Cranfield University

Cranfield College of Aeronautics

PhD Thesis

Yon Han CHONG

Monotone Integrated Large Eddy Simulation of Supersonic Boundary Layer Flows

Supervisor: Professor Ning Qin

24 February 2001

This thesis is submitted for the degree of Doctor of Philosophy.

Contents

Acknowledgements

I would like to dedicate the thesis to my family for the unconditional support they have given to me. Especially, I am indebted to my parents for the rest of my life.

I would like to thank my supervior Prof. Ning Qin for the invaluable academically guidance and advice. I also would like to thank Dr. David Ludlow, Dr. Scott Shaw and Dr. Simon Prince not only for academic discussions, but making life in Cranfield more bearable.

I would like to thank my friends and the members of Cranfield College of Aeronautics for helping me over the difficult times, and giving me moral supports.

Abstract

For simulations of supersonic flows shock-capturing schemes have to be used. A shock-capturing scheme produces more dissipation than a central difference scheme. In fact, the numerical dissipation produced by shock-capturing schemes is problematic when performing Large Eddy Simulation of supersonic flows with shock-waves. Another train of thought is to turn the numerical dissipation to our advantage. If the numerical dissipation of a numerical method can mimic the dissipation of the subgrid-scale(SGS) eddies, not only is SGS modelling unnecessary, but the numerical dissipation will be a positive contribution to the calculation. This approach is called MILES.

As a reference case, a zero-pressure-gradient, flat-plate boundary-layer flow was chosen as there are analytical, experimental, DNS and LES results available. The freestream conditions are a Mach number of 2.25 and a Reynolds number of $1.613 \times 10^4/\text{in}$ or 6.007×10^3 based on the displacement thickness.

The central difference scheme, Osher's scheme and Roe's scheme are tested for suitability in MILES. The central difference scheme is found to be numerically too non-dissipative without SGS modelling. Osher's scheme is too dissipative so that it hinders the development of turbulence. Roe's scheme without use of a limiter seems to have the right amount of numerical dissipation to mimic a SGS model. Two popular slope limiters were also tested, but both affected turbulence development when no shockwave was present.

Nomenclature and Abbreviation

Abbreviation

CFD	Computational Fluid Dynamics
CPU	Central Processing Unit
DNS	Direct Numerical Simulation
FSF	Filter-Structure-Function
LES	Large Eddy Simulation
MILES	Monotone Integrated Large Eddy Simulation
MPI	Message Passing Interface
MUSCL	Monotone Upstream-centred Scheme for Conservation Law
ENO	Essentially Non-oscillatory
PSE	Parabolized Stability Equation
PVM	Parallel Virtual Machines
RANS	Reynolds Averaged Navier-Stokes
SF	Structure Function
SGS	Subgrid-Scale
SIMD	single-instruction multiple-data
TVD	Total Variation Diminishing
VLES	Very Large Eddy Simulation

Nomenclature

C_{ij}	SGS cross-term stress
C_d	dynamic Smagorinsky coefficient
C_K	Kolmogorov constant
C_p	specific heat at constant pressure
C_R	compressible Smagorinsky coefficient
C_S	Smagorinsky coefficient
C_v	specific heat at constant volume
$E(\kappa)$	energy spectral density or energy spectrum function
E_p	efficiency of a parallel computer
f_s	fraction of a code which is serial
$G(\mathbf{x}')$	grid-level filter function
(κ)	Fourier transform of G
h	space between two grid points
k_t	thermal conductivity
k_e	kinetic energy contained in the turbulence
l	SGS length scale
l_{int}	integral scale
L_{ij}	SGS Leonard stress
M_t	turbulent Mach number
np	number of processors
p	pressure
Pr	Prandtl number
Pr_T	turbulent Prandtl number
q_k	SGS heat flux
q_k^L	SGS Leonard heat flux
q_k^C	SGS cross-term heat flux
q_k^R	SGS Reynolds heat flux
$\Re'$	residuals
$\Re^*$	sum of residuals
R_{ij}	SGS Reynolds stress
R_{ij}^D	deviatoric part of compressible SGS Reynolds stress
R_{ij}^I	isotropic part of compressible SGS Reynolds stress
S	mean strain rate
S_p	speedup of a parallel computer
Re	Reynolds number
Re_m	Reynolds number based on the half width of the channel

t	time
T	time or temperature
$T_{s(1)}$	the time taken to execute a serial code
$T_{s(n)}$	time taken to execute a unparallelisable part of parallel code
T_p	the time taken to execute parallelised part of parallel code
T_o	overhead time for the communication
$_i$	Fourier transform of u_i
u, u_i	local velocity
$\overline{u}_i$	averaged value or large-scale/resolvable-scale component of u_i
$\widetilde{u}_i$	Favre filtered large-scale/resolvable-scale component of u_i
u_i'	local velocity fluctuation or small-scale/subgrid-scale component of u_i
U_∞	freestream velocity
v	velocity component in y-direction
$\overline{v}$	time averaged velocity component in y-direction
v^*	wall-friction velocity
w	velocity component in z-direction
$\mathbf{x}, x_i$	physical-space coordinates
$\overline{w}$	time averaged velocity component in z-direction
Δt	timestep in computation
Δ	smallest turbulence scale allowed by a filter
$\Delta_f, \overline{\Delta}$	filter width
Δ_s	grid size
$\widehat{\overline{\Delta}}$	test-level filter width, $\left(\Delta x_1 \Delta x_2 \Delta x_3\right)^{1/3}$
δ_{ij}	Kronecher delta
δ_∞	boundary layer thickness
ε	the rate at which the small scale has to dissipate energy or a small number
η	Kolmogorov scale
κ	wavenumbers
λ	Taylor microscale
μ	viscosity
ν	kinematic viscosity
ϕ	any value
ρ	density
τ_{ij}	SGS stress tensor
τ_K	Kolmogorov time scale
τ_w	wall shear stress

Chapter 1

Introduction

Combination of recent computer hardware developments and the advancement of numerical methods, enable even more computationally demanding cases to be attempted in the field of Computational Fluid Dynamics.

But, still, Direct Numerical Simulations (DNSs) are too expensive except for simple geometries and so it cannot be done on a regular basis. It is even more true with the cases involving compressible supersonic flows. Compared to incompressible flow, there is an extra set of equations to be solved and shock-capturing methods add even more complexities. Therefore it is not surprising that not many simulations have been performed for this category of cases. The prospect of using compressible DNS in engineering problems in the near future is very slim and consequently it is likely to remain solely as a research tool for fundamental study of the physics of turbulence in the foreseeable future.

Large Eddy Simulation (LES) promises a brighter future by reducing the difficulties of turbulence modelling with a substantially less computation, as compared to DNS. The saving on computation time can be 10 times or more greater than DNS but modelling is needed to make up the unresolved small eddy details (subgrid-scale modelling). Ideally the unresolved small eddies should be homogeneous turbulence and simple models can be used. LES with an insufficient grid resolution requires a SGS model to model the heterogeneous turbulence which may produce a greater error.

So why is it that LES is not regularly used to simulate engineering problems?

Firstly, despite the computational time saved compared to DNS, LES is still too expensive for widely available workstations. Parallel computing has really taken off in the last 5–6 years for two main reasons; (i) the standardization of communication libraries and (ii) the escalating cost of developing new faster CPUs. Message Passing Interface (MPI) is a communication standard drawn from the experiences of many professionals. It is now adopted for most operating systems and available free of charge. This means once a parallel version of a code has been developed it can be easily ported to many different computer platforms. Nevertheless, transforming a serial code to a parallel version using MPI and maintaining it takes time and expertise. This is especially true if an unfamiliar code needs to be parallelized. Softwares, which automatically parallelise other computer software, are limited. One such example is written by SGI, but their resulting parallel codes when compiled with simple parallel options are at best adequate when 10 or more processors are used. In years to come, further development will make automatic parallelisation more efficient but at present, parallel efficiencies are poor in many cases.

Secondly, the simple Subgrid-Scale (SGS) models are found to be insufficient for many cases. Wall bounded flow is one case where the performance of the SGS model can be called into question. The simple Smagorinsky SGS model produces high turbulent viscosity towards the wall when it should behave oppositely. More complicated SGS models have appeared but only a couple of models are widely used.

Thirdly, the contribution to numerical errors from different numerical methods is still unclear and debatable. In many cases, the magnitude of numerical error is comparable to the contribution of the SGS model. In simulations of flows involving complex geometries the use of finite volume methods with accuracy of 3-orders or less may be unavoidable. In this case, the magnitude of numerical errors will be comparable to the contribution by the SGS model and usefulness of the model has to be questioned.

Fourthly, when shock capturing methods are used there will be additional numerical dissipation. Ideally, shock capturing methods should only affect the shock region. But as will be shown in this thesis, this is not the case.

To turn the problem on its head, rather than increasing the numerical accuracy by using more complicated numerical methods and SGS models, the numerical dissipation associated with shock capturing methods can be used as dissipation by a SGS model. This is the philosophy of Monotone Integrated Large Eddy Simulation (MILES).

This thesis explores the suitability of different shock capturing schemes used in MILES. As a test case, a turbulent boundary layer on a flat plate has been chosen. There have been numerous experiments for this configuration, and a few DNS and LES investigations. Results from the proposed MILES schemes will be compared with the experimental data and the DNS and LES results. This test case is more difficult to simulate than previous MILES calculations since it involves a solid boundary. There has not been much study of using shock capturing scheme for MILES and most of the past test cases did not involve solid boundaries e.g. jets, free shear layers and homogeneous turbulence.

In Chapter 2, background information and a literature survey concerning the project are presented. It includes a physical description of the turbulent boundary layer, categorization of high end CFD methods with much emphasis on LES and MILES, and general information on parallel computing.

In Chapter 3, the basic governing equations are presented in different forms. Mathematical differences between DNS, LES, MILES and RANS are also described.

In Chapter 4, various numerical methods used in the current simulations are described. These encompasses the finite volume method, Roe's scheme, Osher's scheme, MUSCL, limiters and time marching.

In Chapter 5, techniques used to parallelize computer code are described. Its description is rather specific to the current case, rather than general code parallelization. There also are some practical points for writing and running a parallel code.

In chapter 6, the reference case for the current simulations and the simulation conditions are described. This includes a description on generating the initial conditions and setting the boundary conditions.

In chapter 7, results are presented. The results are compared with experiment, DNS and LES results, and theoretical methods. Also, the suitability of different numerical methods in MILES are discussed.

Chapter 8 contains the discussion, conclusion, and suggestions for further researches based on the current findings.

Chapter 2

Background Information and Literature Survey

2.1 Turbulent Boundary Layer on a Flat Plate

2.1.1 Physical Description and Experiments

Figure 2.1 shows the schematic diagram of a turbulent boundary-layer on a flat plate. A shock-wave forms at the leading edge of the plate due to the boundary layer displacement effect. Underneath the shock-wave a boundary layer develops from laminar to turbulent state.

Compressible turbulent boundary layers can be described well by the law of the wall. Strictly speaking, there are two laws of the wall, one for velocity and one for temperature (or density). Nevertheless, the simplicity and effectiveness of the Crocco-Busemann relation make the velocity law of the wall much more popular and the temperature law of the wall is seldom referred to. From here onwards velocity law of the wall will be referred to as the law of the wall.

Like many of the turbulent-flow analyses the law of the wall is semiempirical in nature. There are at least four separate approaches in the literature for achieving a velocity law of the wall:

1. correlation of measured profiles,
2. the effective-velocity approach of van Driest,

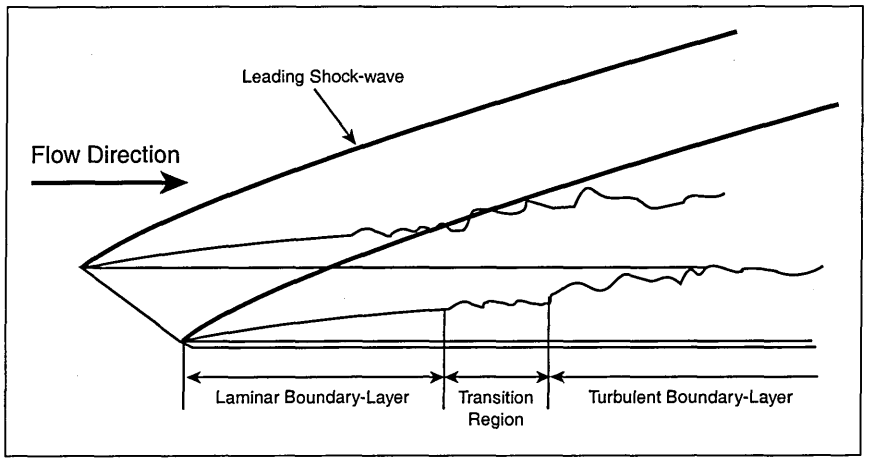

Figure 2.1: Schematic diagram of a turbulent boundary-layer

3. a coordinate transformation which reduces a compressible turbulent flow to an equivalent incompressible turbulent flow,
4. an eddy-viscosity theory which accounts, at least to first order, for compressibility, heat-transfer, and pressure-gradient effects.

Each of these approaches are described in White [1].

Figure 2.2 shows the plot of normalized velocity profile of Mach 2.25 supersonic turbulent boundary from experimental data by Elena and LaCharme [2], together with the law of the wall. The velocity and the distance from the wall are non-dimensionalized as following:

$$u^+ = \frac{\overline{u}}{v^*}, \quad y^+ = \frac{yv^*}{\nu} \tag{2.1}$$

where $\overline{u}$ is time averaged velocity and v^* is *wall-friction velocity*. The latter is defined as:

$$v^* = \sqrt{\frac{\tau_w}{\rho}}, \quad \tau_w = \mu \left.\frac{\partial \overline{u}}{\partial y}\right|_w \tag{2.2}$$

where τ_w is the wall shear stress and subscript w represents value at next to wall.

In the inner layer where the viscous shear dominates, the velocity profile should be linear:

$$u^+ = y^+; \tag{2.3}$$

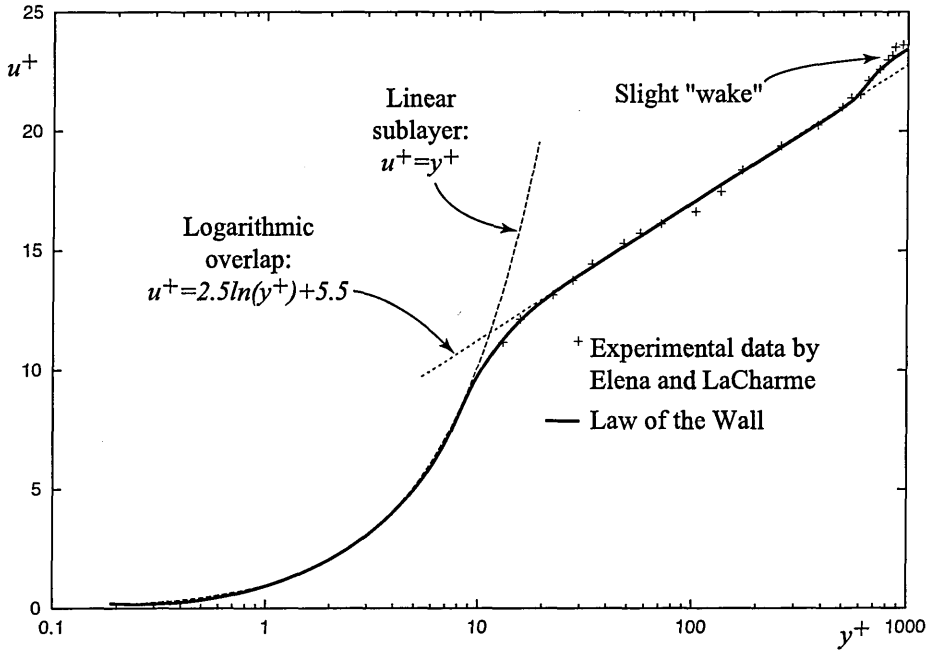

Figure 2.2: Normalized velocity profile from experimental data by Elena and LaCharme [2] with the law of the wall.

this is also called the *linear sublayer*. Then, in the vicinity of $y^+ = 35$, this layer must turn and merge smoothly into the logarithmic region where the velocity profile can be described by:

$$u^+ = 2.5 \ln\left(y^+\right) + 5.5 \tag{2.4}$$

Further away from the wall, there is a slight "wake" region before the flow becomes more or less inviscid. Even though there is a law of the wake [1] it is less successful than the law of the wall, due to the sensitivity to external parameters.

There are many published experiments in the literature concerning high-speed turbulent boundary layers without complicating effects such as a shock-wave interaction, mass transfer, or rarefied-gas densities. There are a number of catalogues and reviews of experimental data on compressible boundary layers including Fernholtz and Finley [3,4], Fernholtz *et al.* [4], Settles and Johnson [5], Spina *et al.* [6], Bradshaw [7], and Lele [8]. The measurement of turbulent quantities in a supersonic flow is not only difficult but also expensive.

2.1.2 Computational Difficulties

This simple geometry still poses a great computational challenge for LES or DNS because of the presence of a wall. It is necessary to have a very fine grid towards a wall. As Pruett *et al.* [9] observed from past calculations there are additional reasons for the great expense of computations of high-speed compressible flow relative to simulations of incompressible or subsonic flows:

1. The time discretization in the compressible case was mostly explicit. In many instances the allowable time step was limited by the viscous stability condition rather than by the advection condition.

2. The second-mode disturbances associated with high-speed transitional flows have a double-peaked structure with amplitude peaks occurring both near the wall and the critical layer. In contrast to low-speed flows, at high speeds the critical layer lies far (approximately one displacement thickness) from the wall, necessitating concentrations of grid points in both regions of strong gradients.

3. At high speeds the growth rates of both the primary and secondary instabilities are much slower than for low-speed flows. This requires much longer time integrations.

4. In contrast to DNS of incompressible flow, flow-field oscillation due to inadequate resolution are potentially fatal in the compressible case since spurious negative densities, pressures, and/or temperatures can arise.

Therefore, it is not surprising that flows over a flat plate are simulated not only to understand the fundamentals of turbulence but to test numerical methods for DNS and LES.

2.1.3 Past Calculations

Direct Numerical Simulations (DNS) and Large Eddy Simulations (LES) of compressible boundary layers on a flat plate can be divided into two categories: temporal or spatially developing. For a thorough background of transitional com-

pressible wall-bounded flows, including a discussion of the temporal and spatial problems see Kleiser and Zang [10].

Temporal Cases

In the temporal case, the computational domain is as if for a flow along the plate with infinite length. This can be achieved by re-feeding the outflow boundary condition to the inflow boundary. The growth of the boundary layer thickness can be accounted for by an extra force term [11] or by compressing the outflow boundary layer [12] before using it for inlet boundary condition.

Sandham *et al.* [13] investigated with DNS the late stages of transition to turbulence in a Mach 2.0 boundary layer with DNS. The transition was initiated by oblique instability waves. They found that quasi-streamwise vortices dominate the early nonlinear stages, but do not directly lead to transition, decaying in strength after their initial development. However, they do generate high shear layers locally, and because of slight skewness and inclination lead to secondary counter-rotating vortices. These vortices trigger roll-up of the high shear layer and the final breakdown to turbulence.

Ducros *et al.* [14] simulated the forced transition of a temporal boundary layer over an adiabatic flat plate by means of direct and large eddy simulations, for an external Mach number of 4.5. They were able to observe the typical phenomena of laminar to turbulent transitions e.g. Kelvin-Helmholtz-like vortices, turbulent formation in the forms of streaks and hairpin vortices.

The nonlinear evolution and laminar-turbulent breakdown of a boundary-layer flow along a cylinder at Mach 4.5 was simulated with LES by El-Hady and Zang [15]. They claimed that their structure-function dynamic SGS model produced better results than the dynamic eddy-viscosity SGS models of Germano [16,17]. They also noted that the temporal simulation offers qualitatively, a more economic way to help understand the physics of transition but the spatial evolution of disturbances and a meaningful transition prediction can only be described accurately by spatial simulations.

Childs and Reisenthel [18] simulated high-speed turbulent boundary layers at Mach 5. Together with experimental studies, various measures of dynamical compressibility effects on turbulence were investigated. They found that compressiblity was small but probably not negligible.

Hatay and Biringen [19] explored the paths of energy transfer through which compressible turbulence is sustained. The structural similarities and differences between incompressible and compressible turbulence were also investigated. They found that the turbulent energy balance is strongly similar to incompressible wall-shear layers but the compressibility effects are responsible for the lower magnitudes and non-zero values of the pressure-strain terms in the turbulent kinetic energy.

Guo *et al.* [11] investigated nonparallel effects in temporal DNS of compressible boundary-layer transition. They modified the DNS method to account for periodic boundary conditions in the streamwise direction and improved the results significantly compared to the spatial DNS results. In a following paper Adams and Kleiser [20] studied the subharmonic transition process of a flat-plate boundary layer at a free-stream Mach number of 4.5 and a Reynolds number of 10000 (based on freestream velocity and initial displacement thickness) with the same method. They were able to observe many detailed phenomena of transition to a turbulent boundary layer including λ-vortices, Y-shaped shear layer, shear layer break-up and breakdown to turbulence.

Urbin and Knight [21] used a three-dimensional unstructured grid of tetrahedral cells and a finite volume formulation to simulate LES of a compressible corner. They also simulated a turbulent boundary layer for the inlet boundary condition. For the flat plate boundary layer, the friction velocity predicted was within 3% of the theoretical value, and profiles of Reynolds shear and normal stresses were in good agreement with experimental data.

Normand and Lesieur [22] simulated transition to turbulence in the three-

dimensional compressible boundary layer over a semi-infinite insulated flat plate with DNS and LES. They studied Mach number 0.5 and 5, but DNS was done only in Mach number 0.5 case due to lack of computational resources. They simulated both in temporal and spatial configurations. A subgrid-scale model, based on the second-order velocity structure function proposed by Métais and Lesieur [23], was used. The SGS model has been further developed and published in Ducros [24]. This work is described further in the following section.

Spatially Developing Cases

In the spatially developing case, the computational domain is fixed to one location and the whole boundary layer development can be seen in the domain. Spatially developing calculations are even more computationally demanding, primarily because of the greater length of the computational domain. Nevertheless, there have been several simulations of spatially developing cases.

Maestrello *et al.* [25] studied the interaction between two-dimensional second modes and three-dimensional first modes which can lead to the development of smaller scale motions within the flow field as the disturbance grow spatially. They concluded that the spatial growth of two-dimensional second-mode waves can lead to a pronounced increase in the three-dimensional character of the flow. The rapidly growing scond mode can promote features such as vortex roll-up and the formation of localized regions of intense vorticity in specific spanwise planes which can accelerate the development of the flow field toward transition.

Fasel *et al.* [26] presented an extensive set of results from DNS of supersonic boundary layer, comparing fundamental-, subharmonic- and oblique-breakdown scenarios. They mainly studied the secondary instability of both fundamental and subharmonic types in Mach 1.6 boundary-layer flows along a flat plate.

Pruett *et al.* [9] simulated spatially evolving high-speed boundary-layer flows in a body-fitted orthogonal coordinate system to validate their algorithm for DNS. They found that results from DNS and the parabolized stability equation (PSE) methods are remarkably in agreement.

Ducros [24] simulated the transition to turbulence in a boundary layer developing spatially over a flat plate to validate their filter-structure-function (FSF) SGS model improved from the structure-function (SF) model of Métais and Lesieur [23]. It consists of removing the large-scale fluctuations of the field before computing its second-order structure function. Iso-surfaces of eddy viscosity confirm that the FSF model does not perturb transition much, and acts mostly in the vicinity of the hairpins.

Shan *et al.* [27] investigated by LES the oblique transition process of a flat-plate boundary layer at a free-stream Mach number of 4.5 and a Reynolds number of 10000 based on free-stream velocity and inflow displacement thickness. A pair of oblique first-mode perturbations are imposed on the inflow boundary. They were able to observe the transitional process in great details and their results were in good agreement with other DNS and theoretical results.

A DNS of a spatially evolving, compressible, flat-plate boundary-layer flow, with freestream Mach number of 2.25 was performed by Rai *et al.* [28]. Their results compared well with results from the Parabolized Stability Equation. Their computed results were not compared with compressible flow experimental data because of the wide variation they found in the various datasets reported in the literature. However they did compare the statistics of near-wall region with experimental data of a *incompressible* flow, and found that they are in good agreement. They postulated that this is because the flow is subsonic and Morkovin's hypothesis is expected to apply.

Compressible LES of turbulent boundary layer on a flat plate by Arad and Martinelli [29] used multigrid driven scheme, originally developed for the solution of RANS. Smagorinsky model with damping on the wall was used as a SGS model. The drawbacks of "upwind" dissipative schemes, which tend to induce non-physical and often excessive damping of turbulent energy, was investigated using CUSP (Convective Upstream Split Pressure) scheme. The simulated time duration is too short to develop turbulent field but the logarithmic region has de-

veloped during this time.

Spyropoulous and Blaisdell [30] simulated the same test case to investigate issues involving the LES of wall-bounded compressible turbulent flows. They have modified the DNS code of Rai *et al.* [28] to perform LES with a dynamic subgrid-scale model. They found that numerical errors from the finite difference method can be large enough so that the SGS model has little effect on the solution. Therefore, they concluded that the effect of discretization errors on LES needs to be addressed further before proceeding with LES in flows of engineering interest.

2.2 Simulations of Turbulent Flow

Ferziger [31] classified methods of computing turbulent flows into five major categories: (i) Correlation Methods, (ii) Integral Methods, (iii) Reynolds-Averaged Navier-Stokes Equations, (iv) Large Eddy Simulations, and (v) Direct Numerical Simulations. In this section, three of the higher end simulations (iii-v) will be discussed and LES will be discussed in more depth in the next section.

2.2.1 The nature of turbulence

Before plunging into Large Eddy Simulation, let's have a look at the general nature of turbulence. Turbulence originates from instabilities of laminar flows and these instabilities produce wave-like structures which can absorb energy from the mean flow [32]. As they grow, nonlinear effects cause energy transfer to other modes, and eventually, the noisy pattern that is generally regarded as "turbulence" results. Fully developed turbulence always reflects its origins to some degree. Since the maintenance of turbulence requires continuous nourishment, turbulent structures must be capable of absorbing energy from the mean flow. Although the mean flow is changed by the presence of turbulence, the energy absorbing structures bear some resemblance to those from which the turbulence originated.

Turbulence flows contain a wide range of length and time scales (Figure 2.3). The largest scale is determined by the geometry of the flow and these **large structures** are the ones which absorb energy from the mean flow. They tend to be

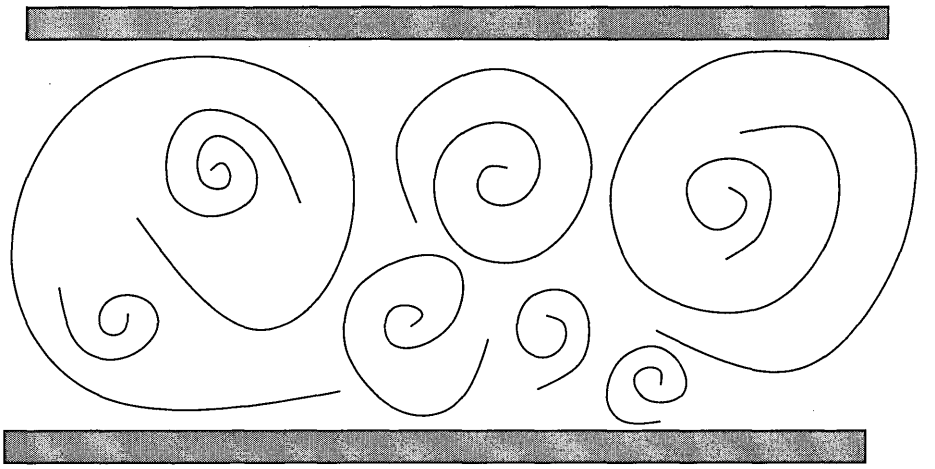

Figure 2.3: Schematic diagram of a turbulent flow

highly *anisotropic* and *vortical* in nature, and quite *variable from flow to flow.* They are also responsible for most of the *property transport* in turbulent flow. Furthermore, since the production mechanism is largely the stretching of vortices, which is a process requiring three dimensions [33], all true turbulent flows are *three dimensional.*

Through nonlinear interactions, these large structures transfer some of their energy to smaller-scale structures, and the major function of the **small structures** is to *dissipate* the energy provided by the larger ones to heat. The cascading process present in all turbulent flows involves a transfer of kinetic energy from larger eddies to smaller eddies. Dissipation of kinetic energy to heat through the action of molecular viscosity occurs at the scale of the smallest eddies. Because small-scale motion tends to occur on a short time scale, we can reasonably assume that such motion is independent of the relatively slow dynamics of the large eddies and of the mean flow [34]. Hence, the smaller eddies should be in a state where the rate of receiving energy from the larger eddies is very nearly equal to the rate at which the smallest eddies dissipate the energy to heat. This is known as Kolmogorov's **universal equilibrium theory**, a corollary of which is his **hypothesis of local isotropy** [35, 36]. Hence, motion at the smallest scales should depend only upon *the rate at which the large eddies supply energy*, ε (or the rate at which the small scale has to dissipate energy) and *the kinematic viscosity*, ν. For this

reason, they are much more universal than the large structures and they are therefore nearly the same in all flows and nearly isotropic i.e. in near equilibrium.

2.2.2 Turbulence scales

There are three turbulence length scales often referred to in the statistical theory of turbulence: Kolmogorov scale η, integral scale l_{int} and Taylor microscale λ. **Kolmogorov scale** is the smallest turbulent scale in a flow. It can be estimated using the following equation:

$$\eta \equiv \left(\frac{\nu^3}{\varepsilon}\right)^{1/4} \tag{2.5}$$

where ν is the kinematic viscosity and ε is the energy dissipation rate. The **integral scale** is appropriate to the energy-bearing eddies. It can be estimated using the following equation:

$$l_{int} \sim k_e^{3/2}/\varepsilon \tag{2.6}$$

where k_e is the kinetic energy contained in the turbulence. For isotropic turbulence, the **Taylor microscale** is defined by:

$$\varepsilon = 15\nu\overline{\left(\frac{\partial u'}{\partial x}\right)^2} \equiv 15\nu\frac{\overline{u'^2}}{\lambda^2} \tag{2.7}$$

Using Equation (2.6) and assuming $k_e \sim \overline{u'^2}$.

$$\lambda \sim \left(l_{int}\eta^2\right)^{1/3}. \tag{2.8}$$

Thus, in general we can say that for high Reynolds number turbulence there is a distinct separation of these scales, i.e.,

$$\eta \ll \lambda \ll l_{int}. \tag{2.9}$$

2.2.3 Spectral analysis of turbulence

A second important consideration to turbulence analysis is the spectral representation of turbulence properties. That is, since turbulence contains a continuous

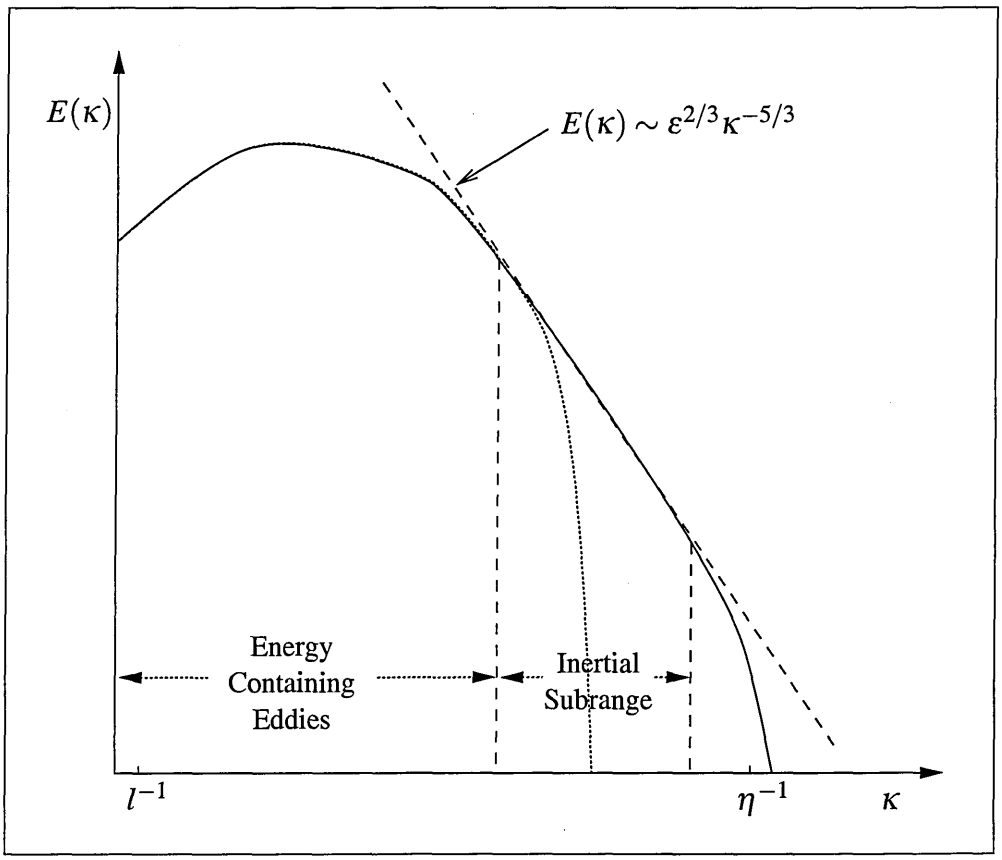

Figure 2.4: The solid line represents the energy spectrum of a turbulent flow (log-log scales); Dotted line represents the energy spectrum of a LES simulated flow without SGS modelling.

spectrum of scales, it is often convenient to cast our analysis in terms of the **spectral distribution** of energy. If κ denotes wavenumber and $E(\kappa)d\kappa$ is the turbulent kinetic energy contained between wavenumbers κ and $\kappa + d\kappa$, we can say

$$k_e \equiv \frac{1}{2}\overline{u'_i u'_i} = \int_0^\infty E(\kappa)d\kappa \tag{2.10}$$

where $E(\kappa)$ is the **energy spectral density** or **energy spectrum function**. In general, a spectral representation can be regarded as a decomposition into wavenumbers (κ) or equivalently, wavelengths ($2\pi/\kappa$). Figure 2.4 shows a typical energy spectrum for a turbulent flow. More detailed discussions of energy spectra are given by Landahl and Mollo-Christensen [37] and Hinze [38]. In the present context, the reciprocal of (κ) can be taken as the eddy size.

In general, $E(\kappa)$ is a function of energy dissipation rate ε, kinematic viscos-

ity ν, integral scale l, wavenumber κ and the mean strain rate S. As part of his universal equilibrium theory, Kolmogorov also made the hypothesis that for very large Reynolds number, there is a range of eddy sizes between the largest and smallest for which the cascade process is independent of the statistics of the energy-containing eddies (so that S and l can be ignored) and of the direct effects of molecular viscosity (so the ν can be ignored) [35]. The idea is that a range of wavenumbers exists in which the energy transfered by inertial effects dominates, therefore $E(\kappa)$ depends only upon ε and κ. On dimensional grounds, he thus concluded that

$$E(\kappa) = C_K \varepsilon^{2/3} \kappa^{-5/3}, \quad \frac{1}{l_{int}} \ll \kappa \ll \frac{1}{\eta} \tag{2.11}$$

where C_K is the Kolmogorov constant. This is Kolmogorov's famous $\kappa^{-5/3}$ law. Because inertial transfer of energy dominates, Kolmogorov identified this range of wavenumbers as the **inertial subrange**. The existence of the inertial subrange has been verified by many experiments and numerical simulations [39, 40]. The $\kappa^{-5/3}$ law is so well established that, as noted by Rogallo and Moin [40], theoretical or numerical predictions are regarded with scepticism if they fail to reproduce it. In fact, analysis of the energy spectrum can be used to quantify MILES.

2.2.4 Direct Numerical Simulation (DNS)

A direct numerical simulation means a complete time-dependent solution of the Navier-Stokes and continuity equations. The value of such simulations is obvious. From a practical standpoint, computed statistics can be used to test proposed closure approximations in engineering models [41]. At the most fundamental level, they can be used to obtain understanding of turbulence structure and processes that can be of value in developing turbulence control methods (e.g., drag reduction) or prediction methods [12]. If conducted with careful attention to numerical methods of at least second-order accuracy, it can be judged as sufficiently reliable to be considered as data [42] (with a demonstration of convergence with numerical resolution). All of these comments assume the DNS is free of significant numerical, and other forms of error. This is a nontrivial consideration, and the primary

concerns in DNS are related to numerical accuracy, specification of boundary and initial conditions.

With DNS, it is important to make optimal use of available computer resources. If one is to avoid using external memory devices with slow access time, the number of grid points to be used has to be estimated beforehand. This will also determine the maximum Reynolds number of the simulating flow. In principle, the grid must be fine enough to resolve the smallest eddies whose sizes are of order of the Kolmogorov length scale η. For an incompressible channel flow with a stretched mesh to the wall, Rogallo and Moin [40] quote the following relationship:

$$Number\ of\ nodes \approx (6\mathrm{Re}_m)^{9/4} \tag{2.12}$$

where Re_m is the Reynolds number based on the half width of the channel. It can be seen that the number of nodes increases faster than the square of the Reynolds number.

A further constraint on the ability of modern computers to do DNS is the time needed. The timestep in the computation Δt should satisfy two conditions: (i) it has to be smaller than the Kolmogorov time scale

$$\tau_K = (\nu/\varepsilon)^{1/2} \tag{2.13}$$

and (ii) it has to be small enough to stabilise the numerical scheme [43]. To satisfy the criteria (ii), as the grid resolution increases the allowable time step decreases. For the channel flow quoted above

$$\Delta t \propto (\mathrm{Re}_m)^{-11/4} \tag{2.14}$$

For these reasons, DNS is restricted to flows of such low Reynolds number that they are only of academic interest at present. Even though the progress of modern computers has been remarkable the prospect of achieving speed to solve problems of engineering interest in the near future is very slim.

2.2.5 Large Eddy Simulation (LES)

As described in Section 2.2.1, turbulent flows contain a wide range of length and time scales. The largest eddies are directly affected by the boundary conditions and must be computed. By contrast, the small-scale turbulence is more nearly isotropic and has universal characteristics; it is thus more amenable to modelling. Therefore a simulation which treats the large eddies more accurately than the small ones make sense; large eddy simulation (LES) is such an approach.

LES explicitly resolves turbulent scales much larger than the Kolmogorov length scale. Therefore much larger grid size and timestep can be used than are possible in a DNS. The unresolved small scales are modelled but since they are nearly isotropic one can attempt to use relatively simple models. The modelling of the small scale eddies is called *Subgrid-Scale (SGS) modelling* since the eddies are smaller than the filtering grid size. Higher accuracy can be achieved by reducing the grid size, increasing the number of mesh points and decreasing the computational timestep but the price paid is increased computational time. If the grid size is reduced to the size of the Kolmogorov length scale there is no need to model the subgrid scale and it becomes a DNS described in the previous section.

In general, because it is more accurate, DNS is to be preferred whenever it is feasible. LES is the preferred method for flows in which the Reynolds number is too high or the geometry is too complex for the application of DNS. LES cannot simulate transition correctly where small eddies plays a greater role in the development of turbulence. Since the smallest resolved eddy is much larger in LES than DNS, transition will occur later if it does occur at all. This is true for MILES as well. Therefore if the transition is to be simulated properly DNS has to be used.

LES still requires large computer resources, although it is more economical than DNS (typically requires 5 to 10% of the CPU time needed for DNS [34]). It is too expensive to be used as an every-day engineering tool but the list of its applications are growing fast. For simple flows, Ferziger [32] observed that LES is more expensive than more conventional Reynolds stress modelling by only a

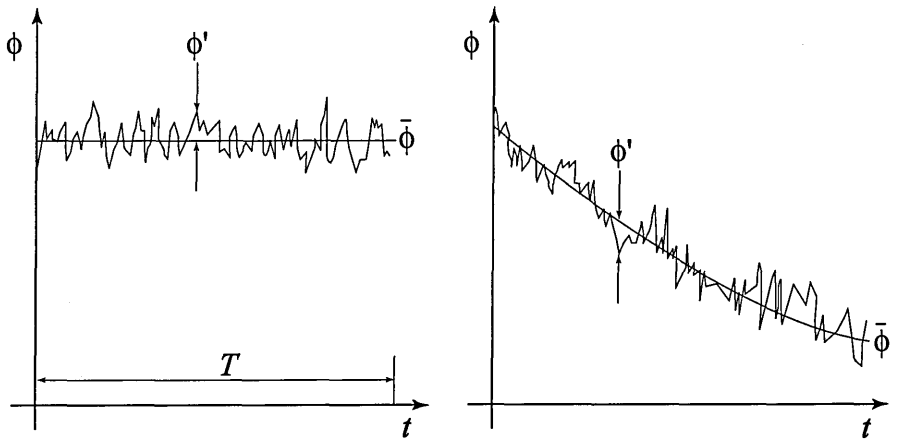

Figure 2.5: The averaging for a statistically steady flow (left) and ensemble averaging for an unsteady flow (right)

factor of two or three in many cases. Some claim that the LES technique, especially for incompressible flow, is coming to a matured state [44].

LES will be looked at in more detail in Section 2.3.

2.2.6 Reynolds-Averaged Navier-Stokes (RANS) Models

Even though DNS and LES produce a wealth of data with no or relatively simple modelling, these methods demand a great deal of computational power and/or time. If the computational power needed for these methods is not available and (as in many engineering problems) if only few quantitative properties of a turbulent flow, such as the average forces on a surface (and, perhaps, its distribution) or average velocity at a point, need to be calculated, a Reynolds-Averaged method is more appropriate.

In a statistically steady flow, every variable can be written as the sum of an average value and a fluctuation about that value:

$$\phi\left(x_i,t\right) = \overline{\phi}\left(x_i\right) + \phi'\left(x_i,t\right) \tag{2.15}$$

where

$$\overline{\phi}\left(x_i\right) = \lim_{T\to\infty} \frac{1}{T} \int_0^T \phi\left(x_i,t\right) dt. \tag{2.16}$$

Here t is the time and T is the averaging interval. This interval must be large compared to the typical time scale of the fluctuations; thus, $T \to \infty$ (Figure 2.5). If T is large enough $\overline{\phi}$ does not depend on the time at which the averaging is started.

If the flow is unsteady, time averaging cannot be used and must be replaced by ensemble averaging (Figure 2.5).

$$\overline{\phi}(x_i) = \lim_{N\to\infty} \frac{1}{N} \sum_{n=1}^{N} \phi_n(x_i, t) \tag{2.17}$$

where N is the number of members of the ensemble (an imagined set of flows in which all controllable variables are identical) which must be large enough to eliminate the effects of the fluctuations. This type of averaging can be applied to any flow and is called *Reynolds averaging*. The result of applying it to the Navier-Stokes equations is the Reynolds-averaged Navier-Stokes (RANS) equations.

From Equation 2.16, it follows that $\overline{\phi'} = 0$. Thus, averaging any linear term in the conservation equations simply gives the identical term for the averaged quantity. From a quadratic nonlinear term we get two terms: the product of the average and a covariance

$$\overline{u_i\phi} = \overline{(\overline{u}_i + u_i')(\overline{\phi}_i + \phi_i')} = \overline{u}_i\overline{\phi} + \overline{u_i'\phi'}. \tag{2.18}$$

The last term is zero only if the two quantities are uncorrelated; this is rarely the case and, as a result, the conservation equations contain terms such as $\rho\overline{u_i'u_j'}$ called the *Reynolds stresses* and $\rho\overline{u_i'\phi'}$, known as the *turbulent scalar flux.*

The presence of Reynolds stresses and turbulent scalar flux in the conservation equations mean that the latter are not closed, that is to say, they contain more variables than there are equations. Closure requires some approximations, which usually take the form of prescribing the Reynolds stress tensor and turbulent scalar fluxes in terms of the mean quantities. The approximations introduced are called *turbulence models.*

The complexity of turbulence makes it unlikely that any single model will be able to represent all turbulent flows. Further research to find more accurate models

is still on-going. Overall, turbulence models should be regarded as engineering approximations rather than scientific laws.

2.3 Large Eddy Simulation

Three categories of LES are described in this section with emphasis on Monotone Integrated Large Eddy Simulation.

2.3.1 Conventional Large Eddy Simulation

Compared to incompressible flows, compressible flows have received little attention from LES researchers. Apart from the additional equations that have to be solved, compressible flows normally involve larger Reynolds numbers. This means that smaller grid size and timestep are required which results in a higher computational cost. As computers become more powerful more researchers will become interested in compressible flows. Most research to date has concentrated on complex flow phenomena on simple geometries. This includes homogeneous turbulent flows [41,45], high-speed boundary layers [15], and mixing layers [46] to mention a few.

Recently, developments of so-call dynamical subgrid scale models [16,17] enable us to calculate the model coefficients, rather than input them *a priori* depending on flow situations. Even though this technique is still in its infancy, results have been very promising. More complicated geometries may be simulated with the dynamic SGS models, e.g. turbulent flows in a channel with a surface-mounted two-dimensional obstacle [47] and the results have been very encouraging. Chapman [48] has speculated that LES together with a dynamic SGS model could become practical for use in aerospace design. Nevertheless the development of dynamic SGS models is still continuing [49] and it is also expected that improved models will be proposed in the near future.

Multiple mesh techniques for LES are also getting attention, in order to improve the efficiency of the numerical method. The meshes are geometrically similar to those used in multi-grid algorithms, though the motivation is different and

the implementation and effects of the technique are quite distinct. Voke [50] has claimed that the cost ratio between the multi-mesh simulation performed on channel flow using two levels of mesh and an equivalent simulation performed wholly on a fine mesh [51] is 7.5. This technique is still new and has yet to be tested with the dynamic SGS model and more complicated geometries.

Another way to achieve a speedup of LES is to parallelise serial code to utilise a parallel computing system (See Chapter 2.4 for parallel computing). The use of so-call virtual parallel machines (a cluster of workstations acting like a bigger parallel machine) can be a very economical way to achieve speedup. Recent developments of message passing libraries such as Message Passing Interface (MPI) [52] have blurred the boundary between real and virtual parallel machines. There is little difference between the parallel code required for real and virtual parallel machines and they can be ported with minimal changes. Robichaux *et al.* [53] successfully performed LES using a Connection Machine-2 (CM-2) with 32 processors. The code was parallelised with a single-instruction multiple-data (SIMD) concept of data parallelism [54]. They claimed that in comparison with similar simulations on a Cray-2 (one processor), the execution time was reduced by a factor of 2.1. Not all numerical algorithms are easily parallelisable and the cost of parallelising serial code has to be weighed against other possible acceleration methods.

