SANDIA REPORT
SAND2020-11530 - Unlimited Release

Printed October 20, 2020

Sandia National Laboratories

SIERRA/Aero Theory Manual – Version 4.58

Sierra/Aero Development Team

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185
Livermore, California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Road
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods

ABSTRACT

SIERRA/Aero is a compressible fluid dynamics program intended to solve a wide variety compressible fluid flows including transonic and hypersonic problems. This document describes the commands for assembling a fluid model for analysis with this module, henceforth referred to simply as Aero for brevity. Aero is an application developed using the SIERRA Toolkit (STK). The intent of STK is to provide a set of tools for handling common tasks that programmers encounter when developing a code for numerical simulation. For example, components of STK provide field allocation and management, and parallel input/output of field and mesh data. These services also allow the development of coupled mechanics analysis software for a massively parallel computing environment.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE

Einstein notation is used extensively throughout this report to imply summation over repeated indices, primarily for multiple directions in integral equations. Indices are also used to denote chemical species in a gas mixture. When dealing with notation for chemical species, Einstein notation is not implied. When summation over chemical species is required, we will use a summation operator.

ENGLISH CHARACTER SYMBOLS

E total internal energy

H total enthalpy

k kinetic energy

M Molecular Weight

P pressure

q heat conduction

R universal gas constant

t time

T temperature

u velocity

x Cartesian coordinates

GREEK CHARACTER SYMBOLS

κ thermal conductivity

μ viscosity

ϕ limiter

ρ density

σ turbulent stress tensor

τ viscous stress tensor

SUPERSCRIPT CHARACTER SYMBOLS

i indicial notation for species number

n iteration or time step number

r indicial notation for reaction number

$'$ fluctuating quantity with respect to time average

$''$ fluctuating quantity with respect to Favre average

$\tilde{}$ Favre-averaged quantity

$\bar{}$ Reynolds-averaged quantity

SUBSCRIPT CHARACTER SYMBOLS

air property associated with air

T Turbulent modeled quantity

DIMENSIONLESS GROUPS

Pr Prandtl number, the ratio of viscous and thermal diffusivities

Re Reynolds number, the ratio of inertial and viscous forces

1. INTRODUCTION

SIERRA/Aero is a two and three dimensional, node-centered, edge-based finite volume code that approximates the compressible Navier-Stokes equations on unstructured meshes. It is applicable to inviscid and high Reynolds number laminar and turbulent flows. Currently, two classes of turbulence models are provided: Reynolds Averaged Navier-Stokes (RANS) and hybrid methods such as Detached Eddy Simulation (DES). Large Eddy Simulation (LES) models are currently under development. The gas may be modeled either as ideal, or as a non-equilibrium, chemically reacting mixture of ideal gases.

This document describes the mathematical models contained in the code, as well as certain implementation details. First, the governing equations are presented, followed by a description of the spatial discretization. Next, the time discretization is described, and finally the boundary conditions. Throughout the document, SIERRA/ Aero is referred to simply as Aero for brevity.

2. GOVERNING EQUATIONS FOR AN IDEAL GAS

Many flows of engineering interest may be modeled as a calorically perfect gas. (In this work, we use the term ideal gas synonymously with calorically perfect.) In this case, the following assumptions are made:

- the gas is in thermodynamic equilibrium
- the gas is not chemically reacting
- the internal energy and enthalpy are functions only of temperature
- the specific heat at constant pressure and constant volume is constant.

In this chapter, the resulting governing equations are presented. In Section 2.1 we discuss the laminar and inviscid flow cases, and in Section 2.2 we consider turbulent flows.

2.1. LAMINAR EQUATIONS

For an ideal gas, the flow is governed by the compressible Navier-Stokes equations in Cartesian coordinates. We do not derive these equations here. We present them in divergence form using Einstein notation, which implies summation over repeated indices. For a derivation of the Navier-Stokes equations in Einstein notation, see [1]. Conservation of mass, momentum and energy are given by (2.1), (2.2), and (2.3), respectively.

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0 \tag{2.1}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_i u_j + P\delta_{ij}\right) = \frac{\partial \tau_{ij}}{\partial x_j} \tag{2.2}$$

$$\frac{\partial \rho E}{\partial t} + \frac{\partial \rho u_j H}{\partial x_j} = -\frac{\partial q_j}{\partial x_j} + \frac{\partial u_i \tau_{ij}}{\partial x_j} \tag{2.3}$$

The first term on the left side of the above equations indicates the local, instantaneous rate of change of the conserved quantity. The second term on the left is the divergence of the Euler (also called the advective) fluxes. The terms that appear on the right are the divergence of the viscous fluxes. The symbol δ_{ij} indicates the Kronecker delta, which has a value of zero if $i \neq j$ and one otherwise.

The primitive variables are the velocity components, u_i, the pressure, P, and the temperature T. The viscous stress tensor is denoted as τ_{ij}, the heat flux vector as q_i, the total enthalpy as H, the total internal energy as E, and the density as ρ. The equations are closed using the ideal gas law, the stress tensor for a Newtonian fluid, and Fourier's Law for heat conduction, which results in the following relations,

$$P = \frac{\rho R T}{M} \tag{2.4}$$

$$H = h + \frac{1}{2} u_k u_k \tag{2.5}$$

$$E = H - P/\rho \tag{2.6}$$

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \mu \frac{\partial u_k}{\partial x_k} \delta_{ij} \tag{2.7}$$

$$q_i = -\kappa \frac{\partial T}{\partial x_i} \tag{2.8}$$

where μ denotes the dynamic viscosity and the thermal conductivity is denoted as κ. We remark that the inviscid Euler equations are obtained by setting the right hand side of (2.1)-(2.3) to zero.

2.2. TURBULENT EQUATIONS

The Navier-Stokes equations (2.1)-(2.3) are, strictly speaking, valid for laminar and turbulent flows. However, once the equations are discretized, current technology does not provide sufficient computer resources to resolve the length and time scales over which turbulent fluctuations occur. Nor are these resources expected to be available in the foreseeable future. Hence many important phenomena occur at sub-grid scales and must be modeled. This is the motivation for Reynolds averaging the conservation equations. We begin this section by discussing the averaging process in Section 2.2.1. Next, we present the averaged equations and briefly discuss the closure problem in Section 2.2.2. In Sections 2.3 through 2.5 we discuss the specific RANS turbulence models and their DES extensions, respectively.

2.2.1. Reynolds and Favre Averaging

To begin, each variable is decomposed as a time-averaged average quantity plus a fluctuating quantity, e.g.

$$\phi = \bar{\phi} + \phi'' \tag{2.9}$$

where the Reynolds average is defined as:

$$\bar{\phi} = \frac{1}{t_f} \int_0^{t_f} \phi dt \tag{2.10}$$

over some time scale t_f, and the fluctuation is denoted ϕ''. Because some terms in the Navier-Stokes equations appear as products with the fluid density, it is helpful to introduce the Favre average, which is defined as

$$\tilde{\phi} = \frac{\overline{\rho \phi}}{\bar{\rho}} \tag{2.11}$$

Some useful identities associate with Favre Averaging are:

$$\bar{\phi''} \neq 0 \tag{2.12}$$

$$\tilde{\phi''} = \overline{\rho\phi''} = 0 \tag{2.13}$$

$$\overline{\rho\phi\psi} = \bar{\rho}\widetilde{\phi\psi} = \bar{\rho}\tilde{\phi}\tilde{\psi} + \bar{\rho}\widetilde{\phi''\psi''} \tag{2.14}$$

$$\bar{\rho}\widetilde{\phi''\psi''} = \bar{\rho}(\widetilde{\rho\psi} - \tilde{\phi}\tilde{\psi}) \tag{2.15}$$

2.2.2. Turbulent Averaged Equations

After performing the Reynolds averaging of the compressible Navier-Stokes equations (2.1) - (2.3) over some time scale, and performing some considerable algebra, it may be shown that the result may be expressed as

$$\frac{\partial \bar{\rho}}{\partial t} = \frac{\partial \bar{\rho}\bar{u}_j}{\partial x_j} = 0 \tag{2.16}$$

$$\frac{\partial \bar{\rho}\tilde{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\tilde{u}_i\tilde{u}_j + \bar{p}\delta_{ij} - \bar{\tau}_{ij} - \bar{\sigma}_{ij}\right] = 0 \tag{2.17}$$

$$\frac{\partial \bar{\rho}\tilde{E}}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\widetilde{Hu_j} + \bar{q}_j - \overline{u_i\tau_{ij}}\right] = 0 \tag{2.18}$$

In the momentum equations, the averaged stress term is

$$\bar{\tau}_{ij} = 2\bar{\mu}\left(\bar{S}_{ij} - \frac{1}{3}\bar{S}_{kk}\delta_{ij}\right) \quad \text{where} \quad \bar{S}_{ij} = \frac{1}{2}\left(\frac{\partial \tilde{u}_i}{\partial x_j} + \frac{\partial \tilde{u}_j}{\partial x_i}\right) \tag{2.19}$$

The turbulent stress term is

$$\bar{\sigma}_{ij} = -\bar{\rho}\left(\widetilde{u_iu_j} - \tilde{u}_i\tilde{u}_j\right) = -\bar{\rho}\left(\widetilde{u_j''u_i''}\right) \tag{2.20}$$

The energy can be rewritten as:

$$\frac{\partial \bar{\rho}\tilde{E}}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\tilde{H}\tilde{u}_j + \bar{q}_j - \overline{u_i\tau_{ij}} - \bar{\rho}\left(\widetilde{H''u_j''}\right)\right] = 0 \tag{2.21}$$

In the above equations, there are two terms that require closure, the turbulent stress term, $\bar{\sigma}_{ij}$, and the turbulent transport of total enthalpy, $\bar{\rho}(\widetilde{H''u_j''})$. Both terms are modeled using the eddy viscosity hypothesis. Consequently, the turbulent stress tensor is approximated in terms of the strain tensor and an eddy viscosity, and may be written as

$$\bar{\sigma}_{ij} = -\bar{\rho}\left(\widetilde{u_iu_j} - \tilde{u}_i\tilde{u}_j\right) = 2\mu_T\left(\bar{S}_{ij} - \frac{1}{3}\bar{S}_{kk}\delta_{ij}\right) - \frac{2}{3}\bar{\rho}\tilde{k}\delta_{ij} \tag{2.22}$$

Similarly, the turbulent transport of total enthalpy may be written as

$$\bar{\rho}\widetilde{H''u_j''} = \frac{\mu_T C_p}{P_{r_T}}\frac{\partial \tilde{T}}{\partial x_j} - \frac{1}{2}\bar{\rho}\left(\widetilde{u_i u_i u_j} - \widetilde{u_i u_i}\tilde{u}_j\right) \tag{2.23}$$

Next, we describe the specific models for the turbulent viscosity and turbulent kinetic energy that have been implemented in Aero.

2.3. SST TURBULENCE MODEL

SST (Shear Stress Transport) is a variant of a k-ω model. Accordingly, a transport equation is solved for the turbulent kinetic energy $\tilde{k}$ and the specific dissipation rate $\tilde{\omega}$. We do not derive these equations here. The interested reader should consult the excellent text by Pope[2] for a broad discussion of turbulence models, and Menter [3] for the details of the specific model that has been implemented in Aero.. The transport equations for $\tilde{k}$ and $\tilde{\omega}$ can be written as

$$\frac{\partial \bar{\rho}\tilde{k}}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\tilde{k}\tilde{u}_j - \left(\bar{\mu} + \frac{\mu_T}{c_{\tilde{k}}}\right)\frac{\partial \tilde{k}}{\partial x_j}\right] = \bar{\sigma}_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} - \beta^\star \bar{\rho}\tilde{k}\tilde{\omega} \tag{2.24}$$

and

$$\frac{\partial \bar{\rho}\tilde{\omega}}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\tilde{\omega}\tilde{u}_j - \left(\bar{\mu} + \frac{\mu_T}{c_{\tilde{\omega}}}\right)\frac{\partial \tilde{\omega}}{\partial x_j}\right] = \gamma\frac{\tilde{\omega}}{\tilde{k}}\bar{\sigma}_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} - \beta^\star \bar{\rho}\tilde{\omega}^2, \tag{2.25}$$

respectively. The eddy viscosity is defined in terms of these two model quantities

$$\mu_T = \bar{\rho}\frac{\tilde{k}}{\tilde{\omega}} \tag{2.26}$$

We explain the parameters $c_{\tilde{k}}$ and $c_{\tilde{\omega}}$ below in Section 2.3.1.

To summarize, the governing equations for the Reynolds Averaged Navier Stokes (RANS) equations consist of (2.24) and (2.25), together with the RANS equations for conservation of mass, momentum and energy, viz.

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial \bar{\rho}\bar{u}_j}{\partial x_j} = 0 \tag{2.27}$$

$$\frac{\partial \bar{\rho}\tilde{u}_i}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\tilde{u}_i\tilde{u}_j + \bar{p}\delta_{ij} - \bar{\tau}_{ij} - \bar{\sigma}_{ij}\right] = 0 \tag{2.28}$$

$$\frac{\partial \bar{\rho}\tilde{E}}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\widetilde{Hu_j} + \bar{q}_j - \tilde{u}_i\left(\bar{\sigma}_{ij} + \bar{\tau}_{ij}\right) - \bar{T}_j\right] = 0 \tag{2.29}$$

$$\bar{\sigma}_{ij} = 2\mu_T\left(\bar{S}_{ij} - \frac{1}{3}\bar{S}_{kk}\delta_{ij}\right) - \frac{2}{3}\bar{\rho}\tilde{k}\delta_{ij} \tag{2.30}$$

$$\bar{T}_j = \frac{\mu_T C_P}{P_{r_T}}\frac{\partial \tilde{T}}{\partial x_j} + \left(\bar{\mu} + \frac{\mu_T}{c_{\tilde{k}}}\right)\frac{\partial \tilde{k}}{\partial x_j} \tag{2.31}$$

$$\mu_T = \bar{\rho}\frac{\tilde{k}}{\tilde{\omega}} \tag{2.32}$$

We remark that, in the form of the turbulent kinetic energy transport equation presented above as (2.24), the turbulent kinetic energy due to Reynolds stresses is not included in the turbulent transport. This is a modeling choice, and the physical argument for not including this term is as follows: this term describes the generation of heat due to viscous work. Unlike the viscous stresses, the turbulent stresses do not directly produce heat, they only cause a cascade of energy down to the viscous scales where the energy can be converted to heat by the viscous stresses.

To simplify the implementation of the RANS equations, the turbulent kinetic energy transport equation can be subtracted from the energy equation. This manipulation avoids any modifications to the calculation of the internal energy from the conserved total energy variable. Recall that the conservation of energy and turbulent kinetic energy equations may be written as

$$\frac{\partial \bar{\rho}\tilde{E}}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\widetilde{Hu_j} + \bar{q}_j - \tilde{u}_i\left(\bar{\sigma}_{ij} + \bar{\tau}_{ij}\right) - \bar{T}_j\right] = 0 \tag{2.33}$$

$$\frac{\partial \bar{\rho}\tilde{k}}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\tilde{k}\tilde{u}_j - \left(\bar{\mu} + \frac{\mu_T}{c_{\tilde{k}}}\right)\frac{\partial \tilde{k}}{\partial x_j}\right] = \bar{\sigma}_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} - \beta^\star \bar{\rho}\tilde{k}\tilde{\omega} \tag{2.34}$$

By definition,

$$\bar{\rho}\tilde{E} = \bar{\rho}\left(\tilde{e} + \frac{\tilde{u}_i\tilde{u}_i}{2} + \tilde{k}\right)$$

Therefore (2.33) may be written as

$$\frac{\partial\left[\bar{\rho}\left(\tilde{e} + \frac{\tilde{u}_i\tilde{u}_i}{2} + \tilde{k}\right)\right]}{\partial t} + \frac{\partial}{\partial x_j}\left[\bar{\rho}\widetilde{u_j}\left(\tilde{e} + \frac{\tilde{u}_i\tilde{u}_i}{2} + \frac{\bar{P}}{\bar{\rho}} + \tilde{k}\right) + \bar{q}_j - \tilde{u}_i\left(\bar{\sigma}_{ij} + \bar{\tau}_{ij}\right) - \bar{T}_j\right] = 0 \tag{2.35}$$

Now, (2.34) may be subtracted from (2.35) to obtain

$$\begin{aligned}\frac{\partial\left[\bar{\rho}\left(\tilde{e} + \frac{\tilde{u}_i\tilde{u}_i}{2}\right)\right]}{\partial t} + \\ \frac{\partial}{\partial x_j}\left[\bar{\rho}\widetilde{u_j}\left(\tilde{e} + \frac{\tilde{u}_i\tilde{u}_i}{2} + \frac{\bar{P}}{\bar{\rho}}\right) + \bar{q}_j - \tilde{u}_i\left(\bar{\sigma}_{ij} + \bar{\tau}_{ij}\right) - \bar{T}_j + \left(\bar{\mu} - \frac{\mu_T}{c_{\tilde{k}}}\right)\frac{\partial \tilde{k}}{\partial x_j}\right] = \\ -\bar{\sigma}_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} + \beta^\star\bar{\rho}\tilde{k}\tilde{\omega}\end{aligned} \tag{2.36}$$

