HAL
open science

High-order adaptive time discretisation of one-dimensional low-Mach reacting flows: a case study of solid propellant combustion

Laurent François, Joël Dupays, Dmitry Davidenko, Marc Massot

To cite this version:

Laurent François, Joël Dupays, Dmitry Davidenko, Marc Massot. High-order adaptive time discretisation of one-dimensional low-Mach reacting flows: a case study of solid propellant combustion. Journal of Computational and Applied Mathematics, 2024, pp.115758. 10.1016/j.cam.2024.115758 . hal-02888035v2

HAL Id: hal-02888035

https://hal.science/hal-02888035v2

Submitted on 4 Jan 2024

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

High-order adaptive time discretisation of one-dimensional low-Mach reacting flows: a case study of solid propellant combustion

Laurent François[a,b], Joël Dupays[a], Dmitry Davidenko[a], Marc Massot[b]

[a] *ONERA, DMPE, 6 Chemin de la Vauve aux Granges, 91120 Palaiseau, France*
[b] *CMAP, CNRS, École polytechnique, Institut Polytechnique de Paris, Route de Saclay, 91120 Palaiseau Cedex, France*

Abstract

Solving the reactive low-Mach Navier-Stokes equations with high-order adaptive methods in time is still a challenging problem, in particular due to the handling of the algebraic variables involved in the mass constraint. We focus on the one-dimensional configuration, where this challenge has long existed in the combustion community. We consider a model of solid propellant combustion, which possesses the characteristic difficulties encountered in the homogeneous or spray combustion cases, with the added complication of an active interface. The system obtained after semi-discretisation in space is shown to be differential-algebraic of index 1. A numerical strategy relying on stiffly accurate Runge-Kutta methods is introduced, with a specific discretisation of the algebraic constraints and time adaptation. High order is shown to be reached on all variables, while handling the constraints properly. Three challenging test cases are investigated: ignition, limit cycle, and unsteady response with detailed gas-phase kinetics. We show that the time integration method can greatly affect the ability to predict the dynamics of the system. The proposed numerical strategy exhibits high efficiency and accuracy for all cases compared to traditional schemes used in the combustion literature.

Keywords: low-Mach reactive flows, solid propellant combustion, differential-algebraic equations, high-order adaptive time integration, nonlinear combustion instability

1. Introduction

Solving for steady and unsteady homogeneous and spray or solid-propellant combustion in one-dimensional flame simulations has attracted enormous attention in the combustion community starting with the seminal work conducted by M.D. Smooke and collaborators between Sandia and Yale University [1–5]. To our knowledge, in these studies and subsequent papers, most of the one-dimensional unsteady CFD codes for such applications use a time integration based on splitting and/or implicit methods, which are usually limited to first-order accuracy in time. No mathematical analysis has been reported regarding the nature of the system of coupled equations obtained after semi-discretisation in space, where the handling of some variables (e.g. surface temperature for the solid propellant case, eigenvalues such as strain rate or mass fluxes) requires careful examination in connection with boundary conditions. Besides, relying on low-order integration methods may prove disadvantageous in terms of accuracy, performance and ability to resolve fine details of the dynamics. Indeed high-order methods are especially important when investigating instabilities and nonlinear behaviours, e.g. limit cycles, where growth of some modes can only be captured by high-fidelity numerical methods [6]. One exception is the work on the dynamics of non-premixed counterflow flames by Im et al. [7], where the high-order time integrator DASSL for differential-algebraic equations (DAEs) was used, but the constraint formulation was rather involved (introduction of compressibility effects to reduce the index) and details on the convergence and efficiency were not the main focus of the paper.

Even if solid propellant combustion brings in additional difficulties and constraints related to the heterogeneous nature of the flow, it involves the same problematic as homogeneous or two-phase flow combustion in one-dimensional low-Mach flows. Therefore, we choose to focus on the more complete case of solid propellant combustion. Before getting into the key original findings and contribution of the present paper, let us introduce the specific topic of solid propellant combustion and explain why the one-dimensional configuration is especially important in this case and requires an efficient and accurate numerical resolution.

*Corresponding author
Email address: laurent.francois@onera.fr (Laurent François)

Solid propellant combustion is a key element in rocket propulsion and has been extensively studied since the 1950s [8–11]. It involves a solid phase and a gas phase, separated by an interface. The solid is heated up by thermal conduction and radiation from the gas phase. At its surface, the solid propellant decomposes, melts and evaporates through a pyrolysis process as the interface regresses. The resulting gaseous products react and form a flame which heats back the solid, allowing for a sustained combustion. A key element is the regression speed of the propellant surface and its dependence on the combustion chamber conditions. This has been extensively studied in a steady-state context through the use of analytical models [12–14]. These served as initial tools for the analysis of unsteady combustion dynamics, e.g. linearised frequency response [15], intrinsic combustion instabilities, and combustion chamber stability [16].

Unfortunately, these simple models are limited in accuracy, and they are not suited to the detailed study of transient dynamics. Since the 1990s, developments have therefore focused on CFD tools, both for steady and unsteady applications. Recent one-dimensional CFD codes make use of complex chemical mechanisms [17–20], however such kinetics is still difficult to evaluate and validate, and the heterogeneity of widely used propellants, such as AP-HTPB, tends to limit the use of the one-dimensional approach to lower pressures [18], where three-dimensional effects (such as diffusion flames) can be neglected. Detailed two- and three-dimensional CFD approaches taking into account the heterogeneity of the propellant have started to emerge [21–23], however they are costly, even with simplified kinetics, and limited to the simulation of a very small burning area.

These models have been developed to simulate laboratory experiments, for example the burning of a small sample in an enclosed vessel. They can however not be used for larger-scale simulations, such as the computation of the ignition transient in a solid rocket motor. In such configurations, a three-dimensional CFD tool can hardly be used to solve the flow field inside the chamber and the surface combustion, due to extreme computational requirements imposed by the representation of the propellant flame. Indeed, this flame is typically only a few hundred micrometers thick, with strong gradients requiring mesh cells thinner than 1 µm near the surface. Using such a refined mesh along the complete length of combustion chamber (1-10 meters) would be prohibitive. Therefore, the only viable option is to use a coarse mesh near the surface, and solve all the surface combustion within a simpler submodel, e.g. a one-dimensional CFD tool. Such a coupling has been reported with very simplified propellant models [24–26], most often that of an initially inert solid propellant transitioning instantly to quasi-steady burning once its surface temperature becomes larger than a predefined ignition temperature. More accurate one-dimensional models such as the ones discussed earlier may lead to a better resolution of the coupled dynamics, however the numerical strategy must be improved to ensure accuracy, efficiency and stability without requiring prohibitively small time steps.

Eventually, there is a convergent need for a detailed one-dimensional combustion model associated with an efficient numerical strategy, either for coupling with a three-dimensional CFD code, or for detailed parametric studies of flame dynamics in a purely one-dimensional context. This is specifically what we wish to investigate in this paper. It is instructive to conduct a short overview of the various numerical approaches presented in the literature for the time integration of one-dimensional solid propellant combustion models, which is typical of what is described in the general field of low-Mach one-dimensional combustion. One of the earliest detailed one-dimensional model is presented by Erikson and Beckstead [27]. A splitting method is implemented to integrate the gas phase equation, using the ICE scheme [28] to compute the pressure and velocity fields with an implicit scheme of first-order accuracy in time. The stiff chemical source terms are handled with DVODE [29]. The solid phase energy equation is integrated implicitly. The surface temperature and regression speed are then iterated upon until the interface conditions are met, each time performing the split integration of both phases. Due to poor computational performance and large splitting errors, they transition in [30] to a fully implicit resolution of the gas phase, using the TWOPNT [31] algorithm to solve the system discretised in time with the first-order implicit Euler method. Both phases still need to be iterated upon at each time step. The authors mention the attempt to use DASSL [32] instead, a high-order adaptive multistep method, however, following implementation difficulties, they reverted to the earlier first-order strategy. Other researchers used a similar approach with an iterative coupling of both phases (see the review [19]), sometimes with dual-time stepping [33] or substepping [18, 20] to improve convergence of the solid phase. To our knowledge, most one-dimensional unsteady solid propellant combustion codes use a similar iterative coupling procedure between the different phases, with a time integration that is overall first-order accurate in time. Furthermore, the fixed-point approach for the coupling of both phases and their various physics only converges if the time step is sufficiently small, potentially hindering the stability and efficiency of the computation, in particular for highly refined meshes. It has indeed been reported [18] that this iterative approach requires the convective CFL-number be smaller than 1 overall, even though implicit solvers are used to some degree.

The present contribution focuses on the one-dimensional solid propellant combustion in the low-Mach number approximation, and proposes a high-order time integration strategy based on existing singly-diagonally implicit Runge-Kutta methods (up to order 5), with a fully-coupled approach involving an original treatment of the mass

conservation constraint, and ensuring unconditional stability of the overall computation. The proposed approach can be generalized to any one-dimensional low-Mach combustion model for homogeneous or spray flames.

In the first part of the paper, we focus on the mathematical nature of the set of coupled equations obtained after semi-discretisation in space. We show that it is a differential-algebraic system of index one. The knowledge of this particular property is decisive when looking for high-order time integration methods, which is the aim of the second part of this paper. Many such methods are reported in the literature for this particular class of problems[1] [32, 35]. Based on a list of computational and mathematical requirements, we choose a family of singly-diagonally implicit Runge-Kutta methods with embedded lower-order solutions. This provide high-order accuracy, unconditional stability and native time adaptation capabilities. We conduct the numerical analysis of the specific form of the algebraic constraints and introduce an original and efficient way of treating, in particular, the mass conservation constraint on the velocity field. This is a key issue for the efficiency and accuracy of the numerical method. We then present a one-dimensional CFD code utilising this strategy to compute the time-dependent solution of the solid propellant combustion model. Spatial and temporal discretisations are verified with two test cases, for which quasi-analytical solutions are available: steady-state combustion and response to pressure oscillations. In order to investigate the potential and efficiency of the method, we tackle three challenging test cases. First, we focus on the simulation of ignition transients with a simplified modelling, showing that the time adaptation capability of the proposed strategy is more efficient than traditional CFL- or variation-limited time step evaluation strategies commonly used in such applications. Second, we study a solid propellant configuration whose steady-state solution is linearly unstable, leading to a initial instability that stabilises nonlinearly on a limit-cycle periodic solution. Simulation of this configuration shows that high-order time integration methods offer a major improvement over low-order methods, in terms of quality of the result, robustness, and ability to capture the limit cycle, as well as restitution time and ease of simulation setup. Eventually, the proposed numerical strategy is tested on the unsteady combustion of an ammonium perchlorate (AP) monopropellant with detailed gas-phase kinetics. It is shown that the convergence order and efficiency of the method are not affected by the increased complexity and stiffness of the detailed modelling level. Important gains in the quality of the result, ease of simulation setup, and computational times are also observed. Finally, we conclude on the proposed strategy and discuss possible extensions.

2. Formulation of a semi-discrete unsteady solid propellant combustion model

In this section, we first present a one-dimensional model of solid propellant combustion, very similar to the ones found in the one-dimensional low-Mach combustion literature [1–5]. The continuous equations are semi-discretised in space via a method of lines based on a finite-volume scheme.

2.1. General modelling

The temperature profile of the one-dimensional combustion of a solid propellant is presented schematically in Figure 1. The space variable is x. The solid phase represents the propellant and is semi-infinite towards $x = -\infty$. As the surface heats up, thermal decomposition occurs inside the solid. However, due to the high activation energy of this process, the pyrolysis is often assumed to be concentrated in an infinitely thin zone at the surface, while the propellant remains inert. A liquid layer is usually observed at the surface, however due to its low thickness and its difficult experimental characterisation, most models assume this layer to be infinitely thin [21, 36, 37]. The modelled interface is the location of a singularity, where all decomposition and gasification phenomena occur. We adopt the same assumption, however the addition of a liquid phase and other effects (e.g in-depth decomposition reactions) would not affect the considerations discussed in Section 3. Flows near the surface of a solid propellant are generally at high temperatures but low velocity (1-10 m/s), therefore the gas phase is modelled as a low-Mach one-dimensional reactive flow with n_e species and uniform thermodynamic pressure P, which is an input of the model. Radiative emission from the gas phase and surface are neglected. The inert solid phase is simply modelled with the heat equation.

We recall the equations for both phases and their coupling at the interface. They are originally expressed in a Galilean reference frame, where the interface position σ varies in time: $d_t\sigma = -r$ with $r \geq 0$ the surface regression rate. We perform the simple variable change $x = x_{galilean} - \int d_t\sigma dt$, so as to keep the interface at $x = 0$.

[1] It is important to stress the fact that, when the problem is considered in more than one dimension, the system may become differential-algebraic of index two, in particular due to the pressure field [32], requiring a partial reformulation of the problem and the use of more advanced integration techniques [34]. However, the main results of the one-dimensional case should remain applicable.

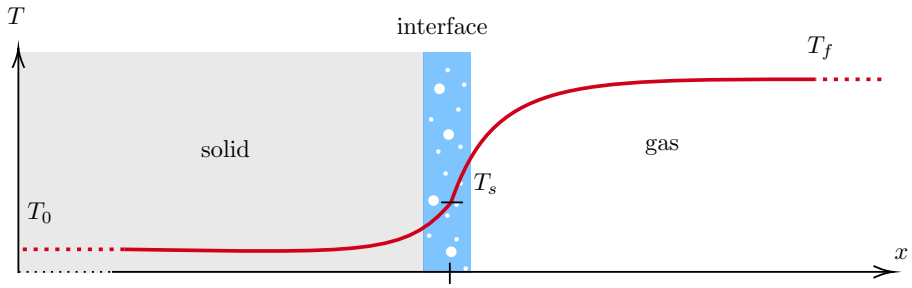


Figure 1 One-dimensional model of solid propellant combustion

This introduces a convective term in the solid phase heat equation, which represents the interface regression. The temperature field T in the solid phase at $x < 0$ is subject to:

$$\rho_c c_c \partial_t T + \rho_c c_c r \partial_x T - \partial_x(\lambda_c \partial_x T) = 0 \tag{1}$$

with ρ_c the propellant density, c_c its heat capacity, λ_c the thermal conductivity. Far below the surface, the solid is at its resting temperature $T(-\infty) = T_0$. The gas phase at $x > 0$ is subject to the following partial differential equations:

$$\begin{cases} \partial_t \rho + \partial_x \rho(v + r) = 0 & (2) \\ \partial_t \rho Y_{\mathfrak{k}} + \partial_x(\rho(v + r)Y_{\mathfrak{k}}) = -\partial_x J_{\mathfrak{k}} + \omega_{\mathfrak{k}} \qquad \forall \mathfrak{k} \in [1, n_e] & (3) \\ \partial_t \rho h + \partial_x(\rho(v + r)h) = -d_t P + \partial_x(\lambda \partial_x T) - \partial_x(\Sigma_1^{n_e} h_{\mathfrak{k}} J_{\mathfrak{k}}) & (4) \end{cases}$$

Here $Y_{\mathfrak{k}}$ is the mass fraction of $\mathfrak{k}$-th species, and $J_{\mathfrak{k}} = -\rho D_{\mathfrak{k}}^{eq} \partial_x Y_{\mathfrak{k}}$ its diffusion flux, where the equivalent diffusion coefficient $D_{\mathfrak{k}}^{eq}$ is evaluated following the Hirschfelder-Curtiss approximation [38]. The volumetric production rate of the $\mathfrak{k}$-th species is $\omega_{\mathfrak{k}}$. The enthalpy h is the sum of the chemical and sensible enthalpies: $h = \Sigma_{\mathfrak{k}=1}^{n_e} Y_{\mathfrak{k}} h_{\mathfrak{k}}$, where $h_{\mathfrak{k}} = (\Delta h_{f,\mathfrak{k}}^0 + \int_{T_0}^{T} c_{p,\mathfrak{k}}(a)da)$, with $c_{p,\mathfrak{k}}$ the heat capacity of the $\mathfrak{k}$-th species, and $\Delta h_{f,\mathfrak{k}}$ its formation enthalpy at T_0. The thermal conductivity is λ. Soret and Dufour effects are neglected. Zero-flux boundary conditions are imposed at $x = \pm\infty$:

$$\partial_x T(-\infty) = 0, \qquad \partial_x T(+\infty) = 0, \qquad \partial_x Y_{\mathfrak{k}}(+\infty) = 0 \qquad \forall \mathfrak{k} \in [1, n_e] \tag{5}$$

Both phases are coupled at the interface by the following conditions, expressing the continuity of the mass flow rate and temperature, as well as the thermal and species flux balance around the interface:

$$\begin{cases} \rho_c r = \rho(0^+)(v(0^+) + r) & (6) \\ T(0^-) = T(0^+) = T_s & (7) \\ (mh - \lambda_c \partial_x T)_{0^-} = (mh + q_r - \lambda \partial_x T + \Sigma_1^{n_e} h_{\mathfrak{k}} J_{\mathfrak{k}})_{0^+} & (8) \\ (mY_{inj,\mathfrak{k}})_{0^-} = (mY_{\mathfrak{k}} + J_{\mathfrak{k}})_{0^+} \quad \forall\ \mathfrak{k}\ \in\ [1, n_e] & (9) \end{cases}$$

with m the mass flow rate ($\rho_c r$ in the solid, $\rho(v + r)$ in the gas), q_r an optional external heat flux (e.g. laser) and Y_{inj} the product mass fractions generated by decomposition and gasification processes, which can be constant or functions of the surface temperature as in [17].

The ideal gas law relates the various state variables in the gas phase:

$$\rho = P / \left(RT \sum_{\mathfrak{k}=1}^{n_e} \frac{Y_{\mathfrak{k}}}{\mathscr{M}_{\mathfrak{k}}} \right) \tag{10}$$

with $\mathscr{M}_{\mathfrak{k}}$ the molar mass of the $\mathfrak{k}$-th species. The surface pyrolysis mass flow rate is given by the pyrolysis law:

$$m(0) = \rho_c r = f(T_s, P) \tag{11}$$

In the following, we denote by u the gas velocity relative to the interface $(v + r)$, and we simplify our notations by using the mass flow rate $m = \rho u$.