2.3.2 Very Large Eddy Simulation (VLES)

It is true that LES saves considerable computational time compared to DNS, but it still demands a great deal of computational power. Although developments in computer hardware and numerical algorithms enable us to push the Reynolds number of simulated flow even higher, some of the most interesting aeronautical problems are out of reach with conventional LES, e.g. flow around supersonic aircraft, where experiments are expensive and difficult to conduct. However, some simple high Reynolds number flows of engineering interest are in reach of present computational power with a subset of LES, sometimes known as Very Large Eddy Simulations (VLES).

In conventional LES (described in previous section), the large scales are turbulence productive anisotropic eddies which were calculated directly using a filtered Navior-Stokes Equation. The small scales are dissipative isotropic eddies which were modelled with simple SGS models. To insure that the division of the scales are as described above, the filter size should be small enough to simulate some of the Inertial Subrange (Figure 2.4).

In VLES, the filter size is larger than conventional LES and the SGS scales contain some of the larger anisotropic structures that contribute significantly to the Reynolds Stress generation in the flow field [55]. These scales must be modelled separately from the smaller dissipative scales if the SGS model is to be realistic. This is particularly important in wall bounded flows close to the surface, where the smallest scales play a significant part in generating Reynolds stress.

Despite this, a number of flows have been successfully simulated where Reynolds-Averaged Navier-Stokes Models have failed. These include the upwash fountain under a VTOL aircraft [56] and compression ramp flow [12].

Speziale [57] also suggested that the Reynolds stress models for Reynolds-Averaged Navier-Stokes (RANS) computation can be used as a subgrid scale model for VLES blurring the boundaries of DNS, LES, RANS.

2.3.3 Monotone Integrated Large Eddy Simulation (MILES)

The MILES approach was introduced by Boris *et al.* [58] in 1992 and has been gradually gathering support.

Comparison between traditional LES and MILES

In traditional LES for every dynamical variable u (e.g. pressure) one explicitly defines a smoothed or 'filtered' variable $\overline{u}$ defined as an appropriate local average of u, so that $\overline{u}$ is a smoother function that only follows the large scale structure of u and lacks the small scale fluctuations. One then derives from the Navier-

Stokes equations, a set of equations satisfied by the corresponding filtered variables. These equations are *exact*, as no approximation has been made so far. However, the LES equations are unclosed. At this stage an approximate model is chosen for the unclosed terms to give a set of model LES equations. These equations are then solved numerically.

Another approach to LES is to use the inherent dissipation of numerical schemes to mimic transfer of energy to smaller scales and MILES takes this approach.

The main advantage of the traditional approach is it enables one to address the issues of subgrid modelling and numerical methods separately thereby enabling one to deal with two very difficult problems one at a time. This has a distinct advantage for any theoretical analysis of either subgrid modelling or numerical errors. The MILES approach may be cheaper (no overheads for subgrid models) when it works. However, when it doesn't work it is more difficult to figure out exactly what went wrong (e.g. trying to improve grid resolution to resolve mean variables may trigger an instability because of decreased numerical dissipation.)

The traditional LES approach also has disadvantages. To maintain low numerical dissipation, a high-order spectral central difference scheme has to be used. This numerical method is very difficult to be applied to a complex geometry. Therefore much of the research effort has been concentrated on complex flows over simple geometries like homogeneous turbulence [41, 45], mixing layer [59] or flow over flat plates [22, 24, 27, 30]. Finite difference [46] and finite volume methods [45, 60] can be applied to more complicated geometries but they suffer from higher numerical dissipation.

Also, modelling of subgrid-scale eddies introduces errors into a solution (see next section). Boris [61] lists five properties that an ideal subgrid model should have:

1. It should apply without restriction to the fluid dynamics model being solved macroscopically, e.g., include compressibility, high Mach number, multi-species effects, etc., as appropriate to the problem at hand.

2. It should satisfy the global conservation laws of the system as integrated over the unresolved scales.

3. It should minimize the contamination of macroscopic scales by the inaccurately resolved flow structures on the grid scale and by the numerical filtering. This allows the resolvable linear and nonlinear processes which physically drive the subgrid dynamics to be calculated as accurately as possible.

4. It should accomplish the physical mixing and averaging expected of the complex but unresolved flows on the correct macroscopic space and timescales.

5. It should match smoothly onto the resolved macroscale solutions at each point in space, even for variable grid size. The effects of all scale lengths, whether modelled or resolved, should be included exactly once.

In addition, several conditions have to be satisfied when a high Reynolds number fluid dynamic system is being modelled to ensure that the SGS model approach makes sense. These conditions are based in part on the self evident assertions made above and in part on distinctions between LES and VLES as introduced by Reynolds [62]:

1. The problem being solved is such that the macroscopic LES model can resolve the dynamics of the energy containing turbulence-driving scales.

2. The macroscopic convection velocities are sufficiently larger than the unresolved turbulence velocities so that small-scale turbulent motion of material, mass, momentum, and energy accounts for a small portion of the global transport in the problem.

3. Unresolved "turbulent" diffusion dominates molecular transport or else the molecular effects are explicitly included in the LES model equations.

While some of these requirements can be met by relatively simple models, it is not easy to find a model to satisfy all of them. Boris [61] argued that errors introduced by modelling could be a worse enemy than numerical dissipation introduced by

the MILES approach.

Reynolds [62] argued that in real turbulence the dissipation is set by the non-linear cascade in the inertial range, and the only role of viscosity is to set the smallest scales of motion. Therefore the MILES approach could produce useful results if the dissipative process is confined to small scales. In this case the simulation could produce useful results in flows where Reynolds number effects are unimportant and separation points are set by sharp edges rather than by viscous fluid dynamics. Indeed, Sagaut *et al.* [63] produced reasonable results for flow behind a backward-facing step without any SGS modelling. But Reynolds also stressed that because there is no relationship between numerical viscosity and fluid viscosity, the MILES approach cannot predict important Reynolds number effects.

Held and Fuchs [64] reports that while the mean quantities can be predicted by MILES as well as conventional LES with SGS modelling, higher order statistics such as the Root Mean Square (RMS) of the skin friction coefficient fluctuations, can be significantly different.

Numerical dissipation

Numerical dissipation in a scheme is not always unwanted. With an appropriate amount it can make calculations more stable and converge faster. In the case of a central difference scheme which has very little numerical dissipation, some kind of artificial viscosity has to be added to make the calculation stable. This has been the traditional approach of LES where the non-dissipative central difference schemes were used together with SGS modelling to put just enough viscosity to mimic the physical dissipation by small eddies (see Section 2.3.1). Therefore, one of the difficulties of conventional LES is to minimize the numerical dissipation. But all numerical methods have some kind of numerical dissipation. In fact, there are a number of sources for numerical dissipation and it is hard to see which one contributes more than the others.

Discretization of a governing equation will introduce truncation errors and

therefore numerical dissipation which can be reduced by using higher order schemes with more computational resources. Some numerical schemes are more dissipative than others. For example, an upwind scheme is more dissipative than a central difference scheme. The spectral method [41,65] was popular because it calculates the lower wavenumber regions more accurately than other methods but it also has the problem of aliasing error associated with Fourier transformations.

Ghosal [66,67] has tested some numerical schemes and increased understanding of the contribution of numerical errors as compared to the subgrid force[1]. He used a finite difference method together with dynamic subgrid scale modelling. He categorized the errors into three types: the subgrid modelling error, the truncation error and the aliasing error. The origin of the subgrid modelling error is clear, it appears because the subgrid model does not exactly equal the true "subgrid stress". Since we do not know what the true subgrid stress is, any theoretical treatment is difficult. But Ghosal was able to show that the size of the errors were comparable with the subgrid force in low order schemes. Even for an eighth order central difference scheme calculation, the subgrid force in lower wavenumbers was comparable with errors. This means subgrid-scale modelling can be hindered by the errors.

From a numerical simulation of an axisymmetric spatially developing jet, Olsson and Fuchs [68] estimated

- the subgrid terms of the dynamic SGS model
- the articial viscosity by upwind scheme
- the convective term
- the viscous term
- the truncation errors

by comparing 2nd and 4th-order central difference scheme and 3rd-order upwind scheme. They observed that:

[1]The subgrid force is the term that appears in the LES equation, which with homogeneous filters is the divergence of the subgrid stress.

1. the equations are dominated by the convective term,
2. the $O\left(h^4\right)$ truncation errors are less than all other terms,
3. the $O\left(h^2\right)$ truncation errors are of the same order of magnitude as the subgrid terms,
4. the artifical viscosity terms are at least of the same order of magnitude as the subgrid terms.

From these observations it seems adequate to use schemes of $O\left(h^4\right)$ but it is questionable to use $O\left(h^2\right)$ schemes, due to the size of the truncation errors as compared to the subgrid terms. They also observed that the artificial viscosity was located at the right place and was of the right order of magnitude. Therefore they concluded that the artifical viscosity of $O\left(h^3\right)$ upwind schemes dissipate energy like an SGS-model.

Nevertheless, as Ducros *et al.* [69] pointed out, to be able to solve complex geometries, lower order methods have to be used. In fact some have shown that it is possible to obtain reasonable LES results with lower order methods. Examples includes turbulent mixing layers by Ansari [59], shock/turbulence interaction by Ducros [69], compressible homogeneous and isotropic decaying turbulence by Vreman [45].

Shock-Capturing Schemes and MILES

In simulations of supersonic flow with shock-waves, the use of shock-capturing methods will contribute to numerical dissipation. The MUSCL scheme (see Section 4.2.3) has to use a limiter in the presence of shock-waves and this will also increase numerical dissipation.

Garnier *et al.* [70] tested several shock-capturing schemes (Jameson, TVD-MUSCL, ENO) to evaluate the relevance of the use of such schemes in the context of LES. Although some known physical trends were respected, it was found that the small scale accuracy of the simulated flow suffered from high numerical damping. They also compared the numerical dissipation of shock-capturing schemes

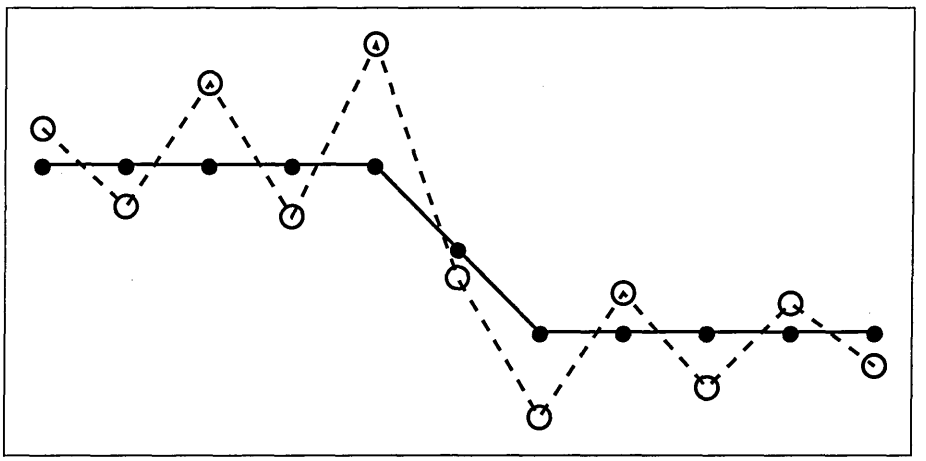

Figure 2.6: Illustration of the numerical phenomenon of spurious oscillations near high gradients

with the dissipation provided by the SGS models (Smagorinsky, Dynamic), and concluded that the addition of explicit SGS models to shock-capturing schemes is unnecessary and inconvenient. It was also clear that different limiters have different numerical dissipation [70, 71] and some of them are too dissipative to be used in LES. They also recommended the development of a sensor able to distinguish a turbulent fluctuation from a shock.

Ducros *et al.* [69] developed this kind of sensor for triggering artifical dissipation to perform LES of the shock/turbulence interaction. They claimed that the sensor allows the global scheme to capture the shock and to predict a right decay of turbulence kinetic energy in regions out of the shock.

Yee *et al.* [71] proposed a low-dissipative high-order shock-capturing method using characteristic-based filters. The method can be adapted to a second or third-order total variation diminishing (TVD) or an essentially non-oscillatory (ENO) scheme with little computational penalty. They reported that higher accuracy can be achieved with fewer grid points when compared with standard TVD or ENO schemes, and the modified scheme is capable of sustaining turbulence even when shock-waves are not present.

Godunov [72] proved that the numerical phenomenon of spurious oscillations

will occur near high gradients with use of any linear method of second or higher order of accuracy. A limiter is required to eliminate the oscillations depicted in Figure 2.6, where the full line denotes the solution without oscillation and the dotted line denotes the numerical solution obtained by some linear method of second or higher order of accuracy. Different methods will produce different patterns for the oscillatory profile. A limiter function is to add numerical dissipation near the high gradients.

One popular family of limiters [72] is the minimum-modulus (minmod) type which are activated suddenly in regions of rapidly changing gradients. Smooth limiters [73] are also popular. Smooth limiters have the disadvantage of a higher spectral radius than the minmod limiters, however the solution converges better on steady calculations. Later developments of limiters have flooded into the literature and have created much debate. Most of the improvements made have been problem dependent.

Yee *et al.* [71, 74] tested several slope limiters in an attempt to eliminate the unwanted oscillation and found that some minmod type limiters work well with LES while some limiters were too dissipative to be used in LES, i.e. the calculation cannot sustain turbulence. Garnier *et al.* [70] also tested minmod limiters and found similar results.

Previous MILES Calculations

Even though there have been other publications which can be classified as MILES (Kawamura and Kuwahara [75] in an incompressible case and Porter *et al.* [76] in a compressible case), Boris *et al.* [58, 77] is generally credited for the introduction of MILES. Boris *et al.* used the Flux-Corrected Transport (FCT) scheme in a cylinderical jet entrainment simulation to show that some high-resolution monotone computational fluid dynamics algorithms have intrinsic subgrid turbulence models built into them. Other MILES applications using FCT to freejets include axisymmetric (Grinstein *et al.* [78]) and rectangular (Grinstein and Dovore [79]) jet at moderately high Reynolds number.

Porter *et al.* [76] were able to show that MILES is capable of capturing the dominant inertial subrange of the energy spectrum with a self-similar simulated dissipation range. Their calculation used the piecewise parabolic method (PPM) for MILES application. The supersonic confined mixing layer calculation of Gathmann *et al.* [80] is another example of the use of PPM.

Comparative studies with MILES and other LES models of nonreactive shear flow by Fureby [81, 82] and reactive shear flows by Möller *et al.* [83] suggest that LES results are fairly resilient to SGS modelling provided that the cutoff wavelength is within the inertial subrange.

Other MILES simulations include supersonic base flow by Fureby *et al.* [84] and separated transonic flow around a wing section by Held and Fuch [64].

2.4 Parallel Computing

2.4.1 Computer Architectures in High Performance Computer

Advancement in computer hardware has been phenomenal over the past three decades. Since the introduction of the IBM 650 in 1953 the relative computational cost has been decreasing at a rate of 1/10 every 8 years [85]. The earlier high-speed digital computers were serial machines, capable of one computational operation at a time. The finite speed of electrons, close to the speed of light, poses an inherent limitation on the ultimate execution speed of such serial computers. In recent years, the rapid growth in capability of single processor computers has slowed down [86]. To avoid this limitation two types of computer architecture are getting more attention.

Vector architecture is a configuration that allows a string of identical operations on an array of numbers simultaneously, thus saving both time and memory.

Parallel architecture is a configuration which contains more than two fully functional central process units (CPUs), each of which can handle different instruction and data streams and which can execute separate parts of a program

simultaneously, working independently or in concert with other CPUs.

Vector computers were widely used in the 80's and even though parallel computers came onto the scene much later, they are spreading more rapidly. Because the cost of developing ever more powerful processors is high, it makes sense to use widely available serial processors to make a parallel computer. CRAY vector computers (e.g. CRAY YM-P) were traditionally regarded as supercomputers, more recently massively parallel computers, developed using existing DEC Alpha chips (e.g. CRAY T3D), have taken over this role. These massively parallel computers have a large number of fully functional processors and are designed to accommodate up to 2048 processors giving a peak performance of 300GFlops.

The idea of creating a so-called virtual parallel machine (a cluster of workstations acting like a bigger parallel machine) is nothing new. The recent development in communication libraries (e.g. MPI, PVM) make the creation of such machines easier and simpler than before. The concept of a virtual parallel machine, which is also known as Distributed Processing, is very much similar to a real massively parallel machine with distributed memories i.e. each processor has its own memory. However, in a virtual parallel machine, the communication between the processors uses ethernet rather than special inter-connection in a real parallel machine.

A Beowulf system consists of commodity PCs connected together with a private ethernet connection. They don't have fast inter-connectivity like specially designed parallel machines but one can be built at a fraction of the cost. Also they can be upgraded easily and cheaply when new hardwares are introduced.

As far as programming strategies are concerned, there is no difference between real and virtual parallel machines. The main difference is that the communication time between the processors in a virtual parallel machine is much greater than in some real parallel machines where communication is done through shared memories or very fast links. The ratio of communication to computational time is very important in a parallel machine especially in a virtual parallel machine. This issue

will be discussed in more details in Section 2.4.3.

2.4.2 Advantages and Disadvantage of Using a Parallel Computer in CFD

Even with all the developments in computer hardwares and numerical methods, the computational demand in CFD still far exceeds the present capability of any supercomputers. All the effort on turbulence modelling is to mimic the unresolved part of turbulence and this proves to be a very difficult task. Turbulence modelling is unnecessary if we could resolve the turbulence to the Kolmogorov scale (See Section 2.2.4 for DNS). At present, DNS of flows over very simple geometries are possible with lots of computational time on some supercomputers. This is not a satisfactory situation (not many people have easy access to a supercomputer to do very large calculations).

One very promising development which reduces the computational cost is the massively parallel computer. This has been encouraged by the development of communication libraries (see Section 2.4.4 for current standards and libraries). Initially most parallel computers had (and still have) their own communication libraries which were incompatible with other parallel computers. Efforts to develop communication libraries between workstations has merged with the one in parallel computers. Now, it is common that a supercomputer supports other communication libraries together with their own. Use of a virtual parallel computer (see previous section) can dramatically reduce computational cost. In many organisations where large numbers of workstations are used, these resources can be utilised by creating a virtual parallel computer to run a large job over night. Another benefit is the access to larger memories. Since memory is still expensive and using external memory system (swap space) is very slow, this is a very good economic incentive.

The biggest disadvantage of using a parallel computer is that not all numerical methods are suitable for parallelisation, especially some acceleration algorithms. Sometimes new numerical methods have to be developed for the parallel programming and the numerical libraries required for parallel programming are not

yet widely available. There are a few very good textbooks devoted to this subject and the interested reader should refer to Bertsekas and Tsitsiklis [87], for example. This complication also arises if a serial code has to be parallelised.

In a virtual parallel computer each machine is exposed to the usual malfunctions. Therefore it is often more vulnerable to malfunction than dedicated parallel machines. It is possible to check the performance of a virtual parallel machine and to move loads between the constituting computers during the calculation, but to do this programming complexity is increased. This task management might be automated in future.

There is also an initial task of learning a particular communication library.

2.4.3 Parallel Performance

Speedup

Speedup using a parallel computer S_p is defined as:

$$S_p = \frac{T_s}{T_p} \tag{2.19}$$

where T_s is the time taken to execute a program on a single processor and T_p is the time taken by p processors for an equivalent parallel code. It can be expressed as:

$$S_p = \frac{T_{s(1)}}{T_{s(n)} + T_p + T_o} \tag{2.20}$$

where $T_{s(1)}$ is the time taken to execute a serial code, $T_{s(n)}$ is the time taken to execute a unparallelisable part of the parallel code, T_p is the time taken to execute the parallelised part of the parallel code and T_o is the overhead time for communication. The overhead time includes the time to start up communication (start-up latency) and the time to communicate actual data. For more details on start-up latency of communication see Section 5.2.2.

Parallel Efficiency

Efficiency of a parallel code E_p is defined as:

$$E_p = \frac{S_p}{np} \times 100 \tag{2.21}$$

where S_p is the speedup and np is the number of processors.

Peak performance of a parallel computer is calculated as the speed of a single processor multiplied by the number of processors in the system. The peak performance does not take account of the communication time i.e. the overhead for parallel computing.

Amdahl's Law

Amdahl's Law states that the speedup of a parallel code against the serial code is limited by the fraction of the code that cannot be parallelised. Mathematically this can be written as:

$$S_p^{max} = \frac{1}{(1-f_s)\frac{1}{np} + f_s}, \tag{2.22}$$

where S_p^{max} is the maximum speed up of a parallel computer and f_s is the fraction of code which is serial. For example, if we have a parallel code of which 20% of the code in terms of time is serial (i.e. $f_s = 0.2$) then for 2 processors the maximum speedup possible will be 1.67 and for 4 processors it will be 2.5. Corresponding efficiencies will be 83.5% and 62.5% respectively. It can therefore be seen that as the number of processors increases the speed up decreases and the serial code dominates the efficiency of the parallel code. If the problem size increases then the serial fraction in terms of time decreases as a proportion of the time taken. Therefore as the problem size increases the effect of Amdahl's Law becomes negligible.

2.4.4 Current Major Communication Standards

Currently, there are three major communication standards or libraries which support parallel computing: High Performance Fortran, Parallel Virtual Machine and Message Passing Interface. Which one of these is the best choice for parallel computing is still debatable.

High Performance Fortran (HPF)

There have been several extensions of Fortran77 for parallel computing and HPF is one of the latest standards. It is an extension of Fortran90 to include:

- data distribution features,
- data parallel execution features,
- extended intrinsic functions to be used with parallel machines.

Even though it has its limitations, Fortran is the most well-known computer language among the scientists and engineers alike. The number of scientific programs written in Fortran far exceeds those written in any other languages. Fortran90 is fully backward compatible with, and can use subroutines written in Fortran77 (you can also call Fortran subroutines from other languages e.g. C).

HPF is still in an early stage of development, and it is still to be seen whether it will be widely used. For a history overview and current status of HPF see Reference [88].

Parallel Virtual Machines (PVM)

PVM is not a standard but provides a communication library and a programming environment for parallel programming on various machines including virtual parallel machines. It is in the public-domain and therefore free of charge. Although being general in nature it is slow. It can be used with Fortran and C.

For a user's guide and tutorial see Reference [89] and for the complete manual see Reference [90].

Message Passing Interface (MPI)

Over the last ten years, numerous message passing systems have been created all using very similar concepts but with some variations. MPI is a standard that takes the best features of all present message passing systems including PVM, PARMACS, Zipcode and PICL. The aim is to establish a message passing standard that is both portable and easy to use.

The development of MPI has been so fast that an MPI Forum is already concentrating on MPI-2 before the MPI-1 standard has been published as a book [91,

92]. Only Fortran77 and C bindings are specified with MPI-1 but in MPI-2, Fortran90 and C++ bindings will also be added.

Currently there are three major implementations of MPI:

- *MPI Chameleon (MPICH)* from Argonne National Laboratory & Mississippi State University,
- *Local Area Multicomputer (LAM)* from the Ohio Supercomputer Center,
- *Common High-level Interface to Message Passing (CHIMP)* from the Edinburgh Parallel Computing Centre.

Among them they cover most of the popular workstations and some of the supercomputers including the Cray-T3D. Even the PC can be used to create a virtual parallel machine. Mississippi State University also developed the *UNIFY* implementation which allows both PVM and MPI. The list of implementations is still growing, but many of the current implementations are in the public domain. For the history, overview and current status of MPI see Reference [93].

Chapter 3

Governing Equations

3.1 Conventional Large Eddy Simulation (LES)

3.1.1 Filtering

In LES, the velocity field that contains only the large scale components of the total field is calculated explicitly while small scale components are modelled or calculated implicitly. The large scale components are isolated from small scales by filtering.

Basic Concept of Filtering

The values of flow properties at discrete points in a numerical simulation represent averaged values. To see this explicitly, consider the central-difference approximation for the first derivative of a continuous variable, $u(x)$, in a uniform grid with points spaced a distance h apart. We can write this as follows:

$$\frac{u(x+h)-u(x-h)}{2h} = \frac{d}{dx}\left[\frac{1}{2h}\int_{x-h}^{x+h} u(\xi)d\xi\right] \tag{3.1}$$

This shows that the central-difference approximation can be thought of as an operator that *filters out scales smaller than the grid size* [34] i.e. the *filter width* Δ_f is the same as the *grid size* Δ_g. Furthermore, the approximation yields the derivative of an averaged value of $u(x)$. Having the same size for Δ_f and Δ_g is convenient and sometimes the only way for complex geometries. It is often assumed that Δ_f and Δ_g are the same and one speaks of the two terms interchangeably. However, if all scales up to Δ_f are to be resolved correctly Δ_g has to be several times smaller.

This could severely increase the computational cost. For example, if $\Delta_f/\Delta_g = 8$ the number of grid points are increased by a factor of $8^3 = 512$ and the timestep is decreased by another factor of 8; therefore the total computational cost will be increased by a factor of 4096. Ghosal [67] and Vreman *et al.* [94] both separately concluded that $\Delta_f/\Delta_g = 2$ with the fourth-order or higher discretization scheme should be used to minimise the numerical error. However, these two terms will be used interchangeably and denoted as $\overline{\Delta}$ in this thesis.

Leonard [95] defines a generalised filter as a convolution integral:

$$\overline{u}_i(\mathbf{x}) = \iiint G(\mathbf{x}-\mathbf{x}')\, u_i(\mathbf{x}')\, d\mathbf{x}' \tag{3.2}$$

where the large-scale or resolvable-scale component of u_i is denoted by $\overline{u}_i$ and is defined by a convolution of u_i with a filter function $G(\mathbf{x}')$. The small-scale or subgrid-scale (SGS) component is defined by:

$$u'_i = u_i - \overline{u}_i \tag{3.3}$$

Filters

Many kinds of filters have been proposed and used, some of which are neither isotropic nor homogeneous. In all cases however, the filter introduces a scale Δ that represents the smallest turbulence scale allowed by the filter. Some popular filters are listed here.

Box Filter or Top-Hat Filter The simplest type of filter is the volume-average box or top-hat filter used by Deardorff [96]:

$$\overline{u}_i(\mathbf{x}) = \frac{1}{\overline{\Delta}^3} \int_{x_1-\frac{1}{2}\Delta x_1}^{x_1+\frac{1}{2}\Delta x_1} \int_{x_2-\frac{1}{2}\Delta x_2}^{x_2+\frac{1}{2}\Delta x_2} \int_{x_3-\frac{1}{2}\Delta x_3}^{x_3+\frac{1}{2}\Delta x_3} u_i(\mathbf{x}')\, d\mathbf{x}' \tag{3.4}$$

where $\overline{\Delta}$ is the *filter width* given by:

$$\overline{\Delta} = \left(\Delta x_1 \Delta x_2 \Delta x_3\right)^{1/3} \tag{3.5}$$

In terms of the filter function, the box filter is:

$$G(\mathbf{x}-\mathbf{x}') = \begin{cases} \frac{1}{\overline{\Delta}^3} & \text{for } |x_i - x'_i| \le \frac{\Delta x_i}{2} \quad (i = 1, 2, 3), \\ 0 & \text{elsewhere} \end{cases} \tag{3.6}$$

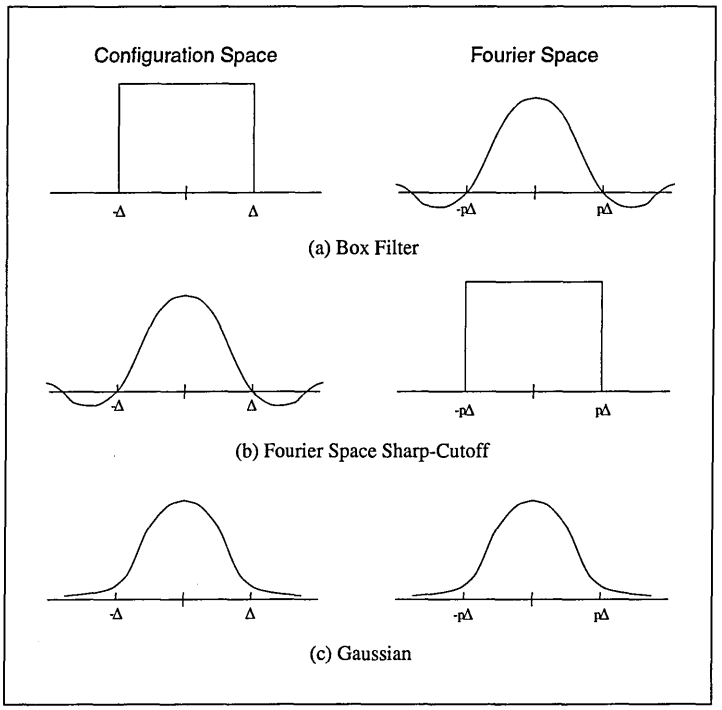

Figure 3.1: One-dimensional filters in physical, or configuration, space (left) and in transformed, or Fourier, space (right): (a) box, or top-hat, filter; (b) sharp-cutoff filter; (c) Gaussian filter

Sharp-Cutoff filter The Fourier transform of Equation (3.2) is:

$$\overline{\quad}_i(\kappa) = \quad(\kappa) \quad _i(\kappa) \tag{3.7}$$

where $_i$ and represent the Fourier transforms of u_i and G. The sharp-cutoff filter [97] is essentially the Fourier-space version of the box filter. Fourier spectral methods implicitly filter with

$$(\kappa) = \begin{cases} 1 & \text{for } |\kappa_i| \leq \kappa_c = \frac{2\pi}{\Delta} \quad (i = 1, 2, 3), \\ 0 & \text{elsewhere.} \end{cases} \tag{3.8}$$

In physical space, this filter is a damped sinusoid, with decreasing amplitude away from $\mathbf{x} = x'$; its characteristic width is defined as π/κ_c.

Gaussian filter Another widely used filter by those using spectral methods and Fourier transforms is the Gaussian filter. Its Fourier transform is also Gaussian, therefore it can be differentiated as many times as one likes in both spaces. It is

defined in physical space by:

$$G(\mathbf{x}-\mathbf{x}') = \left(\frac{6}{\pi\overline{\Delta}^2}\right)^{3/2} \exp\left\{-6\frac{(\mathbf{x}-\mathbf{x}')^2}{\overline{\Delta}^2}\right\} \tag{3.9}$$

or in Fourier space by:

$$(\kappa) = \exp\left(-\frac{\overline{\Delta}^2\kappa^2}{24}\right) \tag{3.10}$$

where $\overline{\Delta}$ is the filter width. The numerical factors have been chosen to make the second moment of this filter the same as that of a box filter of width $\overline{\Delta}$.

For compressible flow simulation a *Favre filter* is used with these filters. The Favre filter is defined as:

$$\boxed{\widetilde{F} = \frac{\overline{\rho F}}{\overline{\rho}}} \tag{3.11}$$

in an analogous manner to the Favre time average which has been used in the studies of compressible turbulent flows [38]. Now, the turbulent fields F are then assumed to be decomposed in the following way:

$$F = \widetilde{F} + F' \tag{3.12}$$

where F' is the fluctuation value.

Choice of a filter

If the governing equations are solved using a spectral method the shape-cutoff filter is the easiest filtering method to implement. Box and shape-cutoff filters have the difficulty that their Fourier transforms have negative regions; they are also difficult to differentiate. For this reason, some of those using spectral methods and Fourier transforms, e.g. the Stanford group [32], prefer to use the Gaussian filter. If the grid is non-uniform then the spectral method cannot be used and the box filter is generally used [47].

It is possible to use different filters in each grid direction. For example, for plane-channel flows, the Gaussian filter may be applied only in the horizontal directions and the finite differencing in the vertical direction implicitly supplies box

filtering [51].

3.1.2 Basic Equations

The compressible turbulent flow of an ideal gas is governed by the continuity, momentum and energy equations. Neglecting body forces, they are [85]:

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_k)}{\partial x_k} = 0 \tag{3.13}$$

$$\frac{\partial (\rho u_k)}{\partial t} + \frac{\partial (\rho u_k u_l)}{\partial x_l} = -\frac{\partial p}{\partial x_l} + \frac{\partial \sigma_{kl}}{\partial x_l} \tag{3.14}$$

$$\frac{\partial (E_t)}{\partial t} + \frac{\partial ((E_t + p) u_k)}{\partial x_k} = \frac{\partial (u_k \sigma_{kl})}{\partial x_j} - \frac{\partial \quad _k}{\partial x_j} \tag{3.15}$$

where u_k is the velocity component in the kth direction, p is the pressure and ρ is the density. E_t is the total energy i. e.

$$E_t = \frac{p}{\gamma - 1} + \frac{1}{2} \rho u_k u_k \tag{3.16}$$

where γ is the ratio of specific heats. The viscous stress tensor σ_{kl} is defined as:

$$\sigma_{kl} = -\frac{2}{3} \mu \frac{\partial u_j}{\partial x_j} \delta_{kl} + \mu \left(\frac{\partial u_k}{\partial x_l} + \frac{\partial u_l}{\partial x_k} \right) \tag{3.17}$$

and $_k$ represents the viscous heat flux vector which is given by:

$$_k = -k_t \frac{\partial T}{\partial x_k} \tag{3.18}$$

where T is the temperature and k_t is the thermal conductivity. μ is the molecular viscosity which is calculated with the Sutherland's law:

$$\frac{\mu}{\mu_0} = \left(\frac{T}{T_0} \right)^{3/2} \frac{T_0 + S}{T + S} \tag{3.19}$$

where where S is an effective temperature, called the *Sutherland constant*, which is characteristic of the gas. For air, $S = 199°\text{R}$, $T_0 = 491.6°\text{R}$ and $\mu_0 = 0.1716\text{mP}$.

Equation (3.13)–(3.15) has to be supplemented with the ideal gas equation of state:

$$p = \rho RT \tag{3.20}$$

where p is pressure, ρ is density, R is the gas constant and T is temperature.

To account for large density fluctuations in high-speed compressible flows, the Favre-filtering operation is employed, where the resolved velocity and temperature fields are written in terms of Favre-filtered quantities, which are defined in Equation (3.11).

After applying the spatial and Favre filter to Equation (3.13)–(3.15), the governing equations for the large-scale field are [45]:

$$\boxed{\frac{\partial \overline{\rho}}{\partial t} = \frac{\partial \left(\overline{\rho}\widetilde{u}_k\right)}{\partial x_k}} \tag{3.21}$$

$$\boxed{\frac{\partial \left(\overline{\rho}\widetilde{u}_k\right)}{\partial t} + \frac{\partial \left(\overline{\rho}\widetilde{u}_k\widetilde{u}_l\right)}{\partial x_l} = -\frac{\partial \overline{p}}{\partial x_k} + \frac{\partial \widetilde{\sigma}_{kl}}{\partial x_l} - \frac{\partial \tau_{kl}}{\partial x_l}} \tag{3.22}$$

$$\boxed{\frac{\partial \left(\widetilde{E}_t\right)}{\partial t} + \frac{\partial \left(\left(\widetilde{E}_t + \overline{p}\right)\widetilde{u}_k\right)}{\partial x_k} = \frac{\partial \left(\widetilde{u_k\sigma_{kl}}\right)}{\partial x_j} - \frac{\partial \widetilde{\ }_k}{\partial x_j} - \frac{\partial q_k}{\partial x_k}} \tag{3.23}$$

where

$$\widetilde{E}_t = \frac{\overline{p}}{\gamma - 1} + \frac{1}{2}\overline{\rho}\widetilde{u}_k\widetilde{u}_k \tag{3.24}$$

$$\widetilde{\sigma}_{kl} = -\frac{2}{3}\mu\frac{\partial \widetilde{u}_j}{\partial x_j}\delta_{kl} + \mu\left(\frac{\partial \widetilde{u}_k}{\partial x_l} + \frac{\partial \widetilde{u}_l}{\partial x_k}\right) \tag{3.25}$$

$$\widetilde{\ }_k = -\widetilde{k}_t\frac{\partial \widetilde{T}}{\partial x_k} \tag{3.26}$$

and

$$\tau_{kl} = \overline{\rho}\left(\widetilde{u_k u_l} - \widetilde{u}_k\widetilde{u}_l\right) \tag{3.27}$$

$$= \overline{\rho}\left(\widetilde{\widetilde{u}_k\widetilde{u}_l} - \widetilde{u}_k\widetilde{u}_l + \widetilde{u'_k\widetilde{u}_l} + \widetilde{\widetilde{u}_k u'_l} + \widetilde{u'_k u'_l}\right) \tag{3.28}$$

$$q_k = \overline{\rho}\left(\widetilde{u_k T_l} - \widetilde{u}_k\widetilde{T}_l\right) \tag{3.29}$$

$$= \overline{\rho}\left(\widetilde{\widetilde{u}_k\widetilde{T}} - \widetilde{u}_k\widetilde{T} + \widetilde{u'_k\widetilde{T}} + \widetilde{\widetilde{u}_k T'} + \widetilde{u'_k T'}\right) \tag{3.30}$$

τ_{kl} in Equation (3.27) is called the *subgrid-scale stress tensor*, and q_k in Equation (3.29) is called the *subgrid-scale heat flux*. Both terms contain nonlinear terms that represent the contribution by eddies smaller than the filter size.

Finally, the filtered ideal gas equation of state(3.20) is:

$$\boxed{\overline{p} = \overline{\rho} R \widetilde{T}} \tag{3.31}$$

3.1.3 Non-dimensional Approach

The basic equations can be made dimensionless by introducing a reference length L, velocity u_∞, density ρ_∞, temperature T_∞ and viscosity μ_∞ [45]:

$$x_k^* = \frac{x}{L} \qquad u_k^* = \frac{u_k}{u_\infty} \qquad t^* = \frac{t}{L/u_\infty} \qquad \mu^* = \frac{\mu}{\mu_\infty}$$
$$\rho^* = \frac{\rho}{\rho_\infty} \qquad p^* = \frac{p}{\rho_\infty u_\infty^2} \qquad T^* = \frac{T}{T_\infty} \qquad E_t^* = \frac{E_t}{\rho_\infty u_\infty^2}$$

where * represents the non-dimensionalized variables. The non-dimensional form of the filtered Navier-Stokes eqautions are exactly the same as Equation (3.21)–(3.23) but where each variable is replaced with a non-dimensional one. The non-dimensional viscous stress tensor σ_{kl}^* is:

$$\sigma_{kl}^* = \frac{1}{Re}\left(-\frac{2}{3}\mu^* \frac{\partial u_j^*}{\partial x_j^*}\delta_{kl} + \mu^*\left(\frac{\partial u_k^*}{\partial x_l^*} + \frac{\partial u_l^*}{\partial x_k^*}\right)\right) \tag{3.32}$$

where the Reynolds number is defined by $Re = \rho_\infty u_\infty L/\mu_\infty$. The non-dimensional viscous heat flux vector $\overset{*}{}_k$ (Equation (3.26)) is:

$$\overset{*}{}_k = -\frac{\mu^*}{(\gamma-1)RePrM_\infty^2}\frac{\partial T^*}{\partial x_k^*} \tag{3.33}$$

where $M_\infty = u_\infty/\sqrt{\gamma R_g T_\infty}$ is the Mach number and R_g is the universal gas constant. The non-dimensionalized ideal gas law (3.20) is expressed as follows:

$$\widetilde{T}^* = \gamma M_\infty^2 \frac{\overline{p}^*}{\overline{\rho}^*} \tag{3.34}$$

The non-dimensionalized SGS stress τ_{kl}^* is the same as Equation (3.27) but the non-dimensionalized SGS heat flux q_k^* (Equation (3.29)) is:

$$q_k^* = \frac{\overline{\rho}^*}{(\gamma-1)\gamma M_\infty^2}\overline{\rho}^*\left(\widetilde{u_k^* T_l^*} - \widetilde{u}_k^* \widetilde{T}_l^*\right) \tag{3.35}$$

3.1.4 Vector Form

The filtered and non-dimensionalized Navier-Stokes Equation (3.21)–(3.23) can be expressed in vector form as:

$$\frac{\partial \vec{\mathbf{Q}}}{\partial t^*} + \nabla \cdot \vec{\mathbf{E}} = 0 \tag{3.36a}$$

or

$$\frac{\partial \vec{\mathbf{Q}}}{\partial t^*} + \frac{\partial \left(\vec{\mathbf{F}}_i - \vec{\mathbf{F}}_v\right)}{\partial x^*} + \frac{\partial \left(\vec{\mathbf{G}}_i - \vec{\mathbf{G}}_v\right)}{\partial y^*} + \frac{\partial \left(\vec{\mathbf{H}}_i - \vec{\mathbf{H}}_v\right)}{\partial z^*} = 0 \tag{3.36b}$$

with

$$\vec{\mathbf{Q}} = \begin{bmatrix} \overline{\rho}^* \\ \overline{\rho}^* \widetilde{u}^* \\ \overline{\rho}^* \widetilde{v}^* \\ \overline{\rho}^* \widetilde{w}^* \\ \widetilde{E}_t^* \end{bmatrix} \tag{3.36c}$$

where $\widetilde{u}^* = \widetilde{u}_1^*$, $\widetilde{v}^* = \widetilde{u}_2^*$ and $\widetilde{w}^* = \widetilde{u}_3^*$ (i.e. link in with previous notation)

$$\vec{\mathbf{F}}_i = \begin{bmatrix} \overline{\rho}^* \widetilde{u}^* \\ \overline{\rho}^* \widetilde{u}^{*2} + \overline{p}^* \\ \overline{\rho}^* \widetilde{u}^* \widetilde{v}^* \\ \overline{\rho}^* \widetilde{u}^* \widetilde{w}^* \\ \left(\widetilde{E}_t^* + \overline{p}^*\right) \widetilde{u}^* \end{bmatrix} \quad \vec{\mathbf{G}}_i = \begin{bmatrix} \overline{\rho}^* \widetilde{v}^* \\ \overline{\rho}^* \widetilde{v}^* \widetilde{u}^* \\ \overline{\rho}^* \widetilde{v}^{*2} + \overline{p}^* \\ \overline{\rho}^* \widetilde{v}^* \widetilde{w}^* \\ \left(\widetilde{E}_t^* + \overline{p}^*\right) \widetilde{v}^* \end{bmatrix} \quad \vec{\mathbf{H}}_i = \begin{bmatrix} \overline{\rho}^* \widetilde{w}^* \\ \overline{\rho}^* \widetilde{w}^* \widetilde{u}^* \\ \overline{\rho}^* \widetilde{w}^* \widetilde{v}^* \\ \overline{\rho}^* \widetilde{w}^{*2} + \overline{p}^* \\ \left(\widetilde{E}_t^* + \overline{p}^*\right) \widetilde{w}^* \end{bmatrix} \tag{3.36d}$$

$$\vec{\mathbf{F}}_v = \begin{bmatrix} 0 \\ \widetilde{\sigma}_{xx}^* - \tau_{xx}^* \\ \widetilde{\sigma}_{xy}^* - \tau_{xy}^* \\ \widetilde{\sigma}_{xz}^* - \tau_{xz}^* \\ \widetilde{u\sigma}_{xx}^* + \widetilde{v\sigma}_{xy}^* + \widetilde{w\sigma}_{xz}^* - \quad {}_x^* - u\tau_{xx}^* - v\tau_{xy}^* - w\tau_{xz}^* - q_x^* \end{bmatrix} \tag{3.36e}$$

$$\vec{\mathbf{G}}_v = \begin{bmatrix} 0 \\ \widetilde{\sigma}_{yx}^* - \tau_{yx}^* \\ \widetilde{\sigma}_{yy}^* - \tau_{yy}^* \\ \widetilde{\sigma}_{yz}^* - \tau_{yz}^* \\ \widetilde{u\sigma}_{yx}^* + \widetilde{v\sigma}_{yy}^* + \widetilde{w\sigma}_{yz}^* - \quad {}_y^* - u\tau_{yx}^* - v\tau_{yy}^* - w\tau_{yz}^* - q_y^* \end{bmatrix} \tag{3.36f}$$

$$\vec{\mathbf{H}}_v = \begin{bmatrix} 0 \\ \widetilde{\sigma}_{zx}^* - \tau_{zx}^* \\ \widetilde{\sigma}_{zy}^* - \tau_{zy}^* \\ \widetilde{\sigma}_{zz}^* - \tau_{zz}^* \\ \widetilde{u\sigma}_{zx}^* + \widetilde{v\sigma}_{zy}^* + \widetilde{w\sigma}_{zz}^* - \quad {}_z^* - u\tau_{zx}^* - v\tau_{zy}^* - w\tau_{zz}^* - q_z^* \end{bmatrix} \tag{3.36g}$$

in which

$$\widetilde{E}_t^* = \frac{\overline{p}^*}{\gamma-1} + \overline{\rho}^* \frac{\left(\widetilde{u}^{*2}+\widetilde{v}^{*2}+\widetilde{w}^{*2}\right)}{2} \tag{3.36h}$$

$$\widetilde{\sigma}_{xx}^* = \frac{2\overline{\mu}^*}{3Re}\left(2\frac{\partial \widetilde{u}^*}{\partial x^*} - \frac{\partial \widetilde{v}^*}{\partial y^*} - \frac{\partial \widetilde{w}^*}{\partial z^*}\right) \tag{3.36i}$$

$$\widetilde{\sigma}_{yy}^* = \frac{2\overline{\mu}^*}{3Re}\left(2\frac{\partial \widetilde{v}^*}{\partial y^*} - \frac{\partial \widetilde{u}^*}{\partial x^*} - \frac{\partial \widetilde{w}^*}{\partial z^*}\right) \tag{3.36j}$$

$$\widetilde{\sigma}_{zz}^* = \frac{2\overline{\mu}^*}{3Re}\left(2\frac{\partial \widetilde{w}^*}{\partial z^*} - \frac{\partial \widetilde{u}^*}{\partial x^*} - \frac{\partial \widetilde{v}^*}{\partial y^*}\right) \tag{3.36k}$$

$$\widetilde{\sigma}_{xy}^* = \widetilde{\sigma}_{yx}^* = \frac{\overline{\mu}^*}{Re}\left(\frac{\partial \widetilde{u}^*}{\partial y^*} + \frac{\partial \widetilde{v}^*}{\partial x^*}\right) \tag{3.36l}$$

$$\widetilde{\sigma}_{xz}^* = \widetilde{\sigma}_{zx}^* = \frac{\overline{\mu}^*}{Re}\left(\frac{\partial \widetilde{u}^*}{\partial z^*} + \frac{\partial \widetilde{w}^*}{\partial x^*}\right) \tag{3.36m}$$

$$\widetilde{\sigma}_{yz}^* = \widetilde{\sigma}_{zy}^* = \frac{\overline{\mu}^*}{Re}\left(\frac{\partial \widetilde{v}^*}{\partial z^*} + \frac{\partial \widetilde{w}^*}{\partial y^*}\right) \tag{3.36n}$$

$$\widetilde{}_x^* = \frac{\overline{\mu}^*}{(\gamma-1)M_\infty^2 RePr}\frac{\partial \widetilde{T}^*}{\partial x^*} \tag{3.36o}$$

$$\widetilde{}_x^* = \frac{\overline{\mu}^*}{(\gamma-1)M_\infty^2 RePr}\frac{\partial \widetilde{T}^*}{\partial y^*} \tag{3.36p}$$

$$\widetilde{}_x^* = \frac{\overline{\mu}^*}{(\gamma-1)M_\infty^2 RePr}\frac{\partial \widetilde{T}^*}{\partial z^*} \tag{3.36q}$$

τ_{kl} is the Subgrid-Scale Stress and q_k is the Subgrid-Scale Heat Flux, defined by Equation 3.27–3.30.