Upon substituting the expression for T_j given in (2.31), and the heat flux given in (2.8), the energy conservation equation may now be written as

$$\begin{aligned}\frac{\partial\left[\bar{\rho}\left(\tilde{e} + \frac{\tilde{u}_i\tilde{u}_i}{2}\right)\right]}{\partial t} + \\ \frac{\partial}{\partial x_j}\left[\bar{\rho}\widetilde{u_j}\left(\tilde{e} + \frac{\tilde{u}_i\tilde{u}_i}{2} + \frac{\bar{P}}{\bar{\rho}}\right) - \tilde{u}_i\left(\bar{\sigma}_{ij} + \bar{\tau}_{ij}\right) - \left(\kappa + \frac{\mu_T C_P}{P_{r_T}}\right)\frac{\partial \tilde{T}}{\partial x_j}\right] = \\ -\bar{\sigma}_{ij}\frac{\partial \tilde{u}_i}{\partial x_j} + \beta^\star\bar{\rho}\tilde{k}\tilde{\omega}\end{aligned} \tag{2.37}$$

Finally, if we redefine E to be the total specific energy minus the turbulent kinetic energy, and H to be the corresponding total specific enthalpy without the turbulent kinetic energy, the RANS equations may be expressed as

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0 \tag{2.38}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_i u_j + p\delta_{ij}\right) = \frac{\partial}{\partial x_j}\left[\mu_{\text{eff}}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right) - \frac{2}{3}\rho k \delta_{ij}\right] \tag{2.39}$$

$$\frac{\partial \rho E}{\partial t} + \frac{\partial \rho H}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\tilde{u}_i\left(\sigma_{ij} + \tau_{ij}\right) + \kappa_{\text{eff}}\frac{\partial T}{\partial x_j}\right] - \sigma_{ij}\frac{\partial u_i}{\partial x_j} + \beta^\star \rho k \omega \tag{2.40}$$

$$\frac{\partial \rho k}{\partial t} + \frac{\partial \rho k u_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{c_k}\right)\frac{\partial k}{\partial x_j}\right] + \sigma_{ij}\frac{\partial u_i}{\partial x_j} - \beta^\star \rho k \omega \tag{2.41}$$

$$\frac{\partial \rho \omega}{\partial t} + \frac{\partial \rho \omega u_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{c_\omega}\right)\frac{\partial \omega}{\partial x_j}\right] + \gamma\frac{\omega}{k}\sigma_{ij}\frac{\partial u_i}{\partial x_j} - \beta^\star \rho \omega^2 \tag{2.42}$$

with

$$E = e_i + \frac{u_k u_k}{2} \tag{2.43}$$

$$H = E + \frac{P}{\rho} \tag{2.44}$$

$$\mu_T = \frac{\rho k}{\omega} \tag{2.45}$$

$$\mu_{\text{eff}} = \mu + \mu_T \tag{2.46}$$

$$\kappa_{\text{eff}} = \frac{\mu C_P}{P_r} + \frac{\mu_T C_P}{P_{r_T}} \tag{2.47}$$

where, for the sake of brevity in later developments, we have dropped the $\bar{()}$ and $\tilde{()}$. It should be clear from this form of the k-ω model that the implementation in a laminar code is simplified because it is not necessary to subtract the turbulent kinetic energy term from the degree of freedom ρE everywhere the internal energy is needed. Furthermore, the flux Jacobian matrices are also simplified because the advective fluxes do not involve the turbulent kinetic energy. However, the turbulent kinetic energy affects the total energy through a source term, as indicated in (2.40).

2.3.1. Variants of the k-ω Model

There are several variants of the k-ω model, which are defined, for example, by various definitions of the parameters such as $\beta^\star$, γ, and μ_T. In this section, we show how four of these variants, namely the 1988 model of Wilcox [4], the 2006 model of Wilcox [5], the baseline (BSL) model due to Menter [3], and the shear stress transport (SST) model due to Menter [3] are related. In order to discriminate among these models it is helpful to rewrite the turbulent transport equations as

$$\frac{\partial \rho k}{\partial t} + \frac{\partial \rho k u_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \sigma_k \mu_T\right)\frac{\partial k}{\partial x_j}\right] + P_k - D_k + S_k \tag{2.48}$$

$$\frac{\partial \rho \omega}{\partial t} + \frac{\partial \rho \omega u_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \sigma_\omega \mu_T\right)\frac{\partial \omega}{\partial x_j}\right] + P_\omega - D_\omega + S_\omega \tag{2.49}$$

Here, P_k and P_ω denote the production terms for turbulent kinetic energy and dissipation rate, D_k and D_ω denote the dissipation, or destruction terms, and S_k and S_ω denote the cross production source terms. The parameters that define each of the four models, Wilcox88, Wilcox06, BSL and SST are given in Table 2.3-1.

2.3.2. Detached eddy simulation (DES)

Detached Eddy Simulation is a hybrid RANS-LES approach that can be used for flows with massive separation[6]. RANS models cannot capture the large scale eddies in the separated region accurately but are efficient and accurate for thin shear layers. LES is expensive in attached boundary layers and thin shear layers but accurately captures the large scale motion of separated flows. Detached Eddy Simulation combines RANS and LES by using an LES subgrid based model in the parts of the domain where the grid resolution is fine enough for LES. Elsewhere a RANS model is used. The switch between the two modes is determined by comparing an integral turbulent length scale and the local grid spacing. The approach is nonzonal and the RANS equations are still solved in all regions. For the k-ω model, this is done by modifying the dissipative term in the turbulent kinetic energy equation. The dissipation term is modified from

$$D_k = \beta^\star \rho \omega k \tag{2.50}$$

to

$$D_k = \frac{\rho k^{3/2}}{l} \tag{2.51}$$

where we have introduced the length scale

$$l = \min(l_{k-\omega}, C_{\text{DES}}\Delta), \qquad l_{k-\omega} = \frac{k^{1/2}}{\beta^\star \omega} \tag{2.52}$$

C_{DES} is a constant whose default value is 0.65 and Δ is a grid spacing measure. This measure is defined at each node as the maximum value over all edges of:

$$\Delta = \max(\delta_x, \delta_y, \delta_z), \tag{2.53}$$

where, e.g. δ_x is the absolute value of the change in the x coordinate across an edge. Equation (2.53) is designed so that the computation is limited by the coarsest spacing for each node.

2.3.3. Linearization of implicit terms

The turbulent transport equations require careful treatment in order to obtain a stable and robust solution algorithm. The basic method that we follow is that the time derivative, convective flux terms, and diffusion terms are treated in a way that is consistent with the rest of the Navier-Stokes equations. The source/sink terms are treated according to the following rules:

- sensitivities are computed so that only diagonal matrix entries are generated
- only sensitivities to turbulent dissipation terms are generated; sensitivities to production terms are ignored.

	Wilcox88	Wilcox06	BSL	SST
μ_T	$\frac{\gamma^\star \rho k}{\omega}$	$\frac{\gamma^\star \rho k}{\omega}$	$\frac{\gamma^\star \rho k}{\omega}$	$\frac{a_1 \rho k}{\max(a_1\omega, \Omega F_2)}$
σ_k	$\frac{1}{2}$	$\frac{3}{5}$	$F_1\sigma_{k1} + (1-F_1)\,\sigma_{k2}$	$F_1\sigma_{k1} + (1-F_1)\,\sigma_{k2}$
P_k	$\sigma_{ij}\frac{\partial u_i}{\partial x_j}$	$\sigma_{ij}\frac{\partial u_i}{\partial x_j}$	$\sigma_{ij}\frac{\partial u_i}{\partial x_j}$	$\sigma_{ij}\frac{\partial u_i}{\partial x_j}$
D_k	$\beta^\star \rho\omega k$	$\beta^\star \rho\omega k$	$\beta^\star \rho\omega k$	$\beta^\star \rho\omega k$
S_k	0	0	0	0
$\gamma^\star$	1	1	1	1
σ_ω	$\frac{1}{2}$	$\frac{1}{2}$	$F_1\sigma_{\omega1} + (1-F_1)\,\sigma_{\omega2}$	$F_1\sigma_{\omega1} + (1-F_1)\,\sigma_{\omega2}$
P_ω	$\frac{\gamma\omega}{k}P_k$	$\frac{\gamma\omega}{k}P_k$	$\frac{\gamma\omega}{k}P_k$	$\frac{\gamma\rho}{\mu_T}P_k$
D_ω	$\beta\rho\omega^2$	$\beta\rho\omega^2$	$\beta\rho\omega^2$	$\beta\rho\omega^2$
S_ω	0	$\sigma_d\frac{\rho}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}$	$\sigma_d\frac{\rho}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}$	$\sigma_d\frac{\rho}{\omega}\frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j}$
$\beta^\star$	0.09	0.09	0.09	0.09
γ	$\frac{5}{9}$	$\frac{13}{25}$	$\frac{13}{25}$	$F_1\gamma_1 + (1-F_1)\,\gamma_2$
β	$\frac{3}{40}$	$\beta_0 f_\beta$	$F_1\beta_1 + (1-F_1)\,\beta_2$	$F_1\beta_1 + (1-F_1)\,\beta_2$
$\omega^\star$	-	$\max\left(\omega, C_{\lim}\sqrt{\frac{2S_{ij}S_{ij}}{\beta^\star}}\right)$	-	-
$C_{\lim}$	-	$\frac{7}{8}$	-	-
β_0	-	0.0708	-	-
f_β	-	$\frac{1+85\chi_\omega}{1+100\chi_\omega}$	-	-
χ_ω	-	$\left\lvert\frac{\Omega_{ij}\Omega_{jk}\hat{S_{ki}}}{(\beta^\star\omega)^3}\right\rvert$	-	-
$\hat{S_{ki}}$	-	$\frac{1}{2}\left(\frac{\partial u_k}{\partial x_i} + \frac{\partial u_i}{\partial x_k} - \frac{\partial u_m}{\partial x_m}\delta_{ki}\right)$	-	-
σ_d	-	$\begin{cases} 0 & : \ \frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} \le 0 \\ \frac{1}{8} & : \ \frac{\partial k}{\partial x_j}\frac{\partial \omega}{\partial x_j} > 0 \end{cases}$	$2(1-F_1)\sigma_{\omega2}$	$2(1-F_1)\sigma_{\omega2}$
σ_{k1}	-	-	$\frac{1}{2}$	0.85
σ_{k2}	-	-	1	1
$\sigma_{\omega1}$	-	-	$\frac{1}{2}$	$\frac{1}{2}$
$\sigma_{\omega2}$	-	-	0.856	0.856
β_1	-	-	0.075	0.075
β_2	-	-	0.0828	0.0828
γ_1	-	-	$\frac{\beta_1}{\beta^\star} - \frac{0.5531\sigma_{\omega1}\kappa^2}{\sqrt{\beta^\star}}$	$\frac{\beta_1}{\beta^\star} - \frac{\sigma_{\omega1}\kappa^2}{\sqrt{\beta^\star}}$
γ_2	-	-	$\frac{\beta_2}{\beta^\star} - \frac{0.44035\sigma_{\omega2}\kappa^2}{\sqrt{\beta^\star}}$	$\frac{\beta_2}{\beta^\star} - \frac{\sigma_{\omega2}\kappa^2}{\sqrt{\beta^\star}}$
F_1	-	-	$\tanh\left(\arg_1^4\right)$	$\tanh\left(\arg_1^4\right)$
$\arg_1$	-	-	$\min\left(\max\left(\frac{\sqrt{k}}{\beta^\star\omega d}, \frac{500\nu}{d^2\omega}\right), \frac{4\rho\sigma_{\omega2}k}{CD_{k\omega}d^2}\right)$	$\min\left(\max\left(\frac{\sqrt{k}}{\beta^\star\omega d}, \frac{500\nu}{d^2\omega}\right), \frac{4\rho\sigma_{\omega2}k}{CD_{k\omega}d^2}\right)$
$CD_{K\omega}$	-	-	$\max\left(2\sigma_{\omega2}\frac{\rho}{\omega}\frac{\partial k}{\partial x_k}\frac{\partial \omega}{\partial x_k}, 10^{-20}\right)$	$\max\left(2\sigma_{\omega2}\frac{\rho}{\omega}\frac{\partial k}{\partial x_k}\frac{\partial \omega}{\partial x_k}, 10^{-20}\right)$
F_2	-	-	-	$\tanh\left(\arg_2^2\right)$
$\arg_2$	-	-		$\max\left(\frac{2\sqrt{k}}{\beta^\star\omega d}, \frac{500\nu}{d^2\omega}\right)$
a_1	-	-	-	0.31

Table 2.3-1.. Definition of parameters for Wilcox88, Wilcox06, BSL, and SST variants of the k-ω models. Ω_{ij} denotes the vorticity tensor, Ω its magnitude, and d is the minimum distance to the wall.

The source terms for the turbulent kinetic energy equation given in (2.41) may be written as

$$\mathcal{S}_k = \sigma_{ij}\frac{\partial u_i}{\partial x_j} - \beta^\star \rho k \omega$$

Then we only compute the sensitivity according to

$$\frac{\partial \mathcal{S}_k}{\partial \rho k} = -\beta^\star \rho \omega$$

and ignore the sensitivities to ρ, ρu_j, ρE and $\rho\omega$.

Similarly, the source terms given in the specific dissipation rate equation (2.42) may be written as

$$\mathcal{S}_\omega = \gamma\frac{\omega}{k}\sigma_{ij}\frac{\partial u_i}{\partial x_j} - \beta^\star \rho \omega^2$$

We compute the sensitivity to $\rho\omega$ according to

$$\frac{\partial \mathcal{S}_\omega}{\partial \rho \omega} = -2\beta^\star \omega$$

2.4. K-ϵ MODEL

If instead of solving a transport equation for the specific dissipation rate ω, a transport equation for the dissipation rate

$$\epsilon = k\omega \tag{2.54}$$

is solved, the resulting turbulence model belongs to the class of models known as k-ϵ. The k-ϵ model implemented in Aero is described in So et al [7] and Brinkman et al [8]. This model is described by the following equations

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0 \tag{2.55}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_i u_j + p\delta_{ij}\right) = \frac{\partial}{\partial x_j}\left[\mu_{\text{eff}}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right) - \frac{2}{3}\rho k \delta_{ij}\right] \tag{2.56}$$

$$\frac{\partial \rho E}{\partial t} + \frac{\partial \rho H}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\tilde{u}_i \tau_{ij} + \kappa_{\text{eff}}\frac{\partial T}{\partial x_j}\right] - \sigma_{ij}\frac{\partial u_i}{\partial x_j} + \beta^\star \rho k \omega \tag{2.57}$$

$$\frac{\partial \rho k}{\partial t} + \frac{\partial \rho k u_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + P_k - \rho\epsilon \tag{2.58}$$

$$\frac{\partial \rho \epsilon}{\partial t} + \frac{\partial \rho \epsilon u_j}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_T}{\sigma_\epsilon}\right)\frac{\partial \epsilon}{\partial x_j}\right] + \frac{\epsilon}{k}\left[\left(C_1 f_1 P_k - C_2 f_2 \rho\epsilon\right) + S_\epsilon\right], \tag{2.59}$$

where

$$P_k = \sigma_{ij}\frac{\partial u_i}{\partial x_j}, \tag{2.60}$$

$$S_\epsilon = \frac{14}{9} C_2 \mu \frac{\partial k^{\frac{1}{2}}}{\partial x_j} \frac{\partial k^{\frac{1}{2}}}{\partial x_j}, \tag{2.61}$$

σ_k, σ_ϵ, C_1 and C_2 are modeling constants, the turbulent viscosity is defined as

$$\mu_T = C_\mu f_\mu \rho \frac{k^2}{\epsilon} \tag{2.62}$$

and f_1, f_2, and f_μ are empirical modeling functions designed to account for low Reynolds number effects that occur near the walls. These functions approach unity as the distance from the wall is increased, and may be written as

$$\begin{aligned} f_1 &= 1 - \exp\left[-\left(\frac{\mathrm{Re}_t}{40}\right)^2\right] + \frac{0.20}{\cosh\left[\log\left(\frac{\mathrm{Re}_k}{100}\right)\right]} \\ f_2 &= 1 - \frac{2}{9}\exp\left[-\left(\mathrm{Re}_t\right)^2\right] \\ f_\mu &= \left(1 + 4\mathrm{Re}_t^{-3/4}\right)\tanh\left(\frac{\mathrm{Re}_k}{125}\right), \end{aligned}$$

where

$$\begin{aligned} \mathrm{Re}_t &= \frac{\rho k^2}{\mu\epsilon} \\ \mathrm{Re}_k &= \frac{\rho d\sqrt{k}}{\mu} \end{aligned}$$

and d is the nearest distance to the wall. The model is completed with the following constants:

c_μ	σ_k	σ_ϵ	c_1	c_2
0.09	1	1.3	1.43	1.92

2.4.1. Detached eddy simulation (DES)

The hybrid RANS-LES model for the k-ϵ model differs from that of the SST and Spalart-Allmaras models in the sense that the method is applied to the turbulent viscosity instead of through the source terms for the turbulent transport equations. We begin the explanation of this approach by exploiting the observation that the turbulent kinetic energy statistics at every point in the flow field satisfies a turbulent spectrum. If each point in the flow field is treated as a realization of these statistics, then a model can be formulated which provides a method to estimate total, resolved and unresolved portions of the kinetic energy in any simulation. This information can be used to formulate a hybrid RANS-LES model. The starting point for this approach is a turbulent kinetic energy spectrum. Here, the Karman-Pao spectrum is used. This form is parameterized by the energy-containing wave number, k_e, the turbulent kinetic energy dissipation rate, ϵ, and the Kolmogorov scale, η, and can be written as

$$E(k) = C_e \epsilon^{-2/3} \left(\frac{k}{k_e}\right)^4 \left[1 + \left(\frac{k}{k_e}\right)^2\right]^{-17/6} \exp\left(-\frac{3}{2}\alpha\left(k\eta\right)^{4/3}\right) \tag{2.63}$$

where k is the wave number, $\alpha = 1.5$, and the constant $C_\epsilon = 1.67$, which calibrates the dissipation spectrum, $D(k) = 2\nu k^2 E(k)$ to the turbulence dissipation rate. (As a matter of notation, throughout this section, k refers to a wave number, not to the turbulent kinetic energy, unless indicated otherwise.) This spectrum can be nondimensionalized using the Kolmogorov length scale $\eta = \left(\frac{\nu}{\epsilon}\right)^{1/4}$. The nondimensional spectrum is defined as

$$\hat{E}(\hat{k}) = E(k)/\left(\nu^5 \epsilon\right)^{1/4}$$

which may be written as

$$\hat{E}(\hat{k}) = C_e \hat{k}_e^{-5/3} \left(\frac{\hat{k}}{\hat{k}_e}\right)^4 \left[1 + \left(\frac{\hat{k}}{\hat{k}_e}\right)^2\right]^{-17/6} \exp\left(-\frac{3}{2}\alpha \hat{k}^{4/3}\right), \tag{2.64}$$

where $\hat{k} = k\eta$, and ν is the kinematic viscosity. The spectrum and various Taylor micro-scale Reynolds numbers is shown in Figure 2.4-1, along with the location of the energy-containing wave number for each Reynold number.

