AIAA 2000–0438

Numerical Simulation of
Supersonic Reactive Flows
Using Explicit Runge-Kutta Methods

A. P. Graf and U. Riedel
Interdisciplinary Center for Scientific Computing
University of Heidelberg
69120 Heidelberg, Germany
Numerical Simulation of Supersonic Reactive Flows Using Explicit Runge-Kutta Methods

A. P. Graf and U. Riedel*
Interdisciplinary Center for Scientific Computing
University of Heidelberg
69120 Heidelberg, Germany

In this study numerical simulations are performed for a supersonic H$_2$-air mixing layer with several explicit Runge-Kutta methods. The flow is described by the compressible Navier–Stokes equations in two space dimensions. A detailed chemical reaction scheme consisting of 9 species and 37 reactions and a multi-component transport model is used. The time integration schemes implemented are: Classical Runge-Kutta, Heun’s method, Fehlberg’s method, and the method of Dormand & Prince. A detailed comparison is given for each of the four methods, each method combined with residual averaging, and each method combined with a point implicit evaluation of the chemical source term. A detailed analysis of the performance of the methods is given with respect to different stages of the simulation.

Introduction

In reactive flow simulations, the evaluation of the right hand side of the system of governing equations has a complicated structure if detailed physicochemical models for chemical kinetics, thermodynamic properties, and transport model are used. Therefore, forming the Jacobians needed with implicit integration methods is a complicated and computer resource consuming task. But it can be expected that such methods will be stable for stiff equation systems typically encountered in reactive flow simulations. But compared to implicit schemes, explicit methods have a simpler data structure, and need only modest computer resources. Furthermore, they can be implemented easily on parallel machines, which can overcompensate the deficiencies with respect to the stiff stable properties of explicit methods.

In this study the well known problem of a H$_2$-air mixing layer is used as a reference configuration to compare the performance of different explicit Runge-Kutta methods and some variants thereof. In all the numerical studies the spatial discretization, the reaction scheme, the transport model, the thermodynamic properties and grid point system remain unchanged. For all methods examined (Classical Runge-Kutta, Heun’s method, Fehlberg’s method, and the method of Dormand & Prince) the cpu–time versus physical integration time is reported for the initial build-up of the radical pool, the subsequent ignition phase, and the relaxation towards the steady state.

The Governing Equations for Viscous Compressible Flow

Equations and Transport Model

The general form of the equations governing reacting viscous nonequilibrium flows with $S$ species is

$$\frac{\partial \vec{U}}{\partial t} + \sum_{i=1}^{2} \frac{\partial F_i(\vec{U})}{\partial x_i} = \vec{S}(\vec{U}),$$

where \( \vec{U} = (\rho, \rho v_1, \rho v_2, e, \rho w_1, \ldots, \rho w_S)^T \) is the vector of conserved quantities and \( \vec{S}(\vec{U}) = (0, 0, 0, M_1 \dot{\omega}_1, \ldots, M_S \dot{\omega}_S)^T \) denotes the source term due to chemical reactions. \( \dot{\omega}_i \) is the molar scale rate of formation of species \( i \) (\( i = 1, \ldots, S \)) and \( M_i \) the molar mass. The flux vectors are:

\[
\vec{F}_i = \vec{F}_{i,\text{inv}} + \vec{F}_{i,\text{vis}} = \begin{pmatrix}
\rho v_i \\
\rho v_1 v_i + \delta_1 p \\
\rho v_2 v_i + \delta_2 p \\
(\rho e + p)v_i \\
\rho w_1 v_i \\
\vdots \\
\rho w_S v_i
\end{pmatrix} + \begin{pmatrix}
0 \\
\Pi_{1i} \\
\Pi_{2i} \\
\sum_{j=1}^{2} v_j \Pi_{ij} + j_{q,i} \vec{j}_{1,x_i} \\
\vdots \\
j_{S,x_i}
\end{pmatrix},
\]

where \( \rho \) is the density and \( v_i \) are the velocity components in the \( x_i \) direction, \( p \) denotes the pressure, \( w_i \) the mass fraction of species \( i \) (\( i = 1, \ldots, S \)) and \( e \) is the specific total energy, which is defined as

\[
e = \sum_{i=1}^{S} w_i e_i + \frac{1}{2} \sum_{j=1}^{2} v_j^2.
\]
with specific internal energy $e_i$ of each species. The shear stress tensor $\mathbf{\Pi}$ is defined as:

$$\mathbf{\Pi} = -\eta \left[ (\text{grad } \vec{v}) \right. + (\text{grad } \vec{v})^T - \frac{2}{3} (\text{div } \vec{v}) \vec{E} \Big] .$$

with the components

$$\mathbf{\Pi}_{ij} = -\eta \left[ \frac{\partial v_i}{\partial x_j} + \frac{\partial v_j}{\partial x_i} - \frac{2}{3} \delta_{ij} \text{div } \vec{v} \right] .$$

