INSTITUTE OF PHYSICS PUBLISHING COMBUSTION THEORY AND MODELLING

Combust. Theory Modelling **8** (2004) 661–682 PII: S1364-7830(04)68717-1

Numerical parameter estimation for chemical models in multidimensional reactive flows

R Becker[1], **M Braack**[2] **and B Vexler**[2]

[1] Laboratoire de Mathematiques Appliquees, Universite de Pau et des Pays de l'Adour, IPRA—Avenue de l'Universite, BP 1155, 64013 Pau Cedex, France
[2] Institute of Applied Mathematics, University of Heidelberg, INF 294, 69120 Heidelberg, Germany

E-mail: roland.becker@univ-pau.fr, malte.braack@iwr.uni-heidelberg.de and boris.vexler@iwr.uni-heidelberg.de

Received 10 September 2003, in final form 6 May 2004
Published 19 July 2004
Online at stacks.iop.org/CTM/8/661
doi:10.1088/1364-7830/8/4/001

Abstract

We present an algorithm for parameter identification in combustion problems modelled by partial differential equations. The method includes local mesh refinement controlled by *a posteriori* error estimation with respect to the error in the parameters. The algorithm is applied to two types of combustion problems. The first one deals with the identification of Arrhenius parameters, while in the second one diffusion coefficients for a hydrogen flame are calibrated.

(Some figures in this article are in colour only in the electronic version)

1. Introduction

Parameter estimation is an important issue in scientific computation. In the context of combustion problems, typical problems are, for instance, the estimation of reaction rates or Arrhenius parameters and the estimation of diffusion coefficients. Since the system of equations that arises is usually very complex, there is a need for 'automatic' algorithms for solving parameter estimation problems. In addition, the extreme stiffness of combustion problems requires appropriate discretizations and optimization algorithms. For computations in two- and three-dimensional domains, algorithms should include automatic mesh refinement in order to reduce to a minimum the numerical effort for achieving a prescribed accuracy in the parameters.

In this work, we present such a method for parameter estimation incorporating automatic mesh refinement on the basis of mathematically rigorous *a posteriori* error estimation. The optimization loop for determining the unknown parameters is intrinsically coupled with local mesh refinement. There are two main advantages:

- At the beginning, the optimization algorithm acts on coarse meshes. When the target functional is sufficiently reduced, the mesh is refined and the optimization continues.

These steps are iterated until a user-specified tolerance is achieved. It allows us to replace optimization iterations on fine meshes with iterations on coarse meshes.

- The decision when and where the mesh is refined is made on the basis of *a posteriori* error estimation. The estimator directly controls the accuracy of the parameters. As a consequence, locally refined meshes are very economic with respect to an error in the parameters.

For discussing the subject, we consider the following simple model problem of a scalar stationary convection–diffusion–reaction equation (cdr equation) for the variable u in a domain $\Omega \subset \mathbb{R}^2$ with a divergence-free vector field β and a diffusion coefficient D:

$$\beta \cdot \nabla u - \operatorname{div}(D\nabla u) + s(u, q) = f \tag{1}$$

with Dirichlet boundary conditions $u = \hat{u}$ at the inflow boundary $\Gamma_{\text{in}} \subset \partial\Omega$ and Neumann conditions $\partial_n u = 0$ on $\partial\Omega \setminus \Gamma_{\text{in}}$. As is usual in combustion problems, the reaction term is of Arrhenius type,

$$s(u, q) := A \exp\left\{\frac{-E}{d-u}\right\} u(c-u). \tag{2}$$

While d, c are fixed parameters, the parameters A, E are considered as unknown and form the vector-valued parameter $q = (A, E) \in \mathbb{R}^2$. Since they are not directly measurable, we assume to have certain measurements $\bar{C} \in \mathbb{R}^{n_m}$, which should match with computed quantities $C(u) \in \mathbb{R}^{n_m}$. Here we may think, e.g. of laser measurements of mean concentrations along fixed lines, see section 5. The calibration of Arrhenius parameters has been done by many scientists, for example by Lohmann [24] for coal pyrolysis, which is frequently used in chemical engineering.

The main ingredients of the algorithm are the use of stabilized finite element discretizations on hierarchies of locally refined meshes, a multigrid algorithm for solving linear subproblems, a special optimization loop and an adaptive algorithm for error control and mesh refinement. The dimension of the parameter space is assumed to be finite and small in comparison with the dimension of the discretized system of the state equations. The proposed optimization algorithm is independent of the type of discretization (finite differences, finite volumes or finite elements). However, the error estimator described in section 4 and used in the numerical results is based on finite elements. The aim of this work is the presentation of the numerical background of the proposed method and its validation through model problems.

We present numerical results for two test problems. The first one deals with the estimation of Arrhenius parameters for a single reaction, as mentioned above (2). In the second example we analyse the diffusion parameters for a hydrogen flame. This system includes the compressible Navier–Stokes equation and nine cdr equations (similar to (2)) for chemical species. To the authors' knowledge, this is the first published work on automatic parameter estimation for multidimensional computation of flames. The method presented in this work can be applied to different aspects of modelling, such as the design of boundary conditions. For an overview of parameter estimation problems in chemistry, we refer to the book by Englezos and Kalogerakis [17]. Therein, many applications of parameter identification in the framework of ordinary differential equations are given. Parameter estimation problems for reactive flows in one space dimension are treated, for instance, by Bock *et al* [27].

This paper is organized as follows. In section 2 we formulate the parameter identification problem as an optimization problem and describe the optimization algorithm for it on the continuous level and the discrete level. The special discretization and stabilization techniques are presented in section 3. Section 4 is devoted to the adaptive mesh refinement algorithm and *a posteriori* error estimation. In section 5, the described algorithms are applied for estimating

the Arrhenius coefficients of the scalar cdr equation (1). A more complex hydrogen flame is analysed in section 6. It includes the equations for compressible flows, a system of cdr equations for nine chemical species and 38 elementary reactions. The parameters considered calibrate a simple diffusion model in order to substitute multicomponent diffusion laws. The conclusion is given in the last section.

2. The optimization algorithm for parameter identification problems

The aim of this section is the description of optimization algorithms for solution of the parameter identification problems in the context of combustion. In section 2.1, we start with the formulation of the parameter identification problem in an abstract form and describe the typical optimization algorithms for it on the continuous level. Here, among other things, we discuss special questions, which arise from the combustion problems, e.g. an alternative to the expensive Newton method. Thereafter, in section 2.2, we discuss a trust-region technique for globalization of the convergence. The development of the described algorithms for the discretized problem is presented in section 2.3.

2.1. The optimization algorithm for the continuous problem

We consider the parameter identification problem in the following abstract form: the state variable u is supposed to be a sum of the function $\hat{u}$ describing the Dirichlet conditions and a function of a Hilbert space V, i.e. $u \in \hat{V} := \hat{u} + V$. The unknown parameter, q, is assumed to be in the space $Q := \mathbb{R}^{n_p}$. In an abstract form, the system of equations for the state variable u reads

$$A(u, q) = f, \tag{3}$$

where $A(u, q)$ is a nonlinear operator acting on the function space $\hat{V} \times Q$ with values in another Hilbert space W. The right-hand side, f, is also in W. The operator A is assumed to be differentiable with respect to u and q with partial derivatives $D_u A$ and $D_q A$, respectively.

Further, the measurable quantities are represented by a linear observation operator $C : \hat{V} \to Z$, which maps the state variable u into the space of measurements $Z := \mathbb{R}^{n_m}$. We assume $n_m \geqslant n_p$. That means, we assume to have at least as many measurements as parameters to be determined. We denote by $\langle \cdot, \cdot \rangle_Z$ the Euclidean scalar product of Z and by $\|\cdot\|_Z$ the corresponding norm. Similar notations are used for the scalar product and norm in the space Q.

The values of the parameters are estimated from a given set of measurements $\bar{C} \in Z$ using a least-squares approach in such a way that we obtain the constrained optimization problem with the cost functional $J : \hat{V} \to \mathbb{R}$:

$$\text{Minimize} \quad J(u) := \tfrac{1}{2}\|C(u) - \bar{C}\|_Z^2 \tag{4}$$

under the constraint (3).

Under a regularity assumption for $D_u A$, the implicit function theorem in Banach spaces (see Dieudonné [16]) implies the existence of an open set $Q_0 \subset Q$, containing the optimal parameter q, and a continuously differentiable solution operator $S : Q_0 \to V$, $q \to S(q)$, so that (3) is fulfilled for $u = S(q)$. Using this solution operator S, we define the reduced observation operator $c : Q_0 \to Z$ by

$$c(q) := C(S(q)),$$

in order to reformulate the problem under consideration as an unconstrained optimization problem with the reduced cost functional $j : Q_0 \to \mathbb{R}$:

$$\text{Minimize} \quad j(q) := \tfrac{1}{2}\|c(q) - \bar{C}\|_Z^2, \qquad q \in Q_0. \tag{5}$$

In general, $c(q)$ is highly nonlinear and includes the numerically expensive solution of (3).