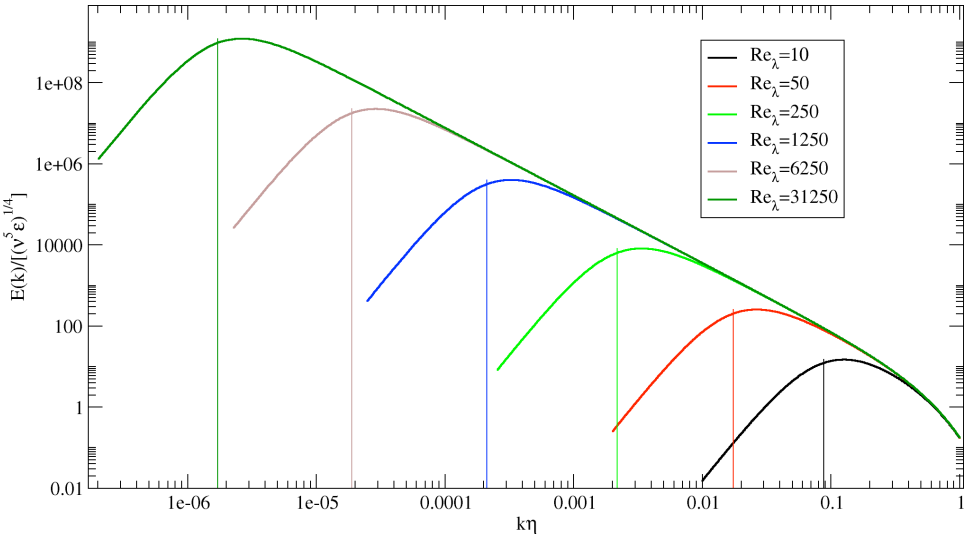

Figure 2.4-1.. The non-dimensional Karman-Pao spectrum at several Taylor micro-scale Reynolds numbers.

The turbulent kinetic energy, K^{RANS} (denoting the total turbulent kinetic energy or the RANS turbulent kinetic energy) is formally related to this energy spectrum as

$$K^{\text{RANS}} = \int_0^\infty E(k)dk = \int_{k_{\min}}^{k_\eta} E(k)dk$$

where k_η represents the Kolmogorov wave number and $k_{\min}$ is the smallest wave number in the flow field, typically taken as $0.1k_e$. The sub-grid kinetic energy is given by

$$K^{\text{SGS}} = \int_{k_\Delta}^{k_\eta} E(k)dk \tag{2.65}$$

2.4.1.1. Original model based on the one equation K^{SGS} model

In the original approach [**?**], the sub-grid kinetic energy K^{SGS} is obtained at every point in the flow field from the solution of a transport equation for K^{SGS} using a one equation sub-grid scale model. Then (2.65) can be solved to determine k_e, the energy containing wave number at that location. However, since k_η is a function of the total dissipation rate, ϵ, which is not known, (2.65) must be solved iteratively. Once k_e is known, the eddy viscosity is computed based on the sub-grid kinetic energy and the dissipation rate from the unresolved portion of the spectrum using

$$\nu_{\text{T,Hyb}} = f_\mu C_\mu \frac{(K^{\text{SGS}})^2}{\epsilon^{\text{SGS}}} \tag{2.66}$$

$$K^{\text{SGS}} = \int_{k_\Delta}^{k_\eta} E(k)dk \tag{2.67}$$

$$\epsilon^{\text{SGS}} = \int_{k_\Delta}^{k_\eta} 2\nu k^2 E(k)dk \tag{2.68}$$

The sub-grid quantities can be obtained by analytically integrating the known spectrum given by (2.63) from the smallest scales to the local mesh resolution scale. Thus, the model not only ensures that the local total range of scales is accounted for, but also that the eddy viscosity used in the momentum equations is consistent with the local mesh resolution and the range of scales. The underlying steps in the above procedure can be summarized as follows

1. From the local values of total turbulent kinetic energy K and the local mesh size Δ, compute the energy-containing wave number $\hat{k}_e$ by iteratively solving (2.65).
2. Given $\hat{k}_e$, analytically integrate the spectrum to compute the sub-grid (unresolved) turbulent kinetic energy K^{SGS} and sub-grid dissipation rate ϵ^{SGS}.
3. Compute the eddy viscosity, which is used in the momentum equation from (2.66).

2.4.2. Linearization of implicit terms

The source terms for the turbulent kinetic energy equation given in (2.41) may be written as

$$\mathcal{S}_k = P_k - \rho\epsilon \tag{2.69}$$

Since P_k is a production term, we ignore its sensitivities. Now there is a difficulty, because this expression for $\mathcal{S}_k$ does not depend directly on ρk, and there will be no contribution to the diagonal block of the Jacobian matrix. To circumvent this difficulty, use (2.62) to express ϵ in terms of k

$$\epsilon = C_\mu f_\mu \rho \frac{k^2}{\mu_T} \tag{2.70}$$

and substitute this into (2.69) to obtain

$$\mathcal{S}_k = P_k - \frac{C_\mu f_\mu}{\mu_T}(\rho k)^2 \tag{2.71}$$

Now, differentiate (2.71) with respect to ρk to obtain

$$\frac{\partial \mathcal{S}_k}{\partial \rho k} = -\frac{2C_\mu f_\mu}{\mu_T}\rho k$$

Finally, use (2.62) to remove the explicit dependence of this result on the turbulent viscosity. Hence, we obtain

$$\frac{\partial \mathcal{S}_k}{\partial \rho k} = -\frac{2\epsilon}{k}$$

The source terms for the dissipation rate equation may be written as

$$\mathcal{S}_\epsilon = \frac{\epsilon}{k}C_1 f_1 P_k - C_2 f_2 \rho \frac{\epsilon^2}{k} + S_\epsilon$$

Following the approach described in Section 2.3.3, we ignore sensitivities to production terms and the gradient magnitude S_ϵ to obtain

$$\frac{\partial \mathcal{S}_\epsilon}{\partial \rho \epsilon} = -\frac{2C_2 f_2 \epsilon}{k}$$

2.5. SPALART-ALLMARAS TURBULENCE MODEL

The Spalart-Allmaras class of turbulence models adds a single transport equation to the Reynolds averaged Navier-Stokes equation. The complete set of conservation equations for this model may be written as

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_j}{\partial x_j} = 0 \tag{2.72}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho u_i u_j + p\delta_{ij}\right) = \frac{\partial}{\partial x_j}\left[\mu_{\text{eff}}\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}\right)\right] \tag{2.73}$$

$$\frac{\partial \rho E}{\partial t} + \frac{\partial \rho H}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\tilde{u}_i \tau_{ij} + \kappa_{\text{eff}}\frac{\partial T}{\partial x_j}\right] - \sigma_{ij}\frac{\partial u_i}{\partial x_j} \tag{2.74}$$

$$\frac{\partial \rho \hat{\nu}}{\partial t} + \frac{\partial \rho \hat{\nu} u_j}{\partial x_j} = \rho c_{b1}\hat{S}\hat{\nu} - \rho c_{w1} f_w \left(\frac{\hat{\nu}}{d}\right)^2 + \frac{\rho c_{b2}}{\sigma}\frac{\partial \hat{\nu}}{\partial x_j}\frac{\partial \hat{\nu}}{\partial x_j} + \frac{1}{\sigma}\frac{\partial}{\partial x_j}\left[\left(\mu + \rho\hat{\nu}\right)\frac{\partial \hat{\nu}}{\partial x_j}\right] \tag{2.75}$$

where $\hat{\nu}$ denotes the so-called 'working variable', c_{b1}, c_{w1}, c_{b2} and σ are model constants, d is the nearest distance to the wall. The turbulent viscosity is defined as

$$\mu_T = \rho\hat{\nu} f_{v1}, \tag{2.76}$$

where

$$\begin{aligned} f_{v1} &= \frac{\chi^3}{\chi^3 + c_{v1}^3} \\ \chi &= \frac{\rho\hat{\nu}}{\mu} \\ \hat{S} &= \Omega + \frac{\hat{\nu}}{\kappa^2 d^2} f_{v2}, \end{aligned}$$

Ω is the vorticity magnitude, and κ is another model constant. The model also contains the definitions

$$\begin{aligned} f_{v2} &= 1 - \frac{\chi}{1 + \chi f_{v1}} \\ f_w &= \left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6} \right)^{\frac{1}{6}} \\ g &= r + c_{w2} \left(r^6 - r \right) \\ r &= \min \left(\frac{\hat{\nu}}{\hat{S} \kappa^2 d^2}, 10 \right) \end{aligned}$$

and the following table provides the values for the model constants.

c_{b1}	c_{w1}	c_{b2}	σ	c_{v1}	κ	c_{w2}	c_{w3}
0.1355	$\frac{c_{b1}}{\kappa^2} + \frac{1+c_{b2}}{\sigma}$	0.622	$\frac{2}{3}$	7.1	0.41	0.3	2

2.5.1. Detached eddy simulation (DES)

The approach for implementing detached eddy simulation in the Spalart-Allmaras model is to replace the nearest distance to the wall, d, with $\tilde{d}$, where

$$\tilde{d} = \min \left(d, C_{\text{DES}} \Delta \right), \tag{2.77}$$

where Δ is given by (2.53). This simple modification makes the turbulence model behave like Large Eddy Simulation away from the walls, and RANS near the walls.

2.5.2. Linearization of implicit terms

The source term for the turbulent transport equation (2.75) may be written as

$$\mathcal{S} = \rho c_{b1} \hat{S} \hat{\nu} - \rho c_{w1} f_w \left(\frac{\hat{\nu}}{d} \right)^2 + \frac{\rho c_{b2}}{\sigma} \frac{\partial \hat{\nu}}{\partial x_j} \frac{\partial \hat{\nu}}{\partial x_j}$$

We ignore the sensitivities of the first and last terms, since they are production. Hence we obtain

$$\frac{\partial \mathcal{S}}{\partial \rho \hat{\nu}} = -\frac{2 c_{w1} f_w}{d^2} \hat{\nu}$$

3. GOVERNING EQUATIONS FOR A CHEMICALLY REACTING GAS

For flows at Mach numbers higher than about Mach 8, the ideal gas model produces temperatures that are unreasonably high. In this regime, the ideal gas approximation breaks down, and we model the gas as a chemically reacting mixture. The conservation of mass can be expressed as a series of equations for the number of species considered in the gas mixture.

$$\frac{\partial \rho_s}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho_s u_i\right) + \frac{\partial}{\partial x_i}\left(\rho \mathcal{D}_s \frac{\partial y_s}{\partial x_i}\right) = \dot{\omega}_s \tag{3.1}$$

In (3.1), ρ_s is the density of species s, $\mathcal{D}_s$ is the species diffusion coefficient (discussed in section **??**), y_s is the mass fraction (ρ_s/ρ) of species s, and $\dot{\omega}_s$ is the rate of production of species s due to chemical reactions.

The energy equations for a fluid in thermal non-equilibrium may be expressed for each of the possible energy modes of a molecule. Hence, energy equations may exist for translational, rotational, vibrational, and electronic states, which each governed by its own separate temperature. A two-temperature model, however, is a common approach for describing thermal nonequilibrium of re-entry aerodynamics with one temperature modeling the translational and rotational energy states and another temperature modeling the vibrational and electronic energy of the molecule. The translational and rotational energy equation is thus expressed as

$$\frac{\partial \rho E}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho u_i \left(E + p/\rho\right)\right) - \frac{\partial}{\partial x_i}\left(\tau_{ij} u_j\right) + \frac{\partial}{\partial x_i}\left(q_i^{t,r} + q_i^v\right) + \frac{\partial}{\partial x_i}\left(\rho \sum_{s=1}^{\text{NS}} h_s \mathcal{D}_s \frac{\partial y_s}{\partial x_i}\right) = 0 \tag{3.2}$$

The vibrational equation is expressed as

$$\frac{\partial \rho E^v}{\partial t} + \frac{\partial}{\partial x_i}\left(\rho E^v u_i\right) + \frac{\partial}{\partial x_i}\left(q_i^v\right) + \frac{\partial}{\partial x_i}\left(\rho \sum_{s=1}^{\text{NS}} e_s^v \mathcal{D}_s \frac{\partial y_s}{\partial x_i}\right) = \dot{\omega}^v \tag{3.3}$$

where E^v is the vibrational total energy and e_s^v is the vibrational total energy per unit mass of species s. The last term on the left side of (3.2) and (3.3) represents an energy flux that arises due to the transport of enthalpy that occurs when one species diffuses into another.

3.1. EQUATIONS OF STATE

The total energy of the gas is defined by

$$\rho E = \sum_{s=1}^{\text{NS}} \rho_s C_{v_s}^{t,r} T + \rho E^v + \sum_{s=1}^{\text{NS}} \rho_s h_s^0 + \frac{1}{2}\rho u_i u_i \tag{3.4}$$

where $C_{v_s}^{t,r}$ is the combined translational and rotational specific heat at constant volume. The specific heats are given by

$$\begin{array}{ll} C_{v_s}^{t,r} = C_{v_s}^t + C_{v_s}^r & \text{(monatomic and polyatomic species)} \\ C_{v_s}^{t,r} = C_{v_s}^t & \text{(polyatomic species)} \end{array} \tag{3.5}$$

The individual translational and rotational specific heats are

$$\begin{array}{ll} C_{v_s}^t = \frac{3}{2}\frac{R^{univ}}{M_s} & \text{(monatomic and polyatomic species)} \\ C_{v_s}^r = \frac{R^{univ}}{M_s} & \text{(polyatomic species)} \end{array} \tag{3.6}$$

where R^{univ} is the universal gas constant and M_s is the species molecular weight.

The total vibrational energy ρE^v appearing in (3.4) is computed by the vibration energy equation, (3.3), and is a function of the vibrational temperature T^v according to the equation

$$\rho E^v\left(T^v\right) = \sum_{s=1}^{\text{NS}} \rho_s e_s^v\left(T^v\right) \tag{3.7}$$

Here, the vibrational energy per unit mass can be expressed as

$$\begin{array}{ll} e_s^v = \frac{R^{univ}}{m_s}\frac{\theta_s^v}{\exp(\theta_s^v/T^v)-1} & \text{(polyatomic species)} \\ e_s^v = 0 & \text{(monatomic species)} \end{array} \tag{3.8}$$

The thermodynamic pressure of the gas is computed using a perfect gas law and Dalton's law of partial pressures

$$p = \sum_{s=1}^{\text{NS}} p_s \tag{3.9}$$

where the partial pressure for species s is

$$p_s = \rho_s \frac{R^{univ}}{M_s} T \tag{3.10}$$

and T is the is the translational-rotational temperature as computed from (3.4).

The species enthalpy per unit mass h_s appearing in (3.2) is computed according to

$$h_s = C_{v_s} T + \frac{p_s}{\rho_s} + e_s^v + h_s^0 \tag{3.11}$$

3.2. DIFFUSION TERMS

Recall that the viscous stress tensor that appears in the conservation of momentum equation, (2.2), may be written as

$$\tau_{ij} = \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right) - \frac{2}{3} \mu \left(\frac{\partial u_i}{\partial x_j} \right) \delta_{ij} \tag{3.12}$$

This equation requires that mixture viscosity μ be computed from individual species viscosities.

