}{h_2}\right)(\hat{E}_{2,hp} - \hat{E}_2)\frac{\partial \hat{v}}{\partial \xi_2} \right\} \mathrm{d}\xi_1\,\mathrm{d}\xi_2 = 0, \quad \forall \hat{v} \in V(\hat{K}). \end{cases} \tag{3.4}$$

The three discussed procedures differ only in the second step, the interpolation along the edges in all three cases is identical.

3.5. The 2D hp algorithm

We are ready now to discuss the 2D version of our mesh optimization algorithm. The logic of the algorithm is identical for both elliptic and Maxwell problems, only the projections being used are different. Given a current coarse mesh, we perform the global *hp*-refinement, and obtain the fine grid solution $u_{h/2,p+1}$. The mesh optimization algorithm consists now of the following steps.

The first two steps are *identical* with the 1D algorithm.

Step 1: For each edge in the coarse mesh, choose between p and h refinement and determine the guaranteed edge error decrease rate.

Step 2: Determine which edges to refine.

After the first two steps, we know which edges to refine and which refinement to choose: h or p.

Step 3: *Perform the requested h-refinements enforcing the* 1*-irregularity of the mesh.*

We loop through elements of the coarse grid. If at least one edge of the element is to be broken, we refine the element accordingly. As in [5,7], we compute the element isotropy flags and enforce the isotropic *h*-refinement, if the error function within the element changes comparably in both element directions.

Step 4: *Determine the optimal orders for the refined edges.* This step is similar but not identical with Step 3 of the 1D algorithm. The difference lies in the presence of *involuntary edge h-refinements*, due to the enforcement of the 1-irregularity of the mesh (see [6] for the definition of 1-irregular hybrid meshes consisting possibly of both quads and triangles). We begin the optimization process again from the best competitive *h*-refinement. For an involuntarily refined edge, the corresponding interpolation error is bigger than for the *p*-refined edge and, since the meshes are not nested, it *may be bigger* than the interpolation error for the coarse mesh. In other words the coarse grid cannot anymore serve as a reference grid for computing the error decrease rate and it has to be replaced with some other grid. Following the *biggest element* (*edge*) *error refinement path*, we identify the reference grid as the first one that reproduces the coarse grid interpolation error,

$$\|u_{h/2,p+1} - u_{hp}^{\text{ref}}\| \leqslant \|u_{h/2,p+1} - u_{hp}\|.$$

Once the reference grid has been determined (for edges selected to be *h*-refined in Step 1, the reference grid coincides simply with the coarse grid), we compute the corresponding decrease rate of the interpolation error,

$$\frac{\|u_{h/2,p+1} - w_{hp}\| - \|u_{h/2,p+1} - w_{hp}^{\text{ref}}\|}{N_{hp} - N_{hp}^{\text{ref}}}.$$

Here N_{hp}, N_{hp}^{ref} denote the number of d.o.f. for the mesh being considered and the reference mesh, respectively. As in the 1D algorithm, we stop increasing the orders p_1, p_2 for the edge sons, once the rate drops below 1/3 of the maximum rate.

We do not stage any competition between p and h refinements at the element level. After the third step, the topology of the new coarse mesh has been determined, and it remains only to determine the optimal distribution of orders for the element middle nodes using the same logic as for the edges. The minimum rule implies a starting point for the minimization procedure.

Step 5: *Determine the maximum element interpolation error decrease rate.* We use the same logic as in the 1D case. Starting with the orders implied by the minimum rule, we follow the *biggest element error refinement path* and compute the corresponding decrease rate for the element interpolation errors. We use the same concept of the *reference grid* as in the previous step for edges. The need for introducing the reference mesh is even more pronounced here. In the case of an h-refined element and the orders for the element sons implied by the minimum rule only, the corresponding interpolation error is frequently bigger than the error for the coarse grid. Another difference with the edge step lies in the way we increase the orders of approximation for the element sons. First of all, the maximum order for a p-refined element is set to $p + 1$. For quad elements, the order in two element directions, "horizontal" order p_h and "vertical" order p_v, may be different, and the max order is then set for each of the directions individually to $p_h + 1$, $p_v + 1$. In the case of an h-refined triangle, the maximum order is set only to p. For h-refined quads, the maximum order is again decided upon taking into account the anisotropy of h-refinements. If an element has been refined across its first axis, the corresponding maximum "horizontal" order is set to p_h, otherwise to $p_h + 1$, with the same rule applied to the second direction. The quad elements orders are also increased accounting for the anisotropy of the interpolation error. We use the same logic as for computing the element isotropy flags [5]. If the interpolation error for a quad element son displays a one-dimensional behavior, we increase its order of approximation in one direction only. This procedure is essential in approximating boundary layers.

