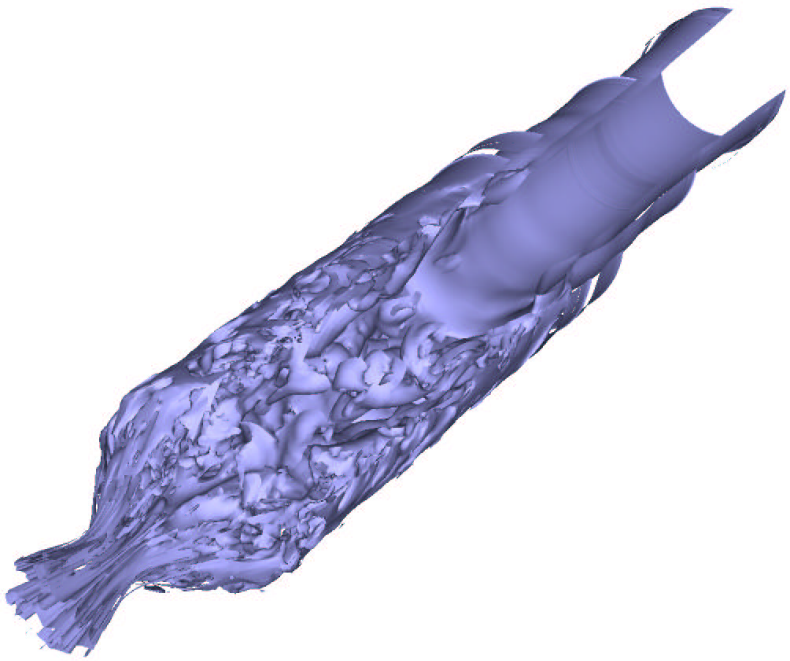
Rocturb Developer's Guide

Bono Wasistho

Center for Simulation of Advanced Rockets
University of Illinois at Urbana-Champaign, Urbana, IL 61801



December 10, 2005

Title:	ROCTURB Developer's Guide
Author:	Bono Wasistho (research scientist)
Subject:	Guide for the theory and implementation of turbulence models and modules for supporting turbulence simulations
Revision:	1
Revision history:	Revision 0: Developer's guide for old Rocflo code Revision 1: Developer's guide for Rocfluid_MP and Rocstar2.5 Revision 2: Developer's guide for current Rocfluid_MP and Rocstar3
Effective date:	12/12/2005

Contents

Chapter 1

Introduction

1.1 Objectives

For computations involving turbulence, direct simulations of pure 3D Navier-Stokes equations in RocfloMP would in many cases be prohibitive due to high resolution requirement. This requirement can be substantially eased with the help of turbulence model because the small scale fluctuations do not have to be resolved, instead they are modeled in terms of the resolved scales on a coarser grid. This Rocturb module is therefore developed in order to provide users with various turbulence models to suit their flow features.

1.2 Model Classes and Application Regimes

We offer users three different classes of turbulence models: full LES (Large Eddy Simulation), RANS (Reynolds Averaged Navier Stokes) and Hybrid model, i.e. LES with near wall model. The near wall model can be based on Rans or turbulent boundary layer profiles in equilibrium or non-equilibrium, depending on the flow pressure gradient. For turbulent flows with thin wall shear layer, one can use LES with wall model, or DES (Detached Eddy Simulation) where the inner and outer layer is more tightly coupled. For a highly unsteady large structure motions away from the wall, such as in wall injection flows, it is recommended to use full LES as the wall requirement is less stringent, while for high Reynolds number flows with weak unsteadiness in the mean and if the interest is mainly on the mean flow, Rans is more suitable due to the less demanding grid requirement. If desired, DES can be used for high Reynolds number flow as well. DES is especially suitable for flows involving massive separation.

1.3 Reference Models Implemented

A) LES class:
1) basic Smagorinsky (Smagorinsky 1963),

2) scale-similarity (Bardina *et al.* 1984),
3) dynamic Smagorinsky (Germano *et al.* 1991 and Germano 1992),
4) dynamic mixed (Zang *et al.* 1993 and Vreman *et al.* 1994).

B) RaNS class:
1) SA (Spalart & Allmaras 1994),
2) $k - \epsilon$ (Jones & Launder 1972 and Launder & Sharma 1974).

C) Hybrid class:
1) DES (Dettached Eddy Simulation) (Nikitin *et al.* 2000),
2) Non-equilibrium wall layer model for LES (developed by Wasistho).
The wall layer model is implemented and currently in the stadium of testing before made available in CVS repository. DES is not available yet.

It is possible to select different models in different zones (zonal modeling). For instance, one can compute LES using a dynamic model in the propellant chamber, but employing the fixed Smagorinsky model in the nozzle to provide more damping along the thin boundary layer (fixed Smagorinsky without Van Driest damping provides finite, non-zero eddy viscosity at the wall)

1.4 Subroutine Organization and Compilation Procedure

There are necessary operations and supporting tools to run turbulence simulation, such as gradient construction for viscous terms and computation of temporal statistics. These procedures are described in the last two chapters of this manual.

All subroutines pertinent to turbulence are located under directory *rocturb* having prefix *TURB_*. All turbulence routines pertinent to LES are named *TURB_Les...*. The same is also applied for RaNS, *TURB_Rans...*, wall layer model, *TURB_Wlm...* and so on. These routines can be shared by Rocflo and Rocflu. Turbulence routines which are Rocflo/Rocflu specific, and do not share commonality between the two, are named *TURB_RFLO_...* and *TURB_RFLU_...*, respectively, if they are public to other directories. If they are private to rocturb, the names are *TURB_Flo...* and *TURB_Flo...*, respectively. Routines which share tight commonality between Rocflo and Rocflu are named *TURB_Co...* (with suffix flo/flu for the file name). This system of nomenclature makes the process of routines update and modifications easier. When developer is to modify a routine of type *TURB_Co...Flo*, he will likely have to modify the *TURB_Co...Flu* counterpart accordingly. All routines with tight commonality between Rocflo and Rocflu will therefore be advanced simultaneously. In similar fashion, when modifying routines which are public to other directories, developer will be more aware to check whether the changes will affect any external routines or not. All gradients routines for structured grid can be found in directory *libflo* under the name

RFLO_CalcGrad.... Most statistics routines are collected in module *ModStatsRoutines* in *modfloflu*. Standalone statistics routines in other directories can be easily identified since they have at least phrase *Stat* as part of their name.

To activate turbulence and statistics module user has to include *TURB=1* and *STATS=1* in the compilation. For instance for structured code, *gmake RFLO=1 TURB=1 STATS=1 MPI=1*

1.5 Manual Organization and Guidelines

The principal equations governing turbulent flows from which all the turbulence models presented in this manual are derived can be found in Chapter 2. The formulation of the various turbulence models for all three classes is given in brief in Chapter 3, including the energy subgrid terms of LES. We discuss the LES implementation details in Chapter 4. The code organization and data structure for LES and RaNS are available in Chapter 5. In Chapter 6 we deal with the construction of the wall layer model for LES in more detail. In this manual we also provide the description of operations or modules required for turbulence simulation, namely the construction of spatial derivatives, in Chapter 7, and the reduction of temporal statistics, in Chapter 8.

Besides the description of code structure in following sections, many relevant equations and procedures are complemented with the routine names in which the procedures are actually programmed. In this way a reader can easily find the piece of code responsible for any specific treatment desribed in this manual.

1.6 Rocturb Verification, Validation, and Application

Rocturb verification, validation, and application cases can be found in the accompanying Rocturb User's Guide and journal papers [16], [17], and [18].

Chapter 2

Governing Equations of Filtered and Averaged Navier-Stokes

In order to enable turbulence simulations with reasonable cost, the system of Navier Stokes (NS) equations are filtered in space, for LES, or averaged in time or realisations (ensemble), for RANS. Favre filtering and averaging is carried out (see Wilcox 1993) as we consider compressible NS. Both filtering for LES and ensemble averaging for RANS leads in the same way to the appearance of additional terms in the NS system.

$$\partial_t \overline{\rho} + \partial_j(\overline{\rho}\widetilde{u}_j) = 0, \tag{2.1}$$

$$\partial_t(\overline{\rho}\widetilde{u}_i) + \partial_j(\overline{\rho}\widetilde{u}_i\widetilde{u}_j) + \partial_i\overline{p} - \partial_j\breve{\sigma}_{ij} = -\partial_j(\overline{\rho}\tau_{ij}), \tag{2.2}$$

$$\partial_t\breve{e} + \partial_j((\breve{e}+\overline{p})\widetilde{u}_j) - \partial_j(\breve{\sigma}_{ij}\widetilde{u}_i - \breve{q}_j) = -\alpha_1 - \alpha_2 - \alpha_3 + \alpha_4 \tag{2.3}$$

$\widetilde{u}$ means Favre filtered or Favre averaged of u, while $\overline{\rho}$ is standard filtered or Reynolds averaged of ρ. This holds for other variables as well. In the energy equation $\breve{e}$ is the total energy, i.e. the sum of internal energy and kinetic energy, that can be expressed in terms of the filtered or averaged variables. Note, in this context filtering is for LES and averaging for RANS. The right hand side terms are the additional terms due to the filtering or averaging and have to be modeled to close the system of equations. The description of the above equations without the additional terms can be found in RocfloMP Developer's guide, therefore we mainly focus on the 'turbulence' terms. These turbulence terms are called Reynolds terms in RANS and subgrid terms in LES. Modeling these terms doesn't need explicit averaging in RANS. In LES, however, explicit filtering is sometimes required, depending on the subgrid model used. The detail of the filtering procedure in LES is decribed in Chapter 4: Implementation Details. Next, we present the expression of the turbulence terms, first in LES then in RANS.