Denoting by $G = c'(q)$ the Jacobian matrix of the reduced observation operator c, the first-order necessary condition $j'(q) = 0$ for (5) reads

$$G^T(c(q) - \bar{C}) = 0. \tag{6}$$

The unconstrained optimization problem (5) is solved iteratively. Starting with an initial guess q^0, the next parameter is obtained by $q^{k+1} = q^k + \delta q$, where the update δq is the solution of the problem

$$H_k \delta q = G_k^T r_k, \tag{7}$$

where

$$r_k := \bar{C} - c(q^k), \qquad G_k := c'(q^k)$$

and H_k is an approximation of the Hessian, $\nabla^2 j(q^k)$, of the reduced cost functional j. Although δq depends on the iterate k, we suppress the index in order to improve the readability. The linear system (7) is small and, therefore, uncritical. The choice of the matrix $H_k \in \mathbb{R}^{n_p \times n_p}$ leads to different variants of the optimization algorithm. We consider the following typical possibilities.

2.1.1. Gauss–Newton algorithm. The choice

$$H_k := G_k^T G_k$$

corresponds to the Gauss–Newton algorithm, which can be interpreted as the solution to the linearized minimization problem,

$$\text{Minimize} \quad \tfrac{1}{2}\|c(q^k) + G_k \delta q - \bar{C}\|^2. \tag{8}$$

The components G_{ij} of the Jacobian G_k can be computed as follows:

$$G_{ij} := \frac{\partial c_i}{\partial q_j}(q^k) = C_i(w_j), \qquad i = 1, \ldots, n_m, \quad j = 1, \ldots, n_p.$$

C_i and c_i denote the components of the observation and the reduced observation operators, respectively, and $w_j \in V$ is the solution of the following tangent problem:

$$D_u A(u^k, q^k) w_j = -D_{q_j} A(u^k, q^k), \qquad j = 1, \ldots, n_p, \tag{9}$$

where $u^k = S(q^k)$. For one Gauss–Newton step, the state equation (3) for $u^k = S(q^k)$ and n_p tangent problems (9) that originate from the same linear operator but with different right-hand sides have to be solved. Note that we suppress the index, k, for the matrix entries, G_{ij}, and the vectors w_j.

2.1.2. Full Newton algorithm. Another possibility is to set

$$H_k := \nabla^2 j(q^k),$$

which leads to the full Newton algorithm. The required Hessian $\nabla^2 j(q^k)$ is given by

$$\nabla^2 j(q^k) = G_k^T G_k + M_k \tag{10}$$

with the matrix $M_k \in \mathbb{R}^{n_p \times n_p}$,

$$M_k = \sum_{i=1}^{n_m} c_i''(q^k)(\bar{C}_i - c_i(q^k)).$$

As before, computation of the Jacobian G_k is required. The entries of the matrix M_k can be computed by a subtle evaluation of several second derivatives of the operator $A(u, q)$ in the directions w_j (the solutions of the tangent problems (9)) and $z \in V$, the solution of the adjoint equation

$$D_u A(u^k, q^k)^* z = -C^* r_k. \tag{11}$$

Since we do not use this method for the combustion problems in this work, we refer to Becker and Vexler [7] for more details. For the convergence theory of the Gauss–Newton and Newton methods, see, e.g. Dennis and Schnabel [15] or Nocedal and Wright [25].

In combustion problems the Gauss–Newton algorithm often shows slow convergence, due to the large least-squares residual $\|C(u) - \bar{C}\|_Z$. The full Newton algorithm has better (local) convergence properties, i.e. it leads to a quadratic convergence. However, for combustion problems, evaluation of the second derivatives of the operator $A(u, q)$ is usually very expensive. Therefore, use of full Newton algorithms is often unattractive or even impossible. We discuss shortly an alternative algorithm that combines the comparative 'low' cost of the Gauss–Newton method and the better convergence properties of the full Newton.

2.1.3. Quasi-Newton algorithm. Based on the ideas of Dennis *et al* [14], we replace the expensive matrix M_k in (10) by an approximation obtained by an update formula. It produces a sequence of iterative computable matrices $\hat{M}_k$ starting with $\hat{M}_0 = 0$:

$$\hat{M}_{k+1} = \hat{M}_k + \frac{1}{y^T \delta q}(xy^T + yx^T) - \frac{x^T \delta q}{(y^T \delta q)^2} yy^T,$$

where

$$y = G_{k+1}^T r_{k+1} - G_k^T r_k,$$
$$x = (G_{k+1}^T - G_k^T) r_{k+1} - \hat{M}_k \delta q.$$

Then, for the matrix H_k we use the following Hessian approximation:

$$H_k := G_k^T G_k + \hat{M}_k.$$

Note that no further equations have to be solved for determination of $\hat{M}_k$.

The matrices $\hat{M}_k$ are chosen in such a way that H_k is a secant approximation of the (exact) Hessian. For derivation and analysis of this update formula, see also [15]. In section 6 we compare this algorithm with the Gauss–Newton method and observe a substantial difference in the required number of iterations.

2.2. Trust-region method

It is well known that the convergence of the algorithms described so far is ensured only if the initial guess, q^0, is in a sufficiently small neighbourhood of the optimal parameter, q. There are two main possibilities for overcoming this difficulty and improving the global convergence: the line search and trust-region methods. A sufficient condition for convergence of a line search is that the matrices H_k are positive definite throughout the optimization algorithm. For the Gauss–Newton method with G_k of full rank, this is always fulfilled. However, for quasi-Newton (or Newton) this sufficient condition is too restrictive for the kind of parameter

identification problems we consider. Therefore, we use trust-region techniques in order to improve the global convergence, see, e.g. Conn *et al* [1] or [25]. In the following, we briefly describe the algorithm we use.

If the matrix H_k is positive definite, the computation of $\delta q \in Q$ in (7) can be interpreted as the solution of a minimization problem (cf (8)):

$$\text{Minimize} \quad m_k(\delta q) := j(q^k) - r_k^T G_k \delta q + \tfrac{1}{2}\delta q^T H_k \delta q, \qquad \delta q \in Q. \tag{12}$$

The cost functional, m_k, of (12) is the so-called local model function, the behaviour of which near the current point, q^k, is similar to that of the actual cost functional, j, defined in (5). However, the local model function, m_k, may not be a good approximation of j for large δq. Therefore, we restrict the search for a minimizer of m_k to a ball (trust region) around q^k. In other words, we replace the problem (12) by the following constrained optimization problem:

$$\text{Minimize} \quad m_k(\delta q), \qquad \text{subject to } \|\delta q\|_Q \leqslant \Delta_k \tag{13}$$

with a trust-region radius Δ_k, to be determined iteratively. This technique is also applied when H_k is not positive definite.

For the convergence properties of the trust-region method, the strategy for choosing the trust-region radius, Δ_k, is crucial. Following the standard approach, we base this choice on agreement between the model function m_k and the cost functional j at the previous iteration. For the increment δq, we define the ratio

$$\rho_k = \frac{j(q^k) - j(q^k + \delta q)}{m_k(0) - m_k(\delta q)}$$

and use it as an indicator of the quality of the local model, m_k. If this ratio is close to 1, there is a good agreement between the model, m_k, and the cost functional, j, for the current step. As a consequence, the trust region is expanded for the next iteration. Otherwise, we do not alter the trust region or shrink it, depending on the distance $|\rho_k - 1|$, see [25] for a precise definition of the algorithm.

The solution of the quadratic minimization problem (13) requires an additional remark. Due to the compactness of the feasible set described by the condition $\|\delta q\|_Q \leqslant \Delta_k$, problem (13) always poses a solution independent of the definiteness of the matrix H_k. If the matrix H_k is positive definite and

$$\|H_k^{-1} G_k^T r_k\|_Q \leqslant \Delta_k,$$

we set $\delta q = H_k^{-1} G_k^T r_k$. Otherwise, the solution δq is searched for on the boundary of the feasible set $\{\delta q \mid \|\delta q\|_Q \leqslant \Delta_k\}$ and is determined by

$$\delta q = (H_k + \lambda I)^{-1} G_k^T r_k,$$

where I is the identity matrix and $\lambda > 0$ is chosen, such that

$$\|\delta q\|_Q = \Delta_k.$$

For computation of λ, the singular value decomposition of H_k is computed and λ is determined by the scalar equation, which is solved using the one-dimensional Newton method, see, e.g. [25] for details.

