Journal of Computational Physics **169,** 44–79 (2001)
doi:10.1006/jcph.2001.6709, available online at http://www.idealibrary.com on IDEAL®

Explicit Time-Scale Splitting Algorithm for Stiff Problems: Auto-ignition of Gaseous Mixtures behind a Steady Shock

Mauro Valorani* and Dimitrios A. Goussis†,[1]

**Department of Mechanical and Aeronautical Engineering, University of Rome "La Sapienza," Italy, Via Eudossiana 18, 00184 Rome, Italy; and †Ag. Georgiou 49, 26500 Rio, Greece*
E-mail: valorani@dma.ing.uniroma1.it; dagoussi@iceht.forth.gr

Received April 5, 2000; revised December 1, 2000

A new explicit algorithm based on the computational singular perturbation (CSP) method is presented. This algorithm is specifically designed to solve stiff problems, and its performance increases with stiffness. The key concept in its structure is the splitting of the fast from the slow time scales in the problem, realized by embedding CSP concepts into an explicit scheme. In simple terms, the algorithm marches in time with only the terms producing the slow time scales, while the contribution of the terms producing the fast time scales is taken into account at the end of each integration step as a correction. The new algorithm is designed for the integration of stiff systems of PDEs by means of explicit schemes. For simplicity in the presentation and discussion of the different features of the new algorithm, a simple test case is considered, involving the auto-ignition of a methane/air mixture behind a normal shock wave, which is described by a system of ODEs. The performance of the new algorithm (accuracy and computational efficiency) is then compared with the well-known LSODE package. Its merits when used for the solution of systems of PDEs are discussed. Although when dealing with a stiff system of ODEs the new algorithm is shown to provide equal accuracy with that delivered by LSODE at the cost of higher execution time, the results indicate that its performance could be superior when facing a stiff system of PDEs. © 2001 Academic Press

Key Words: ordinary differential equation; partial differential equation; stiffness; chemical kinetics; computational singular perturbation.

[1] Present Address: Institute of Chemical Engineering and High Temperature Processes; P.O. Box, 1414, 26500 Rio, Greece.

1. INTRODUCTION

Much attention has recently been devoted to the inclusion of detailed chemical kinetic mechanisms in the simulation of problems in the fields of combustion, hypersonic flows, and pollutant control. However, the extremely fast time-scales introduced by detailed chemistry make the set of governing equations stiff and their numerical solution prohibitively expensive.

The most successful attempts to cope with stiffness have been so far based on implicit schemes. Such a scheme is the family of multistep, variable order, variable integration step implicit BDF methods of Gear, Hindmarsh and Brown [1, 48] (available in the LSODE general purpose package), which is among the most widely used technique to solve stiff ordinary differential equations (ODEs). See for example the CHEMKIN [2, 3] and LSENS [4] packages. Reacting flows in the hypersonic regime [5–7], reactive mixing layers [8], flames [9], detonations [10–12] and nonequilibrium nozzle flows [13] are modeled by systems of partial differential equations (PDEs) and are frequently solved by a local implicit treatment of the stiff source terms according to the *method of lines* and the *time-step* or *operator* splitting approach. Preconditioning techniques are also used to solve steady state problems efficiently [11, 14, 52].

The implicit treatment of the stiff terms provides solutions which are accurate at the slow scales and stable at the fast scales [11, 15]. However, a significant fraction of the total computational time is devoted to solution of the resulting nonlinear systems of algebraic equations, the dimension of which is proportional to the number of chemical species in the detailed mechanism. This is the reason why the relatively limited effort required to deal with simple kinetic mechanisms, such as the one which describes air dissociation in the hypersonic regime, quickly grows dramatically in the modeling of combustion of complex hydrocarbon mixtures. This circumstance prompted the search for reduced chemical mechanisms obtained by applying the steady-state and partial equilibrium approximations to the detailed mechanisms [16–22].

In contrast, explicit schemes are simpler to implement; they provide solutions of high-order accuracy at all scales and do not require the solution of algebraic systems at each integration step. However, their stability requirements force the maximum integration step to be of the order of the fastest (smallest) time-scale. When the problem is stiff, the ratio between the fastest scale and the scale by which the process evolves may grow very large (i.e., several orders of magnitude). In such a case, the calculations might progress uselessly at a very slow pace.

There have been several attempts to design explicit schemes able to deal with stiff problems. Among these, we can cite the contributions of Lebedev [23] and Medovikov [24], who developed a family of stabilized Runge–Kutta multistage schemes, the Runge–Kutta–Chebychev (RKC) schemes, possessing extended real negative stability intervals, and the review article of Verwer [25] on explicit schemes for stiff problems. The stability limits of RKC schemes is roughly proportional to the square of the number of stages [25]. Therefore, higher stability limits results in higher CPU costs. As a consequence, the RKC schemes are efficient only when employed for the solution of mildly stiff problems. In contrast, for severely stiff problems, i.e. when the driving time-scale is several orders of magnitude slower than the fast ones, the efficiency of the explicit RKC schemes equals that of the implicit ones.

Here, a new explicit algorithm is presented which circumvents the stability limitations of the standard and the extended stability explicit schemes by resorting to the concepts embodied in the computational singular perturbation (CSP) method. The original ideas and the mathematical background on which the CSP method is based were presented in Refs. [26–28, 40]. Successful application of CSP in a number of problems involving chemical kinetics, combustion, and optimal control are demonstrated in Refs. [29–32] and [33–35, 42], respectively.

Given a stiff system of ODEs, the steps to construct, on the basis of CSP concepts, an efficient explicit scheme able to deal with very stiff problems are: (i) to split the contributions of the slow and fast time-scales, (ii) to proceed to the next point in time by integrating numerically the slow scales only, and (iii) to separately account for the contribution of the fast scales at the end of each integration step by means of an algebraic correction term.

With stiffness removed, the scheme employed for the integration of the terms, which contribute the slow scale, can be of explicit type. A fourth-order four-stage Runge–Kutta scheme is used in the computations reported in this paper. Of course, any other variant can in principle be used.

Because of this specific feature, the new algorithm can be described as a *time-scale splitting explicit scheme* to emphasize its peculiarity with respect to the conventional *operator* or *time-step splitting explicit/implicit* schemes.

The development of this new explicit, time-scale splitting algorithm is especially done in view of future applications to systems of stiff PDEs, the stiffness of which is mainly related to the presence of a source term. Although standard time-step splitting approaches provide a consistent way of treating the nonlinear coupling between the spatial operator and the source term, they do not provide (and therefore cannot explicitly take advantage of) any information on how the slow spatial scales interact with the fast and slow time scales originated by the source term [36]. In contrast, the proposed time-scale splitting allows coupling the slow spatial scales directly to the slow time scales originated by the source term, when the fast time-scales are related to the source term only. Such a treatment of a stiff problem by an explicit algorithm eliminates the need for implicit or multistep schemes. As a result, the solution of nonlinear systems at each integration step and the extra storage required are avoided. The time scale splitting also provides an estimate of the order of magnitude of the dominant time scale, which can be used both to adjust (maximize) the integration step for time marching and to set the proper spatial discretization (grid resolution) in a PDE problem without resorting to an error control strategy [36].

Although the new time-scale splitting algorithm will be ultimately devoted to solve stiff systems of PDEs, the presentation of its specific features and the discussion of its performance can be made much simpler when it is employed for the solution of a stiff system of ODEs. As a test problem, we selected the auto-ignition process occurring behind a normal, steady shock wave. The combustible mixture considered is methane/air whose detailed kinetic mechanism involves 49 species and 260 reactions. The performance of the new algorithm—in terms of accuracy and computational efficiency—will be compared with that provided by the very well-known and used LSODE package. The numerical results presented are therefore devoted to demonstrate how and under what circumstances the new algorithm works and delivers a satisfactory performance. An attempt to compare the performance of the new scheme to that deliverable by LSODE when dealing with problems involving PDEs will also be presented.

The structure of the manuscript is as follows. First, a brief outline of the CSP method will be presented on the basis of which the explicit, time-scale splitting algorithm will be developed. Next, the governing equations for the physical problem under examination will be stated. Finally, the performance of the new explicit algorithm will be reported and compared with that of the LSODE package.

2. IMPLICIT VS EXPLICIT SCHEMES FOR STIFF PROBLEMS

Consider the nonlinear initial value problem

$$\frac{d\boldsymbol{y}}{dt} = \boldsymbol{g}(y), \quad \boldsymbol{y}(0) = \boldsymbol{y}_0, \tag{1}$$

where $\boldsymbol{y}$ and $\boldsymbol{g}$ are N-dimensional (column) vectors. Suppose that, throughout the time domain of interest, the Jacobian matrix $J^i_j = dg^i/dy_j$ has M eigenvalues (not necessarily real):

- whose magnitude is much larger than the remaining $N - M$;
- which have negative real parts;
- which are located away from the imaginary axis;
- which are ordered according to their magnitude:

$$|\lambda_1| > \cdots > |\lambda_M| \gg |\lambda_{M+1}| > \cdots > |\lambda_N|. \tag{2}$$

If the time domain of interest is of the order of the reciprocal of the $(M + 1)$-th eigenvalue, then Eq. (1) exhibits a boundary-layer type of stiffness.

To illustrate, by a simple example, how and why an implicit scheme is successful in handling this type of stiff problem, whereas a conventional explicit scheme is bound to fail, assume that the source term $\boldsymbol{g}$ of Eq. (1) is linear. Thus, Eq. (1) reads as

$$\frac{d\boldsymbol{y}}{dt} = \boldsymbol{g} = -A\boldsymbol{y} \quad \boldsymbol{y}(0) = \boldsymbol{y}_o, \tag{3}$$

where A is an $(N \times N)$ constant matrix fully populated. For simplicity, it is assumed that all its eigenvalues are distinct, real, positive, and ordered as in Eq. (2). The exact solution of the linear problem is

$$\begin{aligned}\boldsymbol{y}(t) = {} & \left(\boldsymbol{a}_1 \mathrm{e}^{-\lambda_1 t}\boldsymbol{b}^1 + \cdots + \boldsymbol{a}_M \mathrm{e}^{-\lambda_M t}\boldsymbol{b}^M\right)\boldsymbol{y}_o \\ & + \left(\boldsymbol{a}_{M+1}\mathrm{e}^{-\lambda_{M+1} t}\boldsymbol{b}^{M+1} + \cdots + \boldsymbol{a}_N \mathrm{e}^{-\lambda_N t}\boldsymbol{b}^N\right)\boldsymbol{y}_o,\end{aligned} \tag{4}$$

where the column vector $\boldsymbol{a}_i$ and the row vector $\boldsymbol{b}^i$ $(i = 1, N)$ are the right and left eigenvectors of A, respectively, and the two terms on the RHS of Eq. (4) correspond to modes below and above the driving time scale Δt defined by the relation

$$\Delta t \approx O(\lambda_{M+1})^{-1} \gg O(\lambda_1)^{-1}. \tag{5}$$

Now, consider the numerical solution obtained by approximating Eq. (3) first by means of a first-order backward explicit scheme and then by a first-order backward implicit scheme. It can be shown that the explicit scheme yields the solution

$$\begin{aligned}\boldsymbol{y}_{ex}(t) = {} & \left\{\boldsymbol{a}_1 \mathrm{e}^{\frac{\ln|1-\lambda_1\Delta t|}{\Delta t}\, t}\mathrm{e}^{\frac{i\pi t}{\Delta t}}\boldsymbol{b}^1 + \cdots + \boldsymbol{a}_M \mathrm{e}^{\frac{\ln|1-\lambda_M\Delta t|}{\Delta t}\, t}\mathrm{e}^{\frac{i\pi t}{\Delta t}}\boldsymbol{b}^M\right\}\boldsymbol{y}_o \\ & + \left\{\boldsymbol{a}_{M+1}\mathrm{e}^{\frac{\ln(1-\lambda_{M+1}\Delta t)}{\Delta t}\, t}\boldsymbol{b}^{M+1} + \cdots + \boldsymbol{a}_N \mathrm{e}^{\frac{\ln(1-\lambda_N\Delta t)}{\Delta t}\, t}\boldsymbol{b}^N\right\}\boldsymbol{y}_o,\end{aligned} \tag{6}$$

whereas the implicit scheme yields

$$\boldsymbol{y}_{im}(t) = \left\{\boldsymbol{a}_1 \mathrm{e}^{\frac{-\ln(1+\lambda_1 \Delta t)}{\Delta t}\, t} \boldsymbol{b}^1 + \cdots + \boldsymbol{a}_M \mathrm{e}^{\frac{-\ln(1+\lambda_M \Delta t)}{\Delta t}\, t} \boldsymbol{b}^M \right\} \boldsymbol{y}_o + \left\{\boldsymbol{a}_{M+1} \mathrm{e}^{\frac{-\ln(1+\lambda_{M+1} \Delta t)}{\Delta t}\, t} \boldsymbol{b}^{M+1} + \cdots + \boldsymbol{a}_N \mathrm{e}^{\frac{-\ln(1+\lambda_N \Delta t)}{\Delta t}\, t} \boldsymbol{b}^N \right\} \boldsymbol{y}_o. \tag{7}$$

An estimate of the stability interval width required by an explicit scheme to solve Eq. (1) over a time interval of the order of the reciprocal of the $(M+1)$-th eigenvalue is given by the product

$$z = |\lambda_1| \Delta t \approx |\lambda_1| / |\lambda_{M+1}|. \tag{8}$$

However, the simple first-order backward explicit scheme can only afford to integrate Eq. (1) with $z \leq 1$, which yields $\Delta t \leq (|\lambda_1|)^{-1}$; otherwise the first M components of Eq. (6) will become unstable (they will grow exponentially). In a stiff problem, $|\lambda_{M+1}|$ can be several orders of magnitude smaller than $|\lambda_1|$, thus justifying the quest for explicit schemes with an extraordinary large stability limit, that is with very large z values.

