An analytical Jacobian approach to sparse reaction kinetics for computationally efficient combustion modelling with large reaction mechanisms

Federico Perini,[*,†,‡] Emanuele Galligani,[¶] and Rolf D. Reitz[‡]

Dipartimento di Ingegneria Meccanica e Civile, Università di Modena e Reggio Emilia, I-41125 Modena, Italy, Engine Research Center, University of Wisconsin–Madison, Madison 53703, USA, and Dipartimento di Matematica pura ed applicata "G. Vitali", Università di Modena e Reggio Emilia, I-41125 Modena, Italy

E-mail: federico.perini@unimore.it

Abstract

This study presents an analytical Jacobian formulation for detailed gas-phase reaction kinetics, suitable for accurate and computationally efficient combustion simulations using either skeletal or detailed reaction mechanisms. A general chemical kinetics initial value problem in constant volume environments is considered, where the gas-phase mixture thermodynamic properties are polynomial functions of temperature according to the JANAF standard. Three different reaction behaviours are accounted for, including modified Arrhenius kinetic law reactions, third-body collisions, and pressure dependent reactions in Lindemann's or Troe's kinetic law forms. The integration of the chemistry ODE system is carried out using a software package specifically developed in Fortran language, and the solution compared to a reference

[*]To whom correspondence should be addressed
[†]University of Modena
[‡]University of Wisconsin–Madison
[¶]University of Modena

chemical kinetics library. Two analytical Jacobian formulations, an exact one and a sparser, approximate one are proposed, and compared to numerical Jacobians computed by finite differences internally generated by a variety of commonly used stiff ordinary differential equations (ODE) solvers. The results show significant reductions in total computational times for the chemistry ranging from factors of 2 to more than two orders of magnitude for 29 species, 56 reactions to 2878 species, 8555 reactions, respectively. Finally, the code has been coupled to an engine combustion simulation software, where at each timestep the chemistry ODE system is integrated in each cell of the computational grid, allowing 77% faster computations with a 160 species combustion mechanism.

Introduction

Recent developments in combustion research show that there is urgent need for incorporating detailed reaction mechanisms in multidimensional simulations.[1] The development of comprehensive reaction mechanisms, consisting of thousands reactions and species, for the oxidation of a variety of fuels has been growing in the past two decades, and has allowed significant insight in many combustion phenomena to be achieved.[2,3] As a matter of fact, most recently developed reaction mechanisms range from about 1000 to more than 3000 species, up to more than 10000 elementary reactions;[4–7] the computational requirements of simulations involving such large mechanisms are huge even for simple batch reactor simulations, or one dimensional steady laminar flame calculations,[8] thus making their adoption practically unviable – even on distributed computational clusters – for most practical combustion systems of industrial interest, whose computational domains typically require from a few thousands to several million cells.[1] A paradigmatic example is that of internal combustion engine research, where since the early 2000's the introduction of necessarily skeletal or reduced chemistry computations in engine computer modelling has nevertheless allowed the study and the development of new, kinetically- or reactivity-driven combustion concepts such as the homogeneous-charge compression ignition (HCCI) and reactivity-controlled compression ignition (RCCI).[9–12]

For these reasons, a number of techniques have been developed in order to reduce the computational efforts needed by the integration of the chemistry system of ordinary differential equations (ODE). Among them, most approaches target the reaction mechanism, aiming at locally reducing its size, complexity or stiffness, while keeping the introduced error under control.[13–19] The present work belongs instead to a smaller class of methods devoted to tailoring the problem formulation and the integration methodology to the solution of the reaction kinetics problem. When dealing with combustion computations, the solution of the stiff ODE system associated with chemical kinetics usually relies on robust solvers using backward differentiation formulae methods[a] (BDF);[20–25] those methods however pursue Newton iterations which require repeated Jacobian matrix evaluations. In order to cope with the increasing demand of this operation with the increase in mechanism size, some Jacobian approaches to chemical kinetics simulations have been developed, aiming at either approximating[26,27] or preconditioning[28–30] the ODE system's Jacobian matrix. The system Jacobian matrix is usually approximated via finite differences, while the direct, analytical formulation is only common in kinetics problems which do not undergo strong temperature-kinetics coupling.[31–33] For practical combustion simulations, instead, mechanism-specific routines can be built by adopting symbolic analysis tools for pre-processing,[34–36] so that the computation is mechanism-specific, and does not allow insight to be gained for improving the overall computational efficiency. These methods typically do not allow the Jacobian matrix to be built adopting sparse matrix algebra,[37,38] which can enable significant computational savings when coupled to sparse ODE solvers.[39,40] Nevertheless, some strategies can be adopted, such as to reduce the number of calls to the ODE system function for evaluating the finite-difference Jacobian when a sparsity pattern is provided.[41] The DAEPACK automatic differentiation library[42,43] can provide a sparse representation of the Jacobian matrix using proprietary subroutines, as for example applied by Schwer et al.[44] to combustion chemistry cases. The library generates routines for evaluating the Jacobian that assemble the single partial derivatives computations into a sparse

[a]Backward differentiation formulae were first proposed by Curtiss and Hirschfelder[20] up to orders 1 and 2, and by Gear[21] up to order 6. The 'stiff' term referred to ordinary differential equations systems has in fact been introduced by Curtiss and Hirschfelder.[20] The first BDF implementation, with variable order and step, is Gear's well known DIFSUB routine (Gear,[21] Section 9.3).

matrix representation. Such approaches however perform item-by-item operations to compute the partial derivatives, and cannot build the final sparse matrix by just exploiting essential sparse matrix-vector and matrix-matrix operations. Also, the forward mode of automatic differentiation procedure is acknowledged to compute the Jacobian in a non-minimum number of operations[42] that can make the overall evaluation computationally expensive when dealing with large systems. Finally, explicit and matrix-free integration methods are recently gaining interest for their ability to pursue the integration of the stiff ODE system without requiring expensive solutions of linear systems of equations (see, for example, the works by Mosbach and Kraft[45] and Lee and Gear[46]). However, implicit methods provide better performance for the accuracy required for practical stiff combustion calculations.[45] A sparse, universal analytical Jacobian evaluation thus appears to be suitable to address the following important issues arising when integrating chemical kinetics for practical combustion problems:

- Scaling of the computational demand for the system's Jacobian matrix of arbitrary homogeneous gas-phase reactors (of the order of n_s^2 when evaluated through finite differences, n_s being the number of gas-phase species);
- Dense matrix storage requirements (also of the order of n_s^2);
- Scaling of the computational costs for matrix factorization (of the order of n_s^3 if dense matrix algebra is employed);
- Exploitation of mechanism sparsity, which is significant even for small ($n_s < 50$) reaction mechanisms.

We thus focus on the development of an exact, analytical Jacobian formulation for arbitrary detailed chemical kinetics initial value problems, considering different reaction behaviours, and including accurate prediction of gas-phase mixture thermodynamic properties according to the JANAF 7-coefficient polynomial standard.[47] The aim is to reduce the cost of simulations with arbitrary reaction mechanisms, and to achieve an efficient scaling of the computational demand with

increasing number of species and reactions. We show that the proposed approach is able to speed up reference calculations from about two times for small, almost dense reaction mechanisms, and up to more than two orders of magnitude for large comprehensive mechanisms.

This paper is organised as follows. In the first paragraph the approach adopted for the evaluation of chemical kinetics in constant-volume adiabatic environments is presented. Then, in the second section the entries of the whole ODE system's sparse Jacobian matrix are derived, in both a complete, exact formulation and under an approximating assumption which enhances the matrix sparsity. Finally, a fast polynomial approach to the evaluation of the complex temperature-dependent functions is presented in the last section.
The validation and the analysis of the proposed analytical Jacobian approach in terms of computational efficiency is presented in the Results section, where an extensive validation testbench has been set up considering three reaction mechanisms of different sizes, ranging from 29 species and 52 reactions to 2878 species and 8555 reactions. 18 reactor configurations are considered for each combustion mechanism, spanning broad ranges of initial reactor pressure, temperature and mixture composition, in order to maximise the variety of stiffness conditions that may occur during the integration of the chemistry ODE system, together with five well established stiff ODE solvers. Finally, the results of the code's performance when coupled to a widely adopted computational fluid dynamics (CFD) code for internal combustion engine simulations are presented. The Appendices report essential details of the approach, including the partial derivatives, and full initialisation details for the test cases.

Description of the chemical kinetics problem

In the present study the chemical kinetics of reactive gaseous mixtures are evaluated using a modular code written in Fortran, which is used both for computing gaseous mixture thermodynamics and finite reaction kinetics.[48] The initial value problem considers time evolution of an adia-

batic, constant-volume, homogeneous gas-phase chemically reacting environment, whose solution is given throughout the integration of a system of ordinary differential equations. The independent variables are the species mass fractions, and the average system temperature.[49]

Mass conservation

The reaction mechanism is represented by a set of n_r reactions involving a total number n_s of chemical species. Each k-th reaction is expressed as:

$$\sum_{i=1}^{n_s} \nu'_{k,i} M_i \rightleftharpoons \sum_{i=1}^{n_s} \nu''_{k,i} M_i, \qquad k = 1, \cdots, n_r, \tag{1}$$

where the stoichiometric coefficients are stored as sparse matrices ν' and ν'' of n_r rows and n_s columns, and M identifies the names of the chemical species in the mechanism. This general reaction kinetics framework leads to a set of n_s ODEs expressing mass conservation for the system:

$$\frac{dY_i}{dt} = \frac{W_i}{\rho} \sum_{k=1}^{n_r} \left(\nu''_{k,i} - \nu'_{k,i}\right) q_k(Y,T), \qquad i = 1, \cdots, n_s, \tag{2}$$

where W_i is the species molecular weight and $\rho > 0$ the average system density. The reaction rates of progress variables, q_k, are generally dependent on temperature T and on the species mass fractions of the involved species. Their behaviour can indeed have different forms depending on their specific interactions. Four different reaction formulations have been considered, for which specific analytical derivatives have then been obtained. Below, we give full details about these formulations, as they are needed in the following for building the analytical Jacobian matrix. First of all, simple reactions follow the plain law of mass action:

$$\begin{aligned} q_k(Y,T) &= \kappa_{f,k} \prod_{i=1}^{n_s} \left(\frac{\rho Y_i}{W_i}\right)^{\nu'_{k,i}} - \kappa_{b,k} \prod_{i=1}^{n_s} \left(\frac{\rho Y_i}{W_i}\right)^{\nu''_{k,i}} \\ &= q_{f,k}(Y,T) - q_{b,k}(Y,T), \end{aligned} \tag{3}$$

where the forward and backward reaction rate constants are temperature-dependent, and follow the modified Arrhenius kinetic law:

$$\kappa_{f,k}(T) = A_k T^{b_k} \exp\left(-\frac{E_k}{RT}\right). \tag{4}$$

Backward reaction rates, in particular, can be expressed similarly to forward reaction rates, but are usually derived from the equilibrium constant:

$$\begin{aligned} \kappa_{b,k}(T) &= \kappa_{f,k}(T)/Kc_{eq,k}(T), \\ Kc_{eq,k}(T) &= \exp\left(-\Delta g_k^0\right)\left(\frac{p_{atm}}{RT}\right)^{\sum_{i=1}^{ns}\left(\nu''_{k,i}-\nu'_{k,i}\right)}, \end{aligned} \tag{5}$$

where $\Delta g_k^0 = \sum_{i=1,n_s}\left(\nu''_{k,i} - \nu'_{k,i}\right) g_i^0$ is the reaction's change in non-dimensional Gibbs free energy, $g_i^0 = H_i^0/(RT) - S_i^0/R$, between reactants and products. In case a non reversible reaction occurs, simply $\kappa_{b,k} = 0$.

In third-body reactions (TB), the presence in the mixture of other molecules than those participating in the reaction itself enhances the reaction speed proportionally to an 'effective' concentration, or molecularity, M_{eff}:

$$\begin{aligned} q_k^{TB}(Y,T) &= M_{eff,k}(Y,T)\, q_k(Y,T), \\ M_{eff,k}(Y,T) &= \sum_{i=1}^{ns}\left(\alpha_{i,k}\frac{\rho Y_i}{W_i}\right) = C_{tot} - \sum_{i=1}^{ns}\left(\beta_{i,k}\frac{\rho Y_i}{W_i}\right), \end{aligned} \tag{6}$$

where in the simplest case each species equally participates to the reaction improvement (i.e. $\alpha_{i,j} = 1$ or $\beta_{i,j} = 1 - \alpha_{i,j} = 0$), but enhanced reaction molecularity coefficients can be defined for some species.

The effective molecularity value is a parameter also in pressure-dependent reactions, where a simple reaction rate, as from Eq. 4, does not suit the reaction behaviour at different pressure values. In this case, two modified Arrhenius formulations describe the high- and low-pressure limits for the forward reaction rates, namely $\kappa_{f,0}$ and $\kappa_{f,\infty}$. The effective reaction rate at a defined pressure

value is hence determined by a weighting factor to the high-pressure limit rate:

$$\kappa_{f,k}^{PD} = \kappa_{f,k,\infty} F_k^{PD}. \tag{7}$$

In simple pressure-dependent reactions, which follow Lindemann's kinetic law form ($PD, Lind$), the weighting factor for pressure dependence is a function of the reduced pressure value, $Pr = \kappa_{f,0} M_{eff} / \kappa_{f,\infty}$ which defines the actual reaction rate by averaging between the two limits. In Troe's kinetic law formulation ($PD, Troe$), instead, a more complex pressure factor multiplies $\kappa_{f,\infty}$, whose detailed description and derivation is given in the Appendix.

$$\begin{aligned} F_k^{PD,Lind} &= Pr_k/(1+Pr_k) = P_{cor,k}, \\ F_k^{PD,Troe} &= Pr_k/(1+Pr_k) \cdot 10^{logF_k^{Troe}} = P_{cor,k} \cdot 10^{logF_k^{Troe}}. \end{aligned} \tag{8}$$

Due to the thermodynamic constraints between forward and backward reaction rates, the overall rate of progress variable for pressure dependent reactions can hence be written as:

$$q_k^{PD}(Y,T) = F_k^{PD}(Y,T)\, q_k(Y,T), \tag{9}$$

where the regular expression q_k is computed using the high-pressure limit rate $\kappa_{f,k,\infty}$ in Equation 3.

Closure equation

The initial value problem (IVP) for the chemically reacting environment needs closure through the energy conservation equation. In this work, the perfectly adiabatic constant-volume reactor is considered, as this condition is the one usually occurring when incorporating the solution of detailed chemistry into multidimensional CFD codes, where the species and internal energy source terms for each cell are computed as part of a subcycling strategy for the whole code. In particular, the internal energy source term gives a change in the average reactor temperature:

$$\frac{dT}{dt}(Y,T) = -\frac{1}{\bar{c}_v(Y,T)} \sum_{i=1}^{n_s} \left(\frac{U_i(T)}{W_i} \frac{dY_i}{dt}(Y,T) \right), \tag{10}$$

where $\bar{c}_v = \sum_{i=1,n_s} (Y_i C_{v,i}/W_i)$ is the mass average mixture specific heat at constant volume, and U_i the molar internal energy values of the species[b].

