ABSTRACT

Title of Dissertation: A GRID-FREE LAGRANGIAN DILATATION ELEMENT METHOD WITH APPLICATION TO COMPRESSIBLE FLOW

Jun Shen, Doctor of Philosophy, 2004

Dissertation directed by: Professor Peter S. Bernard
Department of Mechanical Engineering

In the computational fluid dynamics research, grid-free methods are getting more attention as an alternative to traditional grid-based methods due to two important reasons. First, grid-free methods can be very easily adapted into applications involving complicated geometries. Secondly, they are less vulnerable to numerical diffusion introduced by spatial discretization than in grid-based schemes.

A new grid-free Lagrangian dilatation element method for compressible flow has been developed in this research as an extension of incompressible vortex methods. It differs from grid-based numerical methods in a number of ways. The discretization is represented by a group of Lagrangian particles that are convected with the fluid flow velocities instead of a fixed spatial grid system. The velocity of

the flow field, necessary in each time step to move the computational elements, is recovered from the dilatation distribution similar to the 'Biot-Savart' law used in incompressible vortex methods. The Fast Multi-pole Method (FMM) is used to speed up the process and reduce the cost from $O(N^2)$ down to $O(N \log N)$. Each computational particle carries physical properties such as dilatation, temperature, density and geometric volume. These properties are governed by the Lagrangian governing equations derived from the Navier-Stokes equations. While the computational elements are convected in the flow, their properties are updated by integrating their corresponding governing equations. The spatial derivatives appearing in the Lagrangian governing equations are evaluated by using moving least-square fitting. The implementation of several different boundary conditions has been developed in this research. The non-penetration wall boundary condition is implemented by adding a potential velocity field to that recovered from the dilatation elements so as to cancel the normal component at the wall. The zero-gradient of properties at the wall such as temperature and density is enforced by a technique called particle reflection. The inflow and outflow conditions are implemented with the help of the characteristic waves moving up and down-stream. The addition and removal of Lagrangian computational elements at the inlet and outlet are implemented to ensure that the computational domain is fully covered by an approximately uniform distribution of particles with roughly comparable volumes.

The new grid-free dilatation method is applied to the compressible oscillating waves in an enclosed tube and a subsonic nozzle flow. Both one-dimensional and two-dimensional results are shown and compared with either the exact solutions or the solutions given by other proven numerical schemes. Good agreement of

these results helps to establish the correctness of the present method. Future work will accommodate viscous terms and shock waves, which is given a brief discussion at the end of this thesis.

# A GRID-FREE
# LAGRANGIAN DILATATION ELEMENT METHOD
# WITH APPLICATION TO COMPRESSIBLE FLOW

by

Jun Shen

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
2004

Advisory Committee:

Professor Peter S. Bernard, Chairman/Advisor
Assistant Professor Elias Balaras
Professor Richard V. Calabrese
Associate Professor Kenneth T. Kiger
Professor James M. Wallace

# DEDICATION

To my parents.

# ACKNOWLEDGEMENTS

I would like to express my deepest gratitude and sincere thanks to Professor Peter S. Bernard for his encouragement, support and guidance during the course of this study. Over past three years, Professor Bernard has shown many of the best academic qualities that I admire and respect deeply. His inspiration, encouragement and kindness are important factors to the success of this research. His scientific insight, wealth of experience and enthusiasm during the pursuit of science and the truth will be lifelong lessons for me to learn and follow.

I would also like to thank Professors Elias Balaras, Richard V. Calabrese, Kenneth T. Kiger, and James M. Wallace for the help and willing to serve in my dissertation committee. The financial support provided by VorCat Inc. is appreciated. I also must thank Drs. Jacob Krispin and Pat Collins from VorCat Inc. for good advice and help. I have lots of appreciation for the help and friendships offered by Mr. Jianming Yang, Dr. Xinan Liu, Dr. Ning Li, and Mr. Victor Ovchinnikov.

There are no simple words that can describe the support and love given by my parents. I owe them so much I can never return. Studying in a different country with so many things they have never seen, I have made them worry and suffer a lot. I wish they are healthy, peaceful and happy forever.

TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# Chapter 1

# Introduction

The governing equations of fluid flow are a set of highly coupled partial differential equations, known as the Navier-Stokes (N-S) equations. Their exact solutions only exist for few very simply cases, so the pursuit of solving the N-S equations relies mostly on numerical techniques. Some popular, widely used schemes are finite difference methods, finite volume methods, finite element methods, and spectral methods. All of these are built upon a fixed spatial grid system. The computational domain is first discretized into interconnected nodes, and the equations are solved on the discrete nodes. These types of solution procedures may be classified as grid-based methods. The difficulty of generating suitable grids inhibits their application to problems with complex geometries. Moreover, grid-based schemes have the potential to bring in nonphysical numerical diffusion to smear the resolution.

Another type of numerical method, named grid free particle methods, have the possibility of addressing the problems encountered by grid based schemes. For example, they do not need a fixed spatial grid to perform simulations, and they

tend not to suffer from numerical diffusion. Vortex methods, as an example of the grid-free particle methods, have been intensively applied to incompressible flows. The generalization of incompressible vortex methods to solving compressible flow is the main goal of this research. A number of grid-free methods have been proposed in the literature in the last several decades that have been developed with both the vorticity and velocity represented by moving elements and for both incompressible and compressible flow. We now give an overview of some of these schemes.

## 1.1 The development of vortex methods

Grid-free methods are being applied in many numerical simulations of fluid flow as an alternative to traditional grid-based methods due to their appealing properties. In general, vortex methods are a type of grid-free numerical method for approximating solutions of the Navier-Stokes equations. The discretization of vortex methods is in vorticity form instead of the usual velocity-pressure form. Vortex methods use fluid particles as the basic computational element that carry concentrations of vorticity, convect with the fluid motion, and thereby account for the evolution with time. Thus they are a class of Lagrangian methods. The velocity field which is necessary to convect fluid particles is recovered from the discretized vorticity field using the Biot-Savart Law.

Vortex methods were originated for solving the incompressible Euler equations. The first vortex method can be traced back to Rosenhead's 1931 work [61] for computing the roll-up of a vortex sheet. It is known as the *point vortex method*. It is apparent that when two point vortices move close to each other, the induced velocities go to infinity. For this reason it was generally believed that

the point vortex method was unstable and impractical for the solutions of the incompressible Euler equations. However, Goodman, Hou, and Lowengrub [39] later proved that two neighboring vortex points which are initially separated by a certain distance $h$ will remain separated by a distance $O(h)$ for any finite time. In fact, the point vortex method does converge to solutions of the two-dimensional Euler equations.

The work of Chorin [16] is believed to be the first practical vortex method. It introduced *vortex blobs* to solve for incompressible slightly viscous flow around a cylinder, in which a bounded kernel of fixed radius was used as a cut-off function to eliminate the velocity singularity existing in the point vortex method. Fractional steps, inherently linked to the viscous splitting algorithm, are taken to handle successively inviscid and viscous parts. The vorticity field calculated after the inviscid convection step is used as the initial condition for the viscous diffusion equation. The Green's function appearing in the integral solution to the diffusion equation is modeled via a random walk (Brownian motion) to represent the diffusion effects experienced by vorticity-carrying particles. The viscous terms then are modeled by adding a random walk to the inviscid convection of vortices. Hence the method is called the *random vortex method*. The no-slip boundary condition in a random vortex method computation is satisfied by creating vortices at grid points on the boundary such that the velocity induced by these vortices cancel the tangential component of the velocity along the boundary.

After further study, Chorin [17] proposed another particle creation algorithm known as the *vortex sheet method*. A stack of vortex sheets are built in the boundary layer region adjacent to the wall while vortex blobs are used in the region away from the wall. The no-slip condition is satisfied by creating vortex

sheets on the wall which subsequently will be convected into the flow. The interior and exterior solutions are matched by converting between sheets and vortex blobs on the interface of the sheet layer and blob region. The vortex sheet method is a widely used vortex method for computing viscous incompressible flow (e.g., see Puckett [60]). The sheet creation process and subsequent movement of the sheets into the interior region is one of the attractive features of the method. This mimics the physical process of vorticity creation at the boundary and its diffusion into the interior flow.

Another method for 3D flow originally proposed by Leonard [45] and later Chorin [18] [19] is called the *vortex filament method*. Here, the vorticity field is represented as a collection of vectors, each of which has a begining and end. The magnitude of the vectors (tubes) is proportional to the length of the vectors and the circulation. The non-penetration boundary condition associated with inviscid incompressible flow can be satisfied by the method of images by deploying symmetric vortices [33] or imposing a potential flow that cancels the normal velocity on the wall due to the vortices.

In contrast to the *random vortex method*, *deterministic vortex methods* replace the random walk used in modeling viscous terms by a non-random technique for solving the diffusion equation. The particle strength exchange (PSE) scheme first developed by Cottet and Mas-Gallic [22] is one of the most commonly used deterministic vortex methods. Other deterministic vortex methods have been developed by Choquin and Lucquin-Desreux(1988) [15], Fishelov(1990) [32], and Degond and Mas-Gallic(1989) [24].

