# Eilmer's Theory Book: Basic Models for Gas Dynamics and Thermochemistry.

Mechanical Engineering Report 2010/09

P. A. Jacobs,[*] R. J. Gollan,[†] A. J. Denman, B. T. O'Flaherty,
D. F. Potter, P. J. Petrie-Repar and I. A. Johnston.

January 27, 2012

**Abstract**

`Eilmer3` is an integrated collection of programs for the simulation of transient compressible flow in two and three spatial dimensions. It is based on a finite-volume formulation of the mass, momentum, energy and species conservation equations and is implemented on block-structured grids. Starting from an initial flow state with boundary conditions, state quantities in each finite-volume cell are updated in time to provide a simulation of the evolving flow field.

[*]Queensland Geothermal Energy Centre of Excellence, The University of Queensland, Brisbane, Australia.
[†]Centre for Hypersonics, The University of Queensland, Brisbane, Australia.

# Contents

# Nomenclature, Units

| | |
|---|---|
| $A$ | : cell area, m$^2$ |
| $a$ | : speed of sound, m/s |
| $C_v$ | : specific heat at constant volume, $\mathrm{J/kg \cdot K}$ |
| $C_p$ | : specific heat at constant pressure, $\mathrm{J/kg \cdot K}$ |
| $E$ | : total energy per unit mass $e + \frac{1}{2}u^2$, J/kg |
| $e$ | : specific internal energy, J/kg |
| $\overline{F}$ | : array of flux terms |
| $f$ | : species mass fraction |
| $H$ | : total enthalpy, J/kg |
| $h$ | : specific enthalpy, J/kg |
| $\hat{i}, \hat{j}, \hat{k}$ | : unit vectors for the cartesian coordinates |
| $i, j, k$ | : cell index |
| $k$ | : coefficient of thermal conductivity |
| $M$ | : Mach number |
| $MW$ | : molecular weight |
| $n$ | : direction cosine |
| $\hat{n}, \hat{p}, \hat{q}$ | : unit vectors for the cell interface |
| $P, p$ | : pressure, Pa |
| $Pr$ | : Prandtl number |
| $Q$ | : source vector in the gas-dynamic equations |
| $q$ | : heat flux, W/m$^2$ |
| $R$ | : gas constant, $\mathrm{J/kg \cdot K}$ |
| $R_0$ | : universal gas constant, $8.314\ \mathrm{J/mole \cdot K}$ |
| $Re$ | : Reynolds number |
| $r$ | : radial coordinate, m |
| $S$ | : control surface of the cell |
| $T$ | : temperature, K |
| $t$ | : time, s |
| $U$ | : state vector in the gas-dynamic equations |
| $u$ | : velocity, m/s |
| $V$ | : cell volume, m$^3$ |
| $x, y, z$ | : cartesian coordinates, m |

| | |
|---|---|
| $\alpha$ | : weighting parameter |
| $\Delta\pm$ | : intermediate variable for interpolation |
| $\lambda$ | : second coefficients of viscosity, Pa.s |
| $\mu$ | : viscosity, Pa.s; friction coefficient |
| $\rho$ | : density, kg/m$^3$ |
| $\gamma$ | : ratio of specific heats |
| $\tau$ | : wall shear stress, Pa |

## Subscripts

$i$          : cell index, inviscid

$j, k$       : cell indices

$L, \ R$     : left and right states for the Riemann solver or flux calculator

$max$      : maximum value

$n$          : normal to the cell interface

$p, q$       : tangent to the cell interface

$s$          / species index

$v$          : viscous

$x, y, z$    : coordinate directions

## Superscripts

$\overline{(\cdots)}$    : cell average

# 1 Introduction

`Eilmer3` is a derivative of the code `mbcns2`, an acronym for Multiple-Block Compressible Navier-Stokes solver, second version. The code solves the compressible Navier–Stokes equations in order to provide simulations of transient compressible flow in two- and three-dimensions. The governing equations are expressed in integral form over cell-centred, finite-volume cells, with the time rate of change of conserved quantities in each cell specified as a summation of the mass, momentum and energy flux through the cell interfaces.

The `mbcns2` code was an experiment in writing the `mb_cns` code in C++. `mb_cns` (written in C) was derived from `cns4u`, a code written at ICASE in 1990-91 to simulate high-performance shock tunnels and expansion tubes. The finite-difference codes of the 1980s did not do a good job of capturing the strong shocks that processed the gas in these machines so we started work on a new code, using (what was at the time) the recently-proven upwinding approach.

From the beginning, the code was intended to run on a parallel computer with blocks of finite-volume cells being allocated to the compute nodes of the Intel Hypercube computer that had been acquired by ICASE in 1990. However, in 1991, the code remained as a single-block code and was run on the Cray vector supercomputers at Langley to compute ideal gas flows in expansion tubes. The code was used and further developed at UQ through the 1990s and into the 2000s. Enhancements included:

- more general thermochemistry, including look-up tables for gas in chemical equilibrium;

- multiple-block capability with an MPI parallel implementation;

- a parametric and programmable front-end for specifying geometry and grids;

- three-dimensional geometry;

- programmable boundary conditions; and

- finite-rate chemistry.

Once it was determined that there were clear benefits in using C++ `mb_cns2`, our three-dimensional flow code Elmer was then reworked in C++ as `Elmer2`. Of course, these codes being experiments in C++, we soon decided that it could all be done much more cleanly be made much more versatile if we just reworked some of the basic modules. Thus, the thermochemistry was reworked and the separate 2- and 3-dimensional codes merged into Eilmer3. The name change from `Elmer` to `Eilmer` is to a void a naming clash with the Elmer finite-element code from Finland.[1]

---

[1] http://www.csc.fi/elmer

`Eilmer3` is actually an integrated collection of programs that includes a preparation program that can be used to set up a database of simulation parameters, a block-structured grid defining the flow domain and an initial flow field. These items are then used as a starting point for the main simulation program which computes a series of snapshots of the evolving flow.

The following sections describe the formulation of the code in terms of the basic gas dynamic model and the thermochemical model for a multiple-species gas with finite-rate chemical kinetics. There is a companion report [1] that desctibes the use of the code and provides a number of case studies.

## 2 Gas Dynamics

The code is formulated around the integral form of the Navier-Stokes equations, which can be expressed as

$$\frac{\partial}{\partial t} \int_V U dV = - \oint_S \left( \overline{F}_i - \overline{F}_v \right) \cdot \hat{n} \, dA + \int_V Q dV \, , \tag{1}$$

where $S$ is the bounding surface and $\hat{n}$ is the outward-facing unit normal of the control surface. Two-dimensional and three-dimensional formulations are implemented somewhat separately in `Eilmer3`, however, there is much of the formulation and code that is the same for both cases.

### 2.1 Governing Equations for Axisymmetric, Two-Dimensional Flow

For axisymmetric flow, the symbol $V$ in Eq.(1) is the volume and $A$ the area of the cell boundary per unit radian in the circumferential direction. The array of conserved quantities is dependent on the thermal model under consideration, and for the thermal nonequilibrium models is

$$U = \begin{bmatrix} \rho \\ \rho u_x \\ \rho u_y \\ \rho E \\ \rho e_{v_m} \\ \rho e_e \\ \rho f_s \end{bmatrix} . \tag{2}$$

Here, the conserved quantities are respectively density, $x$-momentum per volume, $y$-momentum per volume, total energy per volume, vibrational energy for mode $m$, electronic-electron energy and mass density of species $s$. Note that $\rho e_e$ includes both bound and free electron energy. We choose to solve both total and all individual species continuity equations to add rigour to our solver: the redundant information gives us a good idea when the numerics are running into trouble. Conversely, when only solving $n - 1$ species equations, it is easier for undetected error in mass fractions to accumulate. Thus for 11 species air with 6 vibrating molecules and the inclusion of electrons, for example, there are 22 conserved quantities.