Note that, in this one-dimensional framework, the momentum equation is redundant, as the mass flow rate spatial variation is already determined by the continuity equation from the temporal evolution of ρ, which itself is

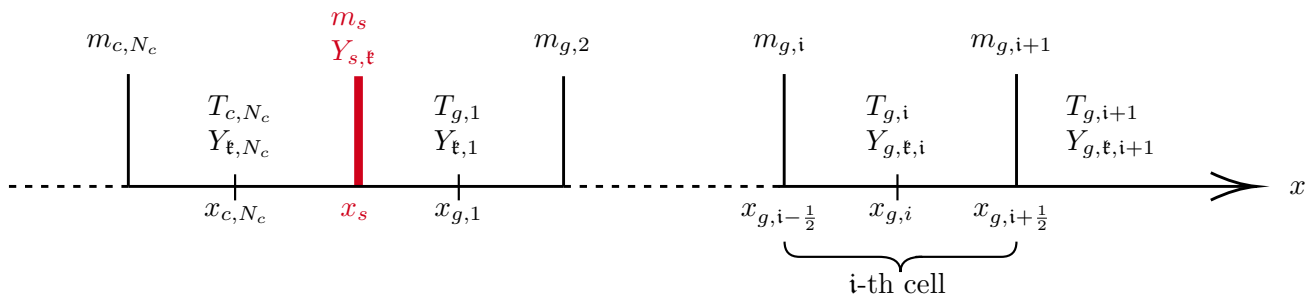


Figure 2 Localisation of the discretised variables in the finite volume mesh. The thick red line represents the surface.

known from the temporal variations of T and $Y_{\mathfrak{k}}$ and the thermodynamic pressure P through the equation of state (10). Hence ρ is not a true variable, although its time derivative is specified by the continuity equation. As we will see further on in Section 3, the continuity equation only acts as a constraint that determines the velocity field. Including the momentum equation would only be necessary for two- or three-dimensional flow models, requiring the hydrodynamic pressure field to be computed for the solution of the velocity field.

2.2. Discretisation with a finite volume approach

For the numerical implementation of a solid propellant combustion model, we apply the method of lines to obtain a set of discrete evolution equations. The semi-discretisation in space is obtained with a finite volume method, however other approaches could be applied without affecting the conclusions drawn in this paper.

2.2.1. Gas phase

The set of conservative equations for the gas phase is semi-discretised in space with a finite volume approach: the domain is split in N_g cells (control volumes). The discretised variables are the temperature T, mass fractions $Y_{\mathfrak{k}}$, mass flow rate m. The temperature and mass fractions are taken at the centers of each cell, while the mass flow rate is taken at the left face of each cell, which is convenient for the free boundary problem with a flux defined at the surface. The surface temperature T_s, the surface mass fraction of the $\mathfrak{k}$-th species $Y_{s,\mathfrak{k}}$ are taken at $x = 0$ (rightmost face of the solid domain, leftmost face of the gas domain). The localisation of each variable is sketched in Figure 2.

Using the notation q to identify any of the conservative variables $\rho Y_{\mathfrak{k}}$ and ρh, or ρ, and the subscript $\mathfrak{i}$ as the index of the mesh cell considered, the conservative equations (2) to (4) become:

$$\Delta x_{\mathfrak{i}} \frac{dq_{\mathfrak{i}}}{dt} = -\left[F_{d,q} + F_{c,q}\right]_{\mathfrak{i}-\frac{1}{2}}^{\mathfrak{i}+\frac{1}{2}} + \Delta x_{\mathfrak{i}} s_{q,\mathfrak{i}} \tag{12}$$

with $\Delta x_{\mathfrak{i}} = x_{\mathfrak{i}+\frac{1}{2}} - x_{\mathfrak{i}-\frac{1}{2}}$ the size of the $\mathfrak{i}$-th cell, $F_{d,q}$ the diffusive fluxes, $F_{c,q}$ the convective fluxes and $s_{q,\mathfrak{i}}$ the chemical source term. Source terms are evaluated at the cell centers. Thermodynamic and transport properties are evaluated at the cell centers, and their values at the interfaces are taken as averages of the adjacent cells values. The gradient at the $(\mathfrak{i} - \frac{1}{2})$-th interface of a variable q discretised at the cell centers is computed as:

$$\nabla q_{\mathfrak{i}-\frac{1}{2}} = \frac{q_{\mathfrak{i}} - q_{\mathfrak{i}-1}}{x_{\mathfrak{i}} - x_{\mathfrak{i}-1}} \tag{13}$$

resulting in a second-order approximation of the diffusive fluxes. The transported interface values at $x_{\mathfrak{i}-\frac{1}{2}}$ are defined as $q_{\mathfrak{i}-\frac{1}{2}} = \Phi^-_{\mathfrak{i}-\frac{1}{2}} q_{\mathfrak{i}-1} + \Phi^+_{\mathfrak{i}-\frac{1}{2}} q_{\mathfrak{i}}$, where $\Phi^-_{\mathfrak{i}-\frac{1}{2}}$ and $\Phi^+_{\mathfrak{i}-\frac{1}{2}}$ are coefficients which sum up to one. Convective fluxes are then computed as $F_{c,q,\mathfrak{i}-\frac{1}{2}} = m_{\mathfrak{i}}\ q_{\mathfrak{i}-\frac{1}{2}}$. The maximal order of accuracy is obtained with both coefficients set to 0.5, resulting in a centered convection scheme. Although unconditionally stable with an appropriate implicit temporal discretisation, the centered scheme may result in a lack of diagonal dominance for the Jacobian used in the Newton algorithm at each step, essentially rendering the linear system solution impossible to perform [39, 40] unless the time step is dramatically reduced. To circumvent this issue, the centered scheme needs to be locally upwinded if the flow is convection-dominated, ensuring diagonal dominance at the expense of falling back to first-order accuracy. This is done dynamically via a Péclet-weighted average of the first-order upwind and second-order centered schemes, similarly to what is done in [41, page 448] with the concept of "mesh Reynolds number". The Péclet number $c_p m/\lambda$ measures the relative importance of diffusive phenomena compared to convective transport in the energy equation. We define a numerical Péclet number at the $(\mathfrak{i} - \frac{1}{2})$-th interface as $\mathrm{Pe}_{\mathfrak{i}-\frac{1}{2}} = (\mathrm{Pe}_{\mathfrak{i}} + \mathrm{Pe}_{\mathfrak{i}-1})\,(x_{\mathfrak{i}} - x_{\mathfrak{i}-1})\,/2$, i.e. it is

an average Péclet number with a reference length taken as the distance between the centers of the neighbouring cells. If $|\mathrm{Pe}_{\mathfrak{i}-\frac{1}{2}}| < 0.5$, we use the centered scheme: $\Phi^+ = \Phi^- = 0.5$, i.e. the centered scheme is used as the thermal diffusion dominates energy convection. If $|\mathrm{Pe}_{\mathfrak{i}-\frac{1}{2}}| > 1$, we use the upwind scheme: $\Phi^+ = 0$ if $\mathrm{Pe} > 0$, else $\Phi^+ = 1$. The transition between these two cases is smooth with respect to $\mathrm{Pe}_{\mathfrak{i}-\frac{1}{2}}$, so as not to cause numerical issues later on. Overall, the second-order scheme is used where the mesh is sufficiently refined, while the first-order scheme is only used in poorly resolved areas, typically far away from the surface, where precise representation of the flow is not required. The final semi-discrete conservative equations in the $\mathfrak{i}$-th cell read:

$$\Delta x_{\mathfrak{i}} \frac{d\rho_{\mathfrak{i}}}{dt} = m_{\mathfrak{i}} - m_{\mathfrak{i}+1} \tag{14}$$

$$\begin{aligned} \Delta x_{\mathfrak{i}} \frac{d\rho_{\mathfrak{i}} Y_{\mathfrak{k},\mathfrak{i}}}{dt} &= m_{\mathfrak{i}} \left(\Phi^+_{\mathfrak{i}-\frac{1}{2}} Y_{\mathfrak{k},\mathfrak{i}} + \Phi^-_{\mathfrak{i}-\frac{1}{2}} Y_{\mathfrak{k},\mathfrak{i}-1} \right) - m_{\mathfrak{i}+1} \left(\Phi^+_{\mathfrak{i}+\frac{1}{2}} Y_{\mathfrak{k},\mathfrak{i}+1} + \Phi^-_{\mathfrak{i}+\frac{1}{2}} Y_{\mathfrak{k},\mathfrak{i}} \right) \\ &+ J_{\mathfrak{k},\mathfrak{i}-\frac{1}{2}} - J_{\mathfrak{k},\mathfrak{i}+\frac{1}{2}} + \Delta x_{\mathfrak{i}} \omega_{\mathfrak{k},\mathfrak{i}} \quad \forall \mathfrak{k} \in [1, n_e] \end{aligned} \tag{15}$$

$$\begin{aligned} \Delta x_{\mathfrak{i}} \frac{d\rho_{\mathfrak{i}} h_{\mathfrak{i}}}{dt} &= \Delta x_{\mathfrak{i}} d_t P + m_{\mathfrak{i}} \left(\Phi^+_{\mathfrak{i}-\frac{1}{2}} h_{\mathfrak{i}} + \Phi^-_{\mathfrak{i}-\frac{1}{2}} h_{\mathfrak{i}-1} \right) - m_{\mathfrak{i}+1} \left(\Phi^+_{\mathfrak{i}+\frac{1}{2}} h_{\mathfrak{i}+1} + \Phi^-_{\mathfrak{i}+\frac{1}{2}} h_{\mathfrak{i}} \right) \\ &- \frac{\lambda_{\mathfrak{i}-1} + \lambda_{\mathfrak{i}}}{2} \frac{T_{\mathfrak{i}} - T_{\mathfrak{i}-1}}{x_{\mathfrak{i}} - x_{\mathfrak{i}-1}} + \frac{\lambda_{\mathfrak{i}} + \lambda_{\mathfrak{i}+1}}{2} \frac{T_{\mathfrak{i}+1} - T_{\mathfrak{i}}}{x_{\mathfrak{i}+1} - x_{\mathfrak{i}}} \\ &+ \sum_{\mathfrak{k}=1}^{n_e} \left(\frac{h_{\mathfrak{k},\mathfrak{i}} + h_{\mathfrak{k},\mathfrak{i}-1}}{2} J_{\mathfrak{k},\mathfrak{i}-\frac{1}{2}} - \frac{h_{\mathfrak{k},\mathfrak{i}+1} + h_{\mathfrak{k},\mathfrak{i}}}{2} J_{\mathfrak{k},\mathfrak{i}+\frac{1}{2}} \right) \end{aligned} \tag{16}$$

2.2.2. Solid Phase

The solid phase energy equation (1) is replaced by a conservative equation for the enthalpy and discretised in the same way as the conservative equations in the gas phase. The solid mesh contains N_c cells. As explained further in Section 4.2, having a block-tridiagonal Jacobian for the complete system is computationally efficient. In order to keep a consistent Jacobian structure between the gas and solid phases, the mass flow rate field m and the species profile $Y_{\mathfrak{k}}$ are kept as dummy variables. As the solid propellant is assumed inert and incompressible, the continuity equation is equivalent to $d_x m = 0$ which is discretised as $m_{\mathfrak{i}} = m_{\mathfrak{i}+1}$, with the boundary condition $m_{N_c+1} = m(T_s, P)$. Species evolution is not considered in the solid phase, therefore the simple equation $Y_{\mathfrak{k},\mathfrak{i}} = 0$ is used. Due to the similarity in structure between the gas and solid phases semi-discrete systems, we assume when necessary in the rest of the article that the solid phase equations are contained in the previous gas phase system.

2.2.3. Surface coupling conditions

The surface variables are the surface temperature T_s and the surface mass fractions $Y_{s,\mathfrak{k}}$ on the gas side. The surface matching conditions (6), (8) and (9) and the continuity equation (2) are discretised as follows, with a first-order approximation of the gradients:

$$\begin{cases} 0 = g_{th} := -\lambda_c \dfrac{T_s - T_{c,-1}}{x_s - x_{c,1}} + \lambda \dfrac{T_{g,1} - T_s}{x_{g,1} - x_s} + m \left(h(T_s, Y_{s,\mathfrak{k}}) - h(T_s, Y_{inj,\mathfrak{k}}) \right) + \Sigma_1^{n_e} h(T_s, Y_{inj,\mathfrak{k}}) J_{s,\mathfrak{k}} & (17) \\ 0 = g_{sp,\mathfrak{k}} := m(T_s)(Y_{inj,\mathfrak{k}} - Y_{s,\mathfrak{k}}) + J_{s,\mathfrak{k}} \qquad \forall \mathfrak{k} \in [1, n_e] & (18) \end{cases}$$

with $T_{c,-1}$ the temperature in the last cell of the solid phase below the surface, and $T_{g,1}$ the temperature in the first cell of the gas phase, just above the surface. The species surface diffusion fluxes are computed as:

$$J_{s,\mathfrak{k}} = \rho(T_s, Y_s, P) D^{eq}_{\mathfrak{k}}(T_s, Y_s, P) \frac{Y_{g,\mathfrak{k},1} - Y_{s,\mathfrak{k}}}{x_{g,1} - x_s} \tag{19}$$

2.3. Final semi-discrete form

Let W be the vector of the discrete variables T (solid and gas) and $Y_{\mathfrak{k}}$ in each cell, and let Z be the vector containing the remaining variables T_s, $Y_{s,\mathfrak{k}}$ and m: $W = (Y_{\mathfrak{k}}, T)^t$ and $Z = (T_s, Y_{s,\mathfrak{k}}, m)^t$. The semi-discretisation in space yields the following system:

$$\begin{cases} d_t Q(W) = f(W, Z) & (20) \\ 0 = g(W, Z) & (21) \end{cases}$$

with $Q(W) = (\rho Y_{\mathfrak{k}}, \rho h)^t$ the vector of conserved differential variables, f the corresponding spatial operator, i.e. the right-hand side of Equations (15) and (16), and g the vector obtained by concatenating Equations (17), (18),

(6) and (14). The continuity equation (14) is not included in Equation (20), but rather in Equation (21), as will be explained later in Sections 3.1 and 4.6. A more classical form can be obtained if we were to reformulate the conservation equations so that they directly give the temporal derivatives of W, not of $Q(W)$:

$$\begin{cases} d_t W &= f_1(W, Z) & (22) \\ 0 &= g(W, Z) & (23) \end{cases}$$

We will use the simpler first form for the numerical implementation, and we will mostly use the second formulation for theoretical discussions (dropping the subscript 1 for f).

3. Differential-algebraic nature of the semi-discretised system

We aim at developing a one-dimensional unsteady CFD tool for high-fidelity simulations of transient phenomena. Relying on a finite-volume space discretisation, we have obtained a system of semi-discrete evolution equations. In this section, we show that a difficulty arises from the nature of this system: some variables are not defined by differential equations but by algebraic ones. The system thus belongs to the class of Differential-Algebraic Equations (DAE). We refer the reader to [32, 35] for details on DAEs and only the necessary aspects of this class of problem will be discussed hereafter.

3.1. Identification of the constraints

The discretised surface variables $Y_{s,\mathfrak{k}}$ and T_s only appear in equations (17) and (18), however no time derivative appear. Such variables are called algebraic and will "instantly" adapt to the variations of the other variables in the cells adjacent to the surface, i.e. they are not directly affected by their time histories. Note that this characteristic is linked to the assumption that no accumulation of energy or mass takes place at the interface, hence the latter has no inertia and its associated variables must always be such that the interface conditions are met.

The remaining algebraic equations come from the discrete gas continuity equation (14), which we recall here:

$$\begin{cases} 0 = g_{m,1} := m_1 - m(T_s) & (24) \\ 0 = g_{m,\mathfrak{i}} := \dfrac{d\rho_{\mathfrak{i}-1}}{dt} + \dfrac{m_{\mathfrak{i}} - m_{\mathfrak{i}-1}}{\Delta x_{\mathfrak{i}}} \qquad \forall \mathfrak{i} \in [2, N_g] & (25) \end{cases}$$

Equation (24) is the boundary condition for the gaseous mass flow rate field and is equivalent to Equation (6). As underlined in Section 2.1, the density $\rho_{\mathfrak{i}}$ in the $\mathfrak{i}$-th cell is a function of $T_{\mathfrak{i}}$ and $Y_{\mathfrak{k},\mathfrak{i}}$ via the ideal gas law (10). These variables are governed by the discrete equations (15) and (16) and by the pressure P. Therefore $\rho_{\mathfrak{i}}$ is not a true differential variable. Its time derivative $d_t\rho_{\mathfrak{i}}$ appearing in Equation (25) is entirely determined from the variations of the other gas-phase variables $T_{\mathfrak{i}}$ and $Y_{\mathfrak{k},\mathfrak{i}}$. The continuity equation is solely used to constrain the flow rate field in the gas phase m, the time derivative of which does not appear in this one-dimensional low-Mach model. As a consequence, the discrete values of the mass flow rate are also algebraic variables, which adapt instantly to variations of the other variables in coherence with the parabolic nature of the low-Mach number limit, where all pressure waves propagate at infinite speed and are relaxed instantly. This situation is generic in low-Mach number combustion modelling.

3.2. Index of the algebraic equations

The presence of algebraic equations usually brings in various types of additional difficulties in the numerical resolution of the system of equations. A useful mathematical criterion, the "index", is used to classify the different types of algebraic equations and is essential in order to chose the proper numerical method. It is defined as the number of times a given algebraic equation must be differentiated with respect to time to obtain a differential equation for the corresponding algebraic variable [32].

Deriving our discretised Equations (17), (18), (24) and (25) with respect to time, terms in d_tT_s, $d_tY_{s,\mathfrak{k}}$ and $d_tm_{\mathfrak{i}}$ appear, and manipulations lead to explicit expressions for each of them. Apart from exceptional cases which we never encountered in practice, the denominators in these expressions will never be zero, thus the obtained ODEs are well-posed and the index of the original DAE system is 1. Note that when differentiating Equation (25) with respect to time, the second-derivative $d_{tt}\rho_{\mathfrak{i}}$ appears. It can however be expressed by differentiating the ideal gas law and the other conservation equations.

