Pavel Šolín; Karel Segeth; Ivo Doležel

Space-time adaptive hp-FEM: Methodology overview

In: Jan Chleboun and Petr Přikryl and Karel Segeth and Tomáš Vejchodský (eds.): Programs and Algorithms of Numerical Mathematics, Proceedings of Seminar. Dolní Maxov, June 1-6, 2008. Institute of Mathematics AS CR, Prague, 2008. pp. 185–200.

Persistent URL: http://dml.cz/dmlcz/702873

SPACE-TIME ADAPTIVE *hp*-FEM: METHODOLOGY OVERVIEW*

Pavel Šolín, Karel Segeth, Ivo Doležel

Abstract

We present a new class of self-adaptive higher-order finite element methods (*hp*-FEM) which are free of analytical error estimates and thus work equally well for virtually all PDE problems ranging from simple linear elliptic equations to complex time-dependent nonlinear multiphysics coupled problems. The methods do not contain any tuning parameters and work reliably with both low- and high-order finite elements. The methodology was used to solve various types of problems including thermoelasticity, microwave heating, flow of thermally conductive liquids etc. In this paper we use a combustion problem described by a system of two coupled nonlinear parabolic equations for illustration. The algorithms presented in this paper are available under the GPL license in the form of a modular C++ library HERMES[1].

1. Introduction

Partial differential equations (PDEs) describe many physical processes whose prediction and control are important to people. The most frequently used technique for the numerical solution of PDEs is the finite element method (FEM). The origins of this method are often associated with R. Courant [2] who solved numerically torsion problems in cylinders, drawing on a large body of earlier results for PDEs developed by Rayleigh, Ritz, and Galerkin. Since the 1940s, the method achieved a high degree of maturity and also the computational standards have changed. Nowadays, the significance of error control is greater than ever before, and the number of computations where adaptive mesh refinement algorithms are employed is rising very quickly.

On the other hand, self-adaptive finite element methods for PDEs have been studied by mathematicians for decades but so far, practitioners have been rather reluctant to use them. To understand why, note that every self-adaptive finite element method is guided by an *error estimator*. With a suitable error estimator in hand, the rest of automatic adaptivity (such as mesh refinement) is a purely technical matter. The current standard in computational PDEs are *analytical error estimators* – mathematical formulae "on paper". However, there is a very large number of such formulae, often they contain simplifying assumptions, are restricted to numerical

*This work was supported by grant 102/07/0496 of the Czech Science Foundation, grants IAA100190803 and IAA100760702 of the Grant Agency of the Academy of Sciences of the Czech Republic, Research Plan of the Academy of Sciences of the Czech Republic AV0Z10190503, and by Research Center 1M4674788502 of the Ministry of Education, Youth, and Sports.

[1]See the home page of the HERMES project http://spilka.math.unr.edu/hermes/.

methods of low order of accuracy, involve constants of unknown size, and/or include problem-dependent parameters that need to be tuned. In general, analytical error estimators are neither simple to use nor universal enough to cover a wide spectrum of problems of interest to practitioners. One cannot use them efficiently without a deep understanding of the underlying mathematics. From the point of view of a practitioner whose expertise is elsewhere and who would like to solve PDEs routinely, in order to obtain information that he or she needs for his research or application, the cost of dealing with burdens associated with self-adaptive methods often is not acceptable.

In this paper, we propose a way to circumvent this problem and make self-adaptive computational methods easily accessible to the broad computational community. The main idea of our approach is to use universal, computational error estimators that are motivated by modern embedded self-adaptive methods for ordinary differential equations (ODEs). These ODE methods are highly popular among engineers and practitioners due to their simplicity and universality: In every step, the algorithm computes two approximations with different orders of accuracy, and the error is estimated by their difference. Note that such error estimator is virtually independent of the underlying equation. The key requirement for practical applicability, however, is that the two approximations are computed efficiently. For example, in embedded Runge-Kutta RK2(3) methods, one evaluates two stages to obtain a second-order accurate approximation, and adds one more stage to obtain an approximation that is third-order accurate. Hence, one only pays the cost of a third-order method but also obtains a second-order error estimator.

The outline of the paper is as follows: In Sections 2 and 3 we present two techniques which are essential for the space-time adaptive algorithms: conforming higher-order approximation with arbitrary-level hanging nodes and the multimesh *hp*-FEM. In Section 4 we present a universal adaptivity algorithm for higher-order finite element methods. In Section 5 we extend this technique to time-dependent problems by combining the multi-mesh FEM with the classical Rothe's method. Example application to a flame propagation problem is shown in Section 7.

2. Approximation with arbitrary-level hanging nodes

The efficiency and algorithmic simplicity of our adaptive higher-order finite element algorithms is largely due to the technique of arbitrary-level hanging nodes [7]. When working with regular meshes (where two elements either share a common vertex, common edge, or their intersection is empty), adaptivity often is done using the *red-green refinement strategy* [1]. This technique first subdivides desired elements into geometrically convenient subelements with hanging nodes and then it eliminates the hanging nodes by forcing refinement of additional elements, as illustrated in Fig. 1.

This approach preserves the regularity of the mesh at the price of producing additional (forced) refinements and new degrees of freedom. Often, it creates elements

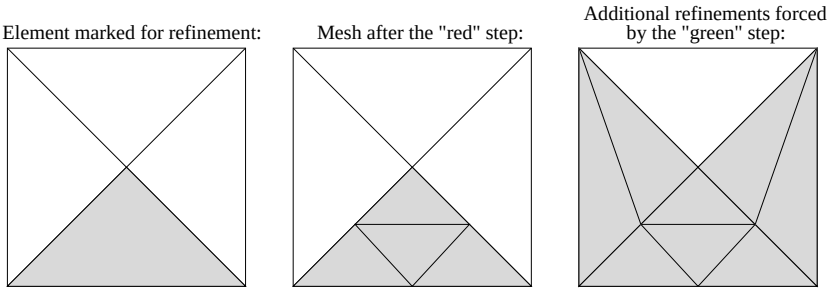

Fig. 1: *Red-green refinement.*

with sharp angles which, in general, are not desirable in finite element analysis.