2.1 Subgrid Terms in LES

In the context of LES the momentum subgrid or turbulent stress term in equation 2.2 is

$$\overline{\rho}\tau_{ij} = \overline{\rho u_i u_j} - \overline{\rho u_i \rho u_j}/\overline{\rho} = \overline{\rho}(\widetilde{u_i u_j} - \widetilde{u_i}\widetilde{u_j}) \tag{2.4}$$

The energy subgrid terms in equation 2.3 can be written as

$$\alpha_1 = \widetilde{u_i}\partial_j(\overline{\rho}\tau_i j), \tag{2.5}$$
$$\alpha_2 = \partial_j(\overline{\rho u_j} - \overline{\rho}\widetilde{u}_j)/(\gamma - 1), \tag{2.6}$$
$$\alpha_3 = \overline{p\partial_j u_j} - \overline{p}\partial_j\widetilde{u}_j, \tag{2.7}$$
$$\alpha_4 = \overline{\sigma_{ij}\partial_j u_i} - \overline{\sigma}_{ij}\partial_j\widetilde{u}_i. \tag{2.8}$$

α_1 is the turbulent stress term, representing the kinetic energy transfer from resolved to subrid scales. α_2 is the pressure velocity subgrid term. It represents the effect of the subgrid turbulence on the heat conduction in the resolved scales. α_3 is the pressure dilatation term, which is purely a compressibility effect. It vanishes in the incompressible limit. The subrid scale turbulent dissipation rate α_4 is the amount subgrid kinetic energy converted into internal energy by viscous dissipation.

Besides those terms above, there are more terms in both momentum and energy equations which result from the fact the Favre filter is not commutative with partial derivatives. These terms are, however, less important that they are neglected.

2.2 Reynolds Terms in RANS

System of equations 2.1 - 2.3, can be seen as compressible RANS equations through the following decomposition before applying time or ensemble averaging to the NS equations.

$$\begin{aligned} u_i &= \widetilde{u}_i + u_i'' \\ \rho &= \overline{\rho}_i + \rho' \\ p &= \overline{p} + p' \\ h &= \widetilde{h} + h'' \\ e &= \breve{e} + e'' \\ T &= \widetilde{T} + T'' \\ q_j &= \breve{q}_j + q_j'' \end{aligned} \tag{2.9}$$

$\overline{(.)}$ and $\widetilde{(.)}$ imply Reynolds and Favre averaged variables, respectively, while $\breve{(.)}$ quantities which are functions of these averaged variables. $(.)''$ and $(.)'$ are the fluctuations of Favre and Reynolds averaged variables, respectively. The turbulence terms in the momentum and

energy equations are expressed in these quantities. In the context of RANS the Reynolds stress or turbulent stress term in equation 2.2 is expressed as

$$\overline{\rho}\tau_{ij} = \overline{\rho u_i'' u_j''} \tag{2.10}$$

The turbulence terms in energy equation 2.3 are given by

$$\alpha_1 = \widetilde{u_i}\partial_j(\overline{\rho}\tau_i j), \tag{2.11}$$
$$\alpha_2 = \partial_j(\overline{p' u_j''})/(\gamma - 1), \tag{2.12}$$
$$\alpha_3 = \overline{p'\partial_j u_j''}, \tag{2.13}$$
$$\alpha_4 = \overline{\sigma_{ij}\partial_j u_i''}. \tag{2.14}$$

Note that these turbulence energy terms are similar to those in the LES context, equations 2.5 - 2.8. They also have the same meaning. Resolved and subgrid scales in LES is analogous to the mean and turbulent scales in RANS. The difference is that the turbulent scales in RANS have a much wider spectrum than the subgrid scales in LES. So, α_1 in RANS context represents the kinetic energy transfer from mean flow to all scales turbulence.

Chapter 3

Models for Turbulence Terms in Momentum Equations

The task of turbulence models is to model the turbulence terms, $\overline{\rho}\tau_{ij}$ in the momentum equations and $\alpha_1 - \alpha_4$ in the energy equation, presented in Chapter 2. For eddy viscosity type of models, the turbulent stress term can be written as $\overline{\rho}\tau_{ij} = \mu_t S_{ij}$ and it reduces to modeling of the eddy viscosity μ_t. In other type of models, the turbulent stress term is modeled as a whole and μ_t is the derived from $\mu_t = \max(0, \tau_{ij}S_{ij}/S_{ij}^2)$. We use unified models for the energy turbulence terms $\alpha_1 - \alpha_4$; the energy models are the same for all turbulent stress models.

3.1 LES Fixed Smagorinsky Model

Smagorinsky model is an eddy viscosity type LES model (Smagorinsky 1963),

$$\overline{\rho}\tau_{ij} = -\mu_t S_{ij}. \qquad [TURB\_CoViscousFluxes] \tag{3.1}$$

The eddy viscosity is formulated as

$$\mu_t = \overline{\rho} C_s^2 \Delta^2 |S(\overline{\mathbf{u}})| \quad \text{with} \quad |S(\overline{\mathbf{u}})|^2 = \frac{1}{2} S_{ij}(\overline{\mathbf{u}}) S_{ij}(\overline{\mathbf{u}}) \qquad [TURB\_LesCalcEddyVis] \tag{3.2}$$

where C_s is the model coefficient which ranges between 0.1 and 0.2, depending on the case being simulated. Δ is the filter width, which is equal to the local grid spacing for Smagorinsky model. The local grid spacing is defined as $\Omega^{\frac{1}{3}}$ where Ω is the cell volume. S_{ij} is defined as

$$S_{ij} = \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \frac{2}{3}\frac{\partial u_k}{\partial x_k}\delta_{ij}. \qquad [TURB\_CalcStrainRate] \tag{3.3}$$

The differential operations in the computation of S_{ij} follows the same finite volume method used in the main system of equations.

3.2 LES Scale Similarity Model

Scale similarity model is proposed by Bardina *et al.* (1984),

$$\overline{\rho}\tau_{ij} = \overline{(\overline{\rho u_i}\ \overline{\rho u_j}/\overline{\rho})} - \overline{\overline{\rho u_i}}\ \overline{\overline{\rho u_j}}/\overline{\overline{\rho}} \qquad [TURB\_LesLij] \tag{3.4}$$

3.3 LES Dynamic Eddy Viscosity Model

Dynamic eddy viscosity model is the dynamic variant of Smagorinsky model, hence also called dynamic Smagorinsky model. In Smagorinsky model, C_s is constant. In the dynamic model C_s is replaced by C_d and determined dynamically (Germano *et al.* 1991 and Germano 1992).

$$\overline{\rho}\tau_{ij} = -\overline{\rho}C_d\Delta^2|S(\widetilde{\mathbf{u}})|S_{ij}(\widetilde{\mathbf{u}}) \tag{3.5}$$

$$Cd = \frac{< M_{ij}L_{ij} >}{< M_{ij}M_{ij} >} \qquad [TURB\_LesContract,\ \ TURB\_LesCoefDynSmag] \tag{3.6}$$

$$L_{ij} = \widehat{(\overline{\rho u_i}\ \overline{\rho u_j}/\overline{\rho})} - \widehat{\overline{\rho u_i}}\ \widehat{\overline{\rho u_j}}/\widehat{\overline{\rho}} \qquad [TURB\_LesLij] \tag{3.7}$$

$$M_{ij} = -\widehat{\overline{\rho}}(\kappa\Delta)^2|S(\mathbf{v})|S_{ij}(\mathbf{v}) + \widehat{(\overline{\rho}\Delta^2|S(\widetilde{u})|S_{ij}(\widetilde{u}))} \qquad [TURB\_LesMij] \tag{3.8}$$

$$v_i = \widehat{\overline{\rho u_i}}/\widehat{\overline{\rho}} \quad \text{and} \quad |S(\mathbf{v})|^2 = \frac{1}{2}S_{ij}^2(\mathbf{v}) \qquad [TURB\_LesTestRhoV] \tag{3.9}$$