Another approach is to directly use System (22) and (23), also known as *semi-explicit form* for DAE problems, where the separation between the differential and algebraic variables W and Z, respectively, leads to a direct

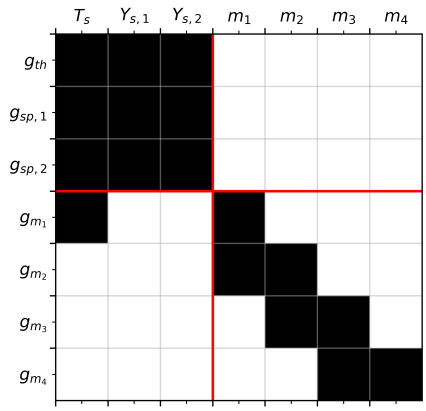


Figure 3 Sparsity pattern of the Jacobian of g, the vector of constraints

identification of the constraints $g = (g_{th}, g_{sp,1} \dots g_{sp,n_e}, g_{m_0} \dots g_{m_{N_g}})^t$. In this form, the index of the DAE is 1 if the Jacobian $\partial_Z g$ is non-singular [35]. We can verify this by forming this Jacobian. For the sake of simplicity, we show in Figure 3 its sparsity pattern when considering only 2 species. The labels on the vertical axis describe the constraint being derived, and the labels on the horizontal axis denote the differentiation variable. We can decompose the matrix into smaller blocks following the red lines. The first block on the diagonal corresponds to the Jacobian of the nonlinear system $(g_{th}, g_{sp,1} \dots g_{sp,ne})^t = 0$. The second block on the diagonal is lower triangular with non-zero elements on the diagonal, therefore it is invertible.

In the simplified case where only 2 species of equal properties are considered with a unitary Lewis number, and assuming $c_p = c_s$, we have been able to show that g_{th} and g_{sp} can be formulated such that the first block is lower-triangular with diagonal terms which are similar to the denominators previously discussed, and thus only zero in specific cases which will never be encountered in practice. For instance, under the previous assumptions, the terms in involving species diffusion and enthalpy in equation (8) for g_{th} can be gathered under the form mQ_p, with Q_p the heat associated to the pyrolysis and gasification processes at the surface, which is a constant under the previous assumptions. The diagonal term $\partial_{T_s} g_{th}$ is $1/(\lambda_c/\Delta x_{c,N_c} + \lambda/\Delta x_{g,1} - Q_p \partial_{T_s} m)$, and similar expressions can be obtained for the other diagonal terms. The term $\partial_{T_s} m$ being positive and Q_p potentially being positive as well, there is no guarantee that the denominator will not cancel. However, this term can be bounded in magnitude so that, for asymptotically fine meshes, it becomes negligible, and therefore the block is ensured to invertible.

A thorough demonstration of the invertibility of this block in the general case seems out of reach, owing to the large number of nonlinear terms that appear via the diffusion fluxes, enthalpy expressions, and the pyrolysis law mostly. However, our numerical experience shows that the Jacobian $\partial_Z g$ is indeed invertible in all studied cases. Overall we claim that the Jacobian is invertible, hence the index of the corresponding algebraic equations is 1.

4. Requirements for the time integration method

Now that we have determined that our semi-discretised problem is an index-1 differential-algebraic system of equations, we look for a high-order time integration method to maximise the performance of our code, while ensuring high-accuracy. Adaptive time stepping is an additional capability that we seek, so as to minimise the number of time steps and thus the computational time, while also guaranteeing a prescribed solution accuracy. We focus on Runge-Kutta methods and shortly discuss the applicability of other methods.

4.1. Ensuring high-order convergence for DAEs

4.1.1. Formulation of Runge-Kutta methods for DAEs

A generic s-stage Runge-Kutta integration method applied to Equations (22) and (23) yields [35]:

$$\begin{cases} w_{ni} &= W_n + \Delta t \sum_{j=1}^{s} a_{ij} f(w_{nj}, z_{nj}) & (26) \\ 0 &= g(w_{ni}, z_{ni}) & (27) \\ W_{n+1} &= w_n + \Delta t \sum_{i=1}^{s} b_i f(w_{ni}, z_{ni}) & (28) \\ Z_{n+1} &= (1 - \sum_{i,j=1} b_i \omega_{ij}) Z_n + \sum_{i,j=1}^{s} b_i \omega_{ij} z_{nj} & (29) \end{cases}$$

where w_{ni} and z_{ni} represent the values of W and Z at stage i (time $t_{ni} = t_n + c_i \Delta t$), and a_{ij}, b_i and c_i are the coefficients of the Runge-Kutta method, and ω_{ij} are the coefficients of the inverse of $A = (a_{ij})$, which we assume to exist, i.e. we temporarily restrict ourselves to implicit methods. At each stage, the algebraic variables Z are determined via Equation (27) such that the constraints are satisfied. This can be interpreted as a systematic projection of the algebraic variables onto the set $\{z \mid g(W, z) = 0\}$. After all stages are computed, the advancement to the next time step is performed via Equations (28) and (29).

Equation (29) which does not necessarily ensure that $g(W_{n+1}, Z_{n+1}) = 0$, hence a deviation from the correct solution may occur. Runge-Kutta methods that are *stiffly accurate* satisfy the following conditions: $W_{n+1} = w_{ns}$, $Z_{n+1} = z_{ns}$, i.e. the last stage is the solution at the next time step. With such methods, we directly obtain $0 = g(W_{n+1}, Z_{n+1})$, and Equation (29) becomes redundant. Non-stiffly accurate methods have their order of convergence severely reduced or might even become unstable when applied to index-1 DAEs [35], whereas stiffly accurate methods retain the same order of convergence as for ODEs. Therefore such methods are of particular interest for our problem. Implicit Runge-Kutta methods also allow for a natural treatment of the mass conservation constraint, preserving the order of convergence on both differential and algebraic variables, as will be discussed in Section 4.6. Finally, fine meshes and chemical kinetics often produce stiff systems, for which L-stability is a very advantageous property. It ensures a stable solution by instantly relaxing modes with time scales much shorter than the time step.

4.1.2. Examples

The most widely used implicit method is implicit Euler (or backward Euler), which is single-stage, first-order, stiffly accurate and L-stable. Applied to our system, it yields:

$$\begin{cases} W_{n+1} &= W_n + \Delta t f(W_{n+1}, Z_{n+1}) & (30) \\ 0 &= g(W_{n+1}, Z_{n+1}) & (31) \end{cases}$$

A classical second-order scheme is the Crank-Nicolson method, which is not L-stable:

$$\begin{cases} W_{n+1} &= W_n + \frac{\Delta t}{2} \left(f(W_{n+1}, Z_{n+1}) + f(W_n, Z_n) \right) & (32) \\ g(W_{n+1}, Z_{n+1}) &= -g(W_n, Z_n) & (33) \end{cases}$$

Equation (33) shows that this method is sensitive to error accumulation on the algebraic variables. In particular, it is crucial that the initial condition satisfies $g(Y_0, Z_0) = 0$.

4.2. Optimising the computational cost

When advancing forward in time, Equations (26) and (27) must be solved, usually via a Newton algorithm, which iterates on the values w_{ni} and z_{ni} for $i \in [1, s]$. Fully implicit Runge-Kutta methods are such that all the stages must be solved simultaneously. The very popular stiffly accurate method Radau5 [35], a 3-stage fifth-order fully implicit method based on Gauss-Radau quadrature points, is one such method. It possesses very interesting properties, however if the problem has N unknowns, each time step requires solving a $3N \times 3N$ system, which can be rather costly. An appealing subclass of Runge-Kutta methods is that of diagonally-implicit Runge-Kutta methods (DIRK) [42]. These methods are such that the summations in Equations (26) and (27) for the i-th stage

only go up to i instead of s, i.e. the stages can be solved sequentially. Such methods require more stages (typically twice as many) to reach the same order of convergence as fully implicit methods, however a complete time step only requires the resolution of s systems of size $N \times N$, which is usually more computationally efficient. Therefore we narrow down our choices to DIRK methods.

For each stage we need to solve Equations (26) and (27) for the unknowns w_{ni} and z_{ni}. These equations can be combined to form the nonlinear problem $F(X) = 0$, with $X = (w_{ni}, z_{ni})^t$ the vector of unknowns. This problem is solved iteratively using a quasi-Newton method. At each Newton step, the Newton increment ΔX is obtained by solving the linear system $J\Delta X = -F(X)$, with $J = \partial_X F$ the jacobian of the residual vector. A more detailed description of the structure of J is given in Appendix D. It is computed by finite differences and is only updated when convergence is poor. We did not implement a way to evaluate and store the separate Jacobians of f, Q and g which appear in J, hence the latter cannot be simply refactored upon time step changes. In the case of nearly linear dynamics, this can be disadvantageous. This matter will be discussed in Section 7.3.3.

Following Section 2.2, in particular the three-point stencil of the spatial discretisation and the ordering of our state vectors (variables sorted by cell), J is block-tridiagonal and a Thomas algorithm can be used, after having performed a LU-decomposition. If the diagonal coefficients a_{ii} of the method are not equal, J needs to be updated at each stage. Therefore, we focus on singly-diagonally implicit Runge-Kutta methods (SDIRK), which satisfy the property $a_{ii} = a_{jj}\ \forall\ (i,j) \in [1,s]$.

4.3. Time adaptation

A typical ignition transient is shown in Figure 6 for a solid propellant ignited by a laser source. The evolution of the surface temperature is very rapid at the beginning of the propellant heating and around the time of ignition. The ignition transient can clearly be split into successive phases with very different time scales: inert heating, ignition *per se* and stabilisation onto steady-state. Consequently, ensuring the time step is dynamically adapted is important to improve accuracy and lower the computational time.

Some simulations presented in the literature use a constant time step, taken as sufficiently low compared to the ignition time [27, 43]. When time adaptation is used, it usually relies on a CFL criterion [18]. CFL limitation may be irrelevant before ignition, as the gas-phase flow velocity is negligible, therefore it may be supplemented with an additional requirement that the relative solution variation between two successive time steps is sufficiently small (e.g. 1%). However such an approach requires fine-tuning and does not provide any guarantee regarding the accuracy of the solution. So-called *embedded Runge-Kutta methods* provide a local error estimate by comparing two solutions X_{n+1} and $\hat{X}_{n+1}$ of different orders p and $\hat{p} < p$, which are constructed to share as many internal stages as possible. This allows to cheaply obtain an objective estimate $\epsilon(\Delta t)$ of the integration error that scales as $\mathcal{O}(\Delta t^{\hat{p}+1})$. We can the compute a normalised error $err(\Delta t) = ||\epsilon(\Delta t)/(atol + rtol|X_n|)||$, where the division is performed element by element, and $atol$, $rtol$ are user-specified absolute and relative error tolerances. In the present article, we use the 2-norm. Requiring $err(\Delta t) < 1$ yields an explicit way to determine an optimal time step Δt_{opt} that satisfies this requirement [44]: $\Delta t_{opt} = \Delta t.err(\Delta t)^{-1/(q+1)}$.

If Δt is such that the asymptotic regime of convergence for the integration method is already reached, then the estimated local integration error for Δt_{opt} will be close to the prescribed tolerance. If the estimated error is larger than the tolerance, the current step is restarted with the new (smaller) time step Δt_{opt}. Otherwise, it is accepted and the next step is computed with the new optimal time step length Δt_{opt}. This allows the time step to be dynamically reduced or increased, ensuring that the solution is solved at least as precisely as specified, while minimising computational cost. In practice, to avoid over-correcting the time step, we do not allow it to change by a factor lower than 0.2 or higher than 5 between two successive error estimations. A safety factor of 0.9 is also applied to Δt_{opt} to ensure the tolerance is strictly satisfied. If a time step fails due to floating arithmetic errors or results in non-convergence of the Newton iterates, the current step is started over again with a decreased step length. Application of a dead zone, i.e. not changing the time step if its required relative variation is below a certain threshold, helps lowering the number of time step changes and Jacobian updates, however it generally leads to worse performance overall in our test cases.

4.4. Final choice of the method

Considerations on the accuracy of the method for index-1 DAEs have led us to consider stiffly accurate Runge-Kutta methods. Minimisation of the computational cost due to Newton iterations is ensured by using singly-diagonally implicit methods, and embedded methods allowing for robust time adaptation are favoured. Additionally, improvements in the error estimation for stiff system and DAEs can be obtained if the lower-order embedded method is stiffly accurate as well, as discussed in [45]. This reference introduces several such schemes with an additional interesting property, which is that the first stage is the last stage of the previous step. This allows to have a

name	number of stages	L-stable	order
ESDIRK-54A	7 (6)	yes	5
ESDIRK-43B	5 (4)	yes	4
ESDIRK-32A	4 (3)	yes	3
CKN (Crank-Nicolson)	2 (1)	no	2
IE (Implicit Euler)	1	yes	1

Table 1 Selected Runge-Kutta methods. The numbers in brackets correspond to the number of stages actually solved.

"free" stage to improve the accuracy of the method without any additional cost. We retain three schemes from this reference: ESDIRK-32A, ESDIRK-43B and ESDIRK-54A. For comparison with more classical schemes, we also include backward Euler and Crank-Nicolson. All methods and their properties are listed in Table 1.

4.5. Alternative choices

From a practical point of view, if an existing code uses an implicit Euler or Crank-Nicolson scheme, as is often the case in the literature, the implementation of ESDIRK methods is straightforward, as it only requires solving more stages. This is also the case if the original code uses a multistep BDF method. Once the ESDIRK schemes are properly implemented, adding time adaptation is also straightforward.

Other methods may however be of interest. Multistep methods can be applied directly to the semi-explicit form of the system. In particular the DASSL algorithm [32] has been extensively applied to index-1 DAEs. However, it is well known that their stability properties degrade for orders higher than 2. Rosenbrock methods [35] are similar to SDIRK methods, but are formulated such that only linear systems need to be solved at each stage. It is however possible that strong system nonlinearities cause severe time step restrictions [46], and these methods may also be less precise with regards to the satisfaction of the algebraic constraints [35].

Another possibility is to take advantage of the fact that the DAE system is of index 1, thus the algebraic variables can be uniquely determined from the differential variables, i.e. we can consider that there exists a function p such that $Z = p(W)$ satisfies $g(W, Z) = 0$. Usually this relation is not explicitly known and Z may instead be iteratively determined via a Newton method. The differential variables W are then governed by the ODE $d_t W = f(W, p(W))$, which can be integrated with any ODE solver, in particular explicit ones. This approach is also referred to as *state-space form* [35]. In practice, we have observed that our fully implicit approach is able to produce accurate results while having CFL numbers much higher than 1, therefore explicit integration algorithms would be relatively inefficient due to their stability limitations. This can be overcome by using an implicit algorithm to integrate f, however the cost of solving for the algebraic variables at each evaluation of f may become prohibitive. All operators (convection, diffusion, reaction) are stiff, thus partially implicit algorithms, e.g. IMEX methods [47], are not suited.

Splitting methods could be used so that each operator is integrated with an adequate efficient method, however the order of accuracy in time would generally not exceed 2. Additional difficulties may appear when handling the interface conditions, and time step adaptation is more involved compared to embedded methods [48]. As already mentioned, fully implicit methods like Radau5 [35] could be used, however the linear algebra becomes more complex as larger systems need to be solved. An interesting alternative could be the parallel DIRK iterations of a fully implicit Runge-Kutta method [49], where multiple stages can be solved in parallel to gradually approach the solution of the fully implicit method.

Finally, it should be pointed out that it is difficult to use existing solvers directly. For the sake of clarity we used the classical form (22) for the differential part in this section, but Equation (20) actually prescribes the time derivatives of the conserved variables $Q(W) = (\rho Y_{\mathfrak{k}}, \rho h)$. Thus, the differential part is of the form $d_t Q(W) = f(W, Z)$, which is not directly compatible with existing ODE/DAE solvers. A first remedy would be to rewrite the conservation equations so that they directly give $d_t Y_{\mathfrak{k}}$ and $d_t T$, in a similar manner to Appendix B.1. This would yield Equations (22) and (23). Another solution is to discretise the conserved variables instead, and use a Newton method at each call of f to compute T, Y_k from $Q = (\rho, \rho Y_{\mathfrak{k}}, \rho h)^t$. Thus, Equation (20) would implicitly take the form $d_t Q = f(W(Q), Z)$. Both solutions would add complexity to the code and are therefore not used in the rest of the article. Also, since the continuity equation (14) gives the time derivative of ρ which is not one of our discretised variables, this Equation would need to reformulated as in Section 4.6.2, so as to transform it into a constraint of the $0 = g_m(W, Z)$. However, as discussed in the next section, this reduces the computational efficiency.

4.6. Handling the continuity equation

In our one-dimensional low-Mach approach, the density ρ cannot be considered a true variable of our problem, as it is uniquely determined from the temperature, mass fractions and pressure. Consequently, the continuity equation

(14) should not be considered as an ODE on ρ, but rather as a constraint on the mass flow rate field. We now focus on the handling of this specific problem. In particular, we show that a Runge-Kutta formulation can be used in order to provide an original and natural treatment of the mass flow rate constraint.

4.6.1. General formulation of our DAE problem

We previously used Equations (22) and (23) to represent our complete semi-discrete system. The continuity equation (14) was included in Equation (21). We can instead use the following more detailed formulation to highlight the particular characteristic of this equation:

$$\begin{cases} \partial_t y = f(y, z_1, z_2) & (34) \\ 0 = \partial_t \phi(y) + g_1(y, z_1, z_2) & (35) \\ 0 = g_2(y, z_2) & (36) \end{cases}$$

with index-1 constraints g_2 and, in our case, $y = (T, Y_k)$, $z_1 = (\rho u)$, $z_2 = (T_s, Y_{s,k})$, and $\phi(y) = \rho$ as per the equation of state. The constraint (35) corresponds to the semi-discrete continuity equation (25), while the constraint (36) gathers the interface coupling conditions (18), (17) and (24). Equation (34) contains the remaining differential equations on ρh, ρY_k, i.e. the semi-discrete equations (15) and (16). This form clearly shows that the time derivative appearing in (35) is not that of any of the discretised variables y, z_1 or z_2. Hence, it is impossible to directly apply existing DAE solvers to this system, as discussed in the previous section. Multiple time discretisation approaches for this particular equation can be envisioned and are described next.