The "green" refinements can be avoided by introducing *hanging nodes*, i.e., by allowing *irregular meshes* where element vertices can lie in the interior of edges of other elements. To ease the computer implementation, most finite element codes working with hanging nodes limit the maximum difference of refinement levels of adjacent elements to one (1-*irregularity rule*) – see, e.g., [3, 9]. In the following, by k-*irregularity rule* (or k-*level hanging nodes*) we mean this type of restriction where the maximum difference of refinement levels of adjacent elements is k. In this context, $k = 0$ corresponds to adaptivity with regular meshes and $k = \infty$ to adaptivity with arbitrary-level hanging nodes. It is illustrated in Fig. 2 that even the 1-irregularity rule does not avoid all forced refinements:

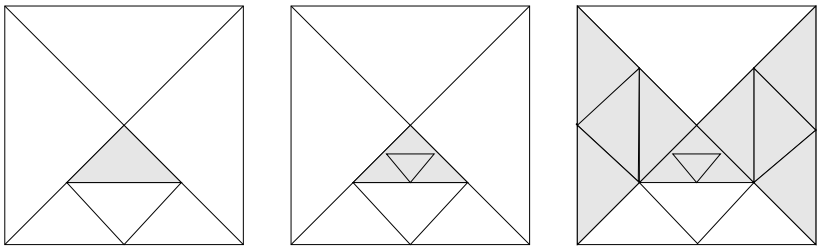

Fig. 2: *Refinement with 1-irregularity rule.*

The amount of forced refinements in the mesh depends on the level of hanging nodes allowed. Let us introduce a simple model problem which shows how the level of hanging nodes influences the number of degrees of freedom and condition number of the stiffness matrices: Consider a square domain $\Omega = (-1, 1)^2$ covered with a mesh consisting of four cubic elements, as shown in Fig. 3.

We solve the Poisson equation $-\Delta u = f$ in Ω with $u = 0$ on the boundary. Assume a right-hand side f such that the corresponding exact solution u is zero everywhere in Ω with the exception of a significant local perturbation contained inside of a small triangle T_n with the vertices $[-2^{-n}, -2^{-n}]$, $[0, 0]$, $[-2^{-n}, 2^{-n}]$. Fig. 4 shows, for $n = 5$, meshes obtained under various irregularity rules:

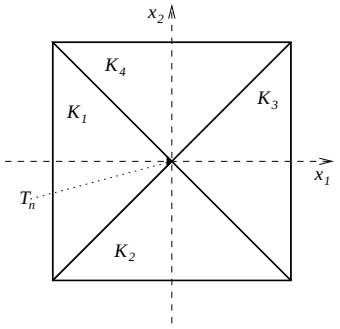

Fig. 3: *Domain Ω and initial coarse mesh.*

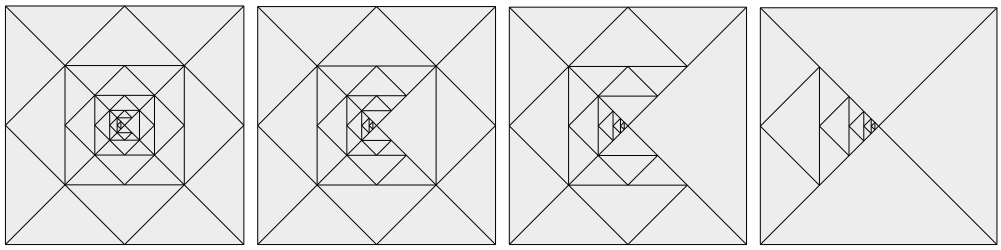

Fig. 4: *Meshes obtained with k-irregularity rules, $k = 1, 2, 3, \infty$.*

Notice that the amount of forced refinements within the elements K_2, K_3, and K_4 decreases as the parameter k grows. Next we run the adaptive procedure with cubic elements for $n = 1, 2, \ldots, 15$. Fig. 5 shows the number of degrees of freedom corresponding to the final meshes. The horizontal axis represents the spatial scale 2^{-n}. Fig. 6 shows the condition number of the corresponding stiffness matrices.

These results demonstrate that the performance of automatic adaptivity with arbitrary-level hanging nodes is superior to adaptivity on regular meshes, and even to adaptivity with one-, two-, or three-irregular meshes. Obviously, quantitative gains in the number of degrees of freedom and condition number of the stiffness matrix depend on specific features of the solved problem. In our experience, the advantages of the technique of arbitrary-level hanging nodes are most apparent in problems containing curvilinear material interfaces or boundary/internal layers.

3. Multi-mesh *hp*-FEM

The basic ingredient for the self-adaptive finite element methods presented in this paper is an algorithmic framework that allows us to work efficiently with various physical fields or various approximations of the same physical field defined on geometrically and polynomially different meshes. The higher-order multi-mesh FEM was first introduced in the context of linear thermoelasticity in [8], where the displacement components u_1, u_2 and the temperature T were approximated on different meshes equipped with independent adaptivity mechanisms.