In contrast, the first M components of Eq. (7) are always stable (they decay exponentially), since the first-order backward implicit scheme is A-stable. The remaining $N - M$ terms in both Eqs. (6) and (7) are stable and predict with the same accuracy the similar components of the exact solution (4).

This simple example shows that if the selection of integration step is based on the local characteristic time scale, individuated by the inverse of $|\lambda_{M+1}|$, the more efficient explicit scheme causes an exponential growth of the solution components related to the fast time scales.

However, at the time period of interest, these components, being exponentially small, do not contribute to the overall solution of the original problem. Therefore, it is tempting to devise a strategy aimed at making the fast time-scales disappear from the problem when they become exhausted, and the evolution of the system depends on the slow modes only. This way, the numerical solution acquired with an explicit scheme might be made more efficient than if an implicit scheme was employed.

The next sections will illustrate how the computational singular perturbation method allows one to pursue this strategy for nonlinear problems as well.

3. STIFF PROBLEMS BY THE CSP METHOD

The CSP method is based on the ability to split the N-dimensional domain of $\boldsymbol{y}$ in two subdomains, each of which exhibits certain characteristics. One subdomain is M-dimensional, contains the fast time scales, and is responsible for the rapid changes the solution might exhibit. The other subdomain is $N - M$ dimensional, contains the slow time scales, and is responsible for the smooth behavior of the solution. When $\boldsymbol{y}$ goes through a period of rapid changes (inner region or boundary layer), the component of the velocity vector $\boldsymbol{g}$ in the fast subdomain is significant. However, it becomes negligible when $\boldsymbol{y}$ exhibits a smooth behavior (outer region).

As the system evolves in time, the two subdomains "rotate." CSP follows the movement of the two subdomains and inspects the projection of $\boldsymbol{g}$ into the fast subdomain. When the trajectory leaves the inner region, this projection becomes exponentially small. CSP then provides a simplified system of equations which produces an approximation of the "exact" solution but contains no fast time scales. This way, the fast time scales, which cause the numerical difficulties, are retained only when needed and are discarded when they have no effect on the evolution of the system. This process is done in such a way that the computed solution stays within the desired accuracy.

To split the source term $\boldsymbol{g}$ into a fast and a slow component, the $\boldsymbol{y}$-domain must be resolved in an appropriate manner. Let R^N be the domain of $\boldsymbol{y}$ and $[\boldsymbol{a}_1(t), \ldots, \boldsymbol{a}_N(t)]$ be a set of column basis vectors which span R^N at time t. The corresponding set of orthogonal row vectors is denoted as $[\boldsymbol{b}^1(t), \ldots, \boldsymbol{b}^N(t)]$. We assume that the first M basis vectors span the fast subdomain. The vector $\boldsymbol{g}$ can now be expanded in terms of these sets of basis vectors as

$$\frac{d\boldsymbol{y}}{dt} = \sum_{i=1}^{N} \boldsymbol{a}_i f^i = \sum_{r=1}^{M} \boldsymbol{a}_r f^r + \sum_{s=M+1}^{N} \boldsymbol{a}_s f^s, \tag{9}$$

where $f^i = \boldsymbol{b}^i \cdot \boldsymbol{g}$ is the "amplitude" of $\boldsymbol{g}$ in the "direction" of the basis vector $\boldsymbol{a}_i$, and the indices r and s will denote $r$apid and $s$low modes, respectively. The projection of $\boldsymbol{g}$ onto the fast subdomain being small can be expressed by the M equations (of partial equilibrium state relations)

$$f^r \approx 0 \quad r = 1, M. \tag{10}$$

The M equations (10) describe the manifold in the space of $\boldsymbol{y}$ on which the trajectory in the outer region moves according to the equation

$$\frac{d\boldsymbol{y}}{dt} \approx \sum_{s=M+1}^{N} \boldsymbol{a}_s f^s. \tag{11}$$

In this simplified form, the fast modes are absent so that only the slow time-scales are present. As a result, the integration with an explicit scheme can advance with larger integration steps. In order for the integration to be stable, this integration step must be of the order of the reciprocal of the magnitude of the largest of the $N - M$ slow eigenvalues of J (Eq. (5)). More details about the CSP method can be found in the Refs. [37–39].

4. STEPS TOWARD THE TIME-SCALE SPLITTING ALGORITHM

There are three basic steps to build the new time-scale splitting, explicit algorithm: (i) identification of the number M of exhausted modes at a given time, (ii) construction of the CSP basis vectors and (iii) integration of the stiff ODE system according to a time-scale explicit algorithm.

4.1. Detection of the Exhausted Fast Modes

The criterion is the following. Let us first introduce an error vector $\boldsymbol{y}_{\text{error}}$ built on the basis of the solution vector $\boldsymbol{y}$, as

$$y^i_{\text{error}} = \epsilon^i_{rel}|y^i| + \epsilon^i_{abs}, \quad i = \Delta, N \tag{12}$$

where ϵ_{rel}^{i} and ϵ_{abs}^{i} are the maximum relative and absolute errors on the i-th variable, respectively. The number M of fast modes which, within the limits of accuracy specified by the given error vector, are considered exhausted is defined as the largest integer lying between 1 and N which satisfies the inequality

$$\left|\tau(M+1)\sum_{j=1}^{M}\boldsymbol{a}_j f^j\right| < \boldsymbol{y}_{\text{error}}, \tag{13}$$

where $\tau(M+1)$, the time scale related to the $(M+1)$-th CSP mode, is defined as

$$\tau(M+1) = \left|\frac{1}{\lambda(M+1)}\right|. \tag{14}$$

The inequality (13) guarantees that the trajectory remains close to the manifold within specified bounds and is not diverted far from it by marching in time according to the simplified nonstiff Eq. (11). In fact, the omission of the M fast modes in the integration along one time step introduces a local error which can be estimated as

$$E_r = O\left(\Delta t\sum_{j=1}^{M}\boldsymbol{a}_j f^j\right). \tag{15}$$

Given the condition (13), it follows that

$$E_r = O\left(\frac{\Delta t\ y_{err}^{i}}{\tau(M+1)}\right). \tag{16}$$

4.2. Construction of the Basis Vectors

The algorithm for the construction of the basis vectors $\boldsymbol{a}_r$ and $\boldsymbol{b}^r$ $(r = 1, M)$, which are desired to define the fast subdomain in the problem, is provided by the recursive formulas reported here below from Refs. [26–28],

$$A(s_1+1) = \left[-\frac{dA(s_1)}{dt} + JA(s_1)\right]\left\{B\left[-\frac{dA(s_1)}{dt} + JA(s_1)\right]\right\}^{-1} \quad B, J = \text{ const} \tag{17}$$

$$B(s_2+1) = \left\{\left[\frac{dB(s_2)}{dt} + B(s_2)J\right]A\right\}^{-1}\left[\frac{dB(s_2)}{dt} + B(s_2)J\right] \quad A, J = \text{ const}, \tag{18}$$

where the matrices A and B collect the right and left M fast basis vectors, respectively,

$$A = (\boldsymbol{a}_1 \quad \cdots \quad \boldsymbol{a}_M); \quad B = (\boldsymbol{b}^1 \quad \cdots \quad \boldsymbol{b}^M)^T$$

and the initial guesses $A(0)$ and $B(0)$ are, in principle, arbitrary matrices. The refinements (17) and (18) are not coupled to each other; that is, the number of s_1-refinements might not be equal to the number of s_2-refinements. Independent of the number of s_1- and s_2-refinements, the resulting vectors produce an orthonormal basis.

The larger the number of the s_1-refinements (Eq. 17) and s_2-refinements (Eq. 18) the more accurate the approximation of the fast subdomain by the basis vectors $\boldsymbol{a}_r$ and $\boldsymbol{b}^r$ $(r = 1, M)$.

In fact, the $s_1(s_2)$-refinements tend to purify the slow (fast) part in the RHS of Eq. (9) from the fast (slow) time scales. Each of the s_1, s_2-refinements becomes more effective as the time-scale separation, defined as

$$\varepsilon = \left| \frac{\lambda(M+1)}{\lambda(M)} \right|, \tag{19}$$

becomes more wide.

Regarding the simplified problem, Eq. (11), the s_1-refinements tend to improve its stability while the s_2-refinements tend to improve the accuracy of the approximate solution by allowing the exponential decay of the fast amplitudes f^r $(r = 1, M)$ to smaller values.

The presence of the time derivatives in Eqs. (17 and 18) signifies the fact that the fast subdomain rotates with time. It is only when the problem is linear, i.e., $\boldsymbol{g} = A\boldsymbol{y}$, that the fast subdomain is fixed with time, being defined by the M eigenvectors of the constant matrix A, which correspond to its M largest eigenvalues. In such a case, when trying to approximate the fast subdomain the time derivative terms can be omitted from Eqs. (17) and (18) without any loss in stability or accuracy, both being improved with the number of the s_1 and s_2-refinements. The simplified recursive formulas constitute then the block-power method for approximating the space spanned by the M fast eigenvectors [37–39, 44].

When the problem is nonlinear, the rotation of the fast subdomain with time must be taken into account, by including the time derivative terms. In the outer region (i.e., away from the fast transients), Eqs. (17) and (18) indicate that these terms are of higher order with respect to JA and BJ. Therefore, in a nonlinear problem their omission will provide leading order accuracy only. Additional refinements (to the first) will not produce any improvement.

However, following the discussion of the linear case, when Eq. (1) is weakly nonlinear (as in the induction and recombination regions in a flame), it is possible to improve the stability and the accuracy of the simplified Eq. (11) by additional refinements.

In the new explicit scheme the basis vectors are computed by neglecting the time derivative terms but by allowing more than one refinement to be executed. As was noted previously, this assumption reduces the refinements (17–18) to the block-power iterations. With such a strategy, the required extra storage and calculations are avoided, resulting in a more efficient scheme. However, from the discussion above it is clear that this improvement is accompanied by the shortcoming of possibly not being able to fully identify the fast subdomain, that is, the maximum possible number of exhausted modes. As a result, the integration might proceed with a time step smaller than the locally characteristic scale of the problem.

The execution of the refinements (17–18) can be combined with the identification of the exhausted modes. The related algorithm is presented next, by considering the case where all eigenvalues of J are real. The generalized version dealing with complex eigenvalues is a straightforward extension [45]. According to this algorithm, the block-power method for computing the basis vectors, which approximates the fast subdomain, is replaced by the simple power method, according to which the M fast eigenvectors are computed one at a time, starting with the ones that correspond to the fastest time-scale [45]. At the first time step, the basis vectors $\boldsymbol{a}_r$ and $\boldsymbol{b}^r$ are arbitrary, while at subsequent steps the vectors computed at the start of the previous integration step can be used as an initial guess. With this algorithm, good estimates of the M fast time-scales $(r = 1, M)$ and of the fastest of the slow scales are obtained as well.