Equations 2 and 10 thus constitute the autonomous IVP in $n_{eq} = n_s + 1$ problem unknowns, represented by the array $y = [T, Y_1, \cdots, Y_{n_s}]^T$:

$$y' = f(y) = \begin{bmatrix} dT/dt \\ dY_1/dt \\ \cdots \\ dY_{n_s}/dt \end{bmatrix}, \; y(t=0) = y_0. \tag{11}$$

Jacobian formulation

The Jacobian matrix $J = \partial f_i / \partial y_j$ has a sparse bordered structure as highlighted in Figure 1: the first element – i.e. the entry in position (1,1) –, the rest of the first row and of the first column are dense, and contain the derivatives of the temperature and species mass fraction rates of change with respect to temperature, and the derivatives of the temperature rate of change with respect to species mass fractions, respectively. The remaining square part of the Jacobian matrix is instead sparse, and contains the derivatives of the species mass fraction time derivatives with respect to the species mass fractions. As a matter of fact, the sparsity in the Jacobian formulation descends from the sparsity of matrices ν' and ν'', where, considering elementary reactions, usually not more than three different reactants and three different products coexist.

[b]Thermodynamic properties: we use lower-case characters for mass specific quantities, and uppercase characters for molar quantities. Bar expresses mixture average properties, bold indicates column vector or 2-d matrix.

Derivatives with respect to species mass fractions

As far as the derivatives of mass fraction changes of species $i = 1, \cdots, n_s$ with respect to the mass fractions themselves ($j = 1, \cdots, n_s$) are concerned, the rate formulation in Eq. 2, yields, for each species:

$$J_{i+1,j+1} = \frac{\partial}{\partial Y_j}\frac{dY_i}{dt} = \frac{W_i}{\rho}\sum_{k=1}^{n_r}\left(\nu_{k,i}\frac{\partial q_k}{\partial Y_j}\right), \tag{12}$$

where $\nu_{k,i} = \nu''_{k,i} - \nu'_{k,i}$, and the reactor density ρ is constant due to the constant-volume assumption, and each of the reactions in the mechanism verifies atom conservation for all the elements in the reaction.

$$\mathbf{J} = \begin{array}{|c|c|} \hline \textcircled{4}\ \dfrac{\partial \dot{T}}{\partial T} & \partial\dot{T}/\partial Y_j \quad \textcircled{2} \\ \hline \dfrac{\partial \dot{Y}_i}{\partial T} & \dfrac{\partial \dot{Y}_i}{\partial Y_j} \\ \textcircled{3} & \textcircled{1} \\ \hline \end{array}$$

Figure 1: Structure of the Jacobian matrix for the chemical kinetics ODE system: 1. derivatives of species mass fraction change rates with respect to species; 2. derivatives of temperature change with respect to species; 3. derivatives of species mass fraction changes with respect to temperature; 4. temperature change derivative with respect to temperature.

With the modified Arrhenius kinetic law, the partial derivative of the rate of reaction progress variable q_k with respect to species Y_j can be expressed as a function of the same terms which define the rate of progress variable itself. Furthermore, if the species mass fractions are considered to be always greater than zero, the derivative formulation is unique, for all the species mass fraction power factors. From Eq. 3,

$$\begin{aligned}\frac{\partial q_k}{\partial Y_j} &= \frac{1}{Y_j}\left[\nu'_{k,j}\kappa_{f,k}\prod_{r=1}^{n_s}\left(\frac{\rho Y_r}{W_r}\right)^{\nu'_{k,r}} - \nu''_{k,j}\kappa_{b,k}\prod_{s=1}^{n_s}\left(\frac{\rho Y_s}{W_s}\right)^{\nu''_{k,s}}\right] \\ &= \frac{1}{Y_j}\left[\nu'_{k,j}q_{f,k} - \nu''_{k,j}q_{b,k}\right],\ \forall Y_j > 0,\ j = 1,\cdots,n_s\end{aligned} \tag{13}$$

and thus only the terms where either $\nu'_{k,j} \neq 0$ or $\nu''_{k,j} \neq 0$ exist. This formulation is complete for modified Arrhenius kinetic law reactions only, but is computed for all the reactions in the mechanism, since more complex formulations can be written as developments of this simple form. In particular, for more complex reaction behaviours such as third-body reactions, the expression of the effective molecularity is common to all of them. From Equation 6,

$$\frac{\partial q_k^{TB}}{\partial Y_j} = \frac{\partial M_{eff,k}}{\partial Y_j}q_k + M_{eff,k}\frac{\partial q_k}{\partial Y_j}. \tag{14}$$

In order to cope with the effects of the derivative of the effective molecularity term, which linearly depends on each species mass fraction, thus leading to dense lines in the Jacobian matrix for each species participating in third-body reactions, an alternative formulation has been tested, and its form is discussed in detail in the next section.

Pressure dependent reactions behave similarly to third-body reactions, but the multiplying factor to the baseline rate of progress variable is now a function of effective molecularity and of both high- and low-pressure limit rate formulations. From Equation 9:

$$\frac{\partial q_k^{PD}}{\partial Y_j} = \frac{\partial F_k^{PD}}{\partial Y_j}q_k + F_k^{PD}\frac{\partial q_k}{\partial Y_j}. \tag{15}$$

Reactions which follow Lindemann's kinetic law have:

$$\begin{aligned}\frac{\partial F_k^{PD,Lind}}{\partial Y_j} &= \frac{\partial P_{cor,k}}{\partial Y_j} \\ &= \frac{\partial M_{eff,k}}{\partial Y_j}\frac{\kappa_{f,k,0}\,\kappa_{f,k,\infty}}{\left(\kappa_{f,k,0}M_{eff,k}+\kappa_{f,k,\infty}\right)^2},\end{aligned} \tag{16}$$

while reactions under Troe's kinetic law form have:

$$\frac{\partial F_k^{PD,Troe}}{\partial Y_j} = 10^{logF_k^{Troe}}\left[\frac{\partial P_{cor,k}}{\partial Y_j}+\log(10)\,P_{cor,k}\,\frac{\partial logF_k^{Troe}}{\partial \log Pr_k}\,\frac{\partial \log Pr_k}{\partial Y_j}\right]. \tag{17}$$

The dependence of the base-10 logarithm of Troe's centering factor with respect to the logarithm of the reduced pressure value is extensive, and is discussed in the Appendix.

The computation of the sparse central part of the Jacobian matrix (zone 1, Figure 1), also gives the bricks for building the Jacobian's upper border, i.e., the derivatives of temperature variation with respect to species mass fractions. From Eq. 10, the components of block $J_{②}$ are:

$$J_{1,j+1} = \frac{\partial}{\partial Y_j}\left(\frac{dT}{dt}\right) = -\frac{1}{\bar{c}_v}\left[\frac{C_{v,j}}{W_j}\frac{dT}{dt}+\sum_{i=1}^{n_s}\frac{U_i}{W_i}\frac{\partial}{\partial Y_j}\left(\frac{dY_i}{dt}\right)\right], \tag{18}$$

it can be noted that the column vector containing all of the derivatives of temperature change with respect to species can be evaluated as the product

$$\begin{aligned}J_{②}^T &= \left(\frac{\partial}{\partial Y_1}\left(\frac{dT}{dt}\right),\cdots,\frac{\partial}{\partial Y_{n_s}}\left(\frac{dT}{dt}\right)\right)^T \\ &= \frac{\partial}{\partial Y}\left(\frac{dT}{dt}\right) = -\frac{1}{\bar{c}_v}\left(c_v\frac{dT}{dt}+J_{①}^T u\right),\end{aligned} \tag{19}$$

where lower-case $c_{v,j}$ and u_j indicate specific quantities in mass units, and the square block $J_{①}$ is usually very sparse.

Derivatives with respect to temperature

In order to build the first column of the Jacobian matrix, derivatives of the species thermodynamic properties need to be evaluated. In the present approach, the 7-coefficient JANAF standard approach[47] is considered. The detailed polynomial expressions of the derivatives involving the species constant-volume molar specific heat, $\partial C_{v,j}/\partial T$, Gibbs free energy $\partial g_j^0/\partial T$ are reported in the Appendices. Two sets of polynomial coefficients are usually provided for each species, for better fitting of the low- and of the high-temperature ranges. The behaviour of the resulting functions is however of class C^2 in the whole temperature domain, including the switching temperature value. Hence, it is guaranteed that the analytically differentiated functions can apply the same sets of coefficients as the parent functions, and over the same temperature validity ranges.

Concerning the derivatives of species mass fraction rates with respect to temperature, they descend from the temperature derivatives of the reaction rates of the progress variable they are involved in:

$$J_{j+1,1} = \frac{\partial}{\partial T}\left(\frac{dY_j}{dt}\right) = \frac{W_i}{\rho}\sum_{k=1}^{n_r} \nu_{k,j}\frac{\partial q_k}{\partial T}, \tag{20}$$

where, again, the reaction-dependent derivatives $\partial q_k/\partial T$ have a base expression for the modified Arrhenius kinetic law reactions, which can be evolved to consider third-body and pressure-dependent reaction behaviours. The modified Arrhenius formulation considers reaction rates in the form of Eq. 4, thus

$$\frac{\partial \kappa_{f,k}}{\partial T} = \frac{\kappa_{f,k}}{T}\left(b_k + \frac{E_k}{RT}\right). \tag{21}$$

Backward reaction rates derivatives follow the same formulation if they are explicitly expressed in modified Arrhenius form, while, in the case where they are computed from the equilibrium constant,

$$\frac{\partial \kappa_{b,k}}{\partial T} = \frac{1}{Kc_{eq,k}}\left(\frac{\partial \kappa_{f,k}}{\partial T} - \frac{1}{Kc_{eq,k}}\frac{\partial Kc_{eq,k}}{\partial T}\,\kappa_{f,k}\right), \tag{22}$$

where the derivative of the equilibrium constant of reaction k in concentration units only depends on temperature:

$$\frac{\partial K c_{eq,k}}{\partial T} = -K c_{eq,k} \left[\frac{\sum_{j=1}^{n_s} \left(\nu''_{k,j} - \nu'_{k,j} \right)}{T} + \frac{\partial \left(\Delta g_k^0 \right)}{\partial T} \right]. \tag{23}$$

As forward and backward reaction rates are the only temperature-dependent parameters in the rate of progress variable formulation, for constant-volume reactors where the average mixture density is constant, the rate of progress variable derivative yields:

$$\frac{\partial q_k}{\partial T} = \frac{\partial \kappa_{f,k}}{\partial T} \prod_{r=1}^{n_s} \left(\frac{\rho Y_r}{W_r} \right)^{\nu'_{k,r}} - \frac{\partial \kappa_{b,k}}{\partial T} \prod_{s=1}^{n_s} \left(\frac{\rho Y_s}{W_s} \right)^{\nu''_{k,s}}. \tag{24}$$

If a third-body reaction is considered, the rate of progress variable is obtained simply multiplying its baseline formulation by an effective molecularity value, thus

$$\frac{\partial q_k^{TB}}{\partial T} = \frac{\partial q_k}{\partial T} M_{eff,k}, \tag{25}$$

as in constant-volume problems the effective molecularity value of Eq. 6 does not show any dependence on temperature. This is not true in presence of pressure-dependent reactions, where the enhancement factor which multiplies the 'standard' rate of progress variable formulation is a function of the average pressure value, and is thus tightly bound to temperature:

$$\begin{aligned}
\frac{\partial q_k^{PD}}{\partial T} &= q_k \frac{\partial F_k^{PD}}{\partial T} + F_k^{PD} \frac{\partial q_k}{\partial T}; \\
\frac{\partial F_k^{PD,Lind}}{\partial T} &= \frac{M_{eff,k}}{\left(\kappa_{f,k,\infty} + \kappa_{f,k,0} M_{eff,k} \right)^2} \left(\kappa_{f,k,\infty} \frac{\partial \kappa_{f,k,0}}{\partial T} - \kappa_{f,k,0} \frac{\partial \kappa_{f,k,\infty}}{\partial T} \right); \\
\frac{\partial F_k^{PD,Troe}}{\partial T} &= 10^{log F_k^{Troe}} \left(\frac{\partial F_k^{PD,Lind}}{\partial T} + \log(10) \frac{\partial log F_k^{Troe}}{\partial T} \right).
\end{aligned} \tag{26}$$

Once the species derivatives with respect to temperature have been assembled considering the

proper reactions contributions, the first item in the Jacobian matrix can be computed, yielding the derivative of the average mixture temperature change rate with respect to the system temperature itself:

$$\frac{\partial}{\partial T}\left(\frac{dT}{dt}\right) = -\frac{1}{c_v}\left\{\frac{dT}{dt}\frac{\partial c_v}{\partial T} + \sum_{i=1}^{n_s}\left[\frac{1}{W_i}\left(C_{v,i}\frac{dY_i}{dt} + U_i J_{i+1,1}\right)\right]\right\}. \tag{27}$$

Jacobian sparsity pattern

As previously mentioned, the standard expression of the effective molecularity for third-body and pressure dependent reactions involves all the species in the mechanism, and the molecularity coefficients $\alpha_{j,k}$ of Eq. 6 are normally unity, except in the case where the molecular structure of some species can improve the efficiency of inter-molecular collisions in the reactions more than the others. Since the presence of enhanced third-body molecularity coefficients is usually limited to a restricted number of species - not more than five or six species per third-body reaction -, Schwer et al.[44] proposed a more efficient storage of the molecularity coefficient: $\beta_{j,k} = 1 - \alpha_{j,k}$. In this case, only the non-zero entries of $\beta_{j,k}$, corresponding to the species which show an enhancing behaviour towards modifying the average collision frequency need to be stored and used for computing the effective molecularity value:

$$M_{eff,k} = C_{tot} - \sum_{i=1}^{n_s}\beta_{i,k}\left(\frac{\rho Y_{i,k}}{W_i}\right) = \rho\sum_{i=1}^{n_s}\frac{Y_i}{W_i} - \sum_{i=1}^{n_s}\beta_{i,k}\left(\frac{\rho Y_{i,k}}{W_i}\right) \tag{28}$$

As Equation 28 shows, the derivatives of the effective reaction molecularity $M_{eff,k}$ with respect to each of the species mass fractions are always non-zero in constant-volume environments, where the total mixture concentration C_{tot}, in mole units, is not conserved during the time evolution of the adiabatic reactor:

$$\frac{\partial C_{tot}}{\partial Y_j} = \frac{\rho}{W_j} = \frac{1}{RT}\frac{\partial p}{\partial Y_j}. \tag{29}$$

Since this condition would not be present in adiabatic constant pressure environments, where

the average mixture concentration in mole units is constant, we have introduced the approximate, yet sparser Jacobian formulation, which assumes that $\partial C_{tot}/\partial Y_j \approx 0, \forall j = 1, \cdots, n_s$. This formulation does not affect the ODE system evaluation, but only the Jacobian estimation. The rows related to the species which take part into third-body reactions are almost completely cleared, and only the columns containing derivatives with respect to species with enhanced molecularity coefficients remain non-zero. The main aim of this approximate approach is to maximize the computational efficiency of the analytical Jacobian computation, through setting up a much sparser matrix layout. The error introduced by the approximation is expected not to affect too much the integration results when using common stiff ODE solvers, which usually perform simplified Newton iterations through computing the Jacobian only once and then using the saved version at each Newton step (see the VODE integrator,[22] for example).