3.2 Differences between DNS, LES, MILES and RANS

3.2.1 Averaging and Filtering

DNS, LES, MILES and RANS, all solve the same Navier-Stokes Equations ((3.13)-(3.15)).

When the numerical method and the grid resolution are good enough to capture all the turbulent scales the simulation is DNS. The values on each grid points are not averaged or modelled. Therefore in DNS, Equations (3.13)-(3.15) should be written without filter signs, and the $\frac{\partial \tau_{kl}}{\partial x_l}$ and $\frac{\partial q_k}{\partial x_k}$ terms.

In LES, MILES and RANS the numerical scheme may not be accurate enough and/or the grid resolution may not be fine enough to capture all turbulent eddies. This will introduce truncation or discretisation errors which degrade (in another word, smear) the solution. From a mathematical point of view this is an averaging process i.e., the values at each grid point are averaged values in space and/or time.

In RANS, the averaging is viewed as the time or Reynolds averaging (see Section 2.2.6) depending on whether the calculation is steady or unsteady. Therefore the time step can be much larger than the turnover time of the smallest resolved eddies.

In LES and MILES the values at each grid point are spatial averaged values. Therefore calculations are always unsteady and marching in time where the time steps should be smaller than the turnover time of the smallest resolved eddy. To mathematically separate the big eddies which the calculation can resolve, and the small eddies which have to be modelled, a filtering concept was introduced (Section 3.1.1).

3.2.2 Modelling

All of the four methods still solve the same basic equation even though the concept of the averaging is different. Differences arise when modelling the missing accuracy. Turbulent eddies can be roughly divided into 2 categories; energy producing big eddies, energy dissipating small eddies (see Section 2.2.1–2.2.3).

RANS's accuracy is only enough to capture the biggest eddies and some of the energy producing eddies are modelled as well. The modelling is so called *turbulence modelling* (see Section 2.2.6). Much research effort and time has been spent on turbulence modelling and many turbulence models have been proposed. But turbulence modelling has been very difficult and some people express a pessimistic view on further developments. This is because modelling something which is not there (energy production) is always going to be difficult.

LES and MILES should have high enough accuracy to capture all of the energy

producing big eddies, and time accuracy. Therefore they only need to model the energy dissipating small eddies, in other words, eddies smaller than the filtering size. This is called subgrid-scale (SGS) modelling. SGS modelling is easier than turbulence modelling because it is only modelling energy dissipation by small eddies (i.e. getting rid of something which is already there).

Very Large Eddy Simulation (VLES) is somewhere between RANS and LES methods and uses combinations of turbulence modelling and SGS modelling.

3.2.3 Differences between LES and MILES

The basic difference between Conventional LES and MILES is in the concept of filtering (see Section 3.1.1 on filtering). In conventional LES, cell values are obtained from successive filtering and cell averaging:

$$\overline{u}_i(\mathbf{x}) = \iiint G(\mathbf{x}-\mathbf{x}')\, u_i(\mathbf{x}')\, d\mathbf{x}', \tag{3.2}$$

whereas in MILES they are obtained from cell averaging only:

$$\overline{u}_i(\mathbf{x}) = \iiint u_i(\mathbf{x}')\, d\mathbf{x}' \tag{3.37}$$

However, in practical LES implementations the prefiltering has no explicit effect on the variables that are solved for because this filtering operation is performed when deriving the LES equations, which are then to be discretized. The prefiltering has the indirect effect of producing unresolved transport or SGS terms that require additional closure modelling, thus affecting the equations to be solved and their solution. One can view the cell averaging as a built-in filtering procedure using essentially a top-hat filter (see Section 3.1.1).

Therefore the same equations for LES in previous section can be used in MILES except there is no separate Subgrid-Scale Stress τ_{kl} and Subgrid-Scale Heat Flux q_k.

Chapter 4

Numerical Methods

4.1 The Finite Volume Method

The finite volume method used in the present thesis is based on the work of Qin *et al.* [98–100]. The basic outline of the method is described here. A more detailed description of the general finite volume concept can also be found in Hirsch [101], Ferziger and Perić [86], and Bui [60].

As stated in Section 3.1.4 the governing equations in vector form can be written as:

$$\frac{\partial \vec{\mathbf{Q}}}{\partial t} + \nabla \cdot \vec{\mathbf{E}} = 0 \tag{3.36a}$$

where $\vec{\mathbf{Q}}$ is the state vector and $\vec{\mathbf{E}}$ is the flux vector of the Navier-Stokes equations. $\vec{\mathbf{E}}$ consists of an inviscid flux vector $\vec{\mathbf{E}}_i$ and a viscous flux vector $\vec{\mathbf{E}}_v$:

$$\vec{\mathbf{E}} = \vec{\mathbf{E}}_i + \vec{\mathbf{E}}_v \tag{4.1}$$

In the Euler equations, the inviscid form of the governing eqautions, the viscous term $\vec{\mathbf{E}}_v$ is ignored and only the inviscid flux vector $\vec{\mathbf{E}}_i$ is included.

The above equation can be expressed in the integral form using Gauss' divergence theorem. The resulting equation is:

$$\frac{d}{dt}\overline{Q} = -\frac{1}{V}\int_S \vec{\mathbf{E}} d\vec{s} \tag{4.2}$$

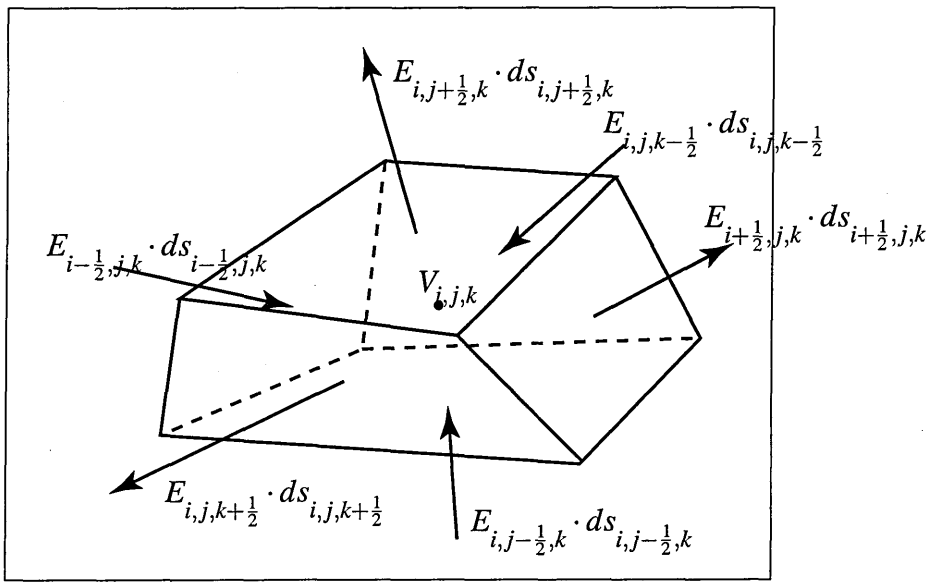

Figure 4.1: A hexahedral cell with a control volume of $V_{i,j,k}$ with the cell values at the cell centre.

where V is the volume of a cell and

$$\overline{Q} = \frac{1}{V}\int_V \vec{\mathbf{Q}} \cdot dv \tag{4.3}$$

and

$$\vec{\mathbf{E}} \cdot d\vec{s} = \vec{\mathbf{E}} \cdot \vec{\mathbf{n}} ds \tag{4.4}$$

The above equation indicates that change of quantity $\vec{\mathbf{Q}}$ in a control volume V is the same as the sum of the flux component normal to the surface of the control volume times the surface area $d\vec{s}$. In other words, the change in a quantity inside the volume $(\vec{\mathbf{E}} \cdot \vec{\mathbf{n}})$ is the same as the sum of what is going out and coming in through the surfaces.

Figure 4.1 shows an example of a hexahedral cell with a control volume of $V_{i,j,k}$ with the cell values at the cell centre. The conservative fluxes through a cell surface are calculated using values from the cells on both sides of the cell interface:

$$\begin{aligned}\frac{\partial}{\partial t}\left(V_{i,j,k}\vec{\mathbf{Q}}\right) =& \vec{\mathbf{E}}_{i-\frac{1}{2},j,k} \cdot d\vec{s}_{i-\frac{1}{2},j,k} - \vec{\mathbf{E}}_{i+\frac{1}{2},j,k} \cdot d\vec{s}_{i+\frac{1}{2},j,k} \\ & \vec{\mathbf{E}}_{i,j-\frac{1}{2},k} \cdot d\vec{s}_{i,j-\frac{1}{2},k} - \vec{\mathbf{E}}_{i,j+\frac{1}{2},k} \cdot d\vec{s}_{i,j+\frac{1}{2},k} \\ & \vec{\mathbf{E}}_{i,j,k-\frac{1}{2}} \cdot d\vec{s}_{i,j,k-\frac{1}{2}} - \vec{\mathbf{E}}_{i,j,k+\frac{1}{2}} \cdot d\vec{s}_{i,j,k+\frac{1}{2}}\end{aligned} \tag{4.5}$$

The surface vector can be calculated from the position vector $\vec{r}$ of the grid points which define the corner of the cell. For example $d\vec{s}_{i+\frac{1}{2},j,k}$ can be calculated as following:

$$d\vec{s}_{i+\frac{1}{2},j,k} = -\frac{1}{2}\left(\vec{r}_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}} - \vec{r}_{i+\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}}\right) \times \left(\vec{r}_{i+\frac{1}{2},j+\frac{1}{2},k-\frac{1}{2}} - \vec{r}_{i+\frac{1}{2},j-\frac{1}{2},k+\frac{1}{2}}\right) \tag{4.6}$$

Then, the cell volume $V_{i,j,k}$ is evaluated as following:

$$V_{i,j,k} = \frac{1}{3}\left(d\vec{s}_{i+\frac{1}{2},j,k} + d\vec{s}_{i,j+\frac{1}{2},k} + d\vec{s}_{i,j,k+\frac{1}{2}}\right) \cdot \left(\vec{r}_{i+\frac{1}{2},j+\frac{1}{2},k+\frac{1}{2}} - \vec{r}_{i-\frac{1}{2},j-\frac{1}{2},k-\frac{1}{2}}\right) \tag{4.7}$$

4.2 Inviscid Flux Calculation

Three methods were used to calculate the inviscid flux $\vec{\mathbf{E}}_i$ on the boundaries; Roe's scheme, Osher's Scheme and the central difference scheme. Among these methods, the central difference scheme is least numerically dissipative. Most traditional LES use this method for this reason. (see Section 2.3.1 for more details.)

The main reason for choosing Roe's and Osher's scheme is its popularity. Even though newer and more complicated schemes have been developed, these schemes are still the most popular schemes today. Our research group also has been using these schemes in a number of projects with tremendous experience.

4.2.1 Roe's Scheme

Perhaps, Roe's scheme [102] is the most well-known approximate Riemann solver used today. The scheme is described below but much more theoretical background can be found in Reference [72].

The Exact Riemann Problem and Godunov Flux

In the exact Rieman Problem we are concerned with solving the general Initial Boundary Value Problem (IBVP):

$$\begin{array}{lll} \text{PDEs} & : & Q_t + E(Q)_x = 0, \\ \text{ICs} & : & Q(x,0) = Q^{(0)}(x), \\ \text{BCs} & : & Q(0,t) = Q_l(t),\ Q(L,t) = Q_r(t) \end{array} \tag{4.8}$$

in a domain $x_l \leq x \leq x_r$, utilising the explicit conservative formula

$$Q_i^{n+1} = Q_i^n + \frac{\Delta t}{\Delta x}\left[E_{i-\frac{1}{2}} - E_{i+\frac{1}{2}}\right] \tag{4.9}$$

along with the Godunov intercell numerical flux

$$E_{i+\frac{1}{2}} = E\left(Q_{i+\frac{1}{2}}(0)\right) \tag{4.10}$$

where $Q_{i+\frac{1}{2}}(0)$ is the exact similarity solution $Q_{i+\frac{1}{2}}(x/t)$ of the Riemann problem

$$Q_t + E(Q)_x = 0 \tag{4.11}$$

$$Q(x,0) = \begin{cases} Q_L, & x < 0 \\ Q_R, & x > 0 \end{cases} \tag{4.12}$$

evaluated at $x/t = 0$.

Numerical Formulae

Instead of calculating the exact Riemann flux, which is expensive, Roe proposed that $E_{i+\frac{1}{2}}$ can be approximated by:

$$E_{i+\frac{1}{2}} = \frac{1}{2}(E_L + E_R) - \frac{1}{2}\sum_{i=1}^{5} \tilde{\alpha}_i |\tilde{\lambda}_i| \tilde{K}^{(i)} \tag{4.13}$$

where E_L and E_R are the left and right flux, $\tilde{\alpha}_i$ are the wave strengths, $\tilde{\lambda}_i$ are the eigenvalues and $\tilde{K}^{(i)}$ are the right eigenvectors. In fact it is a central difference scheme with an extra term $-\frac{1}{2}\sum_{i=1}^{5} \tilde{\alpha}_i |\tilde{\lambda}_i| \tilde{K}^{(i)}$. Definitions of left and right flux, and how to calculate them are presented in Section 4.2.3 while $\tilde{\alpha}_i$, $\tilde{\lambda}_i$ and $\tilde{K}^{(i)}$ are described below.

First, by Roe averaged values $\tilde{u}$, $\tilde{v}$, $\tilde{w}$, $\tilde{H}$ and $\tilde{a}$ are calculated as follows.

$$\tilde{u} = \frac{\sqrt{\rho_L}u_L + \sqrt{\rho_R}u_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{4.14a}$$

$$\tilde{v} = \frac{\sqrt{\rho_L}v_L + \sqrt{\rho_R}v_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{4.14b}$$

$$\tilde{w} = \frac{\sqrt{\rho_L}w_L + \sqrt{\rho_R}w_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{4.14c}$$

$$\tilde{H} = \frac{\sqrt{\rho_L}H_L + \sqrt{\rho_R}H_R}{\sqrt{\rho_L} + \sqrt{\rho_R}} \tag{4.14d}$$

$$\tilde{a} = \sqrt{(\gamma - 1)\left[\tilde{H} - \frac{1}{2}\tilde{V}^2\right]} \tag{4.14e}$$

where

$$\tilde{V}^2 = \tilde{u}^2 + \tilde{u}^2 + \tilde{w}^2 \tag{4.15}$$

and H is the total enthalphy:

$$\tilde{H} = \frac{\tilde{E}_t + \tilde{p}}{\tilde{\rho}} \tag{4.16}$$

where $\tilde{E}_t$ is the total energy per unit volume, which for ideal gases is calculated using Equation (3.16).

The averaged eigenvalues $\tilde{\lambda}_i$ are then:

$$\tilde{\lambda}_1 = \tilde{u} - \tilde{a}, \quad \tilde{\lambda}_2 = \tilde{\lambda}_3 = \tilde{\lambda}_4 = \tilde{u}, \quad \tilde{\lambda}_5 = \tilde{u} + \tilde{a}, \tag{4.17}$$

the averaged right eigenvectors $\tilde{K}^{(i)}$ are:

$$\tilde{K}^{(1)} = \begin{bmatrix} 1 \\ \tilde{u} - \tilde{a} \\ \tilde{v} \\ \tilde{w} \\ \tilde{H} - \tilde{u}\tilde{a} \end{bmatrix} \quad \tilde{K}^{(2)} = \begin{bmatrix} 1 \\ \tilde{u} \\ \tilde{v} \\ \tilde{w} \\ \frac{1}{2}\tilde{V}^2 \end{bmatrix} \quad \tilde{K}^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ \tilde{v} \end{bmatrix}$$

$$\tilde{K}^{(4)} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ \tilde{w}* \end{bmatrix} \quad \tilde{K}^{(5)} = \begin{bmatrix} 1 \\ \tilde{u} + \tilde{a} \\ \tilde{v} \\ \tilde{w} \\ \tilde{H} + \tilde{u}\tilde{a} \end{bmatrix} \tag{4.18}$$

and the wave strengths $\tilde{\alpha}_i$ are calculated as following:

$$\tilde{\alpha}_1 = \frac{1}{2\tilde{a}}\left[\Delta Q_1\left(\tilde{u}+\tilde{a}\right) - \Delta Q_2 - \tilde{a}\tilde{\alpha}_2\right] \tag{4.19a}$$

$$\tilde{\alpha}_2 = \frac{\gamma-1}{\tilde{a}^2}\left[\Delta Q_1\left(\tilde{H}-\tilde{u}^2\right) - \tilde{u}\Delta Q_2 - \overline{\Delta Q_5}\right] \tag{4.19b}$$

$$\tilde{\alpha}_3 = \Delta Q_3 - \tilde{v}\Delta Q_1 \tag{4.19c}$$

$$\tilde{\alpha}_4 = \Delta Q_4 - \tilde{w}\Delta Q_1 \tag{4.19d}$$

$$\tilde{\alpha}_5 = \Delta Q_1 - \left(\tilde{\alpha}_1 + \tilde{\alpha}_2\right) \tag{4.19e}$$

Here Q_i denote the elements of the state vector $\vec{\mathbf{Q}}$,

$$\Delta Q_i = \left(Q_i\right)_R - \left(Q_i\right)_L, \tag{4.20}$$

and

$$\overline{\Delta Q_5} = \Delta Q_5 - \left(\Delta Q_3 - \tilde{v}\Delta Q_1\right)\tilde{v} - \left(\Delta Q_4 - \tilde{w}\Delta Q_1\right)\tilde{w} \tag{4.21}$$

Now we have everything to close Equation (4.13).

4.2.2 Osher's Scheme

The derivation of the Osher intercell numerical flux depends on integration in phase space. Such operation involves the choice of *integration paths*, *intersection points* and *sonic points*. The integration paths are taken to be *integral curves* associated with the set of right eigenvectors and to date there are essentially two ways of *ordering* these integration paths. The most recent approach, which also has been used in our code, orders the integration paths such that these correspond to physically meaningful relations across wave families in *physical space*. This is called *physical ordering* or *P-ordering*. Osher's original scheme utilises the ordering of paths that is percisely the inverse of the P-ordering; this is usually called *O-ordering*. Intersection and sonic points are computed via *Generalised Riemann Invariants*.

Like Roe's Scheme, Osher's approach to upwind differencing provides an approximation to the Godunov flux and results from evaluating the physical flux

$E(Q)$ at various states Q_k; these include the data states Q_L, Q_R, intersection points $Q_{1/3}$, $Q_{2/3}$ and sonic points $\lambda_1(Q)$, $\lambda_3(Q)$.

Osher's Scheme has successfully been applied to a wide range of aerodynamic problems by Qin *et al.* [98,99].

For a fuller description of the theoretical background to Oshers scheme References [72, 103] are recommended. Only enough information to implement the Osher's Scheme with P-ordering is presented below.

Interaction Points

Let u_* and p_* be the common particle velocity and pressure for $Q_{1/3}$ and $Q_{2/3}$, that is:

$$u_{1/3} = u_{2/3} = u_* = constant, \quad p_{1/3} = p_{2/3} = p_* = constant \tag{4.22}$$

and they are calculated as:

$$u_* = \frac{Hu_L/a_L + u_R/a_R + 2(H-1)/(\gamma-1)}{H/a_L + 1/a_R} \tag{4.23}$$

$$p_* = \left[\frac{a_L + a_R - R(u_R - u_L)(\gamma-1)/2}{a_L/p_L^{\gamma^*} + a_/p_R^{\gamma^*}}\right] \tag{4.24}$$

where

$$\gamma^* = \frac{\gamma-1}{2\gamma} \tag{4.25}$$

and

$$H = (p_L/p_R)^{\gamma^*} \tag{4.26}$$

The left and right speed of sound values are calculated as following:

$$a_{1/3} = a_L(p_*/p_L)^{\gamma^*}, \quad a_{2/3} = a_R(p_*/p_R)^{\gamma^*} \tag{4.27}$$

The density values are calculated as follows:

$$\rho_{1/3} = p_L\left(\frac{p_*}{p_L}\right)^{\frac{1}{\gamma}}, \quad \rho_{2/3} = p_R\left(\frac{p_*}{p_R}\right)^{\frac{1}{\gamma}} \tag{4.28}$$

	$u_L - a_L \geq 0$ $u_R + a_R \geq 0$	$u_L - a_L \geq 0$ $u_R + a_R \leq 0$	$u_L - a_L \leq 0$ $u_R + a_R \geq 0$	$u_L - a_L \leq 0$ $u_R + a_R \leq 0$
$u^* \geq 0$ $u^* - a_{1/3} \geq 0$	E_L (supersonic)	$E_L + E_R$ $-E_{SR}$	E_{SL}	$E_{SL} - E_{SR}$ $+E_R$
$u^* \geq 0$ $u^* - a_{1/3} \leq 0$	$E_L - E_{SL}$ $+E_{1/3}$	$E_L - E_{SL}$ $+E_{1/3} - E_{SR}$ $+E_R$	$E_{1/3}$ (subsonic)	$E_R + E_{1/3}$ $-E_{SR}$
$u^* \leq 0$ $u^* + a_{2/3} \geq 0$	$E_L - E_{SL}$ $+E_{2/3}$	$E_L - E_{SL}$ $+E_{2/3} - E_{SR}$ $+E_R$	$E_{2/3}$ (subsonic)	$E_{2/3} - E_{SR}$ $+E_R$
$u^* \leq 0$ $u^* + a_{2/3} \leq 0$	$E_L - E_{SL}$ $+E_{SR}$	$E_L - E_{SL}$ $+E_R$	E_{SR}	E_R (supersonic)

Table 4.1: Osher's flux formulae using P-ordering of integration paths.

Sonic Points

The computation of the sonic points Q_{SL} (left wave) and Q_{SR} (right wave) are as following:

$$\begin{aligned}
u_{SL} &= \tfrac{\gamma-1}{\gamma+1}u_L + \tfrac{2a_L}{\gamma+1}, & u_{SR} &= \tfrac{\gamma-1}{\gamma+1}u_R - \tfrac{2a_R}{\gamma+1} \\
a_{SL} &= u_{SL}, & a_{SR} &= -u_{SR} \\
\rho_{SL} &= \rho_L \left(\tfrac{a_{SL}}{a_L}\right)^{\frac{2}{\gamma-1}} & \rho_{SR} &= \rho_R \left(\tfrac{a_{SR}}{a_R}\right)^{\frac{2}{\gamma-1}} \\
p_{SL} &= p_L \left(\tfrac{\rho_{SL}}{\rho_L}\right)^{\gamma} & p_{SR} &= p_L \left(\tfrac{\rho_{SR}}{\rho_R}\right)^{\gamma}
\end{aligned} \tag{4.29}$$

Integration Along Partial Paths

The integration along each partial path $I_k(U)$ is performed individually and the results are added to produce $E_{i+\frac{1}{2}}$. There are only 16 cases which can be realisable and are tabulated in Table 4.1. The correct $E_{i+\frac{1}{2}}$ can be selected by checking the conditions of the intersection points on the first column and the data on the top row. For normal supersonic and subsonic points, only one flux is needed to be calculated.

4.2.3 Monotone Upstream-Centered Schemes for Conservation Laws (MUSCL)

Original MUSCL

The original MUSCL Scheme uses a quadratic representation on cells of same size which can lead to a third-order spatial discretization. The discrete state variables are representative of the average state within the cells.

If we consider the general local representation, valid within cell i, at a given instant, a discrete state variable U can represented as following:

$$U(x) = U_i + \frac{1}{\Delta x}(x - x_i)\,\delta_i U + \frac{3\kappa}{2\Delta x^2}\left[(x - x_i)^2 - \frac{\Delta x^2}{12}\right]\delta_i^2 U \tag{4.30}$$

where $x_{i-1/2} < x < x_{i+1/2}$ and U_i is the average value, defined by:

$$U_i = \frac{1}{\Delta x}\int_{i-1/2}^{i+1/2} U(x)dx \tag{4.31}$$

and $\delta_i U$, $\delta_i^2 U$ are estimations of the first and second finite difference within cell i.

For $3\kappa = 1$, Equation (4.30) is a correct Taylor expansion up to third order and this parabolic representation then generates a third-order accurate space discretization. For other values of the parameter κ the above representation is considered as linear with various truncation terms. Observe that the nodal value within the cell, $U(x_i)$, is equal to the average value U_i only for $\kappa = 0$, since:

$$U(x_i) = U_i + \frac{\kappa}{8}\delta_i^2 U \tag{4.32}$$

In order to define completely the representation, the derivatives $\delta_i U$ and $\delta_i^2 U$ have to be estimated. If we require these gradients to depend only on quantities of adjacent cells, the only choice is to use central differences of averaged values:

$$\delta_i U = \frac{U_{i+1} - U_{i-1}}{2} \tag{4.33a}$$

$$\delta_i^2 U = U_{i+1} - 2U_i + U_{i-1} \tag{4.33b}$$

To calculate the flux on the left and right hand-side of a cell using the Riemann solvers in Section 4.2.1 and 4.2.2 , only the values at the cell boundaries are

required:

$$U^L_{i+1/2} = U_i + \frac{1}{2}\delta_i U + \frac{\kappa}{4}\delta_i^2 U \tag{4.34a}$$

$$= U_i + \frac{1}{4}(1-\kappa)\left(U_i - U_{i-1}\right) + \frac{1}{4}(1+\kappa)\left(U_{i+1} - U_i\right) \tag{4.34b}$$

$$U^R_{i+1/2} = U_i - \frac{1}{2}\delta_i U + \frac{\kappa}{4}\delta_i^2 U \tag{4.35a}$$

$$= U_i - \frac{1}{4}(1+\kappa)\left(U_i - U_{i-1}\right) - \frac{1}{4}(1-\kappa)\left(U_{i+1} - U_i\right) \tag{4.35b}$$

where the superscripts L and R refer to the left and right sides at the considered boundary. The spatial order of accuracy is determined by the value κ,

$$\begin{array}{ll} \kappa = -1, & \text{fully upwind scheme} \\ \kappa = 0, & \text{Fromm scheme} \\ \kappa = 1/3, & \text{third-order upwind-biased scheme} \\ \kappa = 1, & \text{three-point central-difference scheme} \end{array} \tag{4.36}$$

Modified MUSCL Scheme

The original formulation of the MUSCL scheme assumes that the grid is uniform, therefore if the grid is stretched additional numerical error is introduced. In the case of highly stretched grids numerical accuracy can be reduced to 1st order [104]. To take into account a non-uniform grid, Prince *et al.* [105] proposed a modified MUSCL Scheme.

In this scheme, $U^L_{i+\frac{1}{2}}$ and $U^R_{i+\frac{1}{2}}$ in Equation (4.34b) and (4.35b) can be replaced by the following equations respectively:

$$U^L_{i+\frac{1}{2}} = U_i + \frac{1}{\left(\frac{\Delta x_i + \Delta x_{i-1}}{\Delta x_i}\right) + \left(\frac{\Delta x_i + \Delta x_{i+1}}{\Delta x_i}\right)} \left(\left(\left(\frac{\Delta x_i + \Delta x_{i-1}}{2\Delta x_i}\right) + \kappa\right)\tilde{\Delta}^+ + \left(\left(\frac{\Delta x_i + \Delta x_{i+1}}{2\Delta x_i}\right) - \kappa\right)\tilde{\Delta}^-\right) \tag{4.37}$$

$$U^R_{i+\frac{1}{2}} = U_{i+1} - \frac{1}{\left(\frac{\Delta x_{i+1} + \Delta x_i}{\Delta x_{i+1}}\right) + \left(\frac{\Delta x_{i+1} + \Delta x_{i+2}}{\Delta x_{i+1}}\right)} \left(\left(\left(\frac{\Delta x_{i+1} + \Delta x_i}{2\Delta x_{i+1}}\right) - \kappa\right)\tilde{\tilde{\Delta}}^+ + \left(\left(\frac{\Delta x_{i+1} + \Delta x_{i+2}}{2\Delta x_{i+1}}\right) + \kappa\right)\tilde{\tilde{\Delta}}^-\right) \tag{4.38}$$

Where

$$\begin{aligned}
\tilde{\Delta}^{+} &= \frac{2\Delta x_i\left(U_{i+1}-U_i\right)}{\left(\Delta x_{i+1}+\Delta x_i\right)} \\
\tilde{\Delta}^{-} &= \frac{2\Delta x_i\left(U_i-U_{i-1}\right)}{\left(\Delta x_i+\Delta x_{i-1}\right)} \\
\tilde{\tilde{\Delta}}^{+} &= \frac{2\Delta x_{i+1}\left(U_{i+2}-U_{i+1}\right)}{\left(\Delta x_{i+2}+\Delta x_{i+1}\right)} \\
\tilde{\tilde{\Delta}}^{-} &= \frac{2\Delta x_{i+1}\left(U_{i+1}-U_i\right)}{\left(\Delta x_{i+1}+\Delta x_i\right)}
\end{aligned} \tag{4.39}$$

and

$$\begin{aligned}
\Delta x_{i-1} &= x_{i-\frac{1}{2}} - x_{i-\frac{3}{2}} = 2\left(x_{i-\frac{1}{2}} - x_{i-1}\right) \\
\Delta x_i &= x_{i+\frac{1}{2}} - x_{i-\frac{1}{2}} = 2\left(x_i - x_{i-\frac{1}{2}}\right) \\
\Delta x_{i+1} &= x_{i+\frac{3}{2}} - x_{i+\frac{1}{2}} = 2\left(x_{i+1} - x_{i+\frac{1}{2}}\right)
\end{aligned} \tag{4.40}$$

4.2.4 Limiters

A limiter may be used to reduce the scheme to a fully one-sided scheme of first- or second-order accuracy in the shock region, thus eliminating over and undershoots associated with high order MUSCL schemes. Essentially, a limiter adds artificial viscosity to the shock region to compensate for the unphysical oscillation.

There are two kinds of limiters: flux limiters, and slope limiters [86]. As the name suggests, flux limiters limit the change in a flux and slope limiters limit the change in a variable. Since variables are used to calculate the flux, not surprisingly some flux limiters have equivalent slope limiters. There are many examples of limiters [71, 74, 86], the latest developments in limiters have been problem dependent and have created much debate.

Two slope limiters [73] are chosen to be tested for suitability with MILES: the minmod and smooth limiters. Both of these limiters have been successfully used in many steady and unsteady RANS calculations. The current implementation of limiters is described below in the context of the original MUSCL scheme. They may also be used with the modified MUSCL scheme without modifications.

Minmod Limiter

The Minmod limiter [73] is one of the most popular limiters used with the MUSCL scheme. Minimum-modulus (Minmod) type limiters are activated in regions of rapidly changing gradients. The following describes the Minmod slope limiter, an equivalent Minmod flux limiter will work on a similar manner.

Equations (4.34b) and (4.35b) are modified as following:

$$U^L_{i+1/2} = U_i + \frac{1}{4}(1-\kappa)\widetilde{\Delta_{i+3/2}} + \frac{1}{4}(1+\kappa)\widetilde{\widetilde{\Delta_{i+1/2}}} + \tag{4.41}$$

$$U^R_{i+1/2} = U_i - \frac{1}{4}(1+\kappa)\widetilde{\widetilde{\Delta_{i-1/2}}} - \frac{1}{4}(1-\kappa)\widetilde{\Delta_{i+1/2}} \tag{4.42}$$

where

$$\widetilde{\Delta_{i+3/2}} = \text{minmod}\left(\Delta_{i+3/2}, w\Delta_{i+1/2}\right) \tag{4.43a}$$

$$\widetilde{\widetilde{\Delta_{i+1/2}}} = \text{minmod}\left(\Delta_{i+1/2}, w\Delta_{i+3/2}\right) \tag{4.43b}$$

and

$$\Delta_{i+1/2} = U_{i+1} - U_i \tag{4.44}$$

$$\text{minmod}(p, \omega q) = \text{sgn}(p) \cdot \max\{0, \min[|p|, \omega q \ \text{sgn}(p)]\} \tag{4.45}$$

$$1 \leq \omega \leq (3-\kappa)/(1-\kappa) \quad \text{with} \quad \kappa \neq 1 \tag{4.46}$$

κ is as defined in Equation (4.36).

Smooth Limiter

A smooth limiter [73] acts in a continuously differentiable manner.

The smooth limiter is implemented by rewriting Equation (4.34b) and (4.35b) as:

$$U^L_{i+1/2} = U_i + \frac{s}{4}(1-\kappa s)\Delta_{i-1/2} + \frac{1}{4}(1+\kappa s)\Delta_{i+1/2} \tag{4.47}$$

$$U^R_{i+1/2} = U_i - \frac{s}{4}(1+\kappa s)\Delta_{i-1/2} - \frac{1}{4}(1-\kappa s)\Delta_{i+1/2} \tag{4.48}$$

where

$$\Delta_{i-1/2} = U_i - U_{i-1} \tag{4.49a}$$

$$\Delta_{i+1/2} = U_{i+1} - U_i \tag{4.49b}$$

and

$$s = \frac{2\Delta_{i+1/2}\Delta_{i-1/2} + \varepsilon}{\left(\Delta_{i+1/2}\right)^2 + \left(\Delta_{i-1/2}\right)^2 + \varepsilon} \tag{4.50}$$

Here ε is a small number preventing division by zero in regions of null gradients. In practice, $10^{-7} \leq \varepsilon \leq 10^{-5}$ is commonly used.

4.2.5 Central Difference Scheme

In fact, the MUSCL Scheme (see above section) can be used for Central Difference calculations. By observing Equations (4.34b & 4.35b) one can realize that by setting $\kappa = 1$ the interface values are the arithmetic mean of the adjacent cell values and the upwind character is totally lost. This corresponds to a 2nd-order central scheme since there is no discontinuity at the cell interfaces.

4.3 Viscous Flux Calculation

In our calculation a 2nd order central difference scheme was used to solve the viscous flux $\vec{E}_{v_i}$ The order of the scheme for the viscous flux is less important than inviscid flux as observed by some researchers [55].

For the evaluation of the diffusive fluxes it is necessary to approximate the gradients of values at each boundary of all the cells. The Gauss' Divergence theorem is also used here to calculate the gradients.

$$\nabla u = \frac{1}{V}\int_s u d\vec{s} \tag{4.51}$$

However, the domain for this integral centred at each side of a cell is no longer the cell boundary itself. Therefore an offset boundary around the side centre is used and both the area vectors and the variables on this boundary have to be approximated by values from both sides (Figure 4.2). For instance, to evaluate a

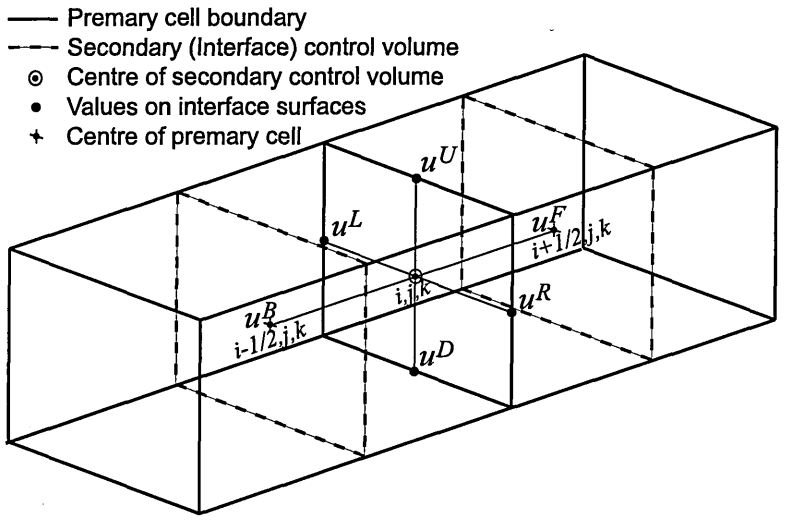

Figure 4.2: Interface flux approximation

derivative $\frac{\partial u}{\partial x}$ at the cell interface between cell i,j,k and $i+1,j,k$, the following approximation is used:

$$\left(\frac{\partial u}{\partial x}\right)_{i+\frac{1}{2},j,k} = \frac{1}{V_{i+\frac{1}{2},j,k}} \left(u^R ds^R - u^L ds^L + u^U ds^U - u^D ds^D + u^F ds^F - u^B ds^B\right) \tag{4.52}$$

where $V_{i,j+1/2,k}$ is the average volume of cells i, j, k and $i,j+1,k$, and

$$u^F = u_{i+1,j,k} \tag{4.53a}$$

$$u^B = u_{i,j,k} \tag{4.53b}$$

$$u^R = \left(u_{i,j+1,k} + u_{i+1,j+1,k} + u_{i,j,k} + u_{i+1,j,k}\right)/4 \tag{4.53c}$$

$$u^L = \left(u_{i,j-1,k} + u_{i+1,j-1,k} + u_{i,j,k} + u_{i+1,j,k}\right)/4 \tag{4.53d}$$

$$u^U = \left(u_{i,j,k+1} + u_{i+1,j,k+1} + u_{i,j,k} + u_{i+1,j,k}\right)/4 \tag{4.53e}$$

$$u^D = \left(u_{i,j,k-1} + u_{i+1,j,k-1} + u_{i,j,k} + u_{i+1,j,k}\right)/4 \tag{4.53f}$$

$$ds^F = \left(ids_{i,j,k} + ids_{i+1,j,k}\right)/2 \tag{4.54a}$$

$$ds^B = \left(ids_{i,j,k} + ids_{i-1,j,k}\right)/2 \tag{4.54b}$$

$$ds^R = \left(ids_{i,j,k} + ids_{i+1,j,k}\right)/2 \tag{4.54c}$$

$$ds^L = \left(ids_{i,j-1,k} + ids_{i+1,j-1,k}\right)/2 \tag{4.54d}$$

$$ds^U = \left(ids_{i,j,k} + ids_{i+1,j,k}\right)/2 \tag{4.54e}$$

$$ds^D = \left(ids_{i,j,k-1} + ids_{i+1,j,k}\right)/2 \tag{4.54f}$$

Similar expressions are employed for the other gradients and cell interfaces.

4.4 Time Marching

4.4.1 Jameson's Runge-Kutta Method

Jameson's 4 stage Runge-Kutta Method is used for the time integration, i.e.

$$\begin{aligned}
Q_i^{(0)} &= Q_i^n \\
Q_i^{(1)} &= Q_i^n - \alpha_1 \Delta t R_i\left(Q_i^{(0)}\right) \\
Q_i^{(2)} &= Q_i^n - \alpha_2 \Delta t R_i\left(Q_i^{(1)}\right) \\
Q_i^{(3)} &= Q_i^n - \alpha_3 \Delta t R_i\left(Q_i^{(2)}\right) \\
Q_i^{n+1} &= Q_i^n - \Delta t R_i\left(Q_i^{(3)}\right)
\end{aligned} \tag{4.55}$$

where Δt is the time step and R_i is the residual. For fourth order time accuracy the following α values are used:

$$\alpha_1 = \frac{1}{4}, \quad \alpha_2 = \frac{1}{3}, \quad \alpha_3 = \frac{1}{2} \tag{4.56}$$

4.4.2 Time Step

The time step Δt is calculated according to the numerical stability condition. It is difficult to predict the turnover time by the smallest resolved eddies in MILES for a time accurate calculation. However the time step governed by the numerical stability condition is normally smaller than the turnover time by the resolved eddies from the experience of other LES calculations [60] where stability condition

governed Δt.

For one-dimensional convection problems numerical stability is predicted using the Courant-Friedrichs-Lewy (CFL) condition [72], where the CFL number C_{cfl} can be regarded as the ratio of two speeds, namely the wave propagation speed S and the grid speed $\Delta x/\Delta t$ defined by the discretisation of the domain:

$$C_{cfl} = \frac{S}{\Delta x/\Delta t}. \tag{4.57}$$

Here Δx is the grid spacing and Δt is the time step. For multidimensional viscous flow, the above equation is generalized to:

$$\Delta t = \frac{C_{cfl}\Delta x}{S}. \tag{4.58}$$

The wave propagation speed S is calculated in 2 parts (inviscid and viscous) by

$$S = \max_{i,j,k} \left[\underbrace{\left\{ \sum_{m=1}^{3} |u_{i,j,k}^{(m)}| + a_{i,j,k} \right\}}_{inviscid} + \underbrace{\frac{2\gamma\mu_{i,j,k}}{\rho_{i,j,k} RePr}}_{viscous} \right] \tag{4.59}$$

where i,j,k represent the 3-spatial dimensions, $u^{(m)}$ are the 3 velocity components, a is local speed of sound, γ is the ratio of specific heats, μ is molecular viscosity, ρ is density, Re is the Reynolds number and Pr is the Prandtl number.

For Jameson's Runge-Kutta Method, a maximum C_{cfl} of $2\sqrt{2}$ (≈ 2.8) can be used in theory; the current calculations use $C_{cfl} = 2.5$.

Chapter 5

Code Parallelization

5.1 Parallelization of Navier-Stokes code

The initial Navier-Stokes code was serial and written in Fortran 77. To be implemented on parallel computers the code has been rewritten in Fortran 90 and parallelized using the MPI communication library. The main reason for using Fortran 90 was to use the dynamic memory allocation function which does not exist in Fortran 77. Otherwise depending on the number of processors which can be used the code has to be recompiled. The grid sizes of test cases were too big to be fitted into a local memory of a processor.

Several different parallelization configurations have been tested and it was concluded that 1-directional domain decomposition, non-blocking communication, persistent request, master and slave topology and user-defined datatypes were best suited for the current situation. All of these techniques will be discussed in the rest of this chapter.

5.1.1 Parallelization Suitability Issues

Not all codes are easily parallelisable. Since a parallelized code is more difficult to debug than its serial version it is worth considering the complexities of parallelisation for the maintenance or modifications of the code.

An explicit Navier-Stokes solver can be easily parallelized by domain decomposition. Domain decomposition is similar to a multi-blocking scheme [29, 106].

In a multi-blocking scheme the computational domain is divided into several sub-domains for ease of grid generations and boundary condition specification. In the case of domain decomposition for parallel computing, an even distribution of grid points and a small amount of grid point data to be exchanged between sub-domains are as important requirement for efficiency. A more detailed description of domain decomposition follows in this section.

A very small portion of the current code is non-parallelisable. Therefore, according to the Amdahl's Law (Section 2.4.3) it is suitable for massive parallelisation.

5.1.2 Domain Decomposition

Basic Concept

Let's take 1-D decomposition of the central differencing scheme as an example where a new value is calculated by:

$$\phi_i^{n+1} = \frac{\phi_{i+1}^n + \phi_{i-1}^n}{2}, \quad i = 1, \cdots, 1000. \tag{5.1}$$

If 4 processors are used to share evenly the workload by data parallelism (i.e. dividing data to be processed by different processors), it is necessary to communicate between neighbouring processors to update the boundary values (Figure 5.1).

To calculate ϕ_{250}^{n+1} in processor 1, two values are needed: ϕ_{249}^n and ϕ_{251}^n. ϕ_{251}^n is a boundary value and cannot be calculated in processor 1 since the processor does not possess ϕ_{252}^n. Therefore ϕ_{251}^n is received from processor 2. In Figure 5.1 the boundary values for each processor to be received from its neighbouring processors is shaded, i.e., these values are not calculated but are received from its neighbouring processors.