The algorithm is as follows:

ALGORITHM 1 (Find CSP Basis Vectors and Time Scales)

```
Begin
    Initial settings:
    $\Delta \boldsymbol{y} = \mathbf{0}$, $\mathbf{J_L^{m=1}} = \mathbf{J_R^{m=1}} = \mathbf{J} = \mathbf{grad}(\boldsymbol{g})$, $\mathbf{P^{(m=0)}} = \mathbf{I}$
    Loop on $m = 1, N$
      Loop on $S$                                   to converge over the value of $\tau_m^m$
        Loop on $s_1$                               to improve the stability of the integration
            Refine vectors $\boldsymbol{a}_r$
              $\boldsymbol{q}_m = J_L^{(m)} \boldsymbol{a}_m^{(s_1)}$
              $(\tau_m^m)_L = (\boldsymbol{b}^m \cdot \boldsymbol{q}_m)^{-1}$
              $\boldsymbol{a}_m^{(s_1+1)} = \boldsymbol{q}_m (\tau_m^m)_L$
        End loop on $s_1$
        Loop on $s_2$                               to improve the accuracy of the integration
            Refine $\boldsymbol{b}^r$ vectors
              $\boldsymbol{q}^m = \boldsymbol{b}^m_{(s_2)} J_R^{(m)}$
              $(\tau_m^m)_R = (\boldsymbol{q}^m \cdot \boldsymbol{a}_m)^{-1}$
              $\boldsymbol{b}^m_{(s_2+1)} = (\tau_m^m)_R \boldsymbol{q}^m$
        End loop on $s_2$
        If[$(\tau_m^m)_R$ has converged] then go to B
      End loop on $S$
      Label (B)
        $\tau(m) = (\tau_m^m)_R$
      Enforce the criterion to declare a mode exhausted
        If[$m \geq 2$ and $|\tau(m)\Delta \boldsymbol{y}| \geq \boldsymbol{y}_{error}$] then go to A
      Update the projection matrix
        $P^{(m)} = P^{(m-1)} - \boldsymbol{a}_m \boldsymbol{b}^m$                      (no sum on m)
      Update the fast mode contribution
        $\Delta \boldsymbol{y} = \Delta \boldsymbol{y} - \boldsymbol{a}_m f^m$                         (no sum on m)
        Note: $f^m$ is computed as follows:
            $if\ m > M_{old}\ then\ f^m = \boldsymbol{b}^m \cdot \boldsymbol{g}$
            or:
            $if\ m \leq M_{old}\ then\ f^m = f_\infty^m := -\tau(m) \boldsymbol{b}^m J_R^{(m)} \boldsymbol{g}$
      Deflate the last mode found from the Jacobian
        $J_L^{(m+1)} = P^{(m)} J$
        $J_R^{(m+1)} = J P^{(m)}$
    End loop on m
    Label (A)
    $M = m - 2$                                                Number of exhausted modes
    $\tau(M+1) = \tau(m-1)$                                    Driving time-scale
    $P^{(M)} = P^{(M+1)} + \boldsymbol{a}_{(M+1)} \boldsymbol{b}^{(M+1)}$       Correct projection matrix
End.
```

The rate of convergence of the S-iterations increases with the ratio between the time-scales of two consecutive modes. Thus, if two time-scales are close to one another, the convergence of the related two basis vectors may become extremely slow. This circumstance does not affect the accuracy of the scheme, since poorly approximated basis vectors cannot identify the exhausted modes. Therefore, no simplification with respect to the related amplitudes which are still present in Eq. (11), is allowed. In addition, there is no diminishing in the efficiency of the scheme, since the appearance of two time scales close to one another indicates that the corresponding two amplitudes decay with a similar rate. As a result, no real gain from increasing the time step will be achieved if the first of the two amplitudes is identified as exhausted by allowing a large number of refinements. This justifies the enforcement of a maximum number of S-iterations, say 10, independent of the achieved convergence of $\tau(m)$. In any case, the basis keeps being improved as the integration progresses in time.

As noted, the effect of the inner s_1- and s_2-refinements is to stabilize and improve the estimate of the basis vectors. It has been observed that the algorithm improves its overall performance if at least three iterations are allowed for both the $\boldsymbol{a}_r$ and $\boldsymbol{b}^r$ basis. The extra cost for the iterations is largely compensated by the more stable evolution of the number of exhausted modes.

To improve the stability of the calculation, care must be taken in the evaluation of the amplitude f^m of an exhausted mode, if computed as the projection of $\boldsymbol{g}$ on the fast directions ($f^m = \boldsymbol{b}^m \cdot \boldsymbol{g}$). In fact, the amplitude becomes small because competing effects of usually large size cancel each other. Under these circumstances, the machine precision might become insufficient to guarantee an accurate evaluation of the amplitude f^m. A better estimate is found by replacing f^m with its asymptotic value f^m_∞, defined later in Eq. (25). In contrast, the amplitude of a slow mode involves no significant cancellations and can therefore be accurately computed by projecting the source term onto the slow directions. This discussion motivates the strategy adopted in the algorithm to compute f^m.

4.3. Integration of the Stiff ODE System

Let us assume that, at a certain stage of the process evolution, the trajectory has exited the inner region and the first M column vector $\boldsymbol{a}_r$ spans the fast subdomain of $\boldsymbol{y}$, while the remaining $N - M$ column vectors $\boldsymbol{a}_s$ span the slow subdomain of $\boldsymbol{y}$. With these definitions, the original ODE system (9) can also be written as

$$\frac{d\boldsymbol{y}}{dt} = \sum_{r=1}^{M} \boldsymbol{a}_r f^r + P\boldsymbol{g}, \tag{20}$$

where the projection matrix P is defined as

$$P = \mathrm{I} - \sum_{r=1}^{M} \boldsymbol{a}_r f^r \tag{21}$$

maps $\boldsymbol{g}$ onto the slow subdomain of $\boldsymbol{y}$. The time change of $\boldsymbol{y}$ is obtained by integrating system (20) over an interval of time Δt, according to the expression

$$\boldsymbol{y}(T + \Delta t) - \boldsymbol{y}(T) = \int_T^{T+\Delta t} \sum_{r=1}^{M} \boldsymbol{a}_r f^r \, dt + \int_T^{T+\Delta t} \mathrm{P}\boldsymbol{g} \, dt. \tag{22}$$

As demonstrated in Refs. [26–28], the amplitudes of the fast modes f^r evolve in time according to the equations

$$\frac{df^r}{dt} = \sum_{r'=1}^{M} \lambda^r_{r'} \left[f^{r'} - f^{r'}_{\infty} \right] \quad r = 1, M, \tag{23}$$

where

$$\lambda^i_j = \left(\frac{d\boldsymbol{b}^i}{dt} + \boldsymbol{b}^i J \right) \boldsymbol{a}_j. \tag{24}$$

Given that the trajectory is in the outer (slow) region, $f^{r'}$ reaches the asymptotically small value $f^{r'}_{\infty}$ which expresses the contribution of the slow scales to the amplitude of the fast modes (“mode mixing”) and is defined as

$$f^{r'}_{\infty} = -\sum_{r=1}^{M} \tau^{r'}_{r} \sum_{s=M+1}^{N} \lambda^r_s f^s, \tag{25}$$

and the matrix $\tau^{r'}_{r}$ is defined as the inverse of $\lambda^r_{r'}$.

Solving Eq. (23) for f^r and substituting it into the integral (22), yields

$$\boldsymbol{y}(T + \Delta t) = \boldsymbol{y}(T) + \int_T^{T+\Delta t} \sum_{r,r'=1}^{M} \boldsymbol{a}_r \tau^r_{r'} \frac{df^{r'}}{dt}\, dt + \int_T^{T+\Delta t} \sum_{r=1}^{M} \boldsymbol{a}_r f^r_{\infty}\, dt + \int_T^{T+\Delta t} \mathrm{P}\boldsymbol{g}\, dt. \tag{26}$$

Integrating by parts and using Eq. (25), yields

$$\begin{aligned} \boldsymbol{y}(T + \Delta t) = \boldsymbol{y}(T) &+ \sum_{r,r'=1}^{M} \boldsymbol{a}_r \tau^r_{r'} f^{r'}|_{t=T+\Delta t} - \sum_{r,r'=1}^{M} \boldsymbol{a}_r \tau^r_{r'} f^{r'}|_{t=T} \\ &- \int_T^{T+\Delta t} \sum_{r,r'=1}^{M} \frac{d}{dt}\left[\boldsymbol{a}_r \tau^r_{r'}\right] f^{r'}\, dt + \int_T^{T+\Delta t} \sum_{r=1}^{M} \boldsymbol{a}_r f^r_{\infty}\, dt + \int_T^{T+\Delta t} \mathrm{P}\boldsymbol{g}\, dt. \end{aligned} \tag{27}$$

In the outer region, where the characteristic time-scale is $O(\tau(M+1))$ and $\tau^r_{r'} \approx O(\tau(M))$, one can derive the estimates

$$\frac{d}{dt}\left[\boldsymbol{a}_r \tau^r_{r'}\right] \approx O(\varepsilon \boldsymbol{a}_r) \tag{28}$$

and

$$\tau^r_{r'} \lambda^{r'}_s \approx O(\varepsilon^{s_2}) \tag{29}$$

from being $\lambda^{r'}_s \approx O(\varepsilon^{s_2-1})$, which allows us to consider the first two integrals on the RHS of Eq. (27) small. Moreover, the quantity $\sum_{r,r'=1}^{M} \boldsymbol{a}_r \tau^r_{r'} f^{r'}$ can be assumed much smaller at $t = T + \Delta t$ than at $t = T$, because of the exponential decay of the mode amplitude $f^{r'}$ along $\Delta t = O(\tau(M+1))$ with a rate of $\tau(M)$. Therefore, Eq. (27) can be written as

$$\boldsymbol{y}(T + \Delta t) = \boldsymbol{y}(T) - \sum_{r,r'=1}^{M} \boldsymbol{a}_r \tau^r_{r'}|_{t=T} f^{r'}(T) + \int_T^{T+\Delta t} \mathrm{P}\boldsymbol{g}\, dt + Small, \tag{30}$$

where

$$Small = \sum_{r,r'=1}^{M} \boldsymbol{a}_r \tau^r_{r'}|_{t=T+\Delta t}\, f^{r'} + \int_T^{T+\Delta t} \sum_{r=1}^{M} \boldsymbol{a}_r\, f^r_\infty\, dt - \int_T^{T+\Delta t} \sum_{r,r'=1}^{M} \frac{d}{dt}\left[\boldsymbol{a}_r \tau^r_{r'}\right] f^{r'}\, dt \tag{31}$$

collects all small contributions. Eq. (30), with the $Small$ terms neglected, is in a form suited to write a time-scale splitting explicit scheme of integration, consisting of two steps. First, the contribution of the slow modes to the time change of $\boldsymbol{y}$ is computed by the expression

$$\tilde{\boldsymbol{y}}(T + \Delta t) = \boldsymbol{y}(T) + \int_T^{T+\Delta t} \mathrm{P}(T)\boldsymbol{g}(t)\, dt, \tag{32}$$

where $P(T)$ is assumed constant between T and $T + \Delta t$. Then, following Eq. (30), the correction that results from the omission of the fast modes from the integration in Eq. (32) is accounted by the expression

$$\boldsymbol{y}(T + \Delta t) = \tilde{\boldsymbol{y}}(T + \Delta t) - \sum_{r,r'=1}^{M} \boldsymbol{a}_r \tau^r_{r'}|_{t=T}\, \hat{f}^{r'}. \tag{33}$$

The second term at the RHS of Eq. (33) is referred to as “radical correction” in Ref. [26]:

$$\Delta \boldsymbol{y}_{fast} = \sum_{r,r'=1}^{M} \boldsymbol{a}_r \tau^r_{r'}|_{t=T}\, \hat{f}^{r'}. \tag{34}$$

However, the off-diagonal terms of the matrix $\tau^r_{r'}$ can be neglected in Eq. (34), so that $\tau^r_{r'}$ can be estimated as

$$\tau^r_{r'} \approx \delta^r_{r'} \tau(r), \tag{35}$$

where the local characteristic time-scales $\tau(r)$ are found by Algorithm 1, and $\delta^r_{r'}$ is the Kronecker symbol. The off-diagonal terms can be neglected on the grounds that the basis vectors are usually a small perturbation of the eigenvectors of J. This makes $\lambda^r_{r'}$, introduced in Eq. (23), and its inverse $\tau^r_{r'}$ diagonally dominant matrices. This approximation allows us to reduce both computing time and storage requirements.

Numerical tests have clearly shown that to make the algorithm stable, it is mandatory to evaluate the amplitude of the fast modes $\hat{f}^r$ according to the expression

$$\hat{f}^r = \boldsymbol{b}^r(T) \cdot \boldsymbol{g}[\tilde{\boldsymbol{y}}(T + \Delta t)], \tag{36}$$

which requires the computation of the source term $\boldsymbol{g}$ at the state value $\tilde{\boldsymbol{y}}(T + \Delta t)$ produced by Eq. (32) and the projection of it onto the basis vector $\boldsymbol{b}^r(T)$ found at the beginning of the integration step.

Eq. (33) introduces a correction which is within the bounds imposed for the desired accuracy (Eq. (13)) and is related to Eq. (10), the latter describing the manifold on which the trajectory—moving at the pace set by the slow scales—is confined by the action of the fast time-scales.

In particular, the simplifications implied by Eq. (32) move the trajectory slightly off this manifold. Eq. (33) corrects this deviation at the end of the time step, by bringing the trajectory back on the manifold. As Eq. (33) shows, this correction takes place along the fast directions only. It is easy to show that Eq. (33) is equivalent to one Newton iteration of Eq. (10).

The time-scale splitting, represented by Eqs. (32) and (33), becomes more and more accurate as the time-scale separation ε, defined in Eq. (19), decreases.

4.4. Choice of the Integration Step

The correct order of magnitude of the time-scale over which the slow contribution evolves is provided by the inverse of $|\lambda(M+1)|$. Therefore, the actual integration step of integration Δt is evaluated as follows. The linear stability analysis of the time-scale splitting explicit scheme prescribes

$$|\lambda(M+1)|\Delta t < 1. \tag{37}$$

Therefore, a tentative integration step $\widetilde{\Delta t}$ is defined as

$$\widetilde{\Delta t} = \alpha(|\lambda(M+1)|)^{-1} = \alpha|\tau(M+1)| \quad \alpha < 1, \tag{38}$$

where $\alpha = \Delta t/|\tau(M+1)|$ is a safety coefficient against nonlinear effects. As a consequence, the error that results from the omission of the fast modes—defined in Eq. (16)—can be estimated as

$$E_r = O\left(\alpha\, y^i_{err}\right). \tag{39}$$

Next, the guessed $\widetilde{\Delta t}$ is compared to the integration step Δt_{old} used at the previous integration step to produce the new integration step according to the following heuristic strategy

$$\text{if}(\widetilde{\Delta t} > \Delta t_{\text{old}}) \text{ then } \Delta t = \min(1.5\Delta t_{\text{old}}, \widetilde{\Delta t}) \tag{40}$$

$$\text{if}(\widetilde{\Delta t} < \Delta t_{\text{old}}) \text{ then } \Delta t = \widetilde{\Delta t}. \tag{41}$$

The rationale behind these formulas is the following. When at a certain point in time it is found that a number of fast time-scales become exhausted, the integration can proceed with the corresponding fast components of the source term removed. As a result, the local characteristic time scale (driving scale) Δt increases. In this case, the stability condition of the explicit integration scheme makes it possible to take an integration step Δt larger than the one adopted at the previous integration step (Δt_{old}). In the explicit time-scale splitting algorithm presented here, this integration step change follows a gradual increase as stated by the condition (40). In contrast, when at the end of an integration step it is found that a number of fast modes are reactivated, that means that the solution accuracy has been degraded and that the integration step has to be repeated ("resetting" procedure) taking into account the modes previously removed. Consequently, the local characteristic time-scale decreases, being the fastest scale of the restored modes. In this case, the integration step must be decreased instantaneously in order to comply with the stability limits. This motivates the adoption of formula (41).