The previously mentioned and other efforts focused mainly on laminar flow problems. Bernard et. al. [7]-[13] recently developed a new form of the vortex

method suitable for incompressible turbulence modeling. Vortex tubes are used as the principal computational element to mimic the actual tube-like vortical structures of turbulence. An unstructured array of triangular vortex sheet prisms are stacked several layer deep adjacent to solid boundaries to provide good resolution in modeling the turbulent viscous sub-layer. A sub-grid scale model that is unlike those used in grid-based LES methods accounts for small scale dissipation. This consists of Chorin's hairpin removal algorithm [20][21] in which small scale hairpin vortices that form in the course of the folding process are eliminated. The potential advantage of this method in regards to traditional LES lies in the opportunity they provide to model vortex dynamics more efficiently and directly than is possible in an Eulerian gird-based setting. The Lagrangian character of this method is also an advantage since it allows for resolution of strong internal shear layers which would otherwise be smoothed in a grid-based method. It not only makes it easier to model turbulence of complex geometries but provides an opportunity to apply new ideas about sub-grid modeling without the same concerns for numerical instability as in grid-based schemes. This approach is the basis of a commercial code by VorCat, Inc. that uses the vortex method to predict turbulence. Figure 1.1 shows the results of a simulation of a turbulent mixing layer by VorCat. Evidently the method is able to capture the physics of this turbulent flow suggesting that grid-free methods are a useful alternative to grid-based schemes. The results of using the VorCat to compute the turbulent boundary layer past a flat plate show good agreement with DNS results [9] suggesting that the approach holds out much promise of maturing into an effective means for implementing complex turbulent problems of industrial interest.

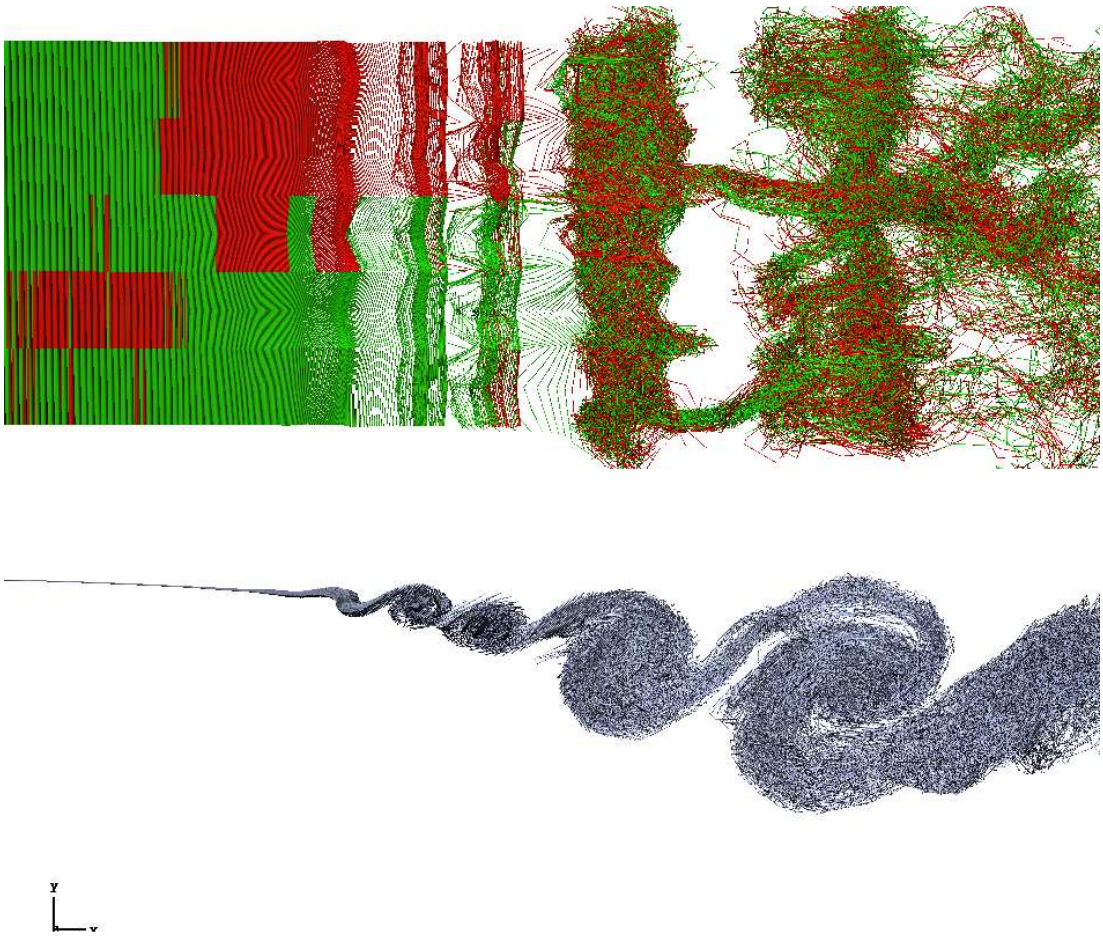


Figure 1.1: Planar and side views of turbulent mixing layer simulation done by VorCat [8].

## 1.2 Overview of velocity particle methods

Gingold and Monaghan [38] developed smooth particle hydrodynamics (SPH) for the treatment of astrophysical hydrodynamic problems. It relies on the same principle of kernel estimation as vortex methods. The velocity instead of vorticity is the primary variable. Subsequently, SPH was used in some other contexts. Monaghan and Gingold [52] applied it to solve a shock-tube problem by using a special artificial viscosity to suppress spurious oscillations. The method has also been used for computing the Rayleigh-Taylor instability [54].

Liszka and Orkisz [48] extended the finite difference method to arbitrary irregular grids. Belytschko et al. [5] developed a meshfree method called the element free Galerkin method (EFG), which uses moving least square (MLS) approximations in the context of a Galerkin method. Duarte and Oden [26] and Babuska and Melenk [51] advanced a meshless method based on partitions of unity. They also recognized that methods based on moving least squares are specific instances of partitions of unity. As pointed out by Duarte and Oden [26], any MLS approximation may serve as a partition of unity. Lohner [49] and Onate et al [58] [59] developed the finite point method (FPM) which uses moving least square (MLS) approximations to evaluate spatial derivatives and a point collocation method to discretize the governing equation. Aluru and Li [1] developed a meshfree method called the finite cloud method using point collocation to discritize the governing equation and the reproducing kernel method to evaluate spatial derivatives.

Basically all of these methods employ either reproducing kernel methods or moving least square methods or a partition of unity to construct the interpolation function of unknown variables. According to the analysis of [1] [6], the reproducing kernel method and the moving least square method are identical under

certain conditions. After getting the interpolation function, either the Galerkin method or point collocation method is used to discretize the governing partial differential equations. The Galerkin method usually needs some kind of background mesh to evaluate the integrals produced by the discretization, so it is not a strictly mesh-free method. On the other hand, the point collocation method is a true mesh-free method.

Most of the above meshless methods are of Eulerian type which use a fixed point configuration throughout the computation. The particles are fixed and stay still at their initial positions. The governing equations are also written in Eulerian forms, and it is possible to develop certain kinds of flux splitting schemes with the use of background meshes. Therefore, these kinds of methods are not strictly meshfree. Also, since a point configuration is fixed, it is possible to analyze the accuracy of the schemes. On the other hand, some of the meshless methods are Lagrangian methods like vortex methods which allow particles to convect with the local fluid velocity and track the trajectories of individual particles. The governing equations are expressed in Lagrangian form. Moreover, the particle positions and the interpolation functions are updated at every time step. Since every particle is convected with the fluid motion and there is not a fixed configuration among them, it is not obvious how to employ flux splitting or other monotonic finite difference techniques to evaluate the spatial derivatives. It poses some difficulty for this type of method to capture shock wave and other discontinuities in compressible flow.

## 1.3 Grid-free compressible particle methods

Over the years there have been a number of efforts aimed at generalizing vortex methods to include compressibility. A class of new elements containing the dilatation field is introduced to represent compressibility in some schemes. Ghoniem et. al. [34] used grid-free dilatation elements to model the volumetric expansion associated with a flame front in a combustion channel. The strength and history of the dilatation is tied to the properties of the flame propagation thereby bypassing the need to directly model the dilatation equation.

Ogami and Cheer [57] developed a so-called "interactive cored particle method" for one-dimensional compressible flow. The diffusion velocity that was defined to be proportional to both the density gradient and the diffusion coefficient and inversely proportional to the density distribution was used to model one-dimensional compressible flows. The density and its derivatives were computed using a core function.

In more recent activity, Strickland [64] considered the general requirements of a grid-free compressible flow solver and tested a specific model on radially and spherically symmetric flows containing disturbances to the vorticity and temperature fields in isentropic conditions. A single set of computational elements containing the vorticity, dilatation and temperature are used. The wave-like character of the propagating disturbances is computed successfully. In an extension of this, Nitsche [56] considered a radially symmetric swirl flow with initially constant temperature. The velocity was recovered from numerical quadrature of the Helmholtz decomposition. Differentiated terms in the equations of motion were computed using finite differences by taking advantage of the ordering of elements that is possible in one-dimensional calculations.