Multi-dimensional Decomposition

In the previous section, a 1-dimensional decomposition is described. But as the current calculation is 3-dimensional, 2 or 3-dimensional decomposition may be better for minimizing the contact area between sub-domains. As the sub-domains

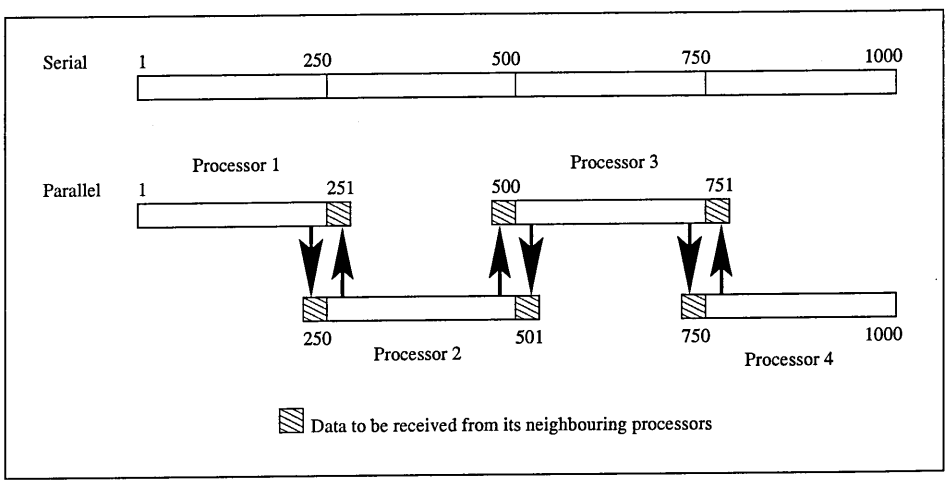

Figure 5.1: Schematic diagram of data transfers between processors

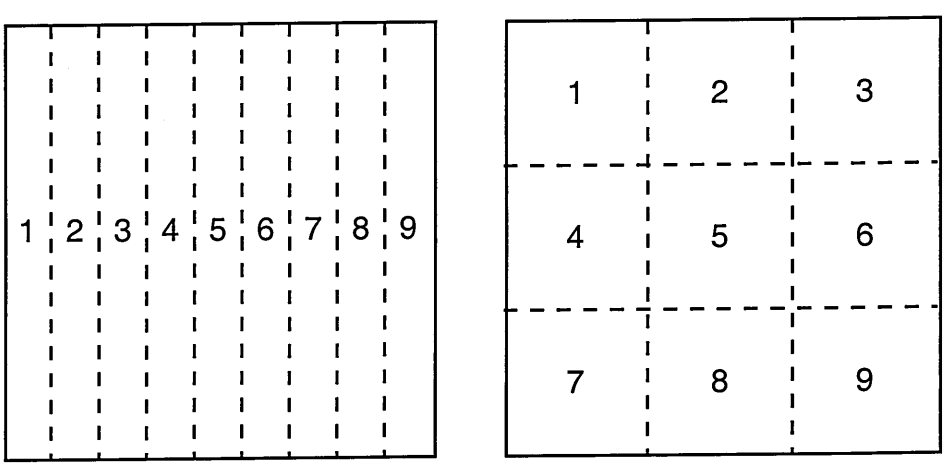

Figure 5.2: 1 and 2-dimensional decomposition

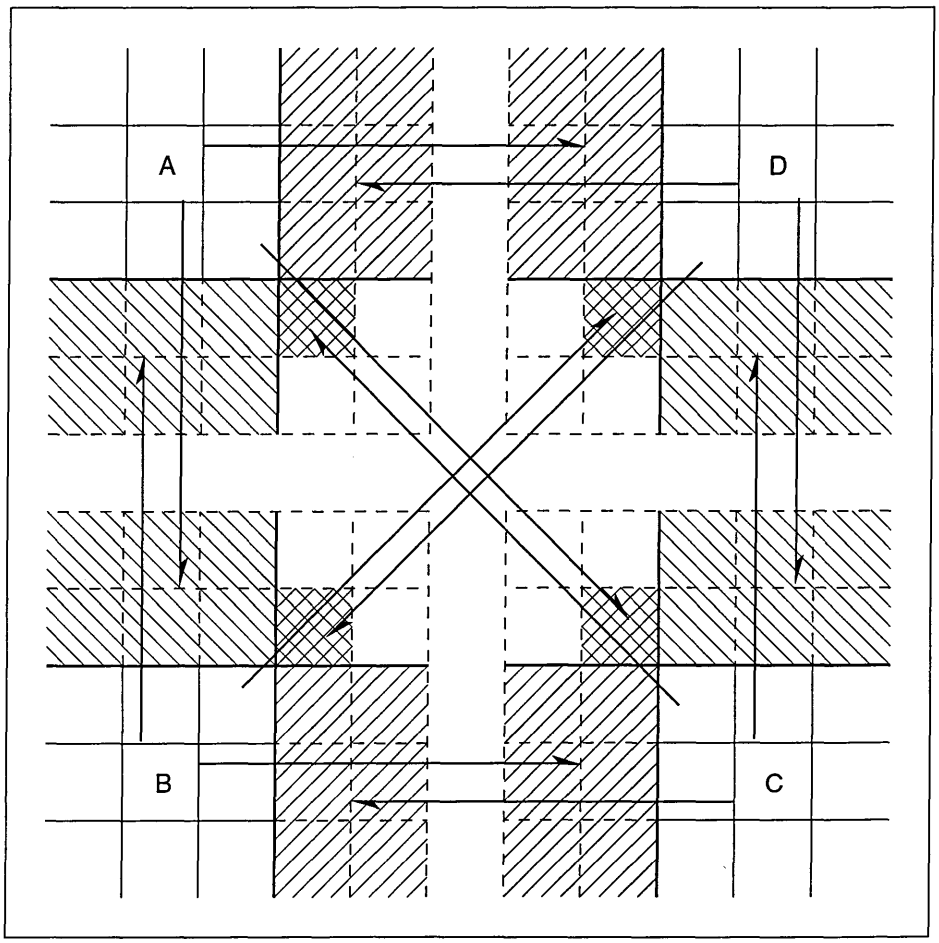

Figure 5.3: sub-domain interaction in a 2-dimensional decomposition

get closer to being cubes (for 3-D decomposition) or squares (for 2-D) the surface to volume or area ratio gets smaller. Figure 5.2 shows an example of 1 and 2-dimensional decomposition but when a domain is decomposed to many sub-domains, 3-dimensional decomposition has to be considered.

Figure 5.3 shows a schematic diagram of communication between the 2-dimensional sub-domains in our situation. It shows that even though the amount of data to be exchanged between the sub-domains with 2-dimensional decomposition is less than 1-dimensional decomposition, the number of sub-domains to be communicated has increased. In 1-dimensional decomposition, the maximum number of sub-domains to be communicated is 2. In 2-dimensional decomposition, it is 8, and in 3-dimensional decomposition, it is 24. At the start of each communication

there is a start-up latency; this is largely machine dependent. Therefore savings on the amount of data to be exchanged can be undone by increasing the number of sub-domains to be communicated.

The decision to decompose the domain into one or multi-dimensions depends largely on the kind of machine the program is most likely to run on and the amount of effort one can put in to parallelize a program.

5.1.3 Exchanged Data

Five different variables need to be stored for a complete solution, therefore five variables need to be exchanged between the sub-domains. In the current calculation ρ, u, v, w and p are chosen, but T could have been chosen instead of p for example.

Two neighbouring points in each direction are needed to calculate the flux through the block surface. Therefore two layers of data should be sent to neighbouring machines (Figure 5.3).

T and μ for the exchanged data are recalculated using Sutherland's law (Equation 3.19) and the ideal gas equation of state (Equation 3.20) respectively after the communication. Only one layer need to be calculated again for T and μ .

5.1.4 Master and Slave Programmes

A master and slave topology is used for the parallel version of the current program. The master program monitors slave programmes and occasionally outputs the progress of the calculation.

While the code runs on 'public' University workstations it also checks the progress of individual workstations to see if any one of them is particularly slow and holds up other workstations. If that is the case, it will make the slave machines finish after the next output of results. Then the master program chooses another set of slave machines to restart the calculation.

The slave programmes do all the calculations. Each slave machine writes its own results directly to the filing system. The results will be collected by a simple program when it is required.

5.2 Communication Time

It is needless to say that it is important to reduce communication time to get good parallel performance. Issues concerning communication time will be examined in this section.

5.2.1 Blocking/Non-Blocking Operations

It is desirable to overlap communicational and computational operations to "hide" communication time. If the computational time is relatively longer than the communication time then most of the communication time will be hidden by the computational time. This overlapping of communication and computational time can achieved with a non-blocking operation.

- In *Blocking operations* once the sending of and the receiving data are requested, all the procedures are stopped until communication is finished and the data buffers are safe to reuse.
- In *Non-blocking operations* after the sending and receiving of data, procedures are initiated and the next operations are started immediately. Later, additional calls are made to confirm the completion of the operations.

It is more complicated to set up the non-blocking operation than the blocking operation: there are more function calls to be made and it is more difficult to debug the code (see Section 5.4.1– 5.4.2). Also it is necessary to change the order of the computational operations so that there is a significant amount of computational operation to be done before the communicated data is required in the computation.

If we refer back to Figure 5.1, in processor 1 all the points $\phi_i, i = 1, \cdots, 249$ can be calculated without receiving data from processor 2 except ϕ_{250}. Therefore,

internal points are calculated first while ϕ_{251} is transferred from processor 2. Finally when ϕ_{251} is needed an additional call is made to confirm its arrival.

By re-arranging the order of the computation, straight comparison between serial and its parallel versions of a code becomes more difficult. This also means the maintenance of two versions of a code. Sometimes it is necessary to balance parallel performance and maintainability.

In the current approach to the non-blocking operation, the communication time was hidden with the computational time required for the boundary conditions. As the computation of characteristic boundary conditions for the far-field requires significant time most of communication can be done during this time.

5.2.2 Communication Start-up Latency and Persistent Communication Request

In all communication there is a start-up latency which increases the communication time. In fact, time will be lost even before starting the data send. In some systems, especially in a virtual parallel machine where the communication start-up latency is high, it is important to buffer the data and send it in bulk. If each communication is short the start-up latency time can be longer than the data transfer time.

In a situation where the same argument list is repeatedly executed within the inner loop of a parallel computation, it is also possible to reduce the communication start-up latency with a *persistent communication request.* The request is called once outside the loop and then it is repeatedly used to initiate and complete massages. It can be thought of as a communication port or "half-channel."

Persistent communication requests are associated with the non-blocking operations which were described in the previous section.

5.3 User-Defined Datatypes vs. Data Packing

There are two mechanisms provided by MPI to send an array section: user-defined datatypes and data packing.

- The user can define *derived* datatypes, that specify more general data layout. User-defined datatypes can be used in place of the basic, predefined datatypes.
- A sending process can explicitly pack noncontiguous data into a contiguous buffer, and then send it; a receiving process can explicitly unpack data received in a contiguous buffer and then store it in noncontiguous locations.

It is often possible to achieve equivalent data transfer using either mechanisms but there are pros and cons to each approach.

One consideration is the programming convenience. It is somewhat less tedious to pack the data into a one-dimensional array for communications rather then defining a derived datatype for the array pattern. On the other hand, it would be very tedious (and inefficient) to pack separately the components of each structure entry in the array. Defining a datatype is more convenient when this definition depends only on declarations; packing may be more convenient when the communication buffer layout is data dependent.

Another consideration is the amount of storage used. The use of packing requires additional storage for a *copy* of the data, whereas the use of derived data types requires additional storage and memory for a description of the data layout.

Another consideration is the required computational time. The packing code executes a function call for each packed item whereas the derived datatype code executes only a fixed number of function calls.

Both mechanisms send the same size message, so that there is no difference in communication time, However, if the buffer described by the derived datatype is not contiguous in memory, it may take longer to access. But in most cases one

may expect to achieve better performance with the derived datatype mechanism. The derived datatype mechanism is used in the current code.

With Fortran 90, it is tempting to use an *array section* to send a part of an array but this will not work with non-blocking communication. As the array section is copied to a temporary array before sending, it will be erased after the execution of the communication initialisation call even though the communication has not been completed.

5.4 Practical Points with MPI

Here are some practical points for developing and running a parallel program.

5.4.1 Fortran Language Binding Issues

The MPI Fortran binding is inconsistent with the Fortran standards in several respects. While these cause few problems for Fortran 77 programs, they become more significant for Fortran 90 programs, so users must exercise care when using new Fortran 90 features. The violations were originally adopted and have been retained because they are important for the usability of MPI.

The following MPI features are inconsistent with Fortran 90.

- An MPI subroutine with a choice argument may be called with different argument types.
- An MPI subroutine with an assumed-size dummy argument may be passed a scalar argument.
- Many MPI routines assume that actual arguments are passed by address and that arguments are not copied on entrance to or exit from the subroutine.
- An MPI implementation may read or modify user data (e.g., communication buffers used by nonblocking communications) concurrently with a user program executing outside MPI calls.

- Several named "constants" are not ordinary Fortran constants and require a special implementation.
- Multi-dimensional arrays may not be one-dimensional arrays of same size in a contiguous memory.

Additionally, MPI is inconsistent with Fortran 77 in a number of ways, as noted below.

- MPI identifiers exceed 6 characters.
- MPI identifiers may contain underscores after the first character.
- MPI requires an include file, `mpif.h`. On systems that do not support include files, the implementation should specify the values of named constants.
- Many routines in the MPI-2 standard have KIND-parameterized integers that hold address information. On systems that do not support Fortran 90-style parameterized types, `INTEGER*8` or `INTEGER` should be used instead.
- The memory allocation routine can't be usefully used in Fortran 77 without a language extension that allows the allocated memory to be associated with a variable.

Some of these problems can be less of a problem since they are widely supported even though they are not strictly Fortran standard. Some of the problems however, should be avoided. There are other more subtle problems like register optimization. A much deeper review of these problems with examples can be found in [107].

5.4.2 Developing and Debugging Parallel codes

There are multiprocess debuggers like **TotalView** but these are expensive to buy. It is possible to debug a parallel code using a conventional debugger like **dbx** or **ladebug** by attaching a debugger to each process. However you do not obtain any

information on the communications. It may be that added print statements are the only practical debugging method available for a parallel program. In this case, *barrier synchronization* [108] should be sensibly so that the print statements are executed on all of the processes before one of the processes fails and sends out a kill signal to rest of the processes.

It is possible that failures might occur from time to time and not all the time. This might happen if you have made a mistake on the non-blocking communication implementation. This kind of problem is hard to catch because the debugging effort slows the program down and so there is more time to finish the communication.

It is needless to say that the debugging of a parallel code is much harder than a serial one. As well as there being extra complexity in a parallel code, its errors can be very unpredictable and hard to track. Therefore, even though it may seems a waste of effort to maintain both serial and parallel code, it could be worthwhile to isolate errors in a parallel part of the code. As mentioned in Section 5.2.1, improvements in the parallel performance by significant modification of the serial code has to be balanced with the maintainability of the parallel version.

5.4.3 Process Creating Issues

One of the advantages of using MPI is that the resultant parallel code can be ported to virtually any computer platforms without much effort. Still there are a few things the MPI programmer needs to be aware of.

Running executables is one of the specifications not set by the MPI standard. This seemingly trivial fact can be troublesome if you have several codes to do different tasks and communicate with the MPI library. Most MPI implementations will let you run multiple executables simultaneously using `mpirun` commands but this is not always the case e.g. Cray computers.

The MPI-2 standard [107] specifies process spawning (creating processes dur-

ing runtime) but only a few MPI implementations support this at present e.g. LAM.

Chapter 6

Simulation of the Turbulent Boundary-Layer on a Flat Plate

A compressible turbulent boundary-layer on a flat plate is simulated using the MILES approach. Details of the simulation are presented in this chapter.

6.1 Reference Case Physical Conditions

As a reference case, the zero-pressure-gradient, flat-plate boundary-layer flow experiment by Shutts *et al.* [109] is chosen (this experiment is discussed by Fernholtz [3]; Case 55010501). As there also are DNS simulations by Rai *et al.* [28] and LES ones by Spyropoulos and Blaisdell [30], it makes an ideal reference case.

The freestream conditions are Mach number 2.25, Reynolds number $1.613 \times 10^4/\text{m}$ or 6.007×10^3 based on the displacement thickness and temperature 305°R (169.44°K).

6.2 Computational Grids and Test Cases

A number of different grid and computational domain sizes have been simulated. Because of limited computer resources, for implementing MILES a minimal number of grid points to capture the turbulence, and a large enough domain size to contain large eddies, had to be determined.

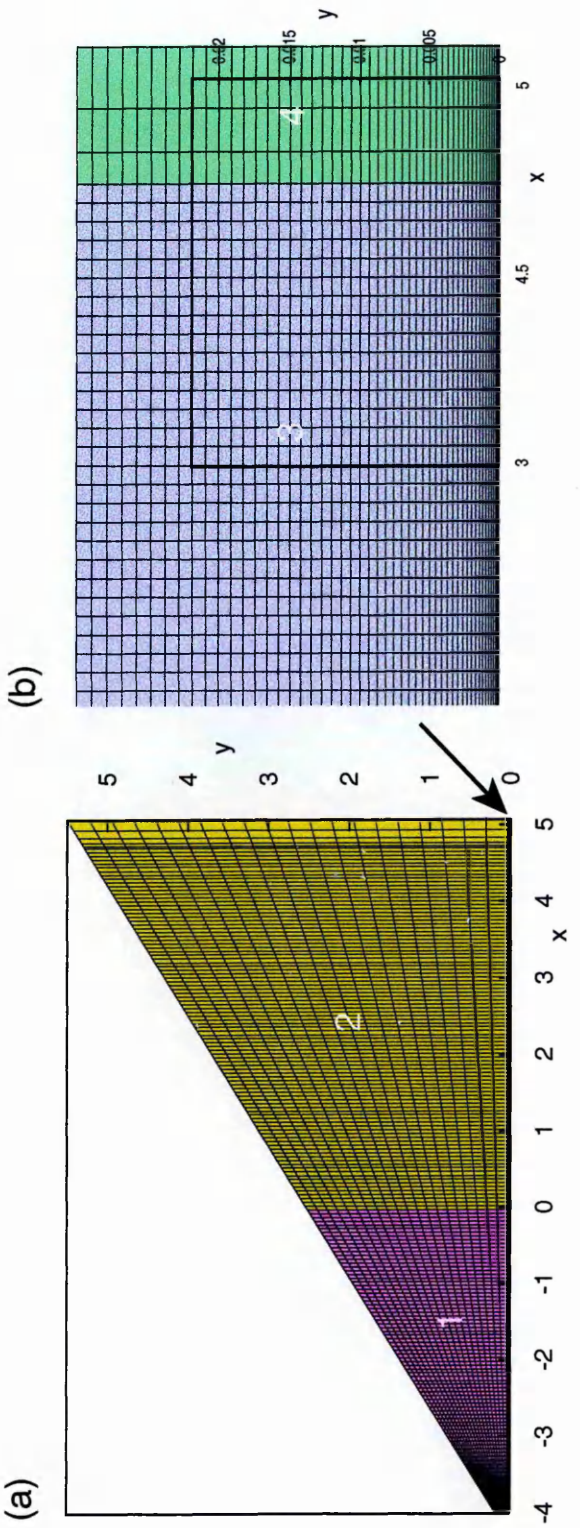

Figure 6.1: Plot of grid

Case	*Computational domain size (in)* $(x \times y \times z)$	*No of Grid points* $(i \times j \times k)$
1	$4.0 \times 0.034745 \times 0.2$	$305 \times 55 \times 31$
2	$4.0 \times 0.034745 \times 0.2$	$305 \times 55 \times 61$

Table 6.1: Test cases for 3-D calculations

It was found that Case 2 in Table 6.1 provides the minimum number of grid point and domain size for our test case. The effect of the grid size will be discussed further in Section 7.6.1.

Figure 6.1(a) shows the 2-D grid which is divided into four regions. The region 3 is so thin compare to other regions it is hardly visible but the boundary layer is resolved well in this region. Figure 6.1(b) is a more closeup plot of Region 3 and buffer region 4. The grid is stretched normal to the wall so that there are more points near the wall where the eddy scales are the smallest. A quadratic equation is used to stretched the grid. The closest point is 0.0001in away from the wall. If the grid is too stretched numerical error away from the wall will offset the benefit of stretching the grid to give more points towards the wall.

Grid spacing in the x-direction is at its finest in Region 1 to capture the leading edge shock. Region 2 is a buffer region which contains the leading shock. It does not really matter that the grid space is coarse in this region because the shock need not be resolved sharply here. Region 4 is another buffer region. This is to prevent flow coming back into the domain to make the calculation unstable (more details in Section 6.4).

In 3-dimensions, only Region 3 and 4 are calculated. The grid spacing in the streamwise and spanwise directions are uniform.

6.3 2-D Calculation for Initial and Boundary Conditions

Two-dimensional laminar boundary layer calculations were run to generate the initial and boundary conditions for 3-D calculations. This can be done because Region 1 and 2 are steady and laminar. It is important to have a sufficiently fine grid at the leading edge to capture the development of the shock. Therefore, it is better to generate inlet and upper boundary conditions for 3-D calculations from 2-D laminar calculations to save computation. Also, the local time step is used in the 2-D calculation because it is steady.

Numerical implementation in 2 dimensions is the same as for the 3-D calculation as described in Section 4, except that all the k components are ignored. In fact, the 2-D code has been used to verified the 3-D code in some situations. Osher's scheme was used to solve the inviscid terms but we do not expect to see any significant difference in 2-D results by using Roe's scheme since they are laminar calculations on well-resolved grids.

6.4 Boundary Conditions

Among the different boundary conditions used, the non-reflecting boundary condition for the far field was the most difficult to implement. There are many proposed methods [110–113] but Osher's Method for the Riemann Boundary problem is chosen for the current calculations. Therefore all the boundary conditions described below are applications of this method. More details of the method can be found in Spekreijse [103].

6.4.1 Far field

The upper buffer region (Region 2 in Figure 6.1) in the 2-D calculation was necessary to avoid the leading shock bouncing off the upper boundary and coming back into the domain. Therefore, in the 2-D calculation, only the supersonic inflow condition is possible whilst in the 3-D calculation the subsonic inflow and outflow

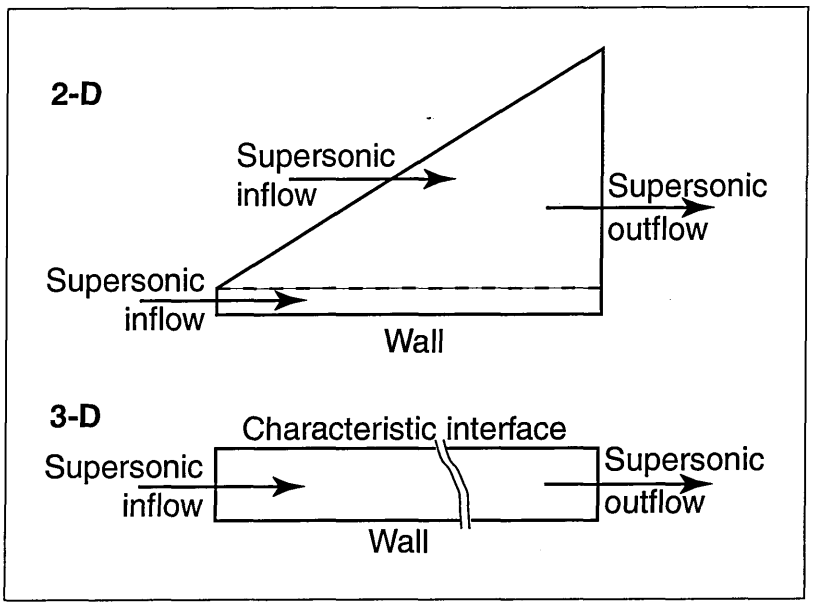

Figure 6.2: Boundary Conditions for 2-D and 3-D calculations

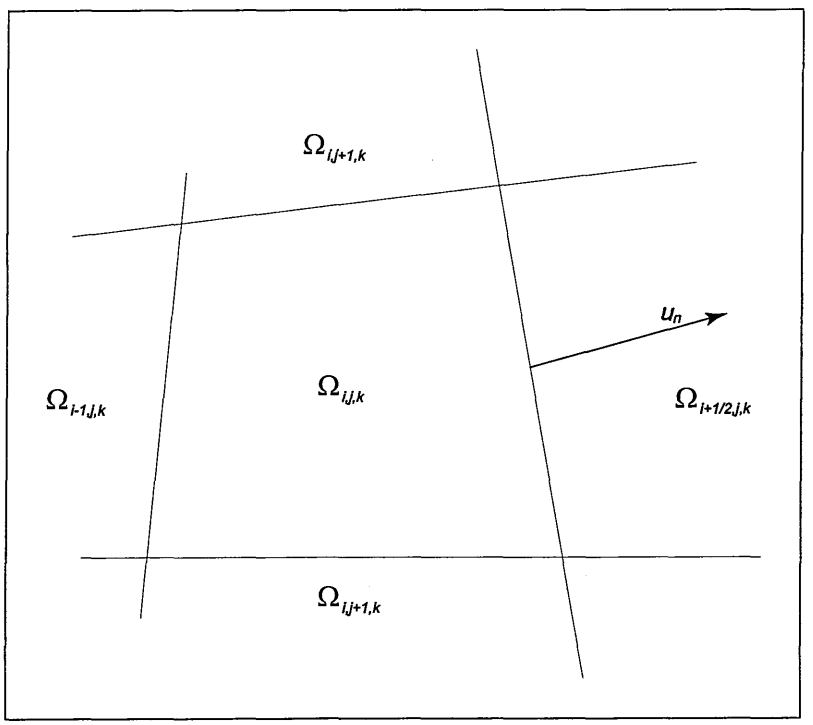

Figure 6.3: The boundary $\partial\Omega_{i+1/2,j,k} \subset \partial\Omega$ with local velocity component normal to the cell face.

conditions are also possible. Note that u_n in the following section is the velocity component normal to the cell face (figure 6.3) and c is the local speed of sound.

Supersonic Inflow ($u_n < -c$)

A full set of five boundary values are necessary. In the current calculations, velocity components u_B, v_B and w_B, pressure p_B and density ρ_B are specified. Subscript B indicates the boundary values. It would be equally sufficient to specify other values like speed of sound c_B. Viscosity μ_B and temperature T_B can be calculated using Equation (3.20) and (3.19) respectively.

Subsonic Inflow ($-c < u_n < 0$)

Four boundary values are necessary. In our case u_B, v_B, w_B, and c_B are specified. p_B and ρ_B are calculated as below. Subscript I represents the intermediate values and L the first value inside the domain.

$$c_I = c_L + \frac{\gamma - 1}{2}(u_L - u_B) \tag{6.1}$$

$$z_L = \ln\left(\frac{p_L}{\rho_L^\gamma}\right) \tag{6.2}$$

$$\rho_I = \left(\frac{c_I^2}{\gamma} e^{-z_L}\right)^{\frac{1}{\gamma-1}} \tag{6.3}$$

$$p_B = p_I = \frac{\rho_I c_I^2}{\gamma} \tag{6.4}$$

$$\rho_B = \frac{\gamma p_B}{c_B^2} \tag{6.5}$$

Subsonic Outflow ($0 < u_n < c$)

One boundary value is necessary. In our case, pressure p_B, is prescribed. Again subscript B indicates the boundary values, subscript I represents the intermediate values and L the first value inside the domain. The remaining boundary values are

determined as follows:

$$v_B = v_L \tag{6.6}$$

$$w_B = w_B \tag{6.7}$$

$$\rho_B = \left(p_B e^{-z_L}\right)^{\frac{1}{\gamma}} \tag{6.8}$$

$$c_B = \sqrt{\gamma p_B / \rho_B} \tag{6.9}$$

$$u_B = u_L + \frac{2}{\gamma - 1}(c_L - c_B) \tag{6.10}$$

6.4.2 Inflow

As the flowfield is supersonic at the inflow boundary, a supersonic inflow boundary condition, as explained in the previous section using a full set of 5 values is necessary. The laminar boundary layer generated in the 2-D calculation (Section 6.3) is superposed with sinusoidal disturbance and gaussian random noise to be used as an inlet boundary condition for the 3-D calculation.

$$u_{in} = u_l - u_l^2 e^{-\alpha y}\left[a_1 \sin(\beta t) + a_2 \Gamma\right] \tag{6.11a}$$

$$v_{in} = v_l + u_l^2 e^{-\alpha y}\left[a_1 \cos(\beta t) + a_2 \Gamma\right] \tag{6.11b}$$

$$w_{in} = w_l + u_l^2 e^{-\alpha y}\left[a_2 \Gamma\right] \tag{6.11c}$$

$$p_{in} = u_l + u_l^2 e^{-\alpha y}\left[a_1 \sin(\beta t) + a_2 \Gamma\right] \tag{6.11d}$$

Here subscript '*in*' represents the inflow condition, subscript l the laminar solution, β the frequency of sinusoidal disturbance, t the time, Γ the gaussian random number, and a_1 and a_2 control the amplitude of the disturbances. Function $e^{-\alpha y}$ exists to let the disturbance gradually disappear when moving away from the wall.

6.4.3 Outflow

The supersonic outflow boundary condition is implemented at the outlet. But it is possible that the outflow may become subsonic inside the boundary layer and even come back into the domain if the boundary layer becomes turbulent enough. Therefore it is necessary to have the buffer region at the outlet (region 4 in Figure 6.1) and pressure correction at the subsonic locations. The grid spaces inside the buffer region are gradually increased to smooth out turbulence. Also, the pressure at subsonic locations is set to the pressure at the first grid point upstream of

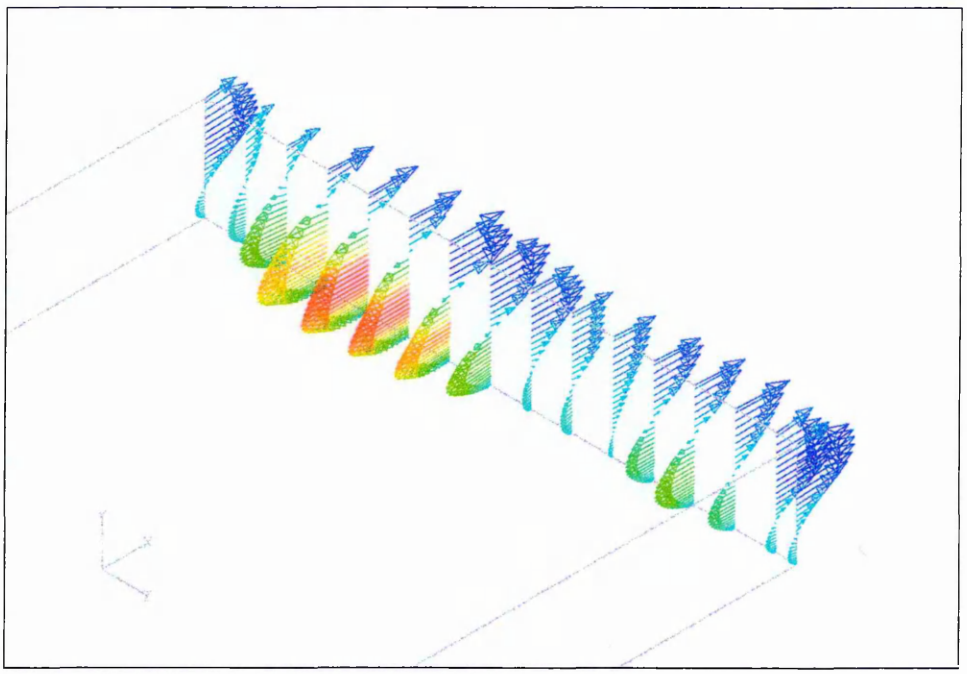

Figure 6.4: Vector plot of the outflow boundary condition just before blow-up in a calculation without the pressure correction at subsonic location.

the exit location (on the same vertical grid line) that becomes supersonic.

Figure 6.4 is a vector plot of the outflow boundary condition just before blow-up in a calculation without the pressure correction at subsonic location. The arrows have been drawn every 3 points in j-direction and 4 points in k-direction to visualize the vector plot better. It is clear from the plot that flow is coming back into the domain. Since the outflow boundary condition is set to supersonic outflow, this will make the calculation unstable and eventually leads to the calculation blowup.

Supersonic Outflow ($u_n > c$)

No boundary values are specified. All flow variables are extrapolated from inside points.

6.4.4 Solid Wall

An adiabatic, no-slip wall condition was implemented i.e. $\left.\frac{\partial p}{\partial y}\right|_w = 0$, $\left.\frac{\partial T}{\partial y}\right|_w = 0$ and all the velocity components were set to 0. The numerical scheme becomes one-sided and only second-order accurate.

6.4.5 Cyclic Boundary Condition

Cyclic boundary condition was imposed in the spanwise direction making the computational domain infinitely wide. The data exchanges are similar to the data communications between blocks in parallel computing (Section 5.1.3) but T and μ are copied rather than recalculated.

Chapter 7

Results

7.1 Monitoring the Calculation

7.1.1 Visual Inspection

With unsteady calculations it can be quite difficult to know whether the flow is settled downed statistically or it is still evolving. This is especially true with high level simulations like MILES, LES and DNS where the boundary and initial conditions have great influence on the calculation progress.

Visual inspection is the obvious choice for monitoring and this normally means using visualization software. Since the amount of data which these high level simulations generate is so huge it may not be practical to do visual inspections for flow evolution. Not only is a large amount of disk space needed to store data at given interval, but the time required to write data to the disk may be significant compared to the computational time. Nevertheless visual inspection is still the best way to monitor flow field development and spot obvious errors.

7.1.2 Residual

One of the parameters that can be viewed to see the change of the simulated flow field is the *residual* $\Re$. $\Re$ can be defined as the mean square root of the change in all variables. The vector form of Navier-Stokes Equation (3.36a) is expressed as following:

$$\frac{\partial \vec{\mathbf{Q}}}{\partial t^*} + \nabla \cdot \vec{\mathbf{E}} = 0 \tag{3.36a}$$

Therefore $\Re$ can be written as:

$$\Re_{i,j,k} = \sqrt{\sum_{l=1}^{5} \left(\frac{\partial Q^l_{i,j,k}}{\partial t^*} \right)^2} = \sqrt{\sum_{l=1}^{5} \left(\nabla \cdot E^l_{i,j,k} \right)^2} \tag{7.1}$$

where superscript l represents 5 variables at each point. $\Re$ is also normalized with the first value i.e. $\Re' = \Re/\Re_1$ so that $\Re'$ starts from 1.

$\Re'_{i,j,k}$ can be analyzed individually to see which part of the flow is changing or sum of all the $\Re'$ can give a good indication of whether whole flow is changing. We will denote the sum of the $\Re'$ as $\Re^*$.

In an unsteady calculation, $\Re^*$ increases if change in flow field becomes more rapid and vice versa. If $\Re^*$ stays in a constant value, the flow field might change locally, but overall, the flow field is statistically stable (see Figure 7.17).

In practice, when turbulent boundary layer calculations become statistically stable, $\Re^*$ will oscillate at a certain value (see Figure 7.41). At this point, the skin friction coefficient can be plotted along the plate to see whether there is an overall change. If there is no significant change then the flow is statistically converged.

$\Re'$ and $\Re^*$ can be analyzed for a steady calculation in the same way but as the solution converges to a final steady state solution, $\Re'$ and $\Re^*$ should approach zero.

7.1.3 Skin Friction coefficient

For a flat plate, skin friction coefficient at the wall C_f is another good indication of flow change and turbulence development. It is defined as:

$$C_f = \frac{2\tau_w}{\rho_\infty u_\infty^2} \tag{7.2}$$

where τ_w is the wall shear stress as defined in Equation 2.2.

Figure 7.1 is a plot of skin friction coefficient, showing the disturbance from the inlet at the start of the simulation travelling down the plate. It is not until

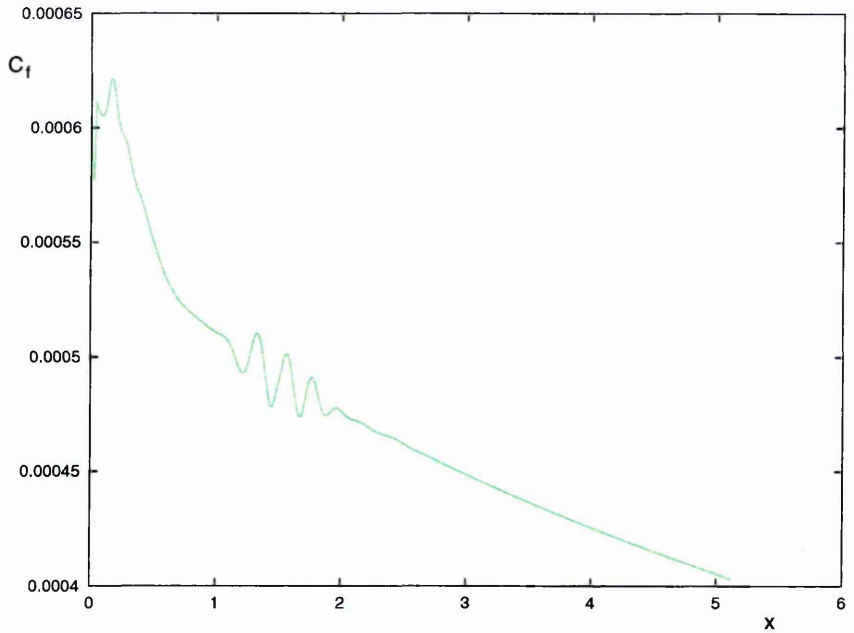

Figure 7.1: Plot of skin friction coefficient on flat plate which shows the disturbance at the inlet traveling down the plate.

the disturbance has travelled completely to the outlet that the turbulent boundary develops in any part of the domain. This reinforces the notion that until all flow conditions are correct in the domain the turbulence will not develop. This skin friction coefficient is calculated after 32,000 iterations of the calculation shown in Figure 7.41.

Figure 7.2 shows a schematic plot of skin friction on a flat plate over various stages of flow state. The real skin friction coefficient will be much more volatile in the turbulent boundary layer region. The line can be viewed as mean values.

At a laminar state the skin friction coefficient of the test case can be described well with the Blasius laminar skin friction coefficient line obtained from a boundary layer analysis [1]. The skin friction coefficient is less than 0.0005 and it is gradually decreasing along the plate. During the transition the skin friction coefficient will raise noticeably to a level between 0.002 and 0.0025. In the turbulent region, the skin friction coefficient will gradually decrease again but the value will still be much higher than the laminar one.

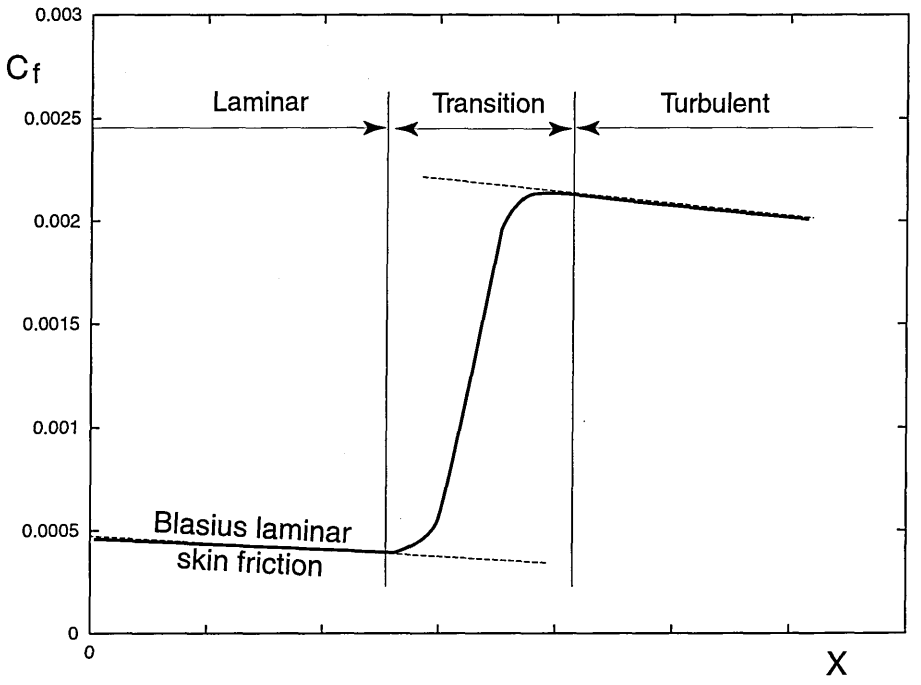

Figure 7.2: Schematic plot of skin friction coefficient on a flat plate

7.1.4 Mean Velocity Profile

As mentioned in Section 2.1.1, the mean velocity profile of a turbulent boundary layer can be described well with the law of the wall. The mean velocity profile can be obtained with either spatial, time or ensemble (i.e. both) averaging. If the boundary layer is statistically unchanging, the three averages should produce the same mean velocity profile if there are enough samples. When the flow is statistically changing, in case of flow over a flat plate, space averaging is the easiest method and the most responsive to change of flow condition. The mean velocity profile of the current results are spatially averaged in the spanwise direction.

Figure 7.3 shows normalized mean velocity profiles at various stages together with the law of the wall at $x = 4.5$. Both laminar and turbulent boundary layers initially follow the $u^+ = y^+$ line. In the turbulent boundary layer this region is called the linear sublayer. The two profiles diverge with each other at $y^+ \simeq 8$. The laminar velocity profile follows the $u^+ = y^+$ line whilst the turbulent boundary layer will start to follow the $u^+ = 2.5 * log(y^+) + 5.5$ line. Both of the profiles

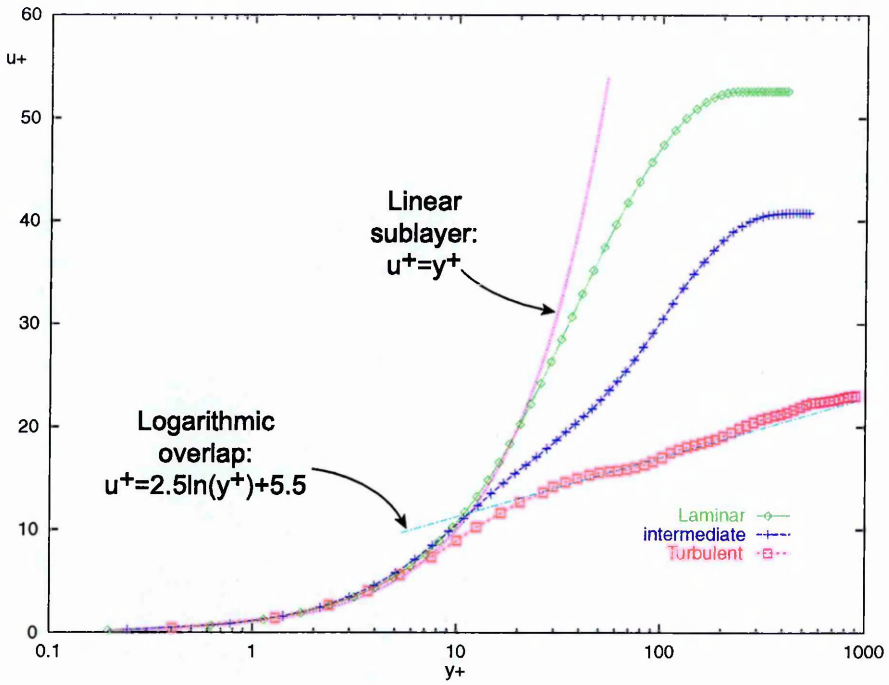

Figure 7.3: Plot of normalized mean velocity profiles at various stages together with the law of the wall.

will level off at a value which means no more change is occurring and it is outside the boundary layer. The laminar profile will level off at much higher u^+ but lower y^+ than the turbulent one.

Figure 7.3 also shows a velocity profile which is in-between the turbulent and laminar boundary layer. The velocity profile in the transition period changes so much that it may not look like this. During the simulation the initial laminar velocity profile at a point will gradually change to this transitional velocity profile before finally becoming a turbulent one. Also if the numerical method is too dissipative or the grid is too coarse to maintain a fully turbulent boundary layer, the velocity profile will look like this when the simulation has stopped changing statistically.

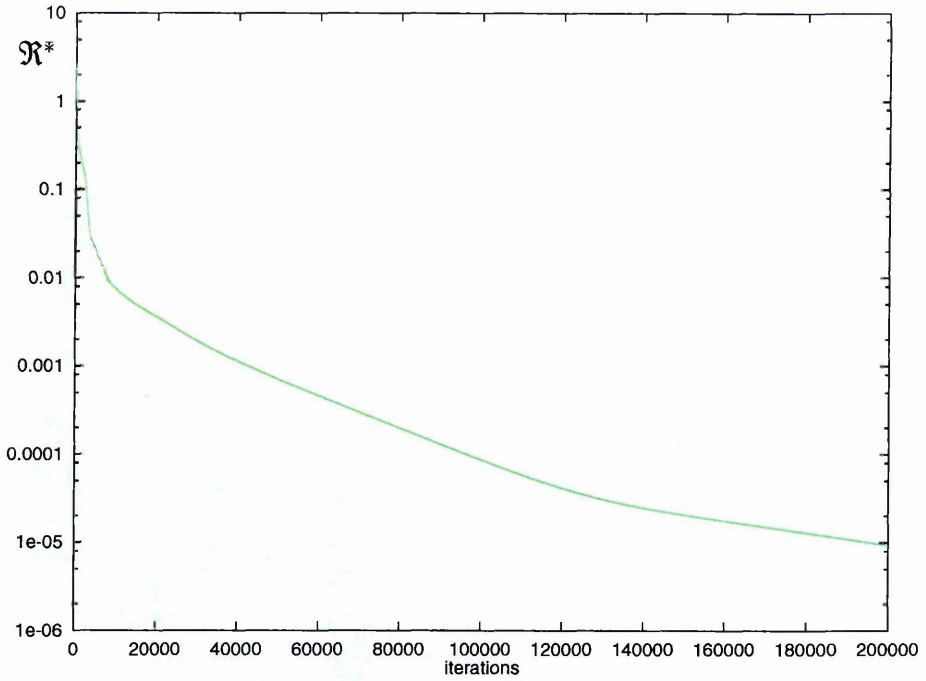

Figure 7.4: Plot of sum of residuals

7.2 2-D Calculation for the Initial and Boundary Conditions

Two-dimensional laminar calculations are run to generate the initial and boundary conditions. Only results from one calculation is presented here.

Figure 7.4 shows the plot of the sum of residuals $\mathfrak{R}'$ against the number of iterations. Since this is a steady calculation the sum of residuals shows the convergence of the solution. Even after 200,000 iterations the sum of residuals are still changing but not very much (note that the y-axis is in log scale). Since the solution will be used for the inlet boundary condition and the initial condition in 3-D, it should be more than adequate to stop at this point.

Figure 7.5 shows the plot of pressure. The leading shockwave is captured in grid region 1 and 2 (Figure 6.1). The resolution of the shockwave in region 2 is generally low but the region is far away from the boundary layer and not simulated in the 3-D calculation.

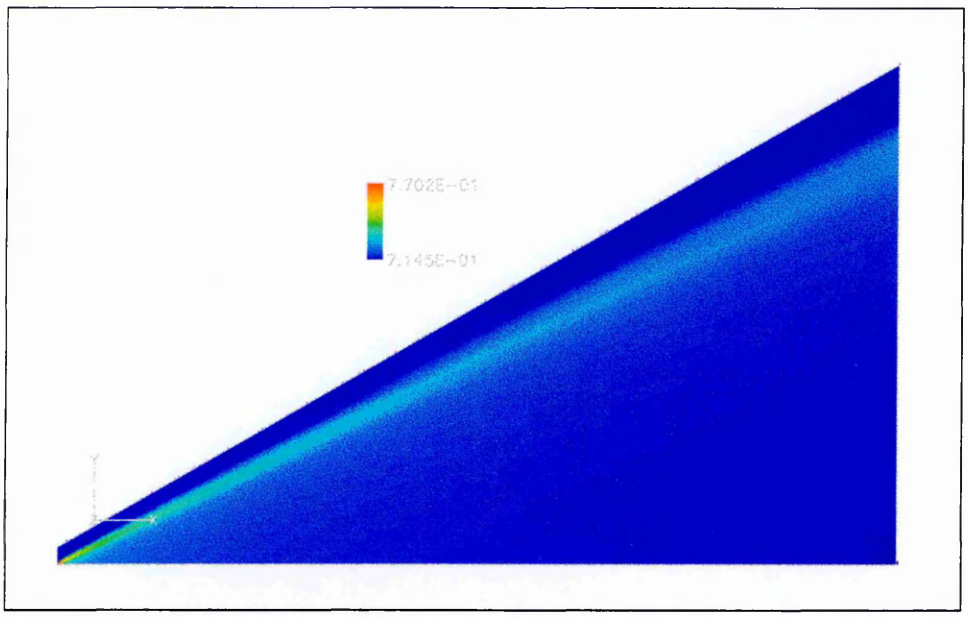

Figure 7.5: Plot of pressure from a 2-D calculation

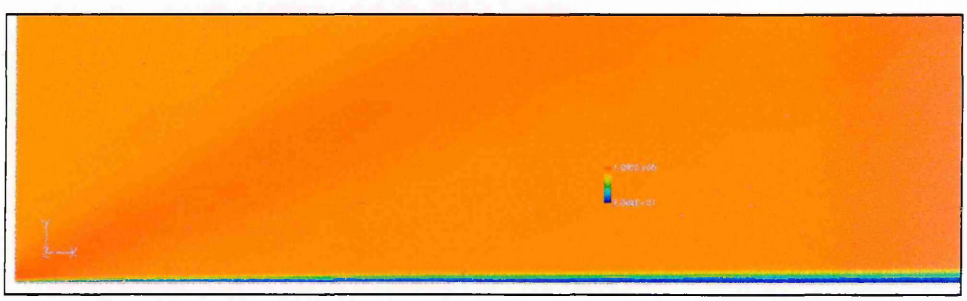

Figure 7.6: Plot of density at the leading edge from a 2-D calculation

The upper boundary is aligned with the shock to improve resolution. If the shockwave does not exit cleanly and touches the uppper boundary it will reflex back into the domain. This is the prime reason to have an upper buffer region so that the shockwave exits from the outflow boundary of the inner domain.