4.6.2. Instantaneous reformulation of the first constraint

Equation (35) can be transformed so that the time derivative of ϕ, which is not a true variable of our system, does not appear. Writing: $\partial_t \phi(y) = (\partial_y \phi) d_t y$, Equation (35) can be replaced by:

$$0 = (\partial_y \phi) f(y, z_1, z_2) + g_1(y, z_1, z_2) \tag{37}$$

First, we may apply a Runge-Kutta scheme on (34) to compute the evolution of y. At each evaluation of f, z_2 is obtained by solving Equation (36), then z_1 is computed by solving Equation (37). Thus, f can be evaluated. The detailed discrete equations for our solid propellant system are derived in Appendix B.1.

This state-space form is often used in the literature for fractional-step approaches, in particular in low-Mach or incompressible flow models to compute the perturbed pressure field by solving a Poisson equation to ensure the velocity field remains divergence-free [50–52]. However in our 1D case, as discussed in Section 4.4, we believe it is best to solve for the algebraic and differential variables at the same time, i.e. use a fully-coupled implicit approach, since the cost of solving the velocity field is small compared to that of the species and energy.

4.6.3. Use of the Runge-Kutta temporal quadrature

Equation (35) constitutes an ODE on $\phi(y_1)$, which is not a true variable of our problem, but is computed from the other ones via the equation of state. We can nonetheless apply the Runge-Kutta scheme to this equation. We then obtain, for the i-th stage of a DIRK method:

$$\begin{cases} Y_{ni} = y_n + \Delta t \sum_{j=1}^{i} a_{ij} f(Y_{nj}, Z_{1,nj}, Z_{2,nj}) & (38) \\ \phi(Y_{ni}) = \phi(y_n) + \Delta t \sum_{j=1}^{i} a_{ij} g_1(Y_{nj}, Z_{1,nj}, Z_{2,nj}) & (39) \\ 0 = g_2(Y_{ni}, Z_{2,ni}) & (40) \end{cases}$$

Equation (39) constitutes an algebraic constraint on $Z_{1,ni}$. The detailed equations are derived in Appendix B.

4.6.4. Comparing both approaches

We can interpret the previous formulation as performing a quadrature on the continuity equation, i.e. approximate the integral in the exact solution:

$$\phi(Y_{ni}) = \phi(y_n) + \int_{t_n}^{t_n + c_i \Delta t} g_1(y(t), z_1(t), z_2(t)) dt \tag{41}$$

The quadrature is the same as the one used for the ODE (34) and has the same order of accuracy in time. On the other hand, the instantaneous reformulation yields:

$$0 = (\partial_y \phi)(Y_{nj}) f(Y_{nj}, Z_{1,nj}, Z_{2,nj}) + g_1(Y_{nj}, Z_{1,nj}, Z_{2,nj}) \tag{42}$$

The quadrature performed in Equation (39) introduces an error $O(\Delta t^p)$ with p the accuracy order of the stage considered (2 for internal stages of ESDIRK methods, and the overall order of the method for the last stage of a stiffly accurate method). An error of the same order is introduced in the instantaneous formulation (42) through Y_{nj} and $Z_{2,nj}$.

It was assessed on the test cases of this article that both approaches yield virtually identical results, in terms of dynamics, time step evolution, orders of convergence in time and space. Still, the instantaneous formulation yields a more nonlinear system because of the term $\partial_y \phi$. Therefore, convergence of the Newton algorithm is more challenging and we have found that, in many cases, this results in a dramatic increase in the number of Jacobian updates, roughly 30% more in highly dynamic simulations (ignition), and up to a factor 10 in certain situations (near steady-state AP combustion from Section 8), substantially slowing down the computation. Thus our original approach is more robust and performant, while also being easier to derive and implement. To our knowledge, the Runge-Kutta quadrature approach has not yet been presented in the context of one-dimensional low-Mach flows, and we claim that it is both simpler and more efficient than the traditional instantaneous reformulation. All the results presented in this paper were obtained with this approach.

5. Numerical verification

The previous numerical strategy has been implemented in the VULC1D Fortran code. In this section, we verify the spatial and temporal discretisations by comparing the results to those obtained with other approaches.

5.1. Steady-state solution

In the case of temperature-independent properties in both phases and unitary Lewis number in the gas phase, we have developed a semi-analytical tool [53], which solves the complete problem in steady-state with arbitrary precision. It can be used as reference to assess the steady-state solution produced by our CFD code. This comparison has been presented in details in [53] and we only summarise the most important aspects here for the sake of completeness.

5.1.1. Model parameters

We consider a simple one-dimensional model of an AP-HTPB-Al propellant, whose main characteristics are summarised hereafter.

Solid phase. The solid phase is composed of the solid species P and has the following properties: $\rho_c = 1806$ kg.m^2, $\Delta h_f^o(P) = 0$ J/kg at $T = 0$ K, $c_c = 1253$ J/kg/K, $\lambda_c = 0.65$ W/m/K. The initial temperature is $T_0 = 300$ K.

Surface. The pyrolysis mass flow rate is computed as: $m = A_p \exp(-T_{ap}/T_s)$, with $A_p = 6.07 \times 10^7$ kg/s/m^2 and $T_{ap} = 15082$ K. The pyrolysis process converts the solid phase into the gaseous species G_1.

Gas phase. Two global species are considered: the reactant G_1 and product G_2, which have the same properties except standard enthalpies. Their molar mass is $\mathscr{M} = 74$ g/mol, and their heat capacities are $c_p = c_c$. The standard enthalpies at $T = 0$ K are $\Delta h_f^o(G_1) = -1.80 \times 10^5$ J/kg and $\Delta h_f^o(G_2) = -4.06 \times 10^6$ J/kg. The unique global reaction is $G_1 \rightarrow G_2$ and is irreversible. The reaction rate is computed as: $\omega = A[G_1] \exp(-T_a/T)$, with $A = 435.5$ s^{-1}, $T_a = 7216$ K, and $[G_1]$ the concentration of G_1. The diffusion coefficients are equal for both species and taken as a linear function of T such that the Lewis number is one throughout the gas phase. The thermal conductivity is $\lambda = 0.464$ W/m/K.

5.1.2. Verification process and results

First, the complete steady-state problem is solved with the semi-analytical tool. Multiple meshes are then generated for the one-dimensional CFD code: knowing the temperature profile from the semi-analytical solution and starting from $x = 0$ (surface), grid points are placed such that the difference in interpolated temperature between two successive grid points is equal to or below a given threshold. By varying this threshold (from 0.05K to 50 K), grids with varying levels of refinement are obtained, whose point distribution is relatively well adapted to the problem. The finite volume mesh is then generated by taking these grid points as the positions of the cell

faces. The generated meshes are then extended with a geometric progression of cell sizes so that their outer limits are far beyond the characteristic thermal length scales ($\approx 10^{-4}$ m here). The semi-analytical solution is taken as the initial state and advanced forward in time with backward Euler and large time steps, until stabilisation of the CFD solution.

We show in Figure 4 the convergence of the CFD result towards the semi-analytical solution for the reference case, as a function of the number of cells. Second-order accuracy in space is attained, and the relative errors reach 10^{-8} on T_s at around 4000 adapted mesh cells, and similar results are obtained on temperature profiles. Pushing the error below 10^{-8} on T_s is difficult since it is very close the maximum accuracy achievable by the Newton algorithm in double-precision arithmetic. Still, the error is sufficiently small so that we can consider the CFD solution spatially converged. The steady-state solutions are coherent between both approaches, thus verifying our spatial semi-discretisation.

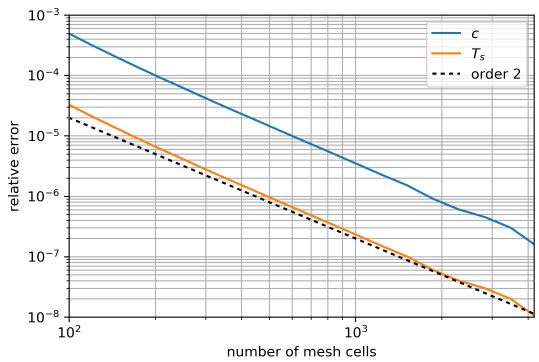


Figure 4 Convergence of the steady-state CFD solution towards the semi-analytical solution [53]

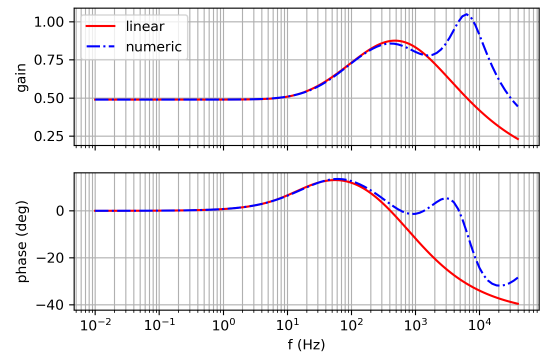


Figure 5 Bode diagram of the response function R_{mp} to pressure fluctuations

5.2. Verification in unsteady regime

We now wish to verify the time integration algorithm implemented in the CFD code. No analytical solution is available in the general unsteady regime, however linear frequency responses to pressure oscillations are available [15], assuming a quasi-steady gas phase. The linearised response can be constructed analytically using coefficients that represent the sensitivity of the steady-state solution to certain parameters. These sensitivities can be computed via finite differences. The 1D CFD code is initialised at the steady-state profile and a sinusoidal pressure oscillation $P(t) = \overline{P} + p' \sin(2\pi f t)$ is enforced in the gas phase, with f a given frequency. The mean pressure $\overline{P}$ is set to $= 50 \times 10^5$ Pa, and the amplitude is set to $p' = 0.001\overline{P}$. In the linear regime (p' small), this leads to sinusoidal fluctuations of amplitude m' in the pyrolysis mass flow rate $m(t) = \overline{m} + m' \sin(2\pi f t + \phi)$, with ϕ the phase shift. After a few periods, these oscillations stabilise and we can determine the response function $R_{mp} = (m'/\overline{m})/(p'/\overline{P})$ at the corresponding frequency.

Figure 5 shows the comparison of the linearised and numerical frequency responses. We see that the agreement between both methods is excellent up to approximately 500 Hz, where the gas phase no longer has a quasi-steady behaviour, thus introducing a larger error in the linear response function. This serves as a global verification of our unsteady model. The secondary peak at high frequencies in the response can be obtained analytically if the unsteady gas phase equations are also linearised, as in [54], however this is a much more involved process. It can also be removed by enforcing the quasi-steadiness of our gas-phase model.

An additional verification of the orders of convergence in time for unsteady simulations is presented in Appendix C, thus completing the verification process in terms of both spatial and temporal discretisations. In particular, algebraic constraints do not hinder the high-order convergence. We now wish to tackle three much more challenging test cases and investigate the behaviour of the proposed strategy in terms of accuracy and computational efficiency.

6. Simulation of ignition transients

6.1. Setup

We use the same simplified model as previously described in Section 5.1.1. The initial solution is a uniform temperature field at 300 K, with only combustion products in the gas phase, as a simpler alternative to adding nitrogen as initial gas, without much effect on the ignition process itself. We add a laser heat flux $q_r = 1$ MW.m^{-2}

absorbed at the surface via Equation (8). The mesh has 99 cells in the solid phase and 291 in the gas phase. The cells are distributed such that the steady-state temperature profile is well resolved.

We compute the ignition transient with the methods listed in Table 1. The ESDIRK methods use the time adaptation strategy presented in Section 4.3, while the classic schemes implicit Euler (IE) and Crank-Nicolson (CKN), with a time adaptation based on the requirement that the solution has a relative variation that is below a certain value between two consecutive time steps. A discussion on a CFL-based time adaptation is presented at the end of this section. We use the abbreviation *rtol* to refer to the relative integration error tolerance for ESDIRK methods, and to the allowed relative variation of the solution between consecutive time steps for IE and CKN. The maximum time step allowed is 0.1 s.

6.2. Physical interpretation of the ignition

Figure 6 shows the evolution of the surface temperature during ignition as computed by ESDIRK-54 with $rtol = 10^{-6}$. The first phase is the inert heating of the solid propellant. The constant laser heat flux produces an evolution of T_s which is proportional to $\sqrt{t}$. When T_s is sufficiently high, the pyrolysis mass flow rate given by (11) increases rapidly, causing the release of gaseous pyrolysis products in the gas phase, which chemically react and form a flame that heats up the solid even more. Typically at this point, more thermal energy is stored in the solid as compared to steady-state. This results in a momentarily higher regression rate at ignition, seen here near $t = 0.35$ s, which evacuates this excess of solid phase thermal energy. The temperature profile then converges the steady-state solution.

6.3. Result

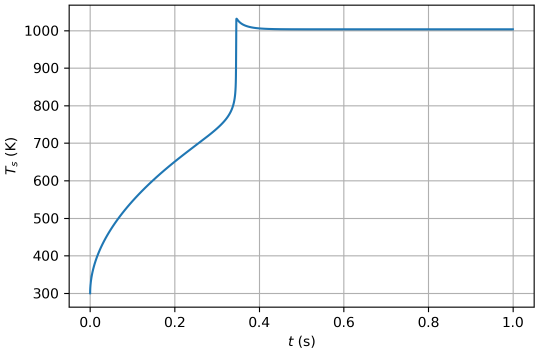


Figure 6 Surface temperature evolution during laser ignition

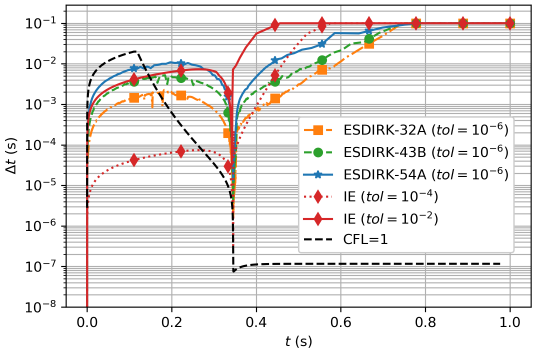
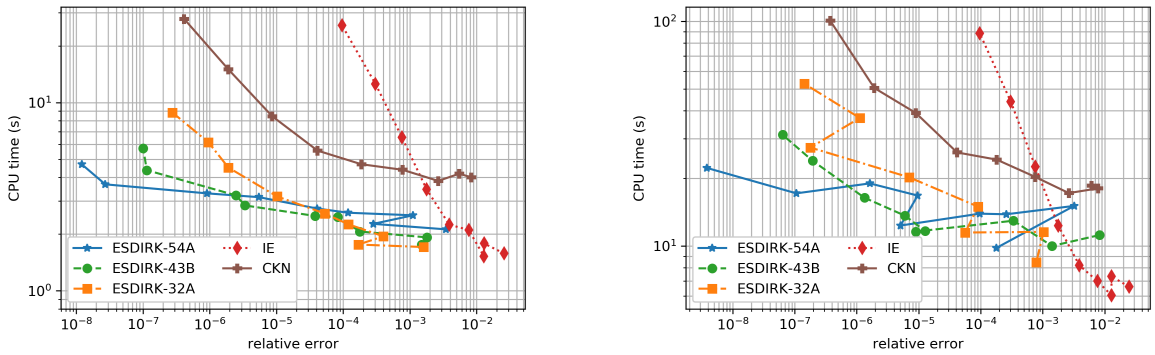


Figure 7 Time steps used by each method

We mainly compare the evolution of T_s, the computational time, and the ignition time t_{ign}. The latter is defined as the time at which the surface temperature first exceeds 1000 K and is obtained by high-order interpolation. Although not shown here, the curves of T_s for each method are very similar, except for IE simulations with large tolerances that deviate slightly during the inert heating and ignite a few milliseconds earlier. Figure 7 shows the evolution of the time step for some of the simulations. We observe that, for ESDIRK embedded methods, increasing the order of the method allows for larger time steps to be used throughout the integration while maintaining the same accuracy. For example, the fifth-order method ESDIRK-54A is capable of taking steps 5 times larger in average than the third-order method ESDIRK-32A. Finally, it is clear that IE needs many more steps to achieve a similar result as the ESDIRK methods.

To more quantitatively assess the efficiency of each method in precisely determining the ignition time, multiple simulations were run with each scheme. For the ESDIRK methods, the relative integration error tolerance was varied from 10^{-1} to 10^{-6}. For IE and CKN, the relative solution variation allowed between successive time steps was varied from 10^{-1} to 10^{-4}, without any CFL-limitation. For each simulation the value of t_{ign} is evaluated and a relative error on this value can be inferred by comparing it to the ignition time obtained with ESDIRK-54A and the tightest tolerance. Figure 8a shows the computational time required by each method to achieve a given level of relative error on the initial mesh. Figure 8b shows the computational time required by each method on a refined mesh with 734 cells in the solid phase and 1315 cells in the gas phase. Overall in both cases, if a relatively large error of 1% on the ignition time is deemed sufficient, IE is a relevant choice. If greater precision is however required, ESDIRK schemes with adaptive time stepping as described in Section 4.3 are much more efficient.



(a) coarse mesh (400 cells) **(b)** fine mesh (2049 cells)

Figure 8 Work-precision diagram for the determination of t_{ign}

The black dashed curve in Figure 7 shows the time step evolution corresponding to CFL = 1 for the coarse mesh. We clearly see that the CFL-limitation is irrelevant during the inert heating phase, as the mass flow rate is very low. If, after the inert heating, the CFL is to be limited to moderate values (1 to 100) as is usually done, many more time steps would be performed. ESDIRK methods are able to give very accurate results without any such limitation. We have determined that, if a simulation was to be performed with a maximum CFL of 10 with ESDIRK-54A and $rtol = 10^{-6}$ on the physical time interval $[0, 0.35]$ s, i.e. only up to $T_s \approx 820$ K before the ignition, the simulation would take 4 times more steps than required without CFL-limitation. If the final physical time is increased to 0.4 s to include most of the transient, this ratio would be 160. Knowing that the simulation without CFL-limitation already achieves an error smaller than 10^{-6} on the ignition time and on the rest of the evolution, this clearly shows that a CFL-constraint is not a good choice in terms of computational efficiency.