The flux vectors are divided into inviscid and viscous contributions. The inviscid component in thermal nonequilibrium is

$$\overline{F}_i = \begin{bmatrix} \rho u_x \\ \rho u_x^2 + p \\ \rho u_y u_x \\ \rho E u_x + p u_x \\ \rho e_{v_m} u_x \\ \rho e_e u_x + p_e u_x \\ \rho f_s u_x \end{bmatrix} \hat{i} + \begin{bmatrix} \rho u_y \\ \rho u_x u_y \\ \rho u_y^2 + p \\ \rho E u_y + p u_y \\ \rho e_{v_m} u_y \\ \rho e_e u_y + p_e u_y \\ \rho f_s u_y \end{bmatrix} \hat{j} , \tag{3}$$

and the viscous component is

$$\overline{F}_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ \tau_{xx} u_x + \tau_{yx} u_y + q_x \\ q_{x,v_m} \\ q_{x,e} \\ J_{x,s} \end{bmatrix} \hat{i} + \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{xy} u_x + \tau_{yy} u_y + q_y \\ q_{y,v_m} \\ q_{y,e} \\ J_{y,s} \end{bmatrix} \hat{j} . \tag{4}$$

where the axisymmetric viscous stresses are

$$\begin{aligned} \tau_{xx} &= 2\mu \frac{\partial u_x}{\partial x} + \lambda \left( \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{u_y}{y} \right) , \\ \tau_{yy} &= 2\mu \frac{\partial u_y}{\partial y} + \lambda \left( \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{u_y}{y} \right) , \\ \tau_{xy} = \tau_{yx} &= \mu \left( \frac{\partial u_x}{dy} + \frac{\partial u_y}{dx} \right) , \end{aligned} \tag{5}$$

and where the secondary viscosity coefficient $\lambda$ is expressed in terms of the primary coefficient $\mu$ via Stokes hypothesis, $\lambda = -\frac{2}{3}\mu$. The viscous heat fluxes are

$$\begin{aligned} q_x &= k_{tr}\frac{\partial T}{\partial x} + \sum_{s=\text{mol.}} k_{v_s}\frac{\partial T_{v_s}}{\partial x} + k_e\frac{\partial T_e}{\partial x} + \sum_{s=\text{all}} J_{x,s}h_s , \\ q_y &= k_{tr}\frac{\partial T}{\partial y} + \sum_{s=\text{mol.}} k_{v_s}\frac{\partial T_{v_s}}{\partial y} + k_e\frac{\partial T_e}{\partial y} + \sum_{s=\text{all}} J_{y,s}h_s , \\ q_{x,v_m} &= k_{v_m}\frac{\partial T_{v_m}}{\partial x} + J_{x,m}h_{v_m} , \\ q_{y,v_m} &= k_{v_m}\frac{\partial T_{v_m}}{\partial y} + J_{y,m}h_{v_m} , \\ q_{x,e} &= k_e\frac{\partial T_e}{\partial x} + \sum_{s=\text{all}} J_{x,s}h_{e_s} , \\ q_{y,e} &= k_e\frac{\partial T_e}{\partial y} + \sum_{s=\text{all}} J_{y,s}h_{e_s} . \end{aligned} \tag{6}$$

The vector of source terms is separated into geometric, chemistry, thermal energy exchange and radiation contributions in order to apply the operator-splitting integration approach, Eq. 7.

$$Q = Q_{\text{geom.}} + Q_{\text{chem.}} + Q_{\text{therm.}} + Q_{\text{rad.}} \tag{7}$$

The geometric source term vector for axisymmetric geometries is

$$
Q_{\text{geom.}} = \begin{bmatrix} 0 \\ 0 \\ (p - \tau_{\theta\theta}) A_{xy}/V \\ 0 \\ 0 \\ 0 \end{bmatrix} , \tag{8}
$$

where $A_{xy}$ is the projected area of the cell in the (x,y)-plane and

$$
\tau_{\theta\theta} = 2\mu \frac{u_y}{y} + \lambda \left( \frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{u_y}{y} \right) . \tag{9}
$$

For planar geometries $Q_{\text{geom.}}$ is a zero vector. See the original ICASE report [2] for a derivation of these terms.

The chemistry source term vector is

$$
Q_{\text{chem.}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \Omega_m^{VC} \\ \sum_{s=\text{ion.}} \Omega_s^{EC} \\ \dot{\omega}_s \end{bmatrix} , \tag{10}
$$

and the thermal energy-exchange source term vector is

$$
Q_{\text{therm.}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \Omega_m^{VT} + \Omega_m^{VV} + \Omega_m^{VE} \\ \sum_{s=\text{mol.}} \Omega_s^{EV} + \sum_{s=\text{all.}} \Omega_s^{ET} \\ 0 \end{bmatrix} , \tag{11}
$$

The radiation source term vector is

$$
Q_{\text{rad.}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -\nabla \cdot q_{rad} \\ 0 \\ -\nabla \cdot q_{rad} \\ 0 \end{bmatrix} \tag{12}
$$

where any purely vibrational component of radiative heat loss (or gain) has been neglected. The transport, thermodynamic and chemical kinetic source term models will be discussed in detail in Section 3.

## 2.2  Conserved Quantities and Fluxes for Three-Dimensional Flow

In three dimensions, we include the z-momentum so that the vector of conserved quantities becomes

$$
U = \begin{bmatrix} \rho \\ \rho u_x \\ \rho u_y \\ \rho u_z \\ \rho E \\ \rho e_{v_m} \\ \rho e_e \\ \rho f_s \end{bmatrix} ,
\tag{13}
$$

and the inviscid component of the fluxes becomes

$$
\overline{F}_i = \begin{bmatrix} \rho u_x \\ \rho u_x^2 + p \\ \rho u_y u_x \\ \rho u_z u_x \\ \rho E u_x + p u_x \\ \rho e_{v_m} u_x \\ \rho e_e u_x + p_e u_x \\ \rho f_s u_x \end{bmatrix} \hat{i} +
\begin{bmatrix} \rho u_y \\ \rho u_x u_y \\ \rho u_y^2 + p \\ \rho u_z u_y \\ \rho E u_y + p u_y \\ \rho e_{v_m} u_y \\ \rho e_e u_y + p_e u_y \\ \rho f_s u_y \end{bmatrix} \hat{j} +
\begin{bmatrix} \rho u_z \\ \rho u_z u_x \\ \rho u_z u_y \\ \rho u_z^2 + p \\ \rho E u_z + p u_z \\ \rho e_{v_m} u_z \\ \rho e_e u_z + p_e u_z \\ \rho f_s u_z \end{bmatrix} \hat{k} .
\tag{14}
$$

The viscous component is

$$
\overline{F}_v = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ \tau_{zx} \\ \tau_{xx} u_x + \tau_{yx} u_y + \tau_{zx} u_z + q_x \\ q_{x,v_m} \\ q_{x,e} \\ J_{x,s} \end{bmatrix} \hat{i} +
\begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{zy} \\ \tau_{xy} u_x + \tau_{yy} u_y + \tau_{zy} u_z + q_y \\ q_{y,v_m} \\ q_{y,e} \\ J_{y,s} \end{bmatrix} \hat{j} +
$$

$$
\begin{bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ \tau_{xz} u_x + \tau_{yz} u_y + \tau_{zz} u_z + q_z \\ q_{z,v_m} \\ q_{z,e} \\ J_{z,s} \end{bmatrix} \hat{k} ,
\tag{15}
$$

and the viscous stresses are

$$
\begin{aligned}
\tau_{xx} &= 2\mu\frac{\partial u_x}{\partial x} + \lambda\left(\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}\right) , \\
\tau_{yy} &= 2\mu\frac{\partial u_y}{\partial y} + \lambda\left(\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}\right) , \\
\tau_{zz} &= 2\mu\frac{\partial u_z}{\partial z} + \lambda\left(\frac{\partial u_x}{\partial x} + \frac{\partial u_y}{\partial y} + \frac{\partial u_z}{\partial z}\right) , \\
\tau_{xy} = \tau_{yx} &= \mu\left(\frac{\partial u_x}{dy} + \frac{\partial u_y}{dx}\right) , \\
\tau_{xz} = \tau_{zx} &= \mu\left(\frac{\partial u_x}{dz} + \frac{\partial u_z}{dx}\right) , \\
\tau_{yz} = \tau_{zy} &= \mu\left(\frac{\partial u_y}{dz} + \frac{\partial u_z}{dy}\right) .
\end{aligned}
\tag{16}
$$

## 2.3  Discretised Equations and Time-Stepping Procedure

The finite-volume core of `Eilmer3` is implemented for 3D flows with some of the components omitted when running a 2D simulation.

In 2D, the conservation equations are applied to straight-edged quadrilateral cells for which the boundary, projected onto the (x,y)-plane, consists of four straight lines (or cell interfaces) labelled North, East, South and West. In 3D, finite-volume cells are hexahedral with 6 (possibly-nonplanar) quadrilateral surfaces interfacing the neighbouring cells. Flux values are estimated at midpoints of the cell interfaces and the integral conservation equation (1) is approximated as the algebraic expression

$$
\frac{dU}{dt} = -\frac{1}{V}\sum_{cell-surface}\left(\overline{F}_i - \overline{F}_v\right)\cdot\hat{n}\,dA + Q ,
\tag{17}
$$

where $U$ and $Q$ now represent cell-average values.