Figure 7.6 shows a close-up plot of density at the leading edge. The shockwave looks very blunt, but in fact, it is the thinness of the boundary layer which makes it look so.

The growth of the boundary layer can be seen more easily with a plot of

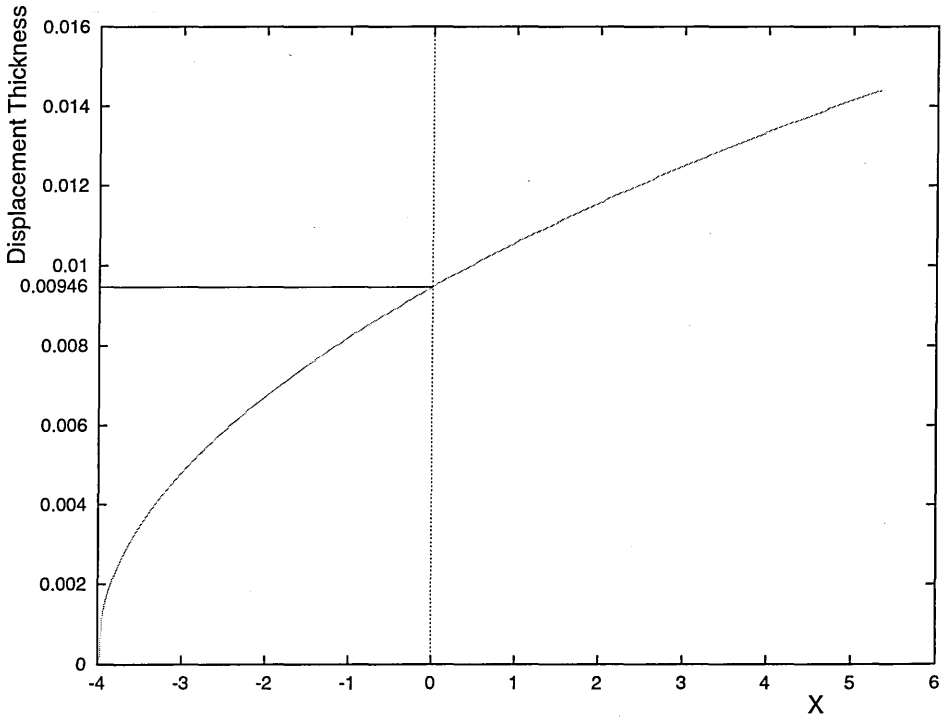

Figure 7.7: Plot of displacement thickness along the line from a 2-D calculation

displacement thickness (Figure 7.7). The boundary layer thickness at $x = 0$, where the data is taken for the inlet boundary conditions in 3-D calculations, is 0.00946. The experimental measurement from the same case by Shutts *et al.* [109] is 0.00953. Therefore, the Reynolds number based on displacement thickness at the inlet boundary is 6.007×10^3.

Figure 7.8 shows a plot of u-velocity component used as an inlet boundary condition for 3-D calculations. The calculated u-velocity is also compared to the Blasius solution for flat plate flow [1] in Figure 7.8 and it matches reasonably well. Figure 7.9 and 7.10 are plots of temperature and density respectively the same position.

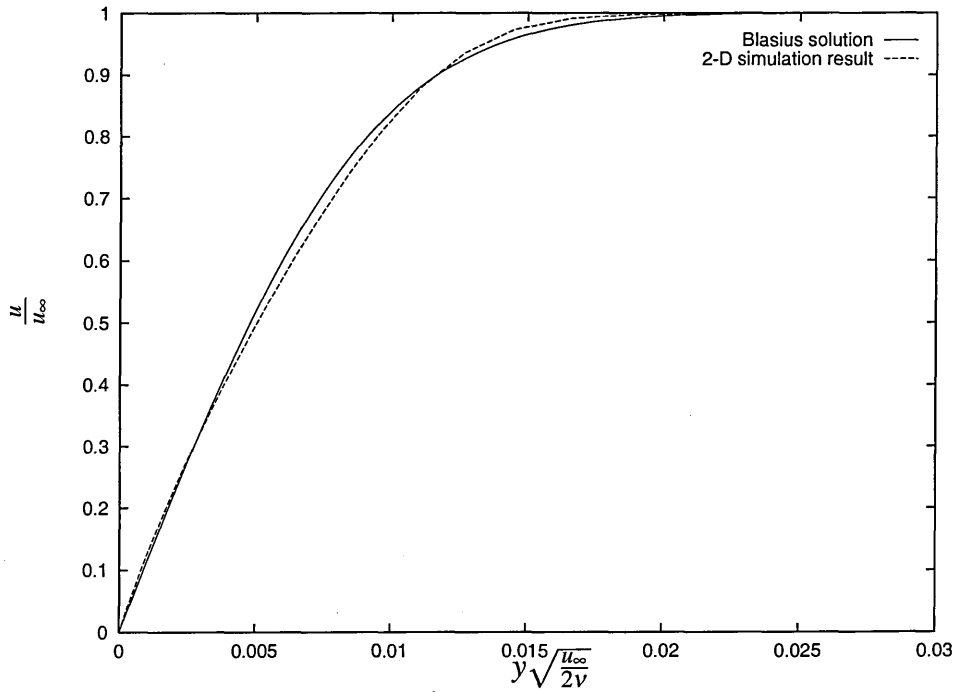

Figure 7.8: Plot of u-velocity component from a 2-D calculation and Blasius solution

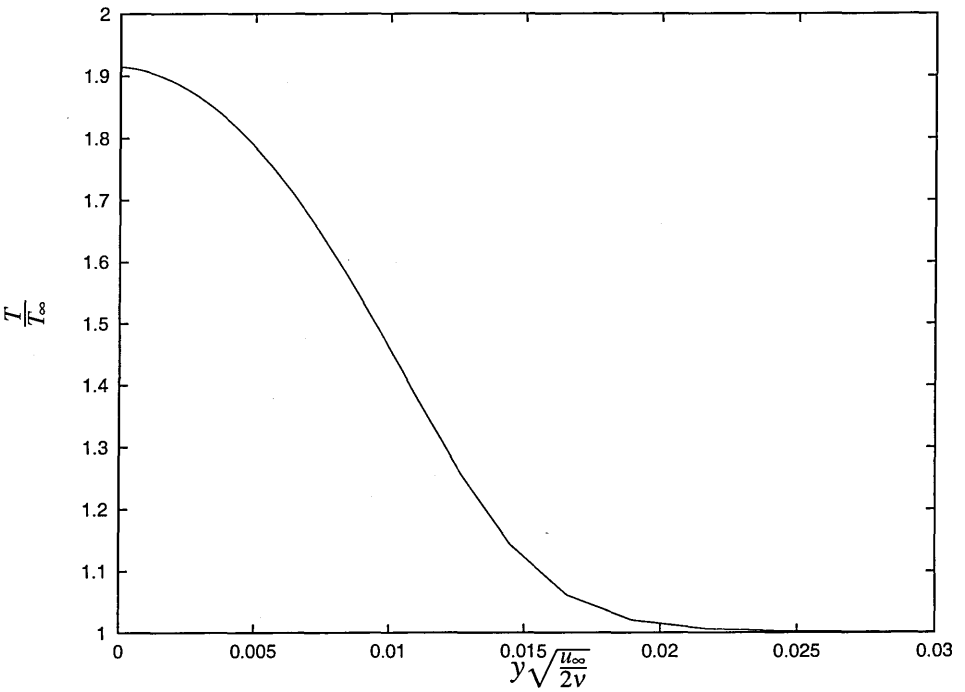

Figure 7.9: Plot of temperature from a 2-D calculation

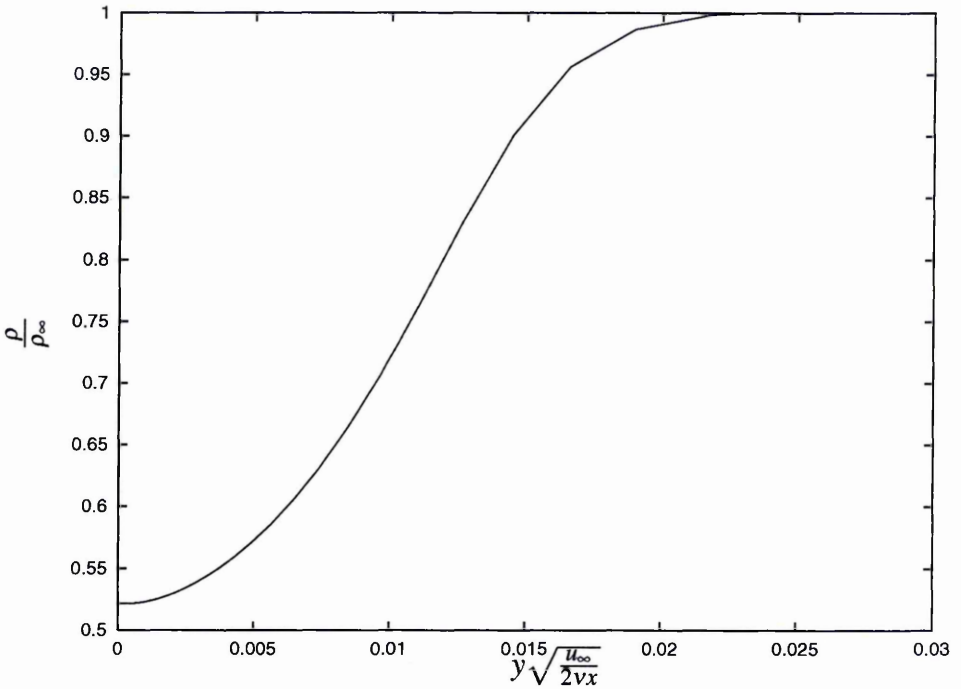

Figure 7.10: plot of density from a 2-D calculation

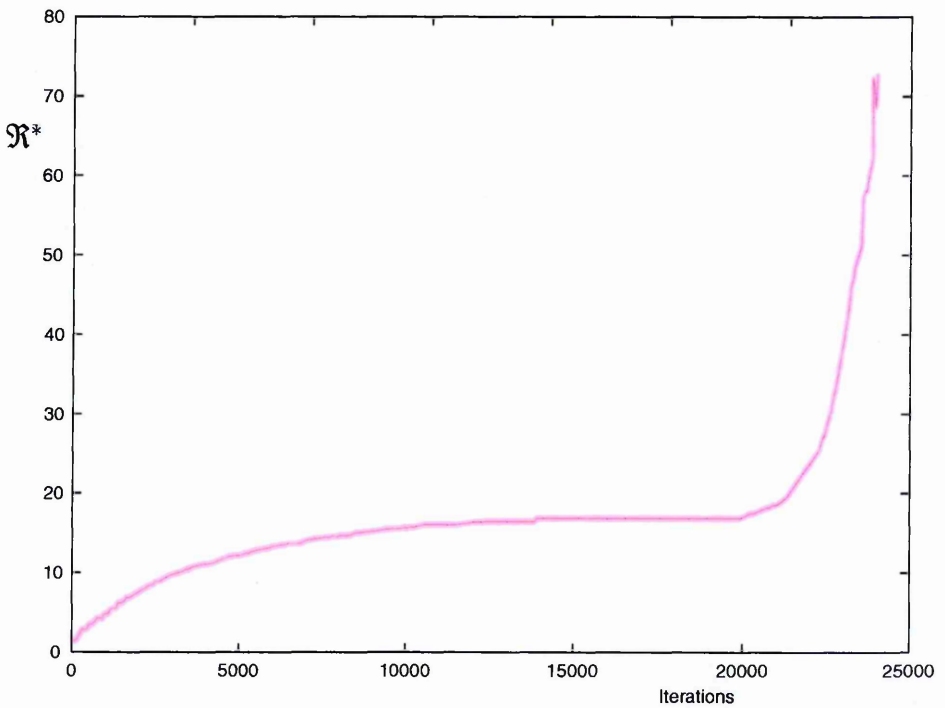

Figure 7.11: Plot of $\Re^*$ when Central Difference Scheme was used.

7.3 Central Difference Scheme

7.3.1 Monitoring Calculation

Among the three difference schemes used to discretise the inviscid terms (central difference, Osher's, Roe's scheme), the central difference scheme has least numerical dissipation. The current central difference scheme for the inviscid term is 2nd-order accurate which is lower than schemes used in traditional LES where sixth or eighth orders are common. Nevertheless, it is found that there is not enough numerical dissipation to prevent the calculation from being unstable and producing floating point errors. Results presented in this section are of case 1 in Table 6.1 and taken just before a floating point error.

Figure 7.11 shows a plot of $\Re^*$. The value of $\Re^*$ initially rises and then flattens out. During this period the disturbances from the inlet boundary condition travels down the domain. After the disturbances have travelled all the way down the domain, $\Re^*$ starts to rise quite sharply. This means the change in solution is getting more vigorous and turbulence is starting to develop. $\Re^*$ reaches a peak and soon after the calculation stops due to a floating point error.

7.3.2 Visualization

Depending on the contour colourings, plotting values and viewing angles, plotted results can look different. This is true for visualizations in other sections as well. Nevertheless, it is still a good method to show the overall pictures.

Figure 7.12 shows the plots of density viewed from (a) the front, (b) the side and (c) the top. It can be seen that chaotic instability resembling a turbulent flow is developing inside the domain. The sinusoidal wave from the inlet boundary condition is visible in plot (c).

Figure 7.13 shows an iso-surface plot of pressure, $p^* = 0.146344$. As well as the instability developing in the flow, the disturbances from the inlet can be seen more clearly than in the density plots. Two types of disturbances were applied

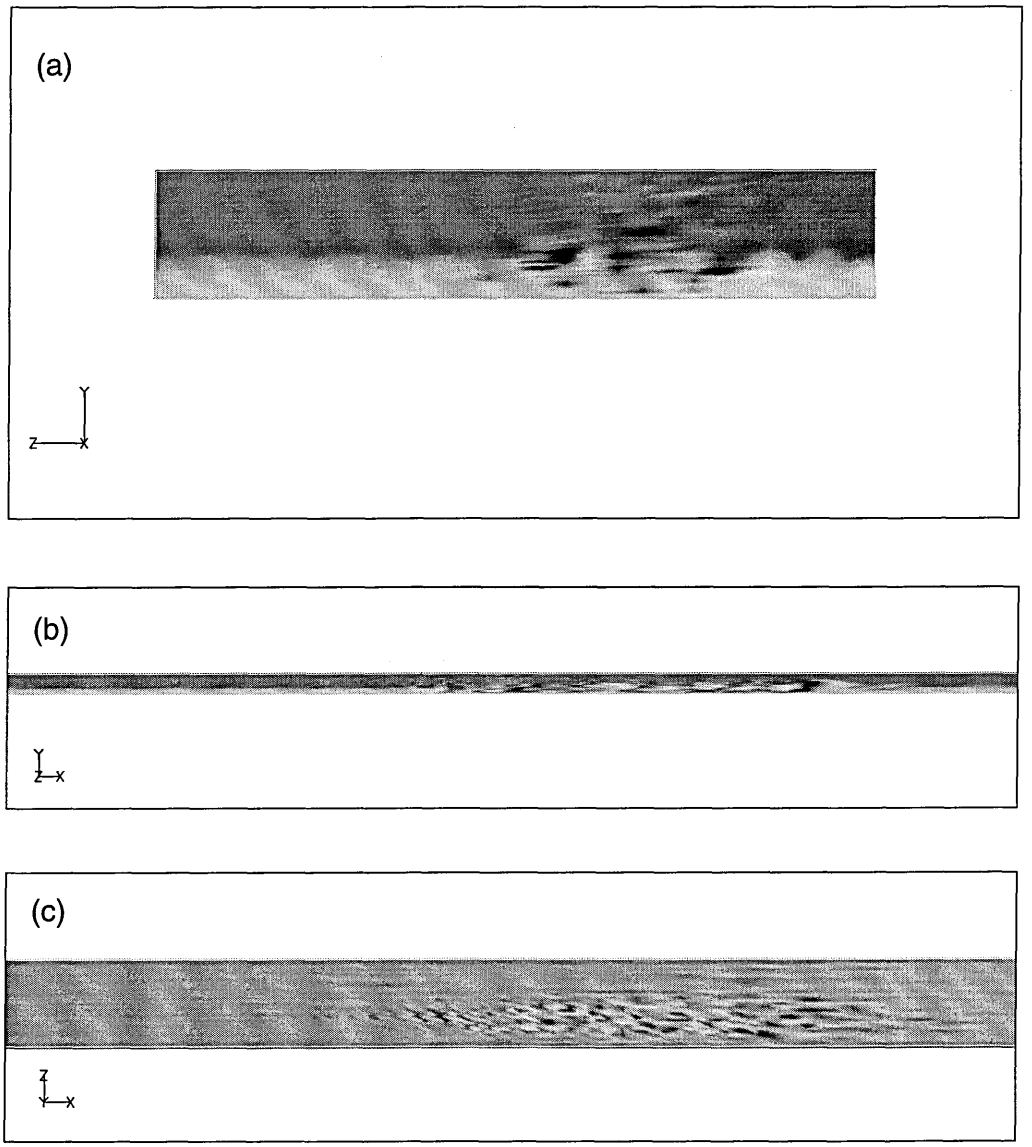

Figure 7.12: Density plots viewed from (a) front, (b) side and (c) top.

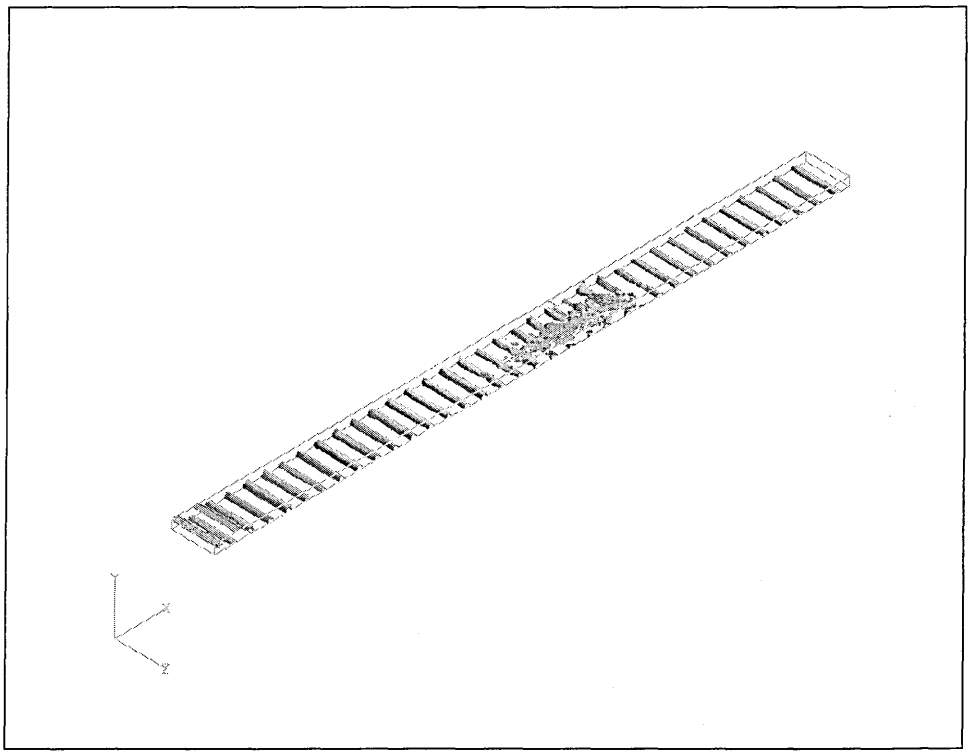

Figure 7.13: Iso-surface plot of pressure, $p^* = 0.146344$ when Central Difference Scheme was used.

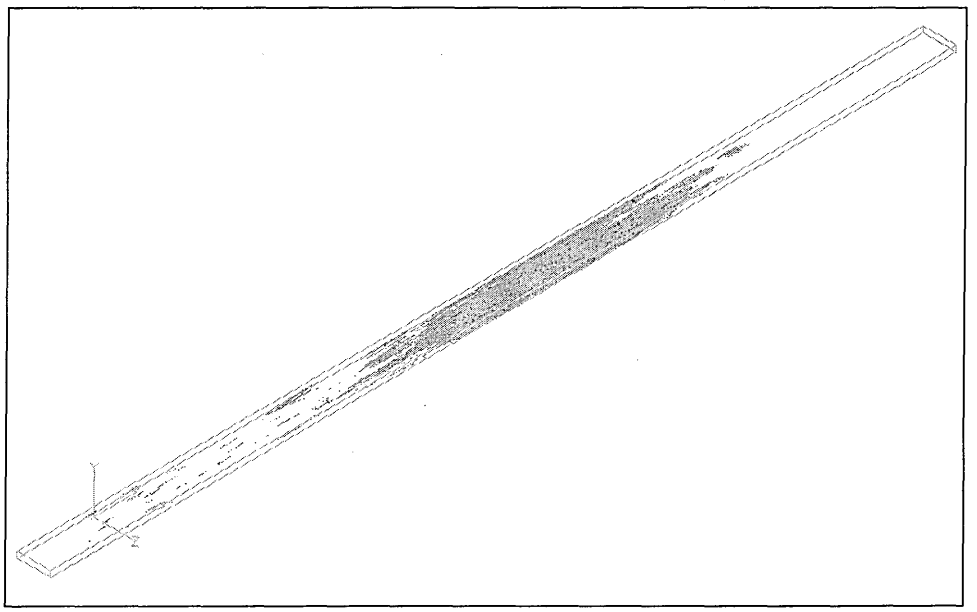

Figure 7.14: Iso-surface plot of $\omega_x = -5$ when Central Difference Scheme was used.

from the inlet; random noise and sinusoidal wave (see Section 6.4.2). The random noise quickly dies away but the sinusoidal wave travels all the way to the outlet. Interestingly, even with the central difference scheme, the disturbance from the inlet had to travel all the way to the outlet before instability in the flow started to develop in the middle of the domain.

The development of turbulence can be more readily seen from the iso-surface plot of $\omega_x = -5$ (Figure 7.14). It seems that the disturbance from the inlet has a strong influence on the start of turbulent development. With other schemes the turbulence might start from the outlet boundary (where Reynolds number based on displacement thickness is the highest) before the development starts to travel upstream.

7.3.3 Analysis of Results

Figure 7.15 shows a plot of skin friction coefficient. The amplitude of disturbances at the inlet are initially amplified and then maintained throughout the domain. The calculation is stopped when turbulence starts to develop at $x = 3.0$. This suggests that the central difference scheme is very non-dissipative and as soon as turbulence

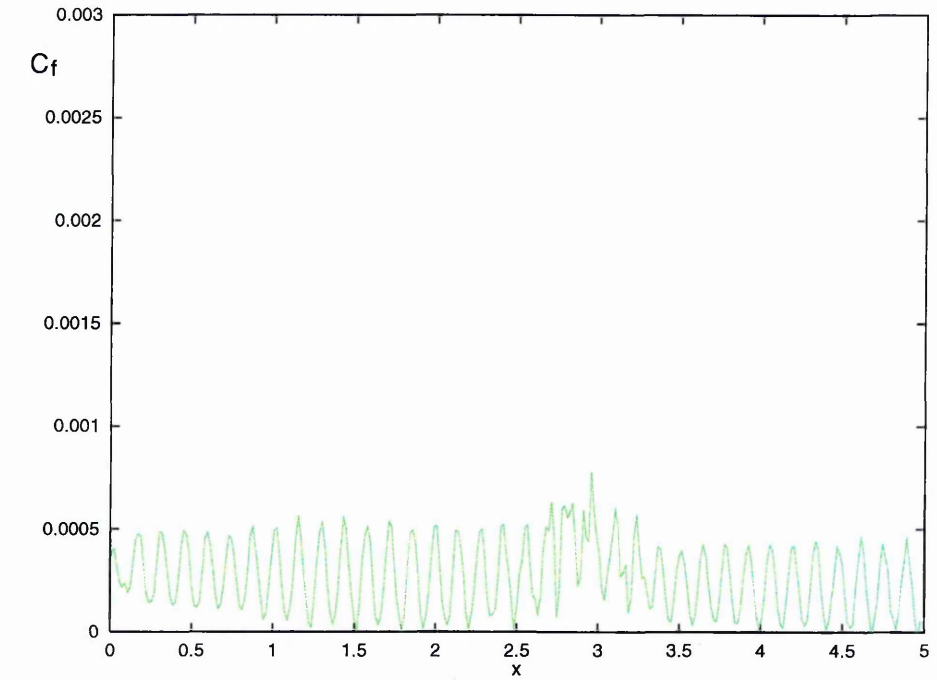

Figure 7.15: Plot of Skin Friction coefficient when Central Difference Scheme was used.

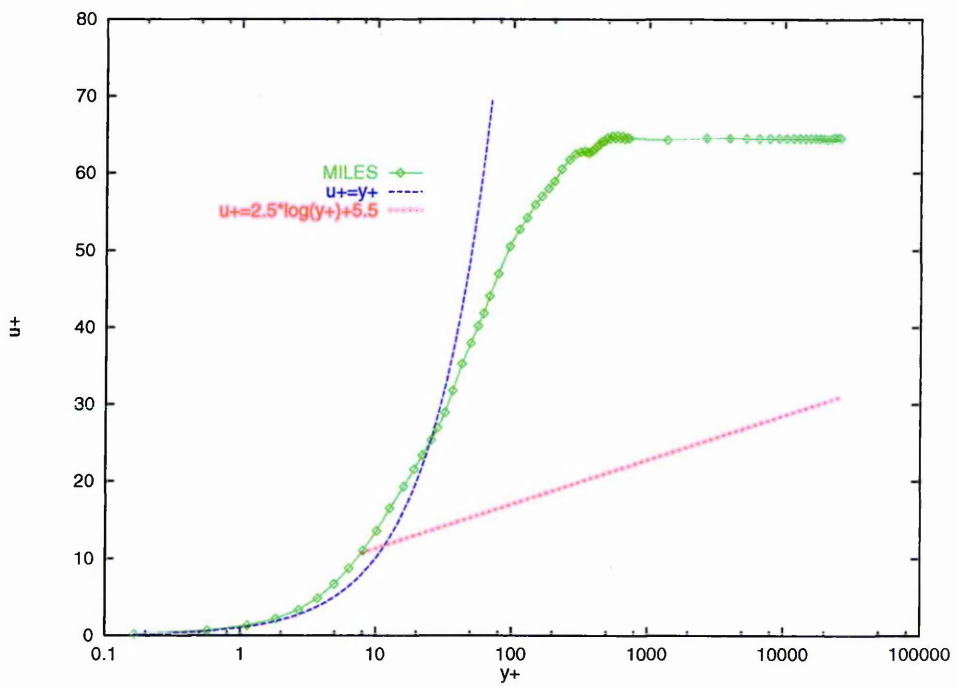

Figure 7.16: Plot of normalized mean velocity profile when Central Difference Scheme was used.

develops, an additional mechanism like SGS modelling is necessary to dissipate the energy build up.

Figure 7.16 shows a plot of the normalized mean velocity profile together with the law of the wall, where the flow instability has developed. The profile shows that the boundary layer is still not turbulent. The profile has been spatially averaged in z-direction. Even so, the line is not smooth and this suggests the flow varies significantly in the z-direction.

For MILES to work there has to be just enough numerical dissipation to maintain the chaotic nature of turbulence and to retain numerical stability. Increasing the number of grid points for high resolution will reduce the amount of energy dissipated by subgrid-scale eddies, but this will require more computational resources. If the resolution is sufficiently high, the amount of energy dissipated by the subgrid-scale eddies will equal the energy dissipated by the numerical errors even with the central difference scheme. In this case, if the numerical accuracy is high enough to capture the smallest eddy scales in a flow, the simulation will be a DNS. If the numerical accuracy is not high enough, it will be an under-resolved DNS and its results are much more debatable.

7.4 Osher's Scheme without a Limiter

7.4.1 Monitoring Calculation

Osher's scheme is a popular shock capturing method because it is numerically very stable. However this proves to be the reason why it is not suitable for MILES. It has too much inherent numerical dissipation to maintain turbulence.

Figure 7.17 shows a plot of $\Re^*$ for a simulation with Osher's scheme. Initially $\Re^*$ rises which means that the flow is changing more vigorously as iteration progresses. After about 4,000 iterations $\Re^*$ levels out and the flow is not changing statistically.

To compare the behaviour of Osher's and Roe's scheme (whose results will

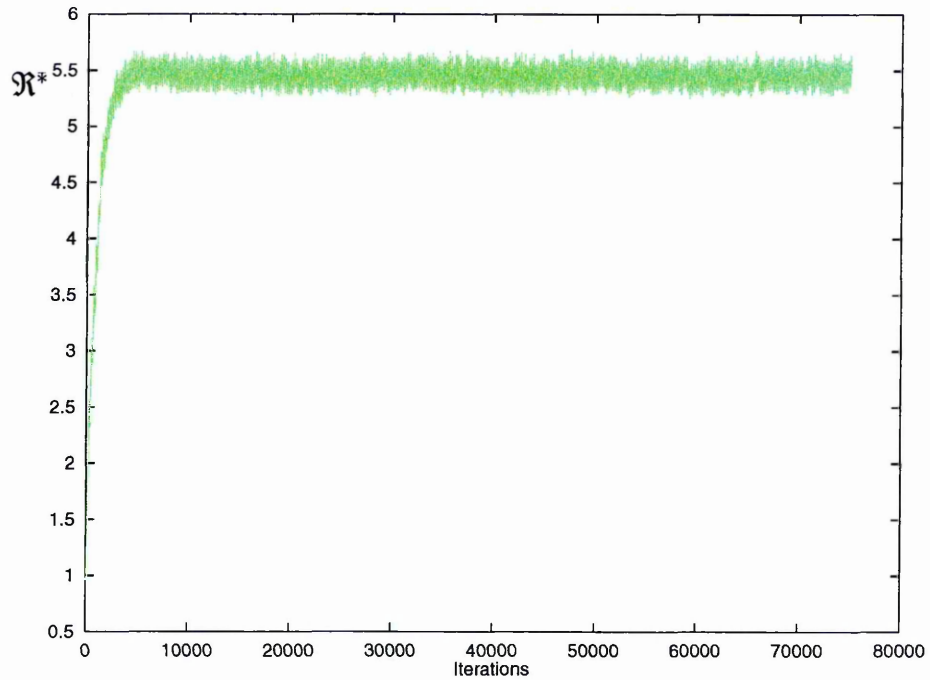

Figure 7.17: Plot of $\Re^*$ for a simulation when Osher's scheme was used.

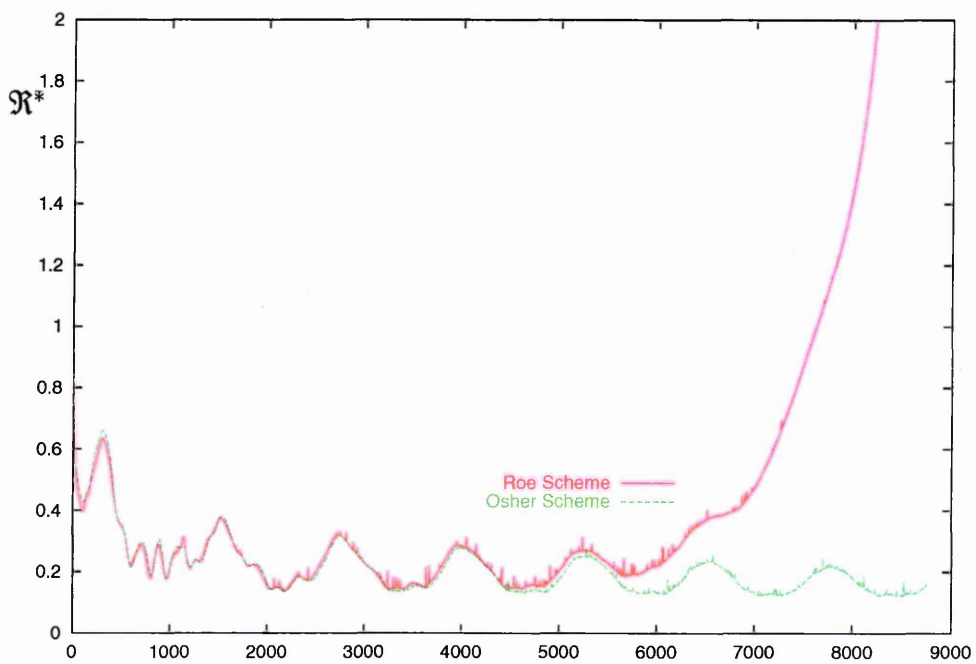

Figure 7.18: Plot of $\Re^*$ comparing the Osher's and Roe's scheme

be presented in the next section), a simulation using Roe's scheme was stopped just before turbulence starts to develop. The two calculations were restarted; one still with Roe's scheme and the other with Osher's scheme. Figure 7.18 shows plots of $\Re^*$ for the two methods. As one can see, the two calculations follow a similar pattern of development until 5,500 iterations. After that, the calculation with Roe's scheme started to develop turbulence while Osher's scheme failed to do so.

7.4.2 Visualization

Figure 7.19 shows density plots viewed from (a) the front, (b) the side and (c) the top. The only irregularity in the plots is the sinusoidal disturbance from the inlet which gradually disappears downstream of the computational domain.

The plot of velocity component v (Figure 7.20) reveals some kind of chaotic movement in the solution towards the end of computational domain. The magnitude of v compared to the dominant u is small and any slight change in the value of v can be detected easily.

The plot of ω_x can be used to displace minute changes and indeed Figure 7.21 reveals that there are unsteady movements in the simulated flow field. Note that the value of $\omega_x = -0.0627108$ for the plotted iso-surface is low compared to the values used in iso-surface plots in other sections. Therefore the plot may give the impression of more unsteadiness than there actually is. Also, a more detailed observation will reveal that the initial disturbances are gradually smoothed out and the flow becomes more stable.

7.4.3 Analysis of Results

Figure 7.22 shows a plot of skin friction coefficient along the plate. It shows that the disturbances at the inlet gradually die down and fail to develop any turbulence.

A plot of the normalized velocity profile (Figure 7.23) near the outflow boundary also confirms that the boundary layer is still laminar.

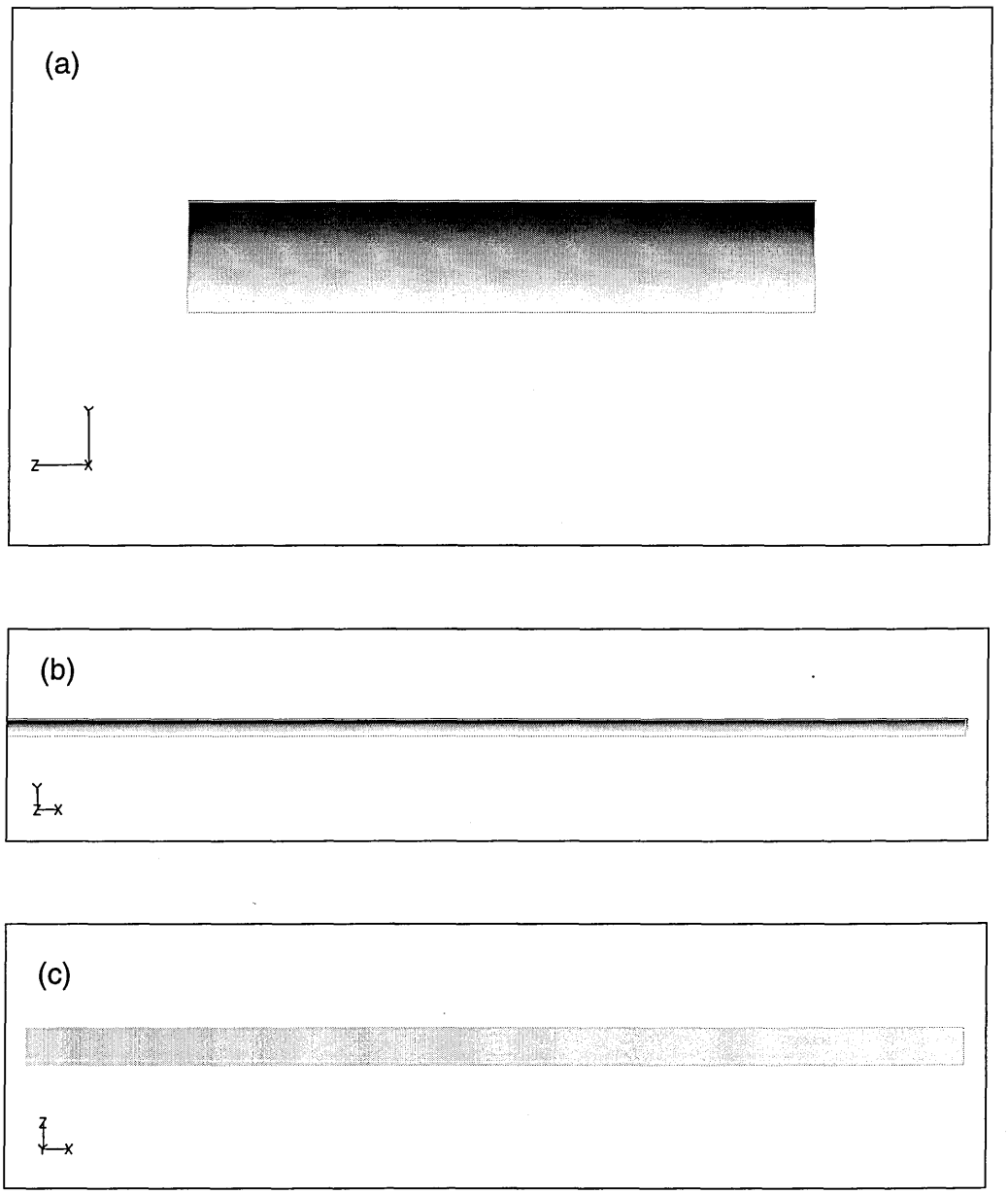

Figure 7.19: Density plots viewed from (a) front, (b) side and (c) top.

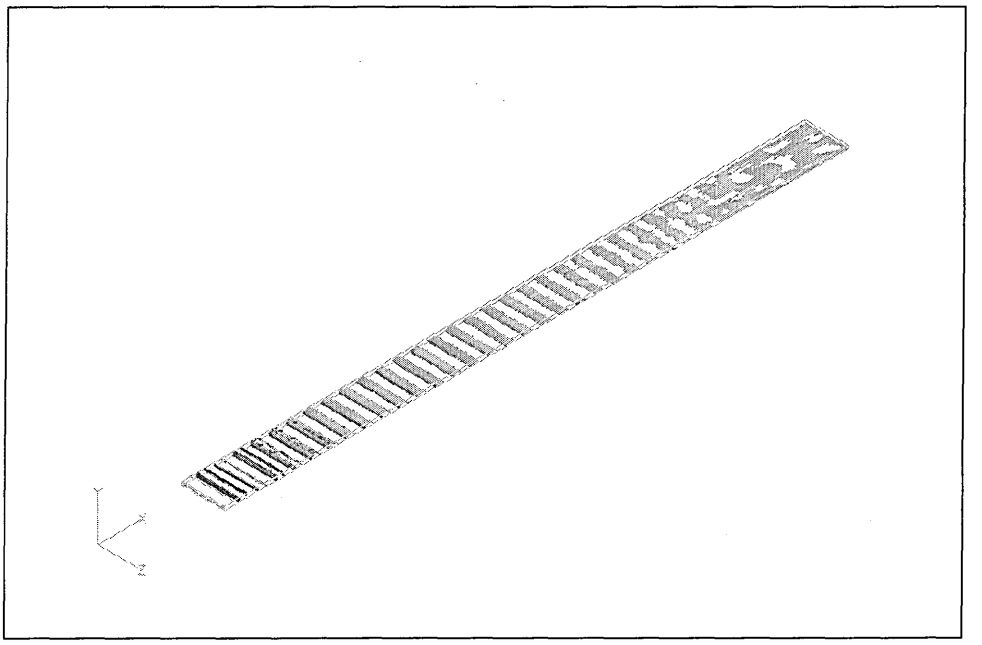

Figure 7.20: Iso-surface plot of velocity component $v = 0.0$ when Osher's scheme was used.

An interesting observation from the normalized mean velocity profile is that at $y^+ = 8$, where the velocity profile of turbulent boundary layer diverges from the laminar one, there is some movement which dies down after $y^+ = 10$. One can speculate that this is the instability which causes the boundary layer transition from laminar to turbulent state.

7.5 Roe's Scheme with a Limiter

In all computations with upwind-biased schemes, undershoots and overshoots in the shock region are expected. A limiter may be used to reduce the scheme to a fully one-sided scheme of either first- or second-oder accuracy in the shock region thus eliminating over- or under-shoots.

The flow over a flat plate has a leading edge shock wave due to boundary layer effects. This shock wave needs to be captured in the 2-D calculation but in 3-D calculation there is no shock wave. Ideally, limiters should only be turned on in

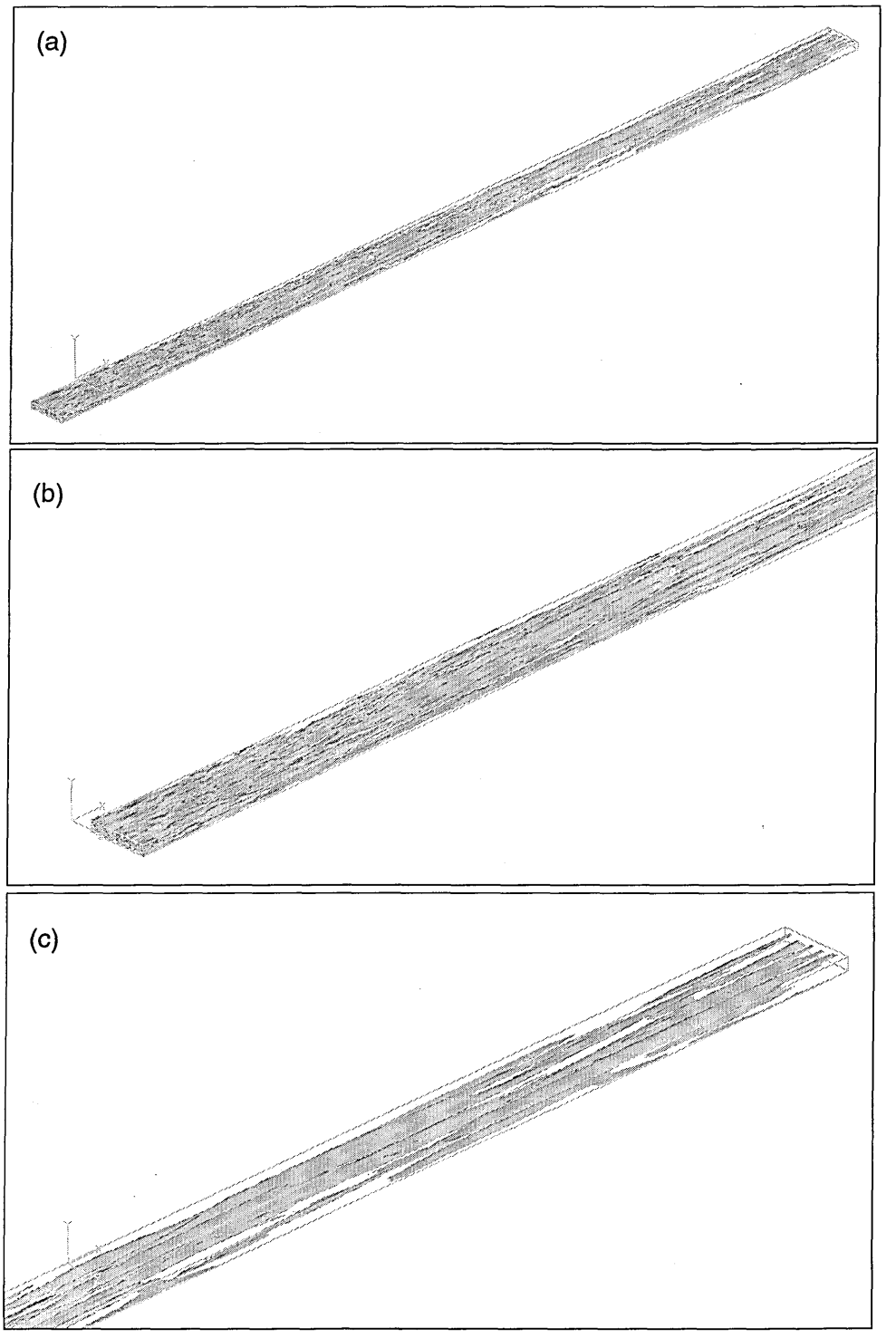

Figure 7.21: Iso-surface plots of $\omega_x = -0.0627108$ when Osher's scheme was used.

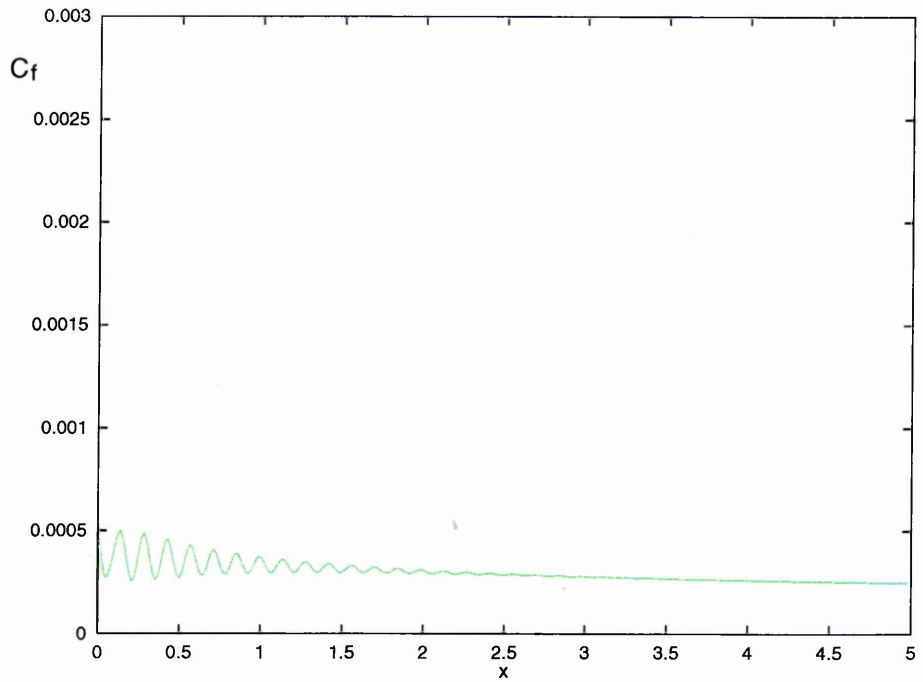

Figure 7.22: Plot of skin friction coefficient when Osher's scheme was used.