Eldredge [27]-[30] developed a dilatation vortex method for compressible flow with applications to aeroacoustics. A single set of Lagrangian particles carrying vorticity, dilatation, enthalpy, entropy and density were deployed and convected with the fluid flow. The particle strength exchange (PSE) method was generalized to be able to treat arbitrary spatial derivatives [29]. However, the treatment of spatial derivatives depends on quadrature. The stability of the quadrature relies on the condition that the ratio of PSE kernel radius to inter-particle spacing is greater than unity, so that the particles overlap with their neighbors. As time progresses, the flow distorts the particle grid and locally the overlap restriction may fail. To counter this, it is important to re-initialize the particles occasionally. This method has thus far been applied to computing co-rotating and leapfrogging vortices in an unbounded compressible flow region, with special interest in computing the associated acoustic field.

## 1.4 The goal and outline of this dissertation

Interest in Large Eddy Simulations (LES) for compressible turbulent flow has been rising by virtue of its reduced reliance on modeling in comparison to traditional Reynolds averaged equations and the opportunities provided by high speed computers. It is also possible to formulate a grid free LES vortex method [8]-[10] although it is traditionally practiced with grid based schemes. Furthermore, a grid-free LES for compressible flow can be envisaged after suitable generalization to include the effects of compressibility.

The motivation for the pursuit of grid free compressible LES schemes is based on a desire to gain the same advantages as those for incompressible flow. For

example: the ability to resolve sharp vortical features without diffusion, natural self-adaptivity and the deployment of novel sub-grid stress models. Moreover, compressible turbulent flow has been shown to be an important phenomenon in its own right (i.e., not conducive to modeling by mere extension of incompressible flow models). In fact, grid-free schemes provide an opportunity to analyze the role of compressibility under turbulent flow conditions from a new perspective. Thus, in a suitably generalized vortex method a class of elements containing dilatation is introduced to represent compressibility; their behavior may give direct and unique information about how compressibility effects turbulence.

How best to extend vortex methods to compressible turbulent flow is not self-evident. The goal of the present research is to develop a scheme for the grid free representation of two-dimensional compressible flow that is the first step in constructing a grid free LES scheme of compressible turbulence. In this simpler setting, we wish to develop proven numerical procedures for accommodating the kind of phenomena that will be present in three-dimensions as well as examine the effectiveness of techniques that will be more or less necessary in turbulent flow applications. This method will be designed to be consistent with future extension to 3D turbulent compressible flow. The long-term goal of the present research is to extend the vortex method into accommodating turbulent compressible flow, a capability that has yet to be realized.

Chapter 2 presents some general steps for implementing Lagrangian dilatation element methods. A simple example is used to show how the method can be applied to solving flow problems. The flow properties to be held in computational elements such as dilatation, temperature and density are introduced in this chapter as well as a set of equations governing their variations. The difficulties

to be encountered in solving the equations by a grid-free scheme are illustrated at the end of the chapter.

Although vorticity is not present in the flow treated in this research, the discussion of velocity calculation in Chapter 3 will apply to the general case where both vorticity and dilatation are present. This prepares the way for future development when viscous effects are present. According to the Helmholtz velocity decomposition, the induced velocities from given dilatation and vorticity fields are the solutions of Poisson equations. The solutions are expressed as integrals of Green's functions. The dilatation and vorticity fields are represented by a collection of discrete particles in grid-free methods, so the induced velocity at each particle is evaluated by a quadrature approximation to the integrals. The direct summation of the quadrature approximation requires $O(N^2)$ operations. Some details of a fast summation technique using the Fast Multi-pole Method (FMM) is illustrated in this chapter. The FMM brings the computational cost down to $O(N \log N)$. A potential velocity field is imposed to satisfy the velocity boundary conditions. A boundary element method is used to deploy a group of discrete sources on solid walls to approximate the potential velocity field.

Chapter 4 discusses the treatment of spatial derivatives. A Moving Least Square (MLS) method is used to compute all the spatial derivatives because this has the advantage of handling non-regular particle distributions. The accuracy and efficiency of MLS is evaluated and compared with other methods as well in this chapter.

Chapter 5 describes the detailed implementation of our numerical algorithm applied to solving a subsonic nozzle flow. The details of time-marching, solid wall boundary condition, inflow and outflow conditions, adding and removing

elements are discussed in this chapter.

Chapter 6 shows some results of applying the Lagrangian dilatation element method. A problem of compressible oscillating waves in an enclosed tube is solved in one and two dimensions. The treatment of solid wall boundary conditions are illustrated through this example. Next a subsonic nozzle flow is solved first by a quasi-1D model and then a fully two-dimensional model. The inflow and outflow conditions are established in this example. The results are compared with either exact solutions or other numerical schemes. The agreement of these comparisons shows the success of the Lagrangian dilatation element method in solving inviscid shock free compressible flow. We discuss future work to further develop this method at the end of the chapter.

# Chapter 2

# Grid-free Lagrangian Dilatation Element Method

The proposed Lagrangian dilatation element method differs from traditional grid-based Eulerian methods in a number of ways. Its discretization is not built upon a fixed spatial grid. The basic computational elements are Lagrangian particles. They are convected with the local fluid motions while the properties they carry such as dilatation, temperature and geometric volume are changed with time. In this chapter, the basic principles of the Lagrangian dilatation element method are first illustrated through the simple example of the viscous Burgers equation. Then the Lagrangian governing equations for compressible flow are developed and their treatment by this grid-free method is given a cursory discussion. Some general considerations and the difficulties contained in these equations are discussed at the end of the chapter.

## 2.1 A simple application: viscous Burgers equation

The Lagrangian dilatation element method is a Lagrangian numerical scheme for solving partial differential equations (PDEs). The computational domain is discretized into a number of Lagrangian particles. The particles are convected by the flow-field velocities with the advancement of time. The motion of the particles is decided by the local flow velocities. The properties carried by the particles are updated according to the Lagrangian governing equations that are formulated from the original PDEs. Before presenting the complete Lagrangian dilatation element method, a simple problem, namely, the viscous Burgers equation (2.1)

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{1}{Re}\frac{\partial^2 u}{\partial x^2}, \tag{2.1}$$

where $Re$ is the Reynolds number, is first used to illustrate the Lagrangian dilatation element method. This problem is defined in the domain $[0, 1]$ with the following initial and boundary conditions:

$$u(0) = u(1) = 0, \tag{2.2}$$

$$u(x, 0) = \sin(\pi x). \tag{2.3}$$

Dilatation $\theta = \nabla \cdot \mathbf{u}$ is used as the primary variable replacing velocity in the Lagrangian dilatation element method. After taking the divergence of the viscous Burgers equation (2.1), the governing equation turns into:

$$\frac{D\theta}{Dt} = -\theta^2 + \frac{1}{Re}\theta_{xx} \tag{2.4}$$

where $\theta \equiv \partial u/\partial x$, $D()/Dt = \partial()/\partial t + \mathbf{u} \cdot \nabla()$ is the material (or total) derivative.

In order to solve equation (2.4) under the framework of a Lagrangian dilatation element method, the domain is discretized into $N$ particles represented by $X_i, \theta_i$ $(i = 1, ..., N)$, where $X_i$ are the positions of particles and $\theta_i$ are their dilatations. The position of each particle is decided by solving

$$\frac{DX_i}{Dt} = u_i, \tag{2.5}$$

where $u_i$ is the convection velocity of the $i^{th}$ particle.

The initial $\theta_i$ are set to be zero at time $t = 0$. Equation (2.4) is then integrated to get the updated dilatation for each particle by a second order Runge-Kutta scheme at each time step $t_k = k\Delta t$, where $k = 1, ..., n$. The new particle positions are updated by also integrating (2.5) with a second order Runge-Kutta scheme.

The second order derivative, $\theta_{xx}$, in equation (2.4) is evaluated by moving least square fitting. Although a finite difference scheme over the moving elements can work for this simple one-dimensional setup, it is not pursued here because it is difficult to be generalized into higher dimensions and is believed not to be as accurate as MLS.

In the present scheme, a moving interpolation window centered at each particle center $X_i$ is used to group a number of neighboring particles. The dilatation carried by this group of particles is fed into a weighted least square subroutine to find a best second-order polynomial fit of this data around the position $X_i$. The coefficients of the second-order polynomial are obtained by minimizing the following quadratic function:

$$Q = \sum_{j=1}^{n} W(X_i - X_j) \left( \sum_{k=1}^{m} p_k(X_j)\alpha_k - \theta_j \right)^2, \tag{2.6}$$

where $X_i$ is the position of the central particle, $X_j$ are particles in the interpolation window, $\theta_j$ are the dilatations carried by each particle, $p_k$ are the selected

basis functions (i.e. polynomials), $\alpha_k$ are the unknown coefficients to be solved for, and $W(X_i - X_j)$ is the weighting function. The derivative at the $i^{th}$ particle is computed from the coefficients of the polynomials. The weighting function used in this calculation is an exponential function

$$e^{\frac{(X_i - X_j)^2}{\delta^2}},$$

where $\delta$ is a parameter to control the skewness of the weighting function and has an important effect on the accuracy of MLS fitting. Here, after some numerical experiments, $\delta$ is taken to be 0.1.