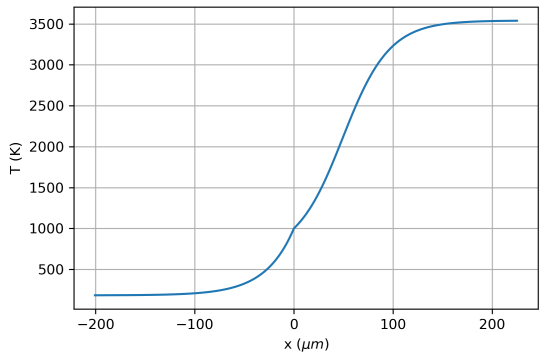
Such an ignition simulation is much more time-consuming when no adaptive time stepping is used. Indeed, the fixed time step used must be such that the fastest phases, i.e. initial heating and transition to ignition, are sufficiently well resolved. For instance, a simulation with ESDIRK-54A and $\Delta t = 10^{-4}$ s took approximately 400 s to complete. This is 20 times more than an adaptive simulation with $rtol = 10^{-6}$, which also produces a more accurate result.

The advantage of embedded methods is that only a relative integration tolerance needs to be specified. No tuning of a CFL-criterion, maximum relative variation or fixed time step is required, hence such methods speed up the engineer task of simulating different scenarios, while still ensuring a controlled error. One interesting observation we made is that the time step values taken by ESDIRK methods were almost identical with both meshes. In all our testing, no correlation was found between the time step evolution of the embedded methods and the mesh refinement. This would not be the case if a CFL-criterion was used.

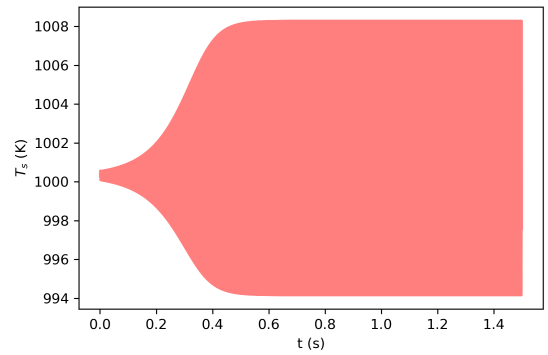
7. Investigation of limit cycles

The effort made in terms of time integration strategy can be used to accurately study the nonlinear behaviour of the propellant combustion, and in particular potential departures from an unstable steady-state travelling wave solution. Using the procedure detailed in Appendix A, we have generated a variant of the simplified combustion model used in Section 6, whose steady state solution is linearly unstable under perturbations but stabilises onto a limit cycle via nonlinear effects. Using its steady-state temperature profile plotted in Figure 9a as initial solution and applying a small pressure perturbation, the transient destabilisation is computed with ESDIRK54-A and $rtol = 10^{-6}$. The surface temperature history is plotted in Figure 9b, clearly exhibiting a limit cycle, the discrete Fourier Transform of which is presented in Figure 9c. The limit cycle can be decomposed as a sum of sinusoidal harmonic oscillations, with a fundamental frequency of 452 Hz, close to the propellant natural frequencies defined in [15] and [36], at 518 Hz and 348 Hz respectively.

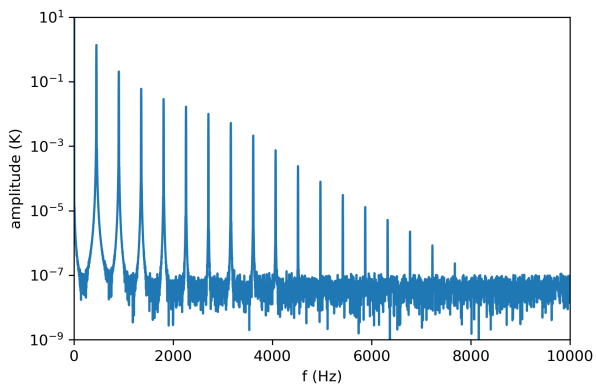
As reported in the literature for other applications [6], high-order methods are often needed to be able to numerically reproduce such a dynamical and nonlinear behaviour. It is therefore instructive to compare the methods from Table 1 to see how they affect the unsteady result. Namely, it is expected that the integration methods will, depending on their order, stability properties and the time step used, dampen the oscillating nature of the system and potentially cause a non-physical stabilisation of the solution.



(a) Steady-state temperature profile



(b) Limit cycle



(c) Discrete Fourier Transform of the limit cycle

Figure 9 Main features of the studied configuration

First, constant time steps simulations are performed. Comparisons are made based on the ability to reproduce the initial amplification of the oscillations, the fundamental frequency of the limit cycle and its amplitude. Second, simulations with time adaptation are presented.

7.1. Method

As described in Appendix A, the simulations are performed on a non-uniform mesh based on the temperature profile of the steady-state solution with 55 cells for the solid phase and 146 for the gas phase. It has been verified that additional refinement would not affect the solution dynamics.

We focus on several aspects. First we analyse visually the envelope curve of the surface temperature time history. For each simulation, this curve is constructed as the junction of the successive maxima of T_s. Unfortunately, for large values of the time step Δt, the sampling frequency $1/\Delta t$ can be too large compared to the fundamental frequency of the limit cycle, causing artefacts to appear in the form of an oscillation of the envelope. This can be improved by using a cubic interpolant of T_s to determine the successive maxima with greater precision, however envelope oscillations are still present, for example in Figure 10e.

Second, we analyse the growth of the linear instability near $t = 0$. For each simulation, the best exponential fit for the envelope of the evolution of T_s is determined, i.e. the curve joining the successive maxima of T_s, obtained from a cubic interpolation of the temporal evolution of T_s. Such a fit is of the form $T_{fit}(t) - T_s(0) = A\exp(bt)$, with b the fitted amplification factor.

Third, the established limit cycle is considered, on the time window $1 \leq t \leq 1.5$ s. A discrete Fourier transform of the surface temperature signal is computed via an FFT algorithm, as shown for the reference simulation in Figure 9c. Interpolation of the solution on a uniform time grid is performed if the simulation was not conducted with a constant time step. This FFT helps determine the approximate frequencies of the different harmonics with a precision of approximately 1-10 Hz. For each of these peaks in the spectrum, the peak frequency is then precisely computed by maximising the correlation between $T_s(t)$ and $\exp(2i\pi ft)$, from which we can also determine the

precise amplitude of the corresponding peak. These values offer a trustworthy and precise indication of how well the limit cycle is captured.

Finally, work-precision diagrams are given which represent the computational time required to achieve a specific level of relative error on the quantitative results, i.e. fitted amplification factor, fundamental frequency and amplitude. Relative errors are computed relative to the values obtained with the most refined solution.

7.2. Analysis of schemes efficiency for a constant time step

7.2.1. Envelope of the surface temperature history

We first observe the envelope curves of T_s for various time step values in Figure 10. We see that all methods dampen out the oscillations when the time step is too large, except CKN which stabilises at a small oscillating amplitude. However, if we gradually decrease the time step, each method eventually produces a limit cycle. We see that ESDIRK-54A has the best behaviour in terms of reproducing the actual reference limit cycle. It generally seems that the higher the order of the method, the larger the time step can be while still resolving the limit cycle. An interesting behaviour is observed for the CKN method: though it is second-order accurate, it finds a relatively correct initial amplification with larger time steps than required by the fourth- and third-order methods, as seen in Figures 11a and 11b where CKN is the first method to obtain a sensible initial amplification of the disturbance for $\Delta t > 4 \times 10^{-4}$ s, which is also confirmed visually in Figure 10. On the opposite, all L-stable methods dampen the oscillations when the time step is too large, and none of them diverges. This can be understood by noting that, for the simple Dahlquist test equation $y' = \lambda y$, $\lambda \in \mathbb{C}$, L-stable schemes possess a bounded "instability" zone (the complement of the domain of absolute stability in $\mathbb{C}$). In our case, the initial linear instability is linked to positive eigenvalues of our system Jacobian. If the time step is too large, the product of the time step times the eigenvalue can be outside of the instability zone, thus the instability is artificially damped by the time scheme. CKN has an unbounded instability domain (the whole half-plane $Re(\lambda) > 0$), therefore this phenomenon does not occur and the instability can freely develop, even though its growth rate is poorly captured when the time step is too large, as we may observe in Figure 10f.

Finally, we see that the first-order IE is not able to correctly reproduce the limit cycle, even with the smallest time step of 10^{-6} s. Using such a time step is already prohibitive, therefore lowering it further cannot be considered a viable solution to achieve an accurate result. Convergence results presented hereafter are obtained on the range $\Delta t \in [10^{-5}, 10^{-1}]$ s, where IE never produces an initial amplification and established limit cycle, therefore its results are omitted for the sake of readability.

7.2.2. Initial growth

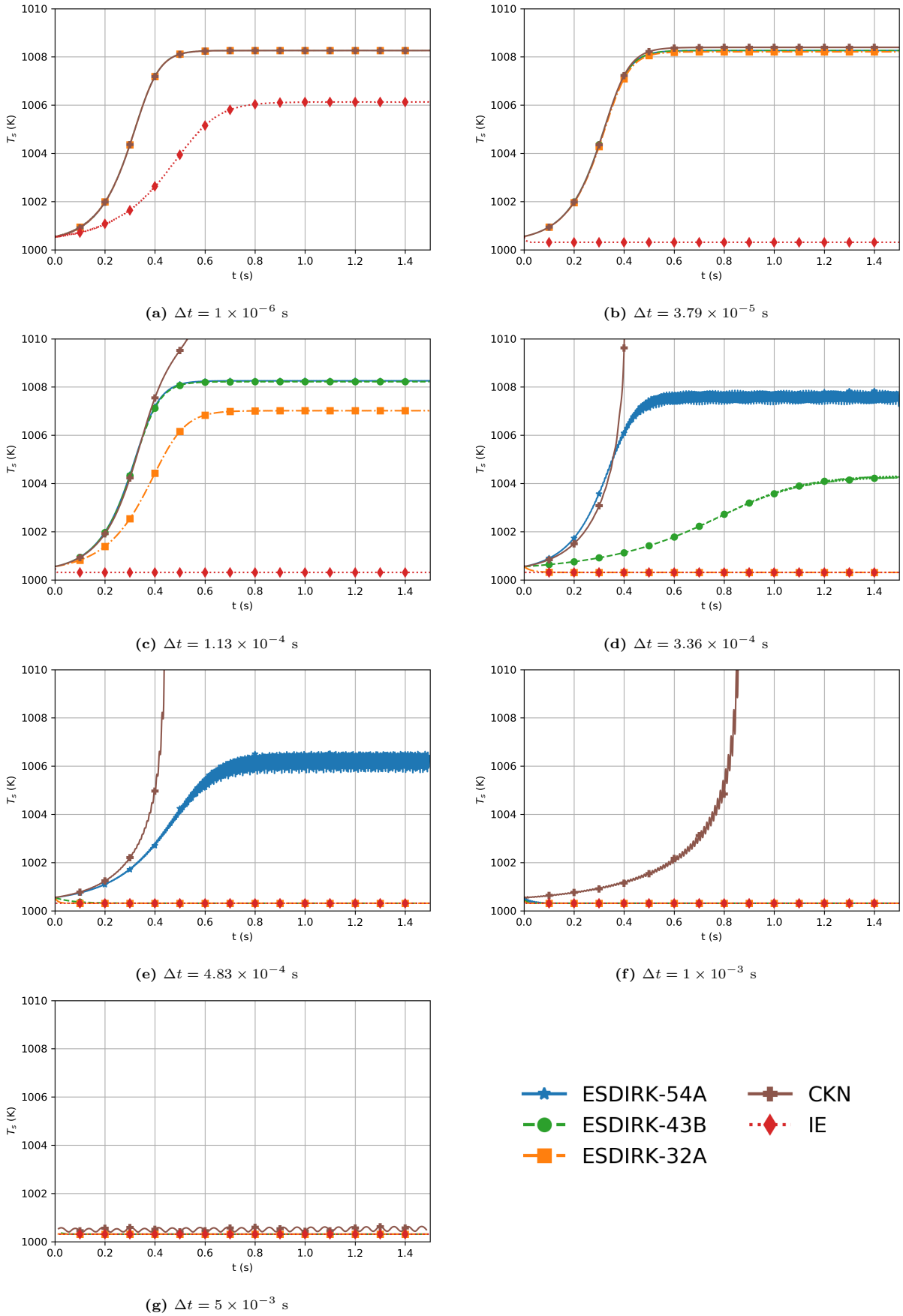
Figure 11a shows the evolution of the fitted exponential amplification factor b as the time step is lowered. We see that all methods converge to the same value, however ESDIRK-54A and CKN are the first methods that manage to capture a growth (crossing the line $b = 0$), and also the quickest to converge to the correct value.

Figure 11b shows the relative error of the amplification factor with respect to the reference solution. We see once again the same ranking in terms of ability to find the correct factor. At any time step $\Delta t \leq 5 \times 10^{-4}$ s, ESDIRK-54A yields the best accuracy. Moreover we can observe that each method has an asymptotic convergence region where the order of convergence is close to the order of the method. In particular CKN, which initially performs well for moderate time steps, is quickly overtaken by the other methods that possess a higher convergence rate. However, we notice that both ESDIRK-43B and ESDIRK-54A converge with order 4 for the value of the amplification factor k. This can be simply explained by the fact that k is determined via cubic interpolation of the curve of $T_s(t)$ (see Section 7.1), which introduces an error proportional to Δt^4 in the determination of k, which dominates the true error of the scheme.

For low time step values, the flattening of the convergence curves can be simply explained. The amplification factor is defined as a coefficient from an exponential fit, however this fit is only an approximation, as the nonlinear behaviour will let the unsteady evolution slightly deviate from the theoretical exponential initial growth. Also, the fit is based only on the successive maxima, not on the complete oscillating curve, which induces additional errors, e.g. imprecision in the abscissas of the maxima. Therefore there is a point at which the precision achieved with an exponential approximation cannot be improved further.

7.2.3. Limit cycle

The fundamental frequency of the established limit cycle is $f \approx 452$ Hz. Figure 12a shows how the relative error on this frequency evolves with the time step, and Figure 12b shows the convergence of the amplitude of the fundamental frequency. The brown crossed curve for CKN is interrupted in the intermediate range of time step values, as the solution diverges, thus not allowing for an established limit cycle to be analysed. We can observe that



(a) $\Delta t = 1 \times 10^{-6}$ s

(b) $\Delta t = 3.79 \times 10^{-5}$ s

(c) $\Delta t = 1.13 \times 10^{-4}$ s

(d) $\Delta t = 3.36 \times 10^{-4}$ s

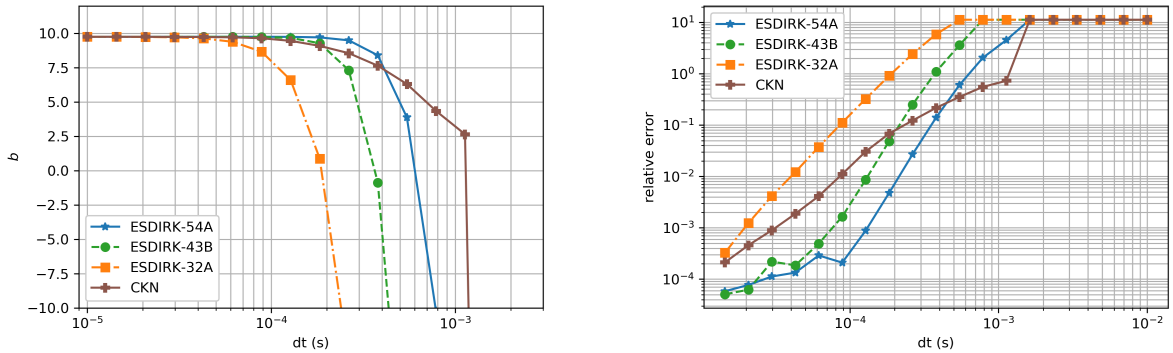
(e) $\Delta t = 4.83 \times 10^{-4}$ s

(f) $\Delta t = 1 \times 10^{-3}$ s

(g) $\Delta t = 5 \times 10^{-3}$ s

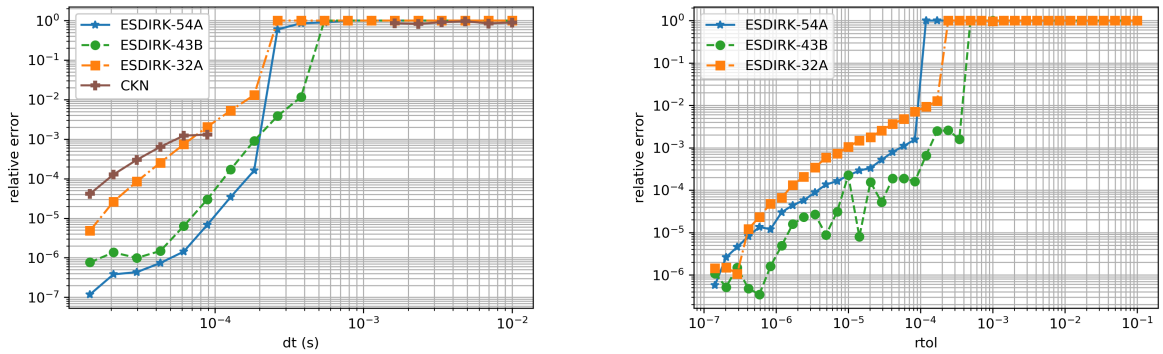
Figure 10 Envelopes of the surface temperature histories computed for different time steps and integration methods

ESDIRK-43B and ESDIRK-54A yield the most precise solutions at any given time step. In particular, ESDIRK-43B is able to capture a non-zero oscillation amplitude with larger time steps than required by the other methods. The frequency-finding process is limited in accuracy for the determination of the fundamental frequency, hence the flattening of the convergence curves when the relative error reaches 10^{-6}.