The local error of an explicit scheme is in general of the order of $(\Delta t/\tau)^n f(\mathbf{y})$, where $n-1$ is the order of the method, τ is the fastest time-scale in the problem, and the function $f(\mathbf{y})$ mainly depends on the particular scheme employed and the degree of the nonlinearity of the source term (Ref. [1]). It follows that the selection of the time step according to Eq. (38) introduces a local error in the explicit integration of the slow part, Eq. (32), which can be estimated as

$$E_s^i = O(\alpha^n f^i(\mathbf{y})). \tag{42}$$

4.5. Evaluation of the Jacobian When the Mixture Contains Species in Low Concentration

The construction of the simplified nonstiff Eq. (11) is based on the CSP splitting between the fast and slow subdomains. To perform the splitting correctly, a reliable Jacobian (i.e., derivatives of the source term with respect to the unknowns) to be fed into Eqs. (17) and (18) is needed. The Jacobian can be evaluated either analytically or numerically by finite differences. In this work, we followed the second option.

Unfortunately, the Jacobian becomes ill-conditioned when the process develops very small values for some unknowns (smaller than 10^{-13} if double precision is used) while other unknowns—such as pressure or velocity—have order of magnitude one. Care in the evaluation of the finite differences involved in the Jacobian might alleviate the problem, but not solve it altogether.

The basis vectors built upon such a Jacobian are not accurate enough and might take the calculation to a stop, because some of the unknowns (species in low concentration) become negative. Species in low concentration cannot simply be removed once and for all from the calculation, since they might not remain small at later times. For example, during most of the induction period, the intermediate species have very low concentrations, whereas their concentration abruptly increases shortly ahead the reaction zone.

Therefore, the problem of the ill-conditioned Jacobian because of species in low concentration has been circumvented as follows: if (i) one species, say y_i, has a low concentration, and (ii) its corresponding source term g^i_{slow} in Eq. (11) is negative—meaning that y_i is attaining even lower values—then the set of ODE is modified so as to force y_i to remain constant at its last positive value. In the CSP terminology, we force the mode associated to the i-th species to become a "dormant" [26] mode. This is obtained as follows:

1. the source term component g^i_{slow} and the fast mode correction element Δy^i_{fast} in Eq. (34) are both set equal to zero;

2. the off-diagonal elements of the Jacobian in the i-th row and i-th column are also set equal to zero (that is, $J^i_j = J^j_i = 0$, $j = 1, N$, $i \neq j$), and a small value is attributed to the diagonal element (that is, $J^i_i = 10^{-20}$). By construction, the diagonal element J^i_i becomes the eigenvalue of the mode driving the small species y^i. This forces the species to change very slowly in time and, in the limit, to remain constant as desired.

5. THE TIME-SCALE SPLITTING EXPLICIT ALGORITHM

The time-scale splitting algorithm is organized in two substeps according to the time-scale splitting between fast and slow modes presented by Eqs. (32) and (33). Starting from a known value of $\boldsymbol{y}$ at time T, the algorithm proceeds to the next time $T + \Delta t$ as explained below.

ALGORITHM 2 (The Time-Scale Splitting Explicit Algorithm)

Begin

Step I: Slow Time Scales Contribution

1. Evaluate the source term $\boldsymbol{g}(T) = \boldsymbol{g}[\boldsymbol{y}(T)]$
2. Find the number $M[\boldsymbol{y}(T)]$ of "exhausted" fast modes, the local characteristic time scales $\tau(r)[\boldsymbol{y}(T)]$ ($r = 1, M + 1$), and the corresponding basis vectors spanning the fast subdomain $\boldsymbol{a}_r[\boldsymbol{y}(T)], \boldsymbol{b}^r[\boldsymbol{y}(T)]$, with $r = 1, M[\boldsymbol{y}(T)]$;
3. Build the projection matrix $P[\boldsymbol{y}(T)] = I - \sum_{r=1}^{M[\boldsymbol{y}(T)]} \boldsymbol{a}_r[\boldsymbol{y}(T)]\boldsymbol{b}^r[\boldsymbol{y}(T)]$;
4. Integrate numerically between time T and $T + \Delta t$ the nonstiff differential equation (32)

$$\frac{d\boldsymbol{y}}{dt} = P[\boldsymbol{y}(T)]\boldsymbol{g}[\boldsymbol{y}(t)] \tag{43}$$

from the initial condition $\boldsymbol{y}(T)$ to the final value $\tilde{\boldsymbol{y}}(T + \Delta t)$, by means of any explicit scheme, of any order of accuracy, with an integration step Δt whose order of magnitude is $\tau(M + 1)$. The integration scheme adopted in our work has been the explicit Runge–Kutta, fourth order, 4-stages scheme [1]. The integration step Δt actually used is defined according to the relations (40 and 41). The value $\tilde{\boldsymbol{y}}(T + \Delta t)$ accounts for the contribution of the slow time-scales only and will be corrected by **Step II** if exhausted fast modes are present.

The projection matrix P *is kept constant* along the integration step at the value obtained for the initial $\boldsymbol{y}(T)$. This assumption is sufficiently accurate since the basis vectors evolve over the slow time-scale.

Step II: Fast Time Scales Contribution (undertaken only if $M \neq 0$)

According to Eq. (33), the contribution of the fast modes to leading order accuracy can be evaluated as follows:

1. Compute $\boldsymbol{g}(T + \Delta t) = \boldsymbol{g}[\tilde{\boldsymbol{y}}(T + \Delta t)]$;
2. Evaluate the fast modes' amplitude $f^r(T + \Delta t) = \boldsymbol{b}^r[\boldsymbol{y}(T)] \cdot \boldsymbol{g}(T + \Delta t)$;
3. Correct for the contribution of the fast time-scales:

$$\boldsymbol{y}(T + \Delta t) = \tilde{\boldsymbol{y}}(T + \Delta t) - \Delta \boldsymbol{y}_{\text{fast}}, \tag{44}$$

where $\Delta \boldsymbol{y}_{\text{fast}}$ is evaluated according to the approximation (35) as

$$\Delta \boldsymbol{y}_{\text{fast}} = \sum_{r=1}^{M[\boldsymbol{y}(T)]} \boldsymbol{a}_r[\boldsymbol{y}(T)]\tau(r)[\boldsymbol{y}(T)] f^r[g(T + \Delta t)] \tag{45}$$

End.

At the beginning of the calculation, the physical process under examination evolves according to the fastest scale and thus the number of exhausted modes is zero ($M = 0$).

This will cause the integration step to be scaled according to the largest eigenvalue of the Jacobian of $\boldsymbol{g}$. In this circumstance the algorithm reduces to an explicit scheme to advance Eq. (43) with $P = I$, that is, to advance the original unsplit ODE. As the fast processes become exhausted, M increases and Eq. (43) advances with larger and larger integration steps as determined by Eq. (41).

6. THE TEST PROBLEM

The validation of the new algorithm is carried out on the basis of the auto-ignition of a stoichiometric mixture of methane and air behind a steady, normal shock wave. The flow is assumed to be steady, inviscid, and nonconducting. The kinetics of the methane/air mixture is governed by a detailed mechanism [50] of finite rate chemical kinetics, which involves 260 reactions, 49 species, and 4 elements.

With these assumptions, the relevant flow model equations are the reactive Euler equations which form a set of PDEs. To evaluate the properties of the new algorithm, the reactive Euler equations are reduced to a set of ODEs, by assuming that the flow is one-dimensional, which can be quickly solved by marching in space.

The thermochemical model adopted for the gaseous mixtures and the details of the chemical kinetics model are reported in the Appendix. This flow displays the main features related to the subject under study, but it remains simple enough to allow a thorough analysis of the performance of the new scheme.

The nonequilibrium state of a mixture of N_s ideal gases can be defined by means of two state variables and the mass fraction of each gaseous species. Density, ρ, and enthalpy, h, can be adopted as thermodynamic state variables, being the most natural to express conservation laws. However, the logarithm of pressure $P = \log p$ and the entropy of the mixture s can be also selected (this yields a formulation more suited to describe wave propagation phenomena). In this event, the energy equation is replaced by an equation for entropy transport and production, obtained by manipulating the Gibbs law of thermochemistry. Under these assumptions, the set of modified reactive Euler equations formed by Eq. (A.20), Eq. (A.21), Eq. (A.22), and Eqs. (A.18), derived in the Appendix, can thus be written as

$$\begin{aligned} \frac{DP}{Dt} + \gamma \nabla \cdot \mathbf{q} &= -\gamma \dot{p} \\ \frac{D\mathbf{q}}{Dt} + \frac{a^2}{\gamma} \nabla P &= 0 \\ \frac{Ds}{Dt} &= \dot{s} \\ \frac{Dy_i}{Dt} &= \dot{y}_i, \end{aligned} \tag{46}$$

where $\mathbf{q}$ is the flow velocity vector, and the source terms appearing at the RHS of Eqs. (46) are defined according to the expressions reported in the Appendix as

$$\dot{p} = \sum_{i}^{N_s} \left(\frac{h_i}{c_p T} - \frac{R_i}{R} \right) \dot{y}_i = \sum_{i}^{N_s} \hat{p}^i \dot{y}_i$$

$$\dot{s} = -\frac{1}{T}\sum_{i=1}^{N_s} \mu_i \dot{y}_i \tag{47}$$

$$\dot{y}_i = \frac{\dot{c}_i \mathcal{M}_i}{\rho}.$$

The temperature is found by inverting numerically the caloric equation of state Eq. (A.7).

If the flow is assumed steady and the flow velocity exactly orthogonal to the shock front (and directed along the x-axis), then Eqs. (46) can be further simplified to yield the set of ODEs

$$A\boldsymbol{w}_{,x} = \dot{\boldsymbol{w}} \tag{48}$$

where

$$A = \begin{pmatrix} u & \gamma & 0 & \mathbf{0}^T \\ \frac{a^2}{\gamma} & u & 0 & \mathbf{0}^T \\ 0 & 0 & u & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}u \end{pmatrix} \quad \boldsymbol{w} = \begin{pmatrix} P \\ u \\ s \\ y_i \end{pmatrix} \quad \dot{\boldsymbol{w}} = \begin{pmatrix} -\gamma\dot{p} \\ 0 \\ \dot{s} \\ \dot{y}_i \end{pmatrix}.$$

Eqs. (48) include $N_s + 3$ equations for the $N_s + 3$ unknowns: logarithm of pressure P, flow velocity u, entropy s, and N_s species mass fractions y_i ($i = 1, N_s$).

Therefore, since the methane/air mixture considered involves $N_s = 49$ species, the total dimension of the system is $N = N_s + 3 = 52$. Four elemental conservation laws (for the 4 elements: C, H, O, N) set the number of linearly independent CSP modes equal to $N - E = 52 - 4 = 48$.

By premultiplying Eqs. (48) with the inverse matrix of A, we obtain

$$\boldsymbol{w}_{,x} = A^{-1}\dot{\boldsymbol{w}} = \boldsymbol{g}, \tag{49}$$

where

$$\boldsymbol{g} = \left(\frac{-\gamma M^2}{(M^2-1)} \frac{\dot{p}}{u} \frac{\dot{p}}{(M^2-1)} \frac{\dot{s}}{u} \frac{\dot{y}_i}{u} \right)^T, \tag{50}$$

and $M = u/a$ is the flow Mach number. Eq. (49) has the same form as Eq. (1), where vector $\boldsymbol{w}$ replaces $\boldsymbol{y}$. Moreover, since the independent variable is x (space) instead of t (time), the time scales τ must be interpreted as spatial scales and the eigenvalues λ have the dimensions of the reciprocal of length instead of time.

7. RESULTS

The performance analysis of the new scheme is evaluated in terms of accuracy, computational efficiency, and consistency. Its performance is compared against that of the LSODE [48] package. All calculations were carried out by using a Macintosh G3/300 computer and the ABSOFT FORTRAN compiler.

7.1. Design and Performance Parameters

A parametric investigation has been carried out to assess the performance of the new algorithm. The main design parameters which control the algorithm operations are perturbed one at the time to identify their effects and relative importance. To simplify the presentation of the results, the main design parameters of the algorithm are summarized in the following. Then the performance parameters used will be enlisted to to compare the different options.