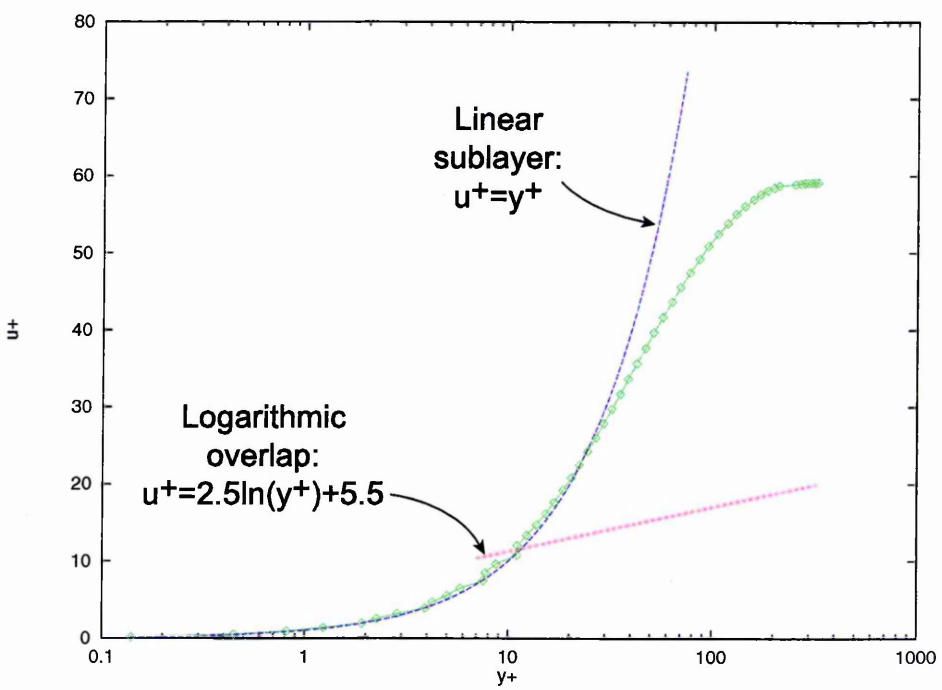

Figure 7.23: Plot of mean velocity Profile when Osher's scheme was used.

the shock region and they should not affect the rest of the flowfield. Therefore if there is no shock wave the limiter should not have any effect on the solution. Unfortunately, as will be shown, both the Minmod & Smooth limiters affect the solution and the development of turbulence.

7.5.1 Minmod Limiter

Figure 7.24 show the plot of density approximately at the middle of boundary layer. There is only little suggestion of unsteadiness near the outlet. Comparison of density plots with the simulation without the limiter (Figure 7.46) immediately shows that the flow with the limiter is much smoother. Still there is a visual sign of unsteadiness towards the outlet of the domain. This unsteadiness can also be seen in the iso-surface plot of $\omega_x = -54.2833$ (Figure 7.25).

Figure 7.26 shows the plot of mean skin friction coefficient along the plate. The skin friction coefficient rises above the laminar function but compared to the skin friction coefficient of fully turbulent boundary layer (Figure 7.46) it is much lower.

A plot of velocity profile (Figure 7.27) in the most turbulent region shows that the boundary layer lies somewhere between the laminar and turbulent state.

7.5.2 Smooth Limiter

Several ε values were tried between 10^{-5} and 10^{-7} which are commonly used values for this method (see Section 4.2.4). The results with different ε were more or less the same, therefore only the result with $\varepsilon = 10^{-7}$ is presented here.

Plots of density (Figure 7.28) show a very smooth boundary layer with very little unsteadiness. Plots of $\omega_x = -63.0931$ show that there is some unsteadiness present towards the outlet boundary, but still, compared to the simulation without any limiter (Figure 7.46), the boundary layer is very smooth.

Figure 7.30 shows a plot of skin friction coefficient along the plate and there is

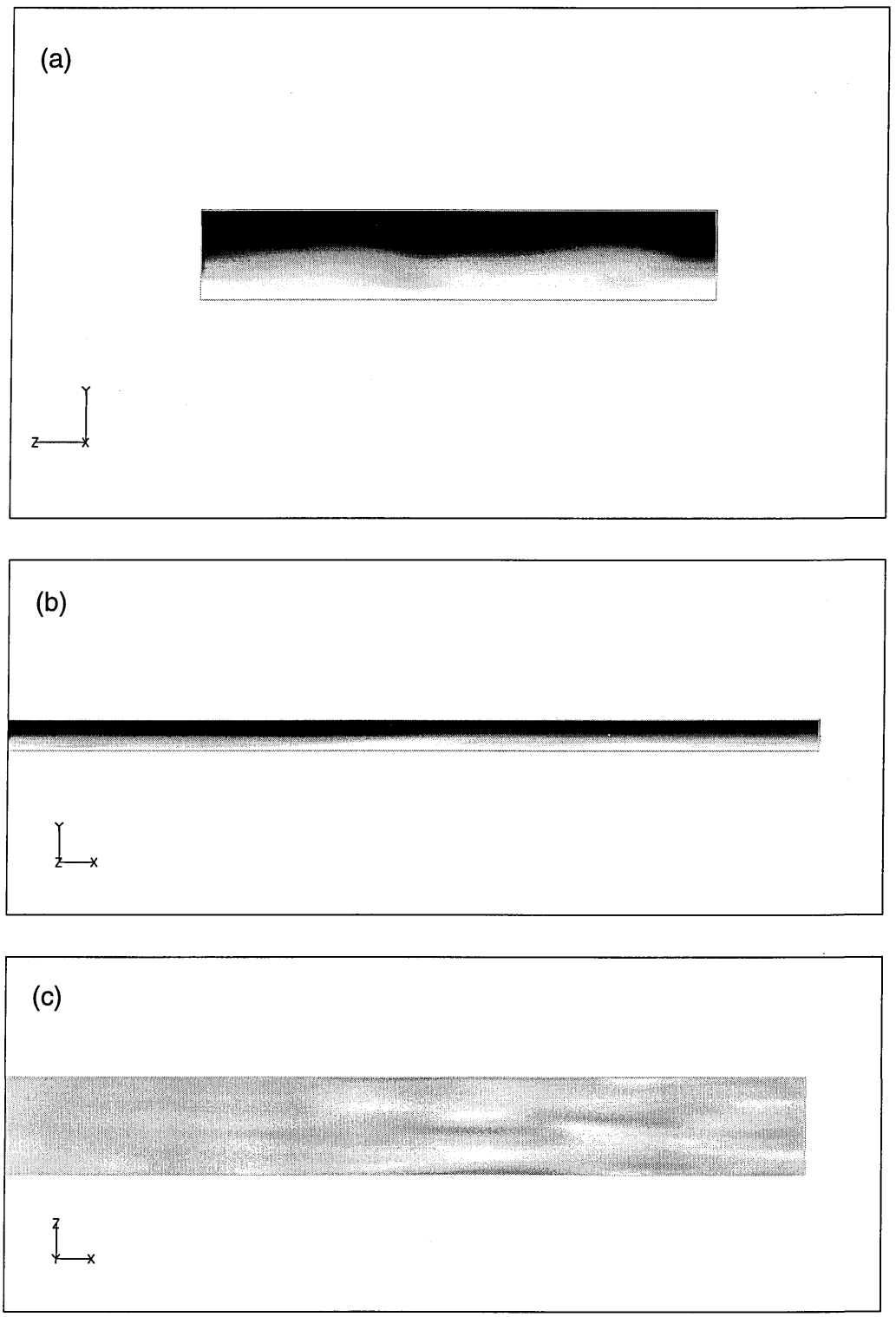

Figure 7.24: Density plots viewed from (a) front, (b) side and (c) top.

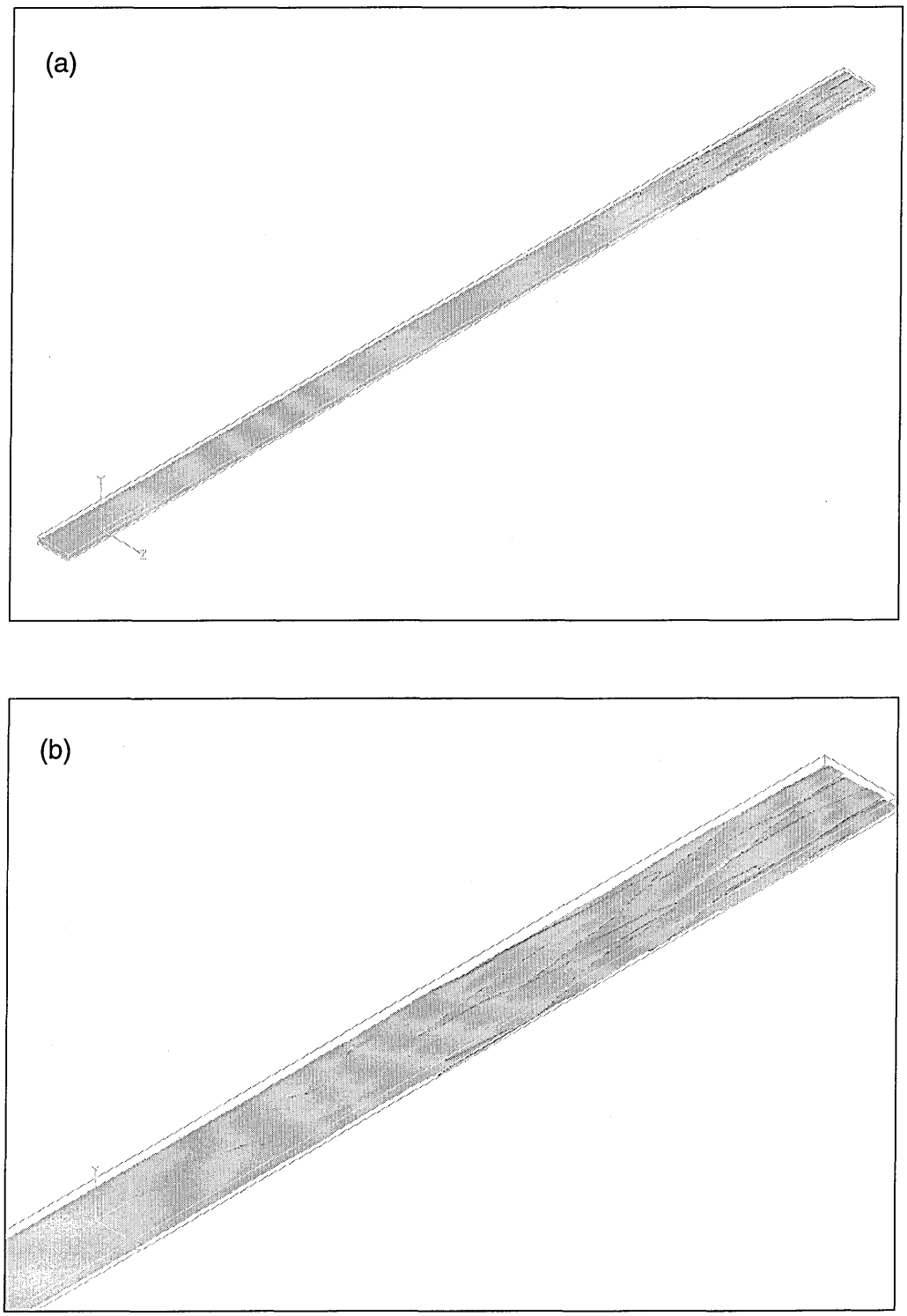

Figure 7.25: Iso-surface plot of $\omega_x = -54.2833$

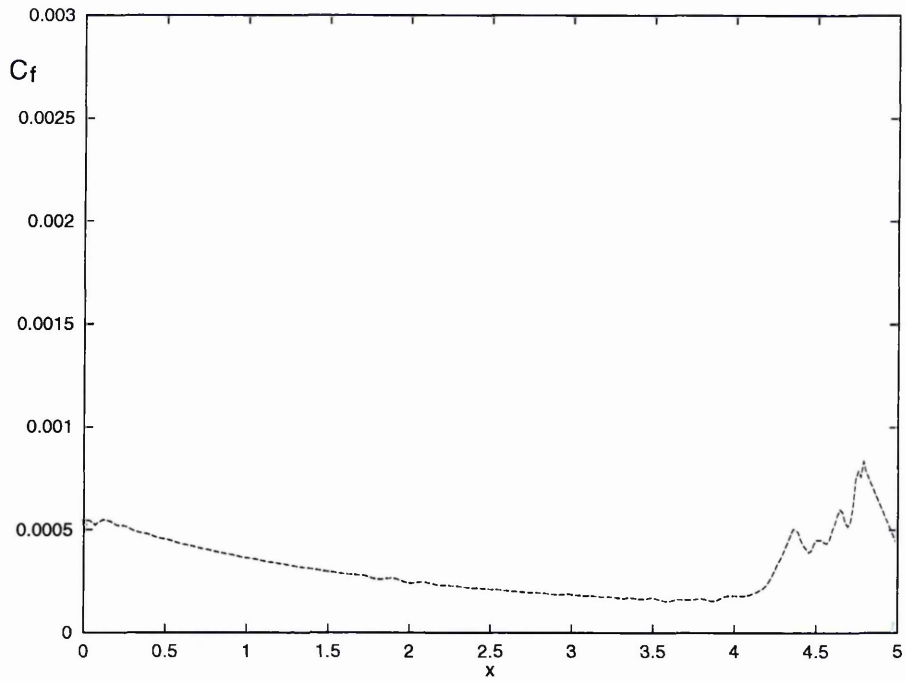

Figure 7.26: Plot of mean skin friction coefficient along the plate when the minmod limiter was used.

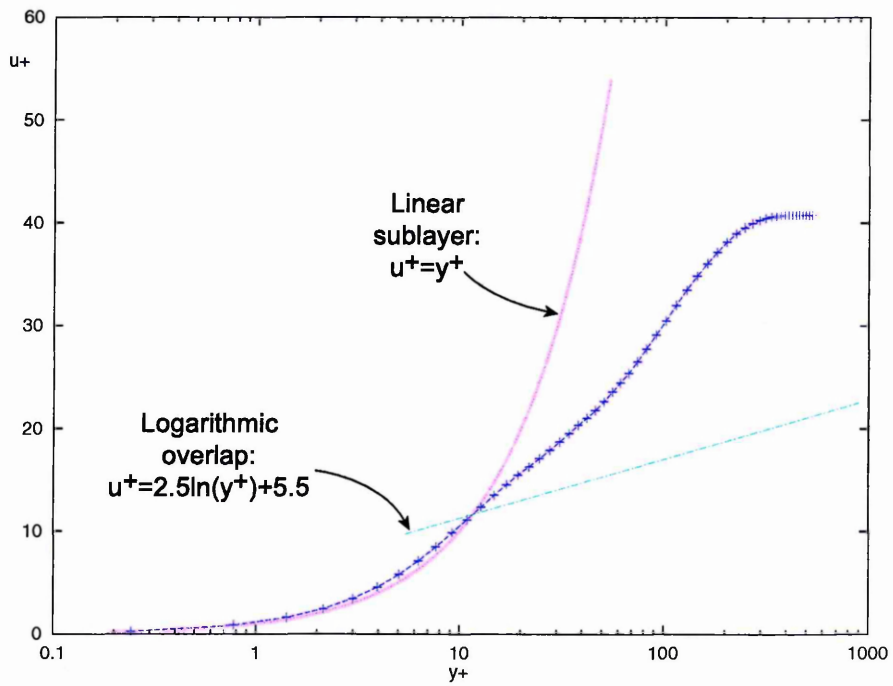

Figure 7.27: Plot of mean velocity profile at $x = 4.7$ when the minmod limiter was used.

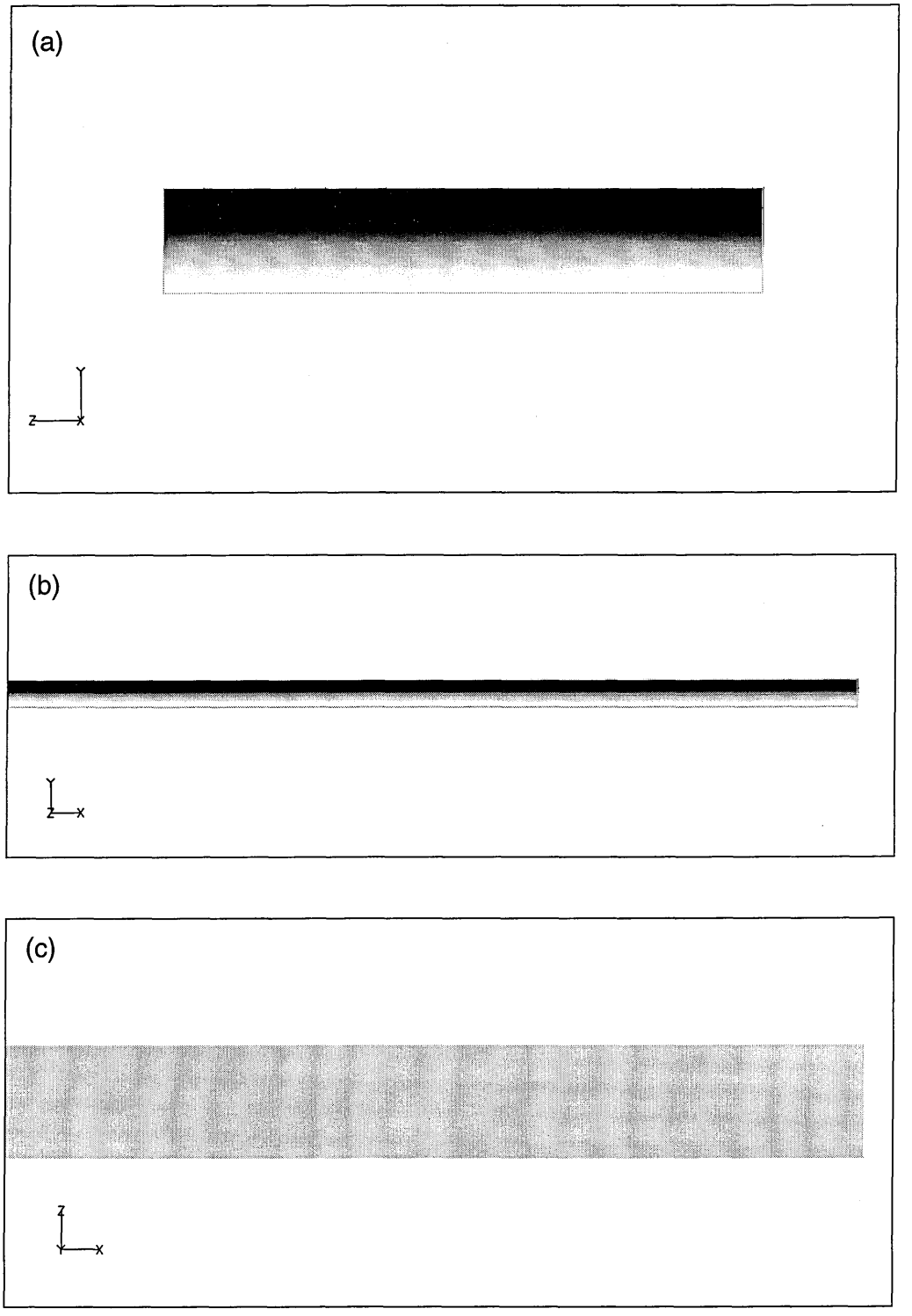

Figure 7.28: Density plots viewed from (a) front, (b) side and (c) top.

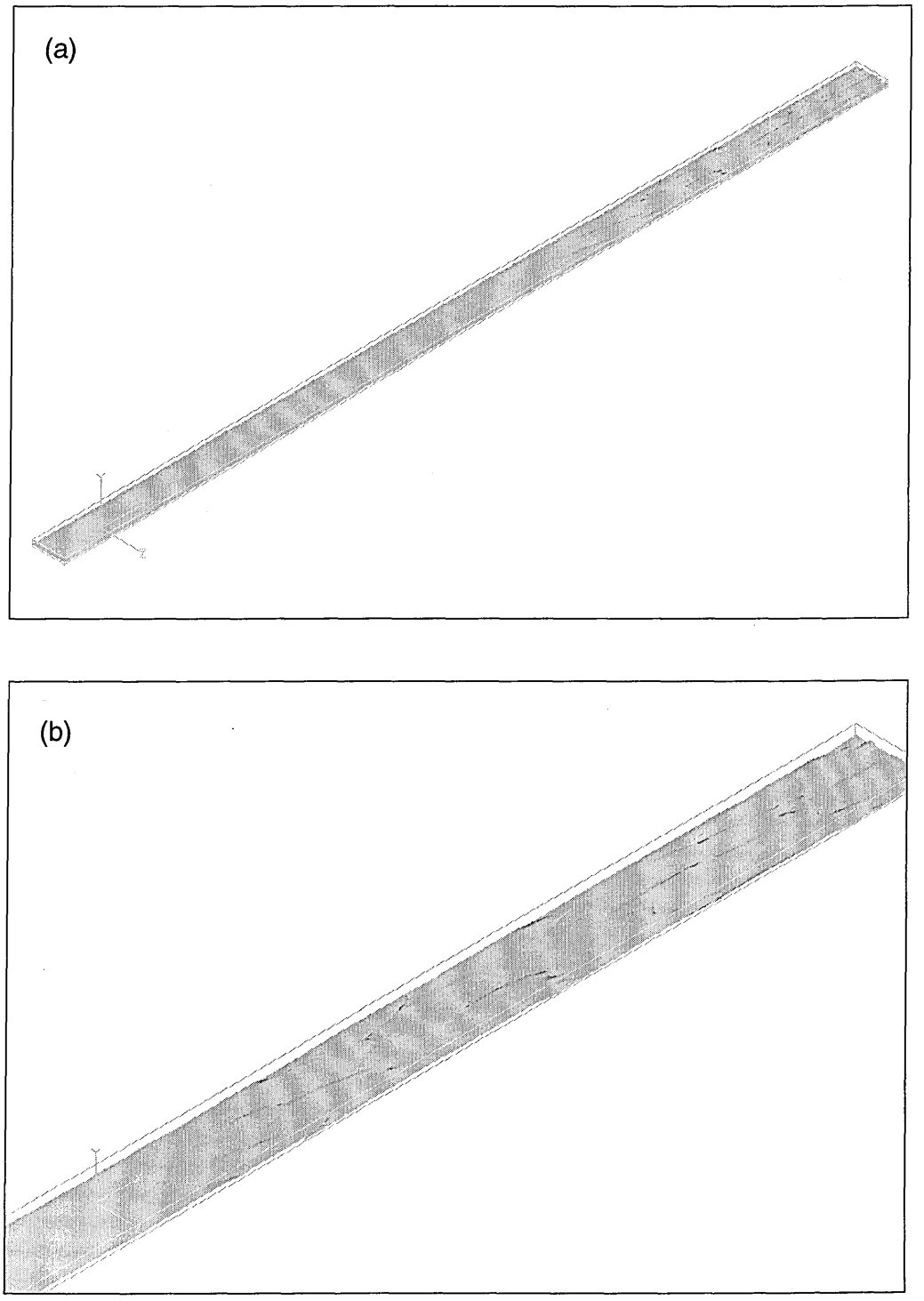

Figure 7.29: Iso-surface plot of $\omega_x = -63.0931$

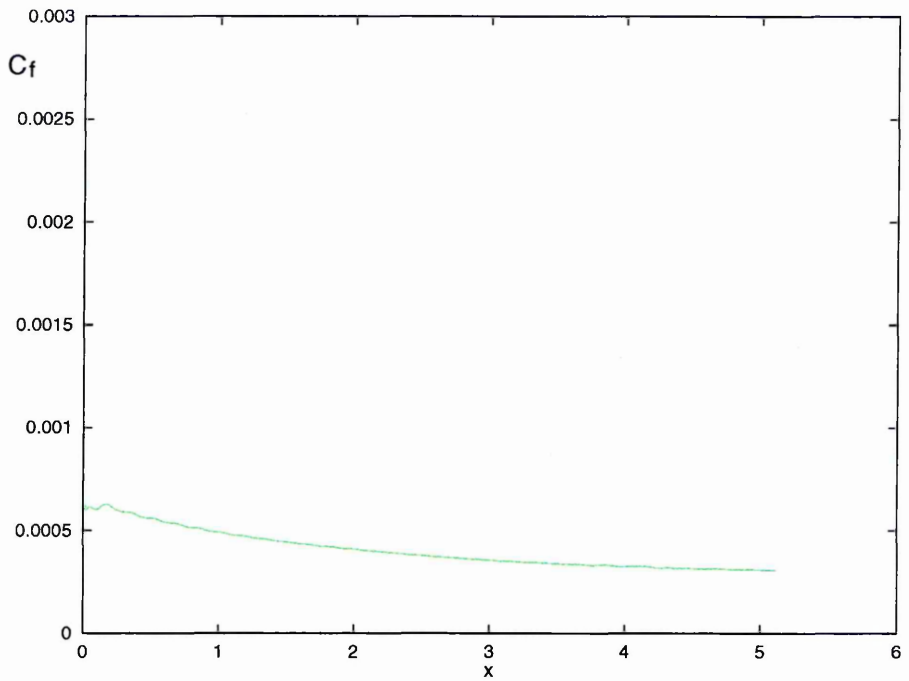

Figure 7.30: Plot of mean skin friction coefficient along the plate when the smooth limiter was used.

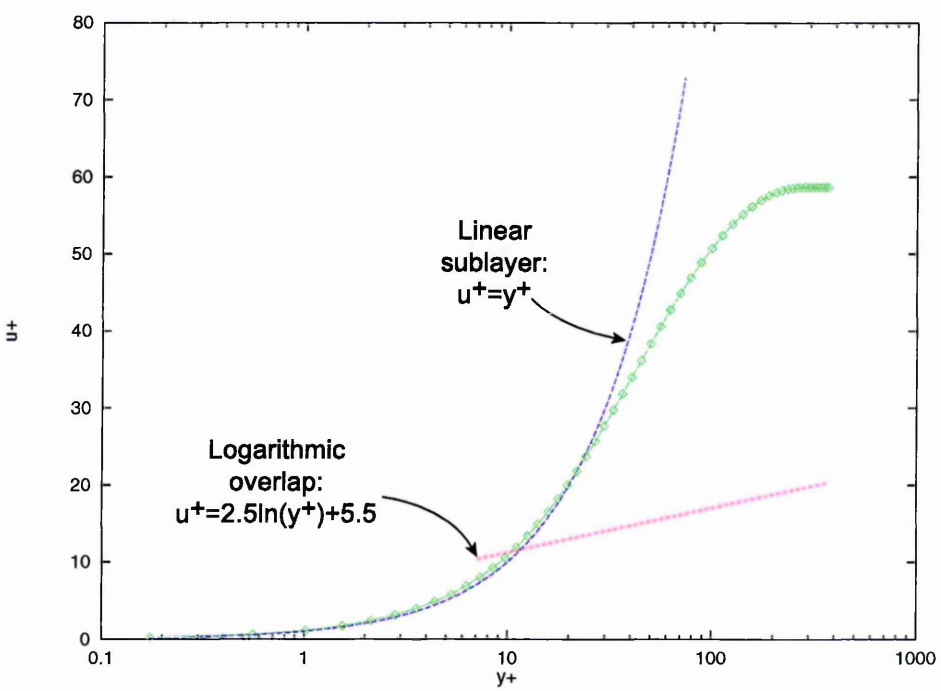

Figure 7.31: Plot of mean velocity profile at $x = 4.7$ when the smooth limiter was used.

no sign of a turbulent boundary layer at all. Also the velocity profile (Figure 7.31) near the outlet boundary shows that it is a laminar boundary layer. Therefore the smooth limiter with ε values between 10^{-5} and 10^{-7} is more dissipative than the Minmod limiter.

A larger value of ε will make the limiter less dissipative. Brief tests with ε values between 10^{-5} and 10^{-1} show that it is less dissipative and there were signs of a turbulent boundary layer developing. Figures 7.32–7.35 show the results with $\varepsilon = -3$.

Nevertheless, it was still too dissipative to be used in MILES. Further if ε values outside the conventional ones are to be used, more tests are needed to see whether the limiter is as effective when a shockwave is present. In theory, with a larger ε value the scheme will be less effective as a limiter.

7.6 Roe's Scheme without a Limiter

7.6.1 Effect of Grid Size

Several grid sizes and domain sizes have been tested to investigate the effect these have on the solution. These tests were important because the available computer resources were limited, and so determining the minimum number of grid points to resolve the turbulence with large enough domain to contain the large eddies was important. After the grid convergence study we concluded that Case 2 in Table 6.1 was the minimum grid resolution required for our test case. Computational requirements will be discussed further in Section 7.7.

During the grid convergence study we have found that the change in the number of grid points in k-direction effects the solution most. Significant change in turbulence production occurs with the change in number of grid points in k-direction. (The effect will be presented in this section.) The change in x-direction also has significant effect but it was less than the change in k-direction. The change in y-direction effected the solution least.

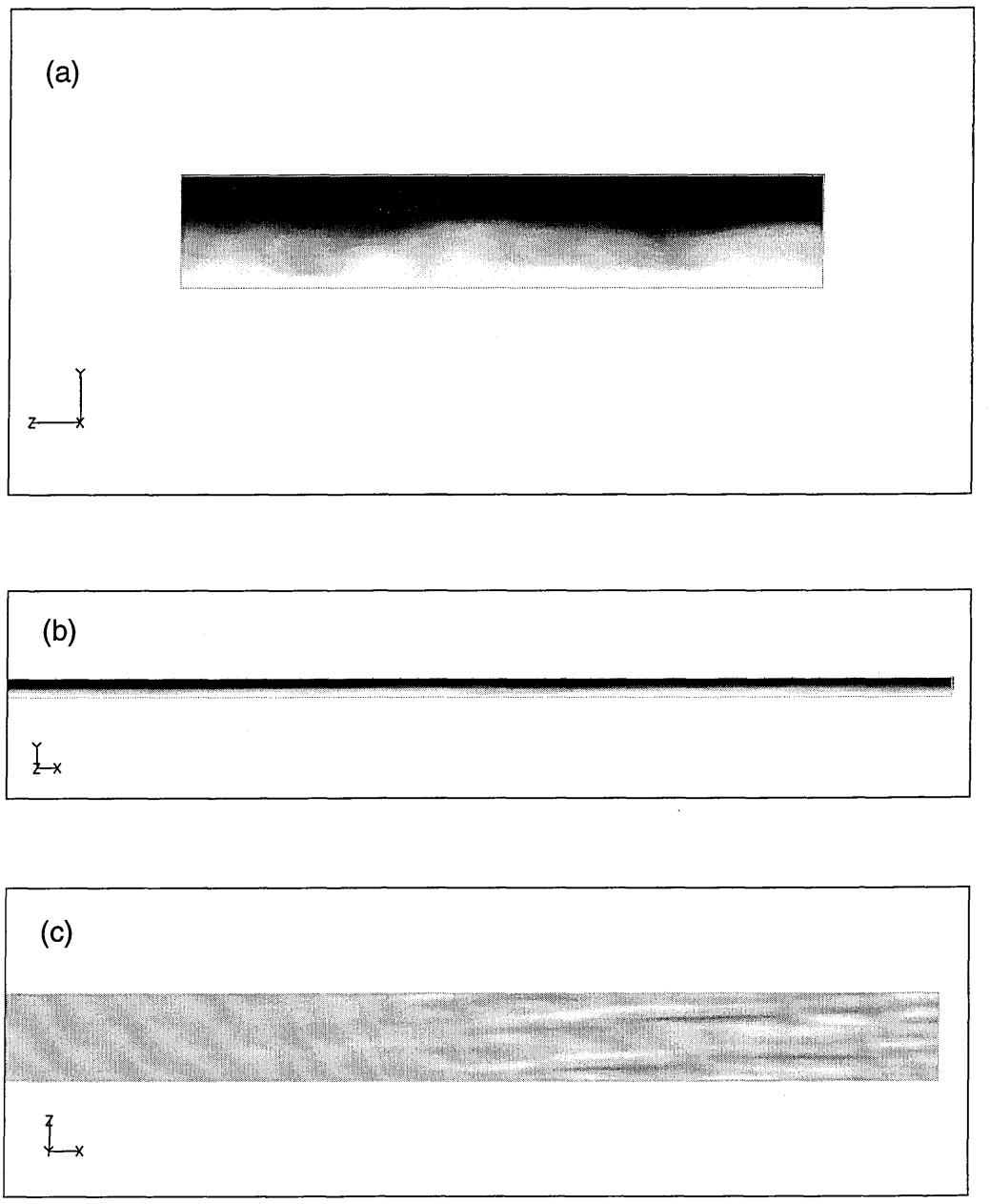

Figure 7.32: Density plots viewed from (a) front, (b) side and (c) top.

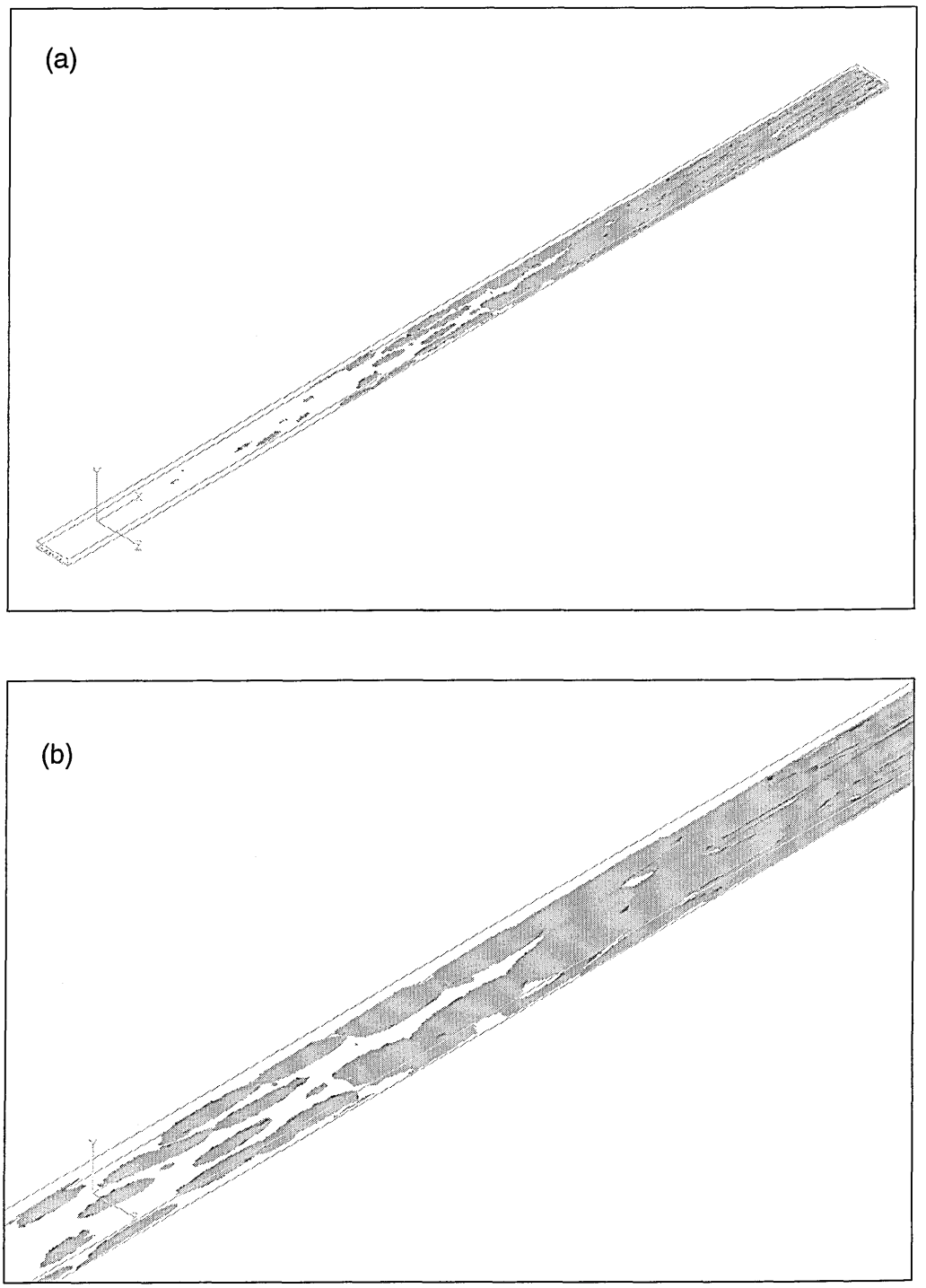

Figure 7.33: Iso-surface plot of $\omega_x = -1.76368$

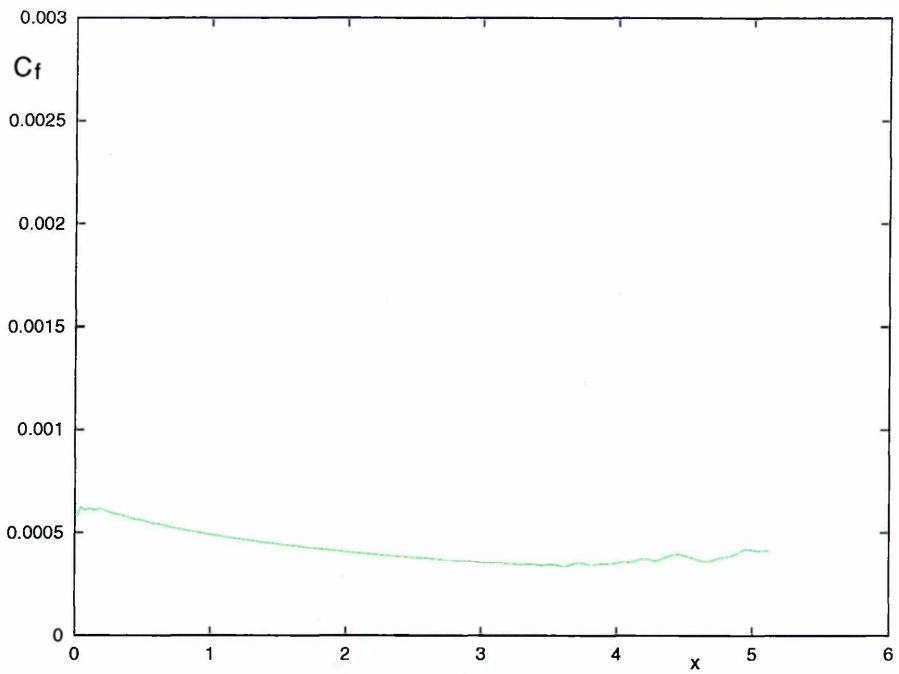

Figure 7.34: Plot of mean skin friction coefficient along the plate when the smooth limiter was used.

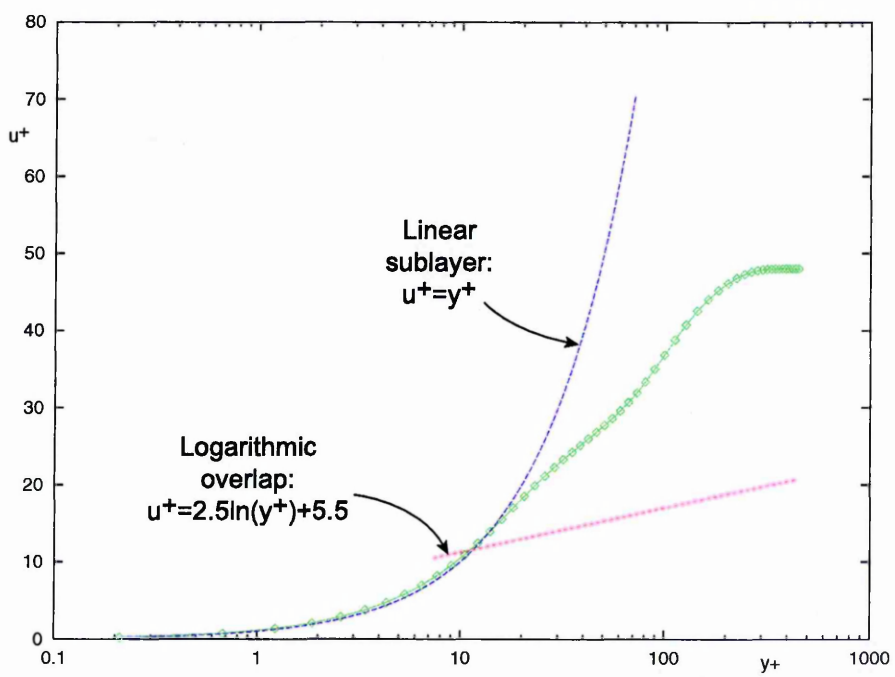

Figure 7.35: Plot of mean velocity profile at $x = 4.7$ when the smooth limiter was used.

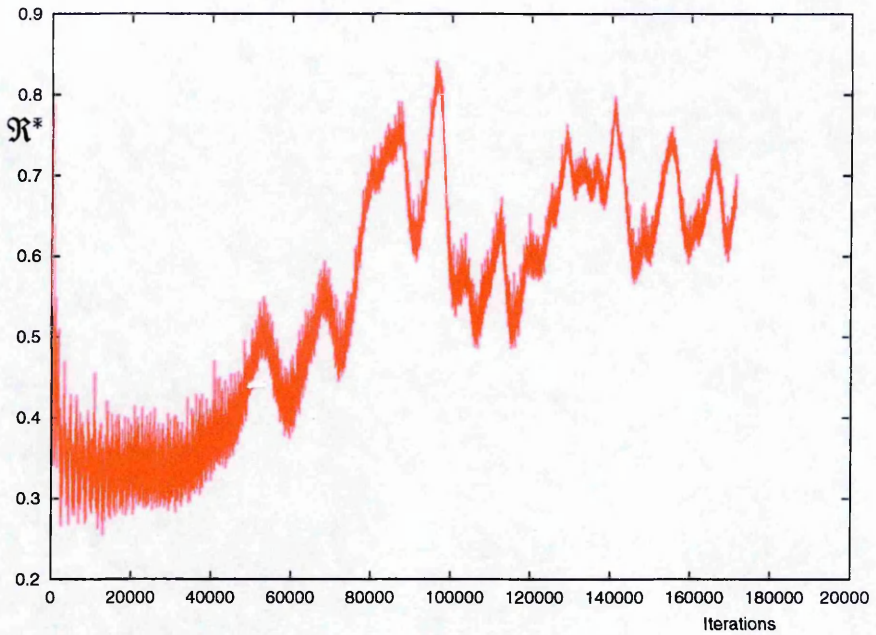

Figure 7.36: Plot of $\mathfrak{R}^*$ of a simulation when Roe's scheme without limiter was used.

The computational results with a coarser grid (Case 1 in Table 6.1) are presented in this section to show the effect of grid size. Note that everything is the same except the number of grid points in the k-direction have been halved to 30.

During the simulation, the sum of residuals $\mathfrak{R}^*$ (Figure 7.36) increases much more than in the simulation with the finer grid (Figure 7.41). This suggests the simulated flow does not have much unsteadiness. All the other plots (Figures 7.37–7.40) confirm this.

Rest of the results in this section are of Case 1.

7.6.2 Monitoring Calculation

Figure 7.41 is a plot of the sum of residual $\mathfrak{R}^*$ for case 2 in Table 6.1. The initial boundary layer generated with a 2-D calculation is completely laminar throughout the domain. Even though disturbances at the inlet are started from the beginning of the calculation nothing much happens until after 40,000 iterations.

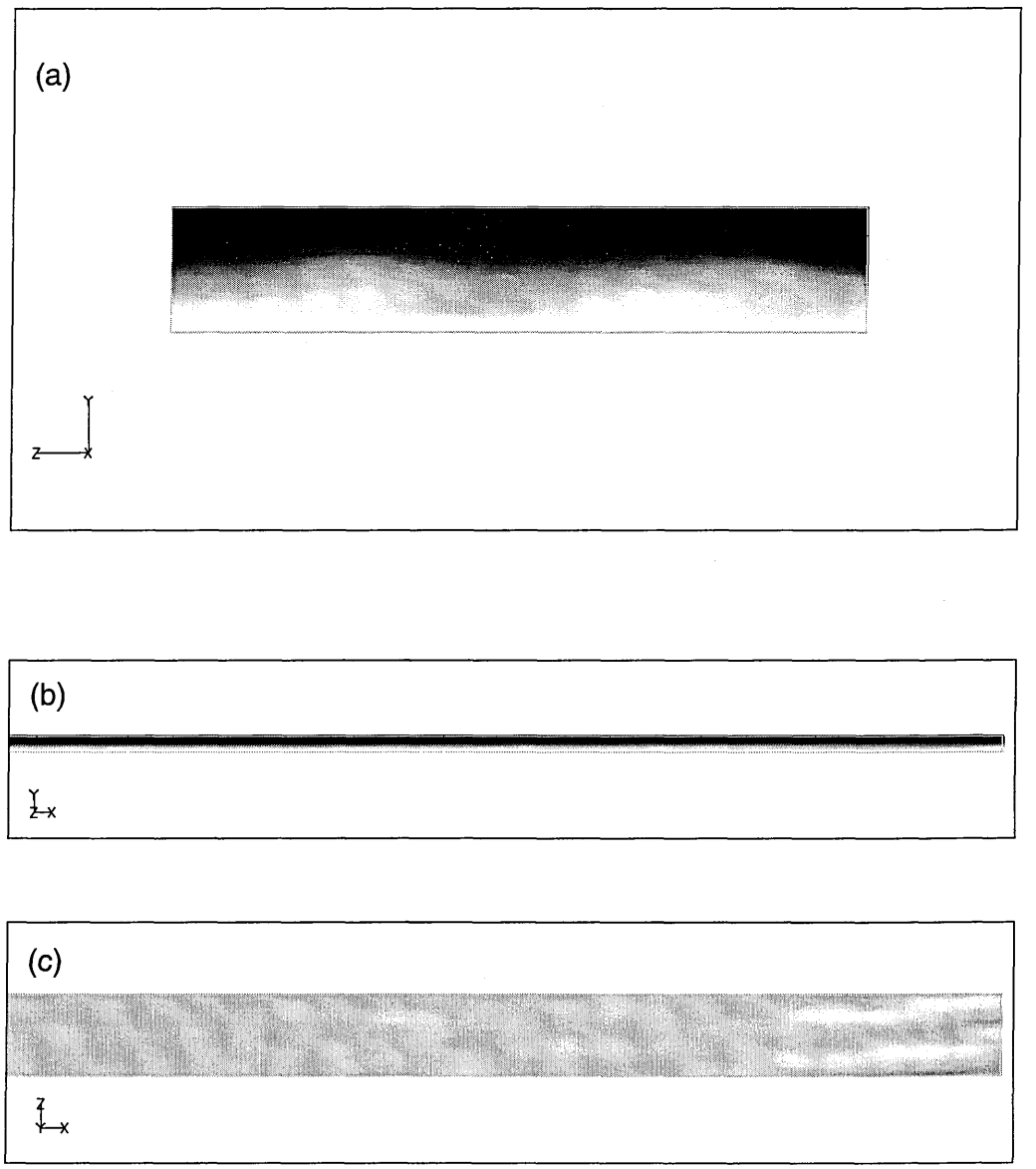

Figure 7.37: Density plots viewed from (a) front, (b) side and (c) top.