An operator-splitting approach as advocated by Oran and Boris [3] (see Chapter 11 of their text) is applied whereby the physical mechanisms are applied in a decoupled fashion. The time integration of the ODE system shown in Eq. 17 is then approximated by

$$
\begin{aligned}
\int_{\Delta t}\frac{dU}{dt}dt &= \int_{\Delta t}\left(\frac{dU}{dt}\right)_{inv.}dt + \int_{\Delta t}\left(\frac{dU}{dt}\right)_{visc.}dt \\
&+ \sum_{N_c}\left[\int_{\Delta t_c}\left(\frac{dU}{dt}\right)_{chem.}dt\right] + \sum_{N_t}\left[\int_{\Delta t_t}\left(\frac{dU}{dt}\right)_{therm.}dt\right] ,
\end{aligned}
\tag{18}
$$

where,

$$\left(\frac{dU}{dt}\right)_{\text{inv.}} = -\frac{1}{V} \sum_{cell-surface} \left(\overline{F}_i\right) \cdot \hat{n}\, dA + Q_{\text{geom.}} + Q_{\text{rad.}} ,\qquad(19)$$

$$\left(\frac{dU}{dt}\right)_{\text{visc.}} = -\frac{1}{V} \sum_{cell-surface} \left(-\overline{F}_v\right) \cdot \hat{n}\, dA ,\qquad(20)$$

$$\left(\frac{dU}{dt}\right)_{\text{chem.}} = Q_{\text{chem.}} ,\qquad(21)$$

$$\left(\frac{dU}{dt}\right)_{\text{therm.}} = Q_{\text{therm.}} .\qquad(22)$$

Integration, in time, of the discretised equations proceeds in a *loosely coupled* fashion. The order of operations for a single time-step for a radiating gas in thermochemical nonequilibrium is shown in Figure 1. Some of the chemical kinetic and thermal energy-exchange ODE systems are "stiff" and so "subcycling" is used over the global integration time step via smaller steps if the system fails to solve. The number of chemical and thermal subcycles are,

$$N_c = \Delta t/\Delta t_c ,$$
$$N_t = \Delta t/\Delta t_t .$$

Currently the radiative source term vector, $Q_{rad}$, is applied closely coupled with the inviscid fluxes. This seems to be adequate for the work done thus far, but may need to be revised for strongly radiatively coupled flows.

The advantage of the operator-splitting approach is that the optimal integration scheme for each component of the physics can be implemented. This is especially useful for solving large chemical kinetic systems. The resultant set of ODE systems are integrated in a time via a simple predictor-corrector scheme for the inviscid and viscous increments, one of a selection of methods (including a method for stiff systems) for the chemistry increment (see Section 3) and the $4^{\text{th}}$ order Runge-Kutta-Fehlberg method for the thermal energy-exchange increment.

## 2.4 Multiple-Block Grids and Parallelisation

As shown in Figure 2, the data arrays for each block are dimensioned such that there is a buffer region, two cells deep, around the *active cells*, which completely defines the flow domain covered by the block. The buffer region contains *ghost cells* which are used to hold a copy of the flow information from adjacent blocks or to implement the boundary conditions.

For a boundary common to two blocks, the ghost cells in the buffer region of each block overlap the active cells of the adjacent block. The only interaction that occurs between blocks is the exchange of boundary data, prior to the reconstruction phase of each time step. For the *shared memory* version of the code, the exchange of cell-average data along the block boundaries takes place as a direct copy from the active-cell of one block to the ghost-cell of the other

1. compute gas transport due to inviscid flux:

    (a) apply inviscid boundary conditions or exchange data at boundaries for each block as appropriate

    (b) reconstruct the flow field sate on both sides of each interface

    (c) compute the inviscid fluxes $\overline{F_i} \cdot \hat{n}$

    (d) compute the radiative source term $-\nabla \cdot q_{rad}$ for each cell

    (e) integrate Eq. 19 over the timestep

    (f) decode the conserved quantities via an equation-of-state call

    (g) repeat for corrector update

2. compute gas transport due to viscous flux:

    (a) apply viscous boundary conditions at solid walls

    (b) compute the viscous fluxes as $\overline{F_v} \cdot \hat{n}$

    (c) integrate Eq. 20 over the timestep

    (d) decode the conserved quantities via an equation-of-state call

    (e) repeat for corrector update

3. compute change of gas state due to chemical reactions:

    (a) compute all chemical source terms

    (b) integrate Eq. 21 over the timestep

    (c) decode the conserved quantities via an equation-of-state call

    (d) redo via smaller subcycles if failed and apply call to equation-of-state more frequently

4. compute change of gas state due to thermal energy-exchange:

    (a) compute all chemical source terms

    (b) integrate Eq. 22 over the timestep

    (c) decode the conserved quantities via an equation-of-state call

    (d) redo via smaller subcycles if failed and apply call to equation-of-state more frequently

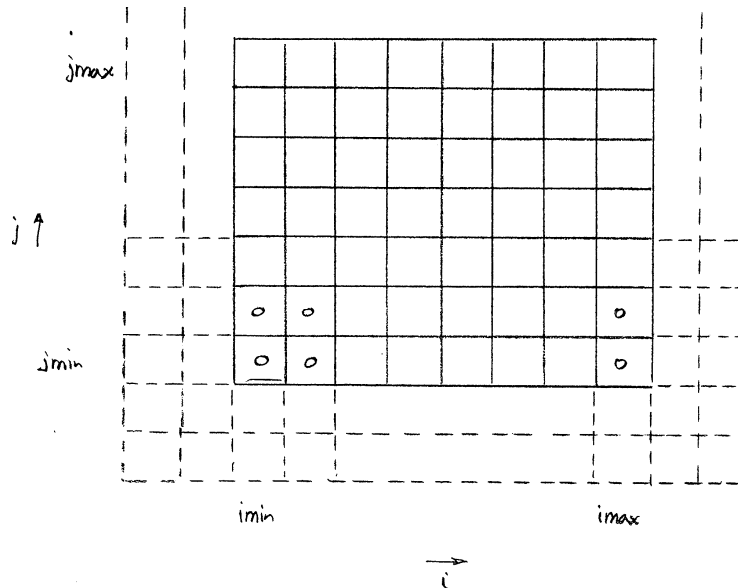Figure 1: Sequence of operations for a time-step update in `Eilmer3`.

Figure 2: Active and ghost cells for a single 2D block grid for `Eilmer3`.

block. Thus, the cells along the common boundary of each block must match in both number and position. Some logic is used within the exchange routines to set the appropriate indexing direction for each boundary. The information on the connections between block boundaries is stored in a (global) connectivity array. For each boundary on each block, this array stores the identity of the adjacent block and the name of the connecting boundary on the adjacent block.

Except for this block-to-block communication (and the occasional checking of time step magnitudes), the rest of the calculation can be done independently for all blocks. Thus, the algorithm is fairly easy to implement on a multiple-instruction, multiple-data (MIMD) parallel computer and we have a single-program-multiple-data (SPMD) version of the code for computationally-intensive facility calculations. When running such simulations, there are many copies of the program running independently on separate processors, with each copy of the program handling the computation for a single block. To exchange block-boundary data, each program instance must communicate with the other programs for adjacent blocks. The communication and synchronisation tasks are handled via a standard message passing library, MPI [4].

## 2.5 Boundary Conditions

The inviscid-component of applied boundary conditions is implemented by also filling in the ghost-cell data and then applying the normal reconstruction and flux calculation without further discrimination of the boundary cells. This approach covers solid/slip walls, inflow and outflow boundaries.

For viscous simulations, boundaries may also be assigned as fixed temperature, no-slip or catalytic (chemical equilibrium at wall gas state) boundary conditions. Such viscous boundary conditions also use data specified at the cell interfaces that lie along the boundary surface. These

13

data are used in the derivative calculations that subsequently feed into the viscous fluxes.

There are also boundary conditions that allow the user to specify the ghost-cell data and boundary interface data via user-written functions (via an embedded Lua interpreter). These functions may also be used to bypass the internal flux calculators and specify the boundary fluxes directly. When exploring specialized boundary conditions, such as the mixing-plane interface for turbomachinery calculations, the user can first implement them as Lua functions.