The molecular mass flux $\vec{J}_i$ in a multicomponent system\(^6\) (neglecting pressure diffusion) is

$$\vec{J}_i = -\frac{1}{M} D^M_i \text{grad } \left( \frac{M}{w_i} \right) - \frac{D^T_i}{T} \text{grad } T$$

with the mole fractions $x_i = w_i \frac{M}{M_i}$, the mean molar mass $M = \left( \sum_{i=1}^{S} w_i \frac{M}{M_i} \right)^{-1}$ and an effective diffusion coefficient

$$D^M_i = (1 - w_i) \left( \sum_{j=1}^{S} \frac{x_j}{D_{ij}} \right)^{-1}$$

of species $i$ in the mixture computed from the binary diffusion coefficients $D_{ij}$ and the thermal diffusion coefficient

$$D^T_i = k_i c M_i \frac{M_i}{\sum_{j \neq i} M_j} .$$

In the energy equation $\vec{\dot{q}}_i$ represents the energy fluxes due to heat conduction and species diffusion:

$$\vec{\dot{q}}_i = \vec{\dot{q}}^c_i + \vec{\dot{q}}^p_i = -\lambda \text{grad } T + \sum_{i=1}^{S} h_i \vec{\dot{q}}_i .$$

Here, $\vec{\dot{q}}^c_i$ denotes the conductive energy flux, $\vec{\dot{q}}^p_i$ the flux caused by diffusion, $T$ the temperature, $\lambda$ the thermal conductivity of the mixture and $h_i$ the specific enthalpy of species $i$. Details about the transport model implemented to calculate the viscosity $\mu$, the thermal conductivity $\lambda$ and binary diffusion coefficients $D_{ij}$ can be found in\(^7,8\) The temperature and pressure are calculated from the equations

$$e = \sum_{i=1}^{S} w_i h_i - \frac{p}{\varrho} + \frac{1}{2} \sum_{j=1}^{3} \varrho_{ij}^2$$

$$h_i = h^0_i + \int_{T_{ref}}^{T} c_{pi} dT$$

$$p = \varrho RT \sum_{i=1}^{S} \frac{w_i}{M_i} .$$

where $h^0_i$ is the heat of formation and $c_{pi}$ the constant pressure specific heat of species $i$ ($R$ denotes the universal gas constant). Fourth order polynomials are used to determine $c_{pi}$ as a function of temperature\(^9\)

$$c_{pi} = a_0 + a_1 T + a_2 T^2 + a_3 T^3 + a_4 T^4$$

and the mixture specific heat

$$C_p = \sum_{i=1}^{S} w_i c_{pi}$$

is obtained by mass fraction weighting.

The Chemical Source Terms

A set of $R$ reactions involving $S$ species can be written in the general form\(^10\)

$$\sum_{s=1}^{S} \nu_{rs}^{(e)} A_s = k_r \sum_{s=1}^{S} \nu_{rs}^{(p)} A_s \text{ with } r = 1, \ldots, R ,$$

where $\nu_{rs}^{(p)}$ and $\nu_{rs}^{(e)}$ are the stoichiometric coefficients of species $s$ appearing in reaction $r$. The rate coefficient $k_r$ is given by a modified Arrhenius expression

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

where $E_r$ represents the activation energy, $A_r$ the pre-exponential factor and $b_r$ the temperature coefficient. The net rate of change of a species is the sum over the contributions of all elementary reaction steps

$$\dot{w}_i = \sum_{r=1}^{R} k_r (\nu_{ri}^{(p)} - \nu_{ri}^{(e)}) \prod_{s=1}^{S} c_{s}^{(e)} \quad i = 1, \ldots, S , \quad (2)$$

with concentrations $c_s$ of the species. From the above equation we observe that the chemical source terms are exponential functions of temperature and a nonlinear function of the species concentrations. Because of this highly nonlinear nature of the kinetics and the strongly differing time scales involved in the elementary reactions the set of species equations in Eq. (1) is generally stiff.