In Table 1, the features of three widely used reaction mechanisms, related to the oxidation of n-heptane and methyl-decanoate, common surrogate species which well suit the ignition characteristics of diesel and biodiesel fuels, are presented. The mechanism dimensions range from 29 species to 2878 species, the first one well representing a class of skeletal mechanisms commonly adopted in multidimensional CFD codes; the latter being among the widest detailed mechanisms available in the literature involving fuel compounds.[1] A graphical representation of the Jacobian matrix sparsity in the three cases, comparing the complete and the sparser approximate formulations, is shown in Figure 2. Note that the sparsity increases fast for the semi-detailed n-heptane mechanism, where even using the 'complete' third-body formulation, the matrix sparsity reaches 72%. Extreme matrix sparsity is observed in the largest mechanism, where only 0.6% of the elements is non-zero in the sparse third-body formulation. Figure 3 shows the relationship between matrix sparsity and mechanism dimension, where a quasi-logarithmical sparsity trend is observed for the Jacobian using the complete third-body formulation, while a lower sparsity increase rate characterizes the sparse formulation, where the number of third-body reactions in the mechanism also becomes an important factor affecting Jacobian sparsity.

Table 1: Jacobian sparsity features for the constant-volume reactor IVP using the three reaction mechanisms considered, according to the complete and sparse third-body formulations

mechanism	ref.	n_s	n_r	complete form		sparse form	
				blacks	sparsity	blacks	sparsity
ERC n-heptane	[50]	29	52	487	45.9%	412	54.2%
LLNL n-heptane	[51]	160	1540	7361	71.6%	3570	86.2%
LLNL MD	[4]	2878	8555	166703	98.0%	50374	99.4%

The ratio between the number of non-zero elements in the Jacobian matrix according to the two different third-body formulations, also plotted in Figure 3, shows an interesting log-like behaviour among the mechanisms considered, suggesting that the relative impact of the third-body treatment on Jacobian computation and usage increases logarithmically with the number of species (i.e., the Jacobian matrix rank).

Tabulation of temperature-dependent properties

Most temperature-dependent functions, such as species' thermodynamic properties and their derivatives, reaction rates, equilibrium concentrations and falloff parameters, require demanding analytical evaluations such as multiple exponential functions and real number powers; we have introduced an approximate, tabulated approach for them. As a matter of fact, modern compilers' optimization features can make intrinsic evaluation of complex functions such as the exponential operator computationally competitive in comparison with faster, yet much more approximate, techniques,[52,53] but not with tabulated data, which retrieval is still always much more efficient.

Tabulation for species- and reaction-related arrays requires significant memory space, of the order $O(n_s n_{tab})$ or $O(n_r n_{tab})$, respectively, where n_{tab} is the number of tabulated temperature points. In particular, a fixed ΔT table span step width has been chosen, and three different interpolation methods have been considered. When simple, linear interpolation is adopted, the retrieval of tabulated values at the two nearest temperature neighbours is needed, plus a total of four algebraic operations (2 sums, 2 multiplications); referring to a generic function $y = y(x)$, as represented in Figure 4:

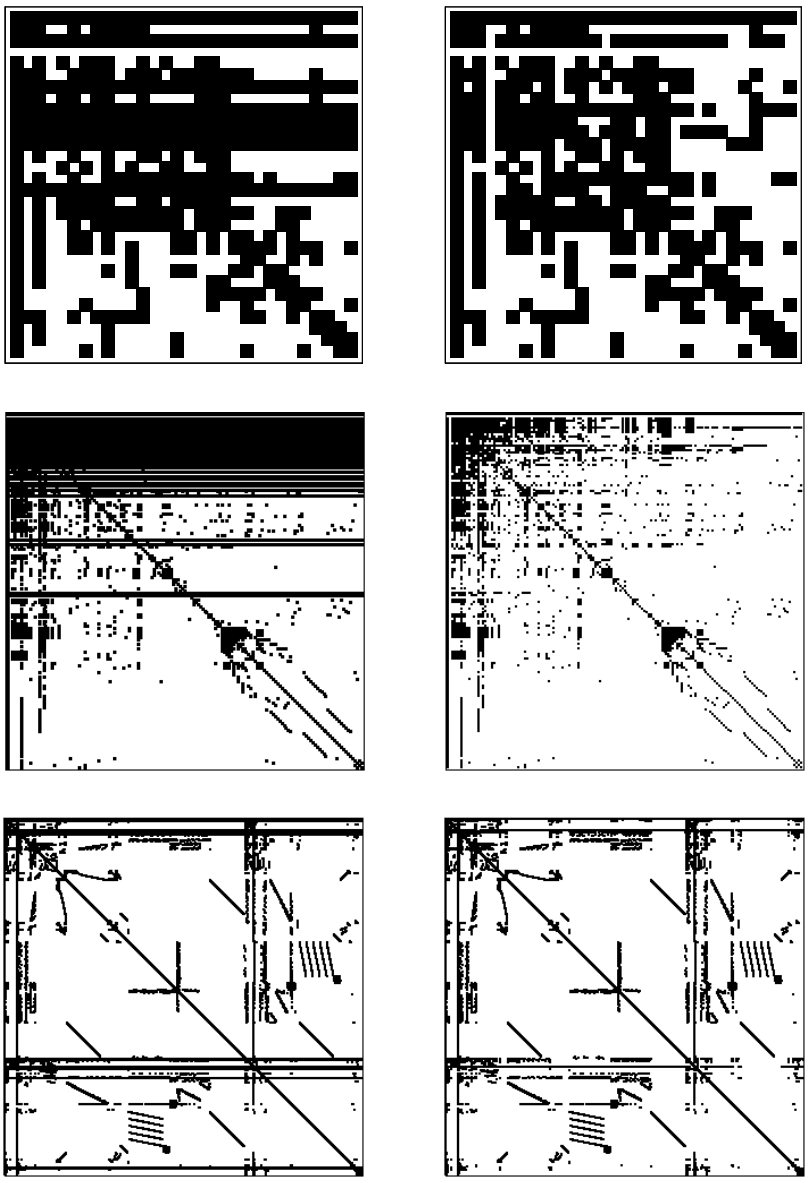

Figure 2: Jacobian matrix sparsity patterns for the three reaction mechanisms considered: ERC n-heptane (top, 29 species), LLNL n-heptane (center, 160 species), LLNL MD (bottom, 2878 species). Complete (left) and sparse (right) third-body formulation.

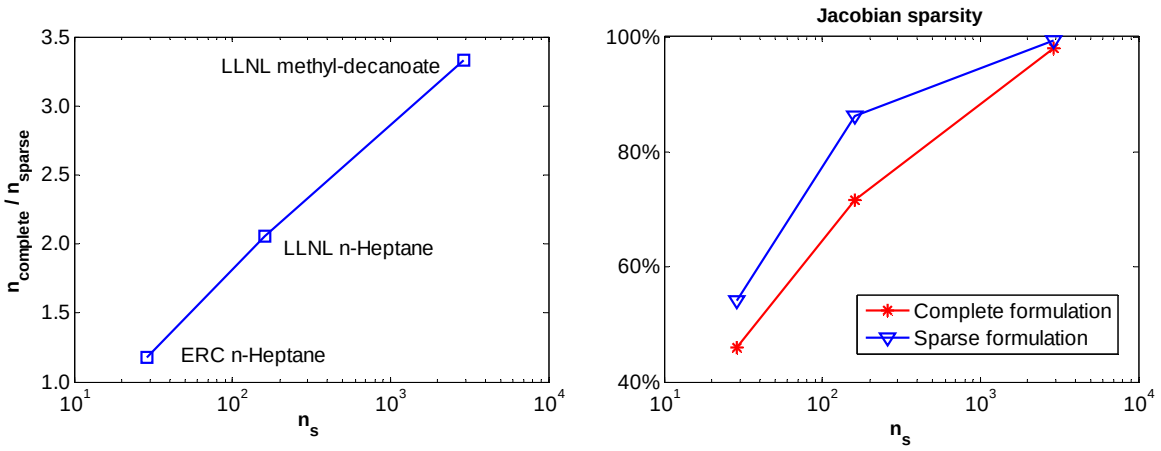

Figure 3: Comparison between complete and approximate Jacobian formulations, for the three mechanisms considered. Left: ratio between non-zero elements in the complete and in the sparser formulation. Right: Jacobian matrix sparsity, percentage of zero elements.

$$f = x - x_0 / (x_1 - x_0), \tag{30}$$
$$y = f\, y_1 + (1 - f)\, y_0;$$

parabolic interpolation requires instead three tabulated value retrievals, plus eleven algebraic operations (5 sums, 6 multiplications):

$$y = \frac{1}{2}\left(f^2 - f\right) y_{-1} + \left(1 - f^2\right) y_0 + \frac{1}{2}\left(f^2 + f\right) y_1. \tag{31}$$

In case the more accurate, yet more computationally demanding degree 4 polynomial interpolation is adopted, 5 points are needed to estimate the function value at point $x = f$, and 32 operations (12 sums, 20 multiplications):

$$
\begin{aligned}
y = y_{-2} \cdot \frac{f}{12}\left(f^2-1\right)\left(\frac{f}{2}-1\right) \\
+ y_{-1} \cdot \frac{f}{3}(f-1)\left(2-\frac{f^2}{2}\right) \\
+ y_0 \cdot \left(\frac{f^2}{4}-1\right)\left(f^2-1\right) \\
+ y_1 \cdot \frac{f}{3}(f+1)\left(2-\frac{f^2}{2}\right) \\
+ y_2 \cdot \frac{f}{12}\left(f^2-1\right)\left(\frac{f}{2}+1\right).
\end{aligned}
\tag{32}
$$

Figure 5 shows the interpolation error of three relevant thermodynamic functions for the carbon dioxide species in the reaction of carbon monoxide with hydroxyl radical: species internal energy, reaction equilibrium constant and forward reaction rate. The plots concerning each of the interpolated methods are for data tabulation at fixed $\Delta T = 10K$, and plot the maximum interpolation errors occurring at maximum distance between tabulation points. It can be observed that, whilst linear and parabolic interpolation techniques yield standard relative accuracies, with relative errors being significantly higher than the double precision machine accuracy, the degree-4 interpolation is much more accurate, shrinking errors up to the order of machine precision. Also, even a wide $\Delta T = 10K$ fixed sampling interval step is shown to be highly accurate, yielding relative errors always lower than 10^{-10} for the forward reaction rate constants, which values span more than three orders of magnitude in the evaluated temperature range. Considering that a full constant-volume reactor simulation with analytical Jacobian computation includes 3 species-related and 7 reaction-related temperature dependent quantities, which can be tabulated for computational efficiency, a total storage requirement of the order $O((3n_s + 7n_r)n_{tab})$ needs to be provided whatever the interpolation method.

As represented in Figure 6, all of the interpolation methods provide an almost one-order-of-magnitude wide computational time saving with respect to the exact, analytical computation of

the three tested temperature-dependent quantities. For this reason, the most accurate, degree 4 interpolation method has been chosen, with almost the same computational needs as the other interpolation methods, and with the same storage requirements. A $\Delta T = 10K$ tabulation step, in the overall temperature interval $[500, 3500]\,K$ has been chosen to provide better tradeoff between interpolation error and memory storage requirements for all the variables to be tabulated. The high accuracy degree of the 4th-order polynomial interpolation method also suggests that more accurate interpolation techniques, which also require an evaluation of the function derivative in the tabulation points, are less suitable as they would need twice the memory storage requirements than standard polynomial interpolation.

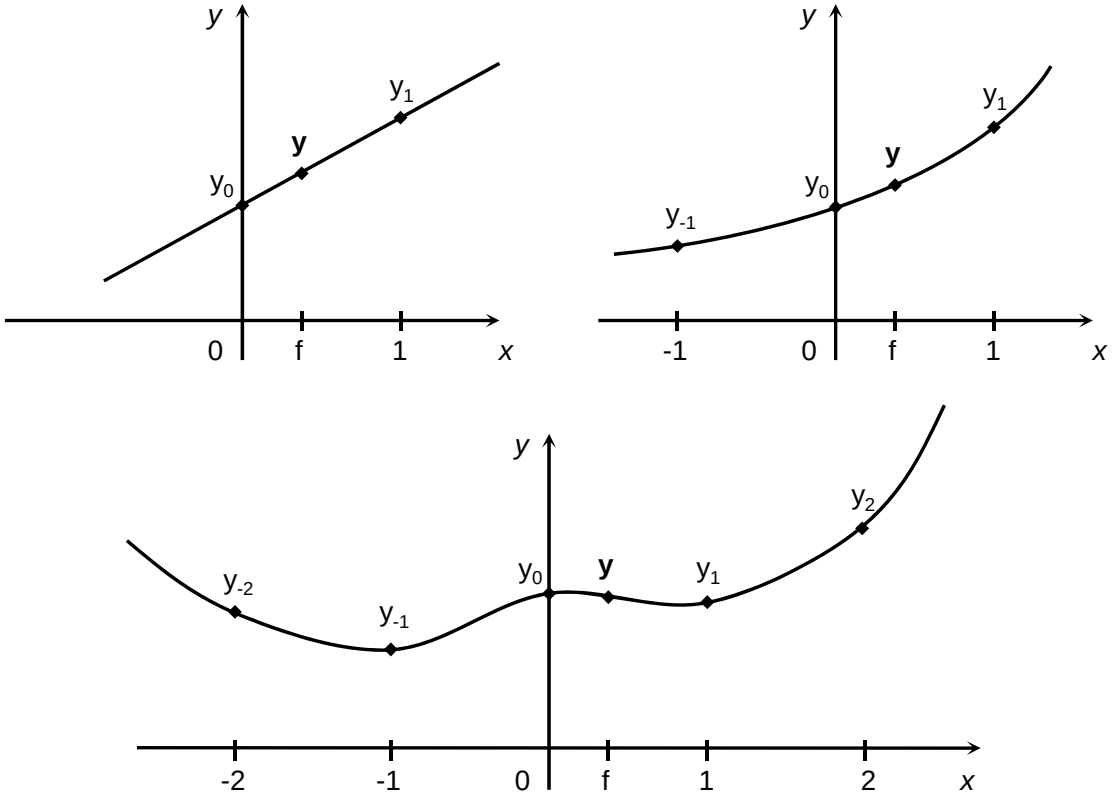

Figure 4: Interpolation of tabulated data with constant sampling step: (in clockwise order) linear interpolation, parabolic interpolation, degree 4 interpolation.