The translational-rotational heat flux vector in (3.2) is expressed as

$$q_i^{t,r} = - \left(\kappa^t + \kappa^r \right) \frac{\partial T}{\partial x_i} \tag{3.13}$$

and the vibrational heat flux vector appearing in equations 3.2 and 3.3 is

$$q_i^v = -\kappa^v \frac{\partial T^v}{\partial x_i} \tag{3.14}$$

These constitutive relations require the calculation of κ^t, κ^r, κ^v, the translational, rotational, and vibrational thermal conductivities, respectively, of the mixture. These mixture transport properties for viscosity and thermal conductivity can be computed in a number of ways. One popular approach is to use Blotter curve fits for computing the species viscosities μ_s according to the relation

$$\mu_s = 0.1 \, exp \left((A_s \ln T + B_s) \ln T + C_s \right) \tag{3.15}$$

with the constants A_s, B_s, C_s having been determined by Blottner [] for a number of species relevant to high-speed reacting flows. The thermal conductivities for the various energy modes can be computed from an Eucken relation [] in conjunction with the Blottner species viscosities (3.15) and the species specific heats (3.5) according to

$$\begin{aligned} \kappa_s^t &= \tfrac{5}{2} \mu_s C_{v_s}^t \\ \kappa_s^r &= \mu_s C_{v_s}^r \\ \kappa_s^v &= \mu_s C_{v_s}^v \end{aligned} \tag{3.16}$$

The translational and rotational specific heats $C_{v_s}^t$ and $C_{v_s}^r$ were previously defined in (3.6). The vibrational specific heat is then computed according to the equation

$$C_{v_s}^v = \frac{\partial e_s^v}{\partial T^v} \tag{3.17}$$

The mixture viscosity and thermal conductivities are then computed using Wilke's semi-empirical mixing rule

$$\begin{aligned} \mu &= \sum_{s=1}^{NS} \frac{X_s \mu_s}{\phi_s} \\ \kappa &= \sum_{s=1}^{NS} \frac{X_s \kappa_s}{\phi_s} \end{aligned} \tag{3.18}$$

where

$$
\begin{aligned}
X_s &= \frac{y_s\, M}{M_s} \\
M &= \left(\sum_{s=1}^{\text{NS}} \frac{y_s}{M_s} \right)^{-1} \\
\phi_s &= \sum_{r=1}^{\text{NS}} X_r \left[1 + \sqrt{\frac{\mu_s}{\mu_r}} \left(\frac{M_r}{M_s} \right)^{1/4} \right]^2 \left[\sqrt{8 \left(1 + \frac{M_s}{M_r} \right)} \right]^{-1}
\end{aligned}
\tag{3.19}
$$

The Blottner curve fits for species viscosities are generally accepted to be accurate up to 10,000 K. Above 10,000 K, the Yos approximate mixing rule is the preferred method for computing the mixture viscosity and thermal conductivity.

The species diffusion coefficients $\mathcal{D}_s$ appearing in equations 3.1 and 3.2 must be defined. Accurate treatment of the species diffusion coefficients has received much attention in the literature. The simplest approach is to assume that all species have the same diffusion coefficient ($\mathcal{D} = \mathcal{D}_s$). This is only valid if the molecular weights of the species are similar. The single binary diffusion coefficient $\mathcal{D}$ can be computed assuming a constant Lewis number according to the relation

$$
\mathcal{D} = \frac{\text{Le}\kappa}{\rho C_p^{tr}} \tag{3.20}
$$

In the event the molecular weights of the species are disparate, determining the individual species diffusion coefficient is necessary. Species specific binary diffusion coefficients can be computed via Gupta and Yos curve fits [] or Ramshaw's Self-Consistent Effective Binary Diffusion method [].

3.3. SOURCE TERMS

The source terms for the mass conservation equations must be computed given a gas model (such as a 5-species or 11-species air model). Cantera, a general toolkit for chemical kinetics, can be used to compute the reaction rates and hence the chemical source terms needed in (3.1).

The vibrational energy source term appearing in (3.3) is computed as follows

$$
\dot{\omega}^v = \dot{Q}^v + \dot{Q}^{t,r-v} \tag{3.21}
$$

where $\dot{Q}^v$ is the vibrational energy production rate and $\dot{Q}^{t,r-v}$ is the translational-vibrational and rotational-vibrational energy exchange rate. All other energy exchange mechanisms are typically neglected when a two-temperature (translation-rotation and vibration) model are used.

The vibrational energy production rate is computed according to

$$
\dot{Q}^v = \sum_{s=1}^{\text{NS}} \dot{\omega}_s \left(e_s^v \right) \tag{3.22}
$$

and the translational-vibrational and rotational-vibrational energy exchange rate is given by

$$\dot{Q}^v = \sum_{s=1}^{\text{NS}} \rho_s \frac{e_s^v\left(T\right) - e_s^v\left(T^v\right)}{\tau_s^v} \tag{3.23}$$

where e_s^v is the specific vibrational internal energy for species s given in (3.8) and τ_s^v is the vibrational relaxation time. τ_s^v is typically computed via the Landau-Teller inter-species relaxation time given by Millikan-White.

4. SPATIAL DISCRETIZATION

The discretization of the governing equations (2.38) - (2.42) may be facilitated by recasting the conservation equations into vector form. For an ideal gas, let

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \rho u_i \\ \rho E \\ \rho k \\ \rho \omega \end{pmatrix} \tag{4.1}$$

$$\boldsymbol{F}_j(\boldsymbol{U}) = \begin{pmatrix} \rho u_j \\ \rho u_i u_j + P\delta_{ij} \\ \rho H u_j \\ \rho k u_j \\ \rho \omega u_j \end{pmatrix} \tag{4.2}$$

$$\boldsymbol{G}_j(\boldsymbol{U}) = \begin{pmatrix} 0 \\ \mu_{\text{eff}} \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3} \frac{\partial u_l}{\partial x_l} \delta_{ij} \right) - \frac{2}{3} \rho k \delta_{ij} \\ u_l \tau_{lj} + \kappa_{\text{eff}} \frac{\partial T}{\partial x_j} \\ \left(\mu + \frac{\sigma_k \rho k}{\omega} \right) \frac{\partial k}{\partial x_j} \\ \left(\mu + \frac{\sigma_\omega \rho k}{\omega} \right) \frac{\partial \omega}{\partial x_j} \end{pmatrix} \tag{4.3}$$

$$\boldsymbol{\mathcal{S}}(\boldsymbol{U}) = \begin{pmatrix} 0 \\ 0_j \\ -\sigma_{lm} \frac{\partial u_m}{\partial x_l} + \beta^\star \rho k \omega \\ P_k - D_k - S_k \\ P_\omega - D_\omega - S_\omega \end{pmatrix} \tag{4.4}$$

In the above definitions, the subscript $()_j$ denotes the coordinate direction associated with each flux vector $\boldsymbol{F}_j$ and $\boldsymbol{G}_j$. The subscript $()_i$ denotes the component of the momentum equation, which expands the length of each vector according to the spatial dimension: e.g., for two spatial dimensions, $\boldsymbol{U}$, $\boldsymbol{F}_j$, $\boldsymbol{G}_j$, and $\boldsymbol{\mathcal{S}}$ are each of length six: one continuity equation, two momentum equations, a total energy equation, and two turbulence model equations. Because some variables use k as a subscript to indicate the quantity is associated with the turbulent kinetic energy, to avoid confusion we will not use k as a Cartesian index subscript. Now, the conservation equations may be written as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}_j(\boldsymbol{U})}{\partial x_j} = \frac{\partial \boldsymbol{G}_j(\boldsymbol{U})}{\partial x_j} + \boldsymbol{\mathcal{S}}(\boldsymbol{U}), \tag{4.5}$$

Currently, we discretize the equations exclusively using a node-centered finite-volume approach. Figure 4.0-1 illustrates a typical finite volume, or cell, associated with node p. Let such a cell be denoted,

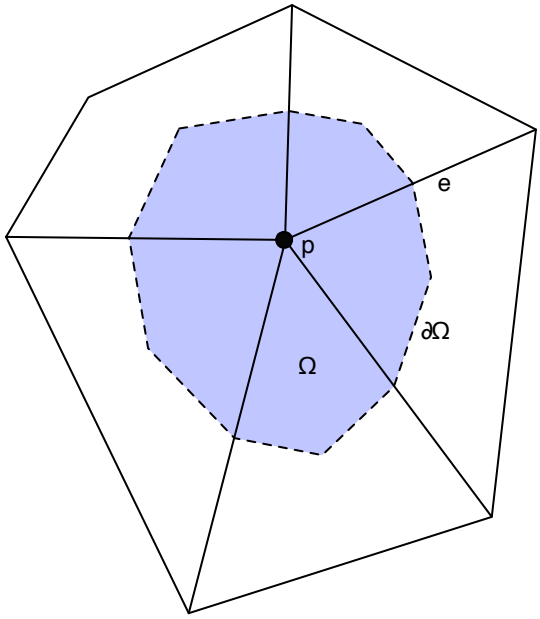

Figure 4.0-1.. Illustration of two-dimensional dual mesh for node p. The dual volume, Ω, is the polygon defined by the edge midpoints to the element centroids.

Ω. If we integrate (4.5) over Ω and apply the Gauss Divergence Theorem, the result may be written as

$$\int_{\Omega} \frac{\partial \boldsymbol{U}}{\partial t} d\mathcal{V} + \oint_{\partial\Omega} (\boldsymbol{F}_j - \boldsymbol{G}_j)\, d\mathcal{A}_j = \int_{\Omega} \boldsymbol{\mathcal{S}} d\mathcal{V} \tag{4.6}$$

where $\partial\Omega$ indicates the boundary of Ω and $d\mathcal{A}_j$ denotes an infinitesimal area vector on the surface $\partial\Omega$, Next, the integrals in (4.6) are approximated by numerical quadrature. The volume integral is approximated simply by multiplying the nodal value times the size of the control volume for that node. Hence, for node p, we may write

$$\int_{\Omega} \frac{\partial \boldsymbol{U}}{\partial t} d\mathcal{V} \simeq \frac{\partial \boldsymbol{U}^p}{\partial t} \mathcal{V}^p \tag{4.7}$$

$$\int_{\Omega} \boldsymbol{\mathcal{S}} d\mathcal{V} \simeq \boldsymbol{\mathcal{S}}^p \mathcal{V}^p \tag{4.8}$$

The surface integral is approximated by evaluating the fluxes at the midpoint of each edge where it is intersected by $\partial\Omega$ and computing the inner product of the flux and the area vector, viz.

$$\oint_{\partial\Omega} (\boldsymbol{F}_j - \boldsymbol{G}_j)\, d\mathcal{A}_j \simeq \sum_{e=1}^{\mathrm{NE}} (\boldsymbol{F}_n^e - \boldsymbol{G}_n^e)\, \mathcal{A}^e \tag{4.9}$$

where $\boldsymbol{F}_n^e = \boldsymbol{F}_j n_j$, n_j is the unit vector in the direction of the area vector, and $\mathcal{A}^e$ is the area of the dual cell face that is intersected by edge e. After substituting (4.7) - (4.9) into (4.6), we may write the semidiscrete residual for node p as

$$\frac{\partial \boldsymbol{U}^p}{\partial t} \mathcal{V}^p + \sum_{e=1}^{\mathrm{NE}} (\boldsymbol{F}_n^e - \boldsymbol{G}_n^e)\, \mathcal{A}^e - \boldsymbol{\mathcal{S}}^p \mathcal{V}^p = 0 \tag{4.10}$$

4.1. ADVECTIVE FLUX EVALUATION

In this section, we consider the details of evaluating the advective fluxes at the edge midpoint. Consider the arbitrary edge shown in Figure 4.1-1. The points p and q are the nodes defining the edge. L and R illustrate that the at the edge midpoint, which is the interface between the dual volumes around p and q, the solution is discontinuous. To construct a conservative flux at this interface, we currently use Roe's numerical flux function [9] or the Steger-Warming flux function [10].

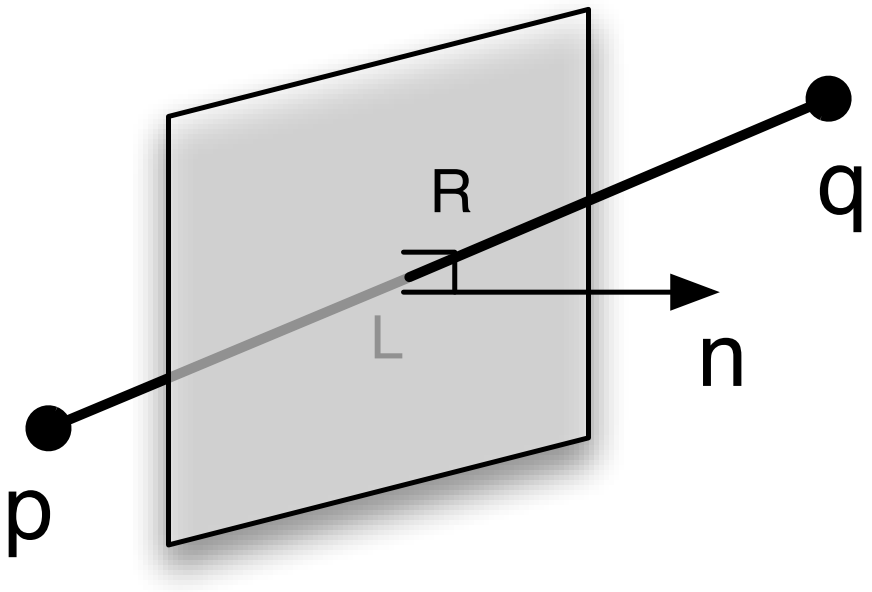

Figure 4.1-1.. The edge is defined by left node p and right node q, with unit normal vector n associated with the area facet, which may not be aligned with the edge

4.1.1. Roe's flux function

The Roe flux at each edge midpoint is defined by

$$\boldsymbol{F}_n^e = \frac{1}{2}\left(\boldsymbol{F}_L + \boldsymbol{F}_R\right) - \left|\bar{\boldsymbol{A}}_n\right|\left(\boldsymbol{U}_R - \boldsymbol{U}_L\right), \tag{4.11}$$

where $\boldsymbol{F}_L = \boldsymbol{F}(\boldsymbol{U}_L), \boldsymbol{U}_L$ is the solution value sampled from the left volume. This dissipation matrix is defined as

$$\left|\bar{\boldsymbol{A}}_n\right| = \boldsymbol{R}_n \left|\hat{\boldsymbol{\Lambda}}_n\right| \boldsymbol{R}_n^{-1}, \tag{4.12}$$

where

$$\boldsymbol{A}_n = \boldsymbol{R}_n \boldsymbol{\Lambda}_n \boldsymbol{R}_n^{-1} = \frac{\partial \boldsymbol{F}_n}{\partial \boldsymbol{U}} \tag{4.13}$$

is evaluated at the Roe average state such that,

$$\boldsymbol{F}_n(\boldsymbol{U}_L) - \boldsymbol{F}_n(\boldsymbol{U}_R) = \bar{\boldsymbol{A}}_n(\boldsymbol{U}_L - \boldsymbol{U}_R). \tag{4.14}$$

When the eigenvalues of the flux Jacobian are equal to zero, then the dissipation of certain characteristic waves vanishes. This can lead to issues with the numerical solution. To keep eigenvalues away from zero, we employ an entropy fix. The eigenvalues in our implementation are

$$\lambda_1 = u_n + c, \quad \lambda_2 = u_n - c, \quad \lambda_k = u_n, \quad k = 3, \ldots, N_q, \tag{4.15}$$

where N_q is the number of conserved variables in the system of equations. One approach to the entropy fix is to modify the eigenvalues with

$$\hat{\lambda}_k = \frac{\lambda_k^2 + \epsilon^2 \lambda_{\text{max}}^2}{2\epsilon\lambda_{\text{max}}} \quad \text{if} \quad |\lambda_k| < \epsilon\lambda_{\text{max}}, \quad \lambda_{\text{max}} = |u_n| + c, \tag{4.16}$$

where ϵ is a user defined value that defaults to $\epsilon = 0.1$. We refer to this as the scaled entropy fix. An alternative approach is to modify the eigenvalues as

$$\hat{\lambda}_k = \sqrt{\lambda_k^2 + \epsilon^2 \lambda_{\text{max}}^2} \quad \text{if} \quad |\lambda_k| < \epsilon\lambda_{\text{max}}, \tag{4.17}$$

which we refer to as the unscaled entropy fix. The modified eigenvalues are used in 4.12.

4.1.2. Steger-Warming flux function

The Steger-Warming flux-vector splitting method separates the inviscid fluxes into positive and negative parts based on the eigenvalues of the flux Jacobian matrix

$$\boldsymbol{F}_n^e = \boldsymbol{F}_n^+ + \boldsymbol{F}_n^- = \boldsymbol{A}_n^+ \boldsymbol{U} + \boldsymbol{A}_n^- \boldsymbol{U} \tag{4.18}$$

The flux Jacobians are defined

$$\boldsymbol{A}_n^+ = \boldsymbol{R}^{-1} \boldsymbol{\Lambda}^+ \boldsymbol{R} \qquad \text{and} \qquad \boldsymbol{A}_n^- = \boldsymbol{R}^{-1} \boldsymbol{\Lambda}^- \boldsymbol{R} \tag{4.19}$$

where $\boldsymbol{R}$ is the column matrix of right eigenvectors of $\boldsymbol{A}_n$, and $\boldsymbol{\Lambda}^{\pm}$ are the diagonal matrices of the positive and negative eigenvalues of $\boldsymbol{A}_n$. The flux Jacobians $\boldsymbol{A}^+$ and $\boldsymbol{A}^-$ are clearly defined in [11].