Spatial Discretization

Equation (1) is written in Cartesian coordinates, however we are interested in applying the numerical method to a curvilinear grid and thus we have to transform to a general coordinate system, $\xi = \xi(x, y)$ and $\eta = \eta(x, y)$. The Cartesian coordinates may be replaced to yield the chain rule conservation law form of the governing equations

$$\frac{\partial \vec{U}}{\partial t} + \xi_x \frac{\partial \vec{F}_1(\vec{U})}{\partial \xi} + \eta_x \frac{\partial \vec{F}_1(\vec{U})}{\partial \eta} +$$

$$\xi_y \frac{\partial \vec{F}_2(\vec{U})}{\partial \xi} + \eta_y \frac{\partial \vec{F}_2(\vec{U})}{\partial \eta} = \vec{S}(\vec{U}) . \quad (3)$$
and the numerical computations with we obtain the form of the equations actually used in the time step with two steps of half the original stepsize. The amount of evaluations of the right-hand side is 11, which is by far the biggest number of all methods implemented in this work.

Method of Fehlberg 4(5)

The method of Fehlberg is an explicit Runge-Kutta method of order 4. The error estimation is done by embedding the method in a Runge-Kutta method of order 5. This needs only one extra evaluation of the right-hand side of the differential equation system. The total number of function evaluations is six.

Method of Dormand & Prince 5(4)

Similar to the method above, the method of Dormand & Prince (1980) is an embedded explicit Runge-Kutta method. The difference is that the coefficients for the higher order method are optimized in account of the local error. Therefore the result of the method of order 5 can be used in the time step. Seven evaluations of the right-hand side of the differential equation system are needed.

Method of Heun

The method of Heun is not a Runge-Kutta method in contrast to the methods described above, but is embedded in a Runge-Kutta method of order three for stepsize control reasons. It can be derived from the implicit trapezoidal method:

\[ y_{k+1} = y_k + \frac{1}{2} \left( f(x_k, y_k) + f(x_{k+1}, y_{k+1}) \right). \]

As a result of using a predictor value \( y_{k+1}^{(p)} \) for \( y_{k+1} \), one gets an explicit method of order 2:

\[ k_1 = f(x_k, y_k), \]
\[ k_2 = f(x_k + h, y_k + h k_1), \]
\[ y_{k+1}^{(H)} = y_k + \frac{h}{2} (k_1 + k_2). \]

Such a method is called predictor-corrector method. It can be embedded into a Runge-Kutta method of order 3 by calculation of only one extra right-hand side of the differential equation system:

\[ k_3 = f \left( x_k + \frac{1}{2} h, y_k + \frac{h}{4} (k_1 + k_2) \right), \]
\[ y_{k+1}^{(RK3)} = y_k + \frac{h}{6} (k_1 + k_2 + 4k_3). \]

The local error of the time step is estimated as the norm of the difference \( y_{k+1}^{(RK3)} - y_{k+1}^{(H)} \). This is an efficient method of error estimation, because only three evaluations of the right-hand side of the differential equation system are necessary compared to two evaluations without error estimate.

Adaptive Stepsize Control

The purpose of an adaptive stepsize control is to achieve some predetermined accuracy in the solution.

Time Integration

Integration Methods

Classical Runge-Kutta Method

The classical Runge-Kutta method of order 4 is often used for various applications, sometimes with slightly different coefficients. The disadvantage of the method is the expensive error estimation. It is not possible to embed this method in a higher order method. Therefore, the only practical way to estimate the local error is to repeat the time step with two steps of
with minimum computational effort. The stepsize $h$ of the next time step needs to be evaluated based on the local integration error of the current time step. With the assumption that the local discretization error of the $k$th time step is $d_k$, the stepsize is $h_k$ and the desired (relative) accuracy is $d_{\text{max}}$, the stepsize of the next step can be estimated as

$$h_{k+1} = S h_k \left| \frac{d_k}{d_{\text{max}}} \right|^{\frac{1}{S}} ,$$

where $n$ is the order of the integration method used. Usually, a safety factor $S$ is put in (typically 0.95). If the achieved local error for a step is too big, then the above recipe is useful to determine the stepsize for the repetition of this step.