Step 6: *Determine the optimal distribution of order of approximation p for the new coarse mesh.* We follow the same "investment strategy" as in Step 4 with edges. Once the maximum element interpolation error decrease rate is known, we revisit all elements again. For each element in the old coarse mesh, starting with the order of approximation implied by the order for the element edges and the minimum rule, we follow again the *biggest element error refinement path*, monitoring the element interpolation error decrease rate, and exit when the rate drops below 1/3 of the maximum rate.

Step 6 completes the mesh optimization procedure. Similarly, as in 1D, the approximation error on the coarse grid is estimated by simply computing the norm of the difference between the coarse and the fine grid solutions. If the difference (relative to the fine grid solution norm) is smaller than a requested error tolerance, we stop.

4. Numerical experiments

We begin with a short discussion of "dirty" implementation details of the algorithm described in the previous sections.

4.1. Interpolating on the master element

Contrary to our original algorithm presented in [5], we have switched to the interpolation over the master element. This allows for precomputing the stiffness matrices corresponding to the edge and element projection problems. In an effort to account for the anisotropy of quad elements, we have studied also the formulations (3.1)–(3.4) involving the two different element lengths h_1 and h_2.

4.2. Ignoring the constraints

In our original implementation [5], the projection-based interpolation accounts for hanging nodes. This means that the interpolation on edges must be done first for edges with unconstrained vertices (the "big" edges), and only once the interpolants on the big edges are known, we can proceed to interpolate on "small" edges with one or two constrained vertex nodes. Besides the increased complexity of the algorithm, the procedure has one significant drawback. As reported in [7], the interpolation procedures for H^1- and $H(\mathrm{curl})$-conforming elements do not commute anymore. This lack of commutativity results in completely erroneous values of the element interpolation error for the edge elements and failure of the *hp* mesh optimization algorithm. A remedy to the problem, is to interpolate not over individual elements of the coarse mesh (some of them with hanging nodes), but over groups of elements resulting from only "partially broken" father elements. In other words, we switch with the interpolation from a father element to its sons *only if all nodes of the father element has been broken.* In practice this means that the projections present in the mesh optimization algorithm would have to be done not only over groups of four element sons but over groups of up to sixteen element "grandsons". This was heading for a coding nightmare, and we have decided simply to ignore the constraints in the interpolation process. Ignoring the constraints also significantly simplifies the bookkeeping, the edge interpolants do not have to be stored in a global data structure, they simply can be recomputed locally when needed. The assumption on recomputing the edge interpolants goes in pair with the assumption on interpolating over the master element. With the stiffness matrices precomputed, the most expensive part of the interpolation procedure—integration of element matrices, has been eliminated.

4.3. Edge norms

The interpolation theory [4] indicates that the optimal edge norms are provided by minimum energy extensions. For the H^1 case, this implies the following definition of the $H^{\frac{1}{2}}_{00}$-norm for an edge e,

$$\|u\|_{H^{\frac{1}{2}}_{00}(e)} = (|\tilde{u}|_{H^1(\Omega_e)})^{\frac{1}{2}} = \left(\int_{\Omega_e} |\nabla\tilde{u}|^2 \mathrm{d}\boldsymbol{x}\right)^{\frac{1}{2}}.$$

Here u is a function defined on the edge e, vanishing at its endpoints, Ω_e denotes the patch consisting of one or two elements adjacent to the edge, and $\tilde{u}$ is the harmonic extension of function u to the patch. In principle, we need to know the metric corresponding to the norm, only for polynomials, or piecewise polynomials corresponding to an h-refined edge. The polynomial extension result [1] implies that we can replace the harmonic extensions with *computable* polynomial harmonic extensions and simply precompute the necessary Gram matrix corresponding to the metric. Computation of such a metric, even when done on the master elements, would have to account for the presence of one or two neighbors and, first of all, for the anisotropy of the neighboring elements. We actually had done such an experiment for a single isotropic master element in our study of the original algorithm in context of the L-shape domain problem [5], and saw no difference between the results obtained with such an optimal edge norm and those obtained with scaled H^1_0 norms. The issue seems to be much more sensitive for anisotropically refined meshes. In the case of elements with large (small) aspect ratio h_1/h_2, it is not clear *which* element size h should be used as the scaling factor in the edge norms. Based on energy considerations, we have decided to study the effect of the factor by replacing the edge length h with a new factor $\sqrt{h_1h_2}$ where h_1, h_2 represent the dimensions of an adjacent quad element. In the case of two quad elements adjacent to an edge, we sum up the contributions corresponding to the two elements,

$$h_e = \sum_K \sqrt{h_{K,1}h_{K,2}}.$$

We shall report effects of this change in the following antenna problem experiments.

4.4. The fine grid solver

All results reported in this paper were obtained using a direct (frontal) solver on the fine grid and, therefore, bear no possible effects of using partially converged fine grid solutions on the performance of the *hp* algorithm.