For the numerical examples in section 5 and 6, the optimization algorithm does not converge without using such globalization techniques.

2.3. The optimization algorithm for the discrete problem

The continuous state equation (3) and several tangent problems (9) have to be discretized. The easiest possibility is to replace these equations with some numerical discrete approximations on a 'sufficient fine' mesh resulting from the uniform refinement of the starting mesh. However, some questions arise naturally: first, how to decide if a mesh is sufficiently fine? Second, are the meshes produced by the uniform refinement economical for computation of parameters? And third, how to design another mesh refinement procedure in order to obtain more efficient meshes? These questions are extremely important in combustion problems because of thin flame fronts arising. Furthermore, in parameter estimation problems the measurements are usually local quantities, which gives rise to the need for appropriate *a posteriori* error estimation and local mesh refinement. The required procedure is described in section 4.

For the adaptive algorithm in section 4, we need an optimization loop for solving the parameter estimation problem on a fixed mesh. Let h denote the mesh size (see section 3.1 for details) and let the discrete function spaces be given by $V_h \subset V$, $\hat{V}_h := \hat{u}_h + V_h$, with an approximation $\hat{u}_h$ of the boundary data $\hat{u}$. Furthermore, let $f_h \in W_h$ be an appropriate approximation of the right-hand side, f. For a fixed parameter $q_h \in Q$, the discrete solution $u_h \in \hat{V}_h$ is determined by the discretized state equation

$$A_h(u_h, q_h) = f_h. \tag{14}$$

The operator A_h results from an arbitrary discretization on the mesh $\mathcal{T}_h$. One may think here of finite difference, finite volume or finite element discretization. For the numerical results in this work, we use the finite element method, and the operator A_h is given by the sum of an appropriate Galerkin projection of A and certain stabilization terms (see section 3 for details). We note that this kind of discretization is not essential for the discrete optimization loop described below. However, the use of finite elements is crucial for derivation of the *a posteriori* error estimation in section 4.

As in section 2.1, we assume the regularity of the discrete operator $D_u A_h$, which implies the existence of a discrete solution operator S_h such that $u_h = S_h(q_h)$ fulfills the discrete state equation (14). Moreover, we introduce the discrete reduced observation operator c_h by setting

$$c_h(q_h) = C(S_h(q_h)).$$

The optimization loop on a given mesh $\mathcal{T}_h$ for the problem

$$\text{Minimize} \quad j_h(q_h) := \tfrac{1}{2}\|c_h(q_h) - \bar{C}\|_Z^2, \qquad q_h \in Q$$

starts with an initial guess for the parameters $q_h^0 \in Q$. Thereafter, the corresponding discrete state u_h^k and the next parameter, q_h^{k+1}, are obtained from the discrete equations

$$\begin{aligned} &u_h^k \in \hat{V}_h: \quad A_h(u_h^k, q_h^k) = f_h, \\ &\delta q_h \in Q: \quad H_h \delta q_h = G_h^T r_h, \qquad r_h := \bar{C} - c_h(q_h^k), \\ &q_h^{k+1} \in Q: \quad q_h^{k+1} = q_h^k + \delta q_h, \end{aligned} \tag{15}$$

where $G_h := c_h'(q_h^k)$ and H_h is the discrete approximation of H_k according to the choice above. The globalization technique formulated for the continuous problems in section 2.2 can be carried over to the discrete case similarly.

3. Discretization by finite elements

In this section, we describe the discretization we use for the numerical results stated later. The operators (3) and (14) are specified for variational formulations. In the applications we

are going to present, there is a large difference between the speed of sound and the fluid velocity. The discretization has to account for this extremely stiff pressure–velocity coupling. Furthermore, we discuss the stabilization techniques for convection-dominated flows.

3.1. Meshes and finite element spaces

For the discretization we use a conforming equal-order Galerkin finite element method defined on quadrilateral meshes $\mathcal{T}_h = \{K\}$ over the computational domain $\Omega \subset \mathbb{R}^2$, with cells denoted by K. The mesh parameter h is defined as a cellwise constant function by setting $h_{|K} = h_K$ and h_K is a diameter of K. The straight parts that make up the boundary, ∂K, of a cell K are called faces.

A mesh $\mathcal{T}_h$ is called regular if it fulfils the standard conditions for a shape-regular finite element mesh, see, e.g., Ciarlet [12]. However, in order to make the mesh refinement easier we allow the cell to have nodes, which lie on the midpoints of the faces of neighbouring cells. But at most one such hanging node is permitted for each face.

The discrete function space $V_h \subset V$ consist of continuous, piecewise polynomial functions (so called Q_1 elements) for all unknowns,

$$V_h = \{\varphi_h \in C(\bar{\Omega}); \varphi_h|_K \in Q_1(K)\},$$

where $Q_1(K)$ is the space of functions obtained by transformations of (isoparametric) bilinear polynomials from a fixed reference unit cell $\hat{K}$ to K. For a detailed description of this standard construction, see [12] or Johnson [23].

The case of hanging nodes requires some additional remarks. There are no degrees of freedom corresponding to these irregular nodes, and the value of the finite element function is determined by pointwise interpolation. This implies continuity and therefore global conformity, i.e. $V_h \subset V$. For implementation details, see, e.g. Carey and Oden [11].

In the context of finite element discretization, we use a weak formulation of the state equation (3) with a form $a(u, q)(\phi)$, such that the operator A in (3) is given by the identification

$$\langle A(u, q), \phi\rangle_{V'\times V} := a(u, q)(\phi) \qquad \forall \phi \in V,$$

where $\langle\cdot, \cdot\rangle_{V'\times V}$ denotes the duality pairing between the Hilbert space V and its dual V'. The right-hand side is in the dual space, $f \in W = V'$. The discrete operator A_h in (14) is determined similarly:

$$\langle A_h(u_h, q_h), \phi_h\rangle_{V_h'\times V_h} := a(u_h, q_h)(\phi_h) \qquad \forall \phi_h \in V_h,$$

where V_h' is the dual space of V_h. For the right-hand side we set $f_h = f_{|V_h}$ due to $V_h \subset V$.

For cdr equation (1), the form $a(u, q)(\phi)$ is obtained by multiplication of equation (1) with test functions $\phi \in V$, integration over the computational domain Ω and partial integration of the diffusive term. The resulting form is

$$a(u, q)(\phi) := \int_\Omega (\beta \cdot \nabla u + s(u, q))\phi \,\mathrm{d}x + \int_\Omega D\nabla u \nabla\phi \,\mathrm{d}x. \tag{16}$$

For several applications, the Galerkin formulation is not stable. For instance, at higher Reynolds numbers, advective terms become unstable. In order to overcome this limitation, stabilization techniques can be used. This question is discussed in the next section.

3.2. Stabilization techniques

For equations involving fluid dynamics, as in our example in section 6, the convective terms have to be stabilized. Furthermore, the pressure–velocity coupling has to be stabilized too

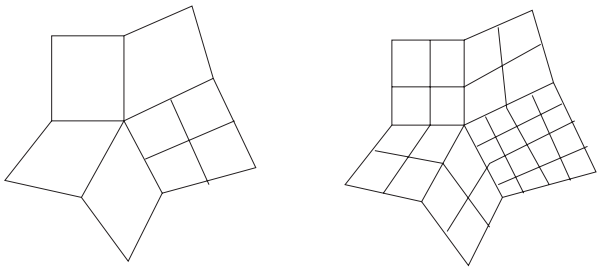

Figure 1. Possible triangulation for approximation of the interpolation error on quadrilaterals with hanging nodes $\mathcal{T}_{2h}$ (left), $\mathcal{T}_h$ (right).

because the Galerkin formulation is known to be unstable for equal-order finite elements. In the following, we explain the specific stabilizations used for the examples presented in this work.

The discrete state equation (14) becomes in stabilized form

$$A_h(u_h, q_h) + S_h(u_h, q_h) = f_h$$

with a discrete approximation, A_h, of A and a stabilization term $S_h(u_h, q_h)$, described below. The Hessian, H_h, in the algorithm (15) is build up by solving discrete versions of (9). Since the operator $D_u A_h(\cdot, \cdot)$ is not necessarily stable, the discrete version of (9) for getting $w_{h,j} \in V_h$ is of the form

$$a'_u(u_h^k, q_h^k)(w_{h,j}, \phi) + \langle S_h(w_{h,j}, q_h^k), \phi)\rangle = -a'_{q_j}(u_h^k, q_h^k)(\phi) \qquad \forall \phi \in V_h$$

with the Frechet derivatives a'_u and a'_{q_j} of $a(u, q)(\cdot)$ with respect to u (in direction $w_{h,j}$) and q_j, respectively. Therefore, even in the case of full Newton iteration, the discrete algorithm does not use the exact derivative of the discrete reduced cost functional j_h but a stable approximation. Hence, the stabilization used is crucial for the whole approach. Our numerical experience indicates that the use of standard stabilization techniques for equal-order finite elements that are based on introducing least-squares terms, such as streamline diffusion, and their modifications (see [23]) is problematic for such optimization purposes. Hence, we proceed differently by using patchwise local projections as described in the following. The resulting algorithm becomes very efficient.