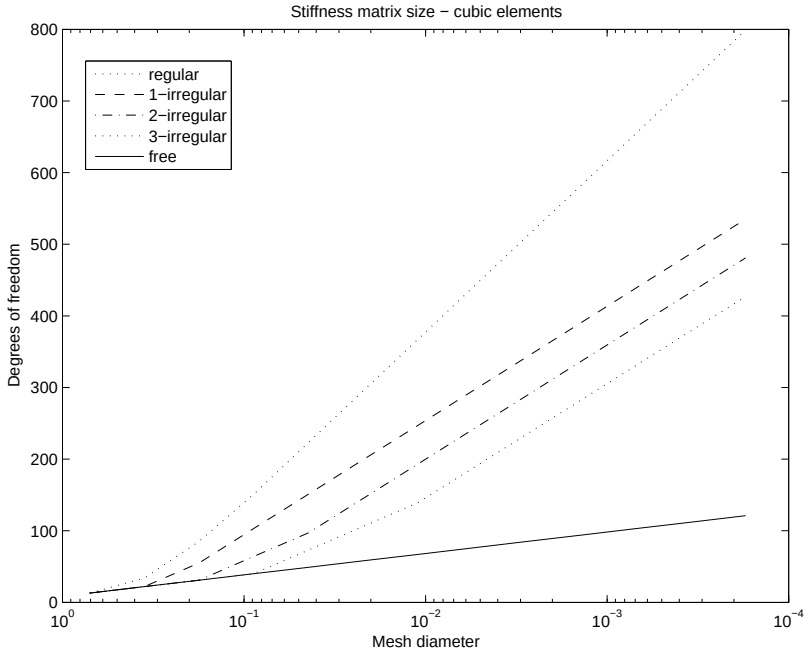

Fig. 5: *Relation between the size of the stiffness matrix and the level k of hanging nodes ($k = 0, 1, 2, 3, \infty$).*

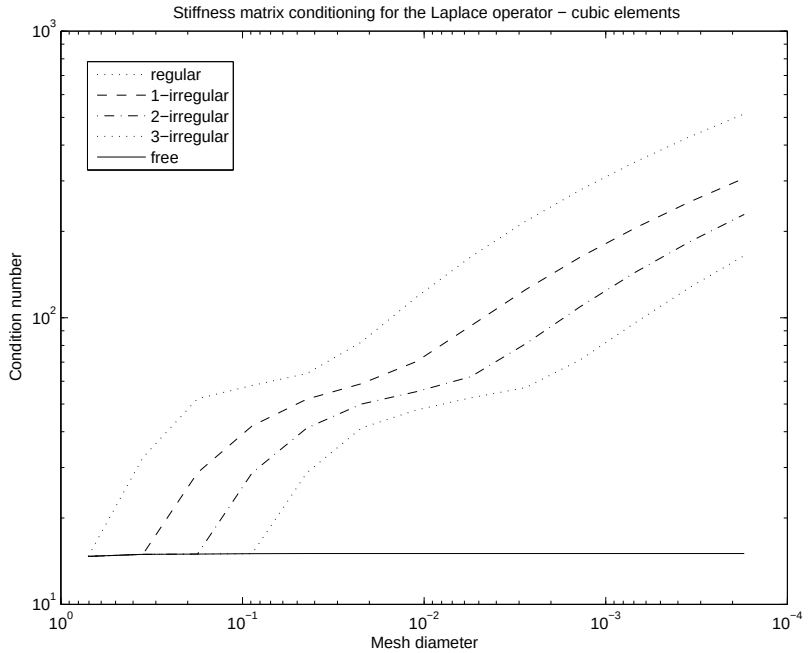

Fig. 6: *Relation between the condition number of the stiffness matrix and the level k of hanging nodes ($k = 0, 1, 2, 3, \infty$).*

The main ideas of the multi-mesh *hp*-FEM are as follows: For the sake of programming feasibility, we restrict ourselves to meshes derived from a common coarse *master mesh* τ_m via sequences of mutually independent local refinements. The master mesh τ_m is very coarse and often it is not even used for discretization purposes – it serves as the top of a tree-like data structure which is utilized by the multi-mesh assembling procedure. The situation is illustrated in parts A – D of Fig. 7. In part E of Fig. 7 we also show the geometrical union of all meshes in the system that we call *union mesh* and denote by τ_u.

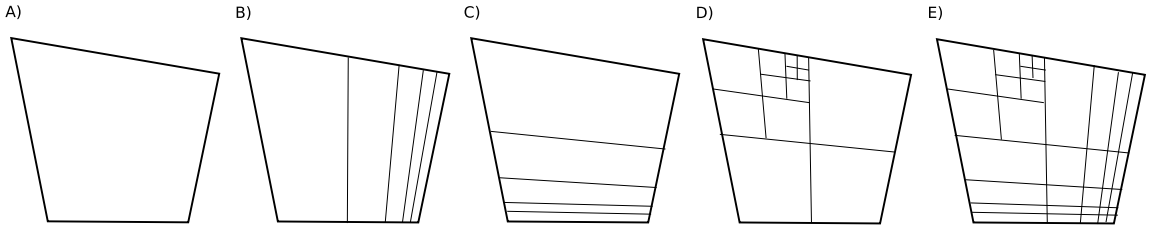

Fig. 7: *Example of a master mesh τ_m (left), meshes τ_1, τ_2, τ_3 obtained by its refinements, and the corresponding union mesh τ_u (right).*

It is worth mentioning that the union mesh is never created physically in the computer memory but its virtual elements are parsed by the element-by-element assembling procedure as usual in higher-order finite element methods [10], and that the multi-mesh FEM uses hanging nodes of arbitrary level (see [7], also available as a preprint on-line[2]). This technique eliminates forced refinements and thus contributes greatly to the modularity and efficiency of automatic adaptivity algorithms. The multi-mesh FEM would work (less efficiently) also with one-level hanging nodes, but not on regular meshes (due to possibly conflicting green refinements).

4. Adaptive *hp*-FEM with arbitrary-level hanging nodes

In contrast to standard adaptive FEM (*h*-FEM), automatic adaptivity in the *hp*-FEM requires more information about the behavior of the error in element interiors (see, e.g., [3, 4, 10] and the references therein). Some authors investigate numerically the analyticity of the solution in every element in order to decide between *p*- and *h*-refinement [4]. Such approach uses two refinement candidates per element, as illustrated in Fig. 8 (the numbers in elements stand for their polynomial degrees). According to our experience, at least for elliptic problems this strategy yields exponential convergence.