7.1.1. *Design Parameters*

An LSODE user can define the level of accuracy of the calculation by varying two (possibly vector) parameters *rtol* and *atol*, which control the maximum relative and absolute errors on each unknown tolerated at each step of the calculation. The size of the integration step and the order of accuracy of the scheme of integration are adjusted algorithmically so that the specified level of accuracy is obtained. In all calculations reported herein, the Jacobian is generated numerically by LSODE using the source term routine provided by the user.

A CSP user can control the level of accuracy of the calculation by setting bounds on two sources of errors, i.e., the numerical integration and the omission of the exhausted amplitudes. First, the specific explicit method selected for the integration of the slow components sets an accuracy level, the exact magnitude of which is controlled by the size of the integration step as a fraction α of the local driving scale $\tau(M+1)$. In addition, the accuracy can be controlled by the two (possibly vector) parameters ϵ_{rel} and ϵ_{abs}, which define the number of fast modes deemed exhausted and therefore neglected from the integration. Note that the mechanism by which ϵ_{rel} and ϵ_{abs} affect the solution accuracy is different from that of *rtol* and *atol*; the former specify the maximum allowed error in the computation of the time derivative of the solution, while the latter specify the maximum allowed error in the solution itself.

Given an explicit integration method, the parameters α, ϵ_{rel}, and ϵ_{abs} can in principle be adjusted so as to satisfy some *rtol* and *atol* as done in LSODE. The association of the three parameters α, ϵ_{rel}, and ϵ_{abs} on the solution accuracy depends on the particular explicit integration method employed.

In fact, the combined local error introduced by the omission of the exhausted amplitudes (Eq. (39)) and by the explicit integration scheme (Eq. (42)) is

$$E^i_{tot} = O\left(\alpha\left[\epsilon^i_{rel}|y^i| + \epsilon^i_{abs}\right]\right) + O(\alpha^n f^i(\boldsymbol{y})). \tag{51}$$

Note that the first term in the expression above gives a direct estimate of the local error, while from the second term only an upper bound can be obtained, since the function $f^i(\boldsymbol{y})$ is in general very complex to compute. In the following, the results of the parametric analysis reported will clearly illustrate the effects of the three parameters on the solution accuracy.

7.1.2. *Performance Parameters*

For this specific reactive flow problem three conservation laws of global mass flow, momentum, and energy have to be satisfied, together with the conservation of E elements. Moreover, if the composition is defined by using nondimensional (mass or mole) fractions

TABLE I
Definition of Test Case

Mixture	φ	T_∞ [K]	p_∞ [atm]	L [m]
CH_4/Air	1.0	1000	0.6	0.80

then these fractions must always sum up to one, and they must always be nonnegative. Thus, following the terminology of Ref. [55–57], the numerical solution should satisfy $3 + E + 1$ invariants and N_s inequalities at each integration step. However, both LSODE and the CSP algorithms cannot exactly satisfy these properties, and there has been no attempt to eliminate unknowns by explicitly resorting to these (linear and nonlinear) invariant relations.

Instead, the deviation of the computed invariants from the exact values will provide useful measures of the numerical errors. To this aim, we will report the relative error on mass flow, *err* (scaled with respect to the initial value) averaged over the integration domain.

To make the presentation of the properties of the algorithm complete, additional performance parameters will be introduced, such as (i) the number of integration steps N_{step}, (ii) the number of function evaluations N_{fe}, and (iii) the number of Jacobian evaluations N_{je} needed to complete the integration.

7.2. Free-Stream Conditions

The selected values of pressure p_∞ and temperature T_∞ ahead of the shock are reported in Table I, where φ is the equivalence ratio of the mixture and L the distance from the shock where the *NO* species reaches equilibrium.

The Chapman–Jouguet (CJ) detonation state parameters can be obtained by using the SP-273 package [58]. By definition, the burned mixture downstream of a CJ detonation is in thermodynamic equilibrium and has an “equilibrium” flow Mach number of one. The CJ parameters corresponding to the free-stream conditions specified in Table I are reported in Table II.

However, for this specific mixture and free-stream conditions, it is impossible to compute the CJ detonation structure by means of a finite rate chemistry, steady-state flow model because the detonation structure develops a singularity before equilibrium is attained, as will be shown in the next section.

Therefore, the test case involves a free-stream Mach number M_∞ ahead of the normal shock slightly higher than the CJ value of $M_{CJ} = 2.7931$. We selected for M_∞ the value 2.810, since this is the minimum value allowing to reach equilibrium. This value yields, for the values of pressure p_∞ and temperature T_∞ selected in Table I, an “overdriven” detonation wave slightly stronger than the corresponding CJ detonation.

TABLE II
CJ Detonation; Species Are Expressed in Mole Fractions

Type	M_{CJ}	V_{det} [m/s]	p [atm]	T [K]	H_2O	NO
C.J.	2.7931	1741.6	3.0981	2892.7	0.147	0.01204

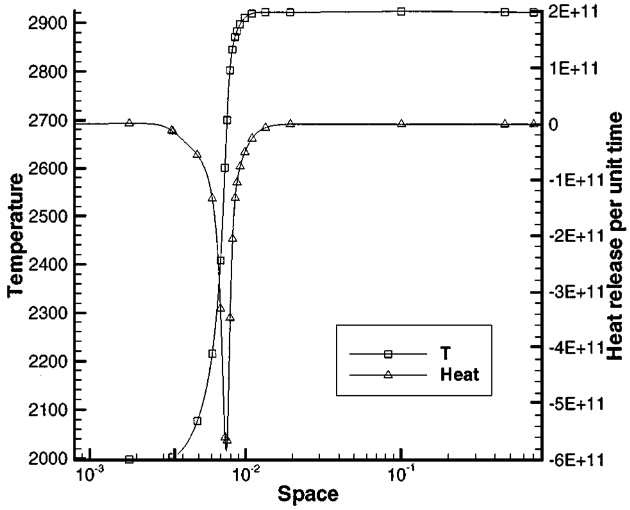

FIG. 1. OD detonation: evolution of temperature and heat released per unit time.

7.3. Numerical Solution by LSODE

The package LSODE is used to obtain the reference solution. The solution is documented by means of plots starting at the post-shock state and ending where *NO* attains its equilibrium value. The evolution in space of the mixture temperature *T* and of the heat released per unit time is shown in Fig. 1. The largest heat release clearly occurs inside the reaction zone lying between 2.e-03 m and 1.e-02 m. Thus, the reaction zone is about 8 mm wide compared with the 1 m width of the integration domain.

The plots of *NO* and CH_4 mass fractions are reported in Fig. 2. These data show that methane is quickly and almost fully consumed within the reaction zone, whereas *NO*, because of its slow kinetics, keeps increasing in the long-lasting phase downstream of the reaction zone.

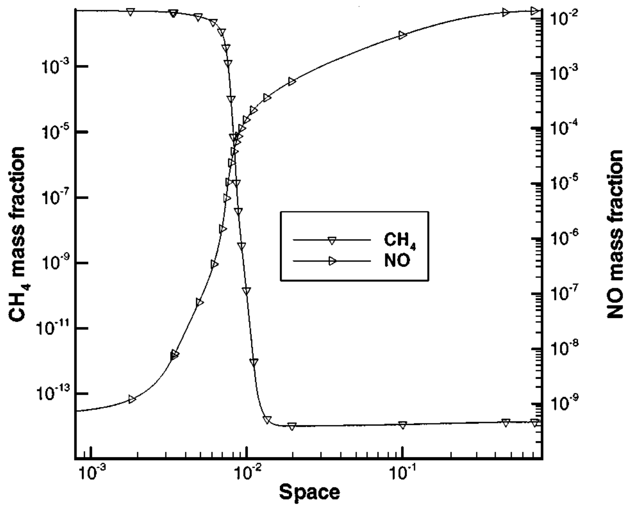

FIG. 2. OD detonation: evolution of CH_4 and NO mass fractions.

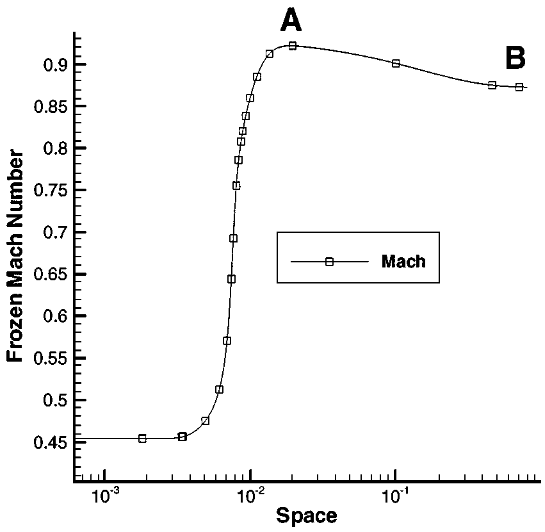

FIG. 3. OD detonation: evolution of flow Mach number M.

The formation of *NO* is an endothermic process, as demonstrated by the slight temperature drop noticeable in Fig. 1. This causes the "frozen" flow Mach number $M = u/a$ to first attain a maximum value right after the reaction zone (point A in Fig. 3 and then to relax to a lower value at equilibrium (point B in Fig. 3.)

Under this circumstance, the CJ state cannot be found as the numerical solution of the steady-state, nonequilibrium, reactive Euler equations, since the model becomes singular (the denominator of Eq. (50) becomes zero) as soon as the flow becomes "frozen" sonic. In fact, if one tries to attain at the CJ condition starting from free-stream Mach numbers M_∞ higher than M_{CJ}, then the "frozen" sonic condition will first occur at the location of maximum Mach (point A), that is far earlier than the achievement of the equilibrium stage (point B).

As previously noted, the lowest free-stream Mach number M_∞ allowing a complete numerical solution for the free-stream pressure and temperature given in Table I was found to be 2.810. For this M_∞, the maximum value of Mach number found at point A is 0.923 and at point B is 0.8737, which is much lower than unity (Fig. 3). Therefore, the slowest detonation found numerically is of the "Over-Driven" type (OD). The detonation parameters of the OD detonation are reported in Table III which may be compared to Table II to appreciate their deviation from the corresponding CJ values.

The effect of *rtol* in the solution accuracy is displayed in Fig. 4, where the evolution of the temperature and the relative error on mass flow downstream of the reaction zone is shown for

TABLE III
OD Detonation

Type	M_∞	V_{det} [m/s]	p [atm]	T [K]	H_2O	NO
O.D.	2.8100	1750.2	3.4247	2922.4	0.146	0.01239

Note. Species are expressed in mole fractions; frozen Mach number at equilibrium is 0.8737; and max Mach number is 0.923.

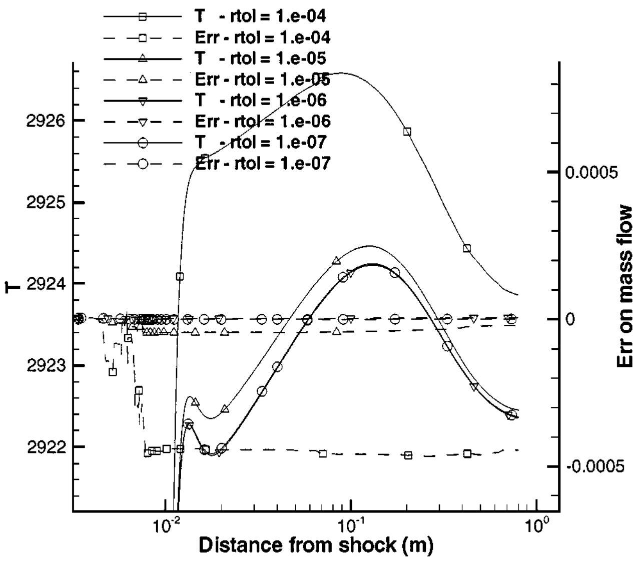

FIG. 4. LSODE's converges as the error controls are tightened; convergence is obtained with *rtol* = 1.*e* − 6.

$atol = 1.e - 14$. Given that as *rtol* decreases the solution accuracy increases, the results of Fig. 4 demonstrate that a converged solution is obtained with $rtol = 1.e - 6$, which yields an average error of the order $O(10^{-6})$. As the results of Table IV show, increasing the required accuracy (smaller *rtol*) asks for more integration steps and functions and Jacobian evaluations.

The LSODE solution obtained with $rtol = 1.e - 6$ (requiring 1010 integration steps, 12492 function evaluations, 210 Jacobian evaluation, and a total of CPU time of 36 s) will be used as a reference for the evaluation of the CSP-explicit algorithm's performance.

7.4. Numerical Solution by the New Explicit Schemes

The following sections will first illustrate how the dynamics of the system can be described and exploited by the new scheme. Then its performance, in terms of accuracy and computational efficiency, will be evaluated.