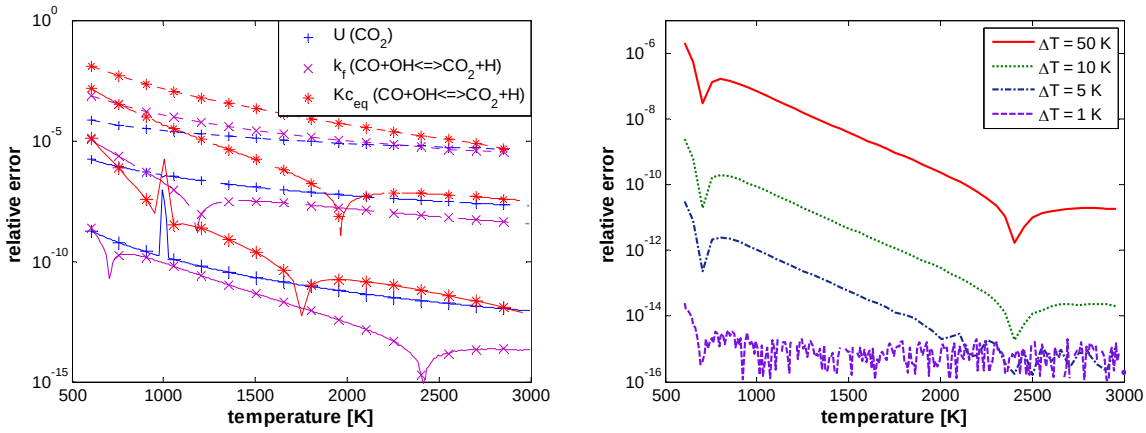

Figure 5: Left: interpolation errors of relevant temperature-dependent properties, namely internal energy $U[J/K/mol]$ of carbon dioxide, forward reaction rate constant and equilibrium constant in concentration units for the reaction $CO + OH <=> CO_2 + H$. Dotted lines: linear interpolation; dashed lines: parabolic interpolation; solid lines: degree-4 interpolation. Right: impact of tabulation step width onto reaction rate constant degree-4-interpolated evaluation error.

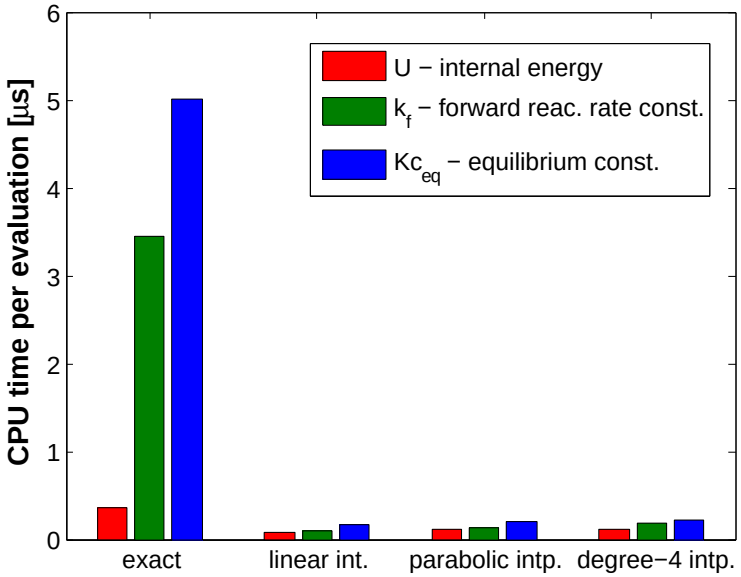

Figure 6: Computation time for evaluating some thermodynamic properties of $n_s = 29$ species, $n_r = 52$ reactions. Exact analytical computatiton vs. tabulated data interpolation (linear, parabolic, degree 4).

Results

Simulation setup

The analytical Jacobian formulation for the constant-volume problem has been implemented into a software package for chemical kinetics computations, whose further details and validation have already been published.[48] In particular, in order to test the accuracy and the computational efficiency of the proposed approach, the three benchmark mechanisms of Table 1 have been adopted. The first one by Patel et al.[50] is a skeletal mechanism for n-heptane ignition consisting of 29 species and 52 reactions, used in internal combustion engine CFD simulations (e.g., Li and Kong[54] and Park and Reitz[55]), and is representative of small mechanisms, for which the Jacobian matrix is scarcely sparse, and where the finite difference Jacobian approximation pursued by most ODE solvers is already a fairly efficient approach. The second mechanism[51] is a mid-size mechanism consisting of 160 species and 1540 reactions, which provides a realistic degree of detail of n-heptane chemistry. Due to its size, this mechanism has been be adopted in simplified, multi-zone or two-dimensional simulations of kinetically-driven HCCI combustion (Guo et al.,[56] Hoffman and Abraham[57]), but needs to be reduced for practical internal combustion engine multidimensional simulations with tens to hundreds of thousands of cells.[58] The last, full methyl-decanoate reaction mechanism by Herbinet et al.,[4] consists of 2878 species and 8555 reactions, and represents a state-of-the-art mechanism suitable for autoignition and flame simulations.

In order to provide a common simulation landscape for each of the mechanisms, 18 initial value problems are simulated for each of them, corresponding to a matrix of cases involving two initial reactor pressure values ($p_0 \in \{2.0; 20.0\}\, bar$), three temperature values ($T_0 \in \{750; 1000; 1500\}\, K$), and three initial lambda values of the air-fuel mixture ($\lambda \in \{0.5; 1.0; 2.0\}$).[59,60] Each ignition case is finally integrated for a time interval corresponding to about 1.5 times the mixture autoignition time. Full details of the initialization matrix, including the species mass fractions, are given in the Simulation Details Appendix. Each IVP integration interval has been subdivided into 100

partial steps in order to provide detailed time output in terms of reactor temperature and species concentrations. This also simulates the integration conditions of the chemistry ODE system when coupled to multidimensional CFD codes, where the species and internal energy source terms of each cell are evaluated as part of an operator-splitting method,[61] and thus the ODE solver is bound to a usually small, fixed integration interval, of the order of 10^{-7} to 10^{-6} seconds.

ODE solver performance

The validation of the proposed formulation has been carried out by testing its performance when integrated through five different ODE solver packages tailored for stiff ODE systems, and compared to a reference solution provided by the CHEMKIN-II package,[62] and integrated using VODE.[22] This choice has been made since CHEMKIN-II is a widely used open-source general purpose code that has been applied to a variety of combustion chemistry simulations, including being integrated in engine simulation codes, and thus it provides a suitable solution against which to compare other codes. Access to the source code in CHEMKIN-II also makes it possible to directly compare its performance by having control on the ODE solver integration parameters, the initialization overhead and the solution output.

Relative and absolute integration error tolerances have been set at $RTOL = 10^{-4}$, $ATOL = 10^{-13}$, equal for each integrated variable. As for the solver choice, the two LSODE solvers, in full and sparse (LSODES) version,[23] Hairer and Wanner's implicit Runge-Kutta solver of order 5 RADAU5,[63] the well known variable-coefficient VODE solver,[22] and the DASPK integrator for large systems of differential-algebraic equations,[64] have been chosen. As far as DASPK is concerned, the system of ODEs has been written in differential-algebraic residual form: $G(t, y, \dot{y}) = \dot{y} - f(t, y) = 0$. This solver's option to solve the linear system using the GMRES algorithm was not used, as it requires the choice of a suitable preconditioner, that would make the integration not comparable to the other solvers, which use direct solution by matrix decomposition.

The 18 IVP cases were run for each solver and combustion mechanism on an Intel core i7 920

-powered machine, with 8GB RAM memory, yielding a matrix of 270 cases to be compared.

First of all, the code's validation in analytical Jacobian form is provided in Figure 7, where the simulation case number 2 is compared for each mechanism with the corresponding CHEMKIN integration. It is an initial low temperature, low pressure case, where the system's stiffness is at its peak. In the plots, the actual integration configuration considers analytical Jacobians in the sparser formulation, with degree 4 interpolation of tabulated temperature-dependent quantities, showing excellent agreement between the time integration of the species' mass fractions, and the corresponding values arising from the CHEMKIN reference simulations.

Figure 8 shows the solver comparison, in terms of overall CPU time summed over all the 18 cases. It is seen that the present code, with VODE integration, has similar or better performance than the CHEMKIN (VODE, too) even in the simplest form (i.e., exact temperature-dependent calculations and finite-difference Jacobian form). When the analytical Jacobian formulation is chosen, a significant improvement in total CPU times is observed, ranging from about 2 times with the smallest mechanism, up to about 140 times with the huge one. The best simulation setup is always the one considering sparse, analytical Jacobian, and interpolated functions.

As for the solver choice, it appears that the VODE and RADAU5 solvers provide optimum performance when dealing with the smallest, almost dense mechanism, where the total integration times are at least about 25% less than when using any other solver. When using medium and large mechanisms, instead, LSODES, the only solver in the set which uses sparse algebra for all the internal operations, including computationally expensive matrix decompositions, always outperforms every other solver, requiring from a minimum of 41% less CPU time (LLNL n-heptane mechanism) up to reducing total time by more than two orders of magnitude (MD mechanism), although its solution procedure is more dated.[25]

Performance of the DASPK integrator is shown to be similar to that of the LSODE solver; moreover, the overall CPU times for the medium and large mechanisms are of the same order of the other solvers which make use of full matrix algebra.

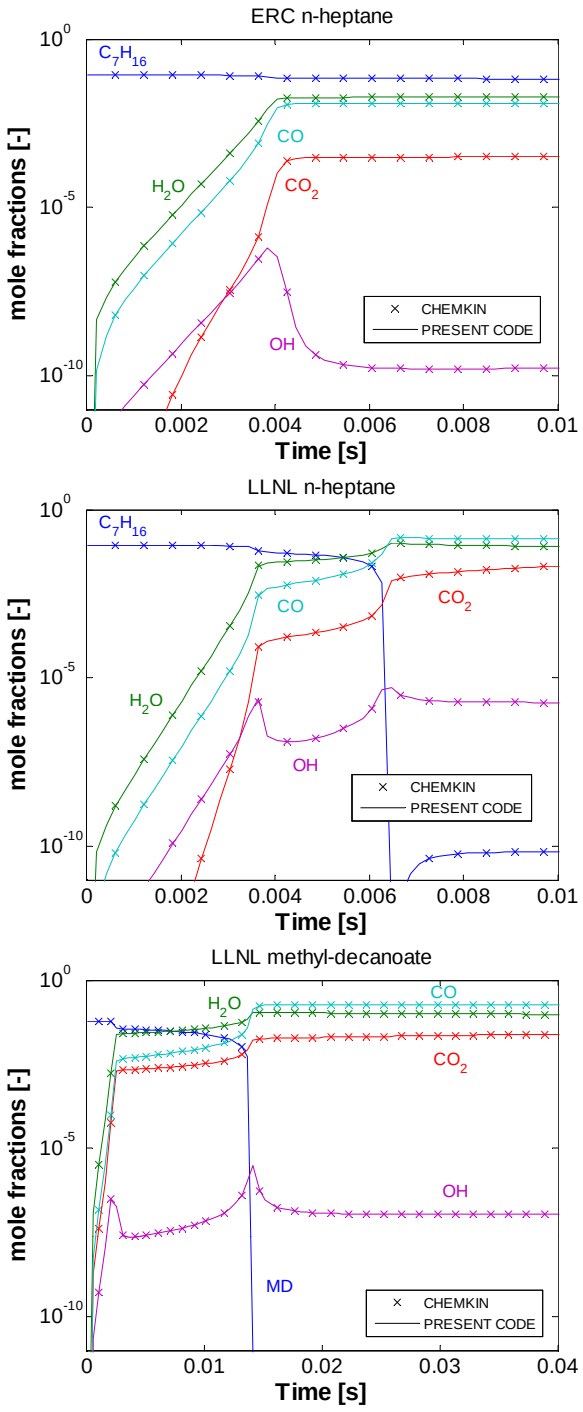

Figure 7: Comparison between simulated time evolution of important species' mass fractions using the three combustion mechanisms: present code using sparse, analytical Jacobian formulation vs. CHEMKIN. IVP initialization: air-fuel mixture equivalence ratio $\phi = 1.0$, initial temperature $T_0 = 750K$, initial reactor pressure $p_0 = 2.0bar$. Details of all the species molar fractions in Table 3.

As for the internal integration details, Figures 9 and 10 show the counts of overall ODE function and Jacobian matrix evaluations required by the solver to complete the integration. In detail, most differences between the cases descend from the different approach each solver has to the integration. The RADAU5 implicit solver has on one hand really wide step sizes, but its high accuracy order limits its computational efficiency as many function evaluations per step are needed. The other solvers show similar numbers of integration steps, with the VODE being the least computationally demanding in terms of Jacobian and function evaluations, thanks to variable order and approximated Newton procedure. The number of residual function and Jacobian evaluations needed by the DASPK integrator is at the upper part of the comparison with the other mechanisms. In particular, at the low- and medium-size mechanisms the integration required approximately 20% to 25% more residual function integrations, which lead to slightly greater overall CPU times.

As fas as polynomial interpolation for temperature-dependent functions is concerned, Tables 4,5,6 show that the accuracy of the interpolated functions is enough not to loosen the computational stability of the integration algorithms. The number of steps needed for the integrations, where tabulated thermodynamic properties are adopted, is almost unchanged in comparison to the corresponding cases which exploit exact analytical properties evaluation. Also, the CPU times needed at the mechanism initialization stage, for tabulating thermodynamic properties, have been measured for each of the three mechanisms tested. They added up to about 0.004 s, 0.104 s and 0.848 s for the ERC n-heptane, LLNL n-heptane and LLNL MD mechanisms, respectively; i.e., their impact in terms of CPU time has never been greater than 0.56% of the total integration time of the fastest solver configuration.