When the interpolation window is centered near the left or right boundaries, a part of the window might be outside the domain. The derivatives calculated in this way are not acceptable due to two reasons. First, it is biased to using more data from one side than the other. Second, it disregards the velocity boundary conditions. A technique called particle reflection is used to fix both problems. Particles in the computational domain are reflected outside the domain to fill up the interpolation windows. The velocity boundary condition gets satisfied by forcing the reflected particles to have the same dilatation values as their corresponding particles in the domain. Thus, the extra particles satisfy:

$$\begin{aligned} X_{reflected} &= 2X_{boundary} - X_{inside}, \\ \theta_{reflected} &= \theta_{inside} \end{aligned}$$

Velocity is computed by summing the contributions from individual dilatation elements since according to the definition ($\theta \equiv \nabla \cdot \mathbf{u}$) velocity is an integral of the dilatation. The $i^{th}$ element at position $X_i$ having size $h_i$ and dilatation $\theta_i$ produces the velocity field

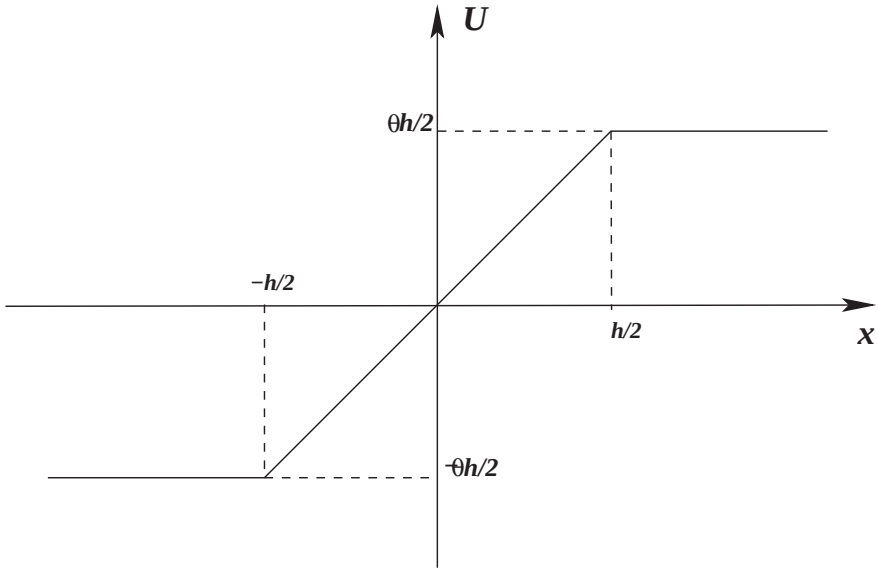



Figure 2.1: Velocity contributed by a one-dimensional dilatation element

$$u_i(x) = \begin{cases} -h_i\theta_i/2, & \text{for} \quad x \leq X_i - h_i/2 \\ \theta_i(x - X_i), & \text{for} \quad |x - X_i| < h_i/2 \\ h_i\theta_i/2, & \text{for} \quad x \geq X_i + h_i/2 \end{cases} \tag{2.7}$$

that is illustrated in Figure 2.1. Clearly, $\theta_i$ is assumed to be distributed uniformly over the element. The total velocity field $u(x)$ is a sum of contributions from all elements plus a potential flow, $u_p$, needed to satisfy the boundary conditions

$$u(x) = \sum_{i=1}^{N} u_i(x) + u_p. \tag{2.8}$$

Since the flow is one-dimensional, $u_p = C$ is a constant. From the above velocity formula, the velocity at the left boundary is

$$u(0) = -\frac{1}{2}\sum_{i=1}^{N} h_i\theta_i + C.$$

This needs to satisfy the boundary condition (2.2). The constant $C$ therefore

must be equal to

$$\frac{1}{2}\sum_{i=1}^{N} h_i\theta_i.$$

The velocity at the right end is

$$u(1) = \frac{1}{2}\sum_{i=1}^{N} h_i\theta_i + C = \sum_{i=1}^{N} h_i\theta_i.$$

In order to satisfy the boundary condition (2.2), it must be that

$$\sum_{i=1}^{N} h_i\theta_i = 0,$$

which is just a numerical approximation to the identity $\int_0^1 \theta dx = 0$.

Figures 2.2 and 2.3 show a comparison of the solutions to the model problem given by the Lagrangian dilatation element method and that of a finite volume method. The particles are represented by $(o)$ in the two figures. The movement of the particles can be easily seen from the figures. Particles near the left-end move away from the boundary with time, so the density of particles is decreasing at the left half of the figures. On the other hand, the distribution of particles at the right half is getting denser. The Reynolds number $Re = 10$ for both cases. The total number, $N = 61$ particles are used for the dilatation element method while the time step $\Delta t = 0.01$. The finite volume calculation uses a $2^{nd}$ order Adamas-Bashforth scheme with 64 evenly spaced cells and time step $\Delta t = 10^{-5}$. Figures 2.2 and 2.3 show that the two solutions closely agree with each other. This helps to establish the effectiveness of the particle method.

## 2.2 Governing equations

Through the above simple example, namely the solution of the viscous Burgers equation, a number of the major features of the Lagrangian dilatation element

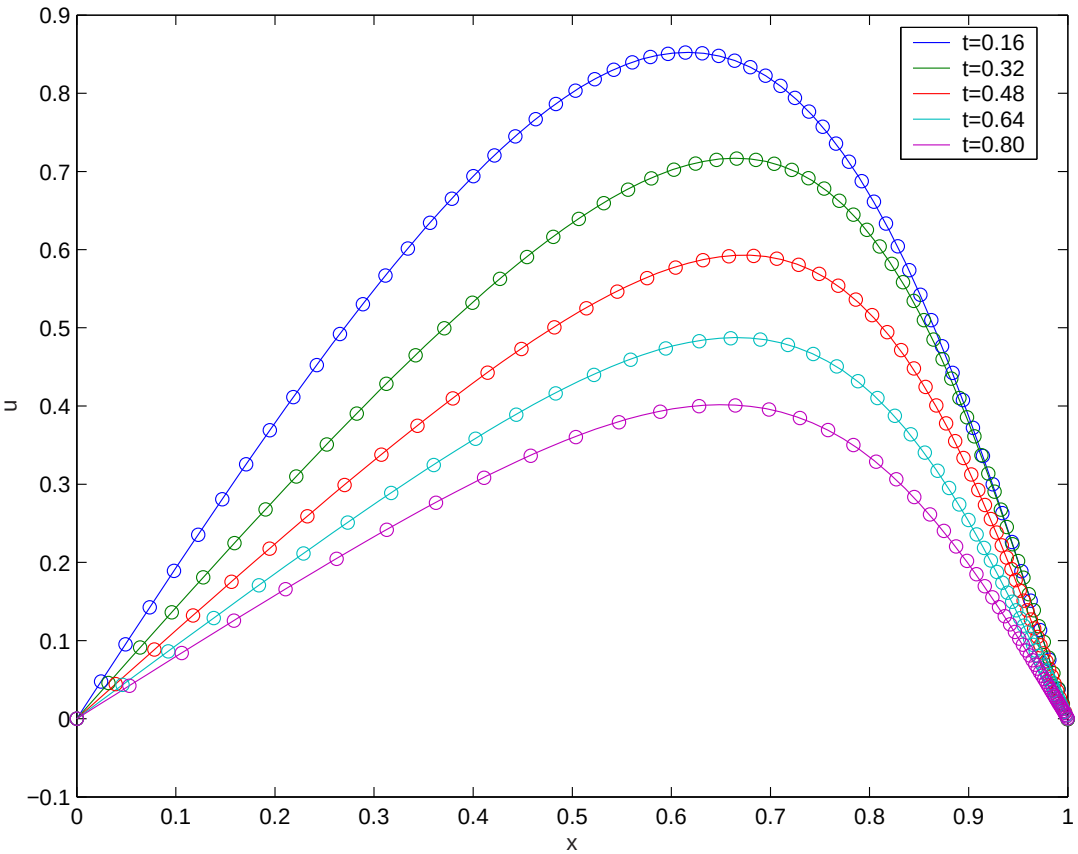



Figure 2.2: Comparison of $u$ at different times; finite volume(—), particle method (o)