Example 1 (an antenna problem). A single loop coil antenna wrapped around a metallic cylinder radiates into a conductive medium, see Fig. 4. The data for the problem are as follows:

$$\text{domain} = 2 \times 4\ \text{m},$$

$$\mu_r = 1,$$

$$\epsilon_r = 1,$$

$$f = 2\ \text{MHz},$$

$$\sigma_{\text{metal}} = 10^7\ \text{S/m},$$

$$\sigma_{\text{medium}} = 1\ \text{S/m}.$$

The problem is axisymmetric and the solution of Maxwell equations can be reduced to the solution of a single Helmholtz-like equation in terms of cylindrical coordinates r, z, to be solved for the transversal com-

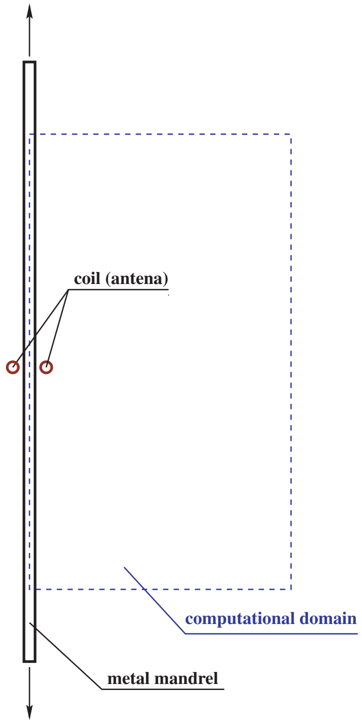

Fig. 4. Radiation from a coil antenna into a dispersive medium.

ponent $E = E_\theta$ of the electric field. Due to the exponential decay of the solution, we truncate the infinite domain to the computational rectangular domain indicated above, and apply Perfect Electric Conductor (PEC) boundary conditions. Notice that the domain includes a portion of the metallic mandrel. We model the cylinder with neither PEC nor approximate impedance boundary conditions, instead we attempt to resolve the boundary layer (skin effect) directly.

The variational formulation of the 2D problem looks as follows:

$$\begin{cases} E \in V, \\ \displaystyle\int_\Omega \left\{ \frac{1}{\mu}\left[\frac{1}{r^2}\left(E + r\frac{\partial E}{\partial r}\right)\left(F + r\frac{\partial F}{\partial r}\right) + \frac{\partial E}{\partial z}\frac{\partial F}{\partial z}\right] - (\omega^2\epsilon - \mathrm{i}\omega\sigma)EF\right\} r\,\mathrm{d}r\,\mathrm{d}z = \mathrm{i}\omega \int_{\Gamma_\mathrm{N}} r J_S^{\mathrm{imp}} F\,\mathrm{d}s, \quad \forall F \in V, \end{cases}$$

where

$$V = \{F \in H^1(\Omega) : F = 0 \text{ on } \Gamma_\mathrm{D}\}.$$

Here Γ_D is the outer boundary of the rectangular domain, and Γ_N corresponds to the boundary of the antenna, modelled with a square crosssection. The density of impressed surface current is assumed to be constant, with the value corresponding to a unit lumped current in the antenna.

Fig. 5 presents the convergence history for four versions of the algorithm. The first result was obtained with the original algorithm described in [5]. The element interpolation there was performed on the physical element. We shall not discuss here the second result which differs only slightly from the result produced with the original algorithm. The third and fourth versions correspond to the implementation described in this paper with the interpolation done on the master element. In the fourth case, we accounted for anisotropy of quad elements by using the scaled version of the element interpolation matrices, see (3.1). Both the old and the new results display exponential convergence, but the results obtained using the new algorithm, especially with anisotropic interpolation are significantly better. The main improvement seems to have come from the use of different scaling factors in the edge norms.

The results obtained with the original algorithm were produced beginning with a mesh of linear elements only. The increase in the reported error for meshes around 1000 d.o.f. illustrates a limitation of using the difference between the coarse and fine grid solution for an error estimate. The initial mesh of linear elements "does not see" the boundary layer. Mesh refinements originate around the antenna and propagate in all directions. Around a 1000 d.o.f. mesh, the refinements reach the metal cylinder and the fine mesh solution

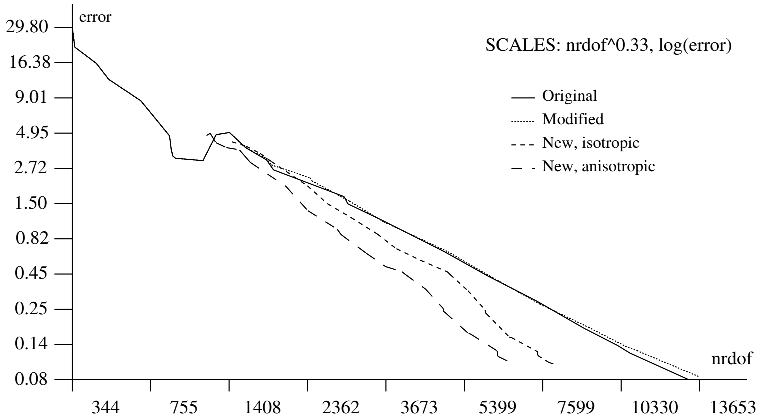

Fig. 5. Radiation from a coil antenna: convergence history, error estimate/norm vs number of d.o.f.