## 2.6  Inviscid Flux Calculation

The flow-states at the cell interfaces are calculated using a piecewise-parabolic scheme. Before computing the inviscid fluxes at each interface, the velocity field is rotated into a local $(n, p, q)$-coordinate system with unit vectors

$$
\begin{aligned}
\hat{n} &= n_x\,\hat{i} + n_y\,\hat{j} + n_z\,\hat{k} \quad, \\
\hat{p} &= p_x\,\hat{i} + p_y\,\hat{j} + p_z\,\hat{k} \quad, \\
\hat{q} &= q_x\,\hat{i} + q_y\,\hat{j} + q_z\,\hat{k} \quad,
\end{aligned}
\tag{23}
$$

where $\hat{n}$ is normal and $\hat{p}, \hat{q}$ are tangental to the cell interface. The normal and tangential velocity components

$$
\begin{aligned}
u_n &= n_x\,u_x + n_y\,u_y + n_z\,u_z \quad, \\
u_p &= p_x\,u_x + p_y\,u_y + p_z\,u_z \quad, \\
u_q &= q_x\,u_x + q_y\,u_y + q_z\,u_z \quad,
\end{aligned}
\tag{24}
$$

are then used, together with the other flow properties either side of the interface, to compute the inviscid fluxes

$$
\begin{bmatrix}
F_{mass} \\
F_{n-momentum} \\
F_{p-momentum} \\
F_{q-momentum} \\
F_{energy} \\
F_{species-s}
\end{bmatrix}
=
\begin{bmatrix}
\rho u_n \\
\rho u_n u_n + p \\
\rho u_n u_p \\
\rho u_n u_q \\
\rho u_n E + p u_n \\
\rho u_n f_s
\end{bmatrix}
,
\tag{25}
$$

in the local reference frame. These are then transformed back to the $(x, y, z)$ coordinate system as

$$
\overline{F} \cdot \hat{n} =
\begin{bmatrix}
F_{mass} \\
F_{x-momentum} \\
F_{y-momentum} \\
F_{z-momentum} \\
F_{energy} \\
F_{species-s}
\end{bmatrix}
=
\begin{bmatrix}
F_{mass} \\
F_{n-momentum}n_x + F_{p-momentum}p_x + F_{q-momentum}q_x \\
F_{n-momentum}n_y + F_{p-momentum}p_y + F_{q-momentum}q_y \\
F_{n-momentum}n_z + F_{p-momentum}p_z + F_{q-momentum}q_z \\
F_{energy} \\
F_{species-s}
\end{bmatrix}
.
\tag{26}
$$

For the simulation of shock and expansion tubes, the shock waves can be extremely strong so we use the default adaptive scheme in which the equilibrium flux method (EFM) [5] is applied near shocks and a modified AUSMDV calculator [6] is applied elsewhere.
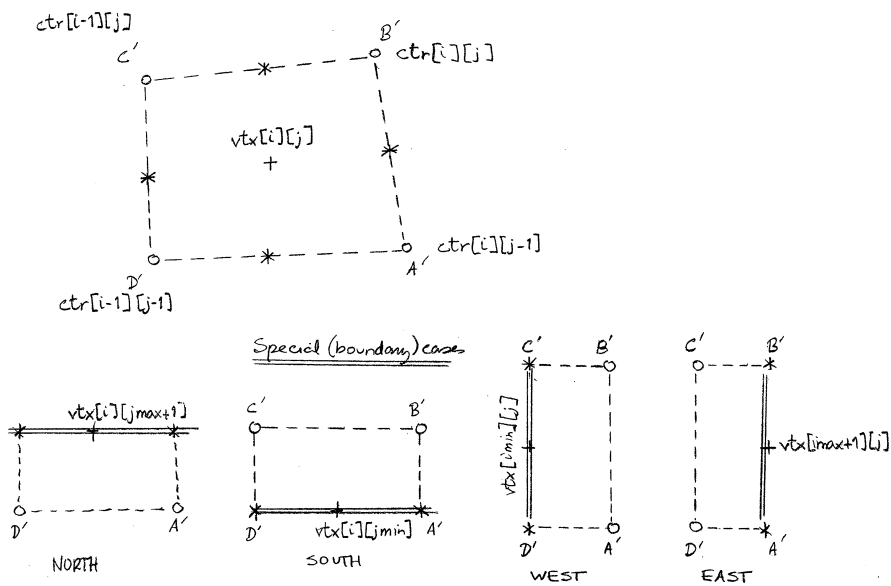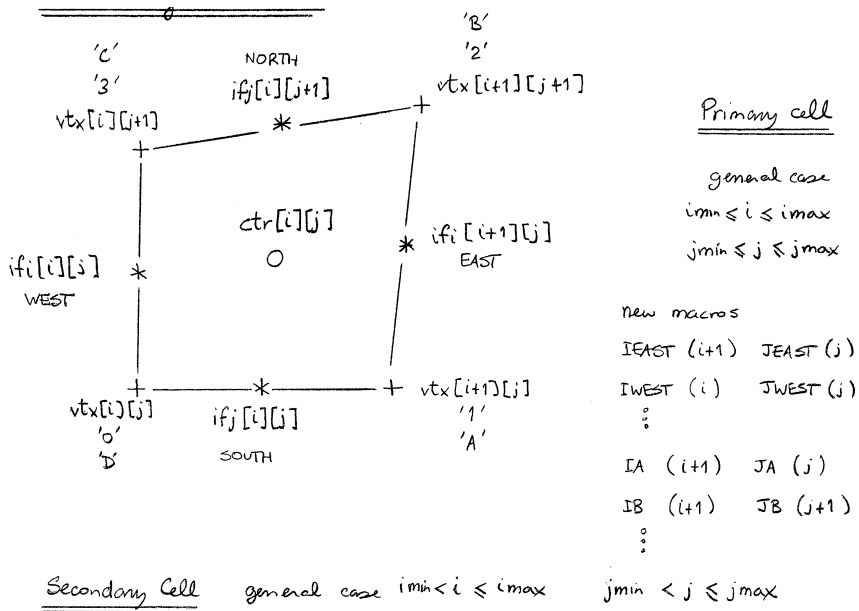
Figure 3: Cell, interface and vertex indexing in 2D for `Eilmer3`. The upper half of the figure shows the primary cells defining the finite-volumes for the conservation equations. The lower part of the figure shows the secondary cells, used for computing spatial derivatives.

### 2.6.1 Reconstruction

The primary data held by the code are cell-average data, associated with cell centres. To get the fluxes at cell interfaces, a variable-by-variable reconstruction is made of the flow field. This is done in a one-dimensional fashion, working along one-index direction at a time. *Left* and *Right* values ($w_L$ and $w_R$ respectively) of a flow variable at a cell interface are evaluated as the corresponding cell average value plus a limited higher-order interpolated increment. Given an array of cell-centres $[L1, L0, R0, R1]$ with an interface located between $L0$ and $R0$, the interpolated values are

$$
\begin{aligned}
w_L &= w_{L0} + \alpha_{L0} \left[ \Delta_{L+} \times (2h_{L0} + h_{L1}) + \Delta_{L-} \times h_{R0} \right] s_L \ , \\
w_R &= w_{R0} - \alpha_{R0} \left[ \Delta_{R+} \times h_{L0} + \Delta_{R-} \times (2h_{R0} + h_{R1}) \right] s_R \ , \\
\Delta_{L-} &= \frac{w_{L0} - w_{L1}}{\frac{1}{2}\left( h_{L0} + h_{L1} \right)} \ , \\
\Delta_{L+} &= \frac{w_{R0} - w_{L0}}{\frac{1}{2}\left( h_{R0} + h_{L0} \right)} = \Delta_{R-} \ , \\
\Delta_{R+} &= \frac{w_{R1} - wR0}{\frac{1}{2}\left( h_{R0} + h_{R1} \right)} \ , \\
\alpha_{L0} &= \frac{h_{L0}/2}{h_{L1} + 2h_{L0} + h_{L0}} \ , \\
\alpha_{R0} &= \frac{h_{R0}/2}{h_{L0} + 2h_{R0} + h_{R1}} \ ,
\end{aligned}
\tag{27}
$$

where the $h$ represents the width of a cell and the van Albada limiter [7] is implemented as

$$
\begin{aligned}
s_L &= \frac{\Delta_{L-}\Delta_{L+} + |\Delta_{L-}\Delta_{L+}|}{\Delta_{L-}^2 + \Delta_{L+}^2 + \epsilon} \ , \\
s_R &= \frac{\Delta_{R-}\Delta_{R+} + |\Delta_{R-}\Delta_{R+}|}{\Delta_{R-}^2 + \Delta_{R+}^2 + \epsilon} \ , \\
\epsilon &= 1.0 \times 10^{-12} \ .
\end{aligned}
$$

Finally, minimum and maximum limits are applied so that the newly interpolated values lie within the range of the original cell-centred values. Unlimited, this reconstruction scheme has third-order truncation errors and, with the limiter active, a sine function is reconstructed with an effective truncation error order of 2.7.

Typically, reconstruction is done for density, internal energy, velocity components, and species mass fractions. Other flow quantities that are needed at the interfaces for the inviscid flux calculation are then obtained from the thermochemical model.