3.4 LES Dynamic Mixed Model

Dynamic mixed model is a combination of dynamic eddy viscosity and scale similarity model. This model has been introduced by Zang *et al.* (1993), and modified by Vreman *et al.* (1994). This model employs the sum of the similarity and dynamic eddy viscosity model as base model,

$$\overline{\rho}\tau_{ij} = \overline{(\overline{\rho u_i}\ \overline{\rho u_j}/\overline{\rho})} - \overline{\overline{\rho u_i}}\ \overline{\overline{\rho u_j}}/\overline{\overline{\rho}} - \overline{\rho}C_d\Delta^2|S(\widetilde{\mathbf{u}})|S_{ij}(\widetilde{\mathbf{u}}) \qquad [TURB\_CoViscousFluxes] \tag{3.10}$$

$$Cd = \frac{< M_{ij}(L_{ij} - H_{ij}) >}{< M_{ij}M_{ij} >} \quad [TURB\_LesContract,\ \ TURB\_LesCoefDynMixd] \tag{3.11}$$

$$L_{ij} = \widehat{(\overline{\rho u_i}\ \overline{\rho u_j}/\overline{\rho})} - \widehat{\overline{\rho u_i}}\ \widehat{\overline{\rho u_j}}/\widehat{\overline{\rho}} \qquad [TURB\_LesLij] \tag{3.12}$$

$$H_{ij} = \widehat{\overline{(\widehat{\overline{\rho u_i}}\ \widehat{\overline{\rho u_j}}/\widehat{\overline{\rho}})}} - \widehat{\widehat{\overline{\rho u_i}}}\ \widehat{\widehat{\overline{\rho u_j}}}/\widehat{\widehat{\overline{\rho}}} - \widehat{(\overline{(\overline{\rho u_i}\ \overline{\rho u_j}/\overline{\rho})} - \overline{\overline{\rho u_i}}\ \overline{\overline{\rho u_j}}/\overline{\overline{\rho}})}. \qquad [TURB\_LesHij] \tag{3.13}$$

$$M_{ij} = -\widehat{\overline{\rho}}(\kappa\Delta)^2|S(\mathbf{v})|S_{ij}(\mathbf{v}) + \widehat{(\overline{\rho}\Delta^2|S(\widetilde{u})|S_{ij}(\widetilde{u}))} \qquad [TURB\_LesMij] \tag{3.14}$$

$$v_i = \widehat{\overline{\rho u_i}}/\widehat{\overline{\rho}} \quad \text{and} \quad |S(\mathbf{v})|^2 = \frac{1}{2}S_{ij}^2(\mathbf{v}) \qquad [TURB\_LesTestRhoV] \tag{3.15}$$

3.5 LES Subgrid Terms in Energy Equation

In this section we present simple models for the turbulence terms $\alpha_1 - \alpha_4$ in the energy equation. These energy models, if activated during simulation, are the same for all turbulence models in the momentum equations.

3.5.1 Kinetic energy transfer

The transfer of kinetic energy from resolved to subgrid scales, or in RANS context from the mean flow to turbulent fluctuations, is represented by

$$\alpha_1 = \widetilde{u_i}\partial_j(\overline{\rho}\tau_{ij}). \qquad [TURB\_VisFluxEddy, \quad TURB\_VFluxHybrid] \tag{3.16}$$

All variables at the right hand side are known as part of the solution. Therefore α_1 is computed directly, needs no model.

3.5.2 Heat transport and pressure dilatation

An eddy diffusivity model is applied to model the sum $\alpha_2 + \alpha_3$. α_2 (equation 2.6 and 2.12) represents the effect of turbulence on the conduction of heat in the resolved scales. α_3 (equation 2.7 and 2.13) is the pressure dilatation term, which is purely a compressibility effect. The model reads

$$\alpha_2 + \alpha_3 = -\partial_j(\frac{\mu_t C_p}{Pr_t}\partial_j\widetilde{T}). \quad [TURB\_VisFluxEddy, \quad TURB\_VFluxHybrid] \tag{3.17}$$

This eddy diffusivity model is similar to the molecular heat flux term, but the molecular viscosity and Prandtl number have been replaced by the eddy viscosity, μ_t, and the turbulent Prandtl number, Pr_t. C_p is the specific heat.

3.5.3 Dissipation rate

The dissipation rate α_4 (equation 2.8 and 2.14), which measures the amount of turbulence (subgrid) kinetic energy dissipated into internal energy, is modeled as follow,

$$\alpha_4 = C_\epsilon\overline{\rho}\frac{k^{3/2}}{\Delta}. \qquad [TURB\_LesEsgModel4] \tag{3.18}$$

C_ϵ is a dynamic coefficient which is assumed to be a function of time only

$$C_\epsilon = \frac{\int \alpha_1 d\mathbf{x}}{\int (\overline{\rho}k^{3/2}/\Delta)d\mathbf{x}} \qquad [TURB\_LesEsgModel4] \tag{3.19}$$

with the integration operation spans the whole flow domain. Δ is the filter width, a ratio of the local grid spacing. In the case eddy viscosity models we follow Yoshizawa model (1986)

for the turbulence kinetic energy k,

$$\overline{\rho}k = \overline{\rho}C_k\Delta^2 S_{ij}^2/2, \qquad [TURB\_VisFluxEddy,\ \ TURB\_VFluxHybrid] \tag{3.20}$$

$$C_k \geq \frac{1}{2}\sqrt{3}C_s^2 \qquad \text{for basic Smagorinsky,} \tag{3.21}$$

$$C_k \geq \frac{1}{2}\sqrt{3}C_d \qquad \text{for dynamic models.} \tag{3.22}$$

For non-eddy-viscosity models, $\overline{\rho}k = 1/2\overline{\rho}\tau_{ii}$, i.e the trace of turbulent stress tensor. In the implementation we apply $C_k = 2C_s^2$ and $C_k = 2C_d$, respectively.

3.6 RANS Spalart-Allmaras Model

SA model is a one-equation eddy viscosity model by Spalart & Allmaras (1994). It is originally developed for aerodynamic flow environment. Following the eddy viscosity principle,

$$\overline{\rho}\tau_{ij} = \mu_t S_{ij} \qquad [TURB\_CoViscousFluxes]. \tag{3.23}$$

In this model we solve an transport equation for μ_t [TURB_RansSA...], [TURB_FloRansSA...],

$$\partial_t\tilde{\mu} + \partial_j(u_j\tilde{\mu}) = C_{b1}\tilde{\mu}_t\tilde{\Omega} - c_{w1}f_w\left(\frac{\tilde{\mu}}{d}\right)^2 + \frac{c_{b2}}{\sigma}(\partial_j\tilde{\mu})^2 + \frac{1}{\sigma}\partial_j\left((\tilde{\mu}+\mu)\partial_j\tilde{\mu}\right) \tag{3.24}$$

$$\mu_t = \tilde{\mu}f_{v1}, \quad f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \quad \chi = \frac{\tilde{\mu}}{\mu}. \tag{3.25}$$

The production term $\tilde{\Omega}$ is given by

$$\tilde{\Omega} = \Omega + \frac{\tilde{\mu}}{\kappa^2 d^2}f_{v2}, \quad f_{v2} = 1 - \frac{\chi}{1+\chi f_{v1}}, \tag{3.26}$$

with Ω being the absolute value of the vorticity. The blending function f_w reads

$$f_w = g\left[\frac{1+c_{w3}^6}{g^6 + c_{w3}^6}\right]^{1/6}, \quad g = r + c_{w2}(r^6 - r), \tag{3.27}$$

$$r = \min\left(\frac{\tilde{\mu}}{\tilde{\Omega}\kappa^2 d^2}, 10\right). \tag{3.28}$$

The coefficents are as follows

$$c_{b1} = 0.135, \quad \sigma = 2/3, \quad c_{b2} = 0.622, \quad \kappa = 0.41, \tag{3.29}$$

$$c_{w1} = \frac{c_{b1}}{\kappa^2} + \frac{1+c_{b2}}{\sigma}, \quad c_{w2} = 0.3, c_{w3} = 2, \quad c_{v1} = 7.1.$$

3.7 Hybrid: Wall Layer Model for LES

In wall modeled LES the region $0 \leq y^+ \leq 25$ is modeled using approximate boundary conditions in which the instantaneous wall shear stress is correlated with the LES velocity near the wall, i.e. at the first layer of grid points away from the wall as described by Piomelly and Balaras (2002). Detailed formulation and implementation of the LES wall model are presented in Chapter 6.

3.8 Hybrid: DES (Dettached Eddy Simulation)

DES is derived originally from SA model (Spalart & Allmaras 1994). The SA model is described above. The DES modification with respect to SA concerns the destruction term, and hinges on the length scales d and $\widetilde{d}$. In SA, d is the distance to the nearest wall and expresses the confinement of the eddies by that wall. In DES, d is replaced by $\widetilde{d}$, which is defined as

$$\widetilde{d} = \min(d, C_{DES}\Delta) \qquad \text{with} \quad \Delta = \max(\Delta x, \Delta y, \Delta z) \qquad [TURB\_CoRansWallDistCoV3.0] \tag{3.30}$$

The role of Δ is to allow the energy cascade down to the grid size, in other words makes the pseudo Kolmogorov length scale, based on the eddy viscosity, proportional to the grid spacing. The largest dimension of the grid cell is used, in contrast with the cube-root definition of Δ.