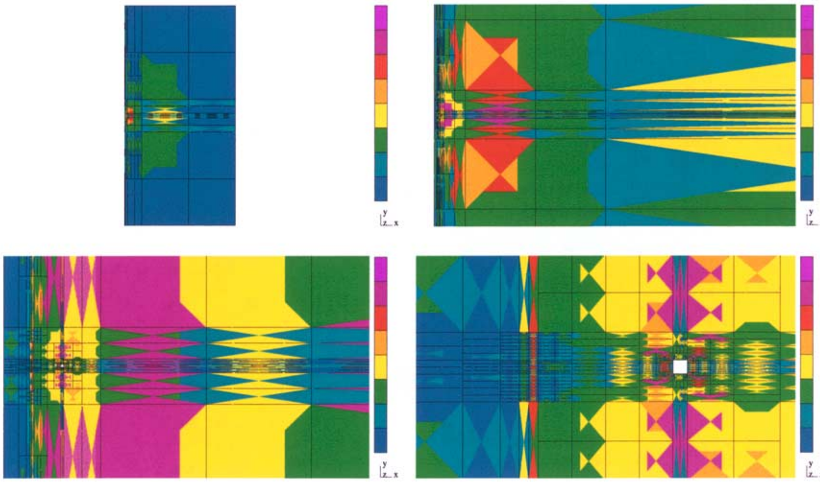

Fig. 6. Radiation from a coil antenna: the optimal *hp* mesh. Zooms with factors 1, 10, 100, 500.

drastically changes—the mesh begins to "see" the boundary layer. Consequently, the error estimate goes up. Once the boundary layer has been resolved by the fine mesh, the convergence curve stabilizes, and the exponential convergence can be seen. The remaining three curves were produced starting with the same coarse mesh and $p = 1$ except for the elements in the metallic cylinder where the *horizontal* order of approximation was increased to 4. The boundary layer is then seen from the very beginning and the reported convergence curve is monotone.

Fig. 6 presents details of our best mesh. Notice the full symmetry of the mesh with respect to the horizontal axis. The corresponding imaginary part of the solution representing the direct radiation from the antenna, and the real part, representing the secondary radiation resulting from the reflection from the mandrel, are shown in Figs. 7 and 8.

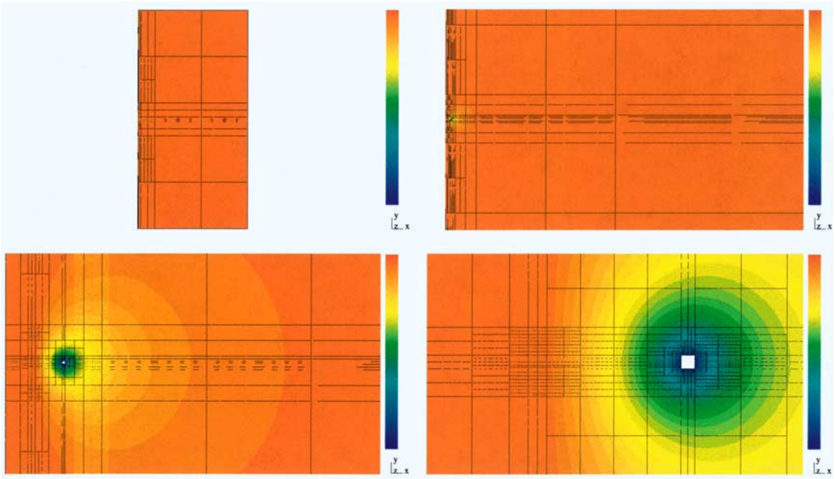

Fig. 7. Radiation from a coil antenna: zooming on the imaginary part of the solution.