The triangulation $\mathcal{T}_h$ is supposed to be constructed in such a way that it results from a coarser quasi-regular mesh $\mathcal{T}_{2h}$ by one global refinement. By a 'patch' of elements (see figure 1) we denote a group of four cells in $\mathcal{T}_h$ that result from a common coarser cell in $\mathcal{T}_{2h}$. The corresponding discrete finite element spaces, V_{2h} and V_h, are nested: $V_{2h} \subset V_h$. By I_{2h}^h we denote the nodal interpolation operator $I_{2h}^h : V_h \to V_{2h}$. By

$$\pi_h : V_h \to V_h, \qquad \pi_h \xi = \xi - I_{2h}^h \xi,$$

we denote the difference between the identity and this interpolation.

3.2.1. Convection stabilization. For a cdr equation (1), the stabilization term added to the Galerkin formulation reads

$$\langle S_h(u_h, q_h), \phi\rangle := \sum_{K \in \mathcal{T}_h} \delta_K \int_K (\beta \cdot \nabla \pi_h u_h)(\beta \cdot \nabla \pi_h \phi)\, \mathrm{d}x, \tag{17}$$

where the cellwise coefficients, δ_K, depend on the local balance of convection and diffusion:

$$\delta_K := \frac{\delta_0 h_K^2}{6D + h_K \|\beta\|_K}.$$

Here, the quantities h_K and $\|\beta\|_K$ are cellwise values for the cell size and the convection, β. The parameter δ_0 is a fixed constant, usually chosen as $\delta_0 = 0.5$. Note that π_h vanishes on V_{2h},

and therefore the stabilization vanishes for test functions of the coarse grid $\xi \in V_{2h}$. For the scalar convection equation, such a modification of streamline diffusion has been introduced and analysed by Guermond [21].

For systems of cdr equations, for each convective term, one stabilization term of type (17) is added. The convection, β, and the particular diffusion coefficient, D, may depend on u and may be different for each subequation.

3.2.2. Pressure–velocity stabilization. For equal-order finite elements, the Galerkin formulation of the Stokes system for the pressure p and velocity v,

$$\operatorname{div} v = 0,$$
$$-\mu \Delta v + \nabla p = f$$

is known to be unstable since the stiff pressure–velocity coupling for (nearly) incompressible flows enforces spurious pressure modes. The same occurs for hydrodynamic incompressible flows since they also involve the saddle–point structure of the Stokes system. Let p_h denote the discrete pressure, v_h the discrete velocity, $u_h = (p_h, v_h)$ and ξ the test function for the divergence equation. The added stabilization term that damps acoustic pressure modes is of the form

$$\langle S_h(u_h, q_h), \phi \rangle = \sum_{K \in \mathcal{T}_h} \alpha_K \int_K (\nabla \pi_h p_h)(\nabla \pi_h \xi) \, \mathrm{d}x \tag{18}$$

with weights $\alpha_K = \alpha_0 h_K^2 / \mu$, depending on the mesh size, h_K, of cell K and the viscosity, μ. The parameter α_0 is usually chosen between 0.2 and 1. The stabilization term (18) acts as a diffusion term on the fine-grid scales of the pressure. The scaling, proportional to h_K^2, gives stability to the discrete equations and maintains accuracy. This type of stabilization is introduced in Becker and Braack [2] for the Stokes equation. Therein, a stability proof and an error analysis are given. The same stabilization is applied to the (compressible) Navier–Stokes equations.

The proposed stabilization is consistent in the sense that the introduced terms vanish for $h \to 0$. As a consequence, the error in the derivative of j_h vanishes as well.

4. Adaptive mesh refinement via *a posteriori* error estimation

In this section, we describe the adaptive algorithm for mesh refinement and error control based on *a posteriori* error estimation for the parameter identification problems developed in [7]. In order to measure the error in the parameters, we introduce an error functional $E : Q \to \mathbb{R}$. The use of the error functional E allows us to weight the relative importance of the different parameters. The following error representation holds:

$$E(q) - E(q_h) = \eta_h + P + R, \tag{19}$$

where η_h denotes the computable *a posteriori* error estimator and P and R are remainder terms due to linearization and the use of the Gauss–Newton algorithm (see Becker and Vexler [7] for details).

The error estimator is based on the optimal control approach to *a posteriori* error estimation developed in Becker and Rannacher [5,6]. However, a direct application of this approach leads to an estimator that controls the error in the cost functional (4). In general, such an estimator does not provide useful error bounds for the parameters, in contrast to the estimator (19) described in the following paragraphs.

We sketch a generic adaptive mesh refinement algorithm. Such an algorithm generates a sequence of locally refined meshes and corresponding finite element spaces until the estimated

error is below a given tolerance, TOL, with respect to E. For the following iteration, we have a mesh refinement procedure that adaptively refines a given regular mesh to obtain a new regular mesh for the next iteration. The refinement procedure is guided by information based on the nodewise contributions of η_h.

Adaptive mesh refinement algorithm

1. Choose an initial mesh $\mathcal{T}_{h_0}$ and set $l = 0$
2. Construct the finite element space, V_{h_l}
3. Compute the discrete optimal $q_{h_l} \in Q$, i.e. iterate (15)
4. Evaluate the *a posteriori* error estimator, η_{h_l}
5. If $\eta_{h_l} \leqslant$ TOL quit
6. Refine $\mathcal{T}_{h_l} \to \mathcal{T}_{h_{l+1}}$ using information from η_{h_l}
7. Increment l and go to 2

In step 3 the least-squares problem is solved on a fixed mesh. As initial data we use the values from the computation on the previous mesh. This allows us to avoid unnecessary iterations of the optimization loop on fine meshes. A suitable stopping criterion for iterating (15) is, for instance, a fixed relative reduction of the remaining residual,

$$Res := \|G_h^T(\bar{C} - c_h(q_h^k))\| \tag{20}$$

of the optimization condition (6) (in the discrete form). For evaluation of our *a posteriori* error estimator, η_h, we consider an additional adjoint equation for the adjoint variable $y \in V$

$$D_u A(u, q)^* \, y = C^* G (G^T G)^{-1} \nabla E(q) \tag{21}$$

and solve the discrete version of it, i.e. $y_h \in V_h$:

$$D_u A_h(u_h, q_h)^* \, y_h = C^* G_h (G_h^T G_h)^{-1} \nabla E(q_h). \tag{22}$$

We denote by ρ and ρ^* the residuals of the state and the adjoint equations, respectively, i.e. we define for test functions $\phi \in V$,

$$\rho(u_h)(\phi) := f(\phi) - a_h(u_h, q_h)(\phi),$$

$$\rho^*(u_h, y_h)(\phi) := \langle G_h (G_h^T G_h)^{-1} \nabla E(q_h), C(\phi) \rangle_Z - a_u'(u_h, q_h)(\phi, y_h),$$

where a_u' is a partial derivative of the form a, introduced in section 3.1, and corresponds to $D_u A$. Using this notation, the error estimator is given by

$$\eta_h = \tfrac{1}{2}\rho(u_h)(y - i_h y) + \tfrac{1}{2}\rho^*(u_h, y_h)(u - i_h u), \tag{23}$$

where $i_h : V \to V_h$ is an appropriate interpolation operator (see Clement [13]). For simplicity we assume that $\hat{u}_h = \hat{u}$, such that $u - i_h u \in V$. For a proof of (19) with the error estimator given by (23) (see [7]).

For evaluation of this error estimator, the local interpolation errors $y - i_h y$ and $u - i_h u$ have to be approximated. In our numerical examples, we use interpolation of the computed bilinear finite element solutions y_h and u_h on the space of biquadratic finite elements on patches of cells.

The main computational cost for the *a posteriori* error estimator described above is the solution of one auxiliary equation (22). This is cheap, even in comparison with only one Gauss–Newton step, which includes solution of the state (nonlinear) and several (linear) tangent equations.