Figure 11 finally shows the CPU times needed for the evaluation of the ODE system function and of the Jacobian matrix. It appears that the computational benefit due to interpolation is dominant in the ODE function evaluation, while its impact is much lower on the evaluation of the

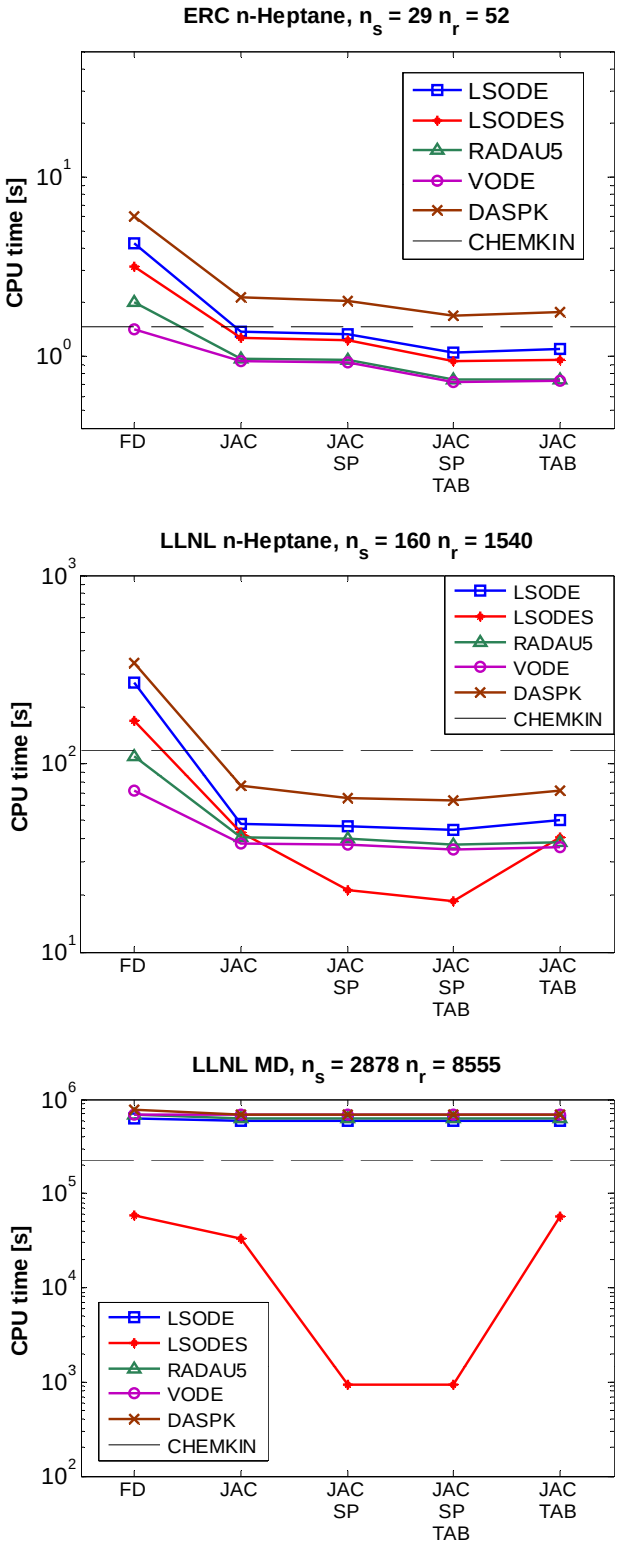

Figure 8: Solver performance comparison for each mechanism considered: total CPU times (s) per case. Details of the 18 initialisation cases in Table 3.

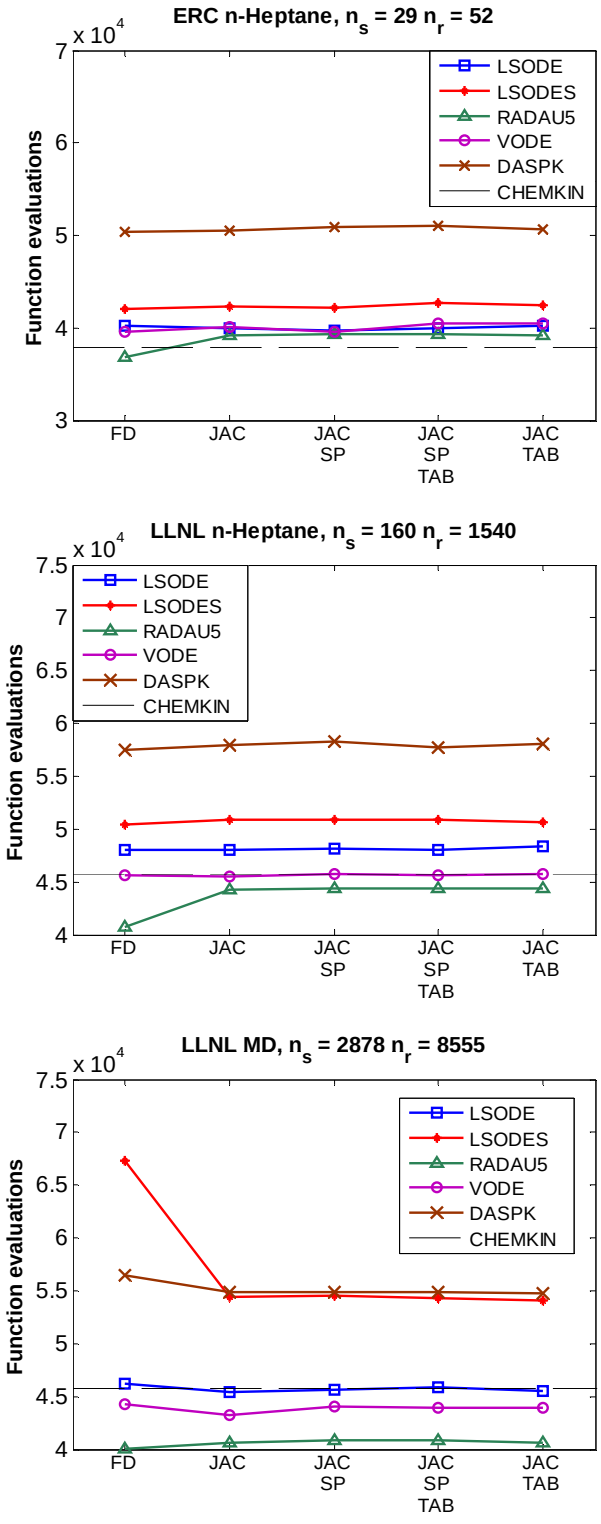

Figure 9: Solver performance comparison for each mechanism considered: number of function evaluations. Details of the 18 initialisation cases in Table 3.

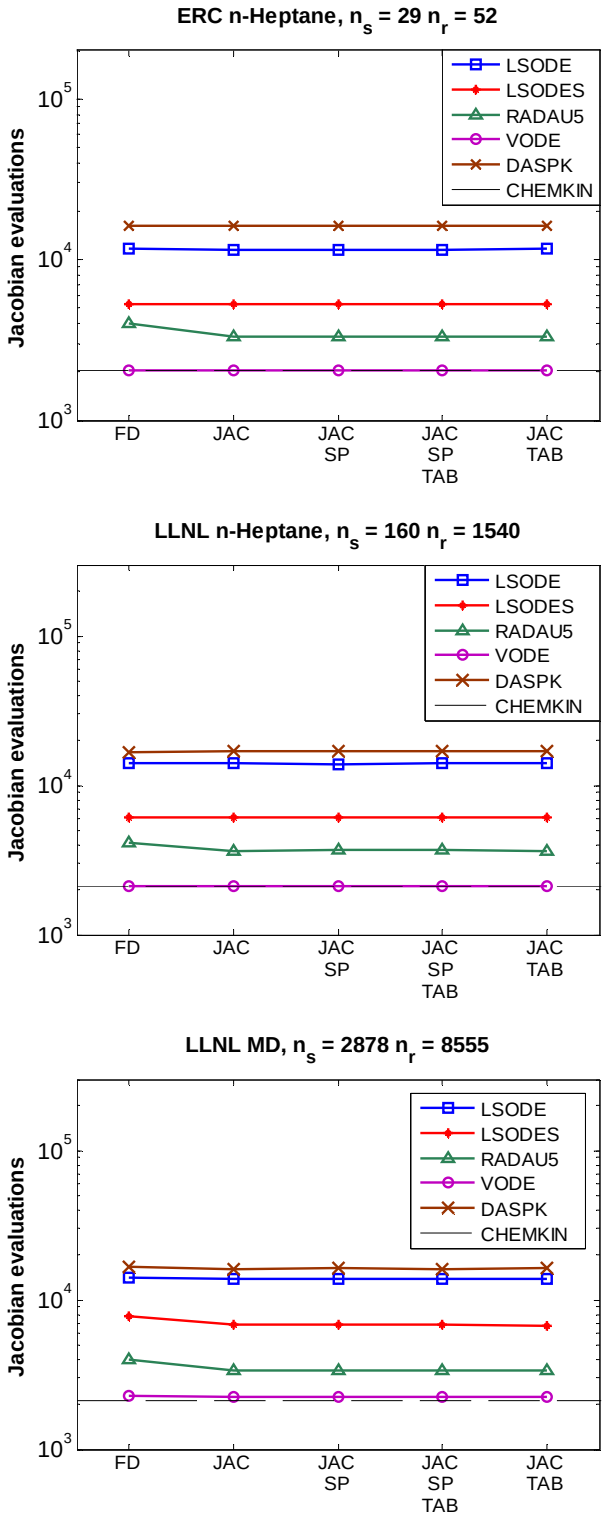

Figure 10: Solver performance comparison for each mechanism considered: number of Jacobian evaluations. Details of the 18 initialisation cases in Table 3.

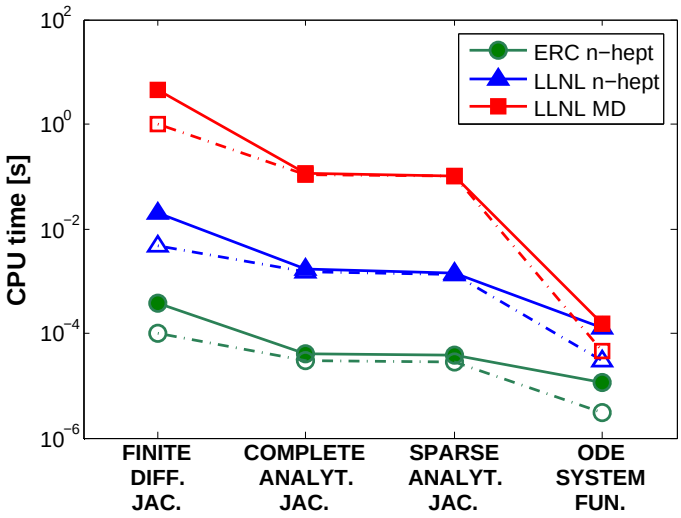

Figure 11: CPU time comparison for the evaluation of chemistry ODE system function and Jacobian matrix. Solid lines and filled marks indicate algebraic evaluation of temperature-dependent parameters, dash-dot lines and empty marks indicate tabulated temperature-dependent parameters.

Jacobian matrix, where also the difference in computational time between the complete and the sparser formulation is negligible. Even though the sparser formulation does not include the contribution of total mixture concentration to the derivative of the effective molecularity with respect to species, which is a $n_s \times 1$ array, the huge reduction in computational times achieved when using the sparse Jacobian formulation is mainly due to a far less expensive matrix decomposition evaluation.

This behaviour is observed also when considering how the total CPU time savings are subdivided between the three main features considered: adoption of analytical Jacobian, sparser formulation for third-body reactions, and temperature-dependent data interpolation. From the analysis shown in Figure 12, the analytical Jacobian formulation accomplishes for most of the CPU time saving at all practical mechanism sizes (i.e., $n_s < 1000$), where it can reach more than 80% of the total speedup. The sparser three-body reactions formulation becomes very effective at the very large mechanism sizes, where it can account for more than 50% the total speedup time, while its effect is negligible for small and almost dense reaction mechanisms. Finally, as seen, polynomial interpolation significantly reduces the CPU times for the evaluation of the ODE system function and of the Jacobian matrix, and thus is significant in relative terms especially for the small mech-

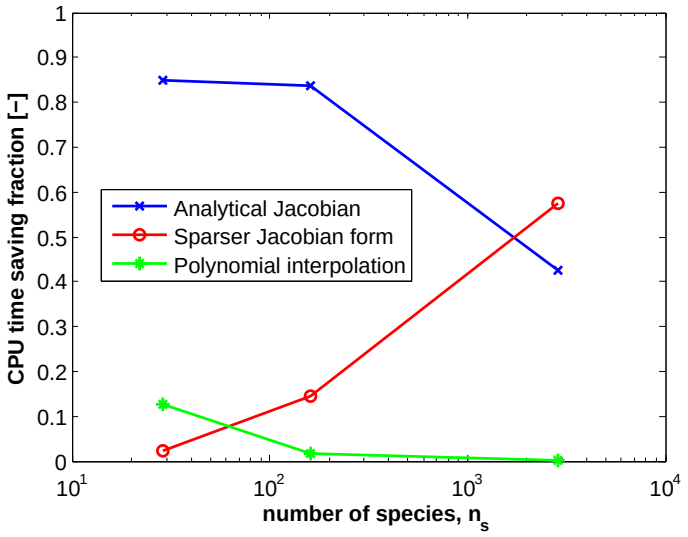

Figure 12: Subdivision of total CPU time savings between adoption of analytical Jacobian estimation, sparser third-body reactions formulation, and tabulation of temperature-dependent quantities, in comparison with the reference finite difference solution. LSODES solver used, RTOL = $1.0e-04$, ATOL = $1.0e-15$.

anism, where the solution of the linear system associated to the iterations of the implicit method is less demanding. Overall, it can account for up to about 15% of the total integration speedup. At large mechanism sizes the relative impact of tabulation becomes lower in comparison to the other approaches, because the linear system algebra is very demanding, and the related CPU time reduction is constant with the number of function and Jacobian evaluations.

Impact of relative integration tolerance.

The tested ODE solvers estimate the local truncation error (i.e., the difference between the exact solution and the numerical solution at time t) limit as $elim_i(t_n) = \mathrm{RTOL}_i \cdot |y_i(t_{n-1})| + \mathrm{ATOL}_i$,[22,24,63,64] where RTOL and ATOL have been considered the same for all the unknowns (i.e., $\mathrm{RTOL}_i = \mathrm{RTOL}$, $\mathrm{ATOL}_i = \mathrm{ATOL}$). The effects of the relative tolerance input value on the accuracy of predicted temperature and species mass fractions for practical cases were investigated in detail. In particular, the 18 LLNL n-heptane mechanism IVP cases were chosen as the reference, the integration being performed using the LSODES solver. Two error cost functions were defined for both tempera-

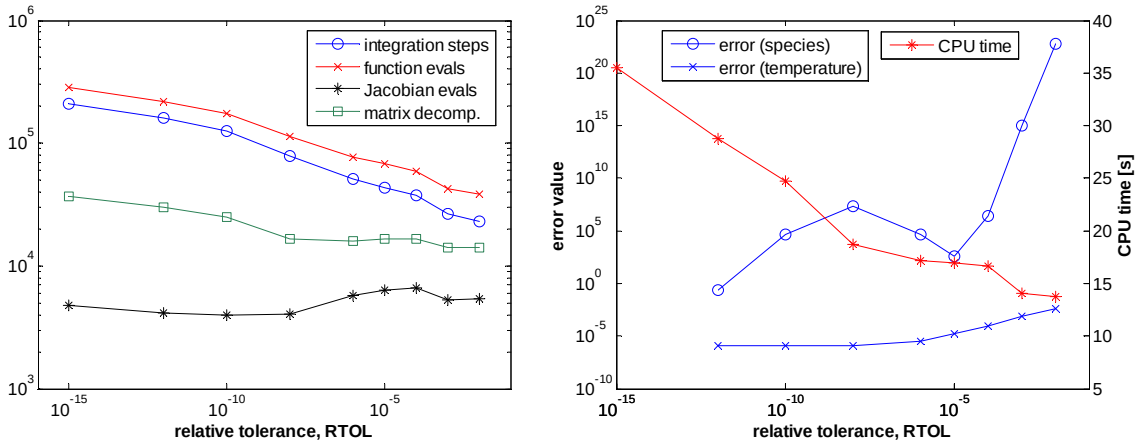

Figure 13: Impact of relative tolerance value on the integration performance, using 18 IVP cases, LLNL n-heptane mechanism, and LSODES solver. (left) error function values and CPU times comparison; (right) integration metrics.

ture and species mass fractions. They compare the current integration results to the solution at RTOL $= 1.0d-15$ (i.e., the closest order to machine round off, in double precision arithmetic), chosen as the most accurate solution. The error of each configuration has been defined as the sum, over the whole integration interval and for all the cases, of the average relative differences of temperature and species mass fractions:[59]

$$e_T = \sum_{i=1}^{n_{\text{cases}}} \frac{1}{t_i} \int_{\tau=0}^{\tau=t_i} \left| \frac{T_i(\tau) - T_i^0(\tau)}{T_i^0(\tau)} \right| d\tau; \tag{33}$$

$$e_Y = \sum_{i=1}^{n_{\text{cases}}} \frac{1}{t_i} \int_{\tau=0}^{\tau=t_i} \sum_{j=1}^{n_s} \left| \frac{Y_{j,i}(\tau) - Y_{j,i}^0(\tau)}{Y_{j,i}^0(\tau)} \right| d\tau.$$

The results of the analysis are plotted in Figure 13. The error function values show that the solution at RTOL $= 1.0e-4$ provides species mass fraction accuracy of similar order as the solutions up to RTOL $= 1.0e-10$; the total error on temperature appears instead less affected by the tolerance threshold. Overall, only the solution at RTOL $= 1.0e-2$ shows deviations from the ones with stricter tolerances. For example, Figure 14 plots predicted temperature profiles for the IVP integration case of Figure 7 at different relative tolerance values.