Chapter 4

LES Implementation Details

4.1 Structured Filtering Approach

The filtered $\bar{f}$ is defined as follows

$$\overline{f}(\mathbf{x}) = \int_{\Omega} G(\mathbf{x},\xi) f(\xi) d\xi, \tag{4.1}$$

where $\mathbf{x}$ and ξ are space vectors in the flow domain Ω. For the filter kelner G we use top hat filter:

$$G = \frac{1}{\Delta_1 \Delta_2 \Delta_3} \qquad \text{if} \quad |x_i - \xi_i| < \Delta_i/2, \tag{4.2}$$

$$= 0 \qquad \text{otherwise} \tag{4.3}$$

Applying to the filtered $\bar{f}$,

$$\overline{f}(\mathbf{x}) = \frac{1}{\Delta_1 \Delta_2 \Delta_3} \int\int\int f(\mathbf{x}) dx_1 dx_2 dx_3 \tag{4.4}$$

$$= \frac{1}{\Delta_3}\left[\int \frac{1}{\Delta_2}\left[\int \frac{1}{\Delta_1}\int f(\mathbf{x}) dx_1\right] dx_2\right] dx_3. \tag{4.5}$$

The three dimensional filtering is hence composed from three subsquent one dimensional filtering in each direction. Δ_i denotes the filter width in the i-direction. Δ_i for the base filter, denoted by $\overline{(.)}$, can be equal to or twice the local grid spacing in the corresponding direction. Δ_i for the test filter, denoted by $\widehat{(.)}$, is by definition twice the base filter width. Hence, for base filter width twice the local grid spacing the test filter width is four times the local spacing. The averaged filter width Δ is defined as

$$\Delta = (\Delta_1 \Delta_2 \Delta_3)^{1/3}, \quad \text{or} \tag{4.6}$$

$$\Delta = (\Delta_1^2 + \Delta_2^2 + \Delta_3^2)^{1/2}. \tag{4.7}$$

The first is cube-root formula while the second cell-diagonal formula. It is a user input option. This averaged filter width is used in the formulation of LES models in Chapter 3.

The discrete filtering in the i-direction is carried out using the following rule. Uniform filtering, i.e. filtering with the assumption that the grid is uniform, is carried out using trapezoidal integration rule. Non-uniform filtering, i.e. taking account the actual grid nonuniformity, is carried out using trapezoidal rule in non-homogeneous directions but ensemble averaging in homogeneous directions. The integration coefficients for non-uniform filtering are space dependent. For uniform filtering they are constant,

$$\text{for} \quad \Delta = dx: \quad \frac{1}{\Delta}\int f dx = \frac{1}{8}f_{i-1} + \frac{6}{8}f_i + \frac{1}{8}f_{i+1}, \tag{4.8}$$

$$\text{for} \quad \Delta = 2dx: \quad \frac{1}{\Delta}\int f dx = \frac{1}{4}f_{i-1} + \frac{1}{2}f_i + \frac{1}{4}f_{i+1}, \tag{4.9}$$

$$\text{for} \quad \Delta = 4dx: \quad \frac{1}{\Delta}\int f dx = \frac{1}{8}(f_{i-2} + f_{i+2}) + \frac{1}{4}(f_{i-1} + f_i + f_{i+1}). \tag{4.10}$$

The bar (basic) filtered variables are not explicitly filtered but considered as the resolved solution. Only the test filter variables (denoted by hat above the variable) requires explicit filtering. All the test filtered quantities are cell-face quantities. Only the Favre filtered velocities, $v_i = \widehat{\overline{\rho u_i}}/\widehat{\overline{\rho}}$, in equations 3.9 and 3.15 are cell center quantities. Hence face-to-face and cell-to-cell filterings are used as we define the filter width explicitly in terms of local grid spacing, not from a transfer function analysis as usually done in unstructured grid.

Subroutines pertinent to filter operation:
Uniform cell-to-cell filtering:
TURB_FloLesUniFiltCC
TURB_FloLesUniFiltCCI
TURB_FloLesUniFiltCCJ
TURB_FloLesUniFiltCCK
Uniform face-to-face filtering:
TURB_FloLesUniFiltFF
TURB_FloLesUniFiltFFI
TURB_FloLesUniFiltFFJ
TURB_FloLesUniFiltFFK
General (non-uniform) cell-to-cell filtering:
TURB_FloLesGenFiltCC
TURB_FloLesGenFiltCCI
TURB_FloLesGenFiltCCJ
TURB_FloLesGenFiltCCK
General (non-uniform) face-to-face filtering:
TURB_FloLesGenFiltFF
TURB_FloLesGenFiltFFI
TURB_FloLesGenFiltFFJ
TURB_FloLesGenFiltFFK
Non-equidistant cell-to-face averaging:
TURB_FloLesGenC2F, with its coefficients:

Subroutines pertinent to general (non-uniform) filter coefficients:
Cell-to-cell filter coefficients:
TURB_FloLesGenCoCC
TURB_FloLesGenCoCCUtil
Face-to-face filter coefficients:
TURB_FloLesGenCoFF
TURB_FloLesGenCoFFUtil
TURB_FloLesGenCoFCUtil

The construction of the general (non-uniform) filter coefficients makes use of the following routines:
TURB_FloFaceVolume
TURB_FloFaceWidth
TURB_FloFaceWidthDummy
TURB_FloFaceWidthDummyConn
TURB_FloFaceWidthDummyPhys

The computation of non-uniform filter coefficients is performed only in the initial stage of the run and every time the grid moves.

4.2 Treatment at Block Boundaries

At block boundaries the integration procedure doesn't alter to one-sided integration. It uses the central integration rule by defining dummy values outside the block boundaries. The dummy values are set in such away that they satisfy the finite volume discretisation method.

Pertinent routines: *TURB_FloFaceWidthDummyConn, TURB_FloFaceWidthDummyPhys*

4.3 Averaging in LES

Dynamic models in LES involve space averaging of certain terms, such as $< M_{ij}L_{ij} > / < M_{ij}M_{ij} >$ with $< . >$ the averaging operator (see dynamic eddy viscosity and dynamic mixed models in Chapter 3). In homogeneous directions this averaging procedure is carried out by ensemble averaging, assuming periodicity. In non-homogeneous direction, local filtering, as described in the previous section, is used.

Pertinent routines: *TURB_FloLesAverageFace*

Chapter 5

Code Organization of Turbulence Models

5.1 Code Structure

The main task of *rocturb* is to provide turbulent contribution of the viscous flux to the main code and to deliver eddy viscosity and eddy diffusivity to other modules. The delivery of these quantities to the main code goes through a single routine, *TURB_CoViscousFluxes*, which the driver of the turbulence module.

In this subroutine, four main steps are carried out.

1. Strain rate tensor is computed at faces [*TURB_CalcStrainRate*]
2. Depending on the turbulence model selected, eddy viscosity type (EVM) or turbulent stress type (TSM) regardless the model class, eddy viscosity or turbulent stress tensor is obtained at faces. In turn, if one is available the other can be derived from it by multiplication or derivation with the strain rate tensor.
3. Viscous fluxes are obtained from the resulting turbulent stress tensor. This performed in [*TURB_VisFluxEddy*] for EV models, or [*TURB_VFluxHybrid*] for TS models.
4. Cell center transport variables are obtained by interpolation of the face defined eddy viscosity and diffusivity [*TURB_GetTvCell*] to be used by the main code (for time step computation) and other modules.

In step 2 above, the EV type of models are

- Spalart Almaras (RaNS), [*RFLO_ViscousFlux*]
- fixed Smagorinsky (LES), [*TURB_LesFluxFixSmag*]
- dynamic Smagorinsky (LES), [*TURB_LesGetEddyVis*]

- DES - Detached Eddy Simulation, [*RFLO_ViscousFlux*]

whereas the TS type of models are

- scale similarity (LES), [*TURB_LesFluxScalSim*]
- dynamic mixed (LES), [*TURB_LesGetEddyVis*].

Computations of model coefficients, Leonard tensor, filtering, etc. are hidden under these four routines. Those computations are performed by calling the routines reported in the previous chapters, which correspond to specific equations or procedures.

5.2 Data Structures

The data structures specific to LES and RaNS and common to both LES-RANS, and those for turbulence statistics can be found in the module file *ModTurbulence* in directory *modfloflu*. The data pertinent to rocturb user input variables and variables which have only single value per region are declared under type *t_turb_input*. Turbulence field variables are declared under type *t_turb*. Data structure for wall layer model (wlm) makes use of the existing patch data structure in the module file *ModBndPatch*. Specifically wlm user input switches are defined under *patch%valTurb%switches*, while surface data of wlm, such as wall stresses and flow information at first point above the wall, is stored in *patch%valTurb%vals*

Chapter 6

Wall Layer Model for LES

6.1 Hierarchy of Physical Models

Adequate physical models are necessary to describe thin turbulent boundary layer along accelerated flow such as in the nozzle. Proposed physical models ordered from a strongly coupled (between inner and outer layer) model to the simplest model:

- DES (needs resolved grid, but only in wall normal direction),
- LES with WLM, where $\langle \tau_w \rangle$ comes from separate RANS,
- LES with WLM, where $\langle \tau_w \rangle$ comes from theoretical/empirical correlations.

The formulation of DES is compatible to SA model as described in Chapter 3 and will be outlined in more detail when the actual implementation is carried out. In this chapter we focus on the wall layer model for LES.