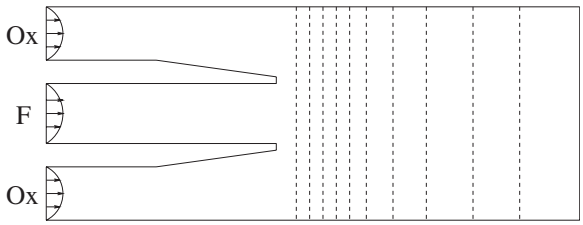

Figure 2. Configuration of the reaction chamber for estimating Arrhenius coefficients. Dashed vertical lines indicate schematically the lines where the measurements are modelled.

For the cdr equation (1), we specify the definition of the residuals ρ and ρ^*. They are given by

$$\rho(u_h)(\phi) = \int_\Omega f\phi \,\mathrm{d}x - \int_\Omega (\beta \cdot \nabla u_h + s(u_h, q_h))\phi \,\mathrm{d}x - \int_\Omega D\nabla u_h \nabla\phi \,\mathrm{d}x$$

and

$$\rho^*(u_h, y_h)(\phi) = \langle G_h(G_h^T G_h)^{-1}\nabla E(q_h), C(\phi)\rangle_Z - \int_\Omega (\beta \cdot \nabla\phi + s'_u(u_h, q_h)(\phi))y_h \,\mathrm{d}x$$
$$- \int_\Omega D\nabla\phi\nabla y_h \,\mathrm{d}x.$$

These residual terms are still global quantities. In order to use it for local mesh adaptation, the estimator η_h has still to be localized to cellwise or nodewise error indicators. For the applications in this work, we perform nodewise localization by summation over all nodes of the mesh. For a mesh $\mathcal{T}_h$ with N nodes, the estimator can be expressed by

$$\eta_h = \sum_{i=1}^{N} \tau_i.$$

Then, the mesh is locally refined with respect to the error indicators $\eta_i := |\tau_i|$. For more details on the localization procedure used, we refer to Braack and Ern [10]. However, there are also methods for localization to cellwise quantities, see [6].

In the case where the quantity to be controlled is $E(q - q_h)$, there are two possible situations. At first, E is linear. Then $E(q - q_h) = E(q) - E(q_h)$, and we are in the situation discussed earlier. Second, when E is nonlinear, a relevant possibility is the case of a norm $E(q - q_h) = \|q - q_h\|$. *A posteriori* error control of norms heavily depends on the kind of norm used and is beyond the scope of this article. However, the optimization algorithm presented may kept completely unchanged.

5. Identification of Arrhenius parameters

The first example we analyse with respect to the proposed optimization algorithm is the scalar cdr equation (1) with $f \equiv 1$, $D = 10^{-6}$ and a chemical source term of Arrhenius type (2). The variable u stands for the mole fraction of a fuel, while the mole fraction of the oxidizer is $0.2 - u$. Since the Arrhenius law is a heuristic law and cannot be derived from physical laws, the involved parameters are *a priori* unknown and have to be calibrated. This parameter fitting is usually done by comparison of experimental data and simulation results. Therefore, this example is well suited for the proposed parameter identification algorithm.

Fuel (F) and oxidizer (Ox) are injected in different pipes and diffuse in a reaction chamber with overall length 35 mm and height 7 mm (figure 2). At the centre tube, the Dirichlet condition for the fuel is $u = u_{\text{in}} := 0.2$, and at the upper and lower tubes, $u = 0$. On all

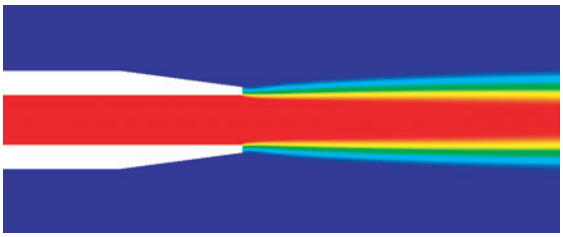

Figure 3. Mole fraction of the fuel (u^0) for the initial parameters q^0. Blue (outside edges): $u = 0$. Red (central): $u = 0.2$.

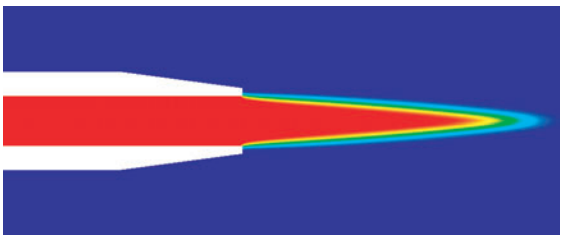

Figure 4. Mole fraction of the fuel (u) for the optimal parameters q (right).

other parts of the boundary, homogeneous Neumann conditions are used. The fixed parameters in the Arrhenius law (2) are $c = u_{\text{in}}$ and $d = 0.24$. The convection direction, $\beta(x, y)$, is a velocity field obtained by solving the incompressible Navier–Stokes equations with a parabolic inflow profile at the tubes with peak flow $\beta_{\max} = 0.2\,\text{m}\,\text{s}^{-1}$. The initial parameters are set to $q^0 = (\log(A^0), E^0) = (4, 0.15)$, leading to low reaction rates and a diffusion-dominated solution. In figure 3 the corresponding state variable (fuel), u^0, is shown.

Instead of experimental data, we choose the optimal parameters to be $q = (6.9, 0.07)$ and replace the measurements with computations mode using these parameters: $\bar{C} := C(S(q))$. As a consequence, the 'measurements' perfectly match for the optimal parameters. This will not be the case in the second example. The state variable $u = S(q)$ is shown in figure 4. For the optimal q, in contrast to the initial guess, q^0, a sharp reaction front occurs. Obviously, the difference in the parameters has a substantial impact on the state variable, u.

The measurements $C(u) \in \mathbb{R}^{n_m}$ are modelled by mean values along $n_m = 10$ straight lines Γ_i at different positions in the reaction chamber (see dashed lines in figure 2):

$$C_i(v) = \int_{\Gamma_i} v \,\mathrm{d}x, \qquad i = 1, \ldots, n_m.$$

For the error functional, we choose the discretization error with respect to the second parameter $E(q) = q_2$. In the optimization loop, we use the Gauss–Newton algorithm with the trust-region strategy described before.

In table 1, the results obtained are listed. The third column displays the corresponding cost functional (difference in the observations). On the first mesh with only $N = 1664$ nodes, eight iterations (see second column) are done. On this mesh, the cost functional is reduced by more than two digits. In the fourth column, the remaining residual, Res, given in (20) is listed. The last two columns show the corresponding parameters obtained. After a reduction of Res by a factor of 10^{-2}, the mesh is adapted locally according the *a posteriori* error estimator, η_h. The second mesh has 2852 nodes. Here, the optimization loop is repeated. However, on the finer meshes, only a few ($\leqslant 3$) iterations are necessary. On the finest mesh, the error in the first parameter is about 0.03% and in the second parameter about 0.3%.

Table 1. Numerical results for Arrhenius parameter identification for $E(q) = q_2$.

N	it	$\|C(u) - \bar{C}\|$	Res	q_1	q_2
1 664	1	5.87e−2	2.54e−3	4.000	1.500e−1
	2	5.86e−2	2.56e−3	4.001	1.499e−1
	3	5.81e−2	2.83e−3	4.132	1.499e−1
	4	4.47e−2	6.42e−3	5.630	1.489e−1
	5	2.36e−2	5.58e−3	6.752	1.481e−1
	6	6.22e−3	1.90e−3	7.433	1.093e−1
	7	8.34e−4	2.55e−4	6.660	4.621e−2
	8	3.00e−4	1.00e−4	6.825	6.394e−2
2 852	1	4.79e−4	1.44e−4	6.798	6.216e−2
	2	3.68e−5	1.04e−5	6.905	7.134e−2
	3	1.92e−5	5.51e−8	6.906	7.158e−2
6 704	1	2.33e−4	7.72e−5	6.906	7.158e−2
	2	1.42e−5	1.91e−8	6.904	7.066e−2
13 676	1	6.91e−5	2.30e−5	6.904	7.063e−2
	2	3.53e−6	6.76e−9	6.905	7.052e−2
21 752	1	1.22e−5	3.24e−6	6.905	7.052e−2
	2	2.84e−6	8.89e−9	6.902	7.022e−2
Exact				6.900	7.000e−2

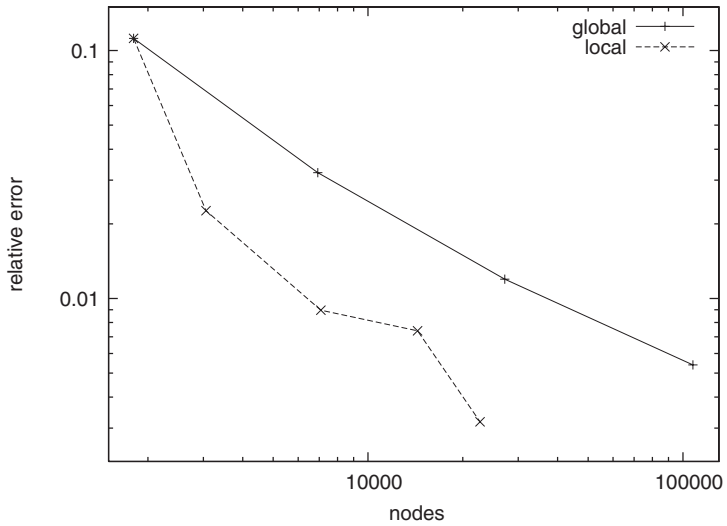

Figure 5. Relative error in the second Arrhenius parameter in dependence of the number of mesh points. ——: globally refined meshes; - - - -: locally refined meshes on the basis of *a posteriori* error estimation.