We prefer a different approach where more refinement candidates are considered, as shown in Fig. 9.

Typically, we vary the polynomial degrees in the subelements by two, which for a triangular element yields $3^4 = 81$ *h*-refinement candidates. The strategy was described in detail in [7]. Since in the latter case every refinement candidate can be

[2]http://www.math.utep.edu/preprints/2006/2006-07.pdf

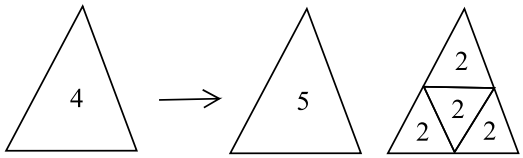

Fig. 8: *hp-adaptivity with two refinement candidates (p and h refinement).*

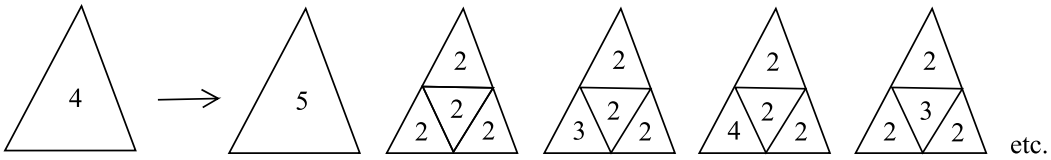

Fig. 9: *hp-adaptivity with multiple refinement candidates.*

reproduced using several steps with the pair of candidates of the former strategy, it is not surprising that usually the convergence curves are almost identical when error is plotted as a function of the number of degrees of freedom. However, according to our experience, computations with the latter approach usually take less CPU time since fewer adaptivity steps are needed and thus the discrete problem is solved less frequently. This is illustrated in Fig. 10.

Obviously, the latter strategy requires even more information about the error than the level of its analyticity. In order to select an optimal refinement candidate, we need to know the approximate *shape* of the error function $\epsilon_{h,p} = u - u_{h,p}$. In principle, this information could be recovered from suitable estimates of higher derivatives of the solution, but such approach is not very practical and it has not been used by anyone to our best knowledge. In practice, we employ the technique of *reference solutions* [3]. The reference solution u_{ref} is sought in an enriched finite element space V_{ref}, and the error function is approximated as $\epsilon_{h,p} \approx u_{ref} - u_{h,p}$. The reference space V_{ref} is constructed in such a way that all elements in the mesh are subdivided uniformly and their polynomial degree is increased, i.e., $V_{ref} = V_{h/2,p+1}$. The method for selecting the optimal refinement candidate will be described in the following.

4.1. Element-by-element adaptivity algorithm

With an a-posteriori error estimate of the form

$$\epsilon_{h,p} \approx u_{ref} - u_{h,p}, \tag{1}$$

the outline of our *hp*-adaptivity algorithm is as follows:

1. Assume an initial coarse mesh $\tau_{h,p}$ consisting of (usually) quadratic elements. Besides other technical data, user input includes a prescribed tolerance $TOL > 0$ for the H^1-norm of the approximate error function (1) and the number D_{DOF} of degrees of freedom to be added in every *hp*-adaptivity step.

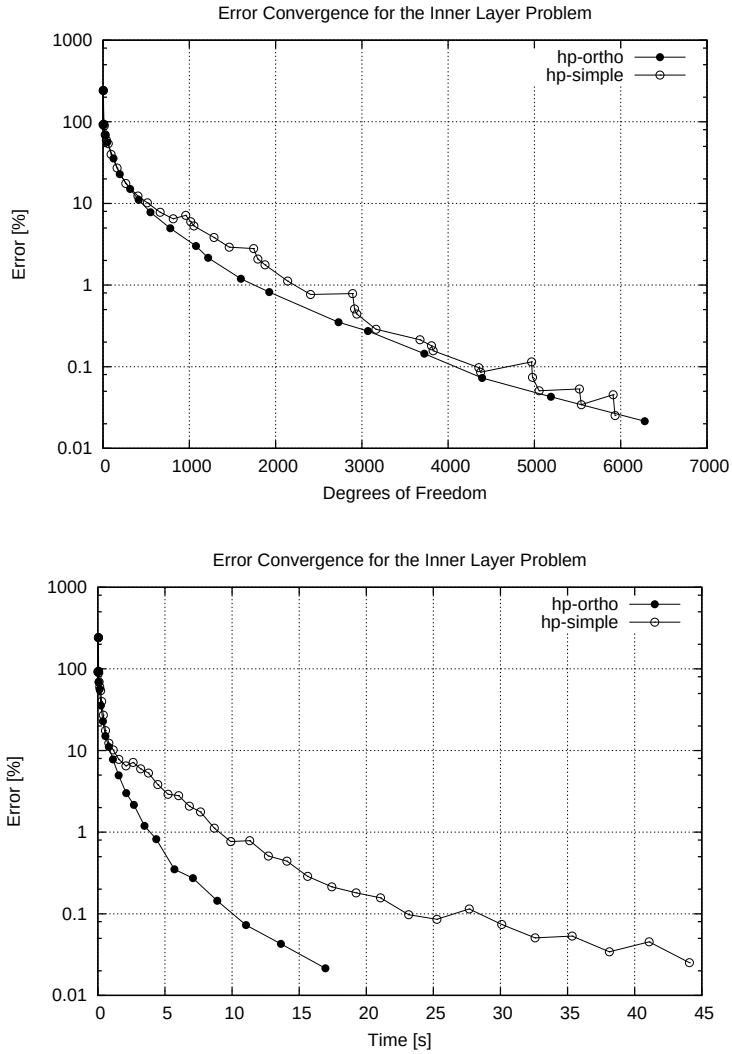

Fig. 10: *Illustration of performance of adaptivity schemes with two refinement candidates per element (simple) and multiple candidates per element (ortho). The horizontal axis shows the number of DOF (top) and CPU time (bottom).*

2. Compute coarse mesh approximation $u_{h,p} \in V_{h,p}$ on $\tau_{h,p}$.

3. Find reference solution $u_{ref} \in V_{ref}$, where V_{ref} is obtained by dividing all elements and increasing the polynomial degrees by one.