**Point-Implicit Treatment of the Source Terms**

The stiffness of the differential equation system for reactive flows is mainly caused by the (spatially local) chemical source terms. In order to incorporate the advantages of implicit integration with respect to the chemical source term into an explicit method, the chemical source term is evaluated at time $t + \Delta t$. The source term $\hat{S}(\vec{U}(t + \Delta t))$ is obtained by a first order Taylor expansion:

$$\hat{S}(\vec{U}(t + \Delta t)) \approx \hat{S}(\vec{U}(t)) + \left( \frac{\partial \hat{S}}{\partial \vec{U}} \right)(t) \hat{S}(\vec{U}(t + \Delta t)) \Delta t ,$$

which can be rearranged as a linear equation system for $\hat{S}(\vec{U}(t + \Delta t))$:

$$\left( \mathbf{I} - \Delta t \left( \frac{\partial \hat{S}}{\partial \vec{U}} \right)(t) \right) \hat{S}(\vec{U}(t + \Delta t)) = \hat{S}(\vec{U}(t)) .$$

The dimension of this equation system is $S \times S$ ($S$ denotes the total number of species) because the first 4 components of the source term vector related to continuity, momentum and energy are zero. In our case, for nine species, the equation system is not sparse. Therefore, we employ a simple Gauss algorithm to solve it.

The evaluation of the Jacobian $\mathbf{A} := \frac{\partial \hat{\omega}}{\partial \vec{U}}$ needs to be done for each grid point:

$$\mathbf{A}_{ij} = M_i \frac{\partial \hat{\omega}_i}{\partial \vec{U}_j} \quad \text{for all } i, j \in \{1, \ldots, S\} .$$

The rate of formation $\hat{\omega}_i$ is given by Eq. (2) and Eq. (11) yields

$$\mathbf{A}_{ij} = M_i \sum_{r=1}^{R} (\nu_{ri}^{(p)} - \nu_{ri}^{(e)}) k_{r} \times \left( \frac{b_r}{T} + \frac{E_r}{RT} \right) \frac{\partial T}{\partial \vec{U}_j} + \nu_{ri}^{(e)} \frac{\partial \vec{U}_j}{\vec{U}_j} \right) \prod_{s=1}^{S} \nu_{ri}^{(e)} ,$$

with

$$\frac{\partial T}{\partial \vec{U}_j} = \frac{R T}{\sum_{i=1}^{S} c_i (e_p, M_i - R)} .$$

Because of the similarity of the elements $\mathbf{A}_{ij}$ with the rate of formation, it is possible to calculate the Jacobian in an efficient way, simultaneous to the rate of formation $\hat{\omega}_i$.

**Residual Smoothing**

The aim of the residual smoothing is to speed up convergence towards the steady state. A method to achieve this is to replace the residuals by weighted averages of the neighbouring residuals, resulting in an increased stepsize. In contrast to the point-implicit modification, calculations with residual averaging are no longer time accurate. However, the steady state remains unchanged. Larger stepizes are achieved, because errors and rapid changing values are smoothed in space.

For our two-dimensional case the averaging is computed according to:

$$\hat{f}_{l,k} = f_{l,k} + \epsilon \left( f_{+,+} + f_{+,+} + f_{+,+} + f_{+,+} - 8 f_{l,k} + f_{l,-} + f_{-,+} + f_{-,+} + f_{-,+} \right) ,$$

where $\hat{f}_{l,k}$ is the averaged residual and $\epsilon$ - and - denote next neighbours in the grid point system on a two-dimensional structured grid with indices $l$ and $k$. $\epsilon$ is a weight factor which is in the range $0 \leq \epsilon \leq \frac{1}{16}$ obtained from the condition that the central point is the dominant contribution to the averaged residual.

**Results**

**Test Case: Simulation of a Laminar Supersonic H2-Air Mixing Layer**

To give a comparison of the different time integration methods and their usefulness in supersonic reactive flow simulations, each method is applied to the
well known problem of a laminar supersonic H₂-air mixing layer. The reaction scheme used consists of 37 reactions of 9 species and is taken from. The setup of the simulation is displayed in Fig. (1). The simulations are performed on a 30 × 30 grid (see Fig. (2)). The mesh is refined along the mixing layer and the inflow boundary. As an initial profile, the inflow composition is set as initial value for all the grid points in streamwise direction.

Results of the Simulation

The steady state profiles of OH and H₂O are displayed in the Fig. (3) and Fig. (4). The flame is on the air side of the system because H₂ has a smaller molecular mass and is therefore diffusing faster than any other species. The profiles of temperature and the OH radical are closely related to each other, because the OH is only created in very hot regions. Also, H₂O and H₂ are related, because the consumption of H₂ goes along with the formation of H₂O.

Fig. 3 Mass fraction of OH.

Fig. 4 Mass fraction of H₂O.

Fig. 5 Number of function evaluations per 10 μs (Line: Method of Heun, dashed line: Dormand & Prince, broken line: Fehlberg, dotted line: Classical Runge-Kutta method). Upper: Explicit integration, middle: Residual averaging, lower: Point-implicit integration.