6.2 Formulation of WLM for LES

In body fitted coordinate ($\xi,\ \eta,\ \zeta$):

$$\tau_{\eta\xi,w} = \frac{\langle \tau_w \rangle}{\langle V_s(\xi, \eta_1, \zeta) \rangle} v_s^{\xi}(\xi, \eta_1, \zeta) \quad [TURB\_FloWlmUpdateLogLay] \tag{6.1}$$

$$\tau_{\eta\zeta,w} = \frac{\langle \tau_w \rangle}{\langle V_s(\xi, \eta_1, \zeta) \rangle} v_s^{\zeta}(\xi, \eta_1, \zeta) \quad [TURB\_FloWlmUpdateLogLay] \tag{6.2}$$

where $\langle \cdot \rangle$ denotes averaging over homogeneous directions and η_1 is the first point in the direction normal to the wall. $\langle V_s \rangle$ is the averaged streamwise velocity over the homogeneous direction at the first point above the wall, while v_s^{ξ} and v_s^{ζ} are its instantaneous components in the wall parallel directions . $\langle \tau_w \rangle$ as function of $\langle V_s \rangle$ is obtained from distribution (separate RANS) or from correlations (Wasistho, private communication). Mapping resulting

wall stresses to cartesian coordinate reads:

1^{st}Step: $[TURB\_FloWlmMetric, \quad TURB\_WlmTauWallMapping]$

$$\tau_{\xi x} = \frac{S_{\xi x}}{|S_\xi|}\tau_{\xi\xi} + \frac{S_{\eta x}}{|S_\eta|}\tau_{\xi\eta} + \frac{S_{\zeta x}}{|S_\zeta|}\tau_{\xi\zeta} \tag{6.3}$$

$$\tau_{\xi y} = \frac{S_{\xi y}}{|S_\xi|}\tau_{\xi\xi} + \frac{S_{\eta y}}{|S_\eta|}\tau_{\xi\eta} + \frac{S_{\zeta y}}{|S_\zeta|}\tau_{\xi\zeta} \tag{6.4}$$

$$\tau_{\xi z} = \frac{S_{\xi z}}{|S_\xi|}\tau_{\xi\xi} + \frac{S_{\eta z}}{|S_\eta|}\tau_{\xi\eta} + \frac{S_{\zeta z}}{|S_\zeta|}\tau_{\xi\zeta} \tag{6.5}$$

$$\tau_{\eta x} = \frac{S_{\xi x}}{|S_\xi|}\tau_{\eta\xi} + \frac{S_{\eta x}}{|S_\eta|}\tau_{\eta\eta} + \frac{S_{\zeta x}}{|S_\zeta|}\tau_{\eta\zeta} \tag{6.6}$$

$$\tau_{\eta y} = \frac{S_{\xi y}}{|S_\xi|}\tau_{\eta\xi} + \frac{S_{\eta y}}{|S_\eta|}\tau_{\eta\eta} + \frac{S_{\zeta y}}{|S_\zeta|}\tau_{\eta\zeta} \tag{6.7}$$

$$\tau_{\eta z} = \frac{S_{\xi z}}{|S_\xi|}\tau_{\eta\xi} + \frac{S_{\eta z}}{|S_\eta|}\tau_{\eta\eta} + \frac{S_{\zeta z}}{|S_\zeta|}\tau_{\eta\zeta} \tag{6.8}$$

$$\tau_{\zeta x} = \frac{S_{\xi x}}{|S_\xi|}\tau_{\zeta\xi} + \frac{S_{\eta x}}{|S_\eta|}\tau_{\zeta\eta} + \frac{S_{\zeta x}}{|S_\zeta|}\tau_{\zeta\zeta} \tag{6.9}$$

$$\tau_{\zeta y} = \frac{S_{\xi y}}{|S_\xi|}\tau_{\zeta\xi} + \frac{S_{\eta y}}{|S_\eta|}\tau_{\zeta\eta} + \frac{S_{\zeta y}}{|S_\zeta|}\tau_{\zeta\zeta} \tag{6.10}$$

$$\tau_{\zeta z} = \frac{S_{\xi z}}{|S_\xi|}\tau_{\zeta\xi} + \frac{S_{\eta z}}{|S_\eta|}\tau_{\zeta\eta} + \frac{S_{\zeta z}}{|S_\zeta|}\tau_{\zeta\zeta} \tag{6.11}$$

2^{nd}Step: $[TURB\_FloWlmMetric, \quad TURB\_WlmTauWallMapping]$

$$\tau_{ux} = \frac{S_{\xi x}}{|S_\xi|}\tau_{\xi x} + \frac{S_{\eta x}}{|S_\eta|}\tau_{\eta x} + \frac{S_{\zeta x}}{|S_\zeta|}\tau_{\zeta x} \tag{6.12}$$

$$\tau_{uy} = \frac{S_{\xi x}}{|S_\xi|}\tau_{\xi y} + \frac{S_{\eta x}}{|S_\eta|}\tau_{\eta y} + \frac{S_{\zeta x}}{|S_\zeta|}\tau_{\zeta y} \tag{6.13}$$

$$\tau_{uz} = \frac{S_{\xi x}}{|S_\xi|}\tau_{\xi z} + \frac{S_{\eta x}}{|S_\eta|}\tau_{\eta z} + \frac{S_{\zeta x}}{|S_\zeta|}\tau_{\zeta z} \tag{6.14}$$

$$\tau_{vx} = \frac{S_{\xi y}}{|S_\xi|}\tau_{\xi x} + \frac{S_{\eta y}}{|S_\eta|}\tau_{\eta x} + \frac{S_{\zeta y}}{|S_\zeta|}\tau_{\zeta x} \tag{6.15}$$

$$\tau_{vy} = \frac{S_{\xi y}}{|S_\xi|}\tau_{\xi y} + \frac{S_{\eta y}}{|S_\eta|}\tau_{\eta y} + \frac{S_{\zeta y}}{|S_\zeta|}\tau_{\zeta y} \tag{6.16}$$

$$\tau_{vz} = \frac{S_{\xi y}}{|S_\xi|}\tau_{\xi z} + \frac{S_{\eta y}}{|S_\eta|}\tau_{\eta z} + \frac{S_{\zeta y}}{|S_\zeta|}\tau_{\zeta z} \tag{6.17}$$

$$\tau_{wx} = \frac{S_{\xi z}}{|S_\xi|}\tau_{\xi x} + \frac{S_{\eta z}}{|S_\eta|}\tau_{\eta x} + \frac{S_{\zeta z}}{|S_\zeta|}\tau_{\zeta x} \tag{6.18}$$

$$\tau_{wy} = \frac{S_{\xi z}}{|S_\xi|}\tau_{\xi y} + \frac{S_{\eta z}}{|S_\eta|}\tau_{\eta y} + \frac{S_{\zeta z}}{|S_\zeta|}\tau_{\zeta y} \tag{6.19}$$

$$\tau_{wz} = \frac{S_{\xi z}}{|S_\xi|}\tau_{\xi z} + \frac{S_{\eta z}}{|S_\eta|}\tau_{\eta z} + \frac{S_{\zeta z}}{|S_\zeta|}\tau_{\zeta z} \tag{6.20}$$

6.3 Assumptions

The LES WLM is built on the following assumptions:

- grid approaches the wall in more or less normal direction,
- there is at least one homogeneous direction in the region where wlm is applied.

This assumption is likely satisfied in most rocket flow configurations, especially in the nozzle.

6.4 Log-law Based Wall Layer Model

This wall layer model is coded in [*TURB_FloWlmUpdateLogLay*] and is based on logarithmic layer assumption for hydraulically smooth wall as well as with roughness effect at zero pressure gradient. Having the information of streamwise velocity at certain distance from the wall and local molecular viscosity, the friction velocity u_τ is approximated using Newton-Raphson iteration until the (η, V_s) point collapses within certain tolerance to the log-law line. The mean wall stress is then computed from $\langle\tau_w\rangle = \rho {u_\tau}^2$. The effect of roughness is to shift the intercept of the log-law in the positive direction,

$$\frac{V_s}{u'_\tau} = \frac{1}{\kappa} ln(\frac{\eta}{k}) + B \tag{6.21}$$

where $\kappa = 0.41$ is the Von Karman constant, $B = 8.5$ is the intercept value of rough wall log-law and k is the dimensional roughness size. The friction velocity above is is large for high k, but turns to negative when k becomes even larger. Theoretically there is maximum value of u_τ, i.e. when k^+ reaches value 70. This contrain is applied here,

$$u_\tau = \min(u'_\tau, (u'_\tau)_m), \qquad \text{with}$$
$$(u'_\tau)_m = \frac{k^+_m \mu}{k\rho},$$

being friction velocity at possibly maximum value of k^+, namely $k^+_m = 70$, and μ being local molecular viscosity.

6.5 Boundary-Layer Based Wall Layer Model

This model is coded in [*TURB_WlmUpdateBndLay*]. Wall stresses are derived by modeling terms in BL equations. Viscous term has been modeled in the previous section, based on mixing length eddy viscosity model which is equivalent to log layer assumption at zero pressure gradient with effect of surface roughness included. The boundary layer convective term is neglected. Pressure gradient and time derivative term are computed here and added to the viscous model. The method is inspired by paper of G. Hoffman and C. Benocci , "Approximate wall BC for LES", Siena, Italy 1994, published in *5th Advances in Turbulence*, ed. R. Benzi, pp.222-28, Dordrecht: Kluwer.