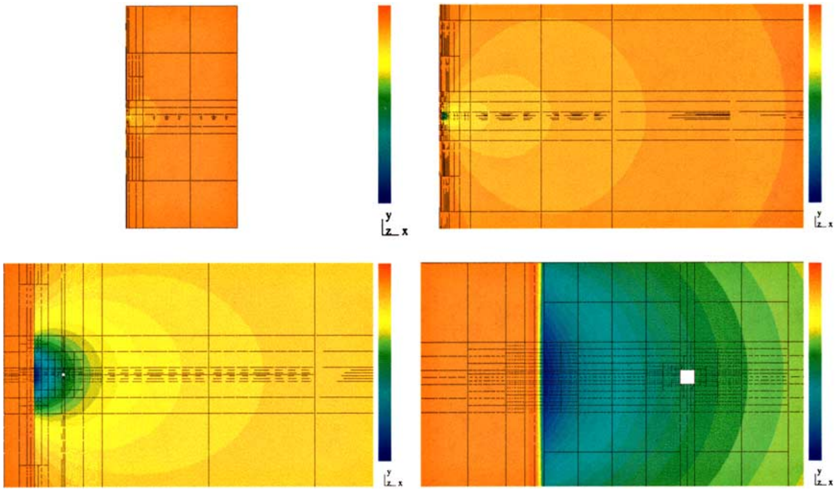

Fig. 8. Radiation from a coil antenna: zooming on the real part of the solution.

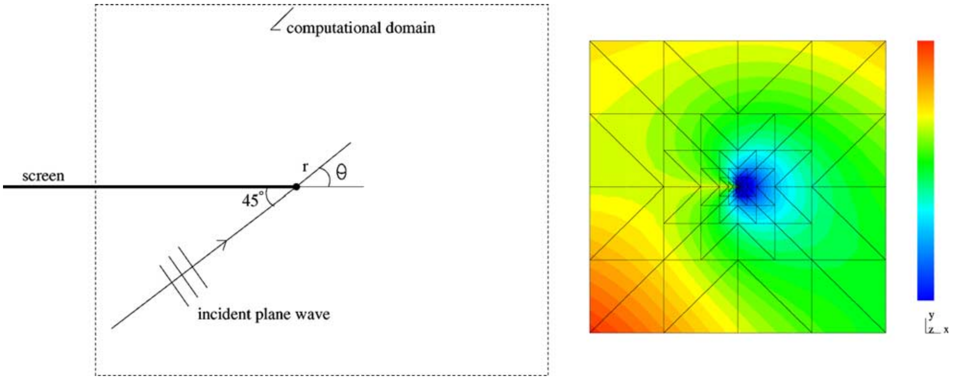

Fig. 9. Diffraction of a plane wave from an edge. Geometry and real part of the solution.

Example 2 (diffraction of a plane wave from an edge). We return to the diffraction problem reported in [9,6]. A plane wave shines at a 45° angle at a diffracting screen, see Fig. 9.

The exact solution is represented in terms of Fresnel integrals.

$$H(r,\theta) = \frac{1}{\sqrt{\pi}} \mathrm{e}^{\frac{\pi j}{4} - jkr} \left\{ F\left[(2kr)^{\frac{1}{2}} \sin\frac{1}{2}\left(\theta - \frac{\pi}{4}\right) \right] + F\left[(2kr)^{\frac{1}{2}} \sin\frac{1}{2}\left(\theta + \frac{\pi}{4}\right) \right] \right\},$$

$$F(u) = \frac{1}{2}\sqrt{\pi} \left\{ \mathrm{e}^{\frac{\pi j}{4}} - \sqrt{2}\left[C\left(\sqrt{\frac{2}{\pi}} u \right) - jS\left(\sqrt{\frac{2}{\pi}} u \right) \right] \right\},$$

$$C(z) - jS(z) := \int_0^s \mathrm{e}^{-\frac{1}{2}\pi j t^2}\, \mathrm{d}t \quad \text{(Fresnel integrals)}.$$

The solution represents the most singular scenario for 2D homogeneous problems. In order to avoid trouble with modeling the open boundary, we cut off a unit square computational domain around the diffracting