The original Steger-Warming method computes the flux at a face/edge using states at the left cell/node and right cell/node according to

$$\boldsymbol{F}_f = \boldsymbol{A}_L^+ \boldsymbol{U}_L + \boldsymbol{A}_R^- \boldsymbol{U}_R \tag{4.20}$$

However, while the original Steger-Warming scheme works well in the vicinity of shocks it is too dissipative to be used elsewhere in the flow. A modification to the original scheme can be introduced by

changing the evaluation of the $\boldsymbol{A}^+$ and $\boldsymbol{A}^-$ matrices to some average of the left and right cell/node states. Hence, the modified scheme can be represented as

$$\boldsymbol{F}_f = \boldsymbol{A}^+_{f^+}\boldsymbol{U}_L + \boldsymbol{A}^-_{f^-}\boldsymbol{U}_R \tag{4.21}$$

where the subscripts f^+ and f^- indicate that the Jacobians are evaluated at the averaged states $\boldsymbol{U}_{f+}$ and $\boldsymbol{U}_{f-}$, respectively. Druguet, Candler, and Nompelis [10] introduced the following pressure weighted averaging which has since become popular for high-speed flows

$$\boldsymbol{U}_{f+} = (1-w)\boldsymbol{U}_L + w\boldsymbol{U}_R \qquad \text{and} \qquad \boldsymbol{U}_{f-} = w\boldsymbol{U}_L + (1-w)\boldsymbol{U}_R \tag{4.22}$$

where

$$w = 1 - \frac{0.5}{(g\delta p)^2 + 1} \qquad \text{and} \qquad \delta p = \frac{p_R - p_L}{min(p_R, p_L)} \tag{4.23}$$

An additional aspect of the Steger-Warming scheme is that the eigenvalues of the $\boldsymbol{\Lambda}^+$ and $\boldsymbol{\Lambda}^-$ matrices are corrected according to

$$\lambda^\pm = 0.5\left(\lambda \pm \sqrt{\lambda^2 + \epsilon^2}\right) \tag{4.24}$$

where λ is the original eigenvalue of A and ϵ is usually computed as $\epsilon = 0.3c$, where c is the speed of sound. We can compute the flux Jacobians using an expansion

$$\boldsymbol{A}^\pm = V_A \otimes K_A + V_B \otimes K_B + \lambda_3^\pm I, \tag{4.25}$$

where for thermochemical transport with turbulence the expansion vectors are

$$\begin{aligned}
V_A &= \frac{1}{c}\left(\frac{\rho_i}{\rho}, u_j, H, k, \omega\right)^T, \quad V_B = (0, n_j, u_n, 0, 0)^T, \\
K_A &= \left(\left(\frac{\partial P}{\partial \rho_i}\frac{\mu_1^\pm}{c} - u_n\mu_2^\pm\right), \left(n_j\mu_2^\pm + \frac{\partial P}{\partial \rho u_j}\frac{\mu_1^\pm}{c}\right), \left(\frac{\partial P}{\partial E}\frac{\mu_1^\pm}{c}\right), 0, 0\right)^T, \\
K_B &= \left(\left(\frac{\partial P}{\partial \rho_i}\frac{\mu_2^\pm}{c} - u_n\mu_1^\pm\right), \left(n_j\mu_1^\pm + \frac{\partial P}{\partial \rho u_j}\frac{\mu_2^\pm}{c}\right), \left(\frac{\partial P}{\partial E}\frac{\mu_2^\pm}{c}\right), 0, 0\right)^T,
\end{aligned} \tag{4.26}$$

where

$$\mu_1^\pm = \frac{1}{2}\left(\lambda_1^\pm + \lambda_2^\pm - 2\lambda_3^\pm\right), \quad \mu_2^\pm = \frac{1}{2}\left(\lambda_1^\pm - \lambda_2^\pm\right). \tag{4.27}$$

The expansion vectors simplify easily for ideal gases or laminar flows.

4.1.3. Steger-Warming Central + Dissipation Form

Delineating the above scheme into clearly defined central and dissipative parts can be advantageous when it comes to modifying the central fluxes to increase the stability or accuracy of the scheme. In this section, the modified Steger-Warming method will be re-written in this form.

Beginning from the modified Steger-Warming expression (equation 4.21) and the definition of the A^+ and A^- matrices (equation 4.19), we recognize that Λ^+ and Λ^- can be expanded to

$$\boldsymbol{\Lambda}^+ = \frac{1}{2}\left(\boldsymbol{\Lambda} + |\boldsymbol{\Lambda}|\right) \qquad \text{and} \qquad \boldsymbol{\Lambda}^- = \frac{1}{2}\left(\boldsymbol{\Lambda} - |\boldsymbol{\Lambda}|\right) \tag{4.28}$$

Expanding all of these expressions give us

$$\boldsymbol{F}_f = \frac{1}{2}\left(\boldsymbol{R}^{-1}\boldsymbol{\Lambda}\boldsymbol{R}\right)_{f^+}\boldsymbol{U_L} + \frac{1}{2}\left(\boldsymbol{R}^{-1}|\boldsymbol{\Lambda}|\boldsymbol{R}\right)_{f^+}\boldsymbol{U}_L + \frac{1}{2}\left(\boldsymbol{R}^{-1}\boldsymbol{\Lambda}\boldsymbol{R}\right)_{f^-}\boldsymbol{U_R} - \frac{1}{2}\left(\boldsymbol{R}^{-1}|\boldsymbol{\Lambda}|\boldsymbol{R}\right)_{f^-}\boldsymbol{U}_R \tag{4.29}$$

or written more compactly as

$$\boldsymbol{F}_f = \frac{1}{2}\boldsymbol{A}_{f^+}\boldsymbol{U}_L + \frac{1}{2}\boldsymbol{A}_{f^-}\boldsymbol{U}_R \quad + \quad \frac{1}{2}\boldsymbol{A}^{diss}_{f^+}\boldsymbol{U}_L - \frac{1}{2}\boldsymbol{A}^{diss}_{f^-}\boldsymbol{U}_R \tag{4.30}$$

The previous equation indicates that we must evaluate the central Jacobian matrices A and the dissipation Jacobian matrices at both the f^+ and f^- average states. Listed explicitly, these four Jacobian matrices are:

$$\boldsymbol{A}_{f^+} = \frac{1}{2}\left(\boldsymbol{R}^{-1}\boldsymbol{\Lambda}\boldsymbol{R}\right)_{f^+} \tag{4.31}$$

$$\boldsymbol{A}_{f^-} = \frac{1}{2}\left(\boldsymbol{R}^{-1}\boldsymbol{\Lambda}\boldsymbol{R}\right)_{f^-} \tag{4.32}$$

$$\boldsymbol{A}^{diss}_{f^+} = \frac{1}{2}\left(\boldsymbol{R}^{-1}|\boldsymbol{\Lambda}|\boldsymbol{R}\right)_{f^+} \tag{4.33}$$

$$\boldsymbol{A}^{diss}_{f^-} = \frac{1}{2}\left(\boldsymbol{R}^{-1}|\boldsymbol{\Lambda}|\boldsymbol{R}\right)_{f^-} \tag{4.34}$$

where we must compute $\boldsymbol{\Lambda}$ and $|\boldsymbol{\Lambda}|$ at both the f^+ and f^- average states. We can get expressions for these matrices by solving the original definition of $\boldsymbol{\Lambda}^+$ and $\boldsymbol{\Lambda}^-$, given by (4.28) for $\boldsymbol{\Lambda}$ and $|\boldsymbol{\Lambda}|$. This yields

$$\boldsymbol{\Lambda} = \boldsymbol{\Lambda}^+ + \boldsymbol{\Lambda}^- \qquad \text{and} \qquad |\boldsymbol{\Lambda}| = \boldsymbol{\Lambda}^+ - \boldsymbol{\Lambda}^- \tag{4.35}$$

4.1.4. Kinetic Energy Preserving

The kinetic energy preserving flux [12] is a nondissipative flux algorithm that in the limit of incompressible flow does not produce any spurious kinetic energy due to the nonlinearity of the flux terms,

$$\boldsymbol{F}_f\left(\boldsymbol{U}_L, \boldsymbol{U}_R\right) = \begin{pmatrix} \frac{1}{2}\left[(\rho u_k n_k)_L + (\rho u_k n_k)_R\right] \\ \frac{1}{4}\left[(\rho u_k n_k)_L (u_1)_R + (\rho u_k n_k)_R (u_1)_L + (\rho u_k n_k u_1)_L + (\rho u_k n_k u_1)_R\right] \\ \quad + \frac{1}{2}\left[(p n_1)_L + (p n_1)_R\right] \\ \frac{1}{4}\left[(\rho u_k n_k)_L (u_2)_R + (\rho u_k n_k)_R (u_2)_L + (\rho u_k n_k u_2)_L + (\rho u_k n_k u_2)_R\right] \\ \quad + \frac{1}{2}\left[(p n_2)_L + (p n_2)_R\right] \\ \frac{1}{4}\left[(\rho u_k n_k)_L (u_3)_R + (\rho u_k n_k)_R (u_3)_L + (\rho u_k n_k u_3)_L + (\rho u_k n_k u_3)_R\right] \\ \quad + \frac{1}{2}\left[(p n_3)_L + (p n_3)_R\right] \\ \frac{1}{2}\left[(\rho u_k n_k H)_L + (\rho u_k n_k H)_R\right] \end{pmatrix}. \tag{4.36}$$

The nondissipative part of the flux can be combined with the dissipation from Roe where needed. This is handled in the hybrid flux approach, discussed below.

4.1.5. Honein and Moin

Another nondissipative algorithm in Aero comes from Honein and Moin [13], which was heuristically designed to exhibit more robustness for compressible flows than the kinetic energy preserving flux,

$$\boldsymbol{F}_f\left(\boldsymbol{U}_L,\boldsymbol{U}_R\right)=\left(\begin{array}{l}\frac{1}{2}\left[(\rho u_k n_k)_L+(\rho u_k n_k)_R\right]\\ \frac{1}{4}\left[(\rho u_k n_k)_L(u_1)_R+(\rho u_k n_k)_R(u_1)_L+(\rho u_k n_k u_1)_L+(\rho u_k n_k u_1)_R\right]\\ \quad+\frac{1}{2}\left[(pn_1)_L+(pn_1)_R\right]\\ \frac{1}{4}\left[(\rho u_k n_k)_L(u_2)_2+(\rho u_k n_k)_R(u_2)_L+(\rho u_k n_k u_2)_L+(\rho u_k n_k u_2)_R\right]\\ \quad+\frac{1}{2}\left[(pn_2)_L+(pn_2)_R\right]\\ \frac{1}{4}\left[(\rho u_k n_k)_L(u_3)_2+(\rho u_k n_k)_R(u_3)_L+(\rho u_k n_k u_3)_L+(\rho u_k n_k u_3)_R\right]\\ \quad+\frac{1}{2}\left[(pn_3)_L+(pn_3)_R\right]\\ \frac{1}{4}\left[(\rho u_j u_k n_k)_L(u_j)_R+(\rho u_j u_k n_k)_R(u_j)_1\right]\\ \quad+\frac{1}{4}\left[(\rho u_k n_k e)_L+(\rho u_k n_k e)_R+(\rho u_k n_k)_L(e)_R+(\rho u_k n_k)_R(e)_L\right]\\ \quad+\frac{1}{2}\left[(p)_L(u_k n_k)_R+(p)_R(u_k n_k)_L\right]\end{array}\right). \tag{4.37}$$

4.1.6. Entropy Preserving

For the compressible Euler equations, the entropy of the system should only change due to the boundary conditions. It is possible to construct a two-point flux that maintains this property [9],

$$\begin{aligned}\boldsymbol{F}_f\left(\boldsymbol{U}_L,\boldsymbol{U}_R\right)&=\left(\hat{\rho}\hat{u}_j,\hat{\rho}\hat{u}_j n_j\hat{u}_1+n_1\hat{p},\hat{\rho}\hat{u}_j n_j\hat{u}_2+n_2\hat{p},\hat{\rho}\hat{u}_j n_j\hat{u}_3+n_3\hat{p},\hat{\rho}\hat{u}_j n_j\hat{H}\right)^T,\\ \hat{u}_i&=\frac{\frac{(u_i)_L}{\sqrt{T_L}}+\frac{(u_i)_R}{\sqrt{T_R}}}{\frac{1}{\sqrt{T_L}}+\frac{1}{\sqrt{T_R}}},\quad \hat{p}=\frac{\frac{\hat{p}_L}{\sqrt{T_L}}+\frac{\hat{p}_R}{\sqrt{T_R}}}{\frac{1}{\sqrt{T_L}}+\frac{1}{\sqrt{T_R}}},\\ \hat{h}&=R\frac{\log\left(\frac{\sqrt{T_L}\rho_L}{\sqrt{T_R}\rho_R}\right)}{\frac{1}{\sqrt{T_L}}+\frac{1}{\sqrt{T_R}}}\left(\frac{\sqrt{T_L}\rho_L+\sqrt{T_R}\rho_R}{\left(\frac{1}{\sqrt{T_L}}+\frac{1}{\sqrt{T_R}}\right)\left(\sqrt{T_L}\rho_L-\sqrt{T_R}\rho_R\right)}\right.\\ &\qquad\left.+\frac{\gamma+1}{\gamma-1}\frac{\log\left(\sqrt{\frac{T_R}{T_L}}\right)}{\log\left(\sqrt{\frac{T_L}{T_R}}\frac{\rho_L}{\rho_R}\right)\left(\frac{1}{\sqrt{T_L}}-\frac{1}{\sqrt{T_R}}\right)}\right),\\ \hat{H}&=\hat{h}+\frac{1}{2}\hat{u}_\ell\hat{u}_\ell,\quad \hat{\rho}=\frac{\left(\frac{1}{\sqrt{T_L}}+\frac{1}{\sqrt{T_R}}\right)\left(\sqrt{T_L}\rho_L-\sqrt{T_R}\rho_R\right)}{2\left(\log(\sqrt{T_L}\rho_L)-\log(\sqrt{T_R}\rho_R)\right)}.\end{aligned} \tag{4.38}$$

4.1.7. First order spatial accuracy

The first order scheme is defined by a constant value of $\boldsymbol{U}$ over each control volume. In this case, $\boldsymbol{U}_L$ and $\boldsymbol{U}_R$ are defined by the two nodal values of the edge.

4.1.8. Second order spatial accuracy

The second order scheme uses MUSCL extrapolation [**?**] to reconstruct a linear variation of $\boldsymbol{U}$ over each control volume. This reconstruction is accomplished by going outside the cell to construct nodal gradients, which are then used to extrapolate the state variables along each edge from the node to the cell face.

4.1.9. Low-dissipation MUSCL-based fluxes

For Large Eddy Simulations (LES) or Detached Eddy Simulations (DES), fully upwinded methods are known to exhibit too much damping of turbulent eddies, resulting in low efficiency calculations for this class of flow. Toward reducing the dissipation in the implemented schemes and thus improving the efficiency of LES and DES, Aero has a hybrid algorithm that utilizes the MUSCL states. Near shocks, the fully dissipative flux is used. Away from shocks, we use a sensor function to blend a non-dissipative flux with a dissipative flux,

$$\boldsymbol{F}_f(\tilde{\boldsymbol{U}}_L, \tilde{\boldsymbol{U}}_R) = \alpha \boldsymbol{F}_f^{nodiss}(\tilde{\boldsymbol{U}}_L, \tilde{\boldsymbol{U}}_R) + (1-\alpha)\boldsymbol{F}_f^{diss}(\tilde{\boldsymbol{U}}_L, \tilde{\boldsymbol{U}}_R), \tag{4.39}$$

where α is determined by limiter values and $\tilde{\boldsymbol{U}}$ denotes the MUSCL extrapolation. Any nondissipative flux can be chosen, but the Kinetic Energy Preserving, Honein and Moin, or Entropy Preserving fluxes are preferred.

4.1.10. High-resolution hybrid fluxes

An alternative low-dissipation method utilizes multiple flux evaluations and results in a higher resolution algorithm on smooth grids,

$$\boldsymbol{F}_f(\hat{\boldsymbol{U}}_{LL}, \boldsymbol{U}_L, \boldsymbol{U}_R, \hat{\boldsymbol{U}}_{RR}) = \frac{4}{3} f_g(\boldsymbol{U}_L, \boldsymbol{U}_R) - \frac{1}{6}\left(f_g(\hat{\boldsymbol{U}}_{LL}, \boldsymbol{U}_R) + f_g(\boldsymbol{U}_L, \hat{\boldsymbol{U}}_{RR}) \right), \tag{4.40}$$

where f_g is a two-point function with the same form as described previously. $\hat{\boldsymbol{U}}_{LL}$ and $\hat{\boldsymbol{U}}_{RR}$ are extrapolated states, where we extrapolate in primitive variables,

$$\boldsymbol{V} = (\rho, u_1, u_2, u_3, T)^T. \tag{4.41}$$

using

$$\hat{\boldsymbol{V}}_{LL} = \boldsymbol{V}_R - 2\delta r \cdot \nabla \boldsymbol{V}|_L, \quad \hat{\boldsymbol{V}}_{RR} = \boldsymbol{V}_L + 2\delta r \cdot \nabla \boldsymbol{V}|_R, \tag{4.42}$$

where $\delta r = x_R - x_L$ and $\nabla \boldsymbol{V}$ represents the reconstructed nodal gradients of the primitive variables. The form of the phantom states may seem non-intuitive, but it recovers the structured definitions of a higher order stencil in one-dimension [14], thus ensuring that in one dimension the secondary preservation properties will be satisfied.

For hybrid fluxes, a very similar form to above is utilized,

$$\boldsymbol{F}_f = \alpha \boldsymbol{F}_f^{nodiss} \boldsymbol{F}_f(\hat{\boldsymbol{U}}_{LL}, \boldsymbol{U}_L, \boldsymbol{U}_R, \hat{\boldsymbol{U}}_{RR}) + (1-\alpha)\boldsymbol{F}_f^{diss}(\tilde{\boldsymbol{U}}_L, \tilde{\boldsymbol{U}}_R). \tag{4.43}$$

Note that the dissipative part again utilizes the MUSCL states. However, near shocks simple upwinding is used.

4.2. HYBRID SENSORS

To compute the sensor value in the hybrid fluxes described above, we first must determine if and where shocks occur in the domain. To do this, we utilize the modified Ducros sensor [15],

$$\begin{aligned} \phi &= \frac{1-\tanh\left(\frac{5}{2}+20\frac{\Delta\theta}{a}\right)}{2}\frac{\theta^2}{\theta^2+\omega\cdot\omega+\epsilon}, \\ \theta &= \frac{\partial u_k}{\partial x_k}, \quad \epsilon = 10^{-6}, \quad \Delta = \mathcal{V}^{1/d}, \end{aligned} \tag{4.44}$$

where ω is the vorticity and d is the number of dimensions. When the maximum of phi over the stencil of a given node is greater than a specified tolerance (1.0e-03), we set $\alpha = 1$ for the hybrid algorithm.