Comparing the error in the parameters with a more conventional strategy on globally refined meshes, our proposed algorithm is much more efficient. In figure 5, the absolute difference in the second parameter is plotted as a function of the number of mesh points. The dashed line represents results from our method on locally refined meshes. The solid line stands for parameter estimation with the same optimization loop but on uniformly refined meshes. For a relative error of less than 1%, only 6704 nodes are necessary with a locally refined mesh, whereas more than 100 000 nodes are necessary on a uniformly refined mesh.

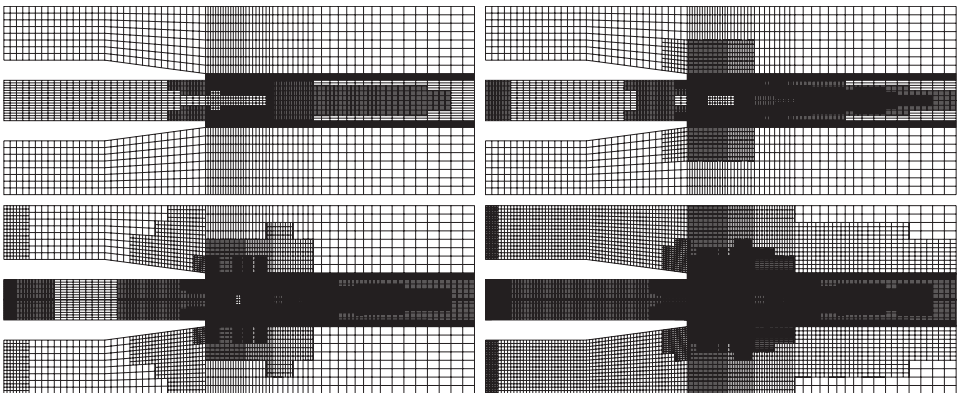

Figure 6. Meshes obtained for estimating Arrhenius parameters with 2852, 6704, 13 676 and 21 752 nodes (from upper left to lower right).

Table 2. Numerical results for Arrhenius parameter identification for $E(q) = q_1 + q_2$.

N	q_1	q_2
1 664	6.825	6.394e−2
3 044	6.897	7.018e−2
7 280	6.898	6.990e−2
17 660	6.902	7.024e−2
37 280	6.900	7.007e−2
Exact	6.900	7.000e−2

In figure 6, a sequence of locally refined meshes produced by the refinement algorithm is shown. After optimization on the first mesh the parameters are already closer to the optimum, so that successive refinement does not resolve a flame front at a wrong position. The layers of small cells off the flame front are due to anisotropic elements along the x-axis on the initial mesh for resolving the thin burner lips. These cells are simply maintained during the adaptive process. The highest amount of mesh points is located near the flame front and close to the region where fuel and oxidizer mix at first and close the vertical measurement lines.

In order to show the flexibility of our approach, we perform the same optimization but with a different error functional $E(q) := q_1 + q_2$. The optimized values on several meshes are given in table 2. These show definitely better accuracy in the first parameter.

6. Identification of diffusion parameters

6.1. Underlying system of equations

In this example, we consider a stationary hydrogen diffusion flame modelled by the following system of equations for velocity v, pressure p, temperature T and mass fraction y_k:

$$\begin{aligned}
&\operatorname{div}(\rho v) = 0,\\
&(\rho v \cdot \nabla)v + \operatorname{div}\pi + \nabla p = 0,\\
&\rho v \cdot \nabla T - \frac{1}{c_p}\operatorname{div}\mathcal{Q} = -\frac{1}{c_p}\sum_{i\in\mathcal{S}} h_i f_i,\\
&\rho v \cdot \nabla y_k + \operatorname{div}\mathcal{F}_k = f_k, \qquad k \in \mathcal{S}.
\end{aligned}$$

The specific enthalpies are denoted by h_i and the heat capacity at constant pressure is denoted by c_p. Both quantities are evaluated using thermodynamic databases. The enthalpy flux is neglected. The set $\mathcal{S}$ denotes the set of chemical species. The density, ρ, is given by the perfect gas law in a mixture with partial molecular weights m_i and a uniform gas constant, R:

$$\rho = \frac{p}{RT}\left(\sum_{i\in\mathcal{S}}\frac{y_i}{m_i}\right)^{-1}.$$

The stress tensor, π, is given as usual for compressible flows:

$$\pi = -\mu\{\nabla v + (\nabla v)^T - \tfrac{2}{3}(\operatorname{div} v)I\},$$

where μ denotes the viscosity, depending on the temperature and the mixture fractions. The reaction terms, f_i are given by a set $\mathcal{R}$ of elementary reactions:

$$f_i := m_i \sum_{r\in\mathcal{R}} (\nu'_{ri} - \nu_{ri}) k_r \prod_{s\in\mathcal{S}} c_s^{\nu_{rs}}.$$

The stoichiometric coefficients of the products and educts for reaction r are denoted by ν'_{ri} and ν_{ri}, respectively. The concentration, c_i, of species i is given by $c_i = \rho y_i/m_i$. The reaction rates, k_r, of a forward reaction, $r \in \mathcal{R}$, is modelled by an Arrhenius law,

$$k_r = A_r T^{\beta_r} \exp\left\{-\frac{E_r}{RT}\right\},$$

while the reaction rate, k_r^b, of the corresponding reverse reactions, $r_b \in \mathcal{R}$, is obtained as $k_r^b = k_r/k_r^e$, with the equilibrium reaction rate

$$k_r^e = \left(\frac{p}{R}\right)^{-\beta_r^e} T^{\beta_r^e} \exp\left\{-\frac{1}{RT}\sum_{j\in S}(\nu'_{rj} - \nu_{rj}) m_j (h_j - T s_j)\right\},$$

where $s_j = s_j(T)$ denotes the species entropy and

$$\beta_r^e = -\sum_{j\in S}(\nu'_{rj} - \nu_{rj}).$$

Since the thermodynamic pressure is constant, we use the isobaric approximation for the chemical source terms and the transport mechanisms. The heat flux, $\mathcal{Q}$, is given by the Fourier law,

$$\mathcal{Q} = -\lambda\nabla T$$

with λ the heat conductivity. The species fluxes, $\mathcal{F}_k$, are modelled by a simple Fick's law:

$$\mathcal{F}_k = q_k D_k^* \nabla y_k.$$

The scaling parameters, q_k, are the free parameters that have to be calibrated in the optimization procedure. Following Hirschfelder and Curtiss [22], the diffusion coefficients in the mixture D_k^* are obtained as

$$D_k^* = (1 - y_k)\left(\sum_{l\neq k}\frac{x_l}{D_{kl}^{\text{bin}}}\right)^{-1}$$

with binary diffusion coefficients D_{kl}^{bin} and mole fractions x_l. In order to ensure that the sum over all species mass fractions is 1 and to have a consistent model, the inert species (N_2) is erased from the set of unknown species.

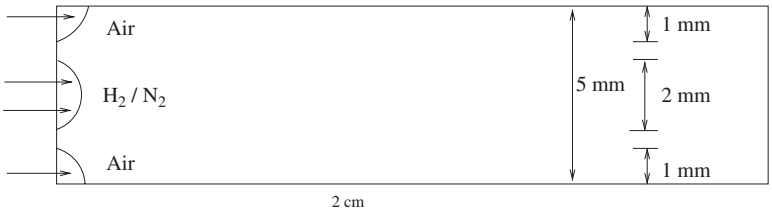

Figure 7. Schematic set-up for the hydrogen diffusion flame.