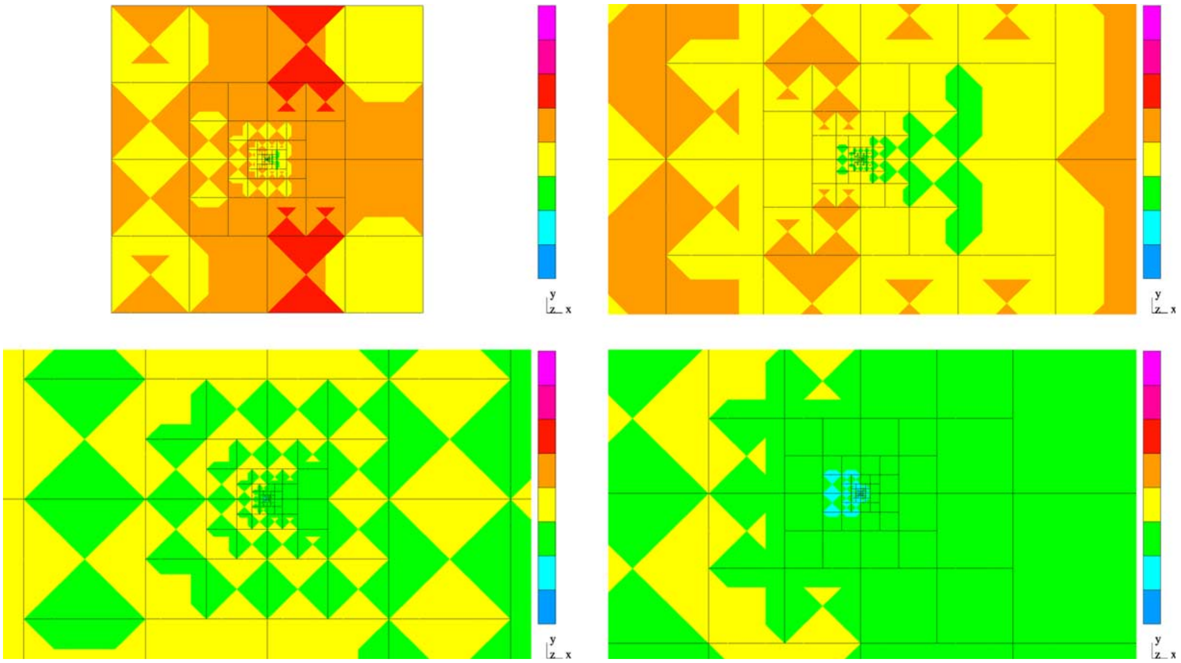

Fig. 10. Diffraction of a plane wave on a screen. Final *hp* mesh, zooming on the diffracting edge (with factor 10).

edge, and use the exact solution to impose Dirichlet boundary conditions. The material data are $\epsilon = \mu = \omega = 1$, $\sigma = 0$.

Starting with an initial mesh of 4×4 quadrilateral elements of third order, [2] we push the algorithm to deliver a rather academic error of 0.01%. Figs. 10 and 11 present eight consecutive zooms on the diffracting edge in the final, optimal mesh.

Figs. 12 and 13 present the corresponding zooms on the imaginary part of the first component of the electric field. Only at the level of the smallest elements, one can see the typical discontinuity artifacts related to the edge element discretization.

Finally, Fig. 14 presents the convergence history on an algebraic—log scale, for three different element projections discussed earlier:

- the H(curl)-projection (3.2),
- the H(curl)-stabilized projection (3.3),
- the actual projection defining the projection-based interpolation for H(curl)-conforming elements, (3.4).

In all three cases, the stabilized variational formulation has been used to solve the global problem. I have to admit that I have been a little bit disappointed to see practically no difference in the influence of the choice of the element projections on the performance of the *hp* algorithm. The first version of the algorithm "died" in a routine performing the element projections due to the (numerical) singularity of the matrix. Replacing the H(curl)-projection with the stabilized H(curl)-projection eliminated that problem but the code stopped only a couple of meshes later in the global solve of the fine grid problem due to a small pivot.

[2] We refer always to the order of H^1-conforming elements. The actual order of the edge elements here is "$2\frac{1}{2}$".