6.6 Code Organization

6.6.1 Required input

Boundary condition (.bc) example file:

```
# BC_WALLMODEL
BLOCK     0  0   ! applies to block ... (0 0 = to all)
PATCH     0  0   ! applies to patch ... (0 0 = to all patches from above range of blocks)
MODEL     2      ! 0=nomodel 1=loglay 2=bndlay 3=external (feed tau_w distribution)
REFPOINT  1      ! reference point (1 to max. wall normal points in the block)
ROUGHNESS 0.0001   ! roughness size [meter]
#
```

6.6.2 Description of initialization routines

Besides the wlm routines in which the wlm correlation and mapping from the body fitted based wall stress tensor to the Cartesian based tensor are programmed, there are also wlm routines for the initialization procedure. To identify the values assigned to the wlm switches and field data, the task of some of these routines are described.

In [*TURB_InitInputValues*]:
- Switches of all no-slip patches at level 1 are given initial values.
- Switches of other bc type are not defined.

In [*TURB_WlmReadBcSection*]:
- Switches and vals (surface data variable) of SELECTED no-slip patches at level 1 are given user input values.
- Switches of unselected no-slip patches at level 1 keep the given initial values.
- Vals of unselected no-slip patches are not defined (not allocated, nor given any values).

In [*TURB_AllocateMemory*]:
- Switches of all no-slip patches (regardless selected or not for wlm) and vals (surface data

variable) at other levels are allocated and given the corresponding values at level 1.
- Vals of selected no-slip patches at other levels are allocate, whereas unselected vals are undefined at all level.

In [*TURB_InitSolution*]:
- Vals of selected no-slip patches are either copied from level 1, interpolated from level 1, or computed, depending on which vals ID is considered.
- Vals of selected no-slip patches at other levels are allocated, whereas unselceted vals are undefined at all level.
- Metric coefficients for wlm mapping is computed by [*TURB_FloWlmMetric*].
- For external model (external τ_w), τ_w is read from external bc file. Otherwise τ_w is computed from log-law [*TURB_FloWlmUpdateLogLay*] or boundary-layer [*TURB_WlmUpdateBndLay*] based models. The obtained τ_w is then used to replace the conventional wall stresses (directly computed from the strain rate tensor) in routine [*TURB_VisFluxEddyPatch*] and [*TURB_VFluxHybridPatch*] for EV models and TS models , respectively.

Chapter 7

Construction of Gradients

7.1 Formulation and Discretization

Momentum and energy viscous terms in Navier Stokes are represented by $\partial_j \tau_{ij}$ and $\partial_j(\tau_{ij}u_i - q_j)$, respectively, where,

$$\tau_{ij} = \mu(T)(\partial_j u_i + \partial_i u_j - 2/3\delta_{ij}\partial_k u_k), \tag{7.1}$$
$$q_j = \mu(T)Cp/Pr\partial_j T, \tag{7.2}$$

with τ_{ij} the momentum viscous flux, q_j the heat flux vector and μ molecular viscosity which depends on the local temperature for perfect gas and other gas properties for non perfect gas. The viscous terms are centrally discretized at the cell center using the information of τ_{ij} and q_j at the cell faces. In turn, the velocity and temperature gradients which define the fluxes are directly discretized at the cell faces using auxiliary control volume. Quantities (velocities and temperature) on the faces of the auxiliary control volume, if not directly available, are obtained by averaging from the surrounding four cells. Cell faces quantities, such as μ and κ, required for the evaluation of the viscous fluxes are computed by two point averaging from the adjacent cells. For steady state problem, the viscous evaluation is only performed at the first stage of Runge-Kutta and frozen in the remaining stages, wheres for unsteady problem it is perfomed every stage.

By cell-volume averaging and Gauss theorem the gradients are defined as

$$\text{FiniteVolume}[\ \mathbf{grad.}f\ ] = \frac{1}{\int dV}\int \mathbf{grad.}f\mathrm{d}V = \frac{1}{V}\sum_{faces} f\mathrm{d}S, \tag{7.3}$$

where f is the velocity component and temperature and $\mathrm{d}S$ is the face vector. The gradient computation is applied on f as a vector. In this way the three velocity components and temperature are treated at once as a vector. If desired, f can represent any other quantities. The evaluation of the gradients is done in the interior domain first including the boundary nodes. The gradients at the region boundaries and their corresponding dummies are further

completed depending on the boundary condition applied at the patches on the boundaries. The faces of the auxiliary control volume for gradient in face i is defined as

$$\zeta_i(i) = [S_i(i) + S_i(i-1)]/2,$$
$$\zeta_j(i) = [S_j(i) + S_j(i-1)]/2,$$
$$\zeta_k(i) = [S_k(i) + S_k(i-1)]/2,$$

where ζ are the face vectors of the auxiliary control volume and S those of the primary cell. The same formulation hold for the auxiliary control volume of face j and k. The gradients computation in the interior domain is carried out by subroutine *RFLO_calcGradFaces* in *libflo*. After calling this routine actual values of velocities and temperature gradients (i.e. volume division is thus completed) in i,j, and k faces are available in the grid domain [1:ni,1:nj,1:nk] of the current region.

7.2 Gradient Evaluation Outside Interior Domain

The evaluation of the gradients depends on the boundary condition applied at the boundary patches, which can be distinguished in three cases:
- Connecting boundary
- Physical boundary
- Symmetry boundary

In all cases the gradients are evaluated at all patches first and then at the faces of the dummy cells. This is because when extrapolating the gradients to the dummy faces at the edges and corners the gradient values from other patches are required.

7.3 Gradient Evaluation at Boundary Patches

7.3.1 Connecting boundary

Connecting boundary is the boundary where periodic b.c or region interfaces with the same grid points is applied. At these boundary patches the gradients evaluation proceeds the same way as in the interior domain and applied at the faces on the patch and faces of the first up to one before last dummy cells. Therefore the evaluation in the case of connecting boundary is performed for patch faces and dummy faces at once. The outerst faces are completed afterwards by zeroth extrapolation. If a boundary surface consists of multiple patches with connecting b.c., the gradients computation is carried out once for the whole surface. This is done to prevent the border of the adjacent patches being treated twice, which may produce an incorrect result. The gradients at the borders of the connecting boundary surface cannot be treated exactly due to the absent of geometry information (grid points) at these locations. Hence, they are copied from their adjacent interior faces. This gradient computation at connecting boundaries is implemented in subroutine *RFLO_calcGradConnBc*.

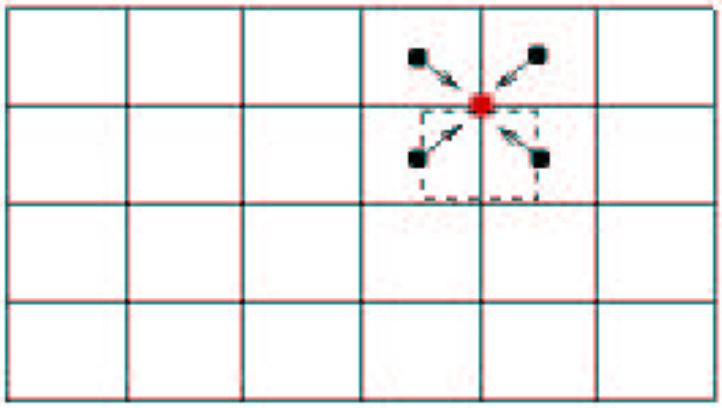


Figure 7.1: Face variables of auxiliary control obtained by averaging from 4 neighboring cells.

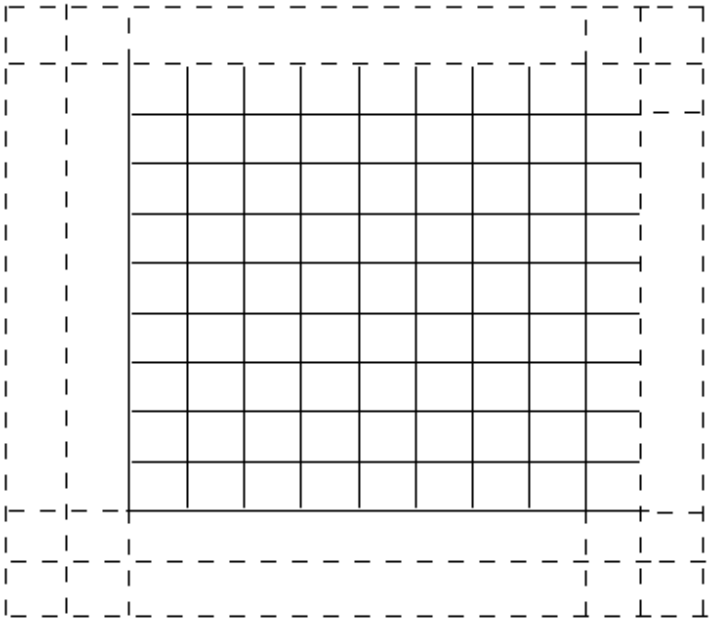


Figure 7.2: Interior (solid) and dummy faces (dash) gradients are first defined at faces drawn with solid lines. Further treatments are required at patches and dummy faces depending on b.c.