(a) Value **(b)** Convergence of relative error

Figure 11 Fitted amplification factor



(a) Fundamental frequency **(b)** Amplitude of the fundamental

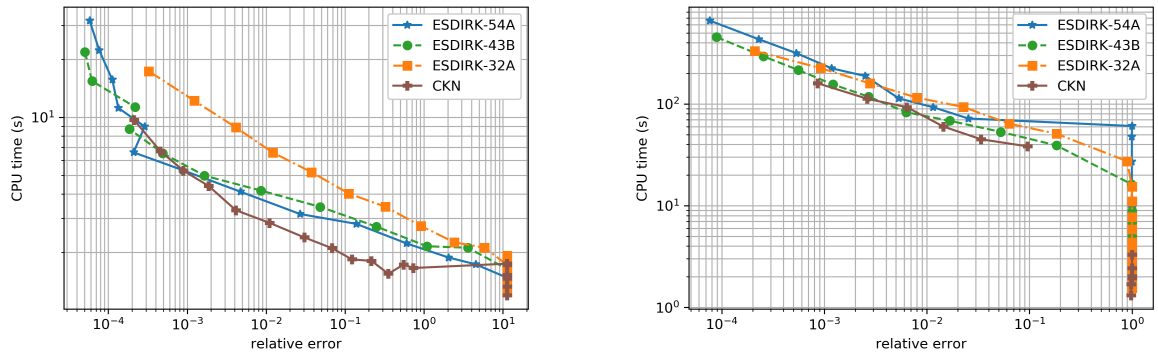
Figure 12 Convergence of the limit cycle properties

7.2.4. Computational cost

Based on the previous analysis, the high-order methods ESDIRK-54A and ESDIRK-43B seem particularly promising, as they require fewer steps to achieve good results. However, since these methods possess more stages than lower-order ones, this does not ensure that the actual computational times will be shorter. Figure 13a shows the computational time versus the achieved relative error on the amplification factor. Note that these simulations were run only on the physical time interval $t \in [0, 0.1]$ s, so that computational times are truly representative. We see that CKN is the fastest method for a relative error higher than 5×10^{-3}, however this roughly corresponds to the zone were its solution diverges in finite time. ESDIRK-32A is not a very good performer, whereas ESDIRK-43B and ESDIRK-54A are performing well and have similar error levels.

Figure 13b shows the computational time required for a given level of relative error on the fundamental amplitude in the established limit cycle. Computational times are those of simulations run on the physical time interval $t \in [0, 1.5]$ s. ESDIRK-43B is only marginally better than the other methods, no clear winner is to be picked.

Overall, when using constant steps, the high-order methods ESDIRK-54A and ESDIRK-43B are almost identical and offer overall a very good performance. The Crank-Nicolson method is slightly misleading: its lack of stability when applied to DAEs leads to an easier destabilisation of the initial solution. However it diverges quickly, unless the time step is severly reduced, thus falsely leading to the conclusion that the configuration is purely unstable and does not produce a limit cycle.



(a) Amplification factor (transient-only)

(b) Amplitude of the fundamental of the established limit cycle

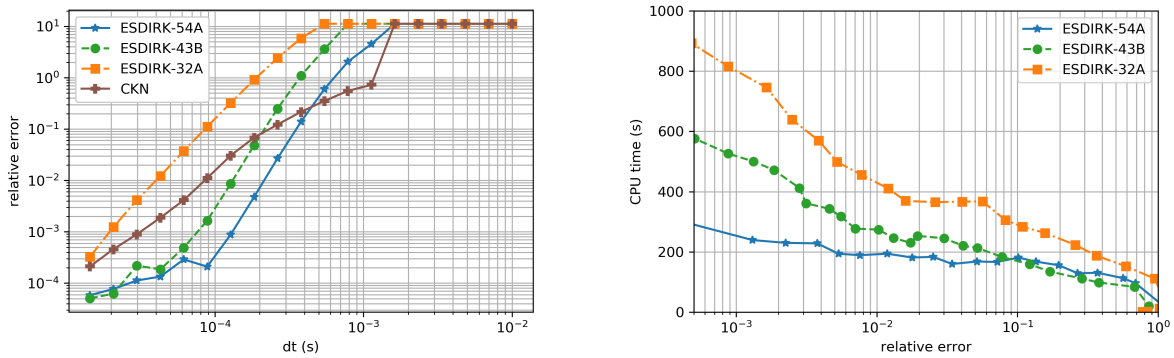
Figure 13 Work-precision diagrams for fixed time step simulations

7.3. Numerical experiment with time adaptation

The previous study with constant steps has shown that high-order methods are interesting for the simulation of a limit cycle. We now compare the embedded ESDIRK schemes with time adaptation enabled, to see if additional computational gains can be obtained. Following the methodology exposed in Section 4.3, the time step is controlled by the relative integration error tolerance $rtol$, which is varied between 10^{-1} and 10^{-7}. We first focus on the relative error achieved on the quantitative criteria used in the previous section. Finally, a comparison of the computational times and relative errors is presented, considering both fixed time step and adaptive simulations.

7.3.1. Initial growth

Figure 14a shows the convergence of the fitted amplification factor b when $rtol$ is decreased. We see that with fine tolerances, all three methods resolve the transient quite well. The achieved relative error is proportional to $rtol$ when the latter is sufficiently small, which indicates that the error estimation based on the embedded schemes behaves correctly. However, we see that there is a difference in each method's error levels for a given value of $rtol$, which is expected as they have different leading truncation error coefficients. In Figure 14b, we plot the accuracy achieved on the amplification factor (which is not equal to $rtol$) versus the computational time. We clearly see that, when the tolerance is sufficiently low, the computational cost decreases as the order of the method increases. For example, if we require a relative error of 10^{-3}, ESDIRK-54A is twice as fast as ESDIRK-43B , and three times as fast as ESDIRK-32A. Only for high levels of error ($> 10^{-1}$) is ESDIRK-43B slightly more efficient than ESDIRK-54A.



(a) Convergence of the initial growth factor

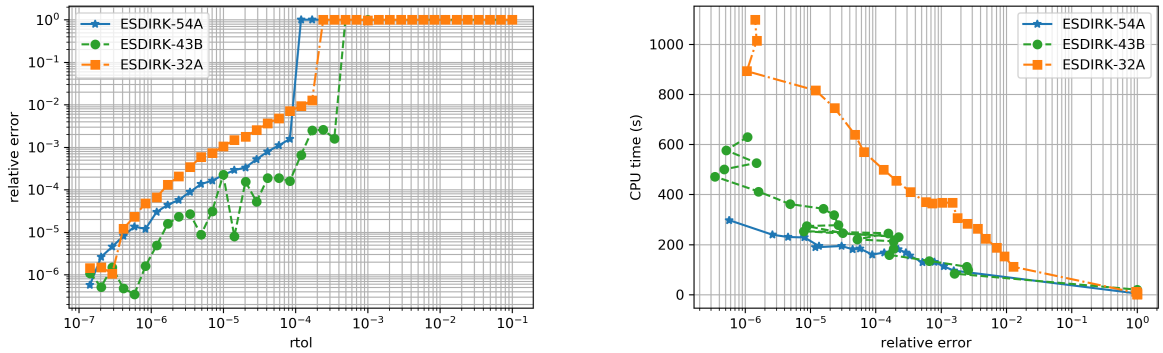
(b) Work-precision diagram

Figure 14 Convergence and computational cost for the amplification factor

7.3.2. Limit cycle

We now compare the computational cost of each method when considering the resolution on the time interval $t \in [0, 1.5]$ s. We have observed that the frequency of the fundamental and its amplitude converge equally well,

therefore we only focus on the amplitude. Figure 15a shows how the relative error on the fundamental amplitude evolves with *rtol*. ESDIRK-43B has, for a given *rtol*, the lowest error, however there is an unexplained oscillation of the relative error. Figure 15b is the corresponding work-precision diagram. ESDIRK-32A is the worst performer by far, whereas ESDIRK-54A is the most efficient method.



(a) **(b)**

Figure 15 Limit cycle with adaptive time stepping: (a) convergence of the fundamental amplitude, (b) work-precision diagram

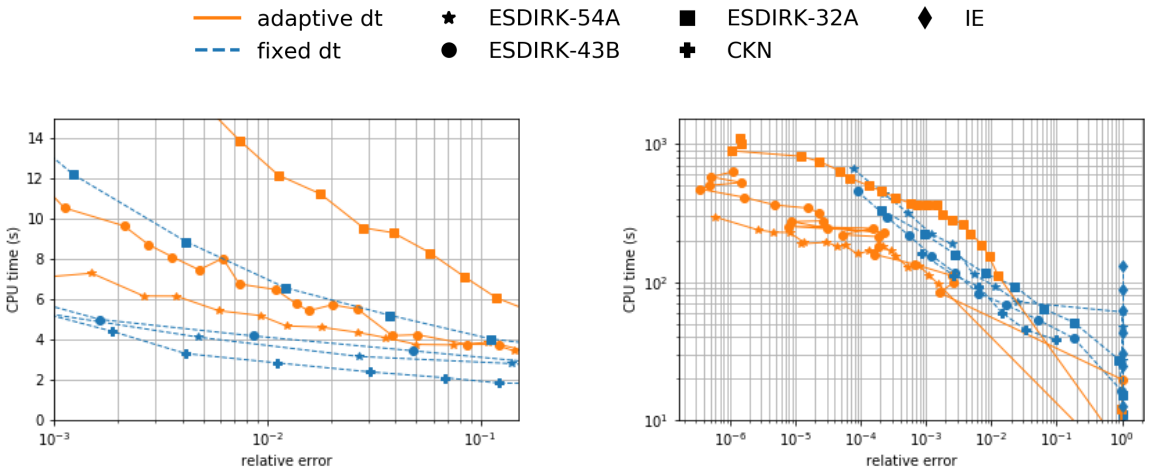
7.3.3. Comparison with fixed time step results

We now wish to assess the performance gain achieved with time adaptation. To this end, we compare the computational times between fixed time step simulations and adaptive simulations for a given level of relative error. Figure 16a shows how the relative error on the amplification factor b during the initial growth evolves with computational time. The simulations were run for only 0.1 s of physical time, i.e. only for the initial growth. We observe that fixed time step implementations of the ESDIRK methods are faster than their adaptive counterparts. It has been assessed that this was due to an advantage in terms of number of Jacobian evaluations for the Newton method. Indeed, for the short time range simulated, the instability remains linear, i.e. the instability is linked with constant positive eigenvalues of the Jacobian of the system. Hence, the Jacobian does not need to be updated if the time step is constant. In practice, fixed time steps solutions only required up to 3 evaluations of this matrix, while adaptive solutions required up to 100 evaluations due to the repeated changes in time step. It is possible to mitigate this issue by adding a "deadzone" for the time step evolution, i.e. refuse time step increases that are lower than a given percentage. In our testing, a 50 % deadzone yields a 25% CPU time improvement, dividing by 3 the number of Jacobian evaluations. However, such a deadzone approach did not perform very well in the ignition test case from Section 6, hence we do not consider it further. Finally, the Jacobian update process could be improved. Currently, as discussed in Section 4.2, every time the Jacobian J of the residuals F needs to be updated, the Jacobians of f, g and Q are also updated. In the case of nearly linear dynamics, large computational gains could be expected from simply refactorising J upon each time step change, and only updating the stored Jacobians of f, g and Q when the Newton fails to converge properly. This would however require a number of modifications in our code, so that f, g and Q can be evaluated separately. Furthermore, in our typical use cases (e.g. ignition transients as in Section 6), we observed that our code often performs successful time steps (i.e. with a sufficiently low estimated temporal error) during which the Newton algorithm has required one or more Jacobian updates for convergence for one or multiple stages of the Runge-Kutta step. This means that nonlinearities encountered across a single step are already sufficiently large to require a full Jacobian update, hence we expect limited gains from being able to only refactor J upon time step changes in these scenarios.

The performance for the computation of the full limit cycle (initial growth and established cycle) is assessed in Figure 16b. The criterion is the relative error on the amplitude of the fundamental in the established limit cycle. One may think that the lightweight second-order Crank-Nicolson method could outperform the other methods for the established limit cycle, as this method is known to have good damping properties for oscillating systems, while only requiring one stage to be computed per time step. Indeed the method is the fastest among the fixed time step ones, and the fastest overall for relative error levels around 10^{-2}. For lower error levels however, the adaptive high-order methods ESDIRK-54A and ESDIRK-43B are the most efficient methods. ESDIRK-32A in adaptive mode is generally slower than in fixed time step mode, unless very low errors are sought. Although not shown here, adaptive simulations require more Jacobian evaluations due to the successive changes in time step, but they require

many fewer steps and Newton iterations overall, which, for the complete simulation, far outweighs the drawback of evaluating the Jacobian more often. This is supported by the observation that the slopes of CPU time versus relative error are smaller for adaptive methods compared to fixed time step implementations. Any additional cost, e.g. Jacobian evaluation, is overcome by the ability to take fewer steps. Still, adaptive methods are at a slight disadvantage in this test case, as the solution oscillates smoothly: the characteristic time scale of the system stays roughly constant throughout the simulation, therefore fixed time step simulations, with dt sufficiently low compared to this time scale, will be favoured by this consistency.

An interesting observation can be made: the adaptive methods always capture the correct limit cycle, unless $rtol$ is too high, leading to a stabilisation of the solution. This is seen in Figure 16b, as all adaptive methods have a jump from important errors (≈ 1) to much lower ones as $rtol$ is lowered. On the opposite, fixed time step implementations do not have such a jump in error and are more likely to capture a non-accurate limit cycle (typically with an error higher than 10^{-2}) for an intermediate range of time step values. Overall, ESDIRK-54A and ESDIRK-43B seem to be the most reliable methods in this comparison.



(a) Fitted amplification factor

(b) Fundamental amplitude in established limit cycle

Figure 16 Comparison of the computational cost for a given level of relative error

Another practical consideration is that the time step corresponding to CFL $= 1$ lies around 10^{-6} s, which is approximately 100 times smaller than the time step necessary to obtain a very precise simulation with the fifth-order method ESDIRK-54A (see Figures 11b and 12b). The CFL-controlled time step is based on a stability analysis of the convection operator with an explicit time integration, which is not relevant for implicit integration and does not guarantee any level of error on the solution. As already discussed for the ignition transient in Section 6, use of a CFL-limitation would result in an important increase in computational time, without any valuable improvement on the solution accuracy. Finally, the constant time step simulations that yield accurate results more efficiently than with adaptive methods correspond to CFL ≈ 100, which would usually not be expected to produce accurate unsteady results. One would rather safely choose a time step such that CFL $= 1$. This again highlights one practical benefit of time adaptation: precision is ensured based on a reliable mathematical criterion, and time step values can be used such that CFL $\gg 1$, while still ensuring a precise solution.

8. Application to unsteady combustion with detailed chemistry

We have now verified that the high-order adaptive methods perform well on nonlinear problems involving simplified chemistry models. Additional stiffness is usually observed when a complex kinetic mechanism is used, thus it is useful to check the behaviour of the proposed numerical strategy in this context. The test case is the unsteady combustion of the AP monopropellant. Gas-phase kinetics is based on the AP-HTPB mechanism developed by Jeppson [55] and Tanner [56]. All reactions involving carbonated species were removed to account for the absence of HTPB, resulting in a pure AP combustion mechanism involving 25 species and 80 reactions. Thermodynamic and transport properties are computed before-hand by CHEMKIN routines [57] and stored as lookup tables.

The solid phase and the surface are handled as in [36]: the solid is assumed inert, and all decomposition and gasification reactions occur at the surface. There are two global surface reactions: a direct dissociative sublimation,

and a quasi-equilibrium decomposition. The regression speed is defined by a pyrolysis law taken from [36], and the proportions of gaseous products generated by the surface reactions are adjusted to obtain the experimentally measured regression rates at 20 atm [17]. These modelling choices allow for the use of detailed combustion kinetics while remaining within the comparatively simpler framework of solid and surface representation used in the present paper.

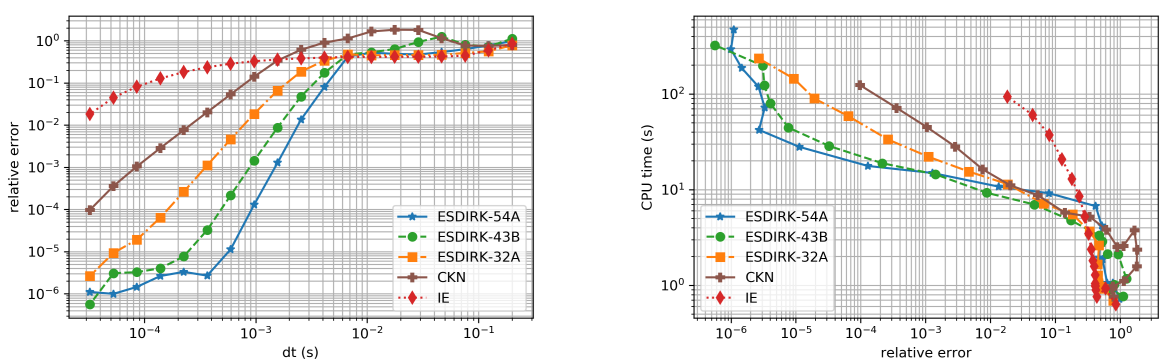
The computational mesh has 49 cells for the solid phase, and 126 cells for the gas phase, distributed in a non-uniform manner so that steady-state gradients are well resolved. Starting from a steady-state solution at $P = 20.265 \times 10^5$ Pa, we study the transient occurring after a pressure step to $P = 20 \times 10^5$ Pa.

8.1. Order of convergence

The orders of convergence of the methods from Table 1 have already been verified for the simpler test case of Section 5.1 and are presented in Appendix C. We now verify that the orders are not affected by the additional complexity and stiffness induced by detailed kinetics. We simulate the unsteady evolution for $t \in [0, 0.2]$ s and perform multiple integrations with various time steps. To quantify the accuracy of the overall time integration, we define the following error: $\epsilon_{T_s} = (1/t_f) \int_0^{t_f} |T_s - T_{s,ref}| dt$, with ref designating the reference simulation and t_f the final physical time. The reference simulation is computed with ESDIRK-54A and $\Delta t = 10^{-5}$ s. Cubic interpolation is used to compare both solutions on the same time grid.

Figure 17a shows the evolution of this error when the time step is varied. We see that each method attains its theoretical order of convergence. Similar results have been obtained for the other variables, both differential or algebraic. No order reduction is observed due to the stiffness of the chemical reactions with detailed kinetics. The ESDIRK methods perform well in this more complex scenario.