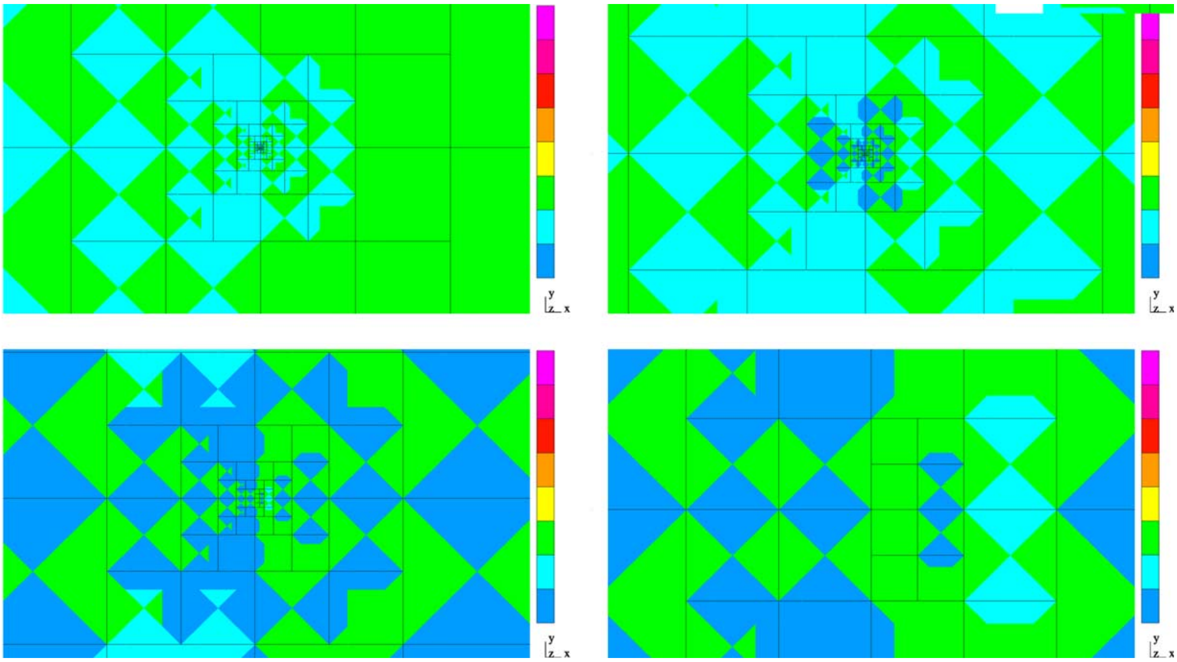

Fig. 11. Diffraction of a plane wave on a screen. Final *hp* mesh, zooming on the diffracting edge (cont.).

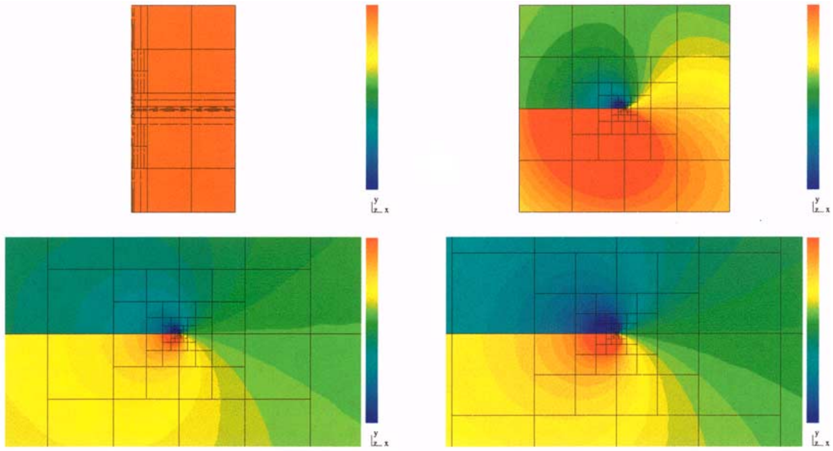

Fig. 12. Diffraction of a plane wave on a screen. Real part of E_1 component, zooming on the diffracting edge (with amplification factor 10).

Replacing the the stabilized H(curl)-projection with formulation (3.4) had effect on neither the solution of local problems nor on the overall performance of the algorithm.

For comparison, Fig. 14 presents also the convergence history for the *h*-refinement strategy with cubic elements (see [5] for the discussion of the strategy). Both *h* and *hp* algorithms display a strong deterioration of the convergence rates once the minimum element size drops below a certain level, the *hp* method seems to be more sensitive—the deterioration of the convergence can be observed earlier. We attribute it to the

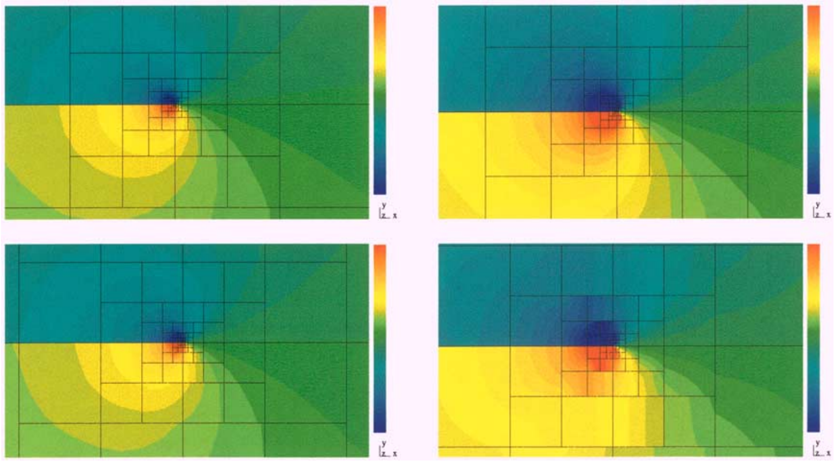

Fig. 13. Diffraction of a plane wave on a screen. Imaginary part of E_1 component, zooming on the diffracting edge (cont.).