### 2.6.2 EFM Calculation

The equilibrium flux method (EFM) [5] is used for its dissipative nature in the vicinity of very strong compressions. The method assumes that the gas is in equilibrium and the molecular

velocities of the gas either side of the interface can be described with the Boltzmann distribution. As implemented in Reference [8], the flux of mass from the left state, moving to the right is

$$F_{massL} = W_L^+ \, \rho_L \, u_{nL} + D_L^+ \, \rho_L \, \sqrt{2RT_L} \ , \tag{28}$$

where

$$
\begin{aligned}
W_L^+ &= \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{u_{nL}}{\sqrt{2RT_L}}\right)\right) \ , \\
D_L^+ &= \frac{1}{2\sqrt{\pi}}\exp\left(-\left(\frac{u_{nL}}{\sqrt{2RT_L}}\right)^2\right) \ , \\
\operatorname{erf}(s) &= \frac{2}{\sqrt{\pi}}\int_0^s \exp(-t^2)\,dt.
\end{aligned}
\tag{29}
$$

Similarly, the flux of mass from the right state, moving to the left is

$$F_{massR} = W_R^- \, \rho_R \, u_{nR} + D_R^- \, \rho_R \, \sqrt{2RT_R} \ , \tag{30}$$

where

$$
\begin{aligned}
W_R^- &= \frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{u_{nR}}{\sqrt{2RT_R}}\right)\right) \ , \\
D_R^- &= \frac{1}{2\sqrt{\pi}}\exp\left(-\left(\frac{u_{nR}}{\sqrt{2RT_R}}\right)^2\right) \ .
\end{aligned}
\tag{31}
$$

The flux vector components are then

$$
\begin{aligned}
F_{mass} &= F_{massL} + F_{massR} \ , \\
F_{n-momentum} &= F_{massL}\, u_{nL} + F_{massR}\, u_{nR} + W_L^+ \, p_L + W_R^- \, p_R \ , \\
F_{p-momentum} &= F_{massL}\, u_{pL} + F_{massR}\, u_{pR} \ , \\
F_{energy} &= \left(W_L^+ \, \rho_L \, u_{nL}\right)\left(e_L + \frac{p_L}{\rho_L} + \frac{1}{2}\left(u_{nL}^2 + u_{pL}^2\right)\right) \\
&\quad + \left(W_R^- \, \rho_R \, u_{nR}\right)\left(e_R + \frac{p_R}{\rho_R} + \frac{1}{2}\left(u_{nR}^2 + u_{pR}^2\right)\right) \\
&\quad + \left(D_L^+ \, \sqrt{2RT_L} \, \rho_L\right)\left(\frac{1}{2}\left(u_{nL}^2 + u_{pL}^2\right) + \frac{1}{2}\frac{\gamma+1}{\gamma-1}RT_L\right) \\
&\quad + \left(D_R^- \, \sqrt{2RT_R} \, \rho_R\right)\left(\frac{1}{2}\left(u_{nR}^2 + u_{pR}^2\right) + \frac{1}{2}\frac{\gamma+1}{\gamma-1}RT_R\right) \ .
\end{aligned}
\tag{32}
$$

Species mass fractions are just transported by the net mass flux as scalar quantities. Note that the gas constants, $R$ and $\gamma$, are not really constant; they are density-weighted averages derived from the local values for left and right gas states.

### 2.6.3 AUSMDV calculation

Most of the flow field fluxes are computed with the AUSMDV [6] because of its reasonably low dissipation. The calculation procedure starts by computing the weighting parameters for

the velocity splitting

$$\alpha_L = \frac{2\,p_L/\rho_L}{p_L/\rho_L + p_R/\rho_R} \;,$$

$$\alpha_R = \frac{2\,p_R/\rho_R}{p_L/\rho_L + p_R/\rho_R} \;, \tag{33}$$

and the sound speed and Mach numbers in the normal direction to the interface

$$a_m = \max(a_L, a_R) \;,$$

$$M_L = \frac{u_{nL}}{a_m} \;,$$

$$M_R = \frac{u_{nR}}{a_m} \;. \tag{34}$$

The components from pressure splitting are then

$$p_L^+ = \frac{p_L}{4}(M_L + 1)^2\,(2 - M_L) \;, \quad |M_L| \le 1.0 \;,$$

$$= \frac{p_L\,du_L}{u_{nL}} \;, \quad \text{otherwise} \;,$$

$$p_R^- = \frac{p_R}{4}(M_R - 1)^2\,(2 + M_R) \;, \quad |M_R| \le 1.0 \;,$$

$$= \frac{p_R\,du_R}{u_{nR}} \;, \quad \text{otherwise} \;, \tag{35}$$

where $du_L = \frac{(u_{nL} + |u_{nL}|)}{2}$ and $du_R = \frac{(u_{nR} - |u_{nR}|)}{2}$. The components from the velocity splitting are

$$u_L^+ = \alpha_L\left(\frac{(u_{nL} + a_m)^2}{4\,a_m} - du_L\right) + du_L \;, \quad |M_L| \le 1.0 \;,$$

$$= du_L \;, \quad \text{otherwise} \;,$$

$$u_R^- = -\alpha_R\left(\frac{(u_{nR} - a_m)^2}{4\,a_m} + du_R\right) + du_R \;, \quad |M_R| \le 1.0 \;,$$

$$= du_R \;, \quad \text{otherwise} \;, \tag{36}$$

These components are then combined into a mass flux

$$(\rho u)_{\frac{1}{2}} = u_L^+\,\rho_L + u_R^-\,\rho_R \tag{37}$$

and a pressure flux

$$p_{\frac{1}{2}} = p_L^+ + p_R^- \tag{38}$$

and a normal-momentum flux $(\rho u^2)_{\frac{1}{2}}$ as a blend of AUSMV and AUSMD fluxes

$$(\rho u^2)_{AUSMV} = u_L^+\,\rho_L\,u_{nL} + u_R^-\,\rho_R\,u_{nR} \;,$$

$$(\rho u^2)_{AUSMD} = \frac{1}{2}\left((\rho u)_{\frac{1}{2}}(u_{nL} + u_{nR}) - |(\rho u)_{\frac{1}{2}}|(u_{nR} - u_{nL})\right) \;,$$

$$(\rho u^2)_{\frac{1}{2}} = (\frac{1}{2} + s)(\rho u^2)_{AUSMV} + (\frac{1}{2} - s)(\rho u^2)_{AUSMD} \;, \tag{39}$$

with the switching function, $s$, based on the pressure gradient

$$s = \frac{1}{2} \min \left( 1, K \frac{|p_R - p_L|}{\min(p_L, p_R)} \right) \; , \tag{40}$$

with $K = 10$.

The flux vector components can be assembled from these pieces as

$$\begin{aligned} F_{mass} &= (\rho u)_{\frac{1}{2}} \; , \\ F_{n-momentum} &= (\rho u^2)_{\frac{1}{2}} + p_{\frac{1}{2}} \; , \end{aligned} \tag{41}$$

and depending on which way the wind is blowing, the remaining flux vector components are assembled from either the right or left flow properties. For $(\rho u)_{\frac{1}{2}} \geq 0$,

$$\begin{aligned} F_{p-momentum} &= (\rho u)_{\frac{1}{2}} \, u_{pL} \; , \\ F_{energy} &= (\rho u)_{\frac{1}{2}} \, H_L \; . \end{aligned} \tag{42}$$

$$\tag{43}$$

otherwise

$$\begin{aligned} F_{p-momentum} &= (\rho u)_{\frac{1}{2}} \, u_{pR} \; , \\ F_{energy} &= (\rho u)_{\frac{1}{2}} \, H_R \; . \end{aligned} \tag{44}$$

$$\tag{45}$$

where $H = e + \frac{p}{\rho} + \frac{1}{2} \left( u_n^2 + u_p^2 \right)$ is the total enthalpy of the gas. Again, species mass fractions are just transported by the mass flux as scalar quantities.

Finally, an entropy fix is applied, as per Section 3.5 in Reference [6]. This first determines if the interface includes an expansion sonic point

$$\text{Case A: } u_{nL} - a_L < 0 \text{ and } u_{nR} - a_R > 0$$
$$\text{Case B: } u_{nL} + a_L < 0 \text{ and } u_{nR} + a_R > 0$$

and increments the flux if only a single expansion wave is detected

$$\begin{aligned} F_{mass} &-= \Delta_{ua} \left( \rho_R - \rho_L \right) \; , \\ F_{n-momentum} &-= \Delta_{ua} \left( \rho_R u_{nR} - \rho_L u_{nL} \right) \; , \\ F_{p-momentum} &-= \Delta_{ua} \left( \rho_R u_{pR} - \rho_L u_{pL} \right) \; , \\ F_{energy} &-= \Delta_{ua} \left( \rho_R H_R - \rho_L H_L \right) \; , \end{aligned} \tag{46}$$

where

$$\begin{aligned} \Delta_{ua} &= 0.125((u_{nR} - a_R) - (u_{nL} - a_L)) \; , \text{ for A and not B,} \\ &= 0.125((u_{nR} + a_R) - (u_{nL} + a_L)) \; , \text{ for B and not A,} \\ &= 0 \; , \text{ otherwise.} \end{aligned} \tag{47}$$

### 2.6.4 Shock Detector

The switching between the two flux calculators is governed by a shock (or compression) detector. This is simply a measure of the relative change in normal velocity at interfaces. Specifically, we indicate a strong compression at cell-interface $i + \frac{1}{2}$ when

$$\frac{u_{n,i+1} - u_{n,i}}{\min(a_{i+1}, a_i)} < \text{Tol} \ , \tag{48}$$

where Tol is the compression tolerance and is typically set at -0.05. This measure is applied to all interfaces in a block and then a second pass propagates the information to near-by interfaces. If a first cell-interface is identified as having a strong compression, the EFM flux calculator is used for all interfaces attached to the cell containing that first cell-interface.