Figure 13 also shows that the reduction of the relative tolerance monotonically increases the num-

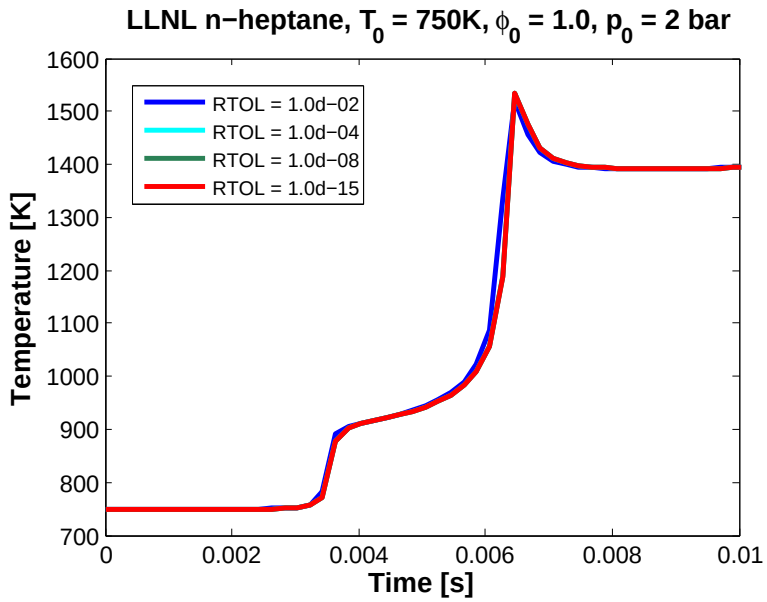

Figure 14: Predicted temperature profiles for the IVP case of Figure 7, LLNL n-heptane mechanism, at different relative integration tolerance values.

ber of integration steps (and, similarly, of evaluations of the ODE system function), as the constraint for accepting a step on the local truncation error value becomes stricter. The corresponding increase in the number of Jacobian matrix evaluations is instead smaller. This indicates that smaller and more frequent integration steps, pursued at smaller relative tolerances, do not always make the LSODES advancement algorithm require an update to the approximate Newton iterations' Jacobian matrix. Only in the RTOL range between $1.0e-04$ and $1.0e-06$, a greater number of Jacobian matrix evaluations also appears to lead to a more accurate prediction in species mass fractions.

Comparison with other codes

The established performance of the proposed approach was compared to other available codes. In particular, the same simulations per mechanism were run with the commercial code CHEMKIN-PRO,[65] and the open-source object-oriented package Cantera.[66] A direct performance comparison among these different programs is not possible, due to different management of IVP case input, program output, since it is not possible to access all the solver parameters. However, the same solution setup on each of the three codes were reproduced, i.e., each of the 18 IVP cases was

Table 2: Performance comparison among the present code, CHEMKIN-II, CHEMKIN-PRO, and Cantera. Overall times for 18 IVP cases.

Code	CPU Time [s]		
	ERC n-heptane	LLNL n-heptane	LLNL MD
Present code	2.6	52.0	2312.0
CHEMKIN-II	6.2	476.9	896099.
Cantera	4.5	405.7	533939.
CHEMKIN-PRO	16.0	46.0	2446.0

subdivided into 100 continuation runs, and hard-drive-intensive solution output was suppressed. All the codes were run on a Pentium IV 3 GHz desktop machine. The performance comparison results are summed up in Table 2. Apart from the small mechanism, where the performance of CHEMKIN-PRO shows some initialization overhead, the presented solution approach yields very similar CPU time performance to CHEMKIN-PRO for the two large mechanisms. No information is available on how the preconditioned Krylov subspace method capability of the DASPK solver – which appears to be used from the output of the CHEMKIN-PRO package – is implemented, and if and which preconditioner is used. It is also not possible to compare the required numbers of integration steps and of function and Jacobian evaluations per simulation. Finally, the Cantera package, that implements the C language version of the VODE solver, appears less suitable for simulations with large mechanisms, where management of the Jacobian matrix in full form leads to significantly higher CPU times than the other two codes.

Internal combustion engine simulations.

In order to complete the validation of the accuracy and computational efficiency of the proposed analytical Jacobian approach, the code has been coupled with the KIVA-4 code,[61] for providing engine CFD simulations with gas-phase reaction kinetics. In the code, a chemical kinetics ODE system is evaluated in each cell of the computational grid and at each overall advancement time-step. The chemistry solver interacts with the code by directly integrating the ODE system for combustion chemistry over a time interval equal to the computational time-step defined by the

fluid flow solver. After the integration, the rates of change of species mass fractions due to chemistry are passed to the fluid flow solver, where they are introduced as part of an operator-splitting approach. The effects of turbulence are accounted for at the resolved scales by the standard RNG $k-\varepsilon$ turbulence model, in terms of transport, mixing, and energy fluxes. This approach has been shown to be accurate by Kokjohn and Reitz[67] for predicting local mixture ignition and dynamics for high- and low-temperature combustion regimes, including the short-delay high-temperature ignition occurring in conventional direct injected diesel engines.

The validation of the chemistry solver coupling has been carried out considering a direct injected diesel engine, featuring turbocharged air intake and a common-rail injection system working at $p = 1600bar$ target operating pressure. The full geometrical and operating details are reported in the paper by Golovitchev et al.[68] A computational grid consisting of 24780 cells at bottom dead centre has been considered, and a reference full-load, maximum engine speed operating condition has been chosen, where a single diesel fuel injection pulse is delivered into the combustion chamber 25.20 crank angle degrees before top dead centre. Figure 15 shows the case's validation using either CHEMKIN or the present code for the detailed chemistry integration, adopting the skeletal ERC n-heptane combustion mechanism.[50] A high degree of accuracy in terms of both in-cylinder pressure trace and instantaneous apparent rate of heat release is observed, with no differences between the two chemistry solvers. The temperature distribution in a vertical domain cross section is shown in Figure 16, showing the computational accuracy of the present analytical Jacobian approach. The same simulation setup has also been run with the LLNL n-heptane mechanism for combustion chemistry. The overall computational times of the simulations are shown in Figure 17, where significant time savings can be noted. The overall reductions in total CPU time are from about 36% up to about 77% despite the fixed amount of time needed by the flow field solution with a high number of species. This shows the possibility to adopt a semi-detailed reaction mechanism in internal combustion engine simulations, which can be completed in a reasonable amount of time (i.e., approximately 60 hours, in this case), and without needing to reduce the combustion

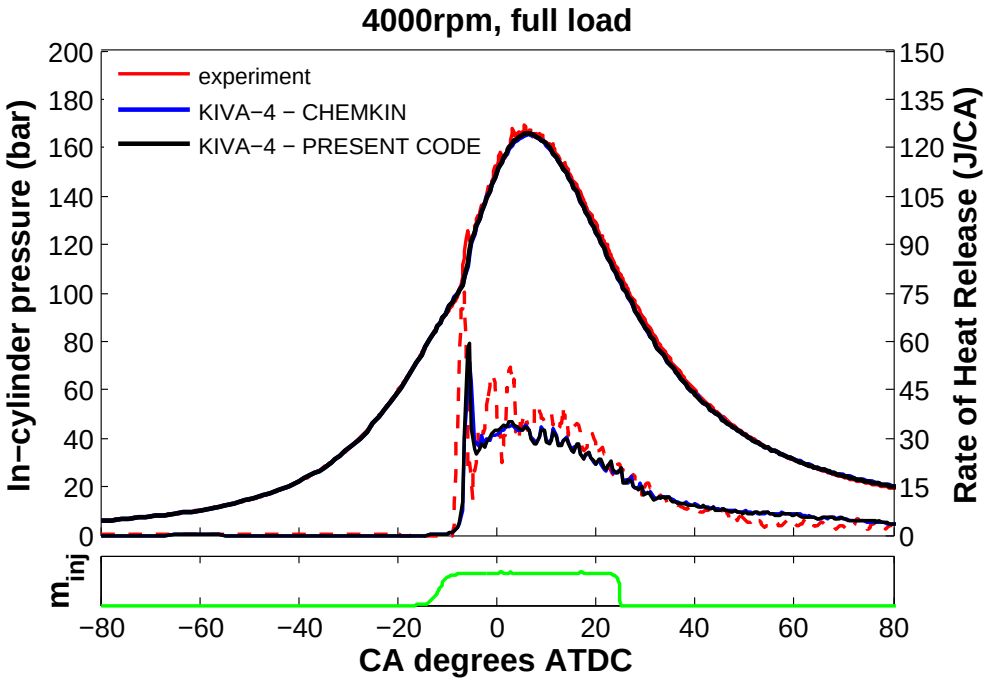

Figure 15: Comparison between KIVA-4[61] calculation of a small, DI diesel engine of current production. Chemkin[62] vs. present chemistry code coupling, solver = VODE with analytical sparse Jacobian, and tabulated thermodynamic properties.

mechanism further.

Concluding remarks

An analytical Jacobian approach for the efficient solution of general chemical kinetics initial value problems involving arbitrary reaction mechanisms for the computation of combusting mixtures has been developed. The methodology features evaluation of gas-phase mixture properties through polynomial coefficients in JANAF standard format, and the computation of reaction rate constants in various forms, including simple reactions following the modified Arrhenius kinetic law, third-body reactions, and falloff reactions following Lindemann's and Troe's kinetic law forms.

An exact analytical formulation for the constant-volume adiabatic reactor, including the ODE system's Jacobian matrix, has been derived, and a further approximate form has been adopted, which significantly increases the matrix sparsity pattern. Interpolation of tabulated temperature-

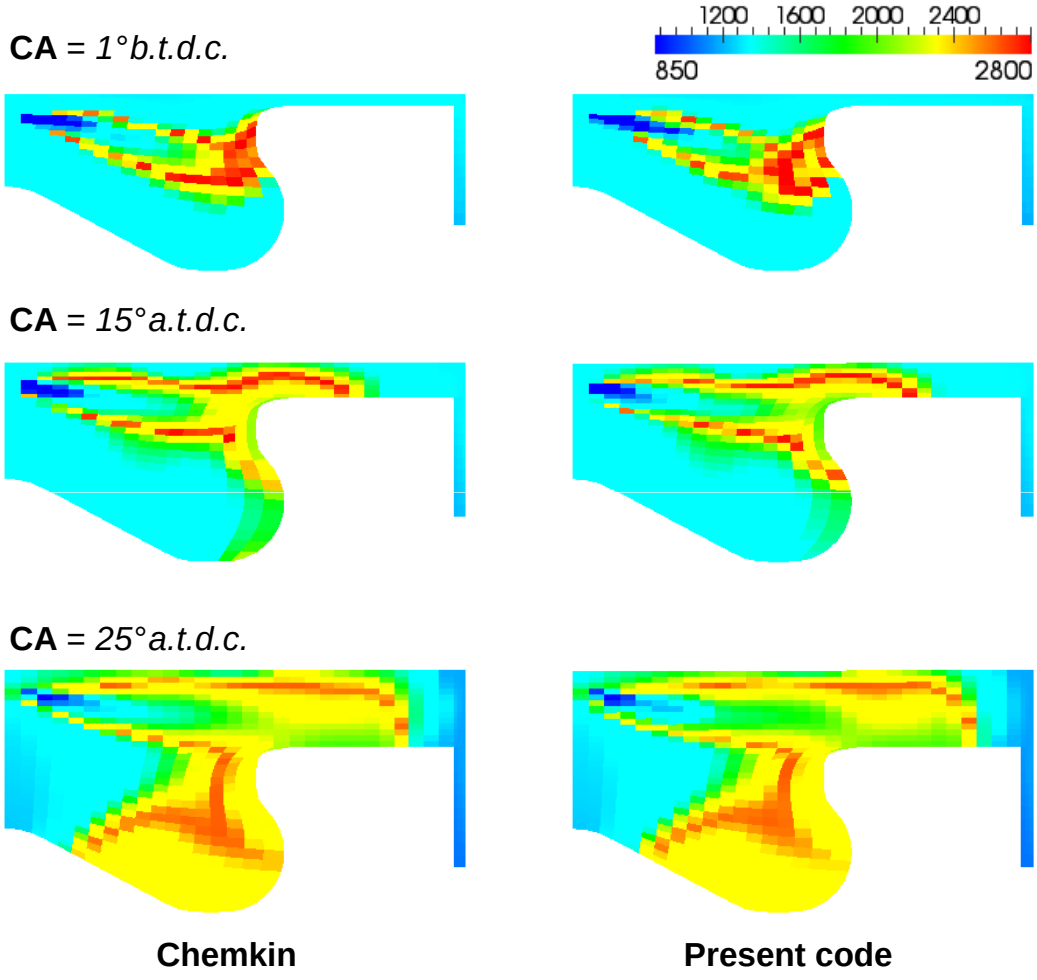

Figure 16: Comparison between predicted in-cylinder local temperature values [K] across the vertical injection axis section using KIVA-4.[61] Chemkin vs. present chemistry code coupling, solver = VODE with analytical sparse Jacobian, and tabulated thermodynamic properties.