The value of α used in the hybrid algorithm is simply the average of left and right state for a given edge. Away from a shock, we may still require dissipation where the grid is nonsmooth or the solution contains underresolved features that lead to oscillations and noise. Thus, if a limiter is used, we add the average edge value of the limiter to α. This greatly increases the robustness of the simulations utilizing hybrid fluxes. More advanced sensors will be explored in the future.

5. TIME MARCHING

Both implicit and explicit time marching methods are included in Aero. Time-accurate and steady-state calculations may be made with implicit techniques, which allow large time steps but require LHS (left-hand-side) sensitivities. Explicit methods do not require sensitivities and have lower memory usage because a linear system is not needed, at the cost of stricter stability limits on timestep size.

The governing equations after the spatial discretization can be written in the semi discrete form

$$\frac{d\boldsymbol{U}}{dt} = \boldsymbol{R}(t, \boldsymbol{U}(t)), \quad \boldsymbol{U}(t_o) = \boldsymbol{U}_o, \tag{5.1}$$

where $\boldsymbol{R}(t, \boldsymbol{U}(t))$ contains the advection, diffusion, and source terms.

5.1. EXPLICIT METHODS

The explicit schemes in Aero can all be cast as low-storage Runge-Kutta (LSRK) schemes[16]. Low-storage schemes require $2N$ units of storage where N is the dimension of the system of ODEs.

A multistage low-storage Runge-Kutta scheme can be written as:

$$t_j = t^{n-1} + c_j \Delta t, \tag{5.2}$$
$$\Delta \boldsymbol{U}_j = A_j \Delta \boldsymbol{U}_{j-1} + \Delta t \boldsymbol{R}(t_{j-1}, \boldsymbol{U}_{j-1}), \tag{5.3}$$
$$\boldsymbol{U}_j = \boldsymbol{U}_{j-1} + B_j \Delta \boldsymbol{U}_j, \tag{5.4}$$
$$\vdots \tag{5.5}$$
$$\boldsymbol{U}^n = \boldsymbol{U}_s \tag{5.6}$$

where the subindex j denotes the stage number, s is the number of stages, and the superscripts denote the time level. The coefficients A_j, B_j, and c_j are designed for many constraints, primarily to maintain order of accuracy and to give a large stability region. Carpenter and Kennedy[16] give details of how to determine these coefficients for third- and fourth-order schemes. Two low-storage Runge-Kutta schemes are included in Aero, based order of accuracy and stability limits of the schemes.

5.1.1. Forward Euler

The simplest explicit time marching scheme is the single-stage Forward Euler method. This is the cheapest consistent integration method, although it is only first-order accurate and has very strict stability requirements. Its LSRK coefficients are found in Table 5.1-1.

$A_1 = 0$	$B_1 = 1$	$c_1 = 0$

Table 5.1-1.. LSRK coefficients for the Forward Euler method

5.1.2. 5-stage RK4

A fourth-order LSRK scheme can be obtained with five stages. Several solutions exist - Aero uses solution three from Carpenter and Kennedy[16]. The coefficients for this method are found in Table 5.1-2.

$A_1 = 0$	$B_1 = 0.1496590219993$	$c_1 = 0$
$A_2 = -0.4178904745$	$B_2 = 0.379210312999$	$c_2 = 0.1496590219993$
$A_3 = -1.192151694643$	$B_3 = 0.8229550293869$	$c_3 = 0.3704009573644$
$A_4 = -1.697784692471$	$B_4 = 0.6994504559488$	$c_4 = 0.6222557631345$
$A_5 = -1.514183444257$	$B_5 = 0.1530572479681$	$c_5 = 0.9582821306748$

Table 5.1-2.. Coefficients for 5-stage RK4 explicit time marching scheme

5.2. POINT IMPLICIT

A point implicit algorithm is used to solve both the steady-state and time-accurate versions of the implicit algorithm. In the steady-state case, the solution is still marched in time to drive the steady-state residuals to zero but the solution is not time accurate because the residual does not include the time term. For the time-accurate case, the time terms are added to the residual and at each timestep the total residual is driven towards zero using Newton's method for solving nonlinear equations.

In all cases, a non-linear equation is solved using Newton's method which requires a series of linear solves and an update of the conserved variables:

$$\boldsymbol{A}_j \Delta \boldsymbol{U}_j = \boldsymbol{R}_j, \tag{5.7}$$

$$\boldsymbol{U}_{j+1}^{n+1} = \boldsymbol{U}_j^{n+1} + \Delta \boldsymbol{U}_j \tag{5.8}$$

where n denotes timestep and j denotes nonlinear iteration, and the LHS matrix $\boldsymbol{A}_j$ is slightly different for each time-marching method and defined below for each one.

The solution to the linear system is obtained through a relaxation method. $\boldsymbol{A}_j$ is split into diagonal and off-diagonal terms

$$\boldsymbol{A}_j = \boldsymbol{D}_j + \boldsymbol{O}_j \tag{5.9}$$

A Jacobi iteration is used and the off-diagonal terms are moved to the RHS(right hand side) and are evaluated using the previous subiteration value of $\boldsymbol{U}^i$ where i denotes the linear subiteration. The resulting scheme is then

$$\boldsymbol{D}_j \Delta \boldsymbol{U}_j^{i+1} = \boldsymbol{R}_j - \boldsymbol{O}_j \Delta \boldsymbol{U}_j^i \tag{5.10}$$

The matrices $\boldsymbol{D}_j$, $\boldsymbol{O}_j$, and $\boldsymbol{R}_j$ are updated at each nonlinear iteration for the time-accurate case and only a single nonlinear iteration is typically used for steady-state problems.

Depending on the solution a full update taken for each nonlinear iteration may result in negative temperatures or negative densities and lead to failure of the nonlinear solver. This is mitigated in Aero by combining a local relaxation approach with a line search algorithm.

5.2.1. Local Relaxation

The local relaxation algorithm is used to limit updates at the nodal level to avoid potential stability problems due to a poor initial guess or unknown/unrealistic initial conditions. Effectively, a coefficient matrix $\Omega \leq I$ is applied in 5.8,

$$\boldsymbol{U}_{j+1}^{n+1} = \boldsymbol{U}_{j}^{n+1} + \Omega \Delta \boldsymbol{U}_j, \tag{5.11}$$

where at each node the local relaxation factor is calculated when $\frac{\Delta P}{P} \geq c$ or $\frac{\Delta T}{T} \geq c$,

$$\Omega_i = \min\left(\frac{Pc}{\Delta P}, \frac{Tc}{\Delta T}\right), \tag{5.12}$$

where Ω_i reduces the update of the full conservative vector at the node uniformly. Because of the nonlinear dependence of P and T on U, this will not necessarily result in $\frac{\Delta P}{P} \leq c$ or $\frac{\Delta T}{T} \leq c$, but will give a close approximation.

Local relaxation is also used to avoid updating the pressure and temperature below user specified minimum values. The predicted pressure and temperatures of the full update, P' and T', respectively, are limited using,

$$\Omega_i = \min\left(\frac{P_{min} - P}{2\Delta P}, \frac{T_{min} - T}{2\Delta T}\right). \tag{5.13}$$

As discussed above, because of the nonlinear nature of P and T with respect to the update, this limiting procedure may fail when P and T are near their respective minimum values relative to the update size.

5.2.2. Line Search

The line search algorithm ensures that the composite residual of the nonlinear solver decreases after the update. In other words,

$$\boldsymbol{R}^*(\boldsymbol{U}_j^{n+1} + \omega \boldsymbol{\Delta U}) < \boldsymbol{R}^*(\boldsymbol{U}_j^{n+1}), \tag{5.14}$$

where ω is the global relaxation factor for the line search. Initially, $\omega = 1$ is used. If equation 5.14 is not satisfied, then $\omega = \frac{1}{2}\omega$ is used until equation 5.14 is satisfied. In Aero, if $\omega = \frac{1}{16}$, then the update is taken anyway.

5.2.3. Steady-State

Steady-state time advancement is based on the linearized backward-Euler time marching scheme.

$$\frac{d\boldsymbol{U}}{dt} = \frac{\boldsymbol{U}^{n+1} - \boldsymbol{U}^n}{\Delta t} = \boldsymbol{R}(t, \boldsymbol{U}^n(t)). \tag{5.15}$$

In this case the time-term is not included in the residual but is included in the sensitivities. Only a single linearization is done for each time step.

$$\boldsymbol{A}_0 = \frac{V}{\Delta t}\boldsymbol{I} - \frac{\partial \boldsymbol{R}}{\partial \boldsymbol{U}^n} \tag{5.16}$$

5.2.4. Time accurate Backward Euler

For time-accurate backward Euler, the time derivative is approximated as:

$$\frac{d\boldsymbol{U}}{dt} = \frac{\boldsymbol{U}^{n+1} - \boldsymbol{U}^n}{\Delta t} \tag{5.17}$$

The time-derivative term is included in the residual term. The modified residual is now

$$\boldsymbol{R}^*(\boldsymbol{U}_j^{n+1}) = \boldsymbol{R}(\boldsymbol{U}_j^{n+1}) - \frac{\boldsymbol{U}_j^{n+1} - \boldsymbol{U}^n}{\Delta t} \tag{5.18}$$

where j denotes the nonlinear iteration and $\boldsymbol{U}_0^{n+1} = \boldsymbol{U}^n$. The resulting nonlinear equation,

$$\boldsymbol{R}^*(\boldsymbol{U}_j^{n+1}) = 0, \tag{5.19}$$

is solved using Newton's method. The LHS is:

$$\boldsymbol{A}_j = -\frac{\partial \boldsymbol{R}_j^*(\boldsymbol{U}_j^{n+1})}{\partial \boldsymbol{U}_j^{n+1}} = \frac{V}{\Delta t}\boldsymbol{I} - \frac{\partial \boldsymbol{R}_j(\boldsymbol{U}_j^{n+1})}{\partial \boldsymbol{U}_j^{n+1}} \tag{5.20}$$

5.2.5. Time accurate BDF2

BDF2 is very similar to time-accurate backward Euler except the time derivative is approximated as:

$$\frac{d\boldsymbol{U}}{dt} = \frac{1}{\Delta t}\left(\frac{3}{2}\boldsymbol{U}^{n+1} - 2\boldsymbol{U}^n + \frac{1}{2}\boldsymbol{U}^{n-1}\right) \tag{5.21}$$

The above formulation assumes constant timestep.

The time-derivative term is included in the residual term. The modified residual is now

$$\boldsymbol{R}^*(\boldsymbol{U}_j^{n+1}) = \boldsymbol{R}(\boldsymbol{U}_j^{n+1}) - \frac{1}{\Delta t}\left(\frac{3}{2}\boldsymbol{U}_j^{n+1} - 2\boldsymbol{U}^n + \frac{1}{2}\boldsymbol{U}^{n-1}\right) \tag{5.22}$$

where j denotes the nonlinear iteration and $\boldsymbol{U}_0^{n+1} = \boldsymbol{U}^n$. The resulting nonlinear equation,

$$\boldsymbol{R}_*^n(\boldsymbol{U}_j^{n+1}) = 0, \tag{5.23}$$

is solved using Newton's method. The LHS is:

$$\boldsymbol{A}_j = -\frac{\partial \boldsymbol{R}_j^*(\boldsymbol{U}_j^{n+1})}{\partial \boldsymbol{U}_j^{n+1}} = \frac{3}{2}\frac{V}{\Delta t}\boldsymbol{I} - \frac{\partial \boldsymbol{R}_j(\boldsymbol{U}_j^{n+1})}{\partial \boldsymbol{U}_j^{n+1}} \tag{5.24}$$

5.3. DIAGONALLY IMPLICIT RUNGE KUTTA

Diagonally implicit Runge Kutta schemes are useful for numerically stiff problems where time accuracy is important. Diagonally implicit Runge Kutta schemes are similar to explicit Runge Kutta schemes in that multiple stages are needed for each time step. However, for the implicit scheme, the solution at the current stage is implicitly solved for and therefore a nonlinear solve is needed for each stage. A general Runge Kutta scheme to solve the semi-discrete equation 5.1 can be written as a series of stage calculations and a solution update. The stage calculation is:

$$U^i = U^n + \Delta t \sum_{j=1}^{s} a_{i,j} \boldsymbol{R}(t^n + c_j \Delta t, \boldsymbol{U}^j) \tag{5.25}$$

where the i index denotes stage and the n index denotes timestep. This equation must be solved nonlinearly if there are non-zero values on the diagonally of $a_{i,j}$.

The update to the solution for the next timestep is:

$$U^{n+1} = U^n + \Delta t \sum_{i=1}^{s} b_i \boldsymbol{R}(t^n + c_i \Delta t, \boldsymbol{U}^i) \tag{5.26}$$

These schemes can be expressed in a Butcher tableau format and includes extra coefficients ($\tilde{b}_i$) that give a solution one order lower than the coefficients b_i. These extra coefficients are useful in determining the time discretization error and can be used for adaptive time stepping.

c_i	$a_{i,j}$
	b_i
	$\tilde{b}_i$

Table 5.3-1.. Butcher tableau format for Runge Kutta methods.

In Aero, the following six stage fourth order diagonally implicit Runge Kutta scheme is used.

0	0	0	0	0	0	0
$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{4}$	0	0	0	0
$\frac{83}{250}$	$\frac{8611}{62500}$	$-\frac{1743}{31250}$	$\frac{1}{4}$	0	0	0
$\frac{31}{50}$	$\frac{5012029}{34652500}$	$-\frac{654441}{2922500}$	$\frac{174375}{388108}$	$\frac{1}{4}$	0	0
$\frac{17}{20}$	$\frac{15267082809}{155376265600}$	$-\frac{71443401}{120774400}$	$\frac{730878875}{902184768}$	$\frac{2285395}{8070912}$	$\frac{1}{4}$	0
1	$\frac{82889}{524892}$	0	$\frac{15625}{83664}$	$\frac{69875}{102672}$	$-\frac{2260}{8211}$	$\frac{1}{4}$
b_i	$\frac{82889}{524892}$	0	$\frac{15625}{83664}$	$\frac{69875}{102672}$	$-\frac{2260}{8211}$	$\frac{1}{4}$
$\tilde{b}_i$	$\frac{4586570599}{29645900160}$	0	$\frac{178811875}{945068544}$	$\frac{814220225}{1159782912}$	$-\frac{3700637}{11593932}$	$\frac{61727}{225920}$

Table 5.3-2.. Coefficients for six stage fourth order Diagonally Implicit Runge Kutta.

5.4. ADAPTIVE TIME-STEPPING & TEMPORAL ERROR CONTROL

This section details the adaptive temporal error control methodology in Aero. Consider the first step of a numerical solution, resulting in an approximate solution v^{n+1}. Given an exact solution $u(t)$, the local temporal error, ℓ^{n+1}, is

$$\ell^{n+1} \equiv u(t^n + \Delta t^n) - v^{n+1}. \tag{5.27}$$

Suppose the numerical method has order of accuracy p. Then the error of a single step is

$$\ell^{n+1} \equiv (\Delta t)^{p+1}\phi^n + \text{higher order terms}, \tag{5.28}$$

where ϕ^n is the principal error function. An estimation, $\hat{\ell}^{n+1}$, of the local error is found with a method of order $q > p$ that gives approximation $\hat{v}^{n+1}$.

$$\hat{\ell}^{n+1} \equiv \hat{v}^{n+1} - v^{n+1} = (\Delta t)^{p+1}\hat{\phi}^n + \text{higher order terms}. \tag{5.29}$$

The objective is to control the local temporal error by adjusting the time step. A target error ε is specified, towards which we drive a norm of the estimate, $r^{n+1} \equiv \|\hat{\ell}^{n+1}\|$ (an error-per-step (EPS) criterion). To avoid risking step rejection we drive the error norm to $\Theta\varepsilon$, where $\Theta = 0.8$ is a factor of safety.

To compute the error norm, we first compute the relative difference, $\delta_j^{n+1} = \hat{\ell}^{n+1}/S_j$, where S_j is a rough scale of the degree of freedom j. The L2-norm of this relative error is integrated over the domain volume,

$$(r^{n+1})^2 = \frac{1}{V}\int_V (\delta^{n+1})^2 dV \approx \frac{1}{V}\sum_{i=1}^{n_{\text{elem}}}\sum_{j=1}^{n_{\text{var}}}(\delta_j^{n+1})^2 V_i, \tag{5.30}$$

in which n_{var} is the number of variables in the local state vector.

There are several popular means of controlling the time step in response to the error sequence. In Aero we provide three digital feedback controllers - the common 'elementary' controller, a PI

(proportional-integral) scheme, and a PID (proportional-integral-derivative) controller. In all of these controllers we use a limiter of the form $\Delta t^{n+1} \leq R\Delta t^n$ where $R = 2.0$ is the 'maximum ramp' that prevents unreasonable step size increases. The controller is disengaged only when limiting the step increase, modifying Δt for FSI coupling, or enforcing the termination time. We do not employ deadzones (cut-outs) wherein the time step is fixed until a substantial change is necessary.