7.3.2 Physical boundary

A boundary patch is categorized as physical boundary if one of the following boundary condition is applied: inflow, outflow, slip-wall, noslip-wall, farfield and injection. The gradient evaluation at these boundaries is consequent with that in the interior domain, except that a half control volume is used because no grid is provided at the dummy layers of patches with physical b.c.. The variable values at the faces lying on the patch is regained depending on the extrapolation method used. If linear extrapolation is used, the quantity at the patch face is computed by averaging from the left and right cell center values, one of them is a dummy cell. If constant (zeroth order) extrapolation is used, the value at the patch face is the same is at its adjacent dummy cell. The option to use constant or linear extrapolation is provided only for injection and slip wall boundaries. For other boundaries linear extrapolation is always used. The implementation of this treatment can be seen in subroutine *RFLO_calcGradPhysBc*.

7.3.3 Symmetry boundary

At symmetry boundary the treatment is the same as at the physical boundary with additional treatment of removing the portion normail to the boundary as follow,

$$\mathbf{grad.}f = \mathbf{grad.}f \; - \; \mathrm{dot}(\mathbf{grad.}f, \mathbf{n}) * \mathbf{n}, \tag{7.4}$$

n the unit normal to the boundary. The symmetry boundary computation can be found in subroutine *RFLO_calcGradSymBc*.

7.4 Gradient at Faces of Dummy Cells

Gradients at the faces of dummy cells are obtained either by constant (zeroth order) or linear extrapolation, depending on the boundary condition applied at the boundary patch. In the code this dummy treatment or extrapolation is called from subroutine *RFLO_calcGradDummy*.

7.4.1 Connecting Boundary

Gradients at the faces of up to the first last dummy cells of the connecting boundary patch have been defined in the patch evaluation. The remaining task is to copy the gradients to the last layer, i.e. the outerst faces of the dummy region. This is carried out in subroutine *RFLO_calcGradDummyConn*.

7.4.2 Physical Boundary

At physical boundary the gradients at the faces of dummy cells are obtained by copying (constant extrapolation). They are copied from the patch face value if the dummy faces have the same direction as the patch faces, while copied from the first interior face

if the dummy faces have different direction. This treatment can be found in subroutine *RFLO_calcGradDummyPhys*.

7.4.3 Symmetry Boundary

At symmetry boundary the gradients at the faces of dummy cells are linearly extrapolated with respect to their values at the symmetry patch. For the faces having not the same direction as the patch face, the patch value is obtained by a two point averaging from the two adjacent faces. This treatment is coded in subroutine *RFLO_calcGradDummySym*.

7.4.4 Edges and Corners

The corners are not treated separately but as elongation of edges. Therefore we treat only eight extended edges, which covers corners at their ends. Every edge has 2 sides of dummy faces at which the gradients are known from the previous treatments. The gradients at the edge faces are averaged of the two known gradients at each side's boundary. The treatments are performed in subroutine *RFLO_copyEdgeFaceNorm* and *RFLO_copyEdgeFaceParal*. The former is for the case where the edge faces are facing to one of the two sides, while the latter is where the edge faces lay on the same plain as the known gradients.

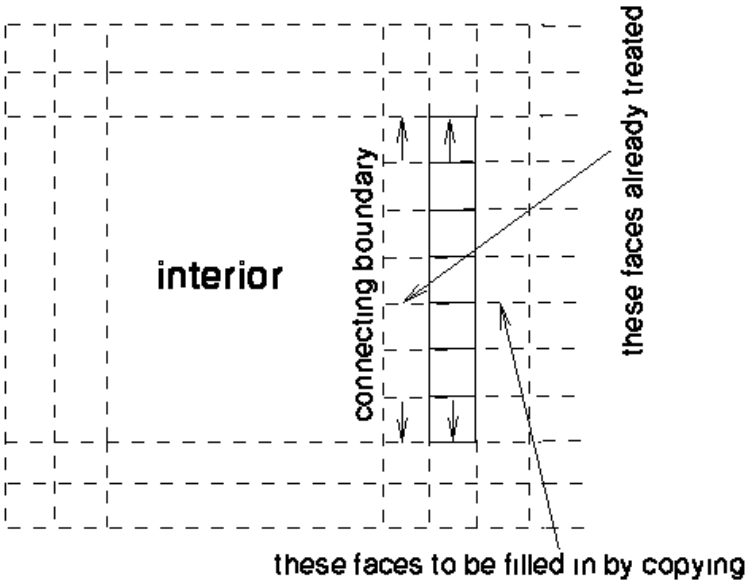


Figure 7.3: Gradiet computations proceed at solid faces at connecting boundary in donfiguration with 3 dummy layers.

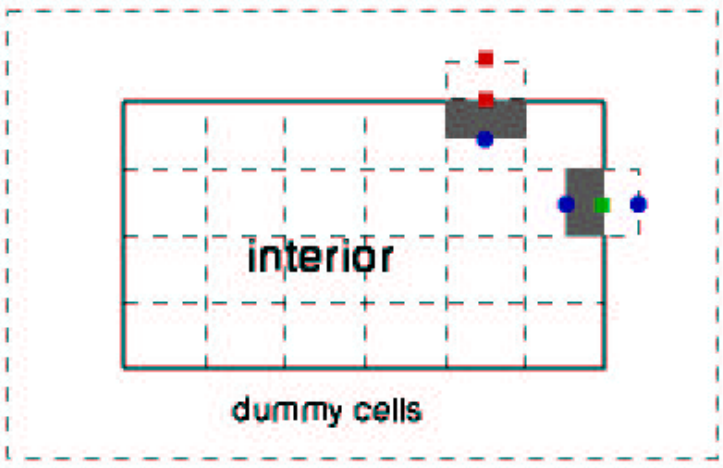


Figure 7.4: Faces at boundary patches where physical b.c. is applied (solid line).

Chapter 8

Reduction of Temporal Statistics

8.1 Features

Time averaged statistics are designed to have the following features:

- Data sampling is taken every time step, weighted by dt.
- Start and end of time interval within which the time averaging proceeds are flexible. In this way the time averaging window is not determined before the process starts. User can continue the averaging process until the averaged quantities are sufficiently smooth. If desired, user can start a new averaging process at any restart time.
- To support the above feature, restart capability is made available.
- The extent of quantities to be time-averaged are 10 first moment variables plus 10! (permutation of 10) second moment variables for each module (mixture, turbulence, particle, smoke, etc).
- In addition to the main fluid statistics above, module interfaces are available for the module developers to construct statistics specific to their modules. These interfaces are *StatDataSampling, TURB_StatMapping, PLAG_StatMapping*, etc. Mixture and physical module statistics are output in the same file. Hence, mixture and module statistics computation has to proceed at the same time frame.
- The statistics routines are common to RFLO and RFLU, except for the I/O, and located in directory *modfloflu* module *ModStatsRoutines*.

8.2 Formulation

The time averaged $< f >$ of vector f can be formulated as follow,

$$F = \int_{T_0}^{T_1} f \mathrm{d}t, \qquad [StatDataAccumulation] \tag{8.1}$$

$$\Delta T = \int_{T_0}^{T_1} 1 \mathrm{d}t, \quad [StatTimeAccumulation] \tag{8.2}$$

$$< f > = F/\Delta T, \tag{8.3}$$

with T_0, T_1 as the starting and ending time of the process, and f the vector of quantities to be time-averaged. The accumulated value F and time interval ΔT are output to a file. If the process is continued at the following restart, it can be described as

$$F = F + \int_{T_1}^{T_2} f \mathrm{d}t, \quad [StatDataAccumulation] \tag{8.4}$$

$$\Delta T = \Delta T + \int_{T_1}^{T_2} 1 \mathrm{d}t. \quad [StatTimeAccumulation] \tag{8.5}$$

8.3 Averaged Quantities

The statistics quantities are determined from user input STATID for mixture, TURBSTATID, PLAGSTATID, etc for physical modules. The ID number of the mixture and module variables can be found in Rocturb User's Guide.

8.4 Algorithm

```
CALL InitStatistics()

DO istep = 1, nstep

  DO istage = 1, nstage
    CALL RungeKutta()
  END DO

  CALL GetStatistics()
  IF( MOD(istep,ioutput) == 0 ) THEN
    CALL RFLO/RFLU_WriteStat()
#ifdef PLAG       CALL PLAG_RFLO/RFLU_CommStatBuffWrapper()
    CALL PLAG_RFLO/RFLU_WriteStat()
#endif   ENDIF

END DO
================================================

SUBROUTINE InitStatistics()

IF ( DOSTAT == 1 ) THEN
```

```
   CALL StatMapping
   IF ( RESTART == 1 ) THEN
      CALL RFLO/RFLU_ReadStat( ΔT, F )
   ELSE
      ΔT = F = 0
   END IF
END IF

END SUBROUTINE InitStatistics()
=================================================

SUBROUTINE StatMapping()

IF ( NSTAT > 0 ) THEN
   DO StatMapping for MIXTURE
END IF
#ifdef TURB
IF ( TURBNSTAT > 0 ) THEN
   CALL TURB_StatMapping()
END IF
#endif
#ifdef PLAG
IF ( PLAGNSTAT > 0 ) THEN
   CALL PLAG_StatMapping()
END IF
#endif
#ifdef PEUL
IF ( PEULNSTAT > 0 ) THEN
   CALL PEUL_StatMapping()
END IF
#endif

END SUBROUTINE StatMapping()
=================================================

SUBROUTINE GetStatistics()

IF ( DOSTAT == 1 ) THEN
   CALL StatDataSampling( f , F )
   CALL StatTimeAccumulation( dt , ΔT )
END IF
```

END SUBROUTINE GetStatistics()
==========

Chapter 9

Data Structures and Input/Output

9.1 Data Types and Variables

The directory `modfloflu` contains module `ModTurbulence` in which the data types and variables used by `rocturb` are declared. Two data types are defined, t_turb and t_turb_input. The former is related to turbulence data used in the turbulence transport (RANS/DES) and Large Eddy Simulation (LES) computations. The latter is used to store user input data and their derived variables.