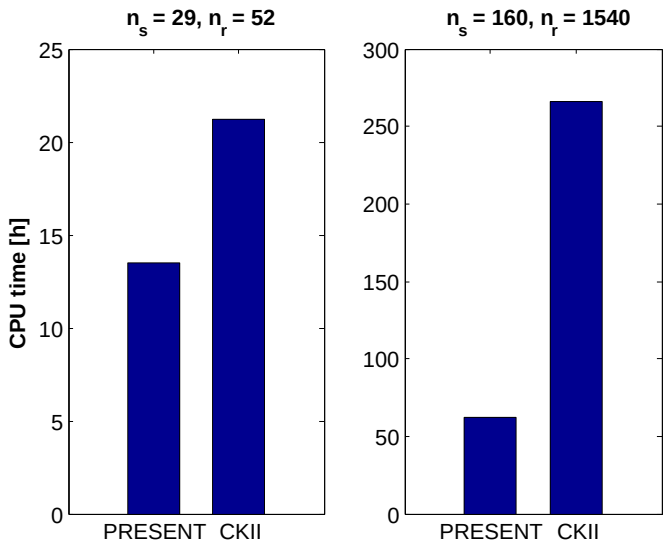

Figure 17: Comparison between overall CPU times needed by the KIVA-4 simulations running for the whole closed valve part of the engine cycle, using the two n-heptane combustion mechanisms tested.

dependent quantities at fixed sampling intervals has been introduced in order to further improve the computational efficiency of the approach, while still mantaining machine-precision accuracy.

The approach has been fully validated through application to three different combustion mechanisms of diesel and biodiesel fuel surrogates. The mechanisms range from 29 species, 52 reactions to 2878 species, 8555 reactions, thus spanning current available mechanism sizes. The code was also integrated in KIVA-4 with the aim of giving internal combustion engine CFD simulations detailed chemistry capability, and applied to an operating case of interest to internal combustion engine research. Comparison of the present code for both constant-volume adiabatic reactor cases and full CFD simulations was made with the CHEMKIN package for chemical kinetics.

Overall, the following conclusions are made:

- the analytical Jacobian formulation allows computation efforts of the chemistry ODE system to be significantly reduced in comparison with the common finite difference approach. Huge savings of about two orders of magnitude are seen for large mechanisms, and about 2x speedup is observed even for the smallest, almost dense reaction mechanism;

- the possiblility to exploit sparse matrix algebra for the Jacobian matrix computation enhanced the sparse LSODES solver capabilities, thus leading to significant computational savings in comparison with other more efficient ODE solvers. This points out the need to tailor the ODE solver to sparse algebra computations for maximum speed when dealing with detailed combustion chemistry;

- tabulation of relevant temperature-dependent properties with a fourth-degree polynomial function allowed high-fidelity representation of more complex functions involving computationally expensive exponentials, powers and logarithms, speeding their evaluation up by almost one order of magnitude. The interpolation accuracy was enough not to affect the stability of the integration algorithms tested;

- the most significant part of the speedup in comparison with the finite-difference Jacobian approach was due to the analytical Jacobian formulation. The impact of tabulation of temperature-dependent quantities was significant especially for the small mechanism size, where linear systems algebra was less demanding. The adoption of a sparser formulation for third-body reactions instead showed greater effects for the largest mechanism, where it accounted for more than 50% of the total CPU time reduction;

- coupling of present the chemistry code with CFD software for internal combustion engine computations enabled detailed simulations with combustion mechanisms of dimensions of the order of hundreds species to be completed reasonable time, of about one order of magnitude less than the time required by standard chemistry packages.

- analytical knowledge of the Jacobian matrix sparsity structure for arbitrary chemical kinetics problems also enables future work on to focus on the development of tailored ODE solution procedures.

Appendix

Troe formulation

Falloff reactions which follow Troe's kinetic law feature a particular behaviour of the pressure-dependent reaction rate enhancement term as described in detail by Wagner and Wardlaw[69] (see Equation 8). Lindemann's factor, describing the effective molecularity impact on reaction rate enhancement, is multiplied by a further term which is function of a new broadening, "centering" factor, $F_{cent,k}$:

$$log_{10}F_k^{Troe} = \log_{10} F_{cent,k} \left[1 + \left(\frac{\log_{10} Pr_k + c_k}{n_k - d_k \left(\log_{10} Pr_k + c_k \right)} \right)^2 \right]^{-1}, \tag{34}$$

where c_k, n_k and d_k are defined as:

$$\begin{aligned} c_k &= -0.40 - 0.67 \log_{10} F_{cent,k}, \\ n_k &= +0.75 - 1.27 \log_{10} F_{cent,k}, \\ d_k &= +0.14; \end{aligned} \tag{35}$$

Troe's centering factor is instead defined as a function of 4 more reaction-specific parameters, a_k, $T_{3,k}$, $T_{2,k}$ and $T_{1,k}$:

$$F_{cent,k} = (1 - a_k) \exp\left(-\frac{T}{T_{3,k}} \right) + a_k \exp\left(-\frac{T}{T_{1,k}} \right) + \exp\left(-\frac{T_{2,k}}{T} \right). \tag{36}$$

In the whole formulation, thus, only the logarithm of the reduced pressure term has nonzero partial derivative with respect to some species' mass fractions, while every term has temperature dependence. As far as the first dependence is concerned, it is better to express the derivative of the whole Troe correction parameter with respect to the logarithm of the reduced pressure value:

$$\frac{\partial \log_{10} F_k^{Troe}}{\partial \log_{10} Pr_k} = -2\, n_k \log_{10} F_{cent,k} \left(\log_{10} Pr_k + c_k\right) \cdot \cdot \left[n_k - d_k \left(\log_{10} Pr_k + c_k\right)\right] \cdot \cdot \left\{n_k^2 - 2\, n_k\, d_k \left(\log_{10} Pr_k + c_k\right) + + \left(d_k + 1\right) \left(\log_{10} Pr_k + c_k\right)^2\right\}^{-2}, \tag{37}$$

and finally to express the partial derivative with respect to species mass fractions as:

$$\frac{\partial F_k^{Troe}}{\partial Y_j} = 10^{\left(\log_{10} F_k^{Troe}\right)} \log\left(10\right) \frac{\partial \log_{10} F_k^{Troe}}{\partial Y_j}, \tag{38}$$

$$\frac{\partial \log_{10} F_k^{Troe}}{\partial Y_j} = \frac{\partial \log_{10} F_k^{Troe}}{\partial \log_{10} Pr_k} \frac{\log_{10} Pr_k}{\partial Y_j},$$

$$\frac{\partial \log_{10} Pr_k}{\partial Y_j} = \frac{\partial M_{eff,k}}{\partial Y_j} \left[\log\left(10\right) M_{eff,k}\right]^{-1}.$$

The overall partial derivative of the reaction enhancement factor with respect to species mass fraction Y_j hence yields the formulation of Eq. 17:

$$\frac{\partial F_k^{PD,Troe}}{\partial Y_j} = \frac{\partial}{\partial Y_j}\left[P_{cor,k} \cdot 10^{\left(\log_{10} F_k^{Troe}\right)}\right] = 10^{\left(\log_{10} F_k^{Troe}\right)} \left[\frac{\partial P_{cor,k}}{\partial Y_j} + P_{cor,k} \log\left(10\right) \frac{\partial \log_{10} F_k^{Troe}}{\partial Y_j}\right]. \tag{39}$$

For the temperature derivative of Troe's factor logarithm $\log_{10} F_k^{Troe}$, (see Equation 26), we have:

$$\frac{\partial \log_{10} F_k^{Troe}}{\partial T} = \frac{\partial \log_{10} F_k^{Troe}}{\partial \log_{10} Pr_k} \frac{\partial \log_{10} Pr_k}{\partial T}, \tag{40}$$

where the derivative with respect to the logarithm of the reduced pressure value is the same term

as already computed in Equation 38, and

$$\frac{\partial \log_{10} Pr_k}{\partial T} = \frac{1}{\log(10)} \left[\frac{1}{\kappa_{f,k,0}} \frac{\partial \kappa_{f,k,0}}{\partial T} - \frac{1}{\kappa_{f,k,\infty}} \frac{\partial \kappa_{f,k,\infty}}{\partial T} \right]. \tag{41}$$

Derivation of species thermodynamic properties

The needed thermodynamic properties of the species for solving the chemical kinetics IVP, and their derivative expressions, are reported here. The JANAF 7-coefficient polynomial standard[47] features two different coefficient sets $\{a, b, ..., g\}$ for fitting two different temperature ranges.

Species internal energy $\left[Jmol^{-1}\right]$ and derivative (constant-volume specific heat) $\left[Jmol^{-1}K^{-1}\right]$:

$$\begin{aligned} U_i &= R_{mol} \left[(a_i - 1) T + \frac{b_i}{2} T^2 + \frac{c_i}{3} T^3 + \frac{d_i}{4} T^4 + \frac{e_i}{5} T^5 + f_i \right]; \\ C_{v,i} &= \frac{\partial U_i}{\partial T} = R_{mol} \left[a_i - 1 + b_i T + c_i T^2 + d_i T^3 + e_i T^4 \right]; \end{aligned} \tag{42}$$

constant volume specific heat's derivative $\left[Jmol^{-1}K^{-2}\right]$:

$$\frac{\partial C_{v,i}}{\partial T} = R_{mol} \left[b_i + 2c_i T + 3d_i T^2 + 4e_i T^3 \right]. \tag{43}$$

Standard non-dimensional Gibbs free energy $[-]$ and derivative $\left[K^{-1}\right]$:

$$\begin{aligned} g_i^0 &= -\left[a_i \left(\log T - 1 \right) + \frac{b_i}{2} T + \frac{c_i}{6} T^2 + \frac{d_i}{12} T^3 + \frac{e_i}{20} T^4 - \frac{f_i}{T} + g_i \right]; \\ \frac{\partial g_i^0}{\partial T} &= -\left[\frac{a_i}{T} + \frac{b_i}{2} + \frac{c_i}{3} T + \frac{d_i}{4} T^2 + \frac{e_i}{5} T^3 + \frac{f_i}{T^2} \right]. \end{aligned} \tag{44}$$

Simulation details

Table 3 summarizes the eighteen test case initialisation for the n-heptane and methyl-decanoate combustion mechanisms. The results are tabulated in Tables 4, 5, 6, in terms of overall CPU time, and solvers' total number of integration steps and calls to the ODE system function, to the Jacobian function, plus matrix decompositions. Data have been gathered exploiting the solvers' optional output features, plus calls to the Fortran intrinsic `cpu_time` routine for the evaluation of elapsed times. The number of calls to the ODE system function do not include those required for building the Jacobian matrix, in case its internal generation by finite differences is selected.

Table 3: Details of the IVP initialisation for the 18 integration cases considered using each combustion mechanism. 'Fuel' species is intended to be n-heptane (C_7H_{16}) for the ERC and LLNL n-heptane mechanisms,[50,51] and methyl-decanoate (MD) for the LLNL biodiesel mechanism.[4] Mole fractions of any other species have been initialised at $1.0E-20$.

Fuel	Case	Temp.	Press.	Mixture	Mole fractions		
		$T_0[K]$	$p_0[bar]$	$\lambda[-]$	Fuel	O_2	N_2
C_7H_{16}	1	750	2.0	0.5	0.1538	0.1777	0.6685
	2	750	2.0	1.0	0.0833	0.1925	0.7241
	3	750	2.0	2.0	0.0435	0.2009	0.7557
	4	1000	2.0	0.5	0.1538	0.1777	0.6685
	5	1000	2.0	1.0	0.0833	0.1925	0.7241
	6	1000	2.0	2.0	0.0435	0.2009	0.7557
	7	1500	2.0	0.5	0.1538	0.1777	0.6685
	8	1500	2.0	1.0	0.0833	0.1925	0.7241
	9	1500	2.0	2.0	0.0435	0.2009	0.7557
	10	750	20.0	0.5	0.1538	0.1777	0.6685
	11	750	20.0	1.0	0.0833	0.1925	0.7241
	12	750	20.0	2.0	0.0435	0.2009	0.7557
	13	1000	20.0	0.5	0.1538	0.1777	0.6685
	14	1000	20.0	1.0	0.0833	0.1925	0.7241
	15	1000	20.0	2.0	0.0435	0.2009	0.7557
	16	1500	20.0	0.5	0.1538	0.1777	0.6685
	17	1500	20.0	1.0	0.0833	0.1925	0.7241
	18	1500	20.0	2.0	0.0435	0.2009	0.7557
MD	1	750	2.0	0.7	0.0844	0.1923	0.7233
	2	750	2.0	1.0	0.0606	0.1973	0.7421
	3	750	2.0	2.0	0.0313	0.2034	0.7653
	4	1000	2.0	0.7	0.0844	0.1923	0.7233
	5	1000	2.0	1.0	0.0606	0.1973	0.7421
	6	1000	2.0	2.0	0.0313	0.2034	0.7653
	7	1500	2.0	0.7	0.0844	0.1923	0.7233
	8	1500	2.0	1.0	0.0606	0.1973	0.7421
	9	1500	2.0	2.0	0.0313	0.2034	0.7653
	10	750	20.0	0.7	0.0844	0.1923	0.7233
	11	750	20.0	1.0	0.0606	0.1973	0.7421
	12	750	20.0	2.0	0.0313	0.2034	0.7653
	13	1000	20.0	0.7	0.0844	0.1923	0.7233
	14	1000	20.0	1.0	0.0606	0.1973	0.7421
	15	1000	20.0	2.0	0.0313	0.2034	0.7653
	16	1500	20.0	0.7	0.0844	0.1923	0.7233
	17	1500	20.0	1.0	0.0606	0.1973	0.7421
	18	1500	20.0	2.0	0.0313	0.2034	0.7653

Table 4: ERC n-heptane

Solver	Formulation			CPU	Number of:			
	analyt. Jac.	sparse third-b.	data tab.	time (s)	integr. steps	funct. evals.	Jac. evals.	matrix fact.
CHEMKIN	-	-	-	1.48	26009	39752	2051	13637
LSODE	-	-	-	4.26	25358	40193	11631	11631
	⋆	-	-	1.37	25157	39956	11568	11568
	⋆	⋆	-	1.33	24916	39674	11494	11494
	⋆	⋆	⋆	1.06	25126	39949	11590	11590
	⋆	-	⋆	1.10	25451	40268	11674	11674
LSODES	-	-	-	3.15	25608	42038	5284	11922
	⋆	-	-	1.28	25751	42261	5273	11968
	⋆	⋆	-	1.23	25796	42207	5258	11880
	⋆	⋆	⋆	0.95	26041	42642	5295	12044
	⋆	-	⋆	0.96	25938	42408	5274	11984
RADAU5	-	-	-	1.99	5290	36828	4044	5230
	⋆	-	-	0.97	4713	39122	3297	4713
	⋆	⋆	-	0.96	4723	39378	3301	4723
	⋆	⋆	⋆	0.75	4721	39367	3300	4721
	⋆	-	⋆	0.74	4714	39128	3298	4714
VODE	-	-	-	1.42	25989	39719	2050	13513
	⋆	-	-	0.95	25744	39487	2056	13571
	⋆	⋆	-	0.93	25926	39793	2053	13520
	⋆	⋆	⋆	0.72	25884	39663	2058	13617
	⋆	-	⋆	0.73	25969	39908	2059	13648
DASPK	-	-	-	5.99	32473	50402	16266	16266
	⋆	-	-	2.13	32501	50490	16285	16285
	⋆	⋆	-	2.03	32796	50918	16278	16278
	⋆	⋆	⋆	1.68	32813	51016	16249	16249
	⋆	-	⋆	1.77	32614	50648	16261	16261