No further discussion of the profiles is included here, because the main objective of the paper is a comparison of different numerical methods in connection with the nonlinearity of detailed physico-chemical models.

The Series of Computations

A series of computations with all four integration methods described above is carried out: Explicit integration, explicit integration with residual averaging and point-implicit treatment of the source term. Each case is computed with three different relative accuracies (10⁻², 10⁻³ and 10⁻⁴). The weight ε for the residual averaging is chosen with ε = 0.06. The steady state is reached after about \( t = 250 \) μs. Only 6 MB of memory are needed to run the simulations. After each 10 μs of physical time the number of time steps, the number of time step repetition (due to the error tolerance prescribed), and the number of function evaluations of the right-hand-side of the equation system are recorded.

5 of 7

AMERICAN INSTITUTE OF AERONAUTICS AND ASTRONAUTICS Paper 2000–0438
Fig. 6 CPU time per 10 μs (Line: Method of Heun, dashed line: Dormand & Prince, broken line: Fehlberg, dotted line: Classical Runge-Kutta method). Upper: Explicit integration, Middle: Residual averaging, lower: point-implicit integration.

Table 1 CPU time per time step on a SGI Indigo (R4400, 200 MHz). Case (a): Explicit, case (b): Residual Averaging, case (c): Point-Implicit.

<table>
<thead>
<tr>
<th>Case</th>
<th>Heun</th>
<th>Fehlb. 4(5)</th>
<th>Dorm. Pr. 5(4)</th>
<th>Class. R. K.</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>1.03</td>
<td>1.84</td>
<td>2.15</td>
<td>3.22</td>
</tr>
<tr>
<td>(b)</td>
<td>1.04</td>
<td>1.90</td>
<td>2.18</td>
<td>3.27</td>
</tr>
<tr>
<td>(c)</td>
<td>2.87</td>
<td>5.56</td>
<td>6.38</td>
<td>10.03</td>
</tr>
</tbody>
</table>

Fig. 7 Total CPU time to reach steady state.

Comparison of the Integration Methods

The results to be discussed here are not sensitive to the relative tolerance. Therefore, only results for a relative tolerance of 10^{-3} are presented here.

The calculation time per time step for each integration method is displayed in Table 1. By increasing the order of a method, the calculation time increases in the same way, because more evaluations of the right-hand side of the differential equation system are necessary. The point-implicit methods are the most expensive methods because of the calculation of the Jacobian.

The comparison of the simulations is displayed in Fig. (5) with regard to the number of evaluations of the right-hand side and in Fig. (6) with regard to the cpu-time. Three stages of the simulation can be identified from these figures:

- 0 µs - 90 µs buildup of radical pool
- 90 µs - 140 µs ignition
- 140 µs - 250 µs relaxation towards the steady state.

The higher the order of an integration method, the smaller the number of time steps. The method of Heun needs twice the number of evaluations of the differential equation system than the classical Runge-Kutta method of order 4.

The calculation time per 10µs doubles during ignition for the explicit method and the residual averaging in contrast to the point-implicit method, which handles the ignition in a better way by construction. Nevertheless, the calculation time is much higher than for the other methods. The Runge-Kutta method of order 4 seems not to be suited for a point-implicit treatment of the source term for this specific application. Here, further investigations are necessary to clarify this findings. The residual averaging performs best for all integration methods with respect to calculation time. As a conclusion, even so the higher order methods are better in terms of number of evaluations of the differential equation system, the method
of Heun (only 2 order) is best in terms of calculation time. The total calculation time for all simulations is displayed in Fig. (7). If one is interested in the steady state only, the method of Heun with residual averaging performs best otherwise the method of Heun without modifications.

Summary

To summarize, there is only little difference found between the method of Fehlberg and Dormand & Prince for all three stages of the computation. With respect to the cpu-time consumed to reach steady state it is shown that the classical Runge Kutta scheme is less efficient than the to aforementioned methods due to the large number of evaluations of the right hand side of the equation system needed to obtain the error estimate for each time step.

The point implicit evaluation of the chemical source term for the whole computation needs more cpu-time compared to the standard explicit evaluation of the source term. However, a combination of both methods, i.e. using the point implicit formulation during the ignition stage, will help to reduce the total amount of cpu-time.

A similar study will be performed with an already existing parallel explicit code for direct numerical simulations of turbulent reacting flows. Such flows are usually low speed flows, in contrast to the application presented here. Here, the performance of the different integration methods needs to be reexamined.

Acknowledgement

One of the authors (U. R.) thanks Prof. J. Warnatz for continuous support and discussions and many helpful discussions on reaction kinetics.

References