5.4.1. Elementary Control

The elementary controller is derived by first assuming that the time step is in the asymptotic range of the numerical method (the higher order terms can be neglected). Then our error norm is exactly $r^{n+1} = (\Delta t)^{p+1}\|\hat{\phi}^n\|$. Adjusting the time step by an amount $\eta(\Delta t)$ and equating the error to $\Theta\varepsilon$ yields $\eta = \Delta t^{n+1}/\Delta t^n = (\Theta\varepsilon/r^{n+1})^{1/p+1}$. Thus the elementary controller modifies the time step according to

$$\Delta t^{n+1} = \Delta t^n \left(\frac{\Theta\varepsilon}{r^{n+1}}\right)^{1/p+1}. \tag{5.31}$$

The elementary controller is very common and straightforward to derive. It has first-order dynamics and its single pole is at the origin (so-called 'deadbeat' control), giving it the best intrinsic stability properties. However it has several shortcomings. It is based entirely upon the process model of $r^{n+1} = (\Delta t)^{p+1}\|\hat{\phi}^n\|$, which requires that Δt is in the asymptotic range. This assumption is not always met - for instance when numerical stability limits the step size or when stiff problems are solved with L-stable implicit methods and large time steps. It is well-known that this controller tends to oscillate around stability boundaries, affecting the smoothness of the resultant numerical solution [17].

A particular challenge with the elementary controller is exposed by its frequency response - there is no attenuation of high frequencies in the stepsize transfer map, meaning that the spectral properties of $\|\hat{\phi}^n\|$, which represents the 'physics' of the underlying solution, are transmitted without attenuation to the stepsize. If the problem is noisy, then the resultant stepsize sequence will be just as noisy. Designing controllers via 'noise-shaping' and moving away from deadbeat control (placing poles away from the origin) allows us to obtain smoother stepsize sequences, as discussed in [18].

5.4.2. PI and PID Control

Gustafsson et al. [19] approached the problem of adaptive time-stepping from a control-theoretic perspective, and derived PI controllers as straightforward improvements to the elementary controller. These methods modify the time step as

$$\Delta t^{n+1} = \Delta t^n \left(\frac{r^n}{r^{n+1}}\right)^{k_P} \left(\frac{\Theta\varepsilon}{r^{n+1}}\right)^{k_I}, \tag{5.32}$$

where k_P and k_I are the proportional and integral gains, respectively. Comparing this formula to Equation (5.31) shows that the elementary controller is an I-controller ($k_P = 0$) with gain $k_I = 1/(p+1)$.

Observe that the integral mode changes Δt according to its distance from the target, while the proportional mode modifies the step by a trend of increasing or decreasing error. In specifying k_P and k_I, we can optimize controller dynamics for stability and monotonic/oscillatory/deadbeat control (pole placement), frequency response, and the 'aggressiveness' of the adaptation. [19] and [18] focus on the problem of obtaining smoothed stepsize histories. The controllers of [18] struggle to operate near the stability boundary of the time-stepping methods because they have extremely low responsiveness to high-frequency forcing. Aero provides one PI controller, custom-tuned for aggressive time-stepping near the stability boundary, with some of the stepsize-smoothing property of controllers in [18]. The gains of this controller are $k_I = 0.6$ and $k_P = -0.1$.

It is common to utilize derivative mode control, as is used by [20] and discussed by [18]. Here the stepsize control structure is

$$\Delta t^{n+1} = \Delta t^n \left(\frac{r^n}{r^{n+1}} \right)^{k_P} \left(\frac{\Theta \varepsilon}{r^{n+1}} \right)^{k_I} \left(\frac{r^n r^n}{r^{n+1} r^{n-1}} \right)^{k_D}, \tag{5.33}$$

where k_D is the derivative gain. Aero provides the controller with the tuning of [20], with $k_P = 0.14$, $k_I = 0.25$, and $k_D = 0.10$.

5.4.3. Adaptive Explicit Methods

Calculation of the error estimate requires a pair of discretizations. For implicit calculations, the pair contained in the DIRK method is provided in §5.3. Aero provides three different pairs of embedded explicit low-storage Runge-Kutta methods. The Forward Euler method (Table 5.1-1) and the second-order explicit trapezoidal method form two of these methods. ERK1(2) uses Forward Euler to update the solution and trapezoidal to estimate the error, while ERK2(1) is the same pair run in local extrapolation mode wherein the higher-order method (trapezoidal) is used for the update.

The fourth-order method given in Table 5.1-2 can be paired with an embedded third-order method. The A and c coefficients are identical, but the update coefficients [21] are given by $\hat{B}_5 = B_5$ and $\hat{B}_i = B_i - A_{i+1} B_{i+1}$ for $i \in 1, 2, 3, 4$. This gives the ERK4(3) scheme, a fourth-order accurate method with a third-order error estimate. The coefficients of the third-order scheme are not designed for time integration and can only be used for error estimation.

6. BOUNDARY CONDITIONS

The enforcement of boundary conditions is important for any numerical solution of partial differential equations. Below, the enforcement of boundary conditions is detailed for the currently supported boundary types in Aero.

6.1. SOLID WALL

6.1.1. Dirichlet no slip wall

On a no slip wall, velocity is set to the prescribed value of the wall. For the continuity equation, this means that there is no mass flux across the wall. This condition is applied weakly. The residual for the momentum equation is replaced by

$$u_j - g_j = 0$$

where g_j is the specified velocity at the wall. For a no-slip wall, $g_j = 0$. The rows in the iterative matrix for the Newton system are similarly removed and replaced by the Jacobian of this new residual. For example, for a three dimensional flow, the matrix equations associated with the momentum equation for a node on a no slip wall are replaced by

$$\begin{pmatrix} \frac{-u_1}{\rho} & \frac{1}{\rho} & 0 & 0 & 0 \\ \frac{-u_2}{\rho} & 0 & \frac{1}{\rho} & 0 & 0 \\ \frac{-u_3}{\rho} & 0 & 0 & \frac{1}{\rho} & 0 \end{pmatrix} \begin{pmatrix} \delta u_1 \\ \delta u_2 \\ \delta u_3 \end{pmatrix} = \begin{pmatrix} u_1 - g_1 \\ u_2 - g_2 \\ u_3 - g_3 \end{pmatrix}$$

For an isothermal wall, the energy equation is also removed and temperature is specified. On an adiabatic wall, temperature is allowed to float, and the specification of zero heat flux is automatically handled because the contribution to the flux balance of the viscous flux through an adiabatic no slip wall is zero. balance is again zero. Turbulence equations are also removed and the turbulent variables are set to a specified state.

6.1.1.1. SST k-ω turbulence model

The turbulent kinetic energy is simply set to zero,

$$(\rho k)_{\text{wall}} = 0 \tag{6.1}$$

Its specific dissipation rate is set to

$$(\rho \omega)_{\text{wall}} = \frac{60\mu}{\beta_1 d^2} \tag{6.2}$$

where β_1 is obtained from Table 2.3-1. Since the limit of (6.2) as $d \to 0$ is infinity, (6.2) cannot be evaluated directly at the wall. Instead, the right hand side of (6.2) is evaluated at a nearby location. The residual and matrix entries for the rows associated with the turbulence equations are replaced by

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \delta(\rho k) \\ \delta(\rho\omega) \end{pmatrix} = \begin{pmatrix} \rho k \\ \rho\omega - (\rho\omega)_{\text{wall}} \end{pmatrix}$$

6.1.1.2. k-ϵ turbulence model

At a solid wall, the turbulent kinetic energy is simply set to zero.

$$(\rho k)_{\text{wall}} = 0$$

Its dissipation rate is set to

$$(\rho\epsilon)_{\text{wall}} = \frac{2\mu k}{d^2} \tag{6.3}$$

Since the limit of (6.3) as $d \to 0$ is infinity, (6.3) cannot be evaluated directly at the wall. Instead, the right hand side of (6.3) is evaluated at a nearby location. The residual and Jacobian matrix rows associated with the solid walls are replaced by the following matrix system

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \delta(\rho k) \\ \delta(\rho\epsilon) \end{pmatrix} = \begin{pmatrix} \rho k \\ \rho\epsilon - (\rho\epsilon)_{\text{wall}} \end{pmatrix},$$

where we have neglected the sensitivities of$(\rho\epsilon)_{\text{wall}}$.

6.1.1.3. Spalart-Allmaras turbulence model

At a solid wall, the working variable is set to zero.

$$(\rho\hat{\nu})_{\text{wall}} = 0 \tag{6.4}$$

The residual and Jacobian matrix rows associated with the solid walls are replaced by

$$\delta(\rho\hat{\nu}) = (\rho\hat{\nu})_{\text{wall}}$$

6.1.2. Weak no slip wall

It is often advantageous to impose the no slip conditions in a weak sense instead of forcing the solution to be equal to the boundary condition at the wall. In the limit of infinite resolution, the weak boundary condition gives the same solution as a strongly enforced boundary condition. However, in practical cases weak boundary conditions typically result in more accurate solutions for the same resolution [22]. Furthermore, the solution at the wall for a weakly enforced no slip condition can be used to estimate the error in the overall solution. As flow features such as the boundary layer become better resolved, the error in the no slip condition decreases. In the asymptotic range of the numerical method used for the

simulation, the error at the boundary will decrease by the designed order of accuracy as the grid is refined. This specification is similar to that of a cell-centered finite volume approach.

To enforce the no slip condition weakly, we specify the total flux at the boundary using the difference between the computed solution and the specified wall condition:

$$\mathbf{F}_b = -\mathbf{F}_n(\mathbf{U}, \mathbf{U}_b) + \mathbf{G}(\mathbf{U}, \widetilde{\nabla \mathbf{V}(\mathbf{U})}), \tag{6.5}$$

where $\widetilde{\nabla \mathbf{V}(\mathbf{U})}$ is a modified projected nodal gradient at the boundaries. The modifications for the velocity gradients are

$$\widetilde{\partial_{x_j} u_i} = \partial_{x_j} u_i - \frac{u_i - (u_{\text{wall}})_i}{d} n_j, \tag{6.6}$$

where d is the some measure of the normal wall spacing. For isothermal walls the modified temperature gradient is

$$\widetilde{\partial_{x_j} T} = \partial_{x_j} T - \frac{T - T_{\text{wall}}}{d} n_j, \tag{6.7}$$

and for specified heat flux walls, the modified temperature gradient is

$$\widetilde{\partial_{x_j} T} = \partial_{x_j} T + (g_{\text{wall}} - \partial x_k T n_k) n_j, \quad g_{\text{wall}} = -\frac{q_{\text{wall}}}{\kappa}, \tag{6.8}$$

where q_{wall} is the specified wall heat flux.

The flux reconstruction, $\mathbf{F}_n$ can use any dissipative reconstruction function. The boundary state used in the reconstruction is defined as

$$\mathbf{U}_b = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \delta_{ij} - 2 n_i n_j & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{U}, \tag{6.9}$$

which reverses the normal velocity, but does not modify any slip velocity, density, or energy.

6.1.2.1. SST k-ω turbulence model

For the SST model, the same wall conditions as above are used:

$$(k)_{\text{wall}} = 0,$$

$$(\omega)_{\text{wall}} = \frac{60\mu}{\beta_1 \rho d^2}.$$

The modified gradients are

$$\begin{aligned} \widetilde{\partial_{x_j} k} &= \partial_{x_j} k - \frac{k - k_{\text{wall}}}{d} n_j, \\ \widetilde{\partial_{x_j} \omega} &= \partial_{x_j} \omega - \frac{\omega - \omega_{\text{wall}}}{d} n_j. \end{aligned} \tag{6.10}$$

6.1.2.2. k-ϵ turbulence model

For the k-ϵ model, wall conditions are:

$$(k)_{\text{wall}} = 0,$$
$$(\epsilon)_{\text{wall}} = \frac{2\mu k}{\rho d^2}$$

The modified gradients are

$$\begin{aligned} \widetilde{\partial_{x_j} k} &= \partial_{x_j} k - \frac{k - k_{\text{wall}}}{d} n_j, \\ \widetilde{\partial_{x_j} \epsilon} &= \partial_{x_j} \epsilon - \frac{\epsilon - \epsilon_{\text{wall}}}{d} n_j. \end{aligned} \tag{6.11}$$

6.1.2.3. Spalart-Allmaras turbulence model

For the Spalart-Allmaras model, wall conditions are

$$\hat{\nu}_{\text{wall}} = 0.$$

The modified gradients are

$$\widetilde{\partial_{x_j} \hat{\nu}} = \partial_{x_j} \hat{\nu} - \frac{\hat{\nu} - \hat{\nu}_{\text{wall}}}{d} n_j. \tag{6.12}$$

6.1.3. Turbulent Wall Function

For a turbulent boundary layer at high Reynolds number, the minimum wall spacing required to resolve the turbulent boundary layer can be quite small. Wall functions can be used to reduce this wall spacing requirement by modeling wall shear stress and heat transfer using the law of the wall[23]. The assumptions of using the law of the wall for the wall function boundary condition are:

- local equilibrium of turbulent kinetic energy production and dissipation
- constant shear stress within the log-law region

Wall functions are used to modify the contribution of the wall shear stress and wall heat flux

$$\int \tau_{ij} n_j dS = F_{wi}, \qquad \int q_j n_j dS \tag{6.13}$$

The velocity parallel to the wall is used as the velocity for all quantities. It can be calculated by projecting the velocity vector onto the surface plane

$$u_{i\|} = (\delta_{ij} - n_i n_j) u_j \tag{6.14}$$

where n_i is the surface unit normal.

The law of the wall for compressible flow [24] can be written as

$$\frac{U_c}{u_\tau} = U_c^+ = \frac{1}{\kappa}\ln y^+ + C, \tag{6.15}$$

$$u_\tau = \sqrt{\frac{\tau_w}{\rho_w}} = \frac{u_\parallel}{u^+}, \qquad y^+ = \frac{u_\tau y}{\nu_w} \tag{6.16}$$

with $C = 5.1$ and $\kappa = 0.41$ by default. This law of the wall is similar to the incompressible version except velocity is transformed using the Van Driest transformation[23]

$$U_c = \frac{1}{a}\left[\sin^{-1}\left(\frac{2a^2 u_\parallel - b}{Q}\right) + \sin^{-1}\left(\frac{b}{Q}\right)\right] \tag{6.17}$$

where

$$a = \sqrt{\left(\frac{Pr_T}{2C_p T_w}\right)}, \qquad b = \frac{T_{aw} - T_w}{T_w u_e}, \qquad Q = \sqrt{b^2 + 4a^2} \tag{6.18}$$

The adiabatic wall temperature is computed assuming that the recovery factor, r is equal to the turbulent Prandtl number, Pr_T

$$T_{aw} = T_1 + \frac{r u_\parallel^2}{2C_p} \tag{6.19}$$

The wall temperature is set to the adiabatic wall temperature for an adiabatic wall. For an isothermal wall it is set to the specific wall temperature. The quantities with a subscript of 1 are the value at the first point off the wall. In the case of edge-based Aero, this point is actually at the wall and the wall value is at a fictitious location. The wall values of density and viscosity are computed using the wall temperature and assuming that the pressure does not vary between the wall and the first point off the wall. The shear velocity, u_τ is determined by solving equation 6.15 using Newton's method. After u_τ is determined the wall shear stress for each component is

$$\tau_{w,i} = \rho_w u_\tau \frac{u_{i\parallel}}{u^+} \tag{6.20}$$

The calculation for obtaining the wall heat flux follows Huang et al[24]. If near a solid surface convection is neglected and $\tau = \tau_w$ is assumed then the energy equation can be integrated resulting in

$$q = q_w + u_\parallel \tau_w \tag{6.21}$$

The heat transfer and wall shear stress are written as

$$q = -\frac{\mu_T C_p}{Pr_T}\frac{\partial T}{\partial y}, \qquad \tau_w = \mu_t \frac{\partial u_\parallel}{\partial y} \tag{6.22}$$

Integrating the above equation with respect to $u_\parallel$ results in

$$T = T_w - \frac{Pr_t q_w u_\parallel}{C_p \tau_w} - \frac{Pr_T u_\parallel^2}{2C_p} \tag{6.23}$$

By using $r = Pr_T$ and solving for q_w, the above equation can be written

$$q_w = \frac{\tau_w C_p}{r u_{\parallel}} \left(T_w - T_{aw}\right) \tag{6.24}$$

which gives the flux boundary condition on the energy equation at the wall. If the first point is too close to the wall and no longer in the log layer typically for $y^+ < 12$, then the viscous wall shear stress and heat flux are given by:

$$\tau_w = \mu \frac{u_{\parallel}}{y}, \qquad q_w = \kappa \frac{T_w - T_1}{y} \tag{6.25}$$

The turbulence quantity model for the SST model are specified as

$$k_{\text{wall}} = \frac{\frac{\rho_w}{\rho_1} u_\tau^2}{\sqrt{\beta^*}}, \qquad \omega_{\text{wall}} = \frac{u_\tau}{\kappa \sqrt{\beta^*} y}. \tag{6.26}$$

For the k-ϵ model, the turbulence quantities are

$$k_{\text{wall}} = \frac{u_\tau^2}{C_\mu}, \qquad \epsilon_{\text{wall}} = \frac{u_\tau^3}{\kappa y}. \tag{6.27}$$

The working variable for the Spalart-Allmaras model is

$$\hat{\nu}_{\text{wall}} = \kappa u_\tau y. \tag{6.28}$$

The turbulence quantities for the above models are imposed weakly at the wall using equations **??**, **??**, and **??**, respectively.

6.1.4. Slip wall

For slip walls, see the tangent flow boundary conditions described below.

6.2. TANGENT FLOW

A tangent flow condition is typically applied for underresolved or slip walls or as a symmetry condition. In Aero, two methods can be used to apply this condition, one based on reflecting the velocity and one based on a pressure integral. The enforcement of additional required boundary conditions for viscous flows is independent of whether velocity reflection or pressure integration is used.