Figure 17b shows for each method the computational time required to achieve a given integration error ϵ_{T_s}. By analysing the different simulations, it was determined that the curve of T_s is visually converged when $\epsilon_{T_s} \leq 10^{-2}$. ESDIRK-43B and ESDIRK-54A are the most efficient methods on this error range, and the speed-up achieved by these high-order methods increases with the precision achieved. They are about 1.5 times faster than ESDIRK-32A and CKN for $\epsilon_{T_s} = 10^{-2}$, and approximately 5 times faster than CKN for $\epsilon_{T_s} = 10^{-4}$. The poor performance of IE clearly advocates high-order methods.



(a) **(b)**

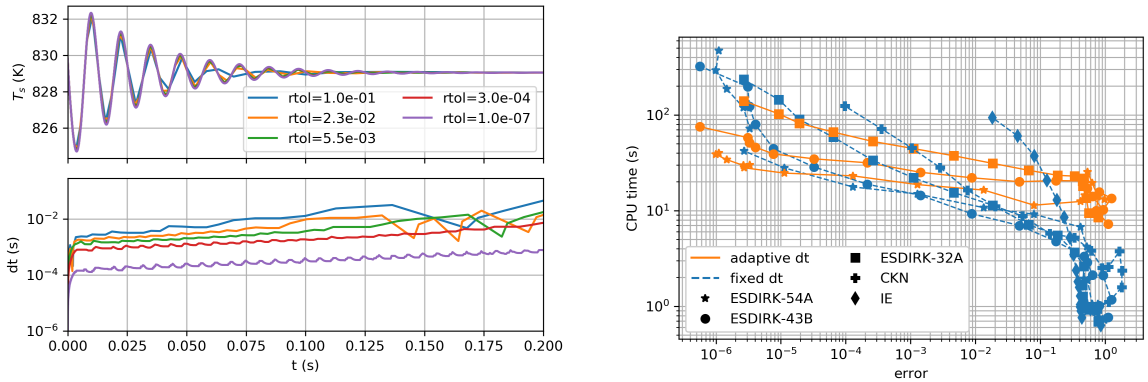
Figure 17 Accuracy of the integration with fixed time steps: (a) Convergence of ϵ_{T_s}, (b) Work-precision diagram for ϵ_{T_s}

8.2. Computational performance with time step adaptation

Now that we have verified that the convergence of the methods is not affected by the stiffness induced by complex kinetics, we use the ESDIRK methods with time step adaptation to see how they compare in terms of results. Different values of the relative integration tolerance $rtol$ are used between 10^{-1} and 10^{-7}.

Figure 18a shows the complete transient for the surface temperature and the time step evolution for various values of $rtol$. We see that the change in time step is smooth, except for low tolerances when the time step becomes large, causing convergence issues. The temporal evolution of the surface temperature is well resolved even with relatively large values of $rtol$. Figure 18b shows the comparison of the computational time required to achieve a given level of error ϵ_{T_s}, both with fixed time steps (blue lines) and adaptive time stepping (orange lines). Here, adaptive schemes do not seem to improve the performance globally. ESDIRK-54A is the best performing adaptive method, however it only becomes the fastest method overall for a very low level of error $\epsilon_{T_s} \leq 10^{-5}$. Its computational time

is relatively close to the one of its fixed time step implementation. We observe that, for a given increase in accuracy, adaptive methods have a lower increase in computational time compared to their fixed time step counterparts.



(a) **(b)**

Figure 18 Integration with adaptive time stepping: (a) time step evolution for ESDIRK-43B, (b) work-precision diagram for ϵ_{T_s}

This comparison is however slightly unfair, because we had no a priori knowledge of the time step needed to properly resolve the transient. Also, the characteristic times of the dynamics does not vary much , as we can see from the small variations of the adapted time step in Figure 18a. Furthermore, the pressure step is sufficiently low so that nonlinearities remain small, allowing for very few Jacobian updates to be performed when using fixed time steps. Without dynamic adaptation, the time step would have typically been limited so that the CFL number is reasonably low, e.g. 1 to 10. A unitary CFL corresponds to $\Delta t \approx 7 \times 10^{-8}$ s, which is much lower than the time step required to achieve a good accuracy with most methods. A simulation has been carried out only on the first 0.03 s of the transient, with CKN and a time step set such that CFL = 10. The computational time was 660 s. An equivalently well resolved transient can be obtained with ESDIRK-54A and $rtol = 10^{-5}$ in only 11 s. As we can see in Figure 18a, even a less stringent tolerance would also be sufficient. From an engineering point of view, this represents a 60 times speed-up, due to the fact that the time adaptation will automatically choose the relevant time step values. This guarantees a proper accuracy, while reaching CFL numbers that one would usually never trust to yield accurate unsteady results. Finally, time adaptation based on embedded methods automatically detects a slowdown of the dynamics as the solution stabilises and is able to increase the time step accordingly, whereas the CFL number stays roughly constant and cannot be an efficient time-step controlling criterion in that situation. This gain in engineering time is not quantifiable precisely, however it is definitely important.

9. Conclusion

This contribution presents the development of a high-fidelity simulation tool for the one-dimensional low-Mach combustion, with a particular focus on solid propellant applications. The emphasis is on the numerical strategy for the time integration of the semi-discretised equations. It has been shown that the system is differential-algebraic in nature. Multiple test cases show that stiffly accurate singly-diagonally implicit Runge-Kutta methods are highly efficient for the time integration of such a system, in particular the embedded ESDIRK methods presented in [45]. Handling the continuity equation with a Runge-Kutta quadrature instead of the classic instantaneous reformulation leads to greater computational efficiency. Applications have been presented for ignition transients and limit cycle with a simplified modelling, and appreciable computational gains have been observed. High-order methods reliably capture dynamics which are practically impossible to reproduce with traditional low-order methods. It has also been verified that the proposed numerical strategy is robust and performs well when the modelling is much more complex, e.g. detailed kinetics in the gas phase.

Adaptive time stepping based on objective error estimates ensures the temporal evolution is well resolved. From an engineering point of view, the single parameter that controls the time step is the relative integration error tolerance $rtol$. In all our test cases, we have observed that $rtol = 10^{-5}$ is sufficient to accurately resolve all unsteady phenomena. Using this value as standard tolerance liberates from the need of iterating over other practical criteria such as CFL limitation or maximum relative variation. We believe that a high-order adaptive method like ESDIRK-54A therefore allows for perceivable gains in computational time, trustworthiness of the results, and engineering time spent parametrising the time integration for a simulation. Furthermore, the proposed numerical strategy is

easy to implement in an existing code if the latter already uses an implicit Euler, Crank-Nicolson or BDF scheme, as is often the case in the literature.

Future work includes applying the presented framework to one-dimensional models involving a multiphase foam layer at the surface, and adapting the approach to higher-dimensional detailed heterogeneous combustion codes for solid propellants, e.g. Compas[23] from ONERA. The code Vulc1D has already been successfully coupled as a dynamic boundary condition for combustion chamber simulations with the 3D multiphysics CFD tool Cedre from ONERA [58], avoiding costly CFD mesh refinement near the propellant surface [59, 60]. Eventually, the proposed time strategy is not dependent on the chosen spatial discretisation and other spatial schemes could be envisioned without changing the conclusion of our study. As mentioned in the course of the paper, the numerical strategy can be applied to any other one-dimensional combustion problem in the low-Mach limit involving homogeneous or spray combustion. When index-2 algebraic variables are involved, the order obtained on such variables is limited by the stage order of the method [32, 35]. This situation occurs either for the strain rate eigenvalue for counterflow diffusion flames or in multi-dimensional Navier-Stokes equations, either incompressible or in the low-Mach limit [34, 61]; however the proposed strategy should be equally of interest in such cases.

Acknowledgments

The present research was conducted thanks to a Ph.D grant co-funded by DGA, Ministry of Defence (E. Faucher, Technical Advisor), and ONERA. The authors would like to thank Gilles Vilmart for interesting technical discussions.

References

[1] R. Kee, J. Grcar, M. Smooke, J. Miller, E. Meeks, Premix: a fortran program for modeling steady laminar one-dimensional premixed flames, Sandia Rep 143 (1985).

[2] M. Smooke, V. Giovangigli, Numerical modeling of axisymmetric laminar diffusion flames, IMPACT of Computing in Science and Engineering 4 (1992) 46–79.

[3] N. Darabiha, Transient behaviour of laminar counterflow hydrogen-air diffusion flames with complex chemistry, Comb. Sci. and Tech 86 (1992) 163–181.

[4] M. Massot, M. Kumar, A. Gomez, M. D. Smooke, Counterflow spray diffusion flames of heptane: computations and experiments, in: Proceedings of the 27th Symp. on Comb., The Comb. Institute, 1998, pp. 1975–1983.

[5] S. Descombes, M. Duarte, T. Dumont, F. Laurent, V. Louvet, M. Massot, Analysis of operator splitting in the nonasymptotic regime for nonlinear reaction-diffusion equations. Application to the dynamics of premixed flames, SIAM J. Numer. Anal. 52 (2014) 1311–1334.

[6] T. Sayadi, V. Le Chenadec, P. J. Schmid, F. Richecoeur, M. Massot, Thermoacoustic instability – a dynamical system and time domain analysis, Journal of Fluid Mechanics 753 (2014) 448–471.

[7] H. Im, L. L. Raja, R. J. Kee, L. R. Petzold, A numerical study of transient ignition in a counterflow nonpremixed methane-air flame using adaptive time integration, Combustion Science and Technology 158 (2000) 341–363.

[8] F. Williams, M. Barrere, N. Huang, Fundamental aspects of solid propellant rockets, Advisory Group for Aerospace Research and Development of NATO, Technivision Services; London, 1969.

[9] L. D. Luca, Theory of Nonsteady Burning and Combustion Stability of Solid Propellants by Flame Models, American Institute of Aeronautics and Astronautics, 1992, pp. 519–600.

[10] B. Novozhilov, Theory of Nonsteady Burning and Combustion Stability of Solid Propellants by the Zeldovich-Novozhilov Method, American Institute of Aeronautics and Astronautics, 1992, pp. 601–641.

[11] G. Lengelle, J. Duterque, J. Trubert, Physico-chemical mechanisms of solid propellant combustion, Solid Propellant Chemistry, Combustion, and Motor Interior Ballistics 185 (2000-01) 287–334.

[12] M. Denison, E. Baum, A simplified model of unstable burning in solid propellants, ARS Journal 31 (1961) 1112–1122.

[13] M. Beckstead, R. Derr, C. Price, A model of composite solid-propellant combustion based on multiple flames, AIAA Journal 8 (1970) 2200–2207.

[14] M. Ward, S. Son, M. Brewster, Steady deflagration of HMX with simple kinetics: A gas phase chain reaction model, Combustion and Flame 114 (1998) 556–568.

[15] F. Culick, A review of calculations for unsteady burning of a solid propellant., AIAA J. 6 (1968) 2241–2255.

[16] F. Culick, Combustion instabilities in solid propellant rocket motors, Internal aerodynamics in solid rocket propulsion (2004). , RTO-EN-023 RTO/NATO.

[17] V. Giovangigli, N. Meynet, M. Smooke, Application of continuation techniques to ammonium perchlorate plane flames, Combustion Theory and Modelling 10 (2006) 771–798.

[18] D. Smith, Modeling Solid Propellant Ignition Events, Ph.D. thesis, Brigham Young University, 2011.

[19] M. Beckstead, K. Puduppakkam, P. Thakre, V. Yang, Modeling of combustion and ignition of solid-propellant ingredients, Progress in Energy and Combustion Science 33 (2007) 497 – 551.

[20] K. Meredith, M. Gross, M. Beckstead, Laser-induced ignition modeling of HMX, Comb. and Flame 162 (2014).

[21] T. Jackson, J. Buckmaster, Heterogeneous propellant combustion, AIAA Journal 40 (2002) 1122–1130.

[22] S. Gallier, A. Ferrand, M. Plaud, Three-dimensional simulations of ignition of composite solid propellants, Combustion and Flame 173 (2016) 2–15.

[23] D. Davidenko, Y. Fabignon, Some aspects of detailed modeling of solid rocket composite propellants, in: 6th European Conference for Aeronautics and Space Sciences (EUCASS), 2015.

[24] Q. Li, P. Liu, G. He, Fluid–solid coupled simulation of the ignition transient of solid rocket motor, Acta Astronautica 110 (2015) 180 – 190. Dynamics and Control of Space Systems.

[25] W. A. Johnston, Solid rocket motor internal flow during ignition, Journal of Propulsion and Power 11 (1995) 489–496.

[26] A. Bizot, Ignition and unsteady combustion of AP-based composite propellants in subscale solid rocket motors, International Journal of Energetic Materials and Chemical Propulsion 4 (1997) 1046–1061.

[27] W. Erikson, M. Beckstead, Modeling unsteady monopropellant combustion with full chemical kinetics, in: 36th AIAA Aerospace Sciences Meeting and Exhibit, 1998.

[28] F. Harlow, A. Amsden, A numerical fluid dynamics calculation method for all flow speeds, Journal of Computational Physics 8 (1971) 197 – 213.

[29] P. N. Brown, G. Byrne, A. Hindmarsh, VODE: A variable-coefficient ODE solver, SIAM J. Sci. Stat. Comput. 10 (1989) 1038–1051.

[30] W. Erikson, M. Beckstead, Modeling pressure and heat flux responses of nitramine monopropellants with detailed chemistry, 1999.

[31] J. Grcar, The Twopnt Program for Boundary Value Problems, Technical Report, Sandia National Labs., Livermore, CA, 1992.

[32] K. E. Brenan, S. L. Campbell, L. R. Petzold, Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations, Society for Industrial and Applied Mathematics, 1995.

[33] Y. Liau, V. Yang, Analysis of RDX monopropellant combustion with two-phase subsurface reactions, Journal of Propulsion and Power 11 (1995) 729–739.

[34] M. N'Guessan, M. Massot, L. Séries, C. Tenaud, High order time integration and mesh adaptation with error control for incompressible navier–stokes and scalar transport resolution on dual grids, Journal of Computational and Applied Mathematics 387 (2021) 112542.

[35] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems, volume 14 of *Springer Series in Comput. Math.*, 2nd ed., Springer-Verlag Berlin Heidelberg, 1996.

[36] S. Rahman, Modélisation et simulation numérique de flammes planes instationnaires de perchlorate d'ammonium, Ph.D. thesis, Université Pierre et Marie Curie, 2012.

[37] N. Meynet, Simulation numérique de la combustion d'un propergol solide, Ph.D. thesis, Université Pierre et Marie Curie, 2005.

[38] J. Hirschfelder, C. Curtiss, R. Bird, Molecular theory of gases and liquids, Wiley, New York, 1954.

[39] A. Jameson, E. Turkel, Implicit schemes and LU decompositions, Mathematics of Computation 37 (1981) 385–397.

[40] P. Khosla, S. Rubin, Filtering of non-linear instabilities, Journal of engineering mathematics 13 (1979) 127–141.

[41] D. Anderson, J. Tannehill, R. Pletcher, Computational fluid mechanics and heat transfer, Third edition, Boca Raton, CRC Press, 2013.

[42] R. Alexander, Diagonally implicit Runge-Kutta methods for stiff O.D.E.s, SIAM Journal on Numerical Analysis 14 (1977) 1006–1021.

[43] Y.-C. Liau, E. Kim, V. Yang, A comprehensive analysis of laser-induced ignition of RDX monopropellant, Combustion and Flame 126 (2001) 1680 – 1698.

[44] E. Hairer, S. Nørsett, G. Wanner, Solving Ordinary Differential Equations I. Nonstiff problems, second ed., Springer, Berlin, 2000.

[45] A. Kværnø, Singly diagonally implicit Runge–Kutta methods with an explicit first stage, BIT 44 (2004) 489–502.

[46] A. Ostermann, P. Kaps, T. Bui, The solution of a combustion problem with Rosenbrock methods, ACM Transactions on Mathematical Software (TOMS) 12 (1986) 354–361.

[47] U. M. Ascher, S. J. Ruuth, B. T. R. Wetton, Implicit-explicit methods for time-dependent partial differential equations, SIAM Journal on Numerical Analysis 32 (1995) 797–823.

[48] S. Descombes, M. Duarte, M. Massot, Operator Splitting Methods with Error Estimator and Adaptive Time-Stepping. Application to the Simulation of Combustion Phenomena, Springer International Publishing, 2016, pp. 627–641.

[49] P. van der Houwen, B. Sommeijer, Iterated Runge–Kutta methods on parallel computers, SIAM journal on scientific and statistical computing 12 (1991) 1000–1028.

[50] E. Motheau, J. Abraham, A high-order numerical algorithm for DNS of low-Mach-number reactive flows with detailed chemistry and quasi-spectral accuracy, Journal of Computational Physics 313 (2016) 430 – 454.

[51] O. Knio, H. Najm, P. Wyckoff, A semi-implicit numerical scheme for reacting flow. II. Stiff, operator-split formulation, J. Comput. Phys. 154 (1999) 482–467.

[52] H. Najm, O. Knio, Modeling Low Mach number reacting flow with detailed chemistry and transport, J. Scientific Computing 25 (2005) 263–287.

[53] L. François, J. Dupays, D. Davidenko, M. Massot, Travelling wave mathematical analysis and efficient numerical resolution for a one-dimensional model of solid propellant combustion, Combustion Theory and Modelling 24 (2020) 775–809.

[54] P. Clavin, D. Lazimi, Theoretical analysis of oscillatory burning of homogeneous solid propellant including non-steady gas phase effects, Combustion Science and Technology 83 (1992) 1–32.

[55] M. Jeppson, M. Beckstead, Q. Jing, A kinetic model for the premixed combustion of a fine AP/HTPB composite propellant, volume 44, 1998.

[56] M. Tanner, Multidimensional Modeling of Solid Propellant Burning Rates and Aluminum Agglomeration and One-Dimensional Modeling of RDX/GAP and AP/HTPB, Ph.D. thesis, Brigham Young University, 2008.

[57] R. Kee, F. Rupley, J. Miller, Chemkin-II: A fortran chemical kinetics package for the analysis of gas-phase chemical kinetics (1989).

[58] A. Refloch, B. Courbet, A. Murrone, P. Villedieu, C. Laurent, P. Gilbank, J. Troyes, L. Tessé, G. Chaineray, J.-B. Dargaud, E. Quémerais, F. Vuillot, Cedre software, AerospaceLab Journal (2011) 1–10.