7.4.1. *Dynamics of Modes*

The detonation structure can be analyzed from the CSP point of view, mainly by inspection of the evolution in time of the number of exhausted modes M, shown in Fig. 5. Right

TABLE IV
LSODE's Package Performance

rtol	atol	CPU ind.	CPU reac.	CPU recom.	CPU tot.	err	N_{st}	N_{fe}	N_{je}
1.e-4	1.e-14	7.85s	7.32s	16.8s	32.0s	−4.51e-04	698	11540	201
1.e-5	1.e-14	8.42s	9.77s	9.25s	27.4s	−2.99e-05	756	9591	162
1.e-6	1.e-14	12.4s	11.5s	12.3s	36.2s	+3.83e-06	1010	12492	210
1.e-7	1.e-14	16.3s	15.6s	28.0s	59.8s	−4.07e-07	1506	21125	362

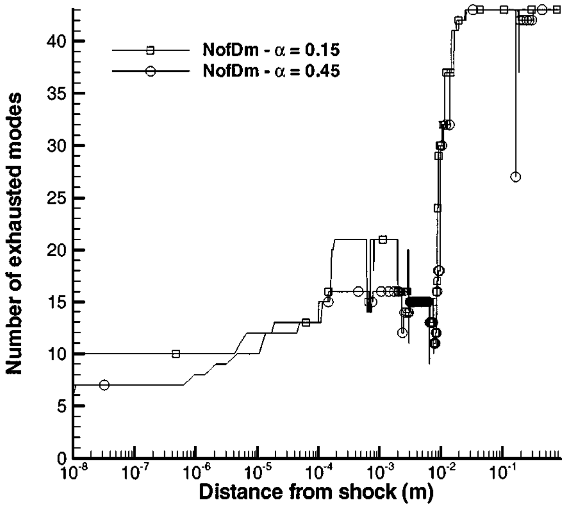

FIG. 5. Exhausted modes evolution as obtained by setting $\alpha = 0.15$ and 0.45.

behind the shock wave, the flow is under strong nonequilibrium conditions. Thus, there are no exhausted modes and $M = 0$. However, as the auto-ignition proceeds, more and more modes become exhausted. Three regions exist where M stays rather constant:

Induction. The first region starts right after the initial transient which follows the shock perturbation at 1.e-04 m and ends at 2.e-03 m. It corresponds to the induction period during which the radicals form. According to the prescribed accuracy, the number of the exhausted modes ranges from 15 to 20. Therefore, in this region an equilibrium manifold forms. This suggests that during induction 15 to 20 species are in quasi steady-state and that reduced models, involving $48 - 15 = 33$ to $48 - 20 = 28$ slowly varying species, can be constructed.

Reaction. When a sufficient pool of intermediates forms, the reaction zone starts; the major chemical activity occurs which can be identified by the fuel breakdown and by the sudden temperature rise. This region lies between 2.e-03 m to 1.e-02 m, and the number of exhausted modes drops to 7–15, meaning that the degree of stiffness has been locally reduced. Moreover, the nonlinear effects are stronger because of the sensitivity of the reaction rates to the large changes in temperature. As shown in the following, the softening of the stiffness and the strong nonlinear effects have a significant effect on the performance of the new algorithm.

Recombination. As soon as the fuel has been consumed, the recombination zone starts and persists up to the end of the integration domain (1 m). During this period, *NO* slowly forms endothermically—the temperature in Fig. 4 decreases slightly—and reaches its equilibrium value asymptotically. This zone is characterized by much slower scales than the induction or reaction zones. As a result, the number of exhausted modes rises to 41–43 indicating that an equal number of species are in quasi steady-state and that a reduced model involving five to seven species, basically those related to the formation of *NO* kinetics, can be constructed.

Note that once the number of exhausted modes and the corresponding CSP vectors are known, it is easy to compute the CSP data (i.e., steady-state pointer, participation, and

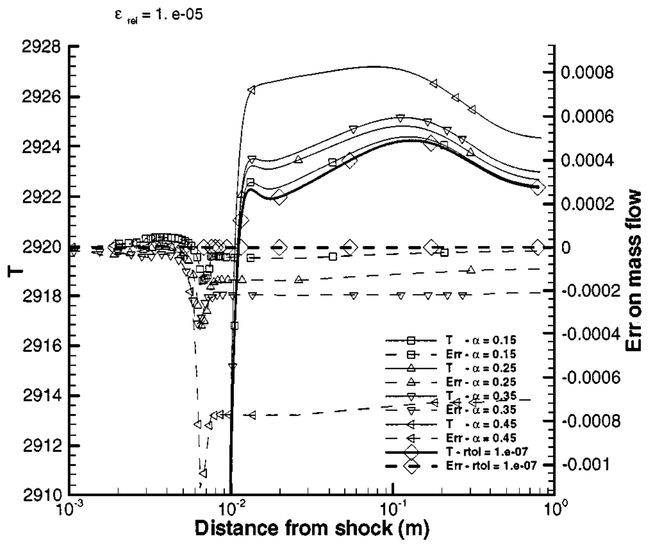

FIG. 6. CSP's convergence; fine dashed lines are the CSP solutions; and thick solid line is the LSODE reference solution.

importance indices) which allow both to analyze the detonation structure and to construct proper reduced mechanism as fully described in Refs. [37–39].

7.4.2. *Accuracy of the New Algorithm*

The evolution of the temperature and the error in the reaction and recombination regions are shown in Fig. 6, for different values of the parameter α and $\epsilon_{rel} = 1.e - 5$ and $\epsilon_{abs} = 1.e - 10$. It is shown that as α decreases, the error reduces and the CSP solution converges to the LSODE reference solution.

The space evolution of the error, displayed in Fig. 6, shows that its magnitude stays within the bounds imposed and that its largest value occurs inside the reaction zone. In this zone, where the solution exhibits the largest slopes, the CSP method encounters the greatest difficulties. This is also demonstrated by the results displayed in Fig. 7, in which the integration steps selected by LSODE and CSP are compared. It is shown that the steps are remarkably similar inside the induction and recombination zones, despite the completely different criterion upon which LSODE and the new algorithm make their decisions, whereas the largest discrepancies are found inside the reaction zone. There the new algorithm takes much smaller integration steps than LSODE.

7.4.3. *Efficiency of the New Algorithm*

The efficiency of the new algorithm can be assessed by inspecting the data summarized on Table V, where the design parameters of the algorithm considered are ϵ_{rel} and α. Table V clearly shows that:

- the new algorithm is approximately one order of magnitude slower than LSODE;
- the new algorithm is able to deliver an accuracy comparable to that of LSODE by altering ϵ_{rel} and α: the increased accuracy is obviously obtained at the expense of a higher computational cost;

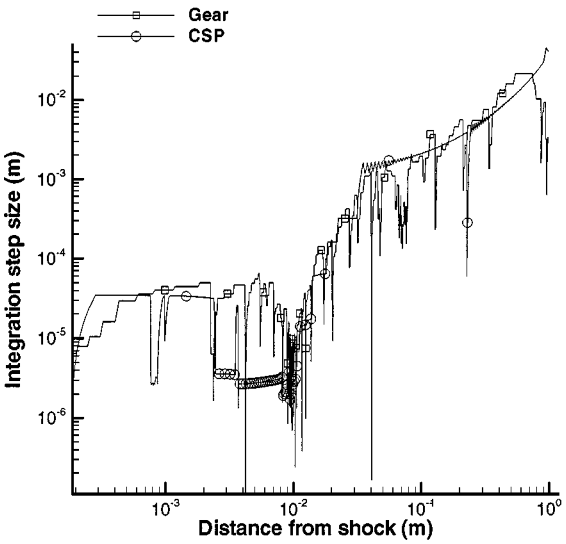

FIG. 7. CSP vs LSODE integartion step size evolution.

- the ratio of the number of function and Jacobian evaluations for the new algorithm and for LSODE scales almost proportionally to the the ratio of the CPU time required by the two methods;
- the difference between the number N_{je} of Jacobian evaluations and the number N_{st} of steps is equal to the number of times the resetting procedure (repetition of the integration step), described with reference to Eq. (41), is enforced; the data of Table V demonstrate that the conditions for resetting occur very rarely.

TABLE V

CSP-Based Algorithm Performance

Effect of α; $\epsilon_{rel} = 1$.e-5; $\epsilon_{abs} = 1$.e-10								
α	CPU ind.	CPU reac.	CPU recom.	CPU tot.	err	N_{st}	N_{fe}	N_{je}
0.45	176 s	228 s	191 s	594 s	−7.21e-04	1931	110946	1947
0.35	225 s	303 s	311 s	839 s	−2.15e-04	2684	153867	2700
0.25	194 s	380 s	140 s	714 s	−1.10e-04	2430	139830	2454
0.15	290 s	606 s	178 s	1070 s	−2.23e-05	3760	215750	3786
Effect of ϵ_{rel}; $\alpha = 0.25$; $\epsilon_{abs} = 1$.e-10								
ϵ_{rel}	CPU ind.	CPU reac.	CPU recom.	CPU tot.	err	N_{st}	N_{fe}	N_{je}
1.e-4	302 s	414 s	182 s	898 s	−4.57e-03	3080	178750	3138
1.e-5	194 s	380 s	140 s	714 s	−1.10e-04	2430	139830	2454
1.e-6	64.2 s	308 s	217 s	590 s	+5.41e-04	1910	110520	1940
1.e-7	68.8 s	738 s	506 s	1313 s	+8.50e-06	4501	257657	4521

7.4.4. *Comments about Accuracy and Efficiency*

The findings related to both the accuracy and the computational efficiency of the new algorithm should not be surprising. The new algorithm, apart from the error caused by the explicit scheme employed for the integration of the slow components of the derivatives, introduces an additional error produced by the higher order terms omitted which are related to the exhausted fast scales.

Inside the reaction zone, where the strongest nonlinearities and the largest slopes of the solution appear, two kinds of problems occur. First, the negligence of the derivative terms in the formulas for computing the basis vectors, i.e., Eqs. (17) and (18), is not such a valid assumption. As a result, the full dimensions of the manifold cannot be identified. Fewer exhausted modes are identified and the integration there proceeds with a time step much smaller than the local characteristic time scale, thus slowing down the computations. Second, the function $f(\boldsymbol{y})$ in the expression for the local error produced by the slow part integration (Eq. (42)) attains the maximum values, causing a deterioration of the solution accuracy. Both these effects are easily recognized in the results displayed in Figs. 6 and 7. These results indicate that the new algorithm will be especially successful when dealing with very stiff and quasi-linear problems.

7.5. A Hybrid Approach

The results showed that the new algorithm, consistently with the assumptions of the theory upon which it is build, performs better in regions of high stiffness and worse elsewhere. Thus, a test has been carried out in which the ODEs are integrated by using the new explicit scheme everywhere but in the reaction zone wherein LSODE is used instead. The performance obtained by this "hybrid" procedure is reported in Table VI.

By comparing the data in Table VI with those obtained by using the new algorithm throughout the whole domain (Table V) reveals that:

- the new algorithm takes most of the time to compute (not very accurately) the flame zone;
- the accuracy provided by the new algorithm in the induction and recombination zone is very high even when large α (up to 0.9) are used;
- inside the recombination zone, the CPU time is still high because there are up to 43 exhausted modes to be found at each integration step.

Now consider that for most applications of engineering interest, the reaction zone (or zones) covers only a minute fraction (however important) of the whole domain of integration,

TABLE VI
Performance of the Hybrid Calculation

Effect of ϵ_{rel}; $\epsilon_{abs} = 1.e - 10$; $\alpha = 0.90$

ϵ_{rel}	CPU ind.	CPU reac.	CPU recom.	CPU tot.	err	N_{st}
1.e-4	25 s	15 s	157 s	197 s	−1.56e-04	934
1.e-5	31 s	16 s	224 s	271 s	−7.66e-07	1119

Note. (i) CSP-based algorithm used everywhere but in the flame, and (ii) LSODE's package used in the flame.

and that almost the whole flowfield is affected by stiffness. In view of solving these kinds of PDEs problems, this test suggests the following tentative strategy:

- to use LSODE, or a stabilized explicit scheme such as a RKC scheme, in the flame (where the problem is mildly stiff);
- to use CSP (after having improved its speed) elsewhere where the problem is very stiff.

8. DISCUSSION OF THE RESULTS

The findings illustrated in the previous section lead to different conclusions, depending on whether one wishes to solve an ODE or a PDE problem. Therefore, the discussion will be done separately for the two options.

8.1. Discussion for ODEs Problems

Comparing the findings of Table V to those of Table IV pointed out that one of the main reasons that makes the new algorithm less performing (in terms of computational efficiency) than LSODE is recomputing the Jacobian at each integration step.

In contrast, LSODE updates the Jacobian (and sometimes only an approximation of the Jacobian) only once over an average of 5–10 time steps. LSODE can adopt this strategy because it exploits the knowledge of the solution and of the Jacobian at previous integration steps to both update the new Jacobian and to take a decision about the order of accuracy of the scheme of integration (LSODE is a variable order, variable step, multistep, implicit algorithm). Overall, this makes LSODE a very efficient solver for ODE problems.

Instead, the new algorithm in the present form is a one-step, fixed-order, variable-step explicit scheme. Therefore, it cannot take advantage of the past history of the solution as does LSODE. The price paid is a higher number of function and Jacobian evaluation.

A further source of computational work for the new algorithm is the refinement procedure which is not required by LSODE. A profiling analysis of the code showed that the most CPU time-demanding calculations of the new algorithm are the two matrix products required to deflate the Jacobian from left and right.

However, LSODE must solve a set of linear equations for the implicit formulation, whereas the new algorithm has not. Therefore, for the new algorithm to become competitive, it is important to bring the cost of the refinement procedure down to the same level of the cost to solve the set of linear equations needed by the implicit formulation.