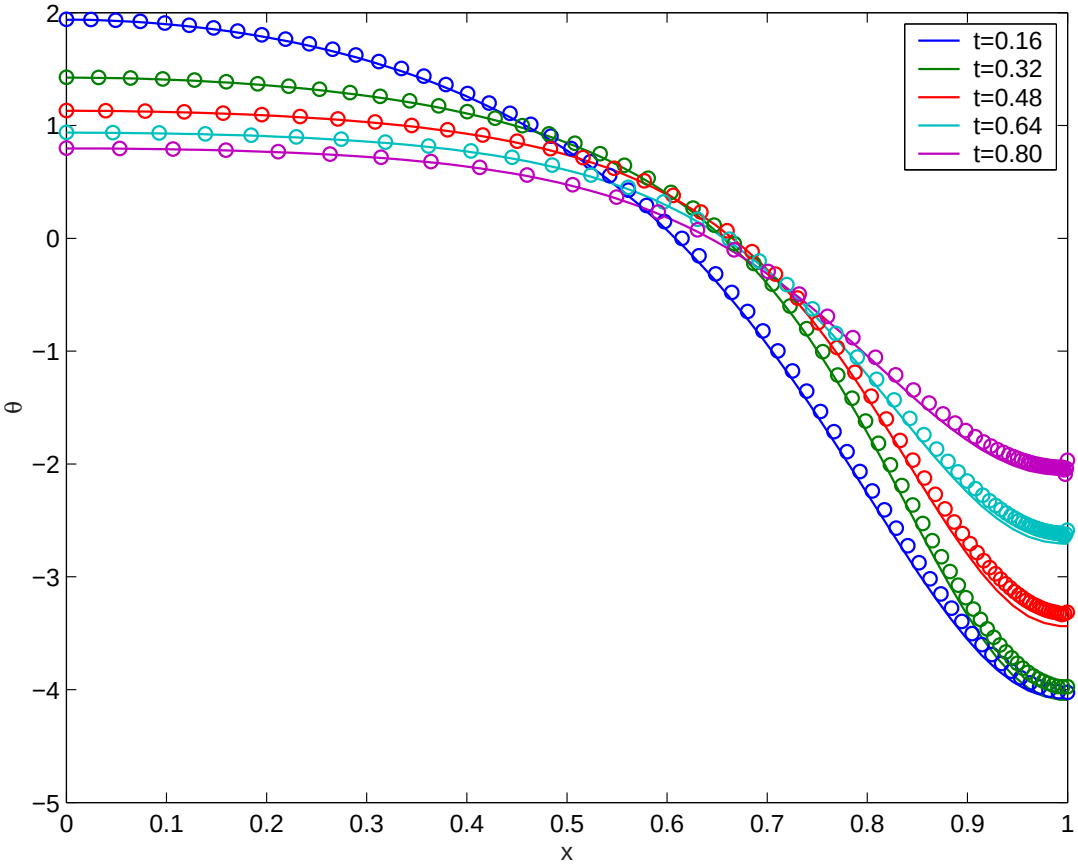



Figure 2.3: Comparison of $\theta$ at different times; finite volume(—), particle method (o)

method were explored. This suggests the following approach for more general cases. For any given problem, first determine a set of suitable Lagrangian governing equations. Then the computational domain should be discretized and represented by a group of Lagrangian particles. The spatial derivatives appearing in the governing equations should be approximated by a scheme purely based on the configuration of free Lagrangian particles. Then a time marching scheme needs to be picked to integrate the governing equations. Most likely the velocity field needs to be constructed from the dilatation and vorticity (if applicable) distributions because the velocity is no longer the primary variable in the framework of a Lagrangian dilatation element method. After completing these steps, the particles are ready to march to the next time step and their positions, volumes and properties such as dilatation, density and temperature can be updated.

Following the above outline, we consider the development of a grid-free method for compressible flow. First the Lagrangian governing equations of compressible flow are presented. For inviscid compressible flow, mass conservation states that

$$\frac{D\rho}{Dt} = -\rho \nabla \cdot \mathbf{u}, \tag{2.9}$$

where $\rho$ is the density and $\mathbf{u}$ is the velocity vector, $D()/Dt = \partial()/\partial t + \mathbf{u} \cdot \nabla()$ is the material derivative. An intention of the vortex method is to use the dilatation and vorticity as primary representations of the velocity field. From the definition of dilatation, the mass conservation equation can be written as

$$\frac{D\rho}{Dt} = -\rho\theta, \tag{2.10}$$

which means that when Lagrangian particles travel with the fluid flow, the density change is equal to the negative product of density and dilatation carried by particles.

Momentum conservation for inviscid compressible flow says

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{\nabla p}{\rho}. \tag{2.11}$$

Taking the divergence of (2.11) yields an equation for dilatation, $\theta$:

$$\frac{D\theta}{Dt} = -\nabla \mathbf{u} : \nabla \mathbf{u} - \nabla \cdot \left( \frac{\nabla p}{\rho} \right). \tag{2.12}$$

Taking the curl of (2.11) yields an equation for vorticity, $\boldsymbol{\omega}$:

$$\frac{D\boldsymbol{\omega}}{Dt} = (\boldsymbol{\omega} \cdot \nabla)\mathbf{u} + \boldsymbol{\omega} \cdot \theta - \nabla \times \left( \frac{\nabla p}{\rho} \right), \tag{2.13}$$

where $p$ is the pressure, and we assume here that $p$ can be related to $T$ and $\rho$ through the perfect gas state equation

$$p = R\rho T,$$

where $R$ is the thermodynamic gas constant.

The energy conservation equation of inviscid compressible flow is

$$\frac{\partial E}{\partial t} + \nabla \cdot (\mathbf{u}(E + p)) = 0, \tag{2.14}$$

where $E$ is the energy per unit volume and consists of the internal and kinetic energies:

$$E = \frac{p}{\gamma - 1} + \frac{1}{2}\rho |\mathbf{u}|^2. \tag{2.15}$$

Using (2.15) and the perfect gas law, (2.14) can be expressed as an equation for $T$:

$$\frac{DT}{dt} = -T\theta(\gamma - 1). \tag{2.16}$$

The governing equations for the inviscid Lagrangian dilatation element method consist of (2.10), (2.12), (2.13), and (2.16). If the flow is assumed to be isentropic,

the Lagrangian governing equations can be further simplified. In this case the density, pressure and temperature are decoupled. There exists a single relationship between any two of the three variables. Therefore, either the mass conservation equation or the energy conservation equation is redundant for isentropic flow. The perfect gas isentropic condition is

$$\mathbf{S} = c_v \ln \frac{p}{\rho^\gamma} = \text{const.}, \tag{2.17}$$

where $c_v$ is the specific heat of a perfect gas and $\gamma \equiv c_p/c_v$ is the ratio of specific heats ($\gamma = 1.4$ for air). From the isentropic condition (2.17) and the equation of state, it is straightforward to work out the following relation between $p$, $\rho$, and $T$:

$$\frac{p}{p_0} = \left(\frac{\rho}{\rho_0}\right)^\gamma = \left(\frac{T}{T_0}\right)^{\frac{\gamma}{\gamma-1}}, \tag{2.18}$$

where $p_0$, $\rho_0$ and $T_0$ are the initial pressure, density and temperature, respectively. For isentropic flow,

$$\nabla \cdot \left(\frac{\nabla p}{\rho}\right) = \frac{R\gamma}{\gamma - 1}\Delta T, \tag{2.19}$$

and

$$\nabla \times \left(\frac{\nabla p}{\rho}\right) = 0. \tag{2.20}$$

So, the governing equations become

$$\frac{D\theta}{Dt} = -\nabla\mathbf{u}{:}\nabla\mathbf{u} - \frac{R\gamma}{\gamma - 1}\Delta T \tag{2.21}$$

$$\frac{D\boldsymbol{\omega}}{Dt} = (\boldsymbol{\omega} \cdot \nabla)\mathbf{u} + \boldsymbol{\omega} \cdot \theta \tag{2.22}$$

$$\frac{DT}{Dt} = -T\theta(\gamma - 1). \tag{2.23}$$

The density equation is redundant and has been dropped.

### 2.2.1 1D governing equations

Since vorticity does not exist for one-dimensional problems, the vorticity equation is dropped for 1D applications. Specializing the previous relations to 1D flow, it is found that governing equations take the following form

$$\frac{D\rho}{Dt} = -\rho\theta \tag{2.24}$$

$$\frac{D\theta}{Dt} = -\theta^2 + R\left(\frac{\rho_{\mathrm{x}}^2 \mathrm{T}}{\rho^2} - \frac{\mathrm{T}_{\mathrm{x}}\rho_{\mathrm{x}}}{\rho} - \frac{\mathrm{T}\rho_{\mathrm{xx}}}{\rho} - T_{xx}\right) \tag{2.25}$$

$$\frac{DT}{Dt} = -T\theta(\gamma - 1). \tag{2.26}$$

Here, $\rho$ and $T$ have been non-dimensionalized by their initial values $\rho_0$ and $T_0$ and length have been scaled by a characteristic length $L$. With the assumption of isentropic flow, the governing equations further simplify to

$$\frac{D\theta}{Dt} = -\theta^2 - \frac{1}{\gamma - 1}T_{xx} \tag{2.27}$$

$$\frac{DT}{Dt} = -T\theta(\gamma - 1), \tag{2.28}$$

The velocity scale is chosen to be the initial sound speed $c_0 = \sqrt{\gamma R T_0}$ and time is scaled by $L/c_0$.