## 2.7 Viscous Fluxes

The viscous flux calculation is then performed based on the the updated cell-centre flow state. The spatial derivatives required in the viscous stress and heat flux terms, Eq. 5 and 6, are evaluated as

$$\nabla \phi = \frac{1}{V} \oint_S \phi \hat{n} \, dA, \tag{49}$$

where $\phi$ a scalar quantity and the surface $S$ defines the secondary cell surrounding a primary-cell vertex, as shown in Figure 3. When used in the viscous fluxes, the vertex values are averaged to obtain a midface value. All evaluations are done within a block and secondary cells of half size are used along the boundary faces of the block. Viscous boundary conditions for velocity (*e.g.* no slip) and temperature are applied by using data that was specified at cell interfaces along the boundary. In 3D, further special cases for the derivatives are encountered for the edges of the block, which are evaluated with a least squares fit of the function $\phi = a\,x + b\,y + c\,z + d$, and for corners where there is just enough data to fit the same linear function. In 2D, a line integral corresponding to Eq.(49) is done, just within the (x,y)-plane. Also in 2D, only the corners need to be treated by fitting a bilinear function.

# 3 Thermochemistry

The gas-dynamic equations of the previous section are closed by a set of relations between the various thermodynamic properties of the gas mixture. This section describes a gas library which provides various models of gas behaviour for use in the simulation codes.

There are a number of gas models available as part of the gas library. They are desribed here:

**ideal gas mix**

> The ideal gas mix is used to model one or more components, all of which have ideal (perfectly elastic collisions and calorically perfect) behaviour.

**equilibrium gas mix**

The equilibrium gas mix models a fixed-composition gas mix which is assumed to be in thermal and chemical equilibrium at the local thermodynamic conditions.

**thermally perfect gas mix**

The thermally perfect gas mix models a gas with one or more components, all of which have perfect (collisional) behaviour but each have all internal energy modes excited to an equilibrium described by a single temperature

**multi-temperature gas mix**

This model represents a mixture of gases with perfect collisional behaviour but the the internal energy modes are ascribed different Boltzmann temperatures in order to model the effect of thermal nonequilibrium.

**Noble-Abel gas mix**

The Nobel-Abel gas mix model is used in interior ballistics work where high pressures mean that real gas effects cannot be neglected. It models the effect of the finite volume occupied by gas particles when computing the equation of state. Collisions between gas particles are still considered to be perfectly elastic.

**van der Waals gas mix**

The van der Waals gas mix models real gas effects at high pressures/densities. The finite volume occupied by the gas particles and the non-ideal particle collisions due to van der Waals' forces are considered in the model.

In simulating impulse facilities, we typically only use the equilibrium gas and thermally perfect gas models. Therefore, these two models are discussed in the remainder of this section. The thermally perfect gas mix is used when we also wish to simulate the effect of finite-rate chemistry between the gases. The multi-temperature gas mix has also been used recently to model the X2 facilities [9], however, we consider the validation of this model a work in progress.

## 3.1   An equilibrium gas mixture

At higher temperatures, the gases in impulse facilities will undergo chemical reactions. For example in air, by $2\,000\,\mathrm{K}$ oxygen molecules will begin to dissociate, and by $4\,000\,\mathrm{K}$ nitrogen molecules will begin to dissociate. When the chemical reactions are rapid compared to the flow transit times, we use a model of the gas which assumes chemical equilibrium. Along with the assumption of chemical equilibrium, thermal equilibrium is also assumed; the internal energy modes of the gas rapidly equilibrate with the translational temperature when compared to the characteristic flow time. This assumption is most appropriate in the high pressure, high temperature gas that is driven by the piston against an unruptured diaphgram: the high pressures and temperatures give rise to rapid changes in chemical composition.

The equilibrium gas mix is implemented as a look-up table where the thermodynamic properties are interpolated (or extrapolated) from a table with indexing based on density and internal energy. The look-up table is built for a fixed gas composition over a range of densities and energies prior to the gas dynamics simulation and is read into memory at the start of the simulation. A tool provided in the gas library builds the look-up table by running the CEA2 program [10] numerous times.

## 3.2 A mixture of thermally perfect gases

As mentioned above, the gas model for a mixture of thermally perfect gases is often used in conjunction with a finite-rate chemistry simulation. The thermodynamic relations for the gas mixture are presented here. The implementation of finite-rate chemical effects is discussed in Section 3.3.

### 3.2.1 A single thermally perfect gas

The assumed behaviour of a thermally perfect gas is that all internal energy modes are in equilibrium at a single temperature. For atoms this means that the Boltzmann distributions for translational and electronic energy are governed by one temperature value. Similarly for molecules, the Boltzmann distributions for translational, rotational, vibrational and electronic energy are described by a single temperature value.

To model a thermally perfect gas requires a knowledge of how the gas stores energy as a function of temperature. It is convenient to have available the specific heat at constant pressure as a function of temperature, $C_p(T)$. From this, specific enthalpy of the gas can be computed as

$$h = \int_{T_{ref}}^{T} C_p(T)dT + h(T_{ref}) \tag{50}$$

and entropy is given as

$$s = \int_{T_{ref}}^{T} \frac{C_P(T)}{T}dT + s(T_{ref}). \tag{51}$$

The transport properties, viscosity and thermal conductivity, can be calculated as a function of temperature for a single component of the gas mix. The transport properties for a single component can be combined by an appropriate mixing rule to give a mixture viscosity and thermal conductivity.

In the implementation, a thermally perfect gas is characterised by five curve fits all of which are functions of temperature:

1. specific heat at constant pressure, $C_p(T)$,

2. enthalpy, $h(T)$,

3. entropy, $s(T)$,

4. viscosity, $\mu(T)$, and

5. thermal conductivity, $k(T)$.

The form of these curve fits follows that used by McBride and Gordon [10]. The curve fits for thermodynamic properties in non-dimensional form are as follows:

$$\frac{C_p(T)}{R} = a_0 T^{-2} + a_1 T^{-1} + a_2 + a_3 T + a_4 T^2 + a_5 T^3 + a_6 T^4 \tag{52}$$

$$\frac{H(T)}{RT} = -a_0 T^{-2} + a_1 T^{-1} \log T + a_2 + a_3 \frac{T}{2} + a_4 \frac{T^2}{3} + a_5 \frac{T^3}{4} + a_6 \frac{T^4}{5} + \frac{a_7}{T} \tag{53}$$

$$\frac{S(T)}{R} = -a_0 \frac{T^{-2}}{2} - a_1 T^{-1} + a_2 \log T + a_3 T + a_4 \frac{T^2}{2} + a_5 \frac{T^3}{3} + a_6 \frac{T^4}{4} + a_8 \tag{54}$$

The coefficients for these curve fits are available for a large number of gaseous species in the CEA program [10] (and associated database files). Each of these curve fits are only valid over a limited temperature range. For example, the thermodynamic curve fits for molecular nitrogen ($N_2$) are comprised of three segments: 200.0–1000.0 K, 1000.0–6000.0 K and 6000.0–20000.0 K. Beyond this range the values are extrapolated in this work. The extrapolations are based on a crude assumption of constant $C_p$ outside of the range. Thus the extrapolations are as follows:

$$\frac{C_p(T < T_{low})}{R} = \frac{C_p(T_{low})}{R}$$

$$\frac{C_p(T > T_{high})}{R} = \frac{C_p(T_{high})}{R}$$

$$\frac{H(T < T_{low})}{RT} = \frac{1}{T} \{H(T_{low})T_{low} - C_p(T_{low})(T_{low} - T)\}$$

$$\frac{H(T > T_{high})}{RT} = \frac{1}{T} \{H(T_{high})T_{high} + C_p(T_{high})(T - T_{high})\}$$

$$\frac{S(T < T_{low})}{R} = S(T_{low}) - C_p(T_{low}) \log \left(\frac{T_{low}}{T}\right)$$

$$\frac{S(T > T_{high})}{R} = S(T_{high}) + C_p(T_{high}) \log \left(\frac{T}{T_{high}}\right)$$

The curve fits for viscosity and thermal conductivity are also in the same form as that used by the CEA program [10]. The curves are as follows.

$$\log \mu(T) = a_0 \log T + \frac{a_1}{T} + \frac{a_2}{T^2} + a_3$$

$$\log k(T) = b_0 \log T + \frac{b_1}{T} + \frac{b_2}{T^2} + b_3$$

### 3.2.2 Mixing rules for a collection of thermally perfect gases

The thermodynamic state for a mixture of thermally perfect gases is uniquely defined by two state variables and the mixture composition. The internal energy is computed as a mass fraction weighted sum of individual internal energies,

$$e = \sum_{i=1}^{N} f_i e_i = \sum_{i=1}^{N} f_i \left( h_i - R_i T \right). \tag{55}$$