6.2.1. Velocity Reflection

The primary condition for a tangent flow boundary is that the velocity normal to the boundary is zero. In Aero, this is enforced using a flux through the boundary face. To calculate the flux, first the flow state at the boundary is copied into a boundary state, U_b. The normal velocity of boundary state is then reflected,

$$u_{bi} = u_i - 2u_k n_k n_i. \tag{6.29}$$

The boundary flux follows simply as

$$F_b = F^R(U, U_b, n), \tag{6.30}$$

where F^R is a dissipative reconstruction, such as Roe or Steger-Warming.

6.2.2. Pressure Integration

Another way to impose that the velocity normal to the boundary is zero is to simply construct the boundary flux such that all terms multiplied by the normal velocity are zero. Thus, only the pressure force in the momentum equation is included. For a turbulent ideal gas, the form of the flux is

$$F_b = \int_{\partial\Omega} \begin{pmatrix} 0 \\ Pn_i \\ 0 \\ 0 \\ 0 \end{pmatrix} dA. \tag{6.31}$$

In Aero, a first order approximation to the integral is used, where the pressure value at the boundary node is multiplied by the area of the boundary face.

The two approaches give different results. For a more strict enforcement of zero normal velocity, the reflection condition is preferred.

6.2.3. Viscous Conditions

The Navier-Stokes equations are incompletely parabolic and require $(N_q - 1)$ linearly independent boundary conditions for tangent flow, where

$$N_q = N_e + N_s + N_d + N_T$$

is the number of differential equations in the system, N_e is the number of energy equations, N_s is the number of species, N_d is the number of physical dimensions, and N_T is the number of turbulence equations. In Aero, this requirement is satisfied by

1. Requiring that the viscous traction force has no component parallel to the boundary, and
2. Requiring no species mass diffusion, heat flux, or turbulent diffusion through the boundary.

The first condition yields $(N_d - 1)$ linearly independent conditions, and the second condition yields $(N_s + N_T)$ independent conditions. The velocity reflection or pressure integration condition yields the final needed condition. For a turbulent ideal gas, the boundary viscous flux takes the form

$$G_b = \int_{\partial\Omega} \left(\ 0, \tilde{T}_i, 0, 0, 0 \ \right) dA, \tag{6.32}$$

where $\tilde{T}_i$ is the modified traction vector,

$$\tilde{T}_i = T_n n_i, \quad T_n = \tau_{kj} n_j n_k. \tag{6.33}$$

Note that there is no contribution of the viscous heating to the energy equation because $u_i \tilde{T}_i = 0$.

6.3. OPEN BOUNDARIES

Open boundaries are used to specify inflows, outflows, and farfield conditions. Aero has many methods of specifying open boundaries, which are detailed below. As for tangent flows, the additional needed conditions for viscous flows are always enforced in the same manner when required for outflows and farfields.

The boundary conditions for open boundaries in Aero are set up with an inviscid part, which will be applied to both the Euler and Navier Stokes equations, and a viscous part, which is only applied to the Navier Stokes equations. The number of inviscid boundary conditions required for an open boundary depends on the number of incoming eigenvalues of the flux Jacobian,

$$\frac{\partial F}{\partial U} = S \Lambda S^{-1}, \tag{6.34}$$

where S denotes the matrix of right eigenvectors and Λ denotes the diagonal matrix of eigenvalues,

$$\Lambda = A \begin{pmatrix} u_k n_k + c & 0 & 0 & \dots \\ 0 & u_k n_k - c & 0 & \dots \\ 0 & 0 & u_k n_k & 0 \\ \vdots & \vdots & 0 & \ddots \end{pmatrix}. \tag{6.35}$$

Since the normal vector is oriented to point outward from the domain, incoming eigenvalues will always be negative.

6.3.1. Viscous Conditions

Open boundaries with one or zero nonzero eigenvalues require additional conditions for well-posedness of the viscous equations.

In Aero, this requirement for open boundaries is satisfied by

1. Requiring that the viscous traction force has no component normal to the boundary, and

2. Requiring no species mass diffusion, heat flux, or turbulent diffusion through the boundary.

The first condition yields one linearly independent condition, and the second condition yields $(N_s + N_T)$ independent conditions. Thus, for some inviscid outflow conditions in Aero, the linear well-posedness may not be met. However, in numerical experiments, the boundary conditions used exhibit correct and robust behavior.

For a turbulent ideal gas, the boundary viscous flux takes the form

$$G_b = \int_{\partial\Omega} \left(\ 0, \tilde{T}_i, u_k \tilde{T}_k, 0, 0 \ \right) dA, \tag{6.36}$$

where $\tilde{T}_i$ is the modified traction vector,

$$\tilde{T}_i = \tau_{ij} n_j - T_n n_i, \quad T_n = \tau_{kj} n_j n_k. \tag{6.37}$$

These viscous conditions are applied to all open boundary conditions unless a Dirichlet condition is specified.

6.3.2. Extrapolation

For a flow with no negative eigenvalues, no inviscid condition should be specified. One method to accomplish this is to simply use the flux calculated from the state at the boundary to specify the boundary flux,

$$F_b = F(U). \tag{6.38}$$

For flows that may be transonic at the boundary, this is not an appropriate choice. The extrapolation boundary condition is an instance in the viscous case where not enough data is imposed for linear well-posedness to be satisfied.

6.3.3. Farfield

In Aero, farfield enforcement is a general open boundary condition that automatically handles all combinations of eigenvalues. In the case of a supersonic outflow (strictly positive eigenvalues), the extrapolation boundary condition is recovered.

For a supersonic inflow (strictly negative eigenvalues), the entire boundary state is calculated from a user-specified flow state, and a flux reconstruction is used to specify the boundary flux,

$$F_b = F^R(U, U_b). \tag{6.39}$$

For subsonic boundary regions, the boundary states are calculated based on Riemann invariants,

$$R_s^{(-)} = u_{sj} n_j - \frac{2}{\gamma - 1} c_s, \quad R^{(+)} = u_j n_j + \frac{2}{\gamma - 1} c, \tag{6.40}$$

where the subscript s denotes the user specified state. These Riemann invariants are used to compute the normal velocity as:

$$u_{bj}n_j = \frac{R_s^{(-)} + R^{(+)}}{2}. \tag{6.41}$$

To compute the full velocity tangent velocity vectors are defined for the specified and computed states,

$$\begin{aligned} u_{t_1 i} &= u_j t_{1j} t_{1i}, \quad u_{t_2 i} = u_j t_{2j} t_{2i}, \\ u_{st_1 i} &= u_{sj} t_{1j} t_{1i}, \quad u_{st_2 i} = u_{sj} t_{2j} t_{2i}, \end{aligned} \tag{6.42}$$

where t_1 and t_2 denote vectors that are orthogonal to each other and the normal. The full boundary state velocity is then calculated as

$$u_{bi} = u_{bj}n_j n_i + \frac{1}{2}\left(1 - \frac{u_j n_j}{c}\right)\left(u_{st_1 i} + u_{st_2 i}\right) + \frac{1}{2}\left(1 + \frac{u_j n_j}{c}\right)\left(u_{t_1 i} + u_{t_2 i}\right). \tag{6.43}$$

The Riemann invariants are additionally used to specify the square of the speed of sound:

$$c_b^2 = \frac{(\gamma - 1)^2}{16}\left(R^{(+)} - R_s^{(-)}\right)^2. \tag{6.44}$$

The speed of sound is used to specify the temperature as

$$T_b = \frac{c_b^2}{\gamma R}. \tag{6.45}$$

For a subsonic inflow, the density is

$$\rho_b = \left(\frac{c_b^2}{\gamma S_s}\right)^{\frac{1}{\gamma - 1}}, \quad S_s = \frac{P_s}{\rho_s^\gamma}, \tag{6.46}$$

and for a subsonic outflow, the density is

$$\rho_b = \left(\frac{c_b^2}{\gamma S}\right)^{\frac{1}{\gamma - 1}}, \quad S = \frac{P}{\rho^\gamma}. \tag{6.47}$$

In other words, the entropy is used from the specified state for an inflow and from the computed state for an outflow to specify the density. This fully specifies the boundary state and the boundary flux is computed using a flux reconstruction.

6.3.4. Characteristic Projection

In Aero, three boundary conditions are applied through characteristic projection, which involves a nonlinear solve to compute the boundary state appropriate for each boundary condition. This boundary state is then computed using a flux reconstruction.

For a subsonic outflow specifying only backpressure, the left eigenvectors, S^{-1} for the current boundary state are computed. The difference between the boundary state and the computed state are then transformed to characteristic space,

$$\delta\hat{U} = S^{-1}\left(U - U_b\right). \tag{6.48}$$

The difference corresponding to the incoming eigenvalue is replaced by $P - P_s$. A linear system

$$M\delta U = -\delta\tilde{U}, \tag{6.49}$$

where M is equivalent to the left eigenmatrix, except that the eigenvector corresponding to the incoming eigenvalue is replaced by the Jacobian of the pressure with respect to the boundary state, $\partial P/\partial U_b$. The boundary state is subsequently updated by

$$U_b = U_b + \delta U, \tag{6.50}$$

and the procedure is iterated until the L_1 norm of $\delta\tilde{U}$ is less than a specified tolerance.

For a subsonic inflow specifying velocity and temperature, the same procedure is followed. A subsonic inflow has $N_q - 1$ incoming eigenvalues. Values of $\delta\tilde{U}$ corresponding to the first N_d incoming eigenvalues are replaced by $u_i - u_{si}$ and the corresponding rows in M are replaced by $\partial u_i/\partial U$. Similarly, the difference values and row corresponding to the next incoming eigenvalue are replaced by $T - T_s$ and $\partial T/\partial U$, respectively. For multicomponent flows, the next $N_s - 1$ values would also be specified.

For a subsonic inflow (pressure reservoir) specifying total pressure, total temperature, and the flow direction, we again follow the same procedure. The incoming differences are replaced by $P^0 - P_s^0$, $T^0 - T_s$, u_{vt_1}, and u_{vt_2}, respectively. u_{vt} denotes the velocity tangent to the specified velocity direction, not the velocity tangent to the boundary. The rows corresponding to incoming eigenvalues are replaced by $\partial P^0/\partial U$, $\partial T^0/\partial U$, $\partial u_{vt_1}/\partial U$, and $\partial u_{vt_2}/\partial U$, respectively.

6.4. SUPERSONIC INFLOW

At a supersonic inflow, all inviscid quantities are directly specified and the equations are not computed at the boundary nodes. However, the boundary fluxes are still computed for post-processing.

6.4.1. k-ω turbulence model

At an inflow boundary, the turbulent kinetic energy is typically computed from an estimate of the turbulence intensity, T_u

$$k_\infty = \frac{3}{2}\left(T_u U_\infty\right)^2 \tag{6.51}$$

Typically, $T_u \sim 0.01$. The symbol U_∞ denotes the free stream flow speed. The turbulent kinetic energy specific dissipation rate is specified as

$$\omega_\infty = C_\mu \frac{\rho_\infty k_\infty}{r\mu_\infty} \tag{6.52}$$

where r_t is the specified ratio of the turbulent viscosity to laminar viscosity, and μ_∞ is the free stream value of the laminar viscosity. Typically, $r \sim 0.1$.

6.4.2. k-ϵ turbulence model

At an inflow boundary, the turbulent kinetic energy is typically computed from an estimate of the turbulence intensity, T_u

$$k_\infty = \frac{3}{2}\left(T_u U_\infty\right)^2 \tag{6.53}$$

Typically, $T_u \sim 0.01$. The symbol U_∞ denotes the free stream flow speed. The turbulent energy dissipation rate is specified as

$$\epsilon_\infty = C_\mu \frac{\rho_\infty k_\infty^2}{r \mu_\infty} \tag{6.54}$$

where r_t is the specified ratio of the turbulent viscosity to laminar viscosity, and μ_∞ is the free stream value of the laminar viscosity. Typically, $r \sim 0.1$.

6.4.3. Spalart-Allmaras turbulence model

At an inflow boundary, the working variable is set to a value computed from the incoming values of the kinematic viscosity, namely

$$\hat{\nu}_\infty = 4\frac{\mu_\infty}{\rho_\infty} \tag{6.55}$$

7. ADAPTIVITY

7.1. ERROR INDICATORS

Error indicators are used to mark the mesh for refinement or unrefinement. This section describes the available error indicators in Aero.

7.1.1. Limiter

Limiters are active at shocks and in regions of high gradients where the mesh resolution is insufficient. Both of these properties make the limiter values a good error indicator for marking regions for refinement. One downside to using limiters as error indicators is that they are noisy.

In Aero, an elemental error indicator is needed for refinement. First, for each node the minimal value of the limiter is taken over all of the variables.

$$ei_{nodal} = \min_{\boldsymbol{U}}(\phi) \tag{7.1}$$

For the elemental error indicator, a low value indicates low error and a high value indicates high error. For the limiter-based error indicator, a lower limiter value indicates higher error and the limiter is between zero and 1. Therefore, for the element error indicator, 1 - the minimal limiter value over all nodes is used, i.e.,

$$ei_{element} = 1 - \min_{nodes}(ei_{nodal}) \tag{7.2}$$

7.1.2. High Low Flux

One good measure of the discretization error is the difference between a low order discretization and a higher order discretization of the inviscid flux. The nodal error indicator is a sum over conserved variable flux differences.

$$ei_{nodal} = \sum_j \frac{1}{U^j_{scale}} \int_{\partial\Omega} (F(U^j_H)^{HO} - F(U^j_H)^{LO}) \cdot \hat{\vec{n}}\, dA \tag{7.3}$$

where j denotes the conserved variable indices and U^j_{scale} is a scale factor usually based on the freestream quantities.

The elemental error indicator is simply the sum of the nodal error indicator over all of the element's nodes.

$$ei_{element} = \sum_{nodes} (ei_{nodal}) \tag{7.4}$$

BIBLIOGRAPHY

[1] I.G. Currie. Fundamental Mechanics of Fluids. McGraw-Hill, 1974.

[2] Stephen B. Pope. Turbulent Flows. Cambridge University Press, 2000.

[3] F.R. Menter. Two-equation eddy viscosity turbulence models for engineering applications. AIAA Journal, 32(8), 1994.

[4] D.C. Wilcox. Reassessment of the scale-determining equation for advanced turbulence models. AIAA Journal, 26(11), 1988.

[5] D.C. Wilcox. Formulation of the k-ω turbulence model revisted. AIAA Journal, 46(11), 2008.

[6] Philippe R. Spalart. Detached-eddy simulation. Annual Review of Fluid Mechanics, 41:181–202, 2009.

[7] R.M.C. So, A. Sarkar, G. Gerodimos, and J. Zhang. A dissipation rate equation for low Reynolds number and near wall turbulence. Theoretical and Computational Fluid Dynamics, 1997.

[8] K.W. Brinkman, W.H. Calhoom, and S.M. Dash. Scalar fluctuation modeling for high-speed aeropropulsive flows. AIAA Journal, 45(5):1036–1046, 2007.

[9] Farzad Ismail and Philip L. Roe. Affordable, entropy-consistent Euler flux functions II: Entropy production at shocks. Journal of Computational Physics, 228:5410–5436, 2009.

[10] M.C. Druguet, G.V. Candler, and I. Nompelis. Effects of numerics on navier-stokes computations of hypersonic double-cone flows. AIAA Journal, 43(3):616–623, 2005.

[11] I. Nompelis. Computational study of hypersonic double-cone experiements for code validation. PhD thesis, University of Minnessota, MN, 2004.

[12] Pramod K. Subbareddy and Graham V. Candler. A fully discrete, kinetic energy consistent finite-volume scheme for compressible flows. Journal of Computational Physics, 228:1374–1364, 2009.

[13] Albert E. Honein and Parviz Moin. Higher entropy conservation and numerical stability of compressible turbulence simulations. Journal of Computational Physics, 201:531–545, 2004.

[14] Travis C. Fisher and Mark H. Carpenter. High-order entropy stable finite difference schemes for nonlinear conservation laws: Finite domains. Technical Report TM 217971, NASA, 2013.

[15] Ankit Bhagatwala and Sanjiva K. Lele. A modified artificial viscosity approach for compressible turbulence simulations. Journal of Computational Physics, 228:4965–4969, 2009.

[16] M.H. Carpenter and C.A. Kennedy. Fourth-order 2N-storage Runge-Kutta schemes. NASA TM NASA, NASA Langley Research Center, Hampton, VA, 1994.

[17] Automatic Control and Adaptive Time-Stepping, 2001.

[18] Gustaf Söderlind. Digital filters in adaptive time-stepping. ACM TOMS, 2005.

[19] K. Gustafsson. Control theoretic techniques for stepsize selection in explicit runge-kutta methods. ACM TOMS, (20):496–517, 1994.

[20] C.A. Kennedy and M.H. Carpenter. Additive runge-kutta schemes for convection-diffusion-reaction equations. Applied Numerical Mathematics, (44):139–181, 2003.

[21] M.H. Carpenter. personal communication.

[22] Magnus Svärd and Jan Nordström. A stable high-order finite difference scheme for the compressible Navier-Stokes equations. Journal of Computational Physics, 227:4805–4824, 2008.

[23] Frank M. White. Viscous Fluid Flow. McGraw-Hill, 2006.

[24] P.G. Huang, P. Bradshaw, and T.J. Coakley. Skin friction and velocity profile family for compressible turbulent boundary layers. AIAA Journal, 31(9):1600–1604, 1993.

DISTRIBUTION

Email—Internal [redacted]

Name	Org.	Sandia Email Address
Technical Library	01177	libref@sandia.gov

Sandia National Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.