4. Construct the approximate error function (1), calculate its norm

$$ERR_i^2 = \|\epsilon_{h,p}\|_{1,2}^2$$

on every element K_i in the mesh, $i = 1, 2, \ldots, M$. Calculate the global error,

$$ERR^2 = \sum_{i=1}^{M} ERR_i^2.$$

5. If $ERR \leq TOL$, stop computation and proceed to postprocessing.

6. Sort all elements into a list L according to their ERR_i values in decreasing order.

7. While the number of newly added degrees of freedom in this step is less than D_{DOF} do:

   (a) Take the next element K from the list L.

   (b) Perform *hp*-refinement of K (to be described in more detail in Paragraph 4.2). Note that the refinement of K may introduce new hanging nodes on its edges, but the surrounding mesh elements are not affected.

8. Adjust polynomial degrees on unconstrained edges (edges without hanging nodes, cf. [10]) using the *minimum rule* (every unconstrained edge is assigned the minimum of the polynomial degrees on the pair of adjacent elements).

9. Continue with step 2.

4.2. Selection of optimal *hp*-refinement of an element

Let $K \in \tau_{h,p}$ be an element of polynomial degree p_K that was marked for refinement. Without loss of generality, assume that K is a triangle, the procedure for refinement of quadrilateral elements is analogous. We consider the following $N_{ref} = k + (k+1)^4$ refinement options, where $k \geq 0$ is a user input parameter:

1. Increase the polynomial degree of K by $1, 2, \ldots, k$ without spatial subdivision. This yields k refinement candidates.

2. Split K into four similar triangles K_1, K_2, K_3, K_4. Define p_0 to be the integer part of $p_K/2$. For each K_i, $1 \leq i \leq 4$ consider $k+1$ polynomial degrees $p_0 \leq p_i \leq p_0 + k$. This yields additional $(k+1)^4$ refinement candidates. In this case, edges lying on the boundary of K inherit the polynomial degree p_j of the adjacent interior element K_j. Polynomial degrees on interior edges are determined using the minimum rule.

For each of these N_{ref} options, we perform a standard H^1-projection of the reference solution u_{ref} onto the corresponding vector-valued piecewise-polynomial space on the refinement candidate. The candidate with minimum projection error relative to the number of added degrees of freedom is selected.

Note that if the technique of arbitrary-level hanging nodes is not in effect, *hp*-refinements involving spatial subdivision can be more costly than *p*-refinement candidates, since the latter never cause forced refinements. If the selection of the optimal element refinement is done locally, i.e., without taking the forced refinements into account, the *hp*-adaptive algorithm may make wrong decisions.

5. Space-time adaptive *hp*-FEM

Our space-time adaptive algorithm is based on a combination of the multi-mesh *hp*-FEM presented in Section 3 with *Rothe's method*. Rothe's method is a natural counterpart of the more widely used Method of Lines (MOL). While MOL preserves the continuity in time and discretizes the spatial variable, yielding a system of ODEs in time, Rothe's method preserves the continuity of the spatial variable while discretizing time, which leads to one or more PDEs in space per time step. The actual number of these PDEs is proportional to the order of accuracy of the time discretization method. For example, the implicit Euler method yields one PDE in space per time step. Note that Rothe's method is fully equivalent to the MOL if no adaptivity takes place. However, in contrast to MOL, Rothe's method provides an excellent opportunity to exploit spatially adaptive algorithms in the context of time-dependent problems.

Let us illustrate the space-time adaptive algorithms on a simple example of a parabolic heat transfer equation of the form

$$\frac{\partial u}{\partial t} - \Delta u = f. \tag{2}$$

Note that the methodology is not restricted to linear parabolic problems as will be shown in Section 7. When applying the backward Euler method to (2), we obtain

$$\frac{\partial u}{\partial t} \approx \frac{u^{n+1} - u^n}{\Delta t} \Rightarrow -\Delta t \Delta u^{n+1} + u^{n+1} = u^n + \Delta t f^{n+1}. \tag{3}$$

If we choose to use a second-order accurate backward differentiation formula instead, we obtain

$$\frac{\partial u}{\partial t} \approx \frac{3u^{n+2} - 4u^{n+1} + u^n}{2\Delta t} \Rightarrow -2\Delta t \Delta u^{n+2} + 3u^{n+2} = 4u^{n+1} - u^n + 2\Delta t f^{n+2}. \tag{4}$$

Note that equations (3) and (4) contain spatial derivatives only, and can therefore be solved using the spatially-adaptive algorithm which was described in Section 4.

Application of the multi-mesh hp-FEM

The approximation u^n on the right-hand side of (3) is defined on a locally refined mesh that was constructed using an adaptive process during the previous time step. The unknown approximation u^{n+1} corresponding to the end of the current time step is obtained using a new adaptive process that starts from some coarse mesh. Thus in every step of the adaptive algorithm, assembling is done over two different meshes. While the mesh for u^n remains the same during the adaptivity process, the mesh for u^{n+1} changes after each refinement step. Assembling over different meshes is done using the multi-mesh technology which was described in Section 3. At the end of the current time step, u^{n+1} is defined on a new locally refined mesh that is different

from the mesh for u^n – it is finer in some regions and coarser in others. This effect can be seen as simultaneous mesh refinement and coarsening between time steps. In every time step, the adaptivity process stops when a prescribed accuracy in space is reached. The stopping criterion is related to the norm of the difference between the reference and coarse mesh solutions, as described in Section 4. Adaptive selection of the time step is a simple matter and one can use standard ODE methods to do that. The process will be illustrated on a flame propagation problem in Section 7.