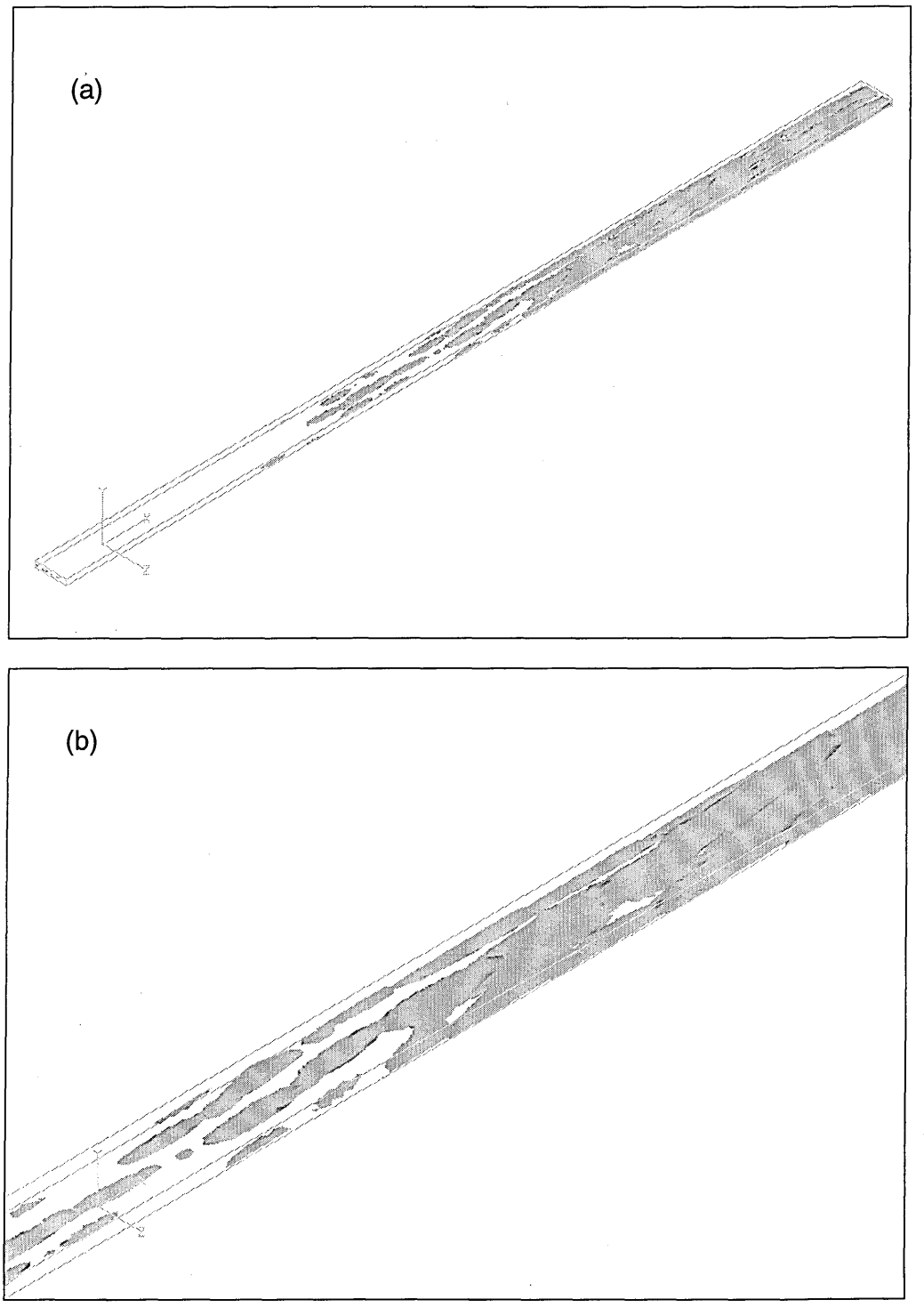

Figure 7.38: Iso-surface plot of $\omega_x = -1.58634$

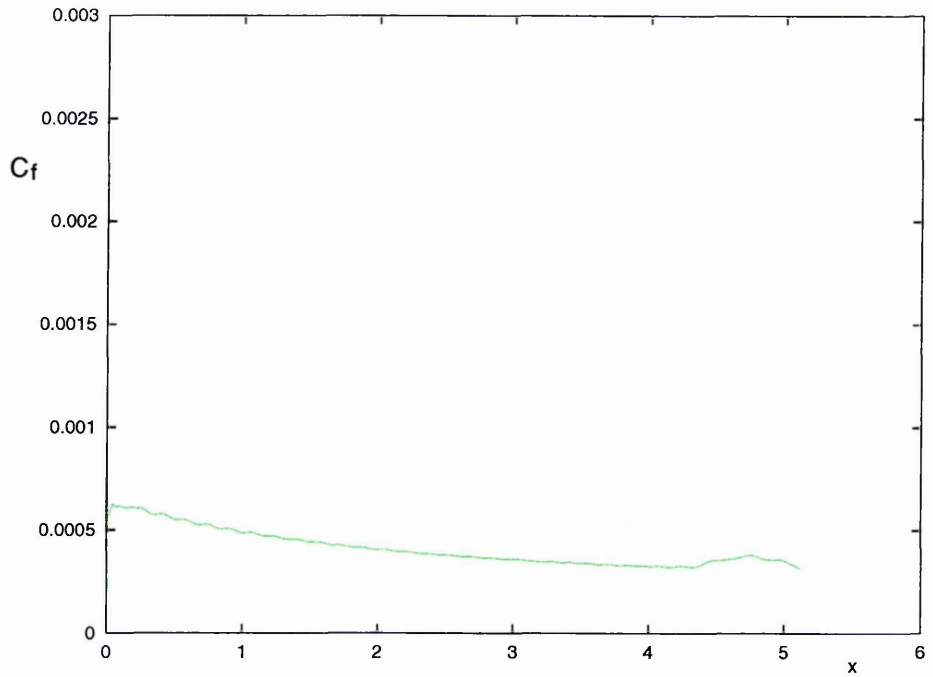

Figure 7.39: Plot of mean skin friction coefficient along the plate (case1 in Table 6.1)

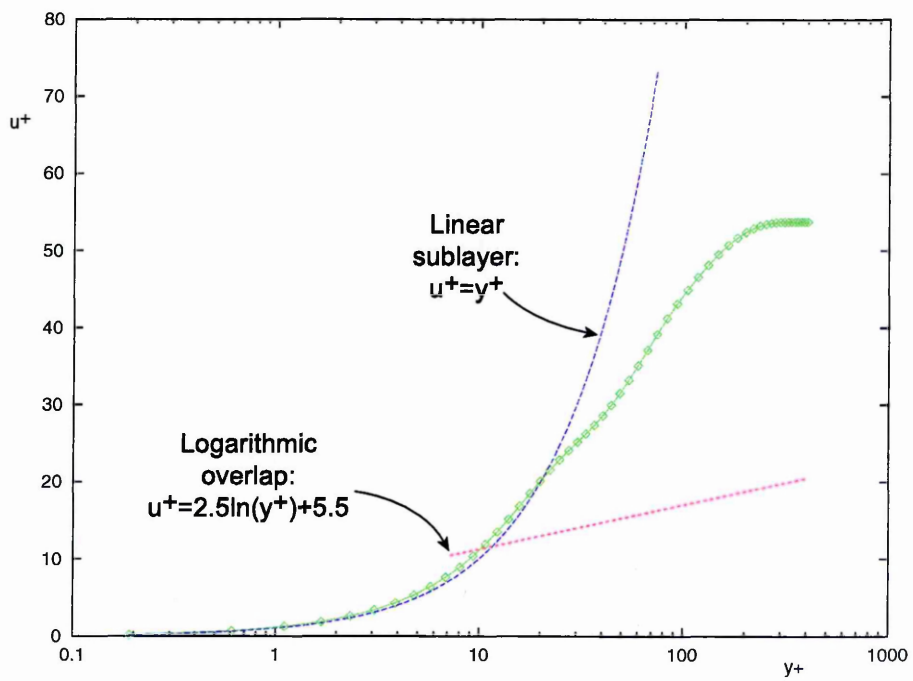

Figure 7.40: Plot of mean velocity profile at $x = 4.7$

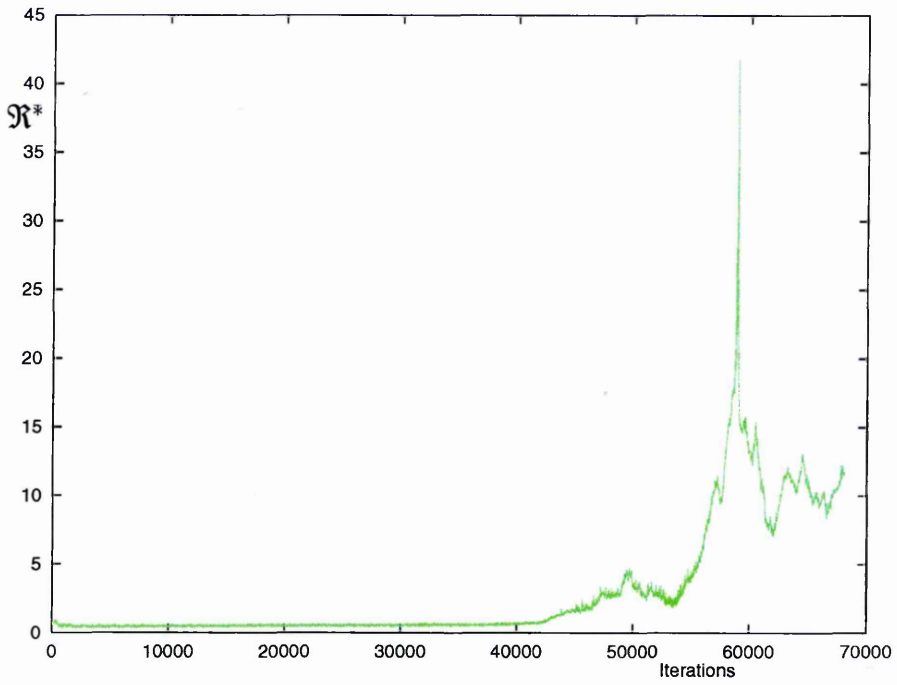

Figure 7.41: Plot of $\mathfrak{R}^*$ of a simulation when Roe's scheme was used.

As turbulence starts to develop in the domain, $\mathfrak{R}^*$ increases. Eventually $\mathfrak{R}^*$ rises in a very rapid fashion at iteration 58,000 to reach a peak. This is when the flow is changing most vigorously. $\mathfrak{R}^*$ decreases again, but to a level still much higher than 1.

$\mathfrak{R}^*$ rises and falls as more iterations are calculated, but it never reaches the height of the first peak and does not drop more than the first fall after the major peak. If the mean value of the oscillating $\mathfrak{R}^*$ does not change very much then the simulated flow is considered to be settled downed statistically.

To double check that it is statistically stable the averaged skin friction and velocity profile along the plate were observed at intervals. Because the skin friction coefficient and velocity profile were seen to be more or less stabilized after 65,000 iterations and $\mathfrak{R}^*$ has maintained a certain level, the simulated flow is judged to be statistically stable.

7.6.3 Visualization

Density plots from 3 view points are shown in Figure 7.42. It is clear from the plots that the boundary layer has turned turbulent. Turbulent eddies of various sizes are captured remarkably well, considering the number of grid points in y- and z-directions are only 55 and 60 respectively.

Figure 7.42 (a) also shows that the spanwise domain size for the calculation is large enough to capture several vortices. Therefore, the cyclic boundary conditions used in the spanwise direction should not effect the calculation at all. Figure 7.42 (b) shows the turnover of large eddies occurs several times in the x-direction before the eddies exit the outflow boundary. Therefore the domain is large enough to capture a fully turbulent boundary layer. This will be also confirmed from plots of the skin friction coefficient and the normalized velocity profiles (see next sections). Both Figure 7.42 (a) and (b) show the turbulent boundary layer near by touches the upper boundary layer. It seems that the characteristic boundary condition on the upper boundary (see Section 6.4) is good enough to mimic the freestream boundary conditions, and there is no evidence of buildup of turbulence on the top boundary. The plot of boundary layer thickness also confirms that the computational domain is just about high enough to contain the boundary layer (see Section 7.6.4). Therefore extra domain height would not add any value to the calculation.

Figure 7.43 shows the density plots along the plate together with density contours on the iso-surface of vorticity in x-axis. It is true that the density contours on different iso-surfaces would look different but this plot seems to illustrate the development of turbulence along the plate quite well. Initially large sized eddies develop as the boundary layer becomes unstable. These big eddies create smaller eddies and these smaller eddies create even smaller eddies and so on until the smallest eddies are diffused into mean flow due to viscosity. This is called an energy cascade and this process can be seen in Figure 7.43.

Figure 7.44 shows the iso-surface plots of $\omega_x = -0.381494$ viewed from the

top. Figure (a) show the whole domain and (b)–(d) show the enlargement of various sections. They shows typical phenomena of laminar to turbulent transition which can be summarized as following.

Until $x \approx 2.2$in the boundary layer is fully laminar and the change in ω_x is very minimal. Even the initial disturbance is not large enough to be shown in these plots. Between $x \approx 2.2$in and 4.0in, oblique waves emerge in a staggered pattern, followed by a streamwise instability giving rise to streaks. From $x \approx 4.0$in onwards, the streaks rapidly become turbulent. Near the outflow boundary there seems to be a vortex stretching along the streamwise direction, but in fact, it is due to grid stretching in the buffer region (see Section 6.2)

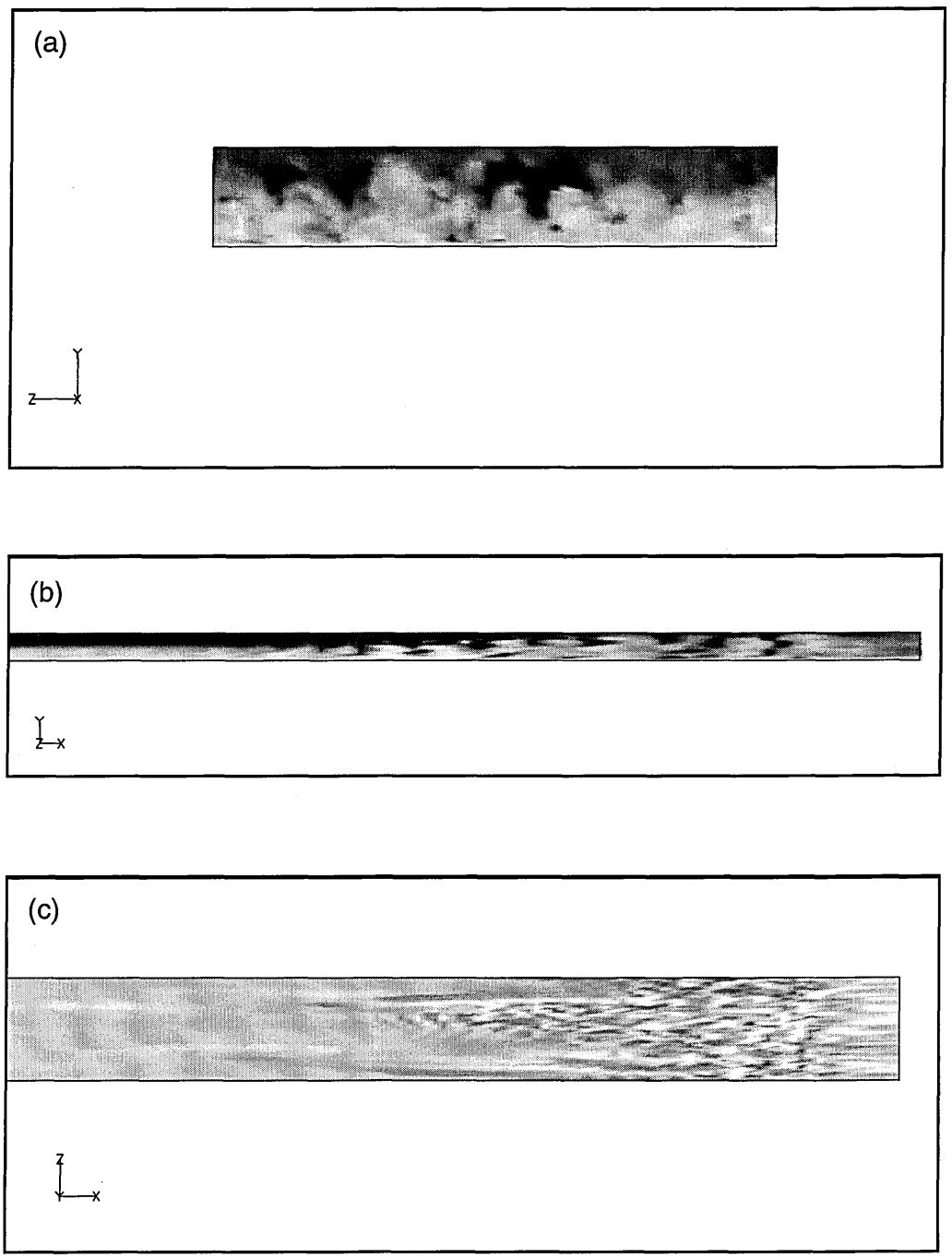

Figure 7.42: Density plots viewed from (a) front, (b) side and (c) top.

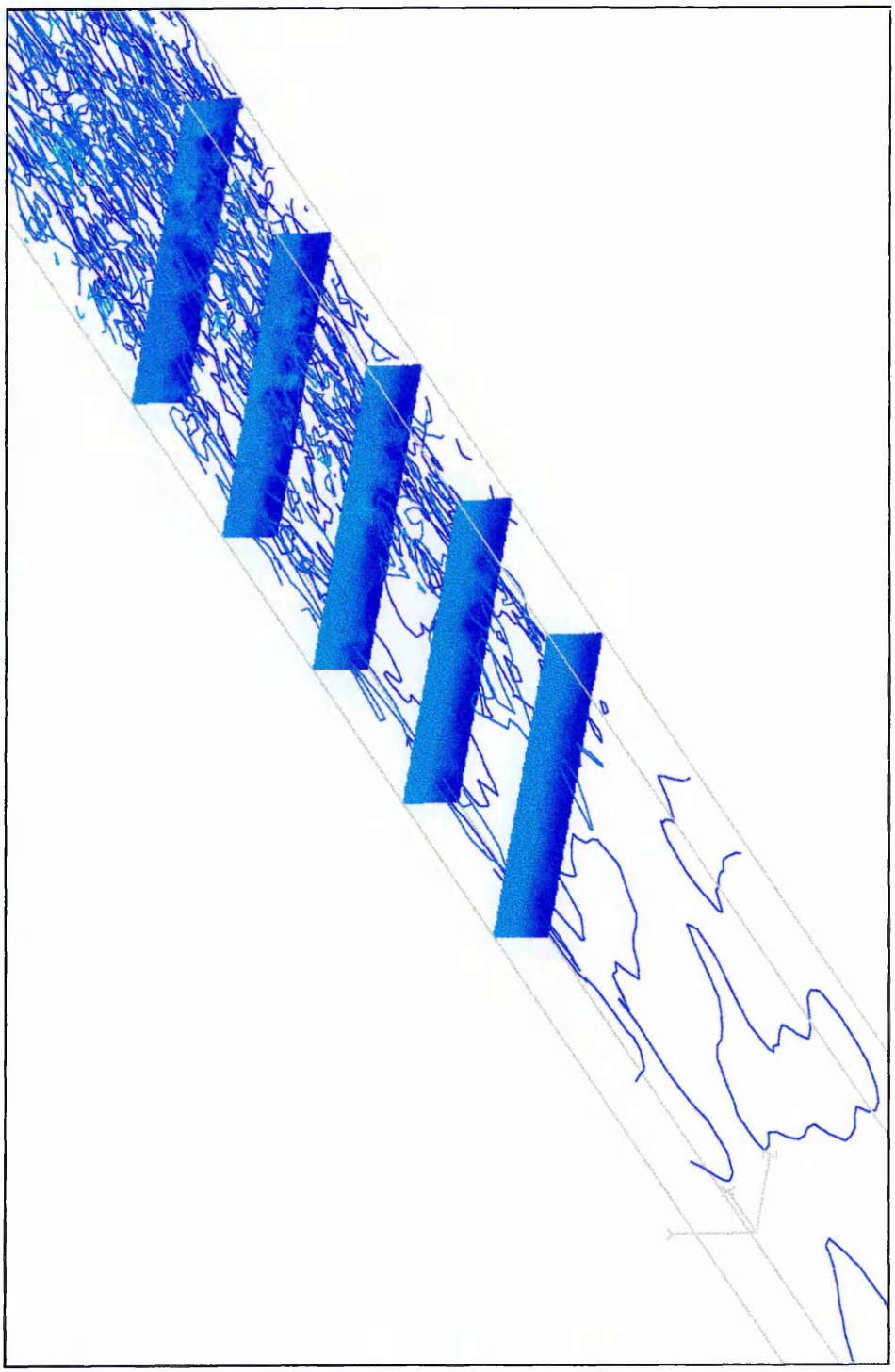

Figure 7.43: Density plots along the plate together with density contours on the iso-surface of vorticity in x-axis.

Iso-surface plots in Figure 7.45 also show $\omega_x = -0.381494$ but in 3 dimensions. Formations of streaks and transition to turbulent can be seen much more realistically.

7.6.4 Analysis of Results

Skin Friction coefficient

Figure 7.46 shows a plots of the skin friction coefficient along the plate. The current result has been plotted together with the Blasius laminar skin friction coefficient, the general trend of experimental measurements, results from DNS by Rai *et al.* [28] and results from LES by Spyropoulos and Blaisdell [30]. The DNS and LES used the blowing and sucking on the plate to trigger the transition. Therefore the location of transition in these and the current simulation are different. For easier comparison, their results are moved along the plate so that the transition starts at the same point.

Also, the current result looks noisier than those obtained with DNS and LES. This is because the current results are only spatially averaged whilst other results has been ensemble (time and spatially) averaged. The number of grid points in the z-direction for the current calculation is much lower than DNS and some of the LES.

Rai *et al.* used a fifth-order accurate, upwind-biased finite difference method to solve the inviscid term and a fourth-order accurate central difference for the viscous terms. Time advancement was performed using an iterative fully implicit second-order accurate scheme. Such schemes are unconditionally stable and allow for accurate advancement using much larger time steps than explicit schemes. However, they are more CPU intensive because they involve the solution of a system of algebraic equations. Their computational domain extends from $x = 5.88$ inches to 20.88 inches with only the first 9.0 inches being well resolved, while the last 6.0 inches are discretized using a grid that gradually becomes very coarse to attenuate the disturbances. The grid resolution in the x direction is 50 points/inch. The first point away from the wall is at y=0.0005 inches. The total number of grid

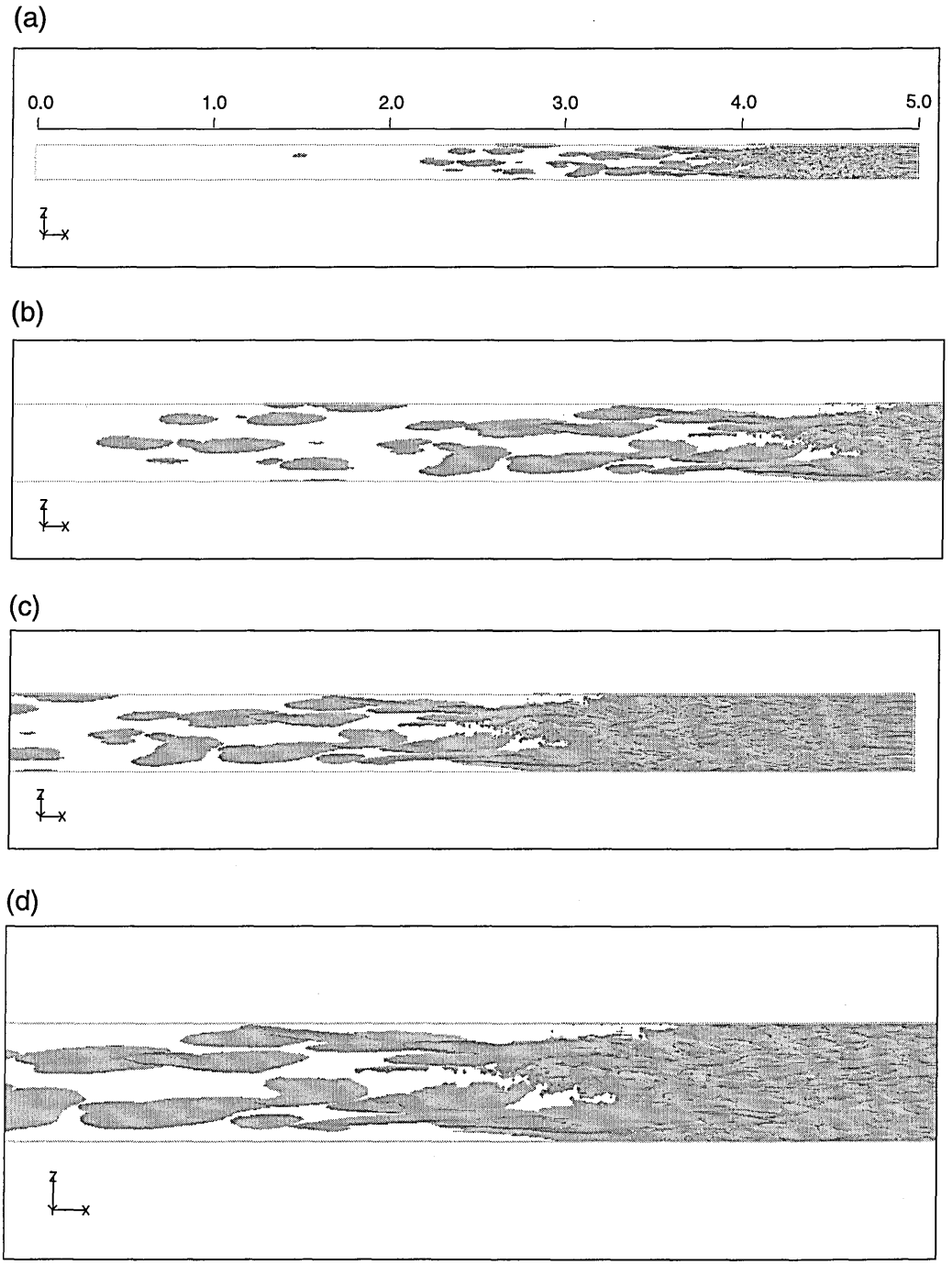

Figure 7.44: Iso-surface plot of $\omega_x = -0.381494$

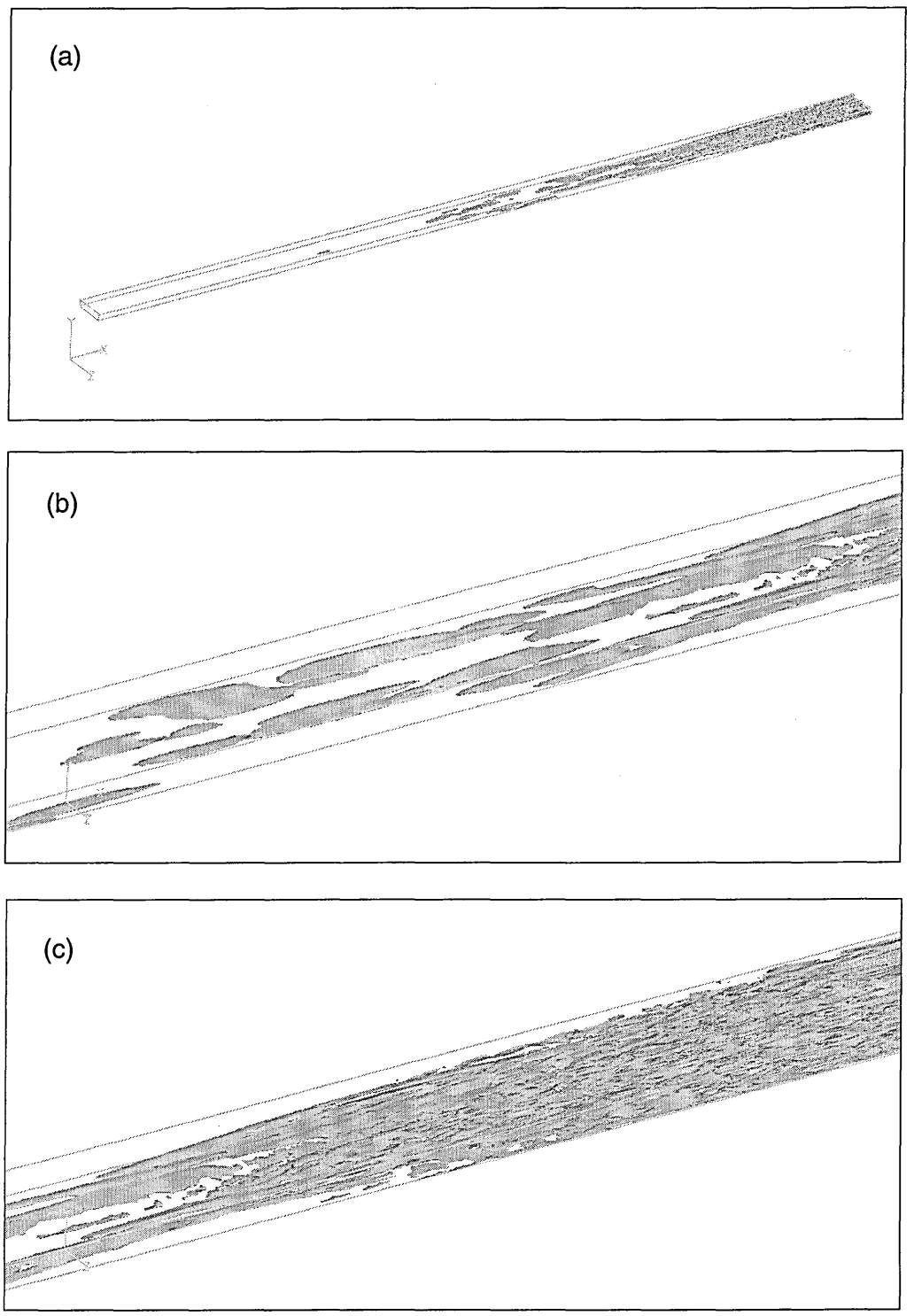

Figure 7.45: Iso-surface plot of $\omega_x = -0.381494$

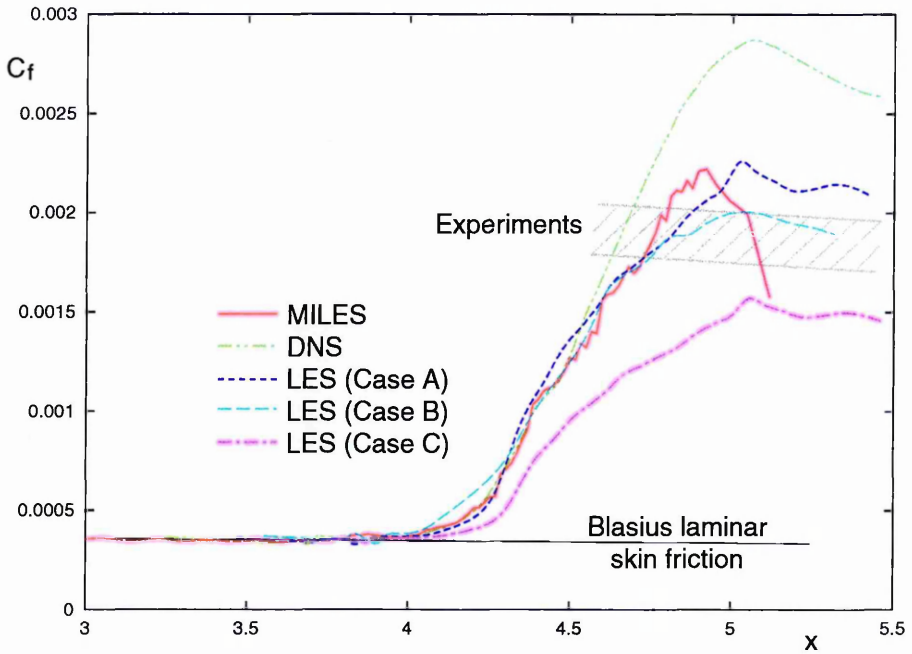

Figure 7.46: Plot of the mean skin friction coefficient along the plate when Roe's scheme was used, compared to the theoretical and experimental results

points used are 120,731 (a 481×251 grid).

Spyropoulous and Blaisdell modified the DNS code of Rai *et al.* to perform LES using a dynamic SGS model [16, 17]. Therefore the numerical methods are the same as the DNS. The computational domain is divided along the streamwise direction into three regions. The first region is 3 inches long and contains the regions of blowing and suction, as well as transition. The second region is 2 inches long and contains the turbulent region. The third region is 6 inches long and becomes gradually very coarse to artificially dissipate all the turbulent fluctuations. Spyropoulos and Blaisdell simulated 9 cases of LES with different combinations of numerical accuracy, grid points, SGS modeling, filtering and disturbance amplitude. Only three cases with the same conditions excepting grid point numbers are plotted in Figure 7.46; these are 4th-order accurate with dynamic SGS modeling, filtering in all three direction and a disturbance amplitude of 4. Case A has the finest grid with $416(116,281,21) \times 55 \times 257$ points, case B the medium grid with $416(116,281,21) \times 55 \times 129$ points and case C the coarsest grid with

$311(116,176,21) \times 55 \times 61$ points. Note that our calculation had a grid size of $305 \times 55 \times 61$ (Section 6.2).

All the simulations follow the Blasius laminar skin friction coefficient function well until the transition starts. In the current simulation, transition starts to occur at $x = 4.0$in and as in other simulations the skin friction coefficient increases rapidly here. Experiments [109] suggest that after transition the skin friction coefficient should be about 0.0018. Our skin friction coefficient is higher than 0.0018 but DNS and fine grid LES also over-predict the skin friction coefficient. Spyropoulos and Blaisdell commented that the overshoot seen in their DNS is typical of bypass transition[1].

Strangely, the LES predicts the skin friction coefficient better than the DNS in general. But when grid resolution is too low the skin friction coefficient is underestimated. Compared to case C of LES which has a similar number of grid points, our skin friction coefficient is better showing the general trend of experimental results.

Spyropoulos and Blaisdell also tried LES without SGS modeling for case A, and concluded that the SGS model does not have a big effect on the accuracy of the LES, at least in predicting the correct skin friction coefficient.

Velocity Profile

Figure 7.47(a) shows the normalized mean velocity profile of the fully turbulent boundary layer from MILES (Case 2, Table 6.1). For comparison, results from DNS by Rai *et al.* [28] and LES by Spyropoulos and Blaisdell [30], and the law of the wall are all plotted in the same graph. The simulation conditions of DNS and LES have been briefly described in an earlier section.

Our Case 2 simulation used a similar number of grid points as the LES (Case C) of Spyropoulos and Blaisdell. While Spyropoulos and Blaisdell's Case C results were about 20% out from the law of the wall and DNS results, our results

[1] Transition emanating from linear mechanisms other than exponential instabilities.

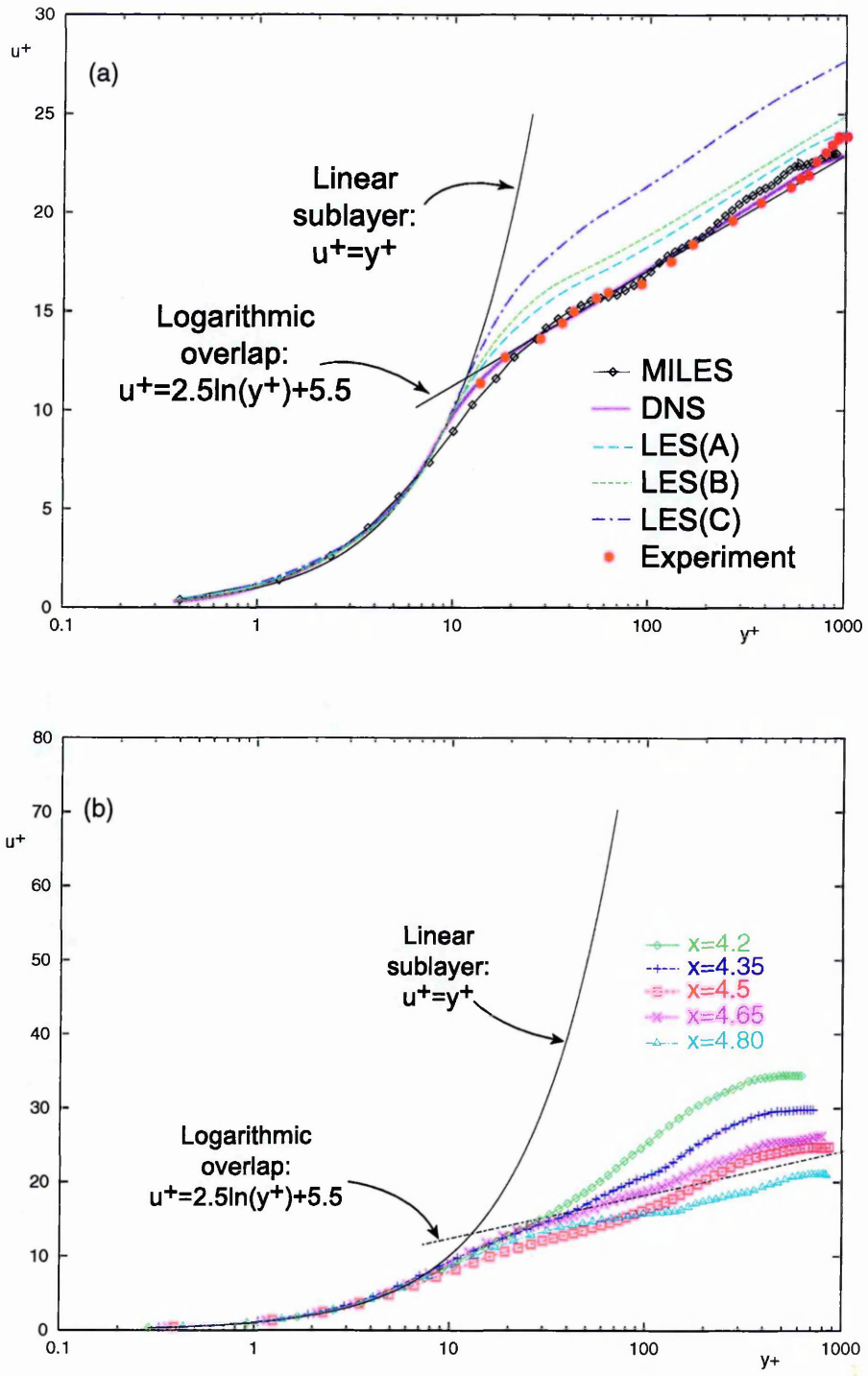

Figure 7.47: (a) Plot of the normalized mean velocity profile at $x = 4.7$ when Roe's scheme was used, together with the law of the wall, DNS by Rai *et al.* [28], and LES by Spyropoulos and Blaisdell [30]; (b) Plot of mean velocity profile along the plate.

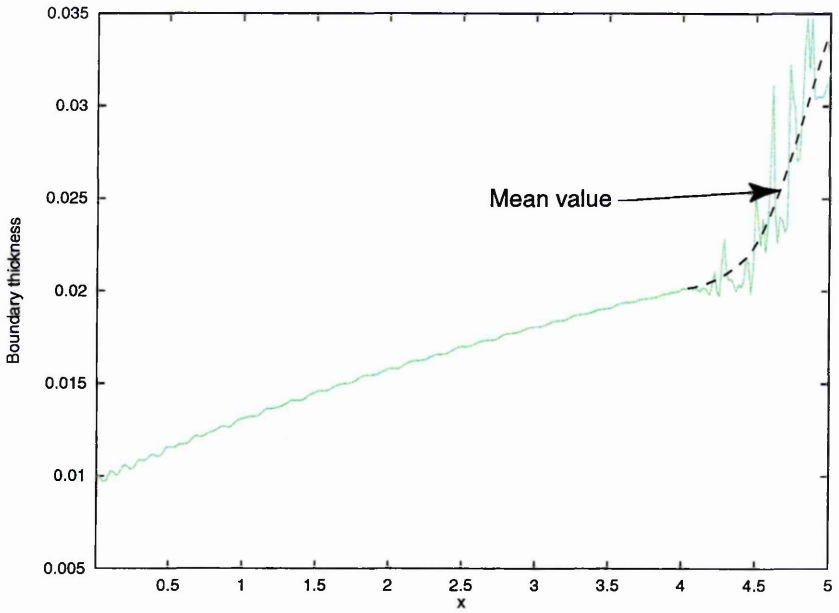

Figure 7.48: Boundary Layer thickness

found excellent agreement with them. The LES results with larger numbers of grid points (Case A and B) showed improvement but they were still not as good as the DNS results or the current results.

Figure 7.47(b) shows the mean velocity profiles along the plate. The velocity profiles more or less match the law of the wall after the boundary layer became turbulent.

Boundary thickness

Figure 7.48 shows the growth of average boundary layer thickness along the plate. The dotted line represents the mean value.

It is clear from the plot as to where the transition point is. As expected, the rate of growth of the boundary layer thickness is much increased after transition.

Turbulent intensities

The turbulence intensities normalized by the freestream velocity along the streamwise, wall-normal, and spanwise directions are plotted in Figures 7.49–7.51. The

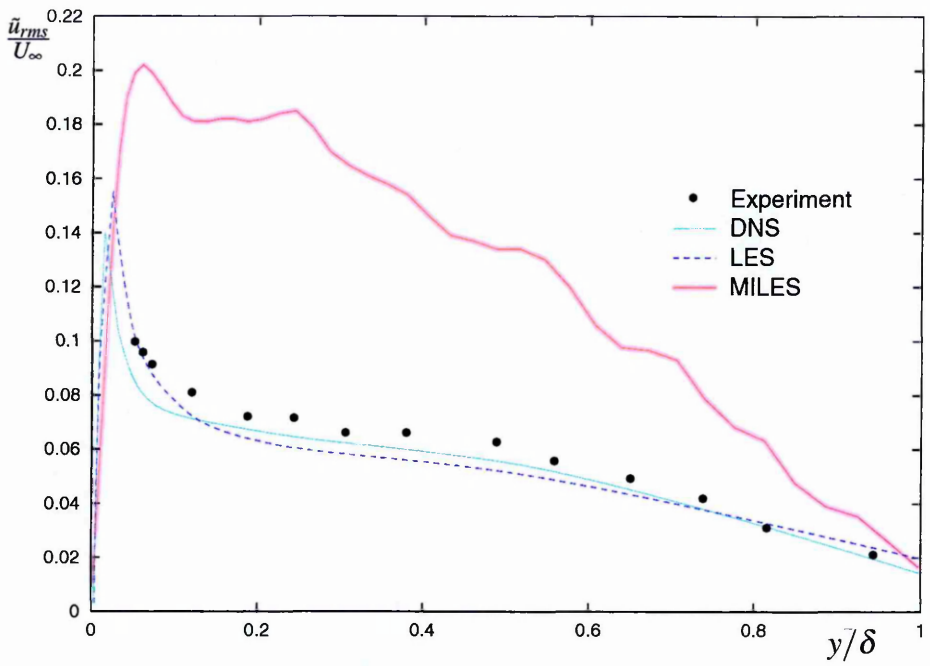

Figure 7.49: Profile of normalized, streamwise turbulent intensities at $x = 4.7$

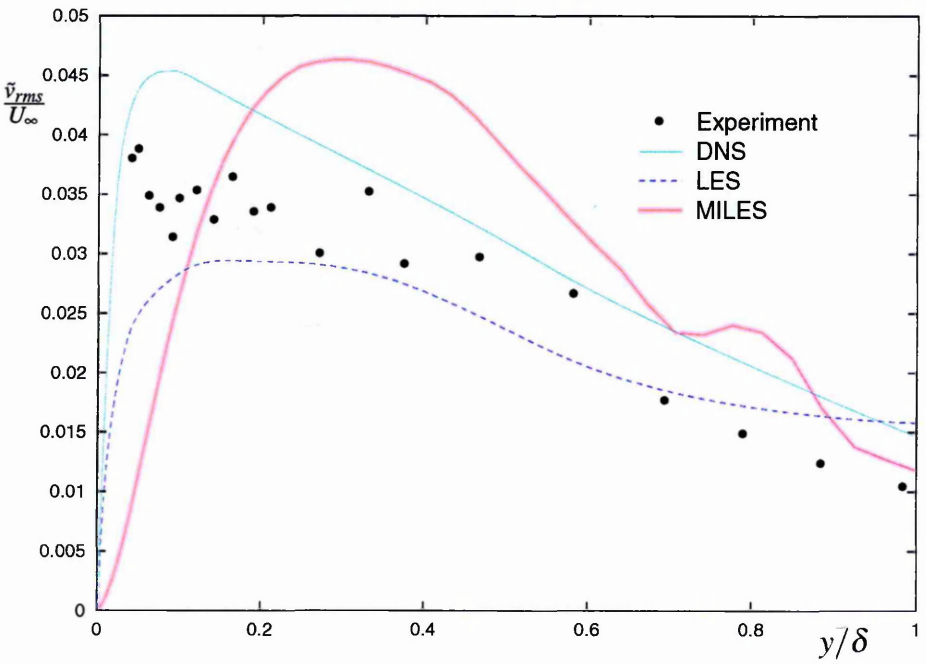

Figure 7.50: Profile of normalized, wall-normal turbulent intensities at $x = 4.7$

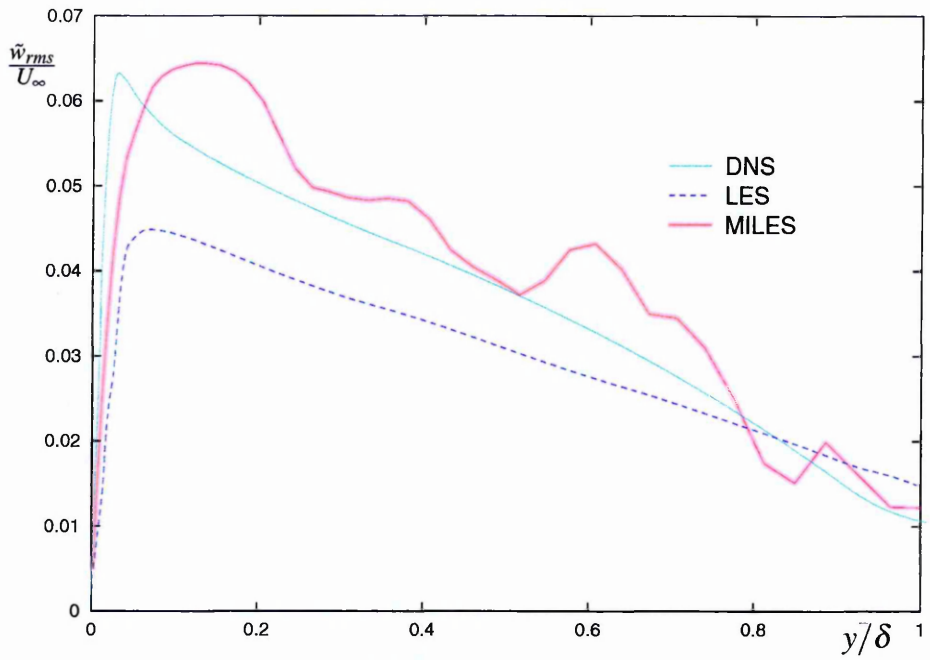

Figure 7.51: Profile of normalized, spanwise turbulent intensities at $x = 4.7$

experimental data is a Laser Doppler Anemometer (LDA) measurement by Elena and LaCharme [2], the DNSs are by Rai *et al.* [28] and LESs are by Spyropoulos and Blaisdell [30] with a similar number of grid points. There is no available experimental data for the spanwise turbulent intensity in the near-wall regions in other directions. This is due to the physical difficulties in making the measurements.