Pressure is computed from Dalton's law of partial pressures,

$$p = \sum_{i=1}^{N} \rho_i R_i T. \tag{56}$$

The specific gas constant for the mixture is defined as

$$R = \sum_{i=1}^{N} f_i R_i. \tag{57}$$

The calculation of $C_p$ is based on a mass fraction weighted sum of component specific heats,

$$C_p = \sum_{i=1}^{N} f_i C_{p_i}. \tag{58}$$

The specific heat at constant volume is then computed as

$$C_v = C_p - R. \tag{59}$$

The ratio of specific heats, $\gamma$, is given by its definition,

$$\gamma = \frac{C_p}{C_v}. \tag{60}$$

The frozen sound speed for the mixture, $a$, is calculated as

$$a = \sqrt{\gamma R T}. \tag{61}$$

During a compressible flow simulation, the values of $\rho$ and $e$ are most readily available from the conserved quantities that are solved for during each time increment. This leads to the specific problem of solving for the thermodynamic state of the gas mixture given $\rho$, $e$, and the mixture composition, $\overrightarrow{f}$. However, the formulae previously presented are all explicit in temperature. We solve for temperature using the Newton iteration technique for zero solving,

$$T_{n+1} = T_n - \frac{f_0(T_n)}{f_0'(T_n)}, \tag{62}$$

where the zero function, $f_0(T)$, is based on the given internal energy, $e$, and a guess for internal energy based on temperature,

$$f_0(T) = e_{guess} - e = \sum_{i=1}^{N} f_i \left( h_i - R_i T_{guess} \right) - e. \tag{63}$$

Using the fact that $C_{vi} = \frac{de_i}{dt}$, we can conveniently find the derivative function for the Newton technique by computing the mixture $C_v$,

$$\frac{df_0(T)}{dT} = \sum_{i=1}^{N} f_i \frac{de_i}{dT} = \sum_{i=1}^{N} f_i C_{vi} = C_v. \tag{64}$$

The Newton iteration is set to converge when the accuracy of the temperature value is within $\pm 1.0 \times 10^{-6}$ K. Personal experience has shown that this kind of error tolerance is required when temperature is used in a finite-rate chemistry calculation to compute rates of composition change.

The calculation of mixture transport properties is not as straight forward as the thermodynamic properties. A mixing rule is required to compute the mixture viscosity and thermal conductivity. Wilke's mixing rule [11] has been implemented in the work presented here. Specifically, the mixing rules used by Gordon and McBride [12] in the CEA program are used for calculating mixture transport properties in this work; these rules are a variant of Wilke's original formulation [11].

$$\mu_{mix} = \sum_{i=1}^{N} \frac{x_i \mu_i}{x_i + \sum_{\substack{j=1 \\ j \neq i}}^{N} x_j \phi_{ij}} \tag{65}$$

and

$$k_{mix} = \sum_{i=1}^{N} \frac{x_i k_i}{x_i + \sum_{\substack{j=1 \\ j \neq i}}^{N} x_j \psi_{ij}} \tag{66}$$

where $x_i$ is the mole fraction of species $i$.

The interaction potentials, $\phi_{ij}$ and $\psi_{ij}$, can be calculated a number of ways. Again, the formulae suggested by Gordon and McBride [12] have been used,

$$\phi_{ij} = \frac{1}{4} \left[ 1 + \left( \frac{\mu_i}{\mu_j} \right)^{1/2} \left( \frac{M_j}{M_i} \right)^{1/4} \right]^2 \left( \frac{2M_j}{M_i + M_j} \right)^{1/2} \tag{67}$$

and

$$\psi_{ij} = \phi_{ij} \left[ 1 + \frac{2.41(M_i - M_j)(M_i - 0.142M_j)}{(M_i + M_j)^2} \right] \tag{68}$$

where $M_i$ and $M_j$ refer to the molecular weights of species $i$ and $j$ respectively.

Once the mixture viscosity and thermal conductivity have been computed, it is possible to compute the mixture Prandtl number from its definition

$$Pr = \frac{\mu C_p}{k}. \tag{69}$$

## 3.3 Finite-rate chemistry implementation

### 3.3.1 Rates of species change due to chemical reaction

By assuming a collection of simple reversible reactions, the chemically reacting system can be represented as,

$$\sum_{i=1}^{N} \alpha_i X_i \rightleftharpoons \sum_{i=1}^{N} \beta_i X_i, \tag{70}$$

where $\alpha_i$ and $\beta_i$ represent the stoichiometric coefficients for the reactants and products respectively. The case of an irreversible reaction is represented by setting the backward rate to zero. For a given reaction $j$, the rate of concentration change of species $i$ is given as,

$$\left( \frac{d[X_i]}{dt} \right)_j = \nu_i \left\{ k_f \prod_i [X_i]^{\alpha_i} - k_b \prod_i [X_i]^{\beta_i} \right\}, \tag{71}$$

where $\nu_i = \beta_i - \alpha_i$. By summation over all reactions, $N_r$, the total rate of concentration change is,

$$\frac{d[X_i]}{dt} = \sum_{j=1}^{N_r} \left( \frac{d[X_i]}{dt} \right)_j . \tag{72}$$

For certain integration schemes it is convenient to have the production and loss rates available as separate quantities. In this case,

$$\frac{d[X_i]}{dt} = q_i - L_i = \sum_{j=1}^{N_r} \dot{\omega}_{app_{i,j}} - \sum_{j=1}^{N_r} \dot{\omega}_{va_{i,j}} \tag{73}$$

The calculation of $\dot{\omega}_{app_{i,j}}$ and $\dot{\omega}_{va_{i,j}}$ depends on the value of $\nu_i$ in reach reaction $j$ as shown in Table 1.

Table 1: The form of the chemical production and loss terms based on the value of $\nu_i$

| | $\nu_i > 0$ | $\nu_i < 0$ |
|---|---|---|
| $\dot{\omega}_{app_i}$ | $\nu_i k_f \prod_i [X_i]^{\alpha_i}$ | $-\nu_i k_b \prod_i [X_i]^{\beta_i}$ |
| $\dot{\omega}_{va_i}$ | $-\nu_i k_b \prod_i [X_i]^{\beta_i}$ | $\nu_i k_f \prod_i [X_i]^{\alpha_i}$ |

The calculation of the reaction rate coefficients, $k_f$ and $k_b$, and the solution methods for the ordinary differential equation system of species concentration changes are discussed in the subsequent sections.

### 3.3.2 Reaction rate coefficients

The reaction rate coefficients for a reaction can be determined by experiment (often shock tube studies are used) or from theory. In a great number of cases, estimates of the reaction rate from

theory can vary by orders of magnitude from experimentally determined values. For this reason, fits to experimental values are most commonly used.

For the single-temperature gas model discussed in this chapter, the forward reaction rate coefficients are calculated using the generalised Arrhenius form,

$$k_f = AT^n \exp\left(\frac{-E_a}{kT}\right) \tag{74}$$

where $k$ is the Boltzmann constant and $A$, $n$ and $E_a$ are constants of the model.

The backward rate coefficient can also be calculated using a modified Arrhenius form,

$$k_b = AT^n \exp\left(\frac{-E_a}{kT}\right) \tag{75}$$

or it can be calculated by first calculating the equilibrium constant for the reaction,

$$k_b = \frac{k_f}{K_c}. \tag{76}$$

If the backward rate coefficient is calculated from the equilibrium constant, then a method of calculation of the equilibrium constant is required. The equilibrium constant for a specific reaction can be calculated from curve fits or, as is done in this work, using the principles of thermodynamics. The equilibrium constant based on concentration is related to the equilibrium constant based on pressure by,

$$K_c = K_p \left(\frac{p_{atm}}{\mathcal{R}T}\right)^\nu \tag{77}$$

where $p_{atm}$ is atmospheric pressure in Pascals, $\mathcal{R}$ is the universal gas constant, $\nu = \sum_i^{N_S} \nu_i$ and

$$K_p = \exp\left(\frac{-\Delta G}{\mathcal{R}T}\right). \tag{78}$$

The derivation of the formula for $K_p$, the equilibrium constant based on partial pressures, can be found in any introductory text on classical thermodynamics which covers chemical equilibrium. The differential Gibbs function for the reaction, $\Delta G$, is calculated using

$$\Delta G = \sum_i^{N_s} \nu_i G_i \tag{79}$$

where each $G_i$ is computed from the definition of Gibbs free energy,

$$G_i(T) = H_i(T) - T \times S_i(T) \tag{80}$$

and $G_i$ is in units of J/mol. $H_i$ and $S_i$ can be computed in the appropriate units by using the CEA polynomials and multiplying by $\mathcal{R}T$ and $\mathcal{R}$ respectively.