In practice, the initial mesh for u^{n+1} is either chosen to be the master mesh (coarsest mesh in the multi-mesh hierarchy, see Section 3), or we take off the last one or two refinement layers from the final mesh for u^n, and start from there. The former approach yields a sequence of meshes which is more optimal from the point of view of the number of DOF used but the computation is longer. On the other hand, the latter one is faster but the final mesh for u^{n+1} may contain some local refinements which are due to u^n and not needed for u^{n+1}. As a consequence, one may need more DOF to obtain u^{n+1} on the desired level of accuracy. In our opinion, the latter approach is preferable for practical computations where CPU time matters. The former one can be used for presentation purposes, such as producing videos of meshes evolving in time, etc. The situation is similar for the second-order backward differentiation formula (4), where we need to assemble simultaneously more than three different meshes.

6. Adaptive control of time step

A simple strategy for adaptive time step control was introduced by Valli et al. [11]. Their *PID controller* is based on the relative changes of a suitable indicator variable (temperature, concentration, turbulent kinetic energy, eddy viscosity etc.) and can be summarized as follows:

1. Compute the relative changes of the chosen indicator variable u

$$e_n = \frac{||u^{n+1} - u^n||}{||u^{n+1}||}.$$

2. If they are too large ($e_n > \delta$), reject u^{n+1} and recompute it using

$$\Delta t_* = \frac{\delta}{e_n} \Delta t_n.$$

3. Adjust the time step smoothly to

$$\Delta t_{n+1} = \left(\frac{e_{n-1}}{e_n}\right)^{k_P} \left(\frac{TOL}{e_n}\right)^{k_I} \left(\frac{e_{n-1}^2}{e_n e_{n-2}}\right)^{k_D} \Delta t_n.$$

4. Limit the growth and reduction of the time step so that

$$\Delta t_{\min} \le \Delta t_{n+1} \le \Delta t_{\max}, \qquad m \le \frac{\Delta t_{n+1}}{\Delta t_n} \le M.$$

The default values of the PID parameters as proposed by Valli et al. [11] are

$$k_P = 0.075, \quad k_I = 0.175, \quad k_D = 0.01.$$

Unlike in the case of adaptive time-stepping based on the local truncation error, there is no need to compute an extra solution with a different time step. Hence, the cost of the feedback mechanism is negligible. The method has been used with favorable results by various researchers including [6].

7. Example: A flame propagation problem

We consider a freely propagating laminar flame and its response to a heat-absorbing obstacle represented by a set of cooled parallel rods with a rectangular cross-section. The domain Ω is shown in Fig. 11.

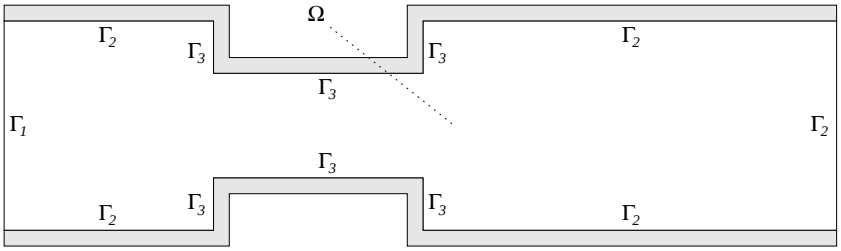

Fig. 11: *Computational domain for the flame propagation problem.*

The domain has dimensionless length $l = 60$ and width $w = 16$. The narrow part has width $w/2$, length $l/4$, and starts at $l/4$ from the left end point.

We use a low Mach number laminar flame propagation model taken from [5] which assumes that the motion of the fluid is independent from the temperature and species concentration. The flow velocity in the burner is considered to be zero. The model consists of a system of two coupled nonlinear parabolic equations

$$\begin{aligned} \frac{\partial \theta}{\partial t} - \Delta\theta &= \omega(\theta, Y) \quad \text{in } \Omega \times (0, T_0), \\ \frac{\partial Y}{\partial t} - \frac{1}{\mathrm{Le}}\Delta Y &= -\omega(\theta, Y) \quad \text{in } \Omega \times (0, T_0) \end{aligned}$$

for the dimensionless temperature θ, $0 \le \theta \le 1$, and dimensionless concentration Y, $0 \le Y \le 1$. The dimensionless time $T_0 = 60$. The goal of the computation is accurate resolution of the nonstationary reaction rate (flame intensity) $\omega(\theta, Y)$ which is defined by the Arrhenius law

$$\omega(\theta, Y) = \frac{\beta^2}{2\mathrm{Le}} Y \exp \frac{\beta(\theta - 1)}{1 + \alpha(\theta - 1)}. \tag{5}$$

Here, Le = 1 is the Lewis number (ratio of diffusivity of heat and diffusivity of mass), $\alpha = 0.8$ the gas expansion coefficient in a flow with nonconstant density, and $\beta = 10$ the dimensionless activation energy.

On the left boundary edge Γ_1, Dirichlet boundary conditions corresponding to the burnt state are prescribed, i.e.,

$$\begin{aligned} \theta &= 1 \quad \text{in } \Gamma_1 \times (0, T_0), \\ Y &= 0 \quad \text{in } \Gamma_1 \times (0, T_0). \end{aligned}$$

The remaining part Γ_2 of the boundary is assumed adiabatic with the homogeneous Neumann conditions

$$\begin{aligned} \frac{\partial \theta}{\partial \nu} &= 0 \quad \text{in } \Gamma_2 \times (0, T_0), \\ \frac{\partial Y}{\partial \nu} &= 0 \quad \text{in } \Gamma_2 \times (0, T_0). \end{aligned}$$

The absorption of heat along Γ_3 is modeled via Newton boundary conditions with a heat loss parameter $k = 0.1$,

$$\begin{aligned} \frac{\partial \theta}{\partial \nu} &= -k\theta \quad \text{in } \Gamma_3 \times (0, T_0), \\ \frac{\partial Y}{\partial \nu} &= 0 \quad \text{in } \Gamma_3 \times (0, T_0). \end{aligned}$$

As the initial condition, we prescribe the analytical solution of a one-dimensional model [5]:

$$\theta(0, x_1) = \begin{cases} 1 & \text{for } x_1 < x^*, \\ \exp(x^* - x_1) & \text{for } x_1 \geq x^* \end{cases} \tag{6}$$

and

$$Y(0, x_1) = \begin{cases} 0 & \text{for } x_1 < x^*, \\ 1 - \exp(\text{Le}(x^* - x_1)) & \text{for } x_1 \geq x^* \end{cases} \tag{7}$$

with $x^* = 9$.