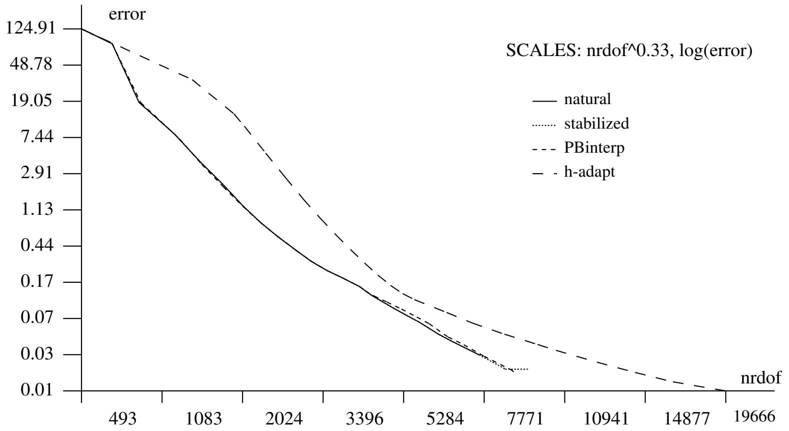

Fig. 14. Diffraction of a plane wave. Convergence history.

round-off error effects. Solutions to elliptic problems are singular only in terms of derivatives—the solution itself is finite. Singularity of the solution to the Maxwell problem is more severe—the solution itself blows up to infinity at the tip of the edge. We observe a need for more *h*-refinements which suggests that resolving the Maxwell singularities may call for more than a double precision.

5. Conclusions

We spell out one more time the main conclusions of the presented effort.

- An *identical* logic of the algorithm applies to both 1D problems, and the edge and element steps in the 2D algorithm, for both elliptic and Maxwell problems.

- Interpolation on the master element weighted with element size h_1, h_2 seems to be an optimal compromise between accuracy and efficiency.
- In presence of anisotropic refinements, the performance of the algorithm is sensitive with respect to the choice of norms used in the edge projections. The issue calls for a further, more systematic study.
- Local implementation of the projection-based interpolation ignoring the constraints, is essential for both reducing the complexity of the algorithm and the commutativity of the projection operators.
- The projection-based interpolation can be replaced with local Hcurl-projections.
- Double precision may be insufficient for capturing Maxwell singularities.

Numerical results reported here were obtained with a new version of our two-dimensional *hp*-adaptive package [6], and represent the fourth implementation for the elliptic and the second implementation for the Maxwell equations. The automatic *hp*-adaptivity package [3] represents a significant improvement over the previous versions but it still definitely leaves a lot of room for improvement. The different implementations reflect not only improved programming but, first of all, a growing understanding of the algorithm and the mathematics behind it. Our current work on a 3D version of the algorithm indicates still a prohibitive complexity and the need for further simplifications before we can speak about a success story in three dimensions.

Finally, let me comment on the issue of starting with an initial mesh that ‘‘sees’’ the boundary layer in the antenna problem. I believe that the moral of the story is that we cannot eliminate completely the human expertize from the FE modeling. The automatic adaptivity, including the *hp*-adaptivity can significantly aid the expertize but it cannot replace it.

Acknowledgments

The work has been supported by Air Force under Contract F49620-98-1-0255.

References

[1] I. Babuška, A. Craig, J. Mandel, J. Pitkaränta, Efficient Preconditioning for the *p*-Version Finite Element Method in Two Dimensions, SIAM J. Numer. Anal. 28 (3) (1991) 624–661.

[2] R. Beck, P. Deuflhard, R. Hiptmair, R.H.W. Hoppe, B. Wohlmuth, Adaptive multilevel methods for edge element discretizations of Maxwell's equations, Surv. Meth. Ind. 8 (1999) 271–312.

[3] L. Demkowicz, P. Monk, L. Vardapetyan, W. Rachowicz, De Rham diagram for hp finite element spaces, Math. Comput. Appl. 39 (7–8) (2000) 29–38.

[4] L. Demkowicz, I. Babuška, Optimal *p* interpolation error estimates for edge finite elements of variable order in 2D, SIAM J. Numer. Anal. 41 (4) (2003) 1195–1208.

[5] L. Demkowicz, W. Rachowicz, Ph. Devloo, A fully automatic *hp*-adaptivity, J. Sci. Comput. 17 (1–3) (2002) 127–155.

[6] L. Demkowicz, 2D *hp*-Adaptive Finite Element Package (2Dhp90). Version 2.1, TICAM Report 02-06.

[7] L. Demkowicz, *hp*-Adaptive finite elements for time-harmonic Maxwell equations, in: Topics in Computational Wave Propagation, in: M. Ainsworth, P. Davies, D. Duncan, P. Martin, B. Rynne (Eds.), Lecture Notes in Computational Science and Engineering, Springer Verlag, Berlin, 2003.

[8] D. Pardo, L. Demkowicz, Integration of *hp*-adaptivity and multigrid. I. A two grid solver for *hp* finite elements, TICAM Report 02-33.

[3] 18 k lines of code in Fortran 90.

[9] W. Rachowicz, L. Demkowicz, L. Vardapetyan, *hp*-Adaptive FE modeling for Maxwell's equations, Evidence of Exponential Convergence ACES' 99, Monterey, CA, 16–20 March 1999.

[10] A. Zdunek, W. Rachowicz, A three-dimensional *hp*-adaptive finite element approach to radar scattering problems, in: Fifth World Congress on Computational Mechanics Vienna, Austria, 7–12 July 2002.