Last but not least, the coding of the new algorithm in its present form is far from being optimal, and it is definitely less efficient than the highly optimized and extensively tested LSODE package. Consider as an example, that the CPU time decreased by a factor of 3 simply by carrying out the matrix multiplications with the help of a routine optimized for processors provided with a cache memory.

8.2. Discussion for PDEs Problems

The standard way to integrate numerically a stiff PDE problem involves first an operator splitting which allows the integration of the stiff source term separately from the spatial differential operators (convection and diffusion) by using the stiff schemes embodied in LSODE (or later versions of it [9]).

The performance of LSODE in this context is much poorer than that achievable when solving ODE problems. This happens for the following reasons:

• The Jacobian at past times is not available anymore, and, therefore, it ought to be recomputed at each step and space location as done in the CSP algorithm, because (i) to save it at different time levels and at each space location would result in unmanageable large storage requirements, and (ii) for unsteady calculations involving traveling waves, it is not possible to extrapolate the new Jacobian from those evaluated at past times but at a fixed space location. Thus, the number of Jacobian evaluations for LSODE and the new CSP algorithm in a PDE problem will become of similar order of magnitude.

• Since LSODE is a multistep scheme, its starting step procedure is much more expensive as compared to the cost of the time steps that follow. In fact, not only the lack of past Jacobians forces the procedure to recompute the full Jacobian, but also the variable step/variable-order features become either unavailable or more expensive when envisioned for the first step. Unfortunately, and for the same reasons explained above, each integration of the stiff operator of a split PDE problem has to be carried out as if composed of all and only first steps.

To quantify the loss of LSODE's efficiency under these circumstances, we attempted to emulate how LSODE would operate in a PDE context using an ODE model. This can be achieved by forcing LSODE to reset the control parameter *istate* equal to 1 at the beginning of each integration step; in so doing, all internal arrays of LSODE, including previous values of the local Jacobian matrix, are cleared.

In a real PDE problem, the integration time-step size is determined on the basis of a CFL number criterion. Given this time step, when LSODE takes over, the integration of the stiff part of the PDE along the CFL-determined time step is performed with a number of smaller steps, determined by the method itself so that the prespecified accuracy is respected. In the numerical test performed in order to test the efficiency of LSODE in a PDE environment using an ODE problem, we had to provide LSODE the time step sequence, since no spatial operator is actually present in the set of ODEs considered here. Two different sequences have been considered (see Fig. 7); the one generated by LSODE when it works in the standard mode (the method chooses the time step using the solution history) and the one generated by the CSP new algorithm, with a time step defined as the locally fastest of the slow time-scales. The numerical test allowed us to draw the following conclusions:

• As the results displayed in Fig. 8 show, by forcing a new start at the beginning of each (externally supplied) time step, LSODE starts the integration with a very small substep. The size of this substep quickly recovers to that attained by the standard mode of integration. However, no matter how quick this recovery might be, this process of integration causes a severe loss of efficiency when compared to its standard mode of operation.

• As the data shown on Table VII indicate, the PDE operation mode on LSODE causes the evaluation of the Jacobian a very large number of times. This development, which is the major cause for the loss in efficiency, is due to the fact that the new start at the beginning of each time step clears out the memory of all previous steps and that the data required by the method to march have to be computed anew.

• The overall effect of the PDE operation mode of LSODE is to increase dramatically the CPU time needed to complete the integration. As Table VII shows, the 36 s for the LSODE operation in an ODE mode grow up to 811 s (or 1598 s) for the operation in a PDE

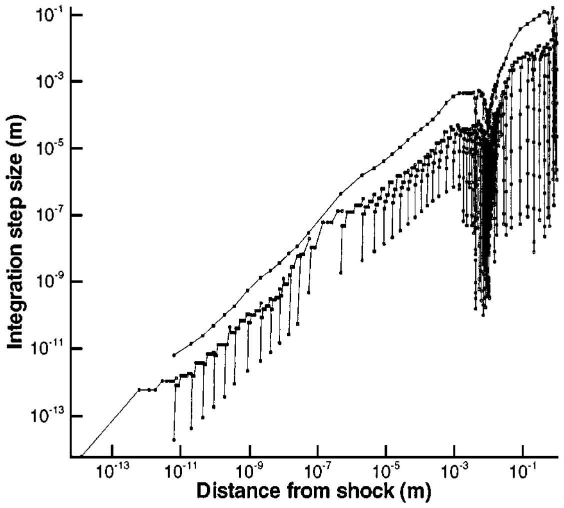

FIG. 8. Integration step size evolution as: (i) generated by LSODE with variable step control (continuous line) and (ii) prespecified integration step size sequence simulating operator splitting mode of integration (short lines).

mode, when the supplied time step sequence is provided by LSODE when it operates in an ODE mode (or by the CSP new algorithm).

On the other hand, the CSP-based algorithm only requires a limited amount of information made available from the previous integration step (although it does not strictly need it), that is, a CSP basis to be used as a first guess in the refinement procedure. This is easily available in PDE problems as well as in ODEs. In this perspective, the CSP-based algorithm can be classified as a self-starting scheme.

To conclude, the performance of LSODE in ODE problems, for a number of reasons, is better than that of the new scheme. However, this result cannot be safely extrapolated to PDE problems, where most of the advantages enjoyed by LSODE, when dealing with ODEs, disappears.

The extension of the new CSP-based scheme to PDE problems is the subject of ongoing research. Note, however, that other integration schemes, suited to integrate the stiff PDEs, are available (see a review in Ref. [11]), which are more efficient than LSODE in dealing with stiff PDEs, although less competitive than LSODE on a single, long-term integration.

TABLE VII
LSODE's Performance with Variable and Two Prespecified Integration Step Size Sequences

dt	rtol	atol	CPU tot.	err	N_{st}	N_{fe}	N_{je}
Variable step	1.e-6	1.e-14	36 s	+3.83e-06	1,010	12,492	210
LSODE sequence	1.e-6	1.e-14	811 s	+1.78e-06	9,839	292,921	5,320
CSP sequence	1.e-6	1.e-14	1598 s	9.32e-06	18,514	621,524	11,335

Note. One sequence generated by LSODE working in the standard variable step size mode, and the other by the CSP-based algorithm.

Therefore, the new time-scale split algorithm should be confronted with these others as well as with LSODE.

9. CONCLUSIONS AND FUTURE DIRECTIONS

A new explicit algorithm based on the CSP concepts has been developed and its performance has been compared against that delivered by the LSODE package. The results demonstrated that (i) the accuracy of the new algorithm is comparable to that provided by LSODE; (ii) the stability limitations of the conventional explicit schemes have been fully circumvented; and (iii) the computational efficiency of the code (algorithm plus coding) is still non satisfactory.

The possible directions to improve the computational efficiency have been identified: (i) the number of Jacobian evaluations and the number of times a new CSP basis must be updated have to be reduced, (ii) the cost of each refinement must be reduced, and (iii) the coding of the package must be optimized.

Before closing this paper, it seems appropriate to address the issue of how the explicit scheme could be extended to solve stiff PDEs problems, and why this might result in a better strategy as opposed to the standard operator (or time-step) splitting approach.

The explicit time-scale splitting should in principle be able to provide a direct coupling of the spatial scales with the slow time-scales originated by the source term, when the fast time-scales are due to the source term only.

The likely advantages of the time-scale splitting compared to the time-step splitting are the following:

- The time-scale splitting should allow the use of an explicit algorithm thus eliminating:
  —the need for locally implicit or multistep scheme;
  —the solution of nonlinear systems at each integration step; and
  —the extra storage associated to a multi-step scheme.
- The time-scale splitting provides an estimate of the order of magnitude of the local driving time-scale obtained by accounting for both convection, diffusion, and reactions. This estimate can be used:
  —to adjust (maximize) the integration step for time marching; and
  —to set the proper spatial discretization (grid resolution).

APPENDIX: THE PHYSICAL MODEL

This appendix describes the thermochemical model, the chemical kinetic model, and the conservation laws used in this work.

A.1. Thermo-Chemical Model

The mixture is composed of thermally and calorically perfect gases. The thermal equation of state for the mixture is formally identical to the one valid for a single ideal inert gas,

$$p = \sum_i p_i = \left(\sum_i \rho_i \right) RT = \rho RT. \tag{A.1}$$

The mixture pressure p and density ρ are defined in terms of partial pressures p_i and densities ρ_i; the mixture gas constant R is defined by the relation $R = \sum_i (\mathcal{R}/M) y_i = \sum_i R_i y_i$, where $\mathcal{R}$ is the universal gas constant, $\mathcal{M}_i$ are the molecular weights of the mixture species, and y_i are the species mass fractions. The reactive mixture of gases is considered to be in thermal equilibrium. Thus, the translational and rotational energies are accounted for as being fully excited at the temperature T, whereas vibration is in equilibrium.

The caloric equation of state is formulated by expressing the mixture enthalpy h in terms of the species enthalpies h_i,

$$h = \sum_i h_i(T) y_i, \tag{A.2}$$

where

$$h_i(T) = \Delta h^\circ_{f_i}(T_0) + \int_{T_0}^{T} c_{p_i}(T)\, dT. \tag{A.3}$$

The mixture is not calorically perfect because the enthalpy is a function of both temperature and mixture composition.

The entropy of the mixture is defined on the basis of the species entropies defined as

$$s^\circ_i(T) = \Delta s^\circ_{f_i} + \int_{T_0}^{T} c_{p_i}(T) \frac{dT}{T} \tag{A.4}$$

$$s_i(T, p_i) = s^\circ_i(T) - R_i \ln(p_i/p_0). \tag{A.5}$$

The species entropies s_i depend on the partial pressure p_i as well as on temperature T. For the mixture, we have

$$s^\circ_f(T, y_i) = \sum_i s^\circ_i(T) y_i \tag{A.6}$$

$$s(T, p, y_i) = s^\circ_f - R\left[\log(p/p_0) + \sum_i x_i \log x_i\right], \tag{A.7}$$

where the partial pressures p_i are expressed in terms of the mixture pressure p and the mole fractions x_i. The formation enthalpy $\Delta h^\circ_{f_i}(T_0)$ and entropy $\Delta s^\circ_{f_i}(T_0)$ are evaluated at standard conditions.

The appropriate sound speed for models of thermal equilibrium and chemical nonequilibrium is the frozen sound speed defined as $a^2 = \gamma p/\rho$, where γ is the ratio between the (frozen) specific heats c_p and c_v of the mixture

$$\gamma = \left(\sum_i c_{p,i} y_i\right) \Bigg/ \left(\sum_i c_{v,i} y_i\right) = c_p/c_v.$$

A.2. Chemical Kinetic Model

Let the N_s species forming the gaseous mixture be formed by N_e different elements. The expression X_i denotes the chemical formula of the i-th species, composed by N_e elements E^j

$$X_i = E^1_{a_{1,i}} E^2_{a_{2,i}} \cdots E^{N_e}_{a_{e,i}} \quad i = 1, N_s.$$

The subscript $a_{j,i}$ is the number of atoms of the element E^j in the species X_i. The set of reactions defining a kinetic model can be identified by the N_r symbolic relations

$$\sum_{i=1}^{N_s} \nu'_{i,k} X_i \rightleftharpoons \sum_{i=1}^{N_s} \nu''_{i,k} X_i \quad k = 1, N_r, \tag{A.8}$$

where N_r is the number of different reactions occurring among the N_s species; $\nu'_{i,k}$ and $\nu''_{i,k}$ are the stoichiometric coefficients of reactants and products, respectively.

The rate at which the k-th reaction progresses in time is quantifiable with the relation

$$r^k = K_f^k \left[\prod_{i=1}^{N_s} c_i^{\nu'_{i,k}} - \frac{(\mathcal{R}T)^{\sum_i (\nu''_{i,k} - \nu'_{i,k})}}{K_p^k} \prod_{i=1}^{N_s} c_i^{\nu''_{i,k}} \right] = r_f^k - r_b^k, \tag{A.9}$$

where the forward rate constant is in Arrhenious form

$$K_f^k = B_k T^{\alpha_k} \exp(-E_k/\mathcal{R}T), \tag{A.10}$$

where E_k è is the activation energy, and B_k is the steric factor. The equilibrium constant K_p^k is evaluated as

$$K_p^k(T) \overset{\text{def}}{=} exp\left(- \frac{\sum_i^{N_s} (\nu''_{i,k} - \nu'_{i,k}) \mu_i^\circ}{\mathcal{R}T} \right). \tag{A.11}$$

In these relations, the chemical potential $\mu_i^\circ = h_i - T s_i^\circ$ is evaluated under equilibrium conditions (NASA polynomials).

The net rate of production of the i-th species $\dot{c}_i$ resulting from the simultaneous action of the N_r reactions can be evaluated with the relations

$$\dot{c}_i = \sum_{k=1}^{N_r} \left(\frac{\rho \psi}{\mathcal{M}} \right)^{l_k} (\nu''_{i,k} - \nu'_{i,k}) r^k. \tag{A.12}$$

The exponent l_k is equal to one when the reaction includes third bodies, and zero elsewhere. Third body efficiencies ψ_i are combined into the term $\psi = \sum_i \psi_i x_i$; $\mathcal{M} = \sum_i \mathcal{M}_i x_i$ is the molecular weight of the mixture. The chemical kinetic model for the system methane/ air can be found in [50].