### 2.2.2 2D governing equations

The application of this scheme to 2D inviscid compressible flow is particularly important because in this case all the difficulties encountered will be present when this scheme is extended to higher dimensions. Thus, if the Lagrangian dilatation element method can be successfully applied in solving a 2D compressible flow problem, then it can likely be straightforwardly generalized into higher dimensional applications. Specializing (2.10), (2.12), (2.13), and (2.16), the 2D

governing equations are

$$\frac{D\rho}{Dt} = -\rho\theta \tag{2.29}$$

$$\frac{D\theta}{Dt} = -(u_x^2 + v_y^2 + 2u_y v_x) + \frac{\rho_x p_x}{\rho^2} - \frac{p_{xx}}{\rho} + \frac{\rho_y p_y}{\rho^2} - \frac{p_{yy}}{\rho} \tag{2.30}$$

$$\frac{D\omega}{Dt} = -\theta\omega + \frac{\rho_x p_y - \rho_y p_x}{\rho^2} \tag{2.31}$$

$$\frac{DT}{Dt} = -T\theta(\gamma - 1). \tag{2.32}$$

Applying the same scaling as in 1D and moreover assuming flow to be isentropic and initially vorticity free, the non-dimensionalized governing equations are

$$\frac{D\theta}{Dt} = -(u_x^2 + v_y^2 + 2u_y v_x) - \frac{1}{\gamma - 1}\Delta T \tag{2.33}$$

$$\frac{DT}{Dt} = -T\theta(\gamma - 1). \tag{2.34}$$

### 2.2.3 Equations for quasi-1D nozzle

We are later concerned with a quasi-one-dimensional model of compressible flow in a variable area duct as shown in Figure 5.2. In this, flow properties are assumed to vary only in the streamwise direction, $x$, and in time, $t$. In this, we assume that the variation in the cross-sectional area of the duct, $A(x)$, is relatively small so that the assumption of one-dimensionality is reasonable. Equations expressing mass, momentum and energy conservation for these conditions may be derived from a control volume analysis in which the streamwise velocity is taken to be constant on cross-sections [2]. The result for the density $\rho$ is

$$\frac{D\rho}{Dt} = -\rho(\theta + u\alpha), \tag{2.35}$$

where $\alpha \equiv \frac{dA(x)/dx}{A(x)}$ is non-zero only in regions where the duct changes size.

A similar consideration of the momentum balance yields

$$\frac{Du}{Dt} = -\frac{1}{\rho}\frac{\partial p}{\partial x}. \tag{2.36}$$

In view of our intention to use $\theta$ as the primary representation of the velocity field, we take a spatial derivative of (2.36) and rewrite it in terms of dilatation, namely,

$$\frac{D\theta}{Dt} = -\theta^2 - \frac{\partial}{\partial x}\big(\frac{1}{\rho}\frac{\partial p}{\partial x}\big). \tag{2.37}$$

Finally applying an energy balance yields an equation for the internal energy, $e$, in the form

$$\frac{De}{Dt} = -\frac{p}{\rho}(\theta + u\alpha), \tag{2.38}$$

which becomes an equation for the temperature $T$ after assuming further that $e = c_v T$.

$\rho$, $p$ and $T$ are assumed to be scaled by their upstream equilibrium values, so that in such locales $\rho = 1$, $p = 1$ and $T = 1$ where temperature is scaled by $T_0$. Lengths are scaled by a characteristic length $L = \sqrt{A_0}$ with $A_0$ denoting the duct area in the upstream, constant region, so that the scaled area $A(x) = 1$ away from the contraction. It is also convenient to choose the velocity scale as the equilibrium sound speed $c_0 = \sqrt{\gamma R T_0}$ so that $u$, after scaling, is a Mach number. With these choices, time is scaled by $L/c_0$.

Under the further assumption of the isentropic condition (2.17), the non-dimensionalized governing equations of the quasi-1D nozzle flow can be simplified to a system for just $\theta$ and $T$ in the form

$$\frac{D\theta}{Dt} = -\theta^2 - \frac{1}{\gamma - 1}\frac{\partial^2 T}{\partial x^2}, \tag{2.39}$$

$$\frac{DT}{Dt} = -(\gamma - 1)T(\theta + u\alpha). \tag{2.40}$$

## 2.3 General considerations

The basic computational elements of the Lagrangian dilatation element method are particles that move with the flow. Assume at a certain time $t_n$ a typical particle located at position $\mathbf{X_i}$ has volume $V_i$ and carries dilatation, $\theta_i$, vorticity, $\omega_i$, temperature, $T_i$ and density, $\rho_i$. The Lagrangian particles convect with the fluid motion and their properties evolve with time according to their corresponding governing equations. Besides these relations, there is an additional set of equations representing the movement and distortion of the computational Lagrangian particles. Particles are generally convected downstream passively. Therefore their motion satisfies

$$\frac{D\mathbf{X_i(t)}}{Dt} = \mathbf{u}(\mathbf{X_i(t)}, \mathbf{t}). \tag{2.41}$$

From the Reynolds transport theorem,

$$\frac{d}{dt}\int_{V(t)_i} f dV = \int_{V(t)_i} \frac{\partial f}{\partial t} dV + \int_{V(t)_i} \nabla \cdot (f\mathbf{u}) dV. \tag{2.42}$$

Letting $f = 1$ and assuming the particle volume $V_i$ is sufficiently small, the volumetric distortion of the particles obeys, approximately:

$$\frac{dV_i(t)}{dt} = V_i(t)\theta_i. \tag{2.43}$$

Even though the velocities are not the primary variables in the governing equations, they must be known at every time step to determine the movement of the Lagrangian particles. From the Helmholtz velocity decomposition law, the velocities $\mathbf{u}$ can be determined from the $\theta$ and $\omega$ fields after using appropriate boundary conditions. For the particle method this involves summing over the contributions from the $N$ Lagrangian elements in the calculation. For direct summation to recover the velocity field requires $O(N^2)$ work. This makes the

practical implementation of the scheme impossible when the number of particles exceeds, approximately $10,000$, which is a very moderate number for higher dimensional computations. The Fast Multipole Method (FMM) can reduce the cost of computing velocities to $O(N \log N)$ and is thus necessary for this research. The details of the velocity recovery will be discussed in the next chapter.

A big advantage of the Lagrangian dilatation element method is that by being grid free, it can be easily applied to complicated problems with complex geometries. At the same time it has the difficulty of evaluating spatial derivatives in the governing equations because there is no fixed spatial configuration between the particles. In this research, the spatial derivatives are computed using a locally weighted moving least-square fit. For each particle, least-square fitting with a polynomial base is performed on the collection of elements lying in a local domain around the center of the particle. The spatial derivatives are analytically computed after knowing the coefficients of the fitting polynomial. The details will be discussed in Chapter 4.

It is evident that when particles are located near boundaries, the collection of elements in the least-square fitting is biased towards the interior of the computational domain. The part of the collection window lying outside the domain is empty. This decreases the accuracy of the least-square fitting and the computed spatial derivatives. To get a collection of symmetrically placed data for fits near boundaries it is necessary to create artificial data outside of the computational domain. This procedure must be considered together with the boundary conditions. This will be discussed in Chapter 5.

How to implement inflow and outflow conditions is not evident for Lagrangian schemes. Because Lagrangian computational elements are convected downstream

as time evolves, how to keep the computational domain well filled with comparably sized elements is an important issue. During the transient development of the flow field in some cases, characteristic waves carrying dilatation and vorticity travel across the inlet and outlet boundaries. Thus, despite our interest in the computational region only, it is still necessary to take into account the presence of dilatation and vorticity outside the computational domain when constructing the velocity field. This must be done without explicitly introducing elements to track the outside dilatation and vorticity. Otherwise, there will be no end to the extent of the computational domain. Chapter 5 will discuss and illustrate how such problems might be solved by using the wave properties of the governing equations.

# Chapter 3

# Velocity Evaluation

Although the example flow problems considered in this research are vorticity free, vorticity is still considered in the evaluation of velocity field because of two concerns. First, vorticity does not bring any special technical difficulty and extra work to the velocity recovery procedure. Second, it makes the method easier to generalize for viscous compressible flow. Dilatation and vorticity instead of velocity are the primary variables in the governing equations of the Lagrangian dilatation element method. However, the velocity must be known at each time step not only to convect Lagrangian particles but also to integrate the dilatation governing equation. For one-dimension velocity recovery is as simple as shown in the Burger's equation case in Chapter 2. Some details related to the application of quasi-1D nozzle flow are discussed in section 3.1. Recovering the velocity field from dilatation and vorticity [4] in higher dimensions is a classical fluid problem discussed in section 3.2. Then in 3.3 we show how the Fast Multipole Method (FMM) developed by Greengard and Rokhlin [40] can reduce the cost of computing velocities down to $O(N \log N)$, thereby making practical computation

possible. Since the velocity computed solely from dilatation and vorticity will not generally satisfy velocity boundary conditions on a bounded domain, a potential velocity component must be added to enforce the velocity boundary condition. The details of this and other aspects of the velocity recovery are discussed in this chapter.