6.2. Configuration of a hydrogen diffusion flame

The set-up for this hydrogen diffusion flame is taken from Braack and Ern [9] and is shown schematically in figure 7. At the inflow boundary of the centre tube, a 10% mass fraction of hydrogen and 90% nitrogen are placed. At the upper and lower tubes, $y_{O_2} = 0.22$ and $Y_{N_2} = 0.78$ are prescribed. The peak velocity of the three parabolic velocity profiles is $1\,\mathrm{m\,s^{-1}}$. A low temperature, $T = 273\,\mathrm{K}$, is prescribed at the inflow boundaries. On the other three boundaries (upper, lower, right), homogeneous Neumann conditions for temperature and mass fractions are opposed. For the velocities, the upper and lower wall are no-slip walls. The right boundary is the natural 'do nothing' outflow boundary condition. The chemical model involves nine chemical species (see the mechanism in table 4 of the appendix).

The initial parameters are set to $q_0 = (1, \ldots, 1) \in \mathbb{R}^{n_p}$, $n_p = 8$ so that Fick's law with conventional diffusion parameters is recovered. Due to the rich fuel condition at the centre tube, there is not enough oxygen for complete burning of the fuel. This kind of flame is called underventilated. However, the peak temperature reaches more than 2000 K in the burnt gas. In figure 8, a magnification of the resulting mass fraction of hydrogen peroxide, H_2O_2, is shown, indicating the flame front.

We substitute the experimental data by computations of the same flame but with a more sophisticated species diffusion model, namely multicomponent diffusion,

$$\mathcal{F}_k^M = -\rho y_k \sum_{l \in \mathcal{S}} D_{kl}(\nabla x_l + \chi_l \nabla(\log T)), \qquad k = 1, \ldots n_s \tag{24}$$

with species diffusion coefficients D_{kl} and thermal thermal diffusion ratios χ_l for describing the Soret effects (see Ern and Giovangigli [19]). The computations are done on a very fine mesh. The fluxes (24) are evaluated using the EGLIB library [18]. The corresponding flame is shown in figure 9, showing a qualitatively different flame front. A heavy impact on diffusion models for hydrogen flames is also observed in [20]. We wish to emphasize that this setting is not chosen to propose better constants for Fick's laws but to show the capability of the parameter identification algorithm.

In contrast to the previous example, the observation values, $\bar{C} \in \mathbb{R}^{n_m}$, consist of 640 point values of mass fractions of hydrogen peroxide. These 'observation points' are distributed equidistant in the region of the flame front, $\Omega' = [0, 3.125\,\mathrm{mm}] \times [0, 2.5\,\mathrm{mm}]$, where we have obtained reference values by computations with multicomponent diffusion models. In contrast to the first example in section 5, the observation operator for the optimized parameters will not match with $\bar{C}$ because Fick's law and multicomponent diffusion are qualitatively different. However, one may expect that optimized parameters will enhance the diffusion model at least with respect to the observations, i.e. point values of H_2O_2 in Ω'. The error functional in this example was chosen as $E(q) := q_1$.

Figure 8. Mass fractions of H_2O_2 for the initial diffusion model (Fick's law).

Figure 9. Mass fractions of H_2O_2 for the multicomponent diffusion model.

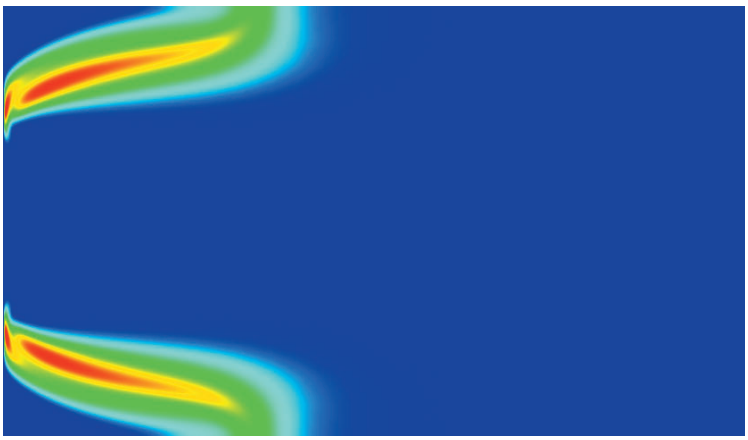

Figure 10. Mass fractions of H_2O_2 for the calibrated Fick's diffusion model.

Table 3. Numerical results for identification of diffusion coefficients.

N	it	$\|C(u) - \bar{C}\|$	Res	q_1	q_2	q_3	q_6
30 353	1	2.439e−4	1.012e−08	0.4367	0.4529	1.077	1.000
	4	2.434e−4	3.586e−09	0.4347	0.4533	1.068	1.001
	8	2.309e−4	4.574e−09	0.3146	0.4561	1.100	1.979
	12	2.193e−4	4.127e−09	0.2675	0.4549	1.096	18.60
	14	2.181e−4	9.471e−10	0.2692	0.4543	1.087	18.61

6.3. Computational results for the hydrogen diffusion flame

The difference in the observation of the initial parameters, q^0, is $J(u^0) = 1.06 \times 10^{-3}$. After optimization, we obtain the optimized parameters for the species ordering H, H_2, O_2, OH, H_2O, HO_2, H_2O_2, O:

$$q = (0.269, 0.454, 1.09, 2.46, 0.946, 18.6, 0.579, 0.179),$$

which corresponds to $J(u) = 2.1 \times 10^{-4}$. In table 3, we list the evolution of several parameters on the finest mesh, with 30 053 nodes. While the first and sixth parameters for the minor species H and HO_2, respectively, are still changing a lot, the parameters for the educts hydrogen H_2 (q_2) and oxygen O_2 (q_3) remain relatively stable. Although q_6 changes between iteration 8 and 12 by a factor of nearly 10, the residual Res remains nearly the same. Hence, the reduced cost functional, $j(q)$, is almost insensitive with respect to q_6. This explains also the large optimal value $q_6 = 18.6$. Such effects can be overcome by reducing the set of free parameters or, even better, by using singular value decompositions of H_k, allowing us to freeze linear combinations of parameters. For a detailed description of these techniques we refer to Bock [8].

A comparison of H_2O_2 for multicomponent diffusion in figure 9 and the corresponding solutions with the parameter-fitted Fick's law, given in figure 10, shows a substantial improvement.

With respect to the numerical algorithm, we observe that the convergence rate for the Gauss–Newton algorithm (see section 2.1.1) is not satisfactory and that the number of iterations is too large. This behaviour can be explained by the fact that even for the optimal parameters, the least-squares residual, $\|C(u) - \bar{C}\|_Z$, does not vanish, nor does the second derivative of $c(q)$. A comparison of the two approaches is given at the end of this section.

In view of this experience, we used for the computation of the hydrogen flame the method with updates for one part of the Hessian (see section 2.1.3). The computations are done on a sequence of four locally refined meshes. On each of these meshes, approximately 10 iterations are needed for reducing the residual Res, by a factor of 10. The overall computing time for determination of the eight parameters on subsequentially refined meshes was about 24 h on a PC Pentium IV running at 2.8 Ghz. This includes about 40 forward solutions (equation (14)) and about $40 \times 8 = 320$ tangent problems that are the discrete versions of equations (9). We show a sequence of parts of the locally refined meshes for this optimization problem in figure 11.

6.4. Comparison of Gauss–Newton and quasi-Newton methods

In order to illustrate the difference between the performance of the Gauss–Newton method and the update method used (see section 2.1.3), we compare the two methods for a similar problem (but less numerically expensive): an ozone decomposition flame with three species and three reactions (see [3]). The resulting residuals of the optimization condition (6) are plotted as a function of the number of iterations in figure 12. While the Gauss–Newton algorithm needs

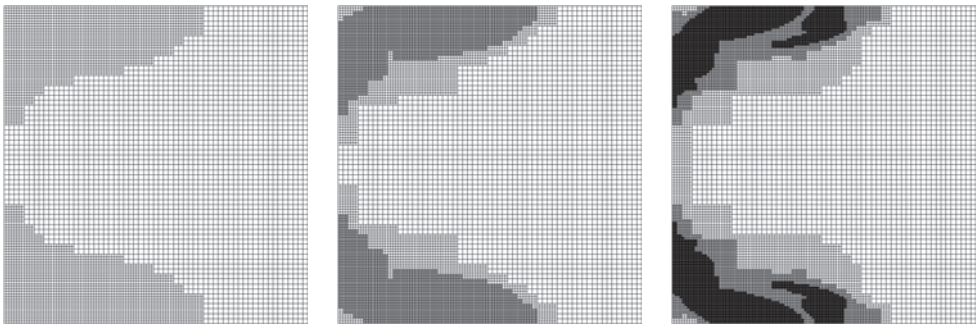

Figure 11. A sequence of zoom-ins ([0, 5 mm] × [0, 5 mm]) of locally refined meshes for the hydrogen flame at the second, third and fourth iterations.