Figs. 12–14 show the reaction rate $\omega(Y, \theta)$ and the underlying *hp*-FEM meshes for three different time instants t_1, t_2, t_3. The numbers inside elements indicate their polynomial degrees. Notice that very small elements on the flame front are adjacent to very large elements. This is possible due to the technique of arbitrary-level hanging nodes [7]. For problems with sharp fronts or curvilinear material interfaces, this technique saves large amount of degrees of freedom which otherwise would be needed to keep the mesh regular. Movies showing the dynamical evolution of $\omega(Y, \theta)$ along with the corresponding *hp*-FEM meshes for this problem can be found at `http://spilka.math.unr.edu/gallery/`.

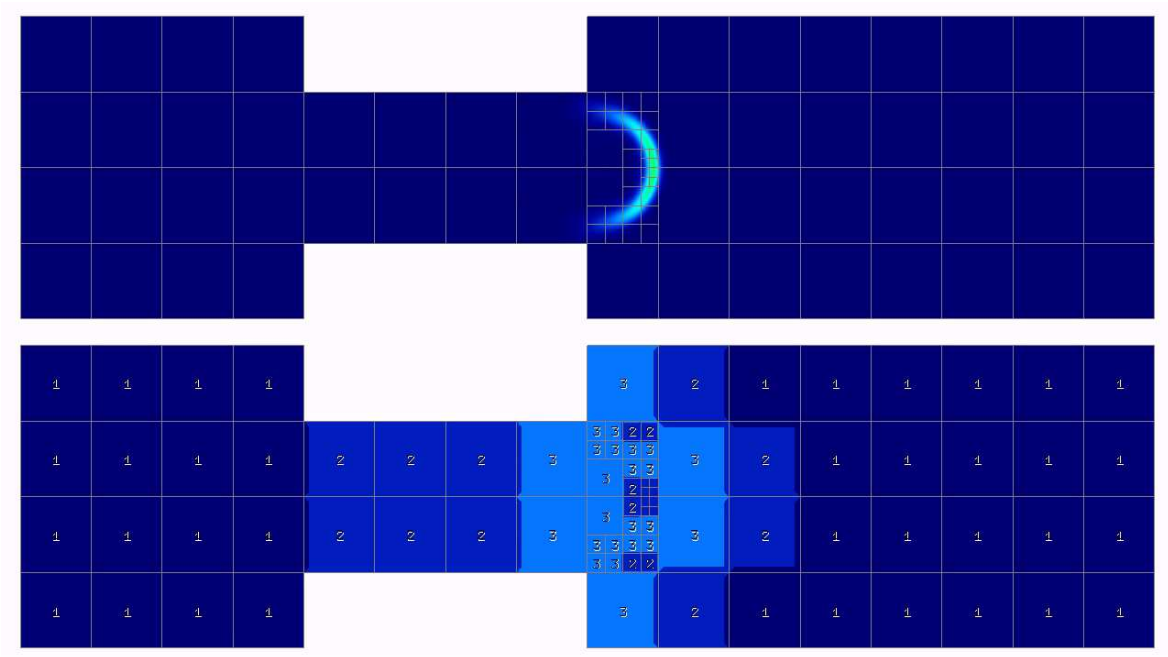

Fig. 12: *Reaction rate and higher-order finite element mesh at time* t_1.

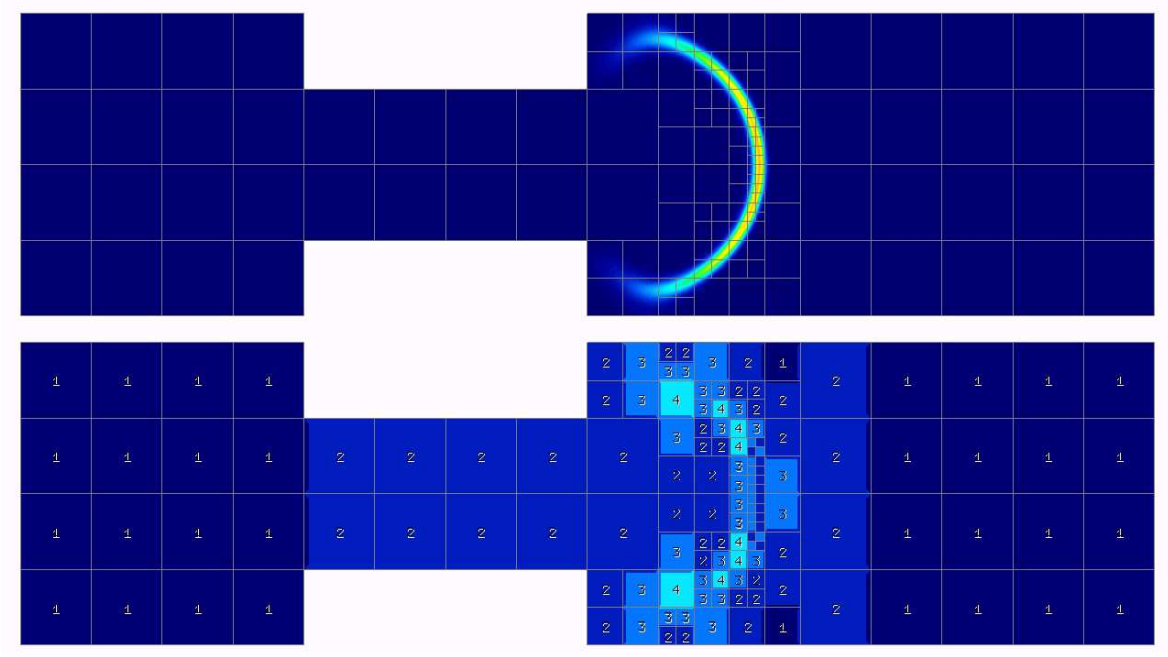

Fig. 13: *Reaction rate and higher-order finite element mesh at time* t_2.