## 3.1 Velocity recovery for quasi-1D nozzle flow

The basic procedure to recover the velocity in the quasi-1D nozzle flow is similar to that shown in the case of Burger's equation in Chapter 2. Equation (2.8) is used to calculate the velocities contributed from all the dilatation elements. However (2.8) needs to satisfy the far field condition $u(-\infty) = u_0$. This means that the constant $u_p$, representing a potential flow, must be computed so as to satisfy the boundary condition. In this case, Eq. (2.8) becomes

$$u(x) = u_0 + \sum_{i=1}^{N} (u_i(x) + h_i\theta_i/2), \tag{3.1}$$

However there is nothing in the argument leading to (3.1) that limits the sum in that equation to dilatation elements lying in the region of interest between $x = x_l$ and $x = x_r$. In fact, all elements containing non-zero dilatation must be included in this relation. In particular, as will be seen below, during the transient development of the duct flow, waves carry $\theta$ out toward $x = \pm\infty$ from the contraction. Thus, despite our interest in the flow in the computational region only, it is still necessary to take into account the presence of dilatation outside the computational domain. This must be done without explicitly providing elements to keep track of this part of dilatation field since otherwise there would be no end to the extent of the computational domain.

For $x$ limited to the computational region (i.e., $x_l \leq x \leq x_r$), it is evident that all elements for which $x_i > x_r$ makes no contribution to the sum in (3.1) since $u_i(x) + h_i\theta_i/2 = 0$ in such cases (see Figure 2.1). Moreover, the contribution of elements for which $x_i < x_l$ can be combined together into a term approximating

$$\Delta_l \equiv \int_{-\infty}^{x_l} \theta(x)dx, \tag{3.2}$$

thereby yielding the result

$$u(x) = u_0 + \Delta_l + \sum_{i=1}^{N}(u_i(x) + h_i\theta_i/2), \tag{3.3}$$

where it is to be understood here and henceforth that the summation in (3.3) is only over the elements lying within the computational domain, and $x$ satisfies $x_l \leq x \leq x_r$. Defining,

$$\Delta_r \equiv \int_{x_r}^{+\infty} \theta(x)dx, \tag{3.4}$$

for the isentropic flow considered here, it is also the case that $u(+\infty) = u_0$ since the duct has the same uniform area both upstream and downstream of the constriction. The integral of the dilatation over the whole domain is equal to zero. Therefore

$$\Delta_l + \sum_{i=1}^{N} h_i\theta_i + \Delta_r = 0, \tag{3.5}$$

in which case (3.3) can be rearranged as

$$u(x) = u_0 + \frac{1}{2}\Delta_l + \sum_{i=1}^{N} u_i(x) - \frac{1}{2}\Delta_r, \tag{3.6}$$

which is the velocity relation that will be used in the later calculation. $\Delta_l$ and $\Delta_r$ depend only on $\theta$ outside the computational domain, and in fact are related to the velocities at the left and right boundaries, respectively, through the identities

$$u_l \equiv u(x_l) = u_0 + \Delta_l, \tag{3.7}$$

and

$$u_r \equiv u(x_r) = u_0 - \Delta_r. \tag{3.8}$$

For (3.6) to be useful in determining velocities in the numerical scheme, the changes of $\Delta_l$ and $\Delta_r$ in time must be determined over the course of the flow evolution. In fact, this information can be found using the wave structure of the equations as will be shown in the course of our presentation of the algorithm in section 5.3.

## 3.2 Helmholtz velocity decomposition

From this section and later on, we start to discuss the velocity recovery in higher dimensions. The dilatation $\theta$, vorticity $\boldsymbol{\omega}$ and the boundary normal velocity $\mathbf{V_n}$ in a domain $\mathcal{V}$ are given by:

$$\nabla \cdot \mathbf{V} = \theta, \tag{3.9}$$

$$\nabla \times \mathbf{V} = \boldsymbol{\omega}, \tag{3.10}$$

$$\mathbf{V} \cdot n = \mathbf{V}_b \cdot n \tag{3.11}$$

where $\mathcal{V}$ is the domain, $\mathcal{S}$ is the boundary of $\mathcal{V}$ and $\mathbf{V}_b$ is the boundary velocity. The goal is to compute the velocity inside of domain $\mathcal{V}$.

According the Helmholtz velocity decomposition law, any velocity vector $\mathbf{V}$ in an unbounded domain can be decomposed into two parts: an irrotational component $\mathbf{V}_1$ and a solenoidal component $\mathbf{V}_2$:

$$\mathbf{V} = \mathbf{V}_1 + \mathbf{V}_2, \tag{3.12}$$

where $\mathbf{V}_1 = \nabla\phi$ and $\mathbf{V}_2 = \nabla \times \mathbf{A}$, $\phi$ is a scalar potential and $\mathbf{A}$ is a vector potential.

Assume velocity $\mathbf{V}$ satisfies (3.9) and (3.10). Taking the divergence and the curl of $\mathbf{V}$ leads to:

$$\nabla \cdot \mathbf{V}_1 = \nabla^2 \phi = \theta, \tag{3.13}$$

$$\nabla \times \mathbf{V}_2 = \nabla^2 \mathbf{A} = -\boldsymbol{\omega}. \tag{3.14}$$

Equations (3.13) and (3.14) are Poisson equations. They can be solved in an unbounded domain using a Green's function in the form

$$\phi = -\frac{1}{2\pi} \int_{\mathcal{V}} \theta' \ln |\mathbf{r} - \mathbf{r}'| dV' \tag{3.15}$$

$$\mathbf{A} = \frac{1}{2\pi} \int_{\mathcal{V}} \boldsymbol{\omega}' \ln |\mathbf{r} - \mathbf{r}'| dV', \tag{3.16}$$

where $\mathcal{V} \in \mathbb{R}^2$. Then the velocity contributions $\mathbf{V}_1$ and $\mathbf{V}_2$ are:

$$\mathbf{V}_1 = \frac{1}{2\pi} \int_{\mathcal{V}} \frac{\theta(\mathbf{r} - \mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|^2} dV' \tag{3.17}$$

$$\mathbf{V}_2 = \frac{1}{2\pi} \int_{\mathcal{V}} \frac{\boldsymbol{\omega} \times (\mathbf{r} - \mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|^2} dV'. \tag{3.18}$$

Equation (3.18) is referred to as the Biot-Savart law. The velocity as the sum of (3.17) and (3.18) will not generally satisfy velocity boundary conditions. A potential velocity field $\mathbf{V}_3$ must be added to force $\mathbf{V}$ to satisfy the velocity boundary condition, namely equation (3.11). This will be discussed in Section 3.4.

If the whole domain is discretized into a group of rectangular elements with centers $(x_j, y_j)$ and it is assumed the dilatation and vorticity distributions are constant on the rectangular elements, then (3.17) and (3.18) can be integrated yielding

$$\mathbf{V}_1 = \frac{1}{2\pi} \sum_{j=1}^{N} \Theta_j \frac{(x - x_j, y - y_j)}{(x - x_j)^2 + (y - y_j)^2} \tag{3.19}$$

$$\mathbf{V}_2 = -\frac{1}{2\pi} \sum_{j=1}^{N} \Omega_j \frac{(y - y_j, x_j - x)}{(x - x_j)^2 + (y - y_j)^2}, \tag{3.20}$$

where $N$ is the number of particles, and

$$\Theta_j = \theta_j \Delta A_j \tag{3.21}$$

$$\Omega_j = \omega_j \Delta A_j, \tag{3.22}$$

$\Theta_j$ and $\Omega_j$ (the circulation) are the total strengths of dilatation and vorticity each particle carries and $A_j$ is the area of the $j^{th}$ element.

## 3.3 Fast Multipole Method

It is convenient to use complex variables in 2D to describe physical coordinates in this analysis. Thus we identify $C : \mathbf{x} = (x, y) \rightarrow z = x + y\hat{i}$ and the $x$ and $y$ components of velocity can be written as the real and imaginary parts of the function $F = u - v\hat{i}$ in the complex plane. Then equations (3.19) and (3.20) can be reformulated into:

$$\mathbf{V}_1 = \frac{1}{2\pi} \sum_{j=1}^{N} \Theta_j \frac{1}{z - z_j} \tag{3.23}$$

$$\mathbf{V}_2 = \frac{1}{2\pi\hat{i}} \sum_{j=1}^{N} \Omega_j \frac{1}{z - z_j}, \tag{3.24}$$

where $\mathbf{V}_1$, $\mathbf{V}_2$ are now complex. Hence, $F$, for the total contribution from dilatation and vorticity elements, takes the form:

$$\begin{aligned} F(z) &= \sum_{j=1}^{N} \frac{1}{2\pi} (\Theta_j - \Omega_j \hat{i}) \frac{1}{z - z_j}, \\ &= \sum_{j=1}^{N} \Gamma_j \phi(z, z_i), \end{aligned} \tag{3.25}$$

where $\Gamma_j = \frac{1}{2\pi}(\Theta_j - \Omega_j \hat{i})$ and $\phi(z, z_i) = \frac{1}{z - z_j}$ .

A quad-tree data structure is necessary for the implementation of the 2D FMM. Two data hierarchies with $N - 1$ levels ($level = N, ..., 2$) have to be built.