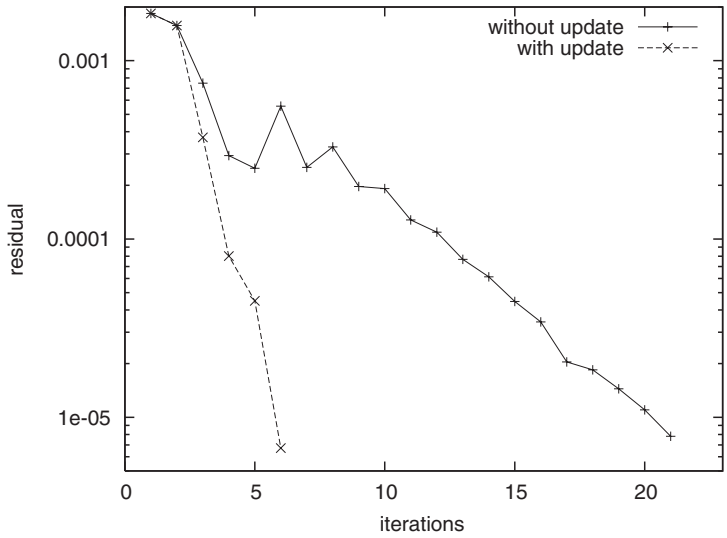

Figure 12. Comparison of Gauss–Newton iterations and the update method of section 2.1.3. The y-axis shows Res, the x-axis the number of iterations.

26 iterations for reducing the residual, Res, given by (20) down to 10^{-5}, only six iterations are needed when the matrices $\hat{M}_k$ are computed. For more details, we refer to [4].

7. Summary and outlook

We proposed an algorithm for parameter identification for partial differential equations including local mesh refinement. The adaptivity is handled by *a posteriori* error estimation with respect to the error in the parameters. Furthermore, we show the performance of the proposed method with two types of combustion problems. In the first one, Arrhenius parameters are estimated. In the second example, diffusion coefficients are calibrated in order to match as well as possible with a more complex diffusion model.

In future work, this concept will be applied to parameter identification problems with real experimental data.

Acknowledgments

This work has been supported by the German Research Foundation (DFG) through the *Sonderforschungsbereich 359* 'Reactive Flows, Diffusion and Transport' and the *Graduiertenkolleg* 'Complex Processes: Modeling, Simulation and Optimization' at the Interdisciplinary Center of Scientific Computing (IWR), University of Heidelberg. We thank J Warnatz for providing the hydrogen mechanism and the thermodynamical and molecular databases. Furthermore, we thank A Ern for a fruitful discussion on the underventilated flame of section 6 and for providing the library EGLIB [18] for our purposes.

Appendix

Here we include the hydrogen/air reaction mechanism taken from Warnatz *et al* [26] for modelling of the hydrogen flame of section 6.

Table 4. Reaction mechanism used for hydrogen combustion. The collision efficiencies are $M(H_2, O_2, H_2O, N_2) = (1, 0.4, 6.5, 0.4)$.

Reaction	A_r	β_r	E_{ar}
$O_2 + H = OH + O$	2.000×10^{14}	0	70.3
$H_2 + O = OH + H$	5.060×10^{04}	2.67	26.3
$H_2 + OH = H_2O + H$	1.000×10^{08}	1.6	13.8
$OH + OH = H_2O + O$	1.500×10^{09}	1.14	0.42
$H + H + M = H_2 + M$	1.800×10^{18}	-1	0
$O + O + M = O_2 + M$	2.900×10^{17}	-1	0
$H + OH + M = H_2O + M$	2.200×10^{22}	-2	0
$H + O_2 + M = HO_2 + M$	2.300×10^{18}	-0.8	0
$HO_2 + H = OH + OH$	1.500×10^{14}	0	4.2
$HO_2 + H = H_2 + O_2$	2.500×10^{13}	0	2.9
$HO_2 + H = H_2O + O$	3.000×10^{13}	0	7.2
$HO_2 + O = OH + O_2$	1.800×10^{13}	0	-1.7
$HO_2 + OH = H_2O + O_2$	6.000×10^{13}	0	0
$HO_2 + HO_2 = H_2O_2 + O_2$	2.500×10^{11}	0	-5.2
$OH + OH + M = H_2O_2 + M$	3.250×10^{22}	-2	0
$H_2O_2 + H = H_2 + HO_2$	1.700×10^{12}	0	15.7
$H_2O_2 + H = H_2O + OH$	1.000×10^{13}	0	15
$H_2O_2 + O = OH + HO_2$	2.803×10^{13}	0	26.8
$H_2O_2 + OH = H_2O + HO_2$	5.400×10^{12}	0	4.2

References

[1] Conn N G A R and Toint P 2000 *Trust-Region Methods* (Philadelphia: SIAM, MPS)
[2] Becker R and Braack M 2001 A finite element pressure gradient stabilization for the Stokes equations based on local projections *Calcolo* **38** 173–99
[3] Becker R, Braack M and Rannacher R 1999 Numerical simulation of laminar flames at low Mach number with adaptive finite elements *Combust. Theory Modelling* **3** 503–34
[4] Becker R, Braack M and Vexler B 2004 Parameter identification for chemical models in combustion problems *Appl. Numer. Math.* at press
[5] Becker R and Rannacher R 1996 A feed-back approach to error control in finite element methods: basic analysis and examples *East-West J. Numer. Math.* **4** 237–64
[6] Becker R and Rannacher R 2001 An optimal control approach to *a posteriori* error estimation in finite element methods *Acta Numerica 2001* ed A Iserles (Cambridge: Cambridge University Press)

[7] Becker R and Vexler B 2004 *A posteriori* error estimation for finite element discretization of parameter identification problems *Numer. Math.* **96** 435–59
[8] Bock H-G 1987 Randwertproblemmethoden zur parameteridentifizierung in systemen nichtlinearer differentialgleichungen, Bonner Mathematische Schriften Nr. 183 *PhD Thesis* Universität Bonn
[9] Braack M and Ern A 2004 Coupling multimodelling with local mesh refinement for the numerical solution of laminar flames *Combust. Theory Modelling* at press
[10] Braack M and Ern A 2003 *A posteriori* control of modeling errors and discretization errors *Multiscale Model. Simul.* **1** 221–38
[11] Carey G and Oden J 1984 *Finite Elements, Computational Aspects* vol III (Englewood Cliffs, NJ: Prentice-Hall)
[12] Ciarlet P 1978 *Finite Element Methods for Elliptic Problems* (Amsterdam: North-Holland)
[13] Clément P 1975 Approximation by finite element functions using local regularization *RAIRO Anal. Numer.* **9** 77–84
[14] Dennis J, Gay D and Welsch R 1981 An adaptive nonlinear least-squares algorithm *ACM Trans. Math. Softw.* **7** 348–68
[15] Dennis J and Schnabel R 1996 Numerical methods for unconstrained optimization and nonlinear equations. *Classics Appl. Math. SIAM*
[16] Dieudonné J 1960 *Foundation of Modern Analysis* (New York: Academic)
[17] Englezos P and Kalogerakis N 2001 *Applied Parameter Estimation for Chemical Engineers* (New York, Basel: Marcel Dekker)
[18] Ern A and Giovangigli V http://www.cmap.polytechnique.fr/www.eglib EGlib server with user's manual
[19] Ern A and Giovangigli V 1994 *Multicomponent Transport Algorithms* (*Lecture Notes in Physics* vol m24) (Berlin: Springer)
[20] Ern A and Giovangigli V 1998 Thermal diffusion effects in hydrogen–air and methane–air flames *Combust. Theory Modelling* **2** 349–72
[21] Guermond J-L 1999 Stabilization of Galerkin approximations of transport equations by subgrid modeling. *Modél. Math. Anal. Numér.* **33** 1293–316
[22] Hirschfelder J O and Curtiss C F 1949 *Flame and Explosion Phenomena* (Baltimore: Williams and Wilkins)
[23] Johnson C 1987 *Numerical Solution of Partial Differential Equations by the Finite Element Method* (Cambridge: Cambridge University Press)
[24] Lohmann T W 1996 Modelling of reaction kinetics in coal pyrolysis *Proc. International Workshop: Modelling of Chemical Reaction Systems* ed J Warnatz and F Behrendt (Heidelberg: IWR)
[25] Nocedal J and Wright S 1999 Numerical optimization *Springer Series in Operations Research* (New York: Springer)
[26] Warnatz J, Maas U and Dibble R W 2001 *Combustion* 3rd edn (New York: Springer)
[27] Ziesse M W, Bock H G, Gallitzendörfer J V and Schlöder J P 1996 Parameter estimation in multispecies transport reaction systems using parallel algorithms *Parameter Identification and Inverse Problems in Hydrology, Geology and Ecology* ed J Gottlieb and P DuChateaux (Dordrecht: Kluwer)