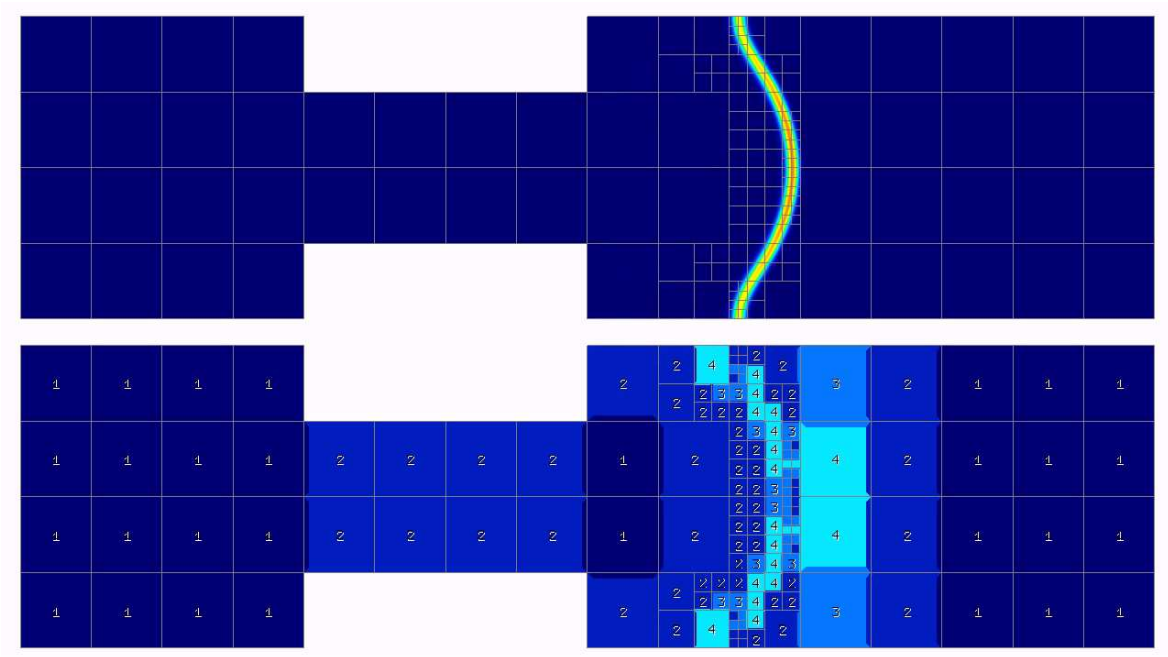

Fig. 14: *Reaction rate and higher-order finite element mesh at time* t_3.

References

[1] B. Aksoylu, S. Bond, M. Holst: *An odyssey into local refinement and multilevel preconditioning III: Implementation and numerical experiments*. SIAM J. Sci. Comput. **25** (2003), 478–498.

[2] R. Courant: *Variational methods for the solution of problems of equilibrium and vibrations*. Bull. Amer. Math. Soc. **49** (1943), 1–23.

[3] M. Paszynski, J. Kurtz, L. Demkowicz: *Parallel, fully automatic hp-adaptive 2D finite element package*. TICAM Report 04-07. The University of Texas at Austin 2004.

[4] T. Eibner, J.M. Melenk: *An adaptive strategy for hp-FEM based on testing for analyticity*. Comput. Mech. **39** (2007), 575–595.

[5] J. Lang: *Adaptive multilevel solution of nonlinear parabolic PDE systems: Theory, algorithm and applications*. Lecture Notes Comput. Sci. Engrg. 16. Berlin, Springer-Verlag, 2001.

[6] D. Kuzmin, S. Turek: *Numerical simulation of turbulent bubbly flows*. In: G.P. Celata, P. Di Marco, A. Mariani, R.K. Shah (eds.): Two-Phase Flow Modeling and Experimentation. Pisa, Edizioni ETS 2004, Vol. I, pp. 179–188.

[7] P. Solin, J. Cerveny, I. Dolezel: *Arbitrary-level hanging nodes and automatic adaptivity in the hp-FEM*. Math. Comput. Simulation **77** (2008), 117–132.

[8] P. Solin, J. Cerveny, L. Dubcova: *Adaptive multi-mesh hp-FEM for linear thermoelasticity*. Research Report No. 2007-08. Department of Mathematical Sciences, University of Texas at El Paso, to be downloaded at `http://www.math.utep.edu/preprints/2007/2007-08.pdf`. Submitted to Math. Comput. Simulation.

[9] P. Solin, L. Demkowicz: *Goal-oriented hp-adaptivity for elliptic problems*. Comput. Methods Appl. Mech. Engrg. **193** (2004), 449–468.

[10] P. Šolín, K. Segeth, I. Doležel: *Higher-order finite element methods*. Boca Raton, FL, Chapman & Hall/CRC Press, 2004.

[11] A.M.P. Valli, G.F. Carey, A.L.G.A. Coutinho: *Control strategies for timestep selection in simulation of coupled viscous flow and heat transfer*. Commun. Numer. Methods Engrg. **18** (2002), 131–139.