A.3. Reactive Euler Equations

On the basis of dimensional estimates, it is possible to safely neglect all transport phenomena in the flow regimes characteristic of detonation waves. The gas dynamics of reactive flows can be adequately described by the reactive Euler equations, i.e., the set of mass, momentum, and energy conservation laws formally identical to the one valid for an inert gas

$$\frac{D\rho}{Dt} + \rho \nabla \cdot \mathbf{q} = 0 \tag{A.13}$$

$$\rho \frac{D\mathbf{q}}{Dt} + \nabla p = 0 \tag{A.14}$$

$$\rho \frac{Dh}{Dt} - \frac{Dp}{Dt} = 0, \tag{A.15}$$

where **q** is the flow velocity vector, plus a set of rate equations for the production and consumption of chemical species

$$\frac{D\rho_i}{Dt} + \rho_i \nabla \cdot \mathbf{q} = \dot{\rho}_i \quad i = 1, N_s, \tag{A.16}$$

where $\dot{\rho}_i$ is the production rate of the i-th species. The set is closed by the thermal and calorical equations of state. By combining the definition of partial density ($\rho_i = y_i \rho$), with the conservation of the total mass, we can recast the conservation of the i-th species in terms of mass fractions

$$\frac{Dy_i}{Dt} = \frac{\dot{\rho}_i}{\rho} \quad i = 1, N_s, \tag{A.17}$$

where $\dot{\rho}_i$ is the rate of production on a mass basis. The rate of production on a molar basis $\dot{c}_i$ is related to $\dot{\rho}_i$ by $\dot{\rho}_i = \dot{c}_i \mathcal{M}_i$. Thus, we obtain the equations

$$\frac{Dy_i}{Dt} = \frac{\dot{c}_i \mathcal{M}_i}{\rho} = \dot{y}_i \quad i = 1, N_s, \tag{A.18}$$

expressing the transport and production of species mass fractions.

Differentiation of the thermal equation of state (A.1) yields

$$\frac{d\rho}{\rho} = d\log(p) - \frac{dh}{c_p} + \sum_i \left(\frac{R_i}{R} - \frac{h_i}{c_p T} \right) dy_i, \tag{A.19}$$

where c_p is the specific heat at constant pressure of the mixture. The differential of the static enthalpy in Eq. (A.19) can be expressed in terms of differential of pressure by using the conservation law of total enthalpy (Eq. (A.15)), and the resulting relation can be substituted in the conservation law of the total mass of the mixture (Eq. (A.13)) to yield

$$\frac{DP}{Dt} + \gamma \nabla \cdot \mathbf{q} = -\gamma \sum_i \left(\frac{R_i}{R} - \frac{h_i}{c_p T} \right) dy_i. \tag{A.20}$$

Moreover, the pressure gradient ∇p in Eq. (A.14) can be expressed in terms of gradient of the pressure logarithm $\nabla P = \nabla(\log(p))$ to cast the conservation law of momentum into the equivalent form

$$\frac{D\mathbf{q}}{Dt} + \frac{a^2}{\gamma} \nabla P = 0. \tag{A.21}$$

Finally, the conservation law of total enthalpy (Eq. (A.15)) can be replaced by the equation expressing the entropy production resulting from chemical reactions [46, 47]:

$$\frac{Ds}{Dt} = -\frac{1}{T} \sum_{i=1}^{N_s} \mu_i \dot{y}_i, \tag{A.22}$$

where the chemical potential of the i-th species is defined as $\mu_i(T, p_i) = h_i(T) - T s_i(T, p_i)$.

ACKNOWLEDGMENTS

Thanks are addressed to Prof. S. H. Lam for the many stimulating criticisms he offered to the authors during the development stages of the algorithm. The research has been partially funded by a bilateral cooperation protocol between Italy and Greece.

REFERENCES

1. W. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations* (Prentice Hall, New Jersey, 1971).
2. R. E. Mitchell and R. J. Kee, *A General Purpose Computer Code for Predicting Chemical Kinetic Behavior Behind Incident and Reflected Shocks*, SANDIA Report, SAND82-8205 UC-401 (1990).
3. R. J. Kee, J. F. Groar, M. D. Smooke, and J. A. Miller, *A FORTRAN Code for Modeling Steady Laminar One-Dimensional Premixed Flames*, SANDIA Report, SAND85-8240 UC-401 (1990).
4. K. Radhakrishnan, *LSENS A General Chemical Kinetics and Sensitivity Analysis Code for Homogeneous Gas-Phase Reactions*, NASA Reference Publication 1328 (1994).
5. G. V. Candler and R. W. MacCormack, *The Computation of Hypersonic Ionized Flows in Chemical and Thermal Non-Equilibrium*, Technical Paper 88-0511 (AIAA Press, Washington, DC, 1988).
6. J. V. Rakich, H. E. Bailey, and C. Park, *AIAA J.* **21**, 834 (1983).
7. B.‘Grossman and P. Cinnella, *J. Comput. Phys.* **88**, 131 (1990).
8. M. Baum, T. J. Poinsot, D. C. Haworth, and N. Darabiha, *J. Fluid Mech.* **281**, 1 (1994).
9. H. N. Najm, P. S. Wyckoff, and O. M. Knio, *J. Comput. Phys.* **143**, 381 (1998).
10. E. S. Oran, T. R. Young, J. P. Boris, and A. Cohen, *Combust. and Flames* **48**, 135 (1982).
11. E. S. Oran and J. P. Boris, *Numerical Simulation of Reacting Flows* (Elsevier, New York, 1987).
12. A. Bourlioux and A. J. Majda, *Combust. Flame* **90**, 211 (1992).
13. H. C. Yee and J. L. Shinn, *AIAA J.* **27**, 299 (1989).
14. T. R. A. Bussing and E. M. Murman, *AIAA J.* **26**, 1070 (1988).
15. W. L. Miranker, *Numerical Methods for Stiff Equations and Singular Perturbation Problems* (Reidel, Dordrecht, 1981).
16. N. Peters, Systematic reduction of flame kinetics: Principles and details, in *Fluid Dynamics of Combustion Theory*, edited by M. Onofri and A. Tesei (Longman, Harlow/New York, 1990) p. 232.
17. J. Y. Chen, *Combust. Sci.* **57**, 89 (1988).
18. N. Peters and B. Rogg, *Reduced Kinetic Mechanisms for Applications in Combustion Systems*, edited by N. Peters and B. Rogg (Springer-Verlag, Berlin, 1993).
19. M. Smooke, *Reduced Kinetic Mechanisms and Asymptotic Approximations for Methane-Air Flames*, edited by M. Smooke Springer Lecture Notes (Springer-Verlag, Berlin, 1991).
20. U. Maas and S. B. Pope, Implementation of simplified chemical kinetics based on intrinsic low-dimensional manifolds, in *24th Symposium (Intl.) on Combustion* (The Combustion Institute, Pittsburgh, 1992), p. 103.
21. A. Massias, D. Diamantis, E. Mastorakos, and D. A. Goussis, *Combust. Flame* **117**, 685 (1999).
22. A. Massias, D. Diamantis, E. Mastorakos, and D. A. Goussis, *Combust. Theory Model.* **3**, 233 (1999).
23. V. I. Lebedev, Explicit difference schemes with time-variable steps for solution of stiff systems of equations, Lecture Notes in Computer Science 1196, Numerical Analysis and its Applications (Springer-Verlag, Berlin, 1997), pp. 274–283.
24. A. A. Medovikov, High order explicit methods for parabolic equations, *BIT* **38**, 372 (1998).
25. J. G. Verwer, Explicit Runge–Kutta methods for parabolic partial differential equations, *Appl. Numer. Math.* **22**, 359 (1996).
26. S. H. Lam, Singular perturbation for stiff equations using numerical methods, in *Recent Advances in the Aerospace Sciences*, edited by Corrado Casci (Plenum, New York and London, 1985), p. 3.

27. S. H. Lam and D. A. Goussis, Understanding complex chemical kinetics with computational singular perturbation, in *22nd Symposium (Intl.) on Combustion* (The Combustion Institute, Pittsburgh, 1988), p. 931.
28. S. H. Lam and D. A., Goussis, Conventional asymptotic and computational singular perturbation for simplified kinetics modelling, in *Reduced Kinetic Mechanisms and Asymptotic Approximations for Methane-Air Flames*, edited by M. O. Smooke Springer Lecture Notes 384 (Springer-Verlag, Berlin, 1991).
29. C. Trevino and F. Mendez, Reduced kinetic mechanism for methane ignition, in *24th Symposium (Intl.) on Combustion* (The Combustion Institute, Pittsburgh, 1988) p. 121.
30. D. A. Goussis and S. H. Lam, A Study of homogeneous methanol oxidation kinetic using CSP, in *24th Symposium (Intl.) on Combustion* (The Combustion Institute, Pittsburgh, 1992), p. 113.
31. Y. Katsabanis and D. A. Goussis, Constructing Reduced Chemical Kinetic Mechanisms, in *Proceedings of the 3rd International Conference on Combustion Technologies, Portugal, 1995*, p. 32.
32. D. A. Goussis, *J. Comput. Phys.* **128**, 261 (1996).
33. M. D. Ardema, Computational Singular Perturbation Method for Dynamic Systems, in *IFAC Workshop on Singular Perturbations and Asymptotic Methods in Systems and Controls* (Boston, MA, 1989).
34. M. D. Ardema, Computational singular perturbation for optimal control, in *1990 Control Conference* (San Diego, CA, 1990).
35. A. V. Rao and K. D. Mease, A New Method for Solving Optimal Control Problems, in *Proceedings of the AIAA Guidance, Navigation and Control Conference* (1994), p. 818.
36. M. Hadjinicolaou and D. A. Goussis, Asymptotic solution of stiff PDEs with the CSP method—The reaction diffusion equation, *SIAM J. Sci. Comp.* **20**, 781 (1999).
37. S. H. Lam, *Combust. Sci. Technol.* **89**, 375 (1993).
38. S.H. Lam and D. A. Goussis, *Int. J. Chem. Kinet.* **26**, 461 (1994).
39. S. H. Lam, Reduced chemistry modeling and sensitivity analysis, in *Lecture Notes for Aerothermochemistry for Hypersonic Technology*, 1994–1995 Lecture Series Programme (Von Karman Institute for Fluid Dynamics, April 24–28, 1995).
40. S. H. Lam and D. A. Goussis, *The Analytical Foundation of CSP*, MAE-Princeton Report # 1800 (Oct. 1991).
41. M. Van Dyke, *Perturbation Methods in Fluid Mechanics* (The Parabolic Press, Stanford, 1975).
42. D. A. Goussis, S. H. Lam, and P. A. Gnoffo, Reduced and simplified chemical kinetics for air dissociation using computational singular perturbation, AIAA Paper 90-0644, presented at the 28th AIAA Aerospace Sciences Meeting, Reno, Nevada, 1990.
43. A. Eden, C. Foias, B. Nicolaenko, and R. Temam, *Exponential Attractors for Dissipative Evolution Equations* (Masson/John Wiley, England, 1994).
44. A. Jennings and J. J. Mc Keown, *Matrix Computation* (Wiley, England, 1992).
45. J. H. Wilkinson, *The Algebraic Eigenvalue Problem* (Oxford Univ. Press, London, 1992).
46. A. H. Shapiro, *Compressible Fluid Flows* (Wiley, New York, 1953).
47. W. Fickett and W. C. Davis, *Detonation* (Univ. of California Press, Berkeley, 1979).
48. A. C. Hindmarsh and Odepack, A systematized collection of ODE solvers, in *Scientific Computing*, edited by R. S. Stepleman *et al*. (North-Holland, Amsterdam, 1983), p. 55.
49. C. J. Jachimovski, *An Analytical Study of the Hydrogen-Air Mechanism With Application to Scramjet Combustion*, NASA TM 2791 (1988).
50. B. Serauskas, T. Bowman, G. Smith, D. Golden, B. Gardiner, and V. Lissianski, *GRI Mechanism for Methane/Air*, Version 1.1, available at http:www.me.berkeley.edu/gri.mech.
51. S. Yungster and M. J. Rabinowitz, Numerical Study of Shock-Induced Combustion in Methane-Air Mixtures, in *29th Joint Propulsion Conference and Exhibit*, AIAA-93-1917 (Monterey, CA, 1993).
52. E. Turkel *Review of Preconditioning Methods for Fluid Dynamics*, NASA CR-189712, ICASE Report No. 92-47 (1992).
53. D. J. Higham, *IMA J. Num. Anal.* **9**, 1 (1989).

54. G. M. Shroff and H. B. Keller, *SIAM J. Numer. Anal.* **30**, 1099 (1993).
55. L. F. Shampine, *Comput. Math. Appl.* **12B**, 1287 (1986).
56. U. M. Ascher, H. Chin, and S. Reich, *Numer. Math.* **67**, 131 (1994).
57. C. W. Gear, *SIAM J. Sci. Stat. Comput.* **7**, 734 (1986).
58. S. Gordon and B. J. McBride, *Computer Program for Calculation of Complex Equilibrium Compositions, Rocket Performance, Incident and Reflected Shocks, and Chapman-Jouguet Detonations*, NASA, SP-273, (1971).