Some caution should be exercised in the selection and use of reaction rates for a specific flow problem. In many cases, a set of reaction rates may only be "tuned" for a specific problem domain. This problem of "tuned" sets of reaction rates and an explanation for why it arises is described by Oran and Boris (p. 38 of Ref. [13]):

A problem that often arises in chemical reactions is that there are fundamental inconsistencies in a measured reaction rate. For example, there may be experimental measurements of both the forward and reverse rate constants, $k_f$ and $k_r$. Nonetheless, when either is combined with the equilibrium coefficient for that reaction, the other is not produced. This appears to represent a violation of equilibrium thermodynamics. The explanation is usually that $k_f$ and $k_r$ have been measured at rather different temperatures or pressures, and so there are inconsistencies when they are extrapolated outside the regime of validity of the experiments.

### 3.3.3 Solving the chemical kinetic ordinary differential equation

The system represented in Equation 72 is a system of ordinary differential equations (ODEs) which can be solved by an appropriate method. For certain chemical systems, the governing ODEs form a stiff system due to rates of change varying by orders of magnitude for certain species. For these systems, special methods for stiff ODEs are required. In this work, four methods for the numerical solution of the ODE system have been implemented.

1. Euler method

2. modified Euler method

3. alpha-QSS method, and

4. Runge-Kutta-Fehlberg method

The Euler method and modified Euler method are standard techniques for solving ODEs and the details can be found in any text dealing with numerical methods and numerical analysis. The fourth-order Runge-Kutta method uses a fifth-order error estimate as a means for controlling the timestep used for integration as proposed by Fehlberg [14]. This is particularly efficient for non-stiff systems.

**alpha-QSS method**    The alpha-QSS (quasi-steady-state) method was proposed in Mott's thesis [15]. It is an ODE solver aimed specifically at the problem of stiffness in chemical systems. This ODE solver makes use of the forward and backwards rates of concentration change as calculated by Equation 73. This is a predictor-corrector type scheme in which the corrector is iterated upon until a desired convergence is achieved. The predictor and corrector are,

$$[X_i]^1 = [X_i]^0 + \frac{\Delta t q_i^0}{1 + \alpha_i^0 \Delta t L_i^0} \tag{81}$$

$$[X_i]^{n+1} = [X_i]^0 + \frac{\Delta t \left( \bar{q}_i - [X_i]^0 \bar{L}_i \right)}{1 + \bar{\alpha}_i \Delta t \bar{L}_i}. \tag{82}$$

In the above equations,

$$\bar{L}_i = \frac{1}{2} \left( L_i^0 + L_i^n \right) \tag{83}$$

and

$$\bar{q}_i = \bar{\alpha}_i q_i^n + (1 - \bar{\alpha}_i) Q_i^0. \tag{84}$$

The key to the scheme is calculating $\alpha$ correctly. This $\alpha$ parameter controls how the update works on a given species integration. Note that $\alpha$ is defined as

$$\alpha(L\Delta t) \equiv \frac{1 - \left(1 - e^{-L\Delta t}\right)/(L\Delta t)}{1 - e^{-L\Delta t}}. \tag{85}$$

Using Pade's approximation,

$$e^x \approx \frac{360 + 120x + 12x^2}{360 - 240x + 72x^2 - 12x^3 + x^4} \tag{86}$$

it is possible to write a form of the expression for $\alpha$ which is more amenable to computation as the expensive exponential function evaluation is avoided. The approximation for $\alpha$ becomes,

$$\alpha(L\Delta t) \approx \frac{180r^3 + 60r^2 + 11r + 1}{360r^3 + 60r^2 + 12r + 1} \tag{87}$$

where $r \equiv 1/(L\Delta t)$.

### 3.3.4 Coupling chemistry effects to the flow solver

Some details about the coupling of the chemistry effects to the gas dynamics simulation are provided here. In an unsteady, time-accurate flow simulation, the allowable timestep is constrained by the Courant-Friedrichs-Lewy (CFL) criterion. In a viscous compressible flow, the CFL criterion allows one to select an appropriate timestep and limit the propagation of flow information to distances less than one cell-width. The speed at which flow information propagates is a function of inviscid wave speeds and viscous effects.

When the effects of finite-rate chemistry are 'split' from the flow simulation, the chemical update is solved in a separate step in which the flow is held frozen. (In fact, in true timestep-splitting, *all* other contributing physics is frozen during the chemistry update.) Thus the chemistry problem is to find the updated species composition at the end of the flow timestep.

It may be, and is quite likely, that the flow timestep is not an appropriate timestep to solve the chemical kinetic ODE problem. When the timestep for the chemistry problem is smaller than the flow timestep, the chemistry problem is subcycled a number of times until the total elapsed time equals that of the flow timestep. It is common to have simulations where the chemistry timestep is 100–1000 times smaller than the flow timestep, that is, 100-1000 subcycles are required to solve the chemistry problem. When the timstep for the chemistry problem is larger than the flow timestep, it is simply set to the value of the flow timestep.

During the simulation process, the chemistry timestep is tracked for each finite-volume cell in the simulation. Although the flow 'moves on' in subsequent timesteps, if the change of flow conditions is not large, then the previous chemistry timestep will be a good estimate to begin

the new chemistry problem in the subsequent timestep. An exceptional case is when a shock passes through the cell: the change of flow conditions does become large. In this instance, the old chemistry step is disregarded and a new step is selected. The selection procedure for a new step is discussed in the next paragraph. When using either the Runge-Kutta-Fehlberg or the alpha-QSS methods, an estimate of the new chemistry timestep is provided as part of the ODE update routine.

So, during a simulation, the old chemistry step at one iteration is used to begin the new chemistry problem in the next iteration. What is needed is a means to select the chemistry step on the initial iteration, or whenever the old suggestion is not reasonable (as in the case of a shock passing through the cell). In this work, the initial step for the chemistry problem is selected based on the suggestion by Young and Boris [16],

$$ dt_{\text{chem}} = \epsilon_1 \min \left( \frac{[X_i](0)}{[\dot{X}_i](0)} \right) \tag{88} $$

where $\epsilon_1$ is taken as $1.0 \times 10^{-3}$ in this work, and the expression is evaulated at the initial values for the chemistry subproblem. Young and Boris [16] suggest that $\epsilon_1$ be scaled from the convergence criteria. We have found that the fixed value is adequate for the problems of interest to our research group.

# References

[1] P. A. Jacobs and R. J. Gollan. The Eilmer3 code: User guide and example book. Mechanical Engineering Report 2008/07, The University of Queensland, Brisbane, Australia, 2010.

[2] P. A. Jacobs. Single-block Navier-Stokes integrator. ICASE Interim Report 18, 1991.

[3] E. Oran and J. Boris. *Numerical Simulation of Reactive Flow*. Cambridge University Press, New York, USA, 2nd edition, 2001.

[4] W. Gropp, E. Lusk, and A. Skjellum. *Using MPI Portable Parallel Programming with the Message-passing Interface.* The MIT Press, Cambridge, Massachusetts, 1994.

[5] M. N. Macrossan. The equilibrium flux method for the calculation of flows with non-equilibrium chemical reactions. *Journal of Computational Physics*, 80(1):204–231, 1989.

[6] Y. Wada and M. S. Liou. A flux splitting scheme with high-resolution and robustness for discontinuities. AIAA Paper 94-0083, January 1994.

[7] G. D. van Albada, B. van Leer, and W. W. Roberts. A comparative study of computational methods in cosmic gas dynamics. ICASE Report 81-24, 1981.

[8] P. J. Petrie-Repar. Numerical simulation of diaphragm rupture. PhD Thesis, The University of Queensland, Brisbane, Australia, December 1997.

[9] D. F. Potter, R. J. Gollan, T. Eichmann, T. J. McIntyre, R. G. Morgan, and P. A. Jacobs. Simulation of $CO_2$-$N_2$ expansion tunnel flows for the study of radiating shock layers. AIAA Paper 2008-1280, 2008.

[10] B. J. McBride and S. Gordon. Computer program for calculation of complex chemical equilibrium compositions and applications. Part 2: Users manual and program description. Reference Publication 1311, NASA, 1996.

[11] C. R. Wilke. A viscosity equation for gas mixtures. *Journal of Chemical Physics*, 18:517–519, 1950.

[12] S. Gordon and B. J. McBride. Computer program for calculation of complex chemical equilibrium compositions and applications. Part 1: Analysis. Reference Publication 1311, NASA, 1994.

[13] E.S. Oran and J.P. Boris. *Numerical Simulation of Reactive Flow*. Cambridge University Press, New York, USA, 2nd edition, 2001.

[14] E. Fehlberg. Low-order classical Runge-Kutta formulas with stepsize control and their application to some heat transfer problems. Technical Report R-315, NASA, 1969.

[15] D. R. Mott. *New Quasi-Steady-State and Partial-Equilibrium Methods for Integrating Chemically Reacting Systems*. PhD thesis, University of Michigan, 1999.

[16] T.R. Young and J.P. Boris. A numerical technique for solving stiff ordinary differential equations associated with the chemical kinetics of reactive-flow problems. *Journal of Physical Chemistry*, 81(25):2424–2427, 1977.