9.1.1 ModTurbulence: t_turb_input (General Input, Flo/Flu)

- *modelClass* –
- *nOutField* –

9.1.2 ModTurbulence: t_turb_input (RANS/DES Input, Flo/Flu)

- *nTurbEqs* –
- *wDistMetho* –
- *functV1* –
- *spaceDiscr* –
- *spaceOrder* –
- *vis2* –
- *vis4* –
- *smoocf* –

- *cDes* –
- *const* –

9.1.3 ModTurbulence: t_turb_input (LES Input, Flo/Flu)

- *cSmag* –
- *wallRough* –
- *delFac2* –
- *filterType* –
- *deltaType* –
- *filterWidth* –
- *homDir* –
- *engModel* –
- *calcVort* –
- *nCv* –
- *nDv* –
- *nSv* –
- *nGrad* –
- *nSt* –
- *nFixSt* –
- *wallModel* –
- *wlmRefPoint* –

9.1.4 ModTurbulence: t_turb (RANS/DES Data, Flo/Flu)

- *cv* –
- *cvOld* –
- *rhs* –
- *rhsSum* –
- *diss* –
- *dsterm* –
- *lens* –
- *tv* –
- *cvn* –
- *cvn1* –
- *cvn2* –
- *sDual* –

9.1.5 ModTurbulence: t_turb (RANS/DES Data, Flo)

- *srad* –
- *epsIrs* –

9.1.6 ModTurbulence: t_turb (LES Data, Flo/Flu)

- *lij* –
- *mij* –
- *fVar* –
- *ffVar* –
- *ccVar* –
- *trace* –
- *coef* –
- *mueT* –

9.1.7 ModTurbulence: t_turb (LES Data, Flo)

- *ccCofi1* –
- *ccCofi2* –
- *ccCofi4* –
- *ccCofj1* –
- *ccCofj2* –
- *ccCofj4* –
- *ccCofk1* –
- *ccCofk2* –
- *ccCofk4* –
- *ffCofi1I/J/K* –
- *ffCofi2I/J/K* –
- *ffCofi4I/J/K* –
- *ffCofj1I/J/K* –
- *ffCofj2I/J/K* –
- *ffCofj4I/J/K* –
- *ffCofk1I/J/K* –
- *ffCofk2I/J/K* –
- *ffCofk4I/J/K* –
- *fVolI/J/K* –
- *fI/J/KSij* –

9.1.8 ModTurbulence: t_turb (LES Data, Flu)

- *avgCoI* –
- *bavgCoI* –
- *ccCofi1* –
- *ccCofi2* –
- *ccCofi4* –
- *ffCofi1I* –
- *ffCofi2I* –
- *ffCofi4I* –
- *bffCofi1I* –
- *bffCofi2I* –
- *bffCofi4I* –
- *fvolI* –
- *bfVolI* –
- *fISij* –
- *bfISij* –
- *bLij* –
- *bMij* –
- *bfVar* –
- *bffVar* –
- *bCoef* –
- *bMueT* –

9.1.9 ModTurbulence: t_turb LES/RANS/DES Data, Flo/Flu)

- *dv* –
- *sv* –
- *vort* –

9.1.10 ModTurbulence: t_turb LES/RANS/DES Data, Flo)

- *gradi/j/k* –
- *mI/J/KSij* –
- *workI/J/K* –

9.1.11 ModTurbulence: t_turb LES/RANS/DES Data, Flu)

- *gradi* –
- *bGradi* –
- *mISij* –
- *bmISij* –

9.1.12 ModTurbulence: t_turb (Statistics Data, Flo/Flu)

- *tav* –
- *st* –
- *stwork* –

9.2 Input Output Files

The execution of **Rocturb** in general depends only the .inp file: # TURBULENCE section. In some cases, one requires input from .bc file, such as for wall layer model and inlet turbulence computations.

9.2.1 Input: .inp file

9.2.2 Input: .bc file

9.2.3 Solution

Bibliography

[1] Bardina, J., Ferziger, J.H., and Reynolds, W.C. (1984). Improved turbulence models based on LES of homogeneous incompressible turbulent flows. Department of Mechanical Engineering. *Report No. TF-19*, Stanford.

[2] Durbin, P.A. (1991). Near wall turbulence clossure modeling without damping functions. *Theoretical Comp. Fluid Dynamics*, **3**, No. 1, 1-13.

[3] Germano, M., Piomelli, U., Moin, P., and Cabot, W.H. (1991). A dynamic subgrid scale eddy viscosity model. *Phys. Fluids A*, **3**, 1760-1765.

[4] Germano, M. (1992). Turbulence: the filtering approach. *J. Fluid Mech.*, **238**, 325-336.

[5] Hoffman, G., and Benocci, C. (1994). Approximate wall boundary conditions for LES", 5th Advances in Turbulence, ed. R. Benzi, pp.222-28. Dordrecht: Kluwer.

[6] Jones, W.P., and Launder, B.E. (1972). The prediction of laminarization with a two equation model of turbulence. *International J. of Heat and Mass Transfer*, **15**, 301-314.

[7] Launder, B.E., and Sharma, B.I. (1974). Application of the energy dissipation model of turbulence to the calculation of flow near a spinning disc. *Letters in Heat and Mass Transfer*, **1**, No. 2, 131-138.

[8] Nikitin, N.V., Nicoud, F., Wasistho, B., Squires, K.D., and Spalart, P.R. (2000). An approach to wall modeling in large eddy simulations. *Phys. Fluids*, **12**, No. 7, 1629-1632.

[9] Smagorinsky, J. (1963). General circulation experiments with the primitive equations. *Mon. Weather Rev.*, **91**, 99-164.

[10] Piomelli, U., and Balaras, E. (2002). Wall layer models for large-eddy simulations, *Annu. Rev. Fluid Mech.* **34**, 349-374.

[11] Poinsot, T.J., and Lele, S.K. (1992). Boundary conditions for direct simulations of compressible viscous flows, *J. Comp. Physics* **101**, 104-129.

[12] Venugopal, P., Najjar, F.M., and Moser, R.D. (2000). DNS and LES computations of model solid rocket motors. *AIAA 2000-3571,* 36^{th} *AIAA/ASME/SAE/ASEE Joint Propulsion Conference and Exhibit*, Huntsville, AL, July 16-19.

[13] Spalart, P.R., and Allmaras, S.R. (1994). A oner equation turbulence model for aerodynamic flows'. *La Recherche Aerospatiale*, No. 1, 5-21.

[14] Vreman, B., Geurts, B., and Kuerten, H. (1994). On the formulation of dynamic mixed subgrid scale model.*Phys. Fluids*, **6**, 4057-4059.

[15] Vreman, A.W., Geurts, B.J., Kuerten, J.G.M., Broeze, J., and Wasistho, B. (1995). Dynamic subrid-scale models for LES of transitional and turbulent compressible flow in 3-D shear layers, *Proc.* 10^{th} *Symposium on Turbulent Shear Flows*, Pennsylvania.

[16] Wasistho, B., Balachandar, S., and Moser, R.D. (2004). Compressible Wall-Injection Flows in Laminar, Transitional, and Turbulent Regimes: Numerical Prediction. *AIAA Journal of Spacecraft and Rocket*, to appear, **41**, No. 6, 915-924.

[17] Wasistho, B., and Moser, R.D. (2005). Simulation strategy of turbulent internal flow in solid rocket motor. *J. Propulsion Power*, **21**, No. 2.

[18] Wasistho, B., Najjar, F., Haselbacher, A., Balachandar, S. and Moser, R.D. Effect of Turbulence and Particle Combustion on Solid Rocket Motor Instabilities, AIAA 2005-4345, Joint Propulsion Conference 2005, Tucson, AZ.

[19] Werner, H., and Wengle, H. (1991). Large eddy simulation of turbulent flow over and around a cube in a plate channel. *Proceedings of the* 8^{th} *Symposium on Turbulent Shear Flows*, Technical Univ., Munich, Germany, 19.4.1-19.4.6.

[20] Wilcox, D.C. (1993). Turbulence Modeling for CFD. *DCW*, California.

[21] Yoshizawa, A. (1986). Statistical theory for compressible turbulent shear flows, with application to subgrid modeling. *Phys. Fluids A*, **29**, 2152-2164.

[22] Zang, Y., Street, R.L., and Koseff, J.R. (1993). A dynamic mixed subgrid scale model and its application to turbulent recirculating flows. *Phys. Fluids A*, **5**, 3186-3196.