In all three directions, MILES overpredicts the turbulent intensities but the prediction of streamwise turbulent intensity is particularly overpredicted. The wall-normal and spanwise turbulent intensity predicted by MILES is better than the streamwise but still higher than other results. Also the peaks are further away from the wall than other results. This could be because the grid is too coarse close to the wall.

The turbulent intensities normalized by the local mean streamwise velocity are plotted in wall units at $x = 4.7$ in Figures 7.52–7.54. Because compressibility effects in this flow are not strong [30] and experimental data close to the wall is not available for this kind of flow, the data from an incompressible flow over a

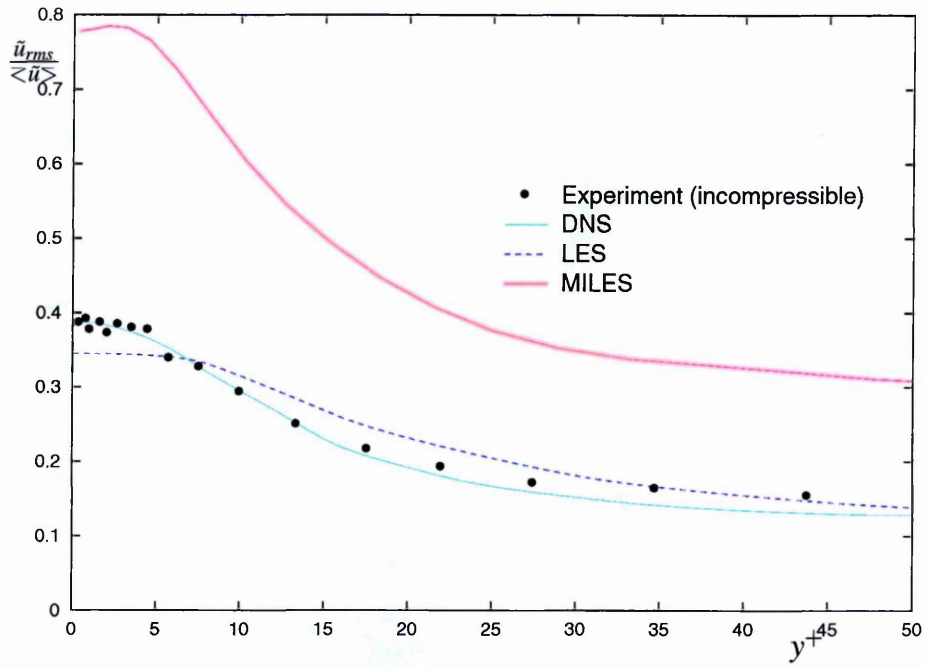

Figure 7.52: Profile of normalized, streamwise turbulent intensities in wall coordinates at $x = 4.7$

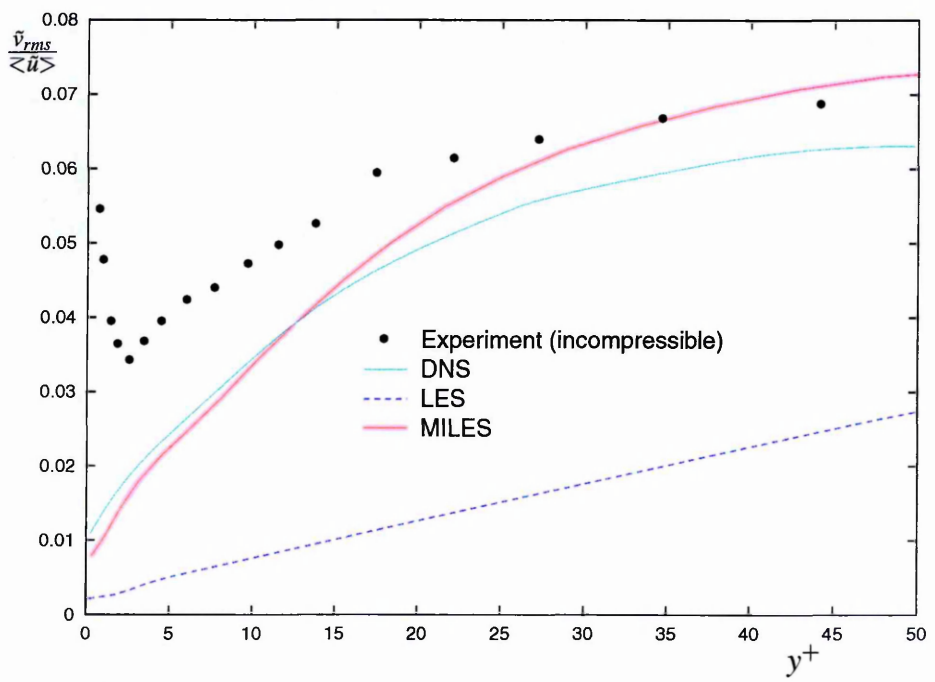

Figure 7.53: Profile of normalized, wall-normal turbulent intensities in wall coordinates at $x = 4.7$

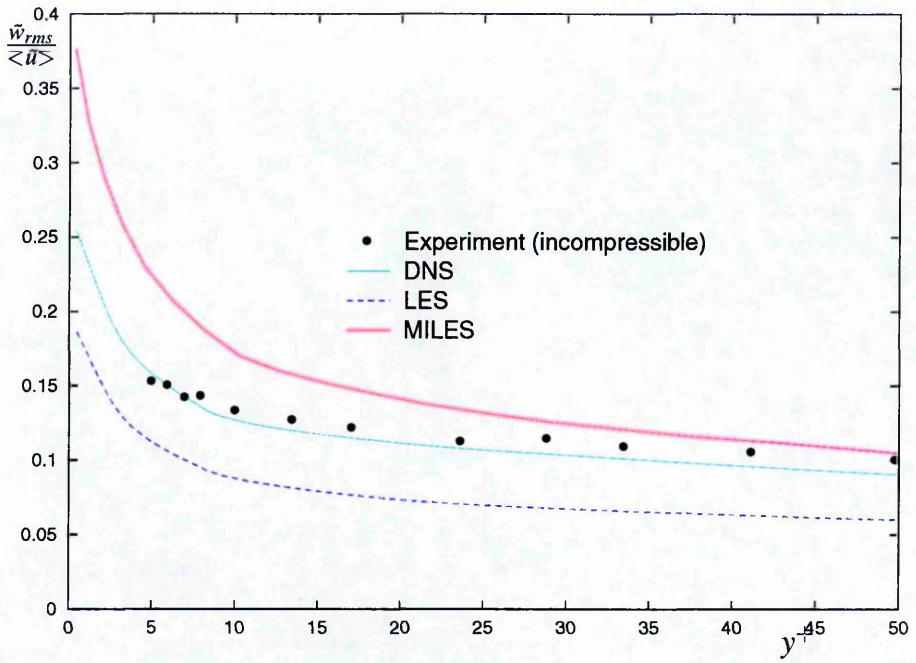

Figure 7.54: Profile of normalized, spanwise turbulent intensities in wall coordinates at $x = 4.7$

flat plate experiment by Karlson and Johansson [114] are included for comparison.

The results are mixed here. The steamwise and spanwise turbulent intensity are overpredicted compared to other results. Especially the steamwise turbulent intensity by MILES is, again, twice as large as other results. Interestingly, the wall-normal intensity is underpredicted compared to experimental measurements but compares quite well with the DNS result. In the region close to the wall, however, the experimental data for the wall-normal intensity unphysically increases and is likely to be in error. LES underpredicts the wall-normal intensity 2-5 times lower than other results depending on the distance from the wall.

Reynolds shear stress

Figure 7.55 shows the Reynolds shear stress profile at $x = 4.7$ in wall units. The Reynolds shear stress of the current result is much higher than the experimental data by Karlson and Johansson [114], DNS and LES. Since the turbulent intensity in streamwise direction was 2 to 3 time rather than other results, it was expected

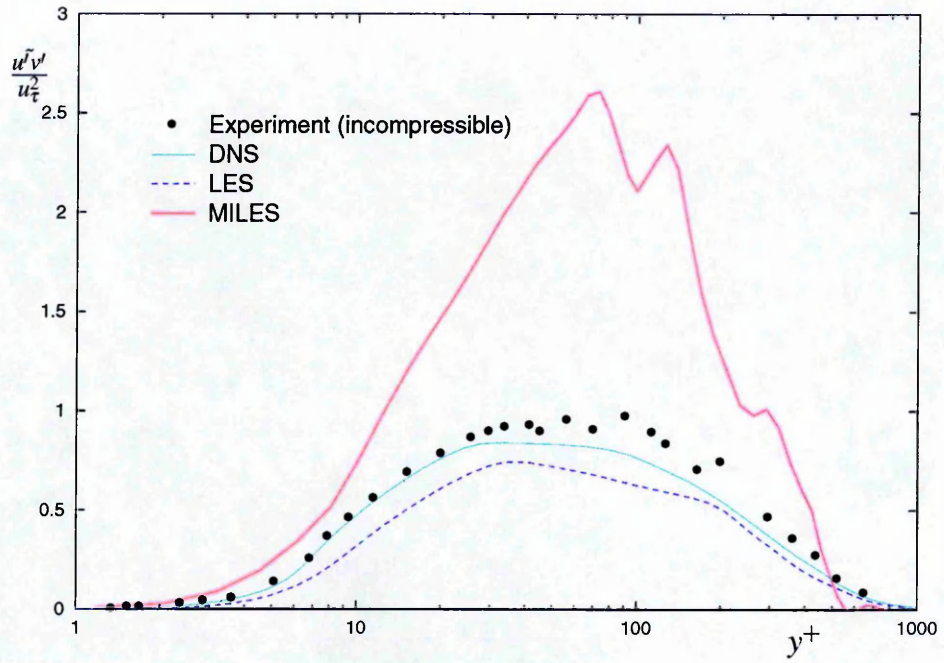

Figure 7.55: Normalized, resolved Reynolds shear stress in wall coordinates at $x = 4.7$

that predicted Reynolds stress will be twice or three time larger than other results.

Nevertheless the Reynolds shear stress peak at the similar position and its shape is similar compared to other results. In fact, if the MILES Reynolds shear stress graph is reduced by 2.5 times it will be more or less on top of other results.

7.7 Computational Requirements and Parallel Efficiency

A LES typically uses a 3–5 times coarser grid in each direction compared to DNS. Since they are both 3 dimensional calculations a LES will be 27–125 times faster than DNS when the same numerical method with the same accuracies are used. This is ignoring the time used to calculate a SGS model. LES assumes that a Navier-Stokes solver will have a numerical error much smaller than the numerical dissipation by a SGS model. As Ghosal [66] has found out from his investigation this may not always be true. Nevertheless, since LESs are done on coarse grids they may not have to use the same numerical method as a DNS and could

be numerically less accurate saving even more computational time. A simple Smagorisky SGS model will add about 10% computational time to a calculation. Dynamic SGS model will add about 30% to a calculation. Overall, LES will be about 20–100 times faster than DNS.

MILES does not explicitly calculate a SGS model therefore the calculation will be faster than LES. Still, MILES requires a great deal of computational resources. It would be close to impossible to calculate Case 2 (Table 6.1) with a current workstation because it would take too much time and require a large amount of memory. Computers are getting faster and cheaper every day but at present, the parallel computing is the most cost effective way to acquire large computational resources (see Section 2.4).

The current code has been run on several different computer systems: a Cray T3E-1200E, a SGI Origin 2000, a Beowulf system (networked PCs), and a cluster of Digital AlphaStations. Since message passing is done using a standard MPI library, the code should run on many more computer systems.

Needless to say the Cray T3E-1200E has the fastest communication speed between the processors. The Cray T3E-1200E is designed to scale to thousands of Alpha chips and one used in the current work has 776 EV5 processors each capable of 1.2Gflops and with 256Mbytes memory. But it is the speed of the interconnections between the processors which make this machine remarkably fast and extremely expensive to buy. It also has very fast hard disks which are useful when writing a large amount of data quickly. Our tests of Case 2 (Table 6.1) showed a near 99% parallel efficiency with 32 processors (in fact, 33 processors were used; one for the master program and 32 for slave programmes, but since the master program only monitors calculations it is not counted). Unfortunately due to limited CPU hours allocated to the author of this thesis, further parallel performance tests were not possible.

The SGI Origin 2000 used has 26 processors (R12,000 MIPS) and 13GB of memory. It uses a mixture of shared memories to communicate with local pro-

cessors and inter-connections to communicate with remote processors. The inter-connections in the Origin 2000 are not as fast as the Cray's but the communication using the shared memory is much faster than using an inter-connection. For our case where the computational time is a lot longer than the communication time, the Origin 2000's parallel performance was similar to the Cray T3E when tested with 16 processors. Also, each EV5 and R12,100 MIPS processor is similar in computational speed.

The Beowulf system used has 17 processors (Pentium Pro 300MHz) with 128Mb of memory each. The processors are connected by a private 100baseTX System Area Network. The inter-connections are much slower than either those of the T3E or Origin 2000 but still the parallel efficiency for test case 2 (Table 6.1) was about 95%. Considering the cost of 100baseTX systems this is a very good efficiency.

The parallelized code has also been run on AlphaStations dotted around the Cranfield campus. They are connected to the University network which reduces the parallel performance. They were mainly used during the debugging stage. When 4 AlphaStations with EV5 processors are tested with test case 2 (Table 6.1) the parallel performance was about 75%. When tested with 8 AlphaStations it was about 60%. It was difficult to conduct tests with larger number of AlphaStations because it was quite difficult to find a reasonable domain of AlphaStations with the same specification which were available and free of other jobs.

Most of the production runs have been done on the beowulf system and the SGI Origin 2000. We were able use all 17 processors of the beowulf system. For each MILES calculations, between 60,000 to 180,000 iterations were required. With the Beowulf system each iteration of Case 2 simulation took an average 18 seconds. Therefore the total run times were between 12.5 and 37.5 days. Only 9 processors of the Origin 2000 was available to us since we were sharing with other researchers. Each iteration of the Case 2 simulation took an average of 19 seconds. Therefore the total run times were similar to the Beowulf system. Since the run times were so long it was paramount that errors were reduced to a minimum.

Chapter 8

Discussions, Conclusions and Suggestions for Future Work

8.1 Numerical Schemes

In this thesis three numerical schemes (central difference, Osher's and Roe's) for the inviscid term have been tested for MILES of a supersonic flow over a flat plate.

8.1.1 Central Difference Scheme

The 2nd-order central difference scheme for the inviscid term without a SGS model was numerically very non-dissipative. As turbulence started to develop, the energy for the SGS eddies failed to be dissipated and built up resulting in computational failure. By looking at the results, the energy build up started near the wall. Addition of SGS modelling to the central difference scheme is therefore necessary as in a traditional LES which has been done by other researchers and their results are widely available.

8.1.2 Osher's Scheme

Osher's and Roe's schemes are the most popular shock-capturing schemes currently being used and have been used to simulate a wide variety of problems very successfully. Osher's scheme is slightly more dissipative than Roe's scheme. As the results showed, Osher's scheme failed to develop any turbulence. The results with Osher's scheme might improve with a finer grid but it seems that the grid

has to be much finer than what could be achieved practically with the available computing resources.

8.1.3 Roe's Scheme

Roe's scheme without any limiters produced much better results. The visualizations of results show the right characteristics of a transitional and turbulent boundary layer. The skin friction coefficient and normalized velocity profiles in the turbulent boundary layer region compare very well with experiments, DNS by Rai *et al.* [28] and theoretical values. These are better results than those obtained with LES by Spyropoulos and Blaisdell [30] using similar and finer grids.

8.1.4 Limiters

More research is needed on limiters to be used in MILES, since two commonly used limiters were found to affect the regions where no shockwaves are present. This hinders the development of turbulence.

Likely reason for the limiters effecting the non-shockwave regions is that the turbulence also creates a flux gradient which might fool a limiters as a shockwave. The limiters which we tested does not have a mechanism to distinguish a flux gradient due to turbulence from a shockwave.

Already, a new sensor for triggering artificial dissipation in a LES of the shock/turbulence interaction is proposed by Ducros *et al.* [69]. Their a priori testing shows that with the proposed scheme the dominant dissipation acting on kinetic energy is (i) the SGS dissipation away from the shock and (ii) both artificial and SGS dissipation in the shock (the former being larger than the latter). Their test case of shock/turbulence interaction did not include a near-wall boundary layer, and they used a 2nd-order central difference scheme. More tests should be done if it is to be accepted for general cases. There is no doubt other new limiters will be proposed for the use in LES in the near future.

8.2 Turbulence Statistics

It is disappointing to note in the current results that the turbulence intensity in the streamwise direction and the Reynolds stress are both 2 or 3 times over predicted. The turbulence intensities in the wall-normal and spanwise directions are predicted better but still slightly overpredicted and the peak value is further away from the wall than in other results. There are a number of possible causes for these poor predictions.

These occurrences may be due to the statistical sampling limitation in the calculation. Better turbulence statistics results might have been obtained if it was sampled over a longer period of time. As it is, the turbulence statistics are obtained by sampling in spanwise direction only.

Even though Roe's scheme in MILES predicted mean statistics, like velocity profiles and skin friction coefficients quite adequately, it may not be suitable to predict higher order statistics, like turbulent intensities and Reynolds stresses. There are other shock-capturing schemes available for testing including Godunov Exact Riemann Solver [115], Convective Upstream Split Pressure [116], Low-Dissipative High-Order Shock-Capturing Methods by Yee *et al.* [71] and Sandham and Yee [117], etc.. These schemes may be more complicated and computationally more expensive than Roe's scheme but they may be more suitable for MILES.

Another possibility could be that the grid resolution may not be fine enough. Especially the shift of the peak away from the wall may suggest that the grid resolution near the wall may not be adequate. Only two set of grid resolutions have been tested. The simulation with the coarse grid produces much less turbulence than that with the finer grid. The reason for there being less turbulence with the coarse grid would be due to the lack of resolution and therefore increased numerical dissipation. Computations with finer grids are desirable but again it was not possible with our current computer resources.

Further the MILES methodology of modelling SGS dissipation with numerical

dissipation may be the cause of the overprediction and the shift of the peak. That the numerical dissipation serves as the subgrid scale physical dissipation needs more research. This is difficult because it is difficult to quantify the numerical dissipation associated with a numerical scheme.

8.3 Numerical Accuracy

It is worth mentioning that the current scheme (with Roe & MUSCL) is numerically only 3rd-order accurate for the inviscid term and 2nd-order for the viscous term whilst previous LES results have been obtained with 5th-order accuracy for the inviscid term and 4th-order accuracy for the viscous term. Other papers [12] suggested that the numerical accuracy for the viscous term is not important compared to the inviscid term. From our results with the Roe's scheme without a limiter, it seems that the 2nd-order central difference scheme is adequate to be used in MILES.

It is not surprising that there is a great deal of interest in using LES on more complicated geometries, but this means not only many more grid points but also the use of numerical methods of 2nd or 3rd-order accuracy which have a high numerical dissipation. The numerical methods of 3rd-order accuracy were thought to be not suitable for traditional LES, because the magnitude of numerical dissipation would be of a order to the contribution of the SGS modelling; this makes the modelling redundant. The possibility of using 3rd-order accurate finite volume schemes for MILES opens up opportunities to simulate complicated geometries.

8.4 Boundary Conditions

So far only numerical dissipation has been discussed but also the boundary conditions are crucial if MILES is to produce correct results (this is also true for LES and DNS). The cyclic boundary condition in the spanwise direction was relatively simple to implement. As long as the width was large enough to contain more than 2 largest eddies a reasonable boundary condition was obtained. The cyclic boundary condition was possible because the side boundary planes are parallel to

each other. If they are not parallel the boundary conditions will be much more complicated.

Whether the inlet boundary condition is created from a 2-D calculations (as in the current scheme) or from an analytical solution or from results from DNS, the velocity profile has to be realistic otherwise there needs to be a very long domain for the flow to be realistic. Two kind of disturbances at the inlet were tried: sinusoidal disturbances superposed with Gaussian random noises, and multi-frequency blowing and suction. Blow and suction at the wall near the inlet was successfully used in DNS by Rai *et al.* [28] and LES by Spyropoulos and Blaisdell [30] but the same algorithm did not work in the current MILES scheme. If the amplitude of the blowing and suction was set to the same level as their calculation, the simulation became unstable and blew-up. If a smaller amplitude was used it created a turbulent boundary layer initially but it eventually washed away. Therefore only the results with the sinusoidal disturbance were presented in this thesis. This also illustrates the importance of a realistic inlet boundary condition.

In other simulations, a large buffer region at the top of the computational domain together with a symmetric boundary condition at the top boundary was used. This works well if the boundary layer is very small compared to the domain height. A characteristic boundary condition on the top boundary worked well in the current simulations. In some of the simulations the turbulent boundary layer appeared to touch the top boundary with no flow reflecting back being observed. This represents a large computational saving since no buffer region needed to be used on the top.

The outflow boundary condition was set to the supersonic outflow condition. However near the wall the flow becomes subsonic. The characteristic boundary condition cannot be used in this region because subsonic outflow condition needs one fixed value (see Section 6.4.1) and none of the values are known in at this region. Still, it is possible to guess the pressure in this region since the pressure does not change very much inside the boundary layer. Therefore the pressure in the subsonic region was reset to the value in the supersonic region. Without this

correction the calculation fails when the turbulent boundary layer exits. Also a 1st-order boundary condition at the outflow seems to work better than a 3rd-order boundary condition. The 1st-order boundary condition was stabler and seemed to transmit turbulence better. The outflow boundary condition does influence the calculations, but by how much is unknown and very difficult to quantify. Further research is required for a better understanding of boundary conditions in MILES.

8.5 Parallel Computing

There is no doubt that parallel computing is the cheapest way to achieve a high computational capability at present. Using MPI, the current code was parallelized by a domain decomposition of the solution domain. The parallelized code was used on 4 different computer systems and high parallel efficiencies were achieved for all of them.

8.6 Final Conclusion

Overall, some of the most popular shock capturing methods are shown to have too much dissipation in MILES. MILES using Roe's scheme without a limiter is capable of simulating a turbulent boundary layer. Its predictions of mean values e.g. velocity profiles and skin friction coefficient are better than traditional LES with an equivalent grid size. However turbulence statistics such as turbulent intensity and Reynolds stress are poorer. The cause of the poor prediction of these higher statistics is not clear and needs to be investigated further by improved sampling or numerical scheme. Nevertheless, it is shown that shock capturing methods with low numerical dissipation can be used in MILES and this has the potential to be used in simulations with more complex geometries.

Reference

[1] White Frank Mangrem. *Viscous Fluid Flow.* McGraw-Hill, New York, 1974.

[2] Elena M. and LaCharme J. P. Experimental study of a supersonic turbulent boundary layer using a laser doppler anemometer. *Journal de Mécanique Théorique et Appliquée*, 7(2):175–190, 1988.

[3] Fernholtz H. H. and Finley P. J. A critical compilation of compressible turbulent boundary layer data. Technical Report AGARDograph 223 (Case 55010501), June 1977.

[4] Fernholtz H. H. and Finley P. J. A further compilation of compressible turbulent boundary layer data with a survey of turbulence data. Technical Report AGARDograph 263, Nov. 1981.

[5] Settles G. S. and Johnson L. J. Hypersonic shock-boundary layer interaction database. Technical Report NASA CR-177577, 1991.

[6] Spina E. F., Smits A. J., and Robinson S. K. The physics of supersonic turbulent boundary layers. *Annual Review of Fluid Mechanics*, 26:287–319, 1994.

[7] Bradshaw P. Compressible turbulent shear layers. *Annual Review of Fluid Mechanics*, 9:33–54, 1977.

[8] Lele S. K. Compressibility effects on turbulence. *Annual Review of Fluid Mechanics*, 26:211–254, 1994.

[9] Pruett C. David, Zang Thomas A., Chang Chau-Lyan, and Carpenter Mark H. Spatial direct numerical simulation of high-speed boundary-layer

flows, part i: Algorithmic consideration and validation. *Theoretical and Computational Fluid Dynamics*, 7:49–76, 1995.

[10] Kleiser L. and Zang T. A. Numerical simulaiton of transition in wall-bounded shear flows. *Annu. Rev. Fluid Mech.*, 23:495–537, 1991.

[11] Guo Y., Adams N., and Kleisler L. Modeling of non-parallel effects in temporal direct numerical simulation of compressible boundary layer. *Theoretical and Computational Fluid Dynamics*, 7:141–157, 1995.

[12] Hunt D. and Nixon D. A very large eddy simulation of an unsteady shock wave/turbulent boundary layer interaction. *AIAA-95-2212*, 1995.

[13] Sandham N. D., Adams N. A., and Kleiser L. Direct simulation of breakdown to turbulence following oblique instability waves in a supersonic boundary layer. In P.R. Voke, editor, *Direct and Large-Eddy Simulation I*, pages 213–223. Kluwer Academic Publishers, Netherlands, 1994.

[14] Ducros Frédéric, Comte Pierre, and Lesieur Marcel. Ropes and lambda-vortices in direct and large-eddy simulations of a high-mach number boundary layer over a flat plate. In *Ninth Symposium on "Turbulent Shear Flows"*, pages 22–5, Kyoto, Japan, August 1993.

[15] El-Hady Nabil M. and Zang Thomas A. Large-eddy simulation of nonlinear evolution and breakdown to turbulence in high-speed boundary layers. *Theoretical and Computational Fluid Dynamics*, 7:217–240, 1995.

[16] Germano M., Piomelli U., Moin P., and Cabot H. A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids A*, 3(7):1760–1765, 1991.

[17] Germano M., Piomelli U., Moin P., and Cabot H. Erratum:"a dynamic subgrid-scale eddy viscosity model" [phys. fluids a 3, 1760 (1991)]. *Phys. Fluids A*, 3(12):3128, 1991.

[18] Childs Robert E. and Patrick H. Reisenthel. Simulation study of compressible turbulent boundary layers. *AIAA 95-0582*, 1995.

[19] Hatay F. F. and Biringen S. Direct numerical simulation of low-Reynolds-number supersonic turbulent boundary layers. *AIAA 95-0581*, 1995.

[20] Adams N. A. and Kleiser L. Subharmonic transition to turbulence in a flat-plate boundary layer at Mach number 4.5. *J. Fluid Mech.*, 317:301–335, 1996.

[21] Urbin Gerald and Doyle Knight. Compressible large eddy simulation using unstructured grid: Supersonic turbulent boundary layer and compression corner. *AIAA 99-0427*, 1999.

[22] Normand X. and Lesieur M. Direct and large-eddy simulations of transition in the compressible boundary layer. *Theor. Comput. Fluid Dyn.*, 3:231–252, 1992.

[23] Métais O. and Lesieur M. Spectral large-eddy simulation of isotropic and stably stratified turbulence. *J. of Fluid Mech.*, 239:157–194, June 1992.

[24] Ducros Frédéric, Comte Pierre, and Lesieur Marcel. Large-eddy simulation of transition to turbulence in a boundary layer developing spatially over a flat plate. *J. Fluid Mech.*, 326:1–36, 1996.

[25] Maestrello L., Bayliss A., and Krishnan R. On the interaction between first and second-mode waves in a supersonic boundary layer. *Phys. Fluids A*, 3(12):3014–3020, 1991.

[26] Fasel H., Thumm A., and Bestek H. Direc numerical simulation of transition in supersonic boundary layers: Oblique breakdown. In Kral L. D. and Zang T. A., editors, *Transitional and Turbulent Compressible Flows*, volume 151. ASME, New York, 1993.

[27] Shan H., Jiang L., Zhao W., and Liu C. Large eddy simulation of flow transition in a supersonic flat-plate boundary layer. *AIAA 99-0425*, 1999.

[28] Rai Man Mohan, Gatski Thomas B., and Erlebacher Gordon. Direct simulation of spatially evolving compressible turbulent boundary layers. *AIAA 95-0583*, 1995.

[29] Arad E. and Martinelli L. Large eddy simulation of compressible flow using a parallel, multigrid driven algorithm. *AIAA-96-2065*, 1996.

[30] Spyropoulos Evangelos T. and Blaisdell Gregory A. Large-eddy simulation of a spatially evolving supersonic turbulent boundary-layer flow. *AIAA Journal*, 36(11):1983–1990, 1998.

[31] Ferziger J.H. Higher level simulations of turbulent flow. In Essers J. A., editor, *Computational Methods for Turbulent, Transonic, and Viscous Flows*, pages 93–182. Hemisphere, 1983.

[32] Ferziger Joel H. Large eddy numerical simulations of turbulent flows. *AIAA Journal*, 15(9):1261–1267, September 1977.

[33] Hunt J. C. R. Vorticity and vortex dynamics in complex, turbulent flows. *Transaction of the CSME*, 11(1):21–35, 1987.

[34] Wilcox David C. *Turbulence Modeling for CFD*. DCW Industries Inc., La Cañada, California, 1993.

[35] Kolmogorov A. N. The local structure of turbulence in an incompressible viscous fluid for very large Reynolds numbers. *C. R. Acad. Sci. URSS*, 30:301–305, 1941.

[36] Kolmogorov A. N. Dissipation of energy in locally isotropic turbulence. *C. R. Acad. Sci. URSS*, 32:16–18, 1941.

[37] Landahl M. T. and Mollo-Christensen E. *Turbulence and Random Processes in Fluid Mechanics*. Cambridge University Press, Cambridge, England, 2nd edition, 1992.

[38] Hinze J. O. *Turbulence*. McGraw-Hill, New York, 2nd edition, 1975.

[39] Gurvich A. S., Koprov B. M., Tsvang L. R., and Yaglom A. M. Data on the small-scale structure of atmospheric turbulence. In A. M. Yaglom and V. I. Tatarskii, editors, *Atmospheric turbulence and radio wave propagation*, pages 30–52. Nauka Press, Moscow, 1967.

[40] Rogallo R. S. and Moin P. Numerical simulation of turbulent flows. *Annu. Rev. Fluid Mech.*, 16:99–137, 1984.

[41] Erlebacher G., Jussaini M. Y., Speziale C. G., and T. A. Zang. Toward the large-eddy simulation of compressible turbulent flows. *J. Fluid Mech.*, 238:155–185, 1992.

[42] Mason P. J. Large-eddy simulation: A critical review of the technique. *Q. J. R. Meteorol. Soc.*, 120:1–26, 1994.

[43] Anderson D. A., Tannehill J. C., and Pletcher R. H. *Computational Fluid Mechanics and Heat Transfer.* Hemispher Publishing Corporation, New York, 1st edition, 1984.

[44] Ciofalo Michele. Large-eddy simulation: A critical survey of models and applications. *Advances in Heat Transfer*, 25:321–419, 1994.

[45] Vreman A. W., Geurts B. J., Kuerten J. G. M., and Zandbergen P. J. A finite volume approach to large eddy simulation of compressible, homogeneous, isotropic, decaying turbulence. *International Journal For Numerical Methods in Fluids*, 15:799–816, 1992.

[46] Ragab Saad A., Shaw-Ching Sheen, and Sreedhar Madhu. An investigation of finite-difference methods for large-eddy simulation of a mixing layer. *AIAA 92-0554*, 1992.

[47] Yang Kyung-Soo and Ferziger Joel H. Large-eddy simulation of turbulent flow in a channel with a surface-mounted two-dimensional obstacle using a dynamic subgrid-scale model. *AIAA 93-0542*, 1993.

[48] Chapman Dean R. A perspective on aerospace CFD. *Aerospace America*, pages 16–58, January 1992.

[49] Ghosal Sandip, Lund Thomas S., Moin Parviz, and Akselvoll Knut. A dynamic localization model for large-eddy simulation of turbulent flows. *J. Fluid Mech.*, 286:229–255, 1995.

[50] Voke P. R. Multiple mesh simulation of turbulence. In Galperin Boris and Steven A. Orszag, editors, *Large Eddy Simulation of Complex Engeineering and Geophysical Flows*. Cambridge University Press, 1993.

[51] Moin Parviz and Kim John. Numerical investigation of turbulent channel flow. *J. Fluid Mech.*, 118:341–377, 1981.

[52] Gropp W., Lush E., and Skjellum A. *Using MPI: Portable Parallel Programming with the Message-Passing Interface*. MIT Press, 1994.

[53] Robichaux Joseph, D. K. Tafti, and S. P. Vanka. Large-eddy simulaitons of tubulence on the CM-2. *Numerical Heat Transfer, Part B*, 21:367–388, 1992.

[54] Hillis D. *The Connection Machine*. MIT Press, Mass., 1985.

[55] Hunt David Leslie. *An Investigation of Supersonic Buffet Using a Large Eddy Simulation*. PhD thesis, Department of Aeronautical Engineering, Queens's university, Belfast, 1995.

[56] Childs R. E. and Nixon D. Unsteady three-dimensional simulations of a VTOL upwash fountain. *AIAA 86-0212*, 1986.

[57] Speziale Charles G. Turbulence modeling for time-dependent RANS and VLES: A review. *AIAA Journal*, 36(2):173–184, 1998.

[58] Boris J. P., Grinstein E. S., Oran F. F., and Kolbe R. L. New insight into large eddy simulation. *Fluid Dynamics Research*, 10:199–228, 1992.

[59] Ansari A. and Strang W. Z. Large-eddy simulation of turbulent mixing layers. *AIAA 96-0684*, 1996.

[60] Bui Trong T. A parallel, finite-volume algorithm for large-eddy simulation of turbulent flows. *AIAA-99-0789*, 1999.

[61] Boris Jay P. On large eddy simulation using subgrid turbulence models. In Lumley J. L., editor, *Whither Turbulence? Turbulence at the Crossroads*, pages 344–353. Springer-Verlag, 1989.

[62] W. C. Reynolds. The potential and limitations of direct and large eddy simulations. In Lumley J. L., editor, *Whither Turbulence? Turbulence at the Crossroads*, pages 313–342. Springer-Verlag, 1989.

[63] Sagaut P., Troff B., Lê T. H., and Loc Ta Phuoc. Two-dimensional simulations with subgrid scale models for separated flow. In Voke Peter R., Kleiser Leonhard, and Chollet Jean-Pierre, editors, *Direct and Large-Eddy Simulation I*, pages 102–120. Kluwer Academic Publishers, 1994.

[64] Held J. and Fuchs L. Large eddy simulation of separated transonic flow around a wing section. *AIAA 98-0405*, 1998.

[65] Canuto C., Hussaini M. Y., Quarteroni A., and Zang T. A. *Spectral Methods in Fluid Dynamics*. Springer, Berlin, 1988.

[66] Ghosal Sandip. Mathematical and physical contraints on LES. *AIAA 98-2803*, 1998.

[67] Ghosal Sandip. An analysis of numerical errors in large-eddy simulation of turbulence. *Journal of Computational Physics*, 125:187–206, 1996.

[68] Olsson M. and Fuchs L. Significant terms in dynamic sgs-modeling. In Voke Peter R., Kleiser Leonhard, and Chollet Jean-Pierre, editors, *Direct and Large-Eddy Simulation I*, pages 73–83. Kluwer Academic Publishers, 1994.

[69] Ducros F., Ferrand V., Nicoud F., Weber C., Darracq D., Gacherieu C., and Poinsot T. On the use of shock-capturing schemes for large eddy simulation. *Journal of Computational Physics*, 153:273–311, 1999.

[70] Garnier Eric, Mossi Michele, Sagaut Pierre, Pierre Comte, and Deville Michel. On the use of shock-capturing schemes for large eddy simulation. *Journal of Computational Physics*, 153:273–311, 1999.

[71] Yee H. C., Sandham N. D., and Djomehri M. J. Low-dissipative high-order shock-capturing methods using characteristic based filters. *Journal of Computational Physics*, 150:199–238, 1999.

[72] Toro Eleuterio F. *Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction.* Springer, 1997.

[73] Anderson W. Kyle and Thomas James L. Comparison of finite volume flux vector splittings for the euler equations. *AIAA Journal*, 24(9):1453–1460, 1986.

[74] Yee H. C. Explicit and implicit multidimensional compact high-resolution shock-capturing methods: Formulation. *Journal of Computational Physics*, 131:216–232, 1997.

[75] Kawamura T. and Kuwahara K. Computation of high Reynolds number flow around a circular cylinder with surface roughness. In *AIAA 22nd Aerospace Sciences Meeting*, 1984.

[76] Porter D. H., Pouquet A., and Woodward P. R. Kolmogorov-like spectra in decaying three-dimensional supersonic flow. *Physics of Fluids*, 6(6):2133–2142, 1994.

[77] Oran Elaine S. and Boris Jay P. Computing turbulent shear flows – a convenient conspiracy. *Computers in Physics*, 7(5):523–533, 1993.

[78] Grinstein F. F., Gutmark E. J., Parr T., Hanson-Parr D., and Obey-sekare U. Streamwise and spanwise vortex interaction in an axisymmetric jet. a computational and experimental study. *Physics of Fluids*, 8(6):1515–1524, 1996.

[79] Grinstein F. F. and Devore C. R. Dynamics of coherent structures and transition to turbulence in free square jets. *Physics of Fluids*, 8(5):1237–1251, 1996.

[80] Gathmann R. J., Si-Ameur M., and Mathey F. Numerical simulations of three-dimensional natural transition in the compressible confined shear layer. *Physics of Fluids*, 8(6):1515–1524, 1996.

[81] Fureby C. On sub grid scale modeling in large eddy simulations of compressible fluid flow. *Physics of Fluids*, 8(5):1301–1311, 1996.

[82] Fureby C. and Grinstein F. F. Monotonically integrated large eddy simulation of free shear flows. *AIAA Journal*, 37(5):544–556, 1999.

[83] Moller S. I., Lundgren E., and Fureby C. Large eddy simulation of unsteady combustion. In *26th Symposium (international) on Combustion Combustion Inst.*, pages 241–248, Pittsburgh, PA, 1996.

[84] Fureby C., Nilsson Y., and Andersson K. Large eddy simulation of supersonic base flow. *AIAA-99-0426*, 1999.

[85] Anderson John D., jr. *Computational Fluid Dynamics The Basic with Applications*. McGraw-Hill Book Co., 1995.

[86] Ferziger Joel H. and Perić. *Computational Methods for Fluid Dynamics*. Springer, Berlin, 1996.

[87] Bertsekas Dimitri P. and Tsitsiklis John N. *Parallel and Distributed Computer Numerical Methods*. Prentice-Hall International Inc., 1989.

[88] Richardson Harvey. High performace fortran history, overview and current status. Technical report, Edinburgh Parallel Computing Centre, University of Edinburgh, September 1995. It can be obtained from the URL: "http://www.epcc.ed.ac.uk/epcc-tec/documents.html".

[89] Geist A., Beguelin A., Dongarra J., Mancheck R., Jiang W., and Sunderam V. *PVM: A Users' Guide and Tutorial for Networked Parallel Computing*. MIT Press, 1994.

[90] Beguelin A., Dongarra J. J., Geist G. A., Manchek R., and Sunderam V. S. A Users' Guide to PVM Parallel Virtual Machine. Technical Report ORNL/TM-12187, Oak Ridge National Laboratory, September 1994. It can be obtained from the URL: "http://netlib2.cs.utk.edu/pvm3/book/pvm-book.ps".

[91] Otto S., Dongarra J., Huss-Lederman S., Snir M., and Walker D. *MPI: The Complete Reference*. MIT Press, 1995.

[92] Message Passing Interface Forum. MPI: A Message-Passing Interface Standard. Technical report, University of Tennessee, Knoxville, Tennessee, U.S.A., June 1995. It can be obtained from the several URLs including: "http://www.epcc.ed.ac.uk/epcc-tec/documents.html" and "http://www.mcs.anl/Projects/mpi/standard.html".

[93] Malard Joël. MPI: A Messag-Passing Interface standard History, Overview and Current Status. Technical report, Edinburgh Parallel Computing Centre, University of Edinburgh, 1995. It can be obtained from the URL: "http://www.epcc.ed.ac.uk/epcc-tec/documents.html".

[94] Vreman Bert, Geurts Bernard, and Kuerten Hans. Discretization error dominance over subgrid terms in large eddy simulation of compressible shear layers in 2d. *Communications in Numerical Methods in Engineering*, 10:785–790, 1994.

[95] Leonard B. P. Energy cascade in large eddy simulations of turbulent fluid flows. *Advances in Geophysics*, 18A:237–248, 1974.

[96] Deardorff James W. A numerical study of three-dimensional turbulent channel flow at large Reynolds numbers. *J. Fluid Mech.*, 41, part2:453–480, 1970.

[97] Orszag S. A. and Israeli M. Numerical simulation of viscous incompressible flows. *Annu. Rev. Fluid Mech.*, 6:281–318, 1974.

[98] Qin N. and Foster G. W. Study of flow interactions due to a supersonic lateral jet using high resolution navier-stokes solutions. *AIAA 95-2151*, 1995.

[99] Qin N. and Redlich A. Massively separated flows due to transverse sonic jet in laminar hypersonic stream. *Shock Waves: an International Journal*, 9(2):87–93, 1999.

[100] Richardson G. and Qin N. Effects of compressibility and roughness for turbulence modelling of hypersonic ramp flow. *AIAA 99-1019*, 1999.

[101] Hirsch C. *Numerical Computation of Internal and External Flows. Vol. 1. Fundamentals of Numerical Discretization.* John Wiley & Sons Ltd., Chichester, England, 1990.

[102] Roe P. L. Aproximate riemann solvers, parameter vectors and difference schemes. *J. Comput. Phys.*, 43:357–372, 1981.

[103] Spekreijse S. P. *Multigrid Solution of the Steady Euler Equation.* PhD thesis, Centrum voor Wiskumde en Informatica, Amstardam, 1987.

[104] Turkel E. Accuracy of schemes with nonuniform meshes for compressible fluid flows. In *Applied Numerical Mathematics 2*, pages 529–550. North-Holland Publishing, 1996.

[105] Prince S. A., Ludlow D. K., and Qin N. Phantom vorticity in Euler solutions on highly stretched grids. In *22nd International Congress of Aeronautical Sciences*, Harrogate, England, 2000. To be appear.

[106] Prince Simon A. *The Aerodynamics of High Speed Aerial Weapons.* PhD thesis, Cranfield College of Aeronautics, Cranfield University, Bedfordshire, U.K., 1999.

[107] Message Passing Interface Form. *MPI-2: Extensions to the Message-Passing Interface.* University of Tennessee, Knoxville, Tennessee, 1997.

[108] Snir Marc, Otto Steve, Huss-Lederman Steven, Walker David, and Dongarra Jack. *MPI: The Complete Reference.* The MPI Press, Cambridge, Massachusetts, 1996.

[109] Shutts W. H., Hartwig W. H., and Weiler J. E. Final report on turbulent boundary layer and skin friction measurements on a smooth, thermally insulated flat plate at supersonic speeds. Technical Report 364, Deutsche Forschungsanstalt für Luft- und Raumfahrt, 1955.

[110] Giles Michael. Non-reflecting boundary conditions for the Euler equations. Technical Report CFDL-TR-88-1, Computational Fluid Dynamics Laboratory, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, 1988.

[111] Thompson Kevin W. Time dependent boundary conditions for hyperbolic systems. *Journal of Computational Physics*, 68:1–24, 1987.

[112] Gustafsson Bertil. Numerical boundary conditions. *Lectures in Applied Mathematics*, 22:279–308, 1985.

[113] Atkins H. L. Nonreflective boundary conditions for high-order methods. *AIAA 93-0152*, 1993.

[114] Karlson R. I. and Johansson T. G. LDV Measurements of higher-order moments of velocity fluctuations in a turbulent boundary layer. *Laser Anemometry in Fluid Mechanics*, 1998.

[115] Okong'o N. and Knight D. Accurate three-dimensional unsteady flow simulation using unstructured grids. *AIAA-98-0787*, 1998.

[116] Tatsumi S., Martinelli L., and Jameson A. A new high resolution scheme for compressible flows with shocks. *AIAA-95-0466*, 1995.

[117] Sandham N.D. and Yee H. C. Performance of low dissipative high order tvd schemes for shock-turbulence interactions. Technical Report RIACS Technical Report 98.10, NASA Ames Research Center, 1998.