[59] L. François, J. Dupays, M. Massot, A new simulation strategy for solid rocket motor ignition: coupling a CFD code with a one-dimensional boundary flame model, verification against a fully resolved approach, in: AIAA Propulsion and Energy 2021 Forum, 2021. AIAA 2021-3695.

[60] L. François, Multiphysics modelling and simulation of the ignition transient in solid rocket motors, Ph.D. thesis, Institut Polytechnique de Paris, 2022.

[61] M.-A. N'Guessan, Space adaptive methods with error control based on adaptive multiresolution for the simulation of low-Mach reactive flows, Ph.D. thesis, Université Paris-Saclay, 2020.

[62] P. Virtanen et al., SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python, Nature Methods 17 (2020) 261–272.

Appendix A. Generating configurations with various degrees of instability

In order to highlight the benefit of the high-order adaptive time integration, we search for configurations which are linearly unstable around the corresponding steady-state solution. We generate such configurations by taking the stable simplified model from Section 5.1.1 as baseline and varying its parameters. We use existing theoretical tools to approximately evaluate the stability of the steady-state solution.

Appendix A.1. Theoretical indicator of intrinsic instability

The Zeldovich-Novozhilov (ZN) framework [10] is a useful tool to study the stability of a steady-state solid propellant combustion . It assumes that m and T_s are linked in steady-state via laws of the form $m = m(T_0, P)$ and $T_s = T_s(T_0, P)$. Considering the steady-state temperature profile $T(x) = T_0 + (T_s - T_0)\exp(xmc_c/\lambda_c)$, the temperature gradient just below the surface is $\partial_x T(0^-) = mc_c(T_s - T_0)/\lambda_c$. Replacing T_0 by $T_s - \phi\lambda_c/mc_c$ (the "apparent" or "initial" temperature), the previous relations can now be used in the unsteady regime. This is usually accepted, as long as the apparent initial temperature remains within acceptable bounds. It is also required that data for this initial temperature be available, or at least reasonably extrapolated.

The amplifications of compact perturbations can be studied by linearising the previous relations and the heat equation in the solid, leading to the definition of a stability criteria, which depends on two steady-state sensitivity coefficients: $\mathrm{r} = \left(\partial_{T_0}\overline{T_s}\right)_P$, $\mathrm{k} = (\overline{T_s} - T_0)\left(\partial_{T_0} ln(\overline{m})\right)_P$, with $\overline{\cdot}$ denoting steady-state values. Steady-state combustion is always stable if $\mathrm{k} < 1$. If $\mathrm{k} > 1$, the steady-state is stable only if $\mathrm{r} > (\mathrm{k}-1)^2/(1+\mathrm{k})$. The line $\mathrm{r} = (\mathrm{k}-1)^2/(1+\mathrm{k})$ is the locus of a Hopf bifurcation, where the steady-state solution becomes linearly unstable in a oscillating manner, with the possibility of stabilising on a limit cycle. If $\mathrm{r} > (\sqrt{\mathrm{k}}-1)^2$, the instability grows purely exponentially. The associated stability diagram is shown in Figure A.19: the leftmost parabola is the first stability limit, the second one is the onset of purely exponential instability. This stability is called "intrinsic" because it is a property of the solid propellant as an isolated system, and is not caused by the coupling with another system (e.g combustion chamber [16]).

The original ZN approach assumes a quasi-steady gas phase. Even though it has been shown that unsteady gas-phase phenomena tend to slightly widen the stability area, this first simplified analysis remains a good indicator of the stability bounds. We refer the reader to [10] for a detailed presentation of the ZN analysis and its extensions.

Appendix A.2. Optimisation problem

Let us denote as X the vector containing the parameters of the simple combustion model that we have chosen as free variables. For a given value of X, we can find the corresponding value of (r, k) by performing three steady-state simulations: one baseline simulation, one simulation with a perturbed initial temperature T_0, and one simulation with a perturbed pressure P. Then, by means of finite differences, r and k may be evaluated. This process can be

summarised as the function $f_{\mathrm{rk}} : X \to (\mathrm{r}, \mathrm{k})$. We can then generate configurations with various values of r and k by solving the following optimisation problem:

$$\min_{X} \quad f_{obj}(X) \tag{A.1a}$$

$$\text{subject to} \quad g(X) \leq 0 \tag{A.1b}$$

$$h(X) = 0 \tag{A.1c}$$

where the objective function is $f_{obj} : X \to ||f_{\mathrm{rk}}(X) - (\mathrm{r}, \mathrm{k})_{target}||_2^2$ with $(\cdot, \cdot)$ denoting a vector formulation. This is simply the constrained minimisation of the distance to the target (r, k) coefficients. Inequality constraints g ensure the physical parameters remain within realistic bounds. They can be supplemented with equality constraints h to enforce certain properties, e.g. steady-state surface temperature. The problem is solved with SLSQP from the Scipy library [62], using finite-difference Jacobians for f, q and h.

Appendix A.3. Numerical assessment of intrinsic stability

We use the previous optimisation problem to generate configurations which have their sensitivity coefficients r and k distributed regularly on a segment defined as $\mathrm{r} = 0.137$ (baseline value) and $\mathrm{k} \in [1.5, 1.75]$ (see Figure A.19), thus crossing the ZN stability limit. The optimisation is constrained to preserve physically sound characteristics (surface temperature at 1000 K, regression speed of 1 cm/s at 50 atm, 3540 K final flame temperature).

For each point, we numerically assess the stability of the corresponding steady-state combustion. A slight constant pressure perturbation (typically 0.1% of the prescribed pressure) is applied and the one-dimensional tool is run with ESDIRK-54A and a relative error tolerance $rtol = 10^{-6}$. The stability of the combustion can then be assessed numerically by analysing whether the perturbation is damped out or not.

The unsteady simulations for a few points are shown in Figure A.20. We see that instability appears between points 4 and 5, slightly further to the right than predicted analytically, as already discussed. Refining the search between these points allows us to find a configuration that exhibits a limit cycle. The corresponding model parameters are the same as in Section 5.1.1, except for the following changes: $T_{ap} = 14668$ K, $c_p = 692.8$ J/kg/K, $c_c = 1253$ J/kg/K, $T_0 = 182.4$K, $\lambda_c = 0.65$ W/m/K, $\lambda = 0.362$ W/m/K, $\mathscr{M} = 57.9$ g/mol, $\Delta h_f^0(G_1) = -2.28 \times 10^5$ J/kg, $\Delta h_f^0(G_2) = -2.22 \times 10^6$ J/kg, $A = 340.4$ s^{-1}.

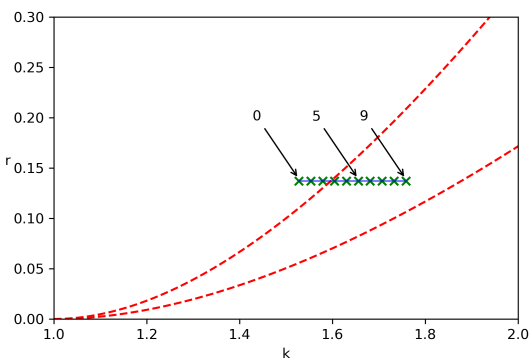


Figure A.19 Segment travelled in the (r, k) stability diagram

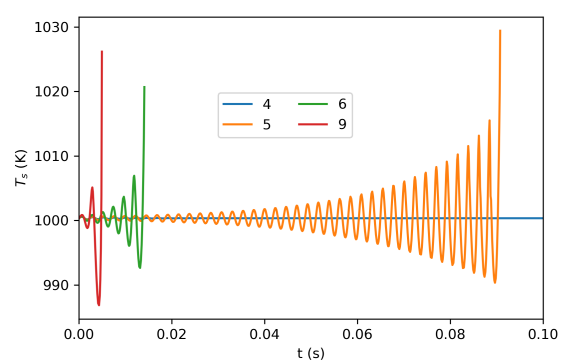


Figure A.20 Unsteady simulations

Appendix B. Detailed formulations of the mass flow rate constraint

Here we present the detailed discrete equations that can be used to account for the mass flow rate constraint arising from the continuity equation, following the discussion from Section 4.6.

Appendix B.1. Reformulation as an instantaneous constraint

The term $-\partial_t \rho$ in the continuity equation $\partial_x m = -\partial_t \rho$ can be considered as a source term which is a function of m. It can be obtained by differentiating the logarithm of the equation of state (10) with respect to time, as classically done in the combustion community [5, 50–52]:

$$\partial_x m = -\rho \left(\frac{\partial_t P}{P} - \frac{\partial_t T}{T} - \frac{\sum_{\mathfrak{k}=1}^{n_e} \frac{\partial_t Y_{\mathfrak{k}}}{\mathscr{M}_{\mathfrak{k}}}}{\sum_{\mathfrak{k}=1}^{n_e} \frac{Y_{\mathfrak{k}}}{\mathscr{M}_{\mathfrak{k}}}} \right) \tag{B.1}$$

In the right hand-side, $\partial_t P$ is an input (prescribed pressure variation, or evolution based on a combustion chamber model). The gas density ρ is directly given by the equation of state (10). The other terms can be constructed based on our original gas phase system. For instance $\partial_t Y_{\mathfrak{k}} = (\partial_t \rho Y_{\mathfrak{k}} - Y_{\mathfrak{k}} \partial_t \rho)/\rho$, where $\partial_t \rho Y_{\mathfrak{k}}$ is given by Equation (3), and $\partial_t \rho$ is replaced by $-\partial_x m$ as per the continuity equation (2). The term $\partial_t T$ is slightly more involved. We can write $\partial_t h = (\partial_t \rho h - h \partial_t \rho)/\rho$, where $\partial_t \rho h$ is given by Equation (4). Let us express $\partial_t T$ by differentiating with respect to time the definition of the enthalpy given in Section 2.1:

$$\partial_t h = \sum_{\mathfrak{k}=1}^{n_e} \partial_t Y_{\mathfrak{k}} \left(\Delta h^0_{f,\mathfrak{k}} + \int_{T_0}^{T} c_{p,\mathfrak{k}}(a) da \right) + \sum_{\mathfrak{k}=1}^{n_e} Y_{\mathfrak{k}} c_{p,\mathfrak{k}}(T) \partial_t T \tag{B.2}$$

$$\Rightarrow \quad \partial_t T = \frac{\partial_t h - \sum_{\mathfrak{k}=1}^{n_e} \partial_t Y_{\mathfrak{k}} \left(\Delta h^0_{f,\mathfrak{k}} + \int_{T_0}^{T} c_{p,\mathfrak{k}}(a) da \right)}{\sum_{\mathfrak{k}=1}^{n_e} Y_{\mathfrak{k}} c_{p,\mathfrak{k}}(T)} \tag{B.3}$$

Overall, in the $\mathfrak{i}$-th cell, the constraint (B.1) can be discretised as:

$$\frac{m_{\mathfrak{i}+1} - m_{\mathfrak{i}}}{\Delta x_{\mathfrak{i}}} = -\rho_{\mathfrak{i}} \left(\frac{\partial_t P}{P} - \frac{\partial_t T_{\mathfrak{i}}}{T_{\mathfrak{i}}} - \frac{\sum_{\mathfrak{k}=1}^{n_e} \frac{\partial_t Y_{\mathfrak{i},\mathfrak{k}}}{\mathscr{M}_{\mathfrak{k}}}}{\sum_{\mathfrak{k}=1}^{n_e} \frac{Y_{\mathfrak{i},\mathfrak{k}}}{\mathscr{M}_{\mathfrak{k}}}} \right) \tag{B.4}$$

where the various terms are evaluated following the previous formulas and the conservation equations (14) to (16). Equation (B.4) is then used used in place of its alternative formulation (14) in our DAE system.

Solving Equation (B.4) for the discrete values m_i can be done by transforming it into a problem of the form $0 = g(m)$ and applying a Newton method. This system is linear in m, thus a single Newton step would suffice. However we can see that the Jacobian $\partial_m g$ involves the solution profiles T, $Y_{\mathfrak{k}}$, therefore it will need to be updated often as the solution evolves in time and this may end up being costly. This is in agreement with our computational findings for the case where all fields are solved in a fully-coupled manner: the instantaneous formulation requires more Jacobian updates than the constraint formulation based on the Runge-Kutta quadrature presented hereafter.

Appendix B.2. Natural formulation with the Runge-Kutta scheme

An alternative and original approach of this paper is to apply the Runge-Kutta scheme to the semi-discrete continuity equation (14) directly to obtain a quadrature formula on $\int_t \partial_t \rho dt$. For the sake of readability, we momentarily change our notations to better distinguish the spatial and time representations. Let $q^{\mathfrak{i}}_n$ be the value of a variable q in the $\mathfrak{i}$-th cell at time step n (or at the $\mathfrak{i}$-th face for m), and $q^{\mathfrak{i}}_{n,i}$ the same variable at the i-th stage of time step n. For the i-th stage of any implicit Runge-Kutta method, we obtain:

$$\rho^{\mathfrak{i}}_{n,i} - \rho^{\mathfrak{i}}_n = \int_{t_n}^{t_{ni}} (d_t \rho^{\mathfrak{i}}) dt \approx \Delta t \sum_{j=1}^{s} a_{ij} \left(d_t \rho^{\mathfrak{i}} \right)_{n,j} \tag{B.5}$$

where $\left(d_t \rho^{\mathfrak{i}} \right)_{n,j}$ is the time derivative of $\rho^{\mathfrak{i}}$ at time $t_n + c_j \Delta t$ (i.e. at the j-th stage). Based on the semi-discrete mass conservation equation (14), it is equal to the numerical approximation of the mass flow rate spatial gradient at this stage. Equation (B.5) can then be interpreted as a constraint on m:

$$-\frac{m^{\mathfrak{i}+1}_{ni} - m^{\mathfrak{i}}_{ni}}{\Delta x^{\mathfrak{i}}} = \frac{\rho^{\mathfrak{i}}_{ni} - \rho^{\mathfrak{i}}_n}{a_{ii} \Delta t} + \sum_{j=1, j \neq i}^{s} \frac{m^{\mathfrak{i}+1}_{nj} - m^{\mathfrak{i}}_{nj}}{\Delta x^{\mathfrak{i}}} \tag{B.6}$$

Comparing this equation to the semi-discrete continuity equation $-(m^{\mathfrak{i}+1}_{ni} - m^{\mathfrak{i}}_{ni})/\Delta x^{\mathfrak{i}} = \left(d_t \rho^{\mathfrak{i}} \right)_{ni}$, we see that the right hand-side is the approximation of the source term $\left(d_t \rho^{\mathfrak{i}} \right)_{ni}$ that is naturally constructed by the Runge-Kutta scheme and which can be entirely expressed in terms of the mass flow rates at various stages.

Appendix C. Verification of the order of convergence

To verify the orders of convergence in time, a simple test case is set up, using the simplified model presented in Section 5. Starting from the steady-state solution at $P = 5.5 \times 10^6$ Pa, a pressure step to $P = 5 \times 10^6$ Pa is applied and the resulting transient is simulated for 10^{-4} s with a fixed time step. The curves of T_s obtained for various time step values are plotted in Figure C.21. The cyan curve represents the most refined solution. A space-averaged relative error, plotted in Figure C.22, is computed on the final mass flow rate field:

$$\epsilon_{\rho u} = \sqrt{\sum_{1}^{N+1} \frac{1}{N} \left(\frac{m_{\mathrm{i}}(t_f; \Delta t) - m_{\mathrm{i}}(t_f; dt_{ref})}{m_{\mathrm{i}}(t_f; \Delta t_{ref})} \right)^2} \tag{C.1}$$

with N the global number of cells and ref denoting the reference simulation. The theoretical orders of convergence are attained as long as the error is not limited by the precision of the Newton algorithm. Similar convergence rates have been observed for the other variables (point-wise or space-averaged errors).

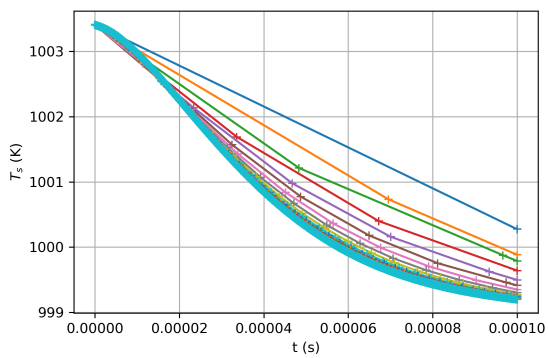


Figure C.21 Surface temperature histories obtained with IE when gradually lowering the time step

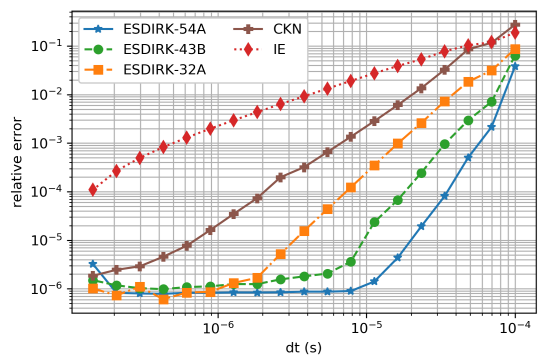


Figure C.22 Global error $\epsilon_{\rho u}$ on the mass flow rate field

Appendix D. Jacobian for the Newton increment

Applying an ESDIRK scheme (stage Equations (26) and (27)) to the system (20)-(21), the i-th stage reads:

$$Q(w_{ni}) = \Delta t a_{ii} f(w_{ni}, z_{ni}) + \overbrace{Q(W^n) + \sum_{j=0}^{i-1} \Delta t a_{ij} f(w_{nj}, z_{nj})}^{b} \tag{D.1}$$

$$0 = g(w_{ni}, z_{ni}) \tag{D.2}$$

The residual vector is therefore $F(w_{ni}, z_{ni}) = (Q(w_{ni}) - \Delta t a_{ii} f(w_{ni}, z_{ni}) - b,\ g(w_{ni}, z_{ni}))^t$. Therefore, the Jacobian $J = \dfrac{\partial F}{\partial(w_{ni},\ z_{ni})}$ needed for the Newton step is:

$$J = \begin{pmatrix} \partial_w Q - \Delta t a_{ii} \partial_w f & -\Delta t a_{ii} \partial_z f \\ \partial_w g & \partial_z g \end{pmatrix} \tag{D.3}$$