Table 5: LLNL n-heptane

Solver	Formulation			CPU	Number of:			
	analyt. Jac.	sparse third-b.	data tab.	time (s)	integr. steps	funct. evals.	Jac. evals.	matrix fact.
CHEMKIN	-	-	-	119.2	30698	45925	2131	15633
LSODE	-	-	-	269.9	30874	48051	14001	14001
	⋆	-	-	47.6	30825	48094	14000	14000
	⋆	⋆	-	46.5	30848	48148	13987	13987
	⋆	⋆	⋆	44.4	30801	47987	13997	13997
	⋆	-	⋆	49.8	31022	48395	14081	14081
LSODES	-	-	-	168.8	31310	50392	6092	14196
	⋆	-	-	43.1	31665	50847	6140	14391
	⋆	⋆	-	21.4	31648	50876	6133	14394
	⋆	⋆	⋆	18.7	31602	50881	6137	14343
	⋆	-	⋆	40.5	31457	50688	6117	14267
RADAU5	-	-	-	109.5	5538	40694	4174	5490
	⋆	-	-	40.9	5273	44296	3669	5107
	⋆	⋆	-	39.8	5268	44379	3686	5115
	⋆	⋆	⋆	37.3	5268	44392	3687	5115
	⋆	-	⋆	38.1	5271	44339	3667	5105
VODE	-	-	-	72.4	30413	45629	2134	15561
	⋆	-	-	37.9	30415	45507	2121	15634
	⋆	⋆	-	36.9	30713	45770	2129	15694
	⋆	⋆	⋆	35.0	30580	45685	2128	15688
	⋆	-	⋆	35.8	30473	45723	2129	15672
DASPK	-	-	-	344.7	36245	57528	16840	16840
	⋆	-	-	76.5	36539	57922	16857	16857
	⋆	⋆	-	66.1	36578	58247	16903	16903
	⋆	⋆	⋆	63.9	36330	57673	16928	16928
	⋆	-	⋆	71.4	36575	58059	16951	16951

Table 6: LLNL methyl-decanoate

Solver	Formulation			CPU	Number of:			
	analyt.	sparse	data	time	integr.	funct.	Jac.	matrix
	Jac.	third-b.	tab.	(h)	steps	evals.	evals.	fact.
CHEMKIN	-	-	-	62.22	30056	44616	2301	16418
LSODE	-	-	-	174.29	28464	46251	14144	14144
	⋆	-	-	165.45	28158	45367	13792	13792
	⋆	⋆	-	165.45	28158	45367	13792	13792
	⋆	⋆	⋆	166.11	28490	45833	13895	13895
	⋆	-	⋆	165.90	28359	45582	13851	13851
LSODES	-	-	-	16.09	42085	67356	7861	21423
	⋆	-	-	9.33	32782	54381	6841	16280
	⋆	⋆	-	0.26	32961	54521	6806	16250
	⋆	⋆	⋆	0.26	32646	54307	6823	16167
	⋆	-	⋆	15.82	32757	54135	6767	16102
RADAU5	-	-	-	188.19	5505	40092	4005	5481
	⋆	-	-	175.77	4662	40596	3349	4592
	⋆	⋆	-	176.07	4670	40825	3354	4602
	⋆	⋆	⋆	176.02	4672	40816	3356	4604
	⋆	-	⋆	175.84	4665	40620	3351	4595
VODE	-	-	-	192.73	29772	44241	2299	16271
	⋆	-	-	192.87	28946	43221	2234	15828
	⋆	⋆	-	193.54	29399	43996	2231	15958
	⋆	⋆	⋆	193.14	29399	43947	2237	15882
	⋆	-	⋆	192.93	29342	43965	2226	15923
DASPK	-	-	-	217.65	35535	56524	16664	16664
	⋆	-	-	193.47	34161	54879	16179	16179
	⋆	⋆	-	193.72	34073	54868	16248	16248
	⋆	⋆	⋆	193.72	34232	54918	16222	16222
	⋆	-	⋆	192.47	34129	54760	16235	16235

References

(1) Lu, T.; Law, C. K. *Prog. Energy Combust. Sci.* **2009**, *35*, 192 – 215.

(2) Miller, J. A.; Kee, R. J.; Westbrook, C. K. *Annu. Rev. Phys. Chem.* **1990**, *41*, 345–387.

(3) Law, C. K. *Proc. Combust. Inst.* **2007**, *31*, 1 – 29.

(4) Herbinet, O.; Pitz, W. J.; Westbrook, C. K. *Combust. Flame* **2008**, *154*, 507 – 528.

(5) Herbinet, O.; Pitz, W. J.; Westbrook, C. K. *Combust. Flame* **2010**, *157*, 893 – 908.

(6) Westbrook, C.; Pitz, W.; Westmoreland, P.; Dryer, F.; Chaos, M.; Osswald, P.; Kohse-Höinghaus, K.; Cool, T.; Wang, J.; Yang, B.; Hansen, N.; Kasper, T. *Proc. Combust. Inst.* **2009**, *32*, 221 – 228.

(7) Westbrook, C. K.; Pitz, W. J.; Herbinet, O.; Curran, H. J.; Silke, E. J. *Combust. Flame* **2009**, *156*, 181 – 199.

(8) Seshadri, K.; Lu, T.; Herbinet, O.; Humer, S.; Niemann, U.; Pitz, W. J.; Seiser, R.; Law, C. K. *Proc. Combust. Inst.* **2009**, *32*, 1067 – 1074.

(9) Kong, S.-C.; Reitz, R. D. *Proc. Combust. Inst.* **2002**, *29*, 663 – 669, Proceedings of the Combustion Institute.

(10) Kong, S.-C.; Reitz, R. D. *J. Eng. Gas Turbines Power* **2002**, *124*, 702–707.

(11) Kokjohn, S. L.; Hanson, R. M.; Splitter, D. A.; Reitz, R. D. *Int. J. Engine Res.* **2011**, *12*, 209–226.

(12) Kokjohn, S.; Hanson, R.; Splitter, D.; Kaddatz, J.; Reitz, R. *SAE Int. J. Engines* **2011**, *4*, 360–374.

(13) Maas, U.; Pope, S. *Combust. Flame* **1992**, *88*, 239 – 264.

(14) Lam, S.; Coussis, D. *Symp. (Int.) Combust.* **1989**, *22*, 931 – 941.

(15) Lu, T.; Law, C. K. *Proc. Combust. Inst.* **2005**, *30*, 1333 – 1341.

(16) Lu, T.; Law, C. K. *Combust. Flame* **2006**, *144*, 24 – 36.

(17) Lu, T.; Law, C. K. *Combust. Flame* **2006**, *146*, 472 – 483.

(18) Lu, T.; Law, C. K.; Yoo, C. S.; Chen, J. H. *Combust. Flame* **2009**, *156*, 1542 – 1551.

(19) Sun, W.; Chen, Z.; Gou, X.; Ju, Y. *Combust. Flame* **2010**, *157*, 1298 – 1307.

(20) Curtiss, C. F.; Hirschfelder, J. O. *Proc. Natl. Acad. Sci. U. S. A.* **1952**, *38*, 235–243.

(21) Gear, C. W. *Numerical Initial Value Problems in Ordinary Differential Equations*; Prentice Hall PTR: Upper Saddle River, NJ, USA, 1971.

(22) Brown, P. N.; Byrne, G. D.; Hindmarsh, A. C. *SIAM J. Sci. Stat. Comp.* **1989**, *10*, 1038–1051.

(23) Hindmarsh, A. *IMACS Transactions on Scientific Computation* **1983**, *1*, 55–64.

(24) Hindmarsh, A. C. *ACM SIGNUM Newsletter* **1980**, *15*, 10–11.

(25) Byrne, G. D.; Dean, A. M. *Comput. Chem.* **1993**, *17*, 297 – 302.

(26) Aro, C. J. *Comput. Phys. Commun.* **1996**, *97*, 304 – 314.

(27) Kim, S.-L.; Choi, J.-Y.; Jeung, I.-S.; Park, Y.-H. *Appl. Numer. Math.* **2001**, *39*, 87 – 104.

(28) Aro, C. J.; Rodrigue, G. H. *Comput. Phys. Commun.* **1995**, *92*, 27 – 53.

(29) Aro, C. J. *Appl. Numer. Math.* **1996**, *21*, 335 – 352.

(30) Nejad, L. A. M. *Astrophys. Space Sci.* **2005**, *299*, 1–29, 10.1007/s10509-005-2100-z.

(31) Carver, M. B.; Boyd, A. W. *J. Phys. Chem.* **1979**, *11*, 1097–1108.

(32) Sandu, A.; Verwer, J.; Blom, J.; Spee, E.; Carmichael, G.; Potra, F. *Atmos. Environ.* **1997**, *31*, 3459 – 3472.

(33) Sandu, A.; Verwer, J.; Loon, M. V.; Carmichael, G.; Potra, F.; Dabdub, D.; Seinfeld, J. *Atmos. Environ.* **1997**, *31*, 3151 – 3166, EUMAC: European Modelling of Atmospheric Constituents.

(34) Bisetti, F. *Combust. Theory Modell.* **2012**, *16*, 387–418.

(35) Sandu, A.; Sander, R. *Atmos. Chem. Phys.* **2006**, *6*, 187–195.

(36) Damian, V.; Sandu, A.; Damian, M.; Potra, F.; Carmichael, G. R. *Comput. Chem. Eng.* **2002**, *26*, 1567 – 1579.

(37) Bank, R.; Douglas, C. *Adv. Comput. Math.* **1993**, *1*, 127–137, 10.1007/BF02070824.

(38) Remington, K.; Pozo, R. *NIST Sparse BLAS User's Guide*; 1996.

(39) Brown, P. N.; Hindmarsh, A. C. *Appl. Math. Comput.* **1989**, *31*, 40 – 91, Special Issue Numerical Ordinary Diferrential Equations (Proceedings of the 1986 ODE Conference).

(40) Barz, T.; Kuntsche, S.; Wozny, G.; Arellano-Garcia, H. *Comput. Chem. Eng.* **2011**, *35*, 2053 – 2065.

(41) Buzzi Ferraris, G.; Manca, D. *Comput. Chem. Eng.* **1998**, *22*, 1595 – 1621.

(42) Tolsma, J. E.; Barton, P. I. *Comput. Chem. Eng.* **1998**, *22*, 475 – 490.

(43) Tolsma, J.; Barton, P. I. *Ind. Eng. Chem. Res.* **2000**, *39*, 1826–1839.

(44) Schwer, D. A.; Tolsma, J. E.; Green, W. H.; Barton, P. I. *Combust. Flame* **2002**, *128*, 270 – 291.

(45) Mosbach, S.; Kraft, M. *Combust. Theory Modell.* **2006**, *10*, 171–182.

(46) Lee, S. L.; Gear, C. W. *J. Comput. Appl. Math.* **2007**, *201*, 258 – 274.

(47) M. W. Chase, J.; Curnutt, J. L.; J. R. Downey, J.; McDonald, R. A.; Syverud, A. N.; Valenzuela, E. A. *J. Phys. Chem. Ref. Data* **1982**, *11*, 695–940.

(48) Perini, F. Optimally reduced reaction mechanisms for Internal Combustion Engines running on biofuels. Ph.D. thesis, University of Modena and Reggio Emilia, http://www.himech-phdschool.unimore.it/site/home/download/tesi-di-dottorato/documento92016812.html, 2011.

(49) Warnatz, J.; Warnatz, J.; Maas, U.; Dibble, R.; Dibble, R. *Verbrennung: Physikalisch-Chemische Grundlagen, Modellierung und Simulation, Experimente, Schadstoffentstehung*; Springer, 2001.

(50) Patel, A.; Kong, S.; Reitz, R. Development and Validation of a Reduced Reaction Mechanism for HCCI Engine Simulations. 2004; http://papers.sae.org/2004-01-0558/.

(51) Seiser, R.; Pitsch, H.; Seshadri, K.; Pitz, W.; Curran, H. *Proc. Combust. Inst.* **2000**, *28*, 2029 – 2037.

(52) Yamamoto, A.; Kitamura, Y.; Yamane, Y. *Ann. Nucl. Energy* **2004**, *31*, 1027 – 1037.

(53) Schraudolph, N. N. *Neural Comput.* **1999**, *11*, 853–862.

(54) Li, Y. H.; Kong, S.-C. *Combust. Theory Modell.* **2008**, *12*, 205–219.

(55) Park, S. W.; Reitz, R. D. *Fuel* **2009**, *88*, 843 – 852.

(56) Guo, H.; Li, H.; Neill, W. S. *ASME ICEF2009 Conference Proceedings* **2009**, *2009*, 489–497.

(57) Hoffman, S. R.; Abraham, J. *Fuel* **2009**, *88*, 1099 – 1108.

(58) Shi, Y.; Ge, H.-W.; Brakora, J. L.; Reitz, R. D. *Energy Fuels* **2010**, *24*, 1646–1654.

(59) Perini, F.; Brakora, J. L.; Reitz, R. D.; Cantore, G. *Combust. Flame* **2012**, *159*, 103 – 119.

(60) Perini, F.; Cantore, G.; Reitz, R. An Analysis on Time Scale Separation for Engine Simulations with Detailed Chemistry. 2011; http://papers.sae.org/2011-24-0028.

(61) Torres, D. J.; Trujillo, M. F. *J. Comput. Phys.* **2006**, *219*, 943 – 975.

(62) Kee, R. J.; Rupley, F. M.; Miller, J. A. *CHEMKIN-II: A FORTRAN Chemical Kinetics Package for the Analysis of Gas-Phase Chemical Kinetics*; 1989.

(63) Hairer, E.; Wanner, G. *Solving Ordinary Differential Equations II: Stiff and differential-algebraic problems*; Springer series in computational mathematics; Springer-Verlag, 1993.

(64) Brown, P. N.; Hindmarsh, A. C.; Petzold, L. R. *SIAM J. Sci. Comput.* **1994**, *15*, 1467–1488.

(65) `http://www.reactiondesign.com/products/open/chemkin-pro.html`.

(66) Goodwin, D. *Chemical Vapor Deposition XVI and EUROCVD 14 (2003)* **2003**, *2003-08*, 155–162.

(67) Kokjohn, S. L.; Reitz, R. D. *J. Eng. Gas Turbines Power* **2011**, *133*, 102805.

(68) Golovitchev, V. I.; Montorsi, L.; Rinaldini, C. A.; Rosetti, A. *ASME ICEF2006 Conference Proceedings* **2006**, *2006*, 349–358.

(69) Wagner, A. F.; Wardlaw, D. M. *J. Phys. Chem.* **1988**, *92*, 2462–2471.