One is a source hierarchy for the record of discrete particle positions and one is a target hierarchy for the evaluation of point positions. A quad-tree structure is constructed by forming a unit square box, subdividing it into four child boxes, subdividing each child box recursively, and so on until some level of refinement is reached. At each level, there are $2^{level}$ boxes. Each box is determined by a unique number $n$ at $level = l$, denoted as $box(n, l)$. $level = i$ is the parent of $level = i - 1$. Meanwhile, $level = i - 1$ is called the child of $level = i$. Boxes can have parents, children, neighbors and siblings (see Fig. 3.1). A box has four children and can have a maximum nine neighbors including itself in the 2D quad-tree structure. Those adjacent neighbor boxes are called siblings.

Four domains are defined, for each $box(n, l)$ as shown in Fig. 3.1.

- $E1(n, l)$ denotes spatial points inside the $box(n, l)$.
- $E2(n, l)$ denotes spatial points inside the $box(n, l)$ and its neighbors.
- $E3(n, l)$ denotes spatial points outside the $box(n, l)$ and its neighbors.
- $E4(n, l)$ denotes spatial points inside the parent box's $E2$ domain from which the $E2$ domain of $box(n, l)$ is excluded (see Fig 3.1).

The FMM algorithm consists of three steps: upward pass, downward pass plus a final summation(see Fig. 3.2).

- UPWARD PASS

  1. At the finest level ($level = N$), build the far-field expansion (S expansion) for sources inside of each non-empty box at the center of that box.

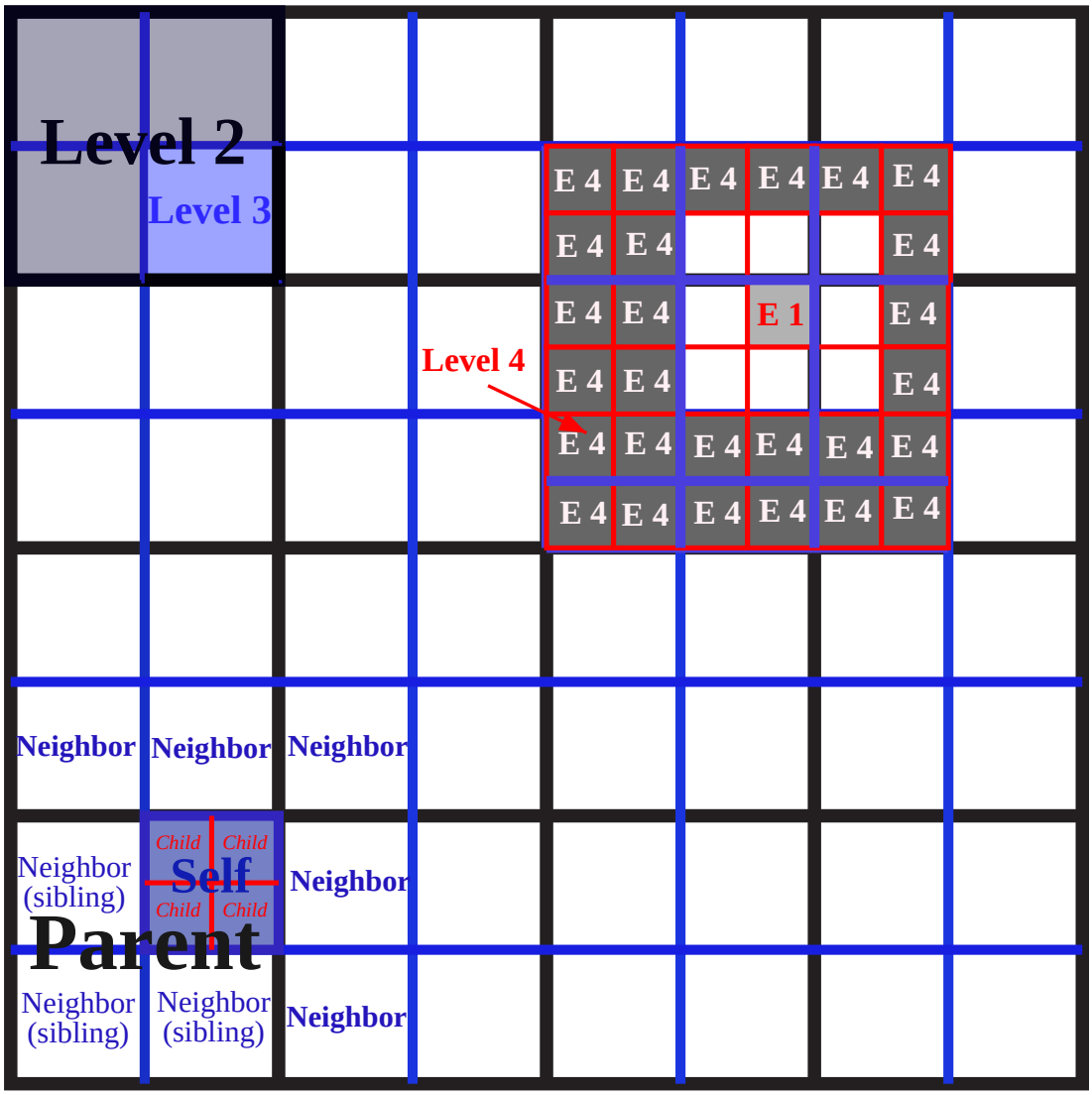



Figure 3.1: The illustration of computational boxes used by FMM with a maximum $level = 4$

2. From level $i = N-1, ..., 2$ recursively use $S|S$ translation to translate the $S$ expansions of each box at $level = i$ up to the center of their parent's box at $level = i-1$.

- DOWNWARD PASS

  1. At the coarsest $level = 2$, apply $S|R$ translation to translate the $S$ expansions of each box into $R$ expansions at the centers of their $E4$ neighborhood.
  2. From level $i = 3, ..., N$, there are two steps: (i) use $R|R$ translation to translate the $R$ expansions from their parents $level = i-1$ down to the box center at $level = i$; and (ii) apply $S|R$ translation to translate the $S$ expansions of each box computed in the upward pass into the $R$ expansions at the centers of their $E4$ neighborhood and sum up with the $R$ expansions from (i).

- FINAL SUMMATION (AT THE FINEST LEVEL $N$)

  1. Use the direct summation to evaluate equation (3.25) at the $E2$ neighborhood of each evaluation position.
  2. Sum up the contributions of the $R$ expansion at the $E3$ neighborhood of each evaluation position.
  3. Sum up the results from the previous two steps.

### Implementation Details

Two sets of hierarchies with tree data structure are used: one for source points and the other for evaluation points. The FMM algorithm automatically skips the empty boxes to save the computation cost.

First, prepare the hierarchical data structure:

- Scale and shift source and evaluation data to fit into a unit box.
- Compute the box indexes corresponding to each source and evaluation point at the finest level using the bit-interleaving to convert the 2D data set into a set of integers.
- Sort the box indexes for the source and target points at the finest level. Then reorder the dilatation and vorticity arrays according to the order of the source points; reorder the evaluation position array according to the order of target points.
- Merge the repeated indexes of the sorted tables to build both source and evaluation hierarchies separately at the finest level and record the repeated numbers for each box. Two arrays are saved: the box ID array and the array saving the number of points in each box corresponding to the box ID array.
- Compute the parent box indexes recursively up to the *level* 2 (no sorting is needed because it is already ordered at the children level) while do the merging process as above with the repeated numbers recorded at each box.

The source and evaluation hierarchies should be available after the above data preparation is done. The average cost of this procedure is proportional to $O(N \log N)$ using the fast sorting and merging algorithm. In the FMM algorithm the box ID tables need to be searched frequently to find if a specified box exists in that level. A fast binary search code is used which is proportional to $O(\log N)$ operations. The FMM implementation consists of an upward pass, a downward pass, and final

summation, as described earlier in this section. The source hierarchy and target hierarchy are used in the following order in the algorithm. The source hierarchy is used to compute the coefficients of $S$ and $S|S$ expansions in the upward pass. The target hierarchy is used to compute the coefficients of $R|R$ expansions in the downward pass. Both hierarchies are used to compute the coefficients of $S|R$ expansions.

After the coefficients of the $R$ expansions at the finest level are ready, the final summation is executed that sums up the contributions from $E2$ neighborhoods by the direct summation and the contributions from $E3$ regions computed from multiplying the $R$ expansions by the source strengths.

### 3.3.1 S, S|S, S|R and R|R expansions

It is necessary to analytically derive the formulas for the expansions involved in the FMM algorithm. Fig. 3.2 shows the flow chart of these expansions.

#### S expansion

If a cluster of particles are contained in a box $A$ centering at $z_A$ such that $|z-z_A| > 2R$ and $|z_i - z_A| < R$, the $S$ expansion of $\phi(z, z_i)$ in (3.25) takes the following form

$$\begin{aligned} \phi(z, z_i) &= \frac{1}{z - z_i} \\ &= \sum_{k=0}^{\infty} b_k(z_i, z_A) S_k(z - z_A) \\ &= \sum_{k=0}^{p} b_k(z_i, z_A) S_k(z - z_A) + \text{Error}, \end{aligned} \tag{3.26}$$

where $b_k(z_i, z_A) = (z_i - z_A)^k$ and $S_k(z - z_A) = (z - z_A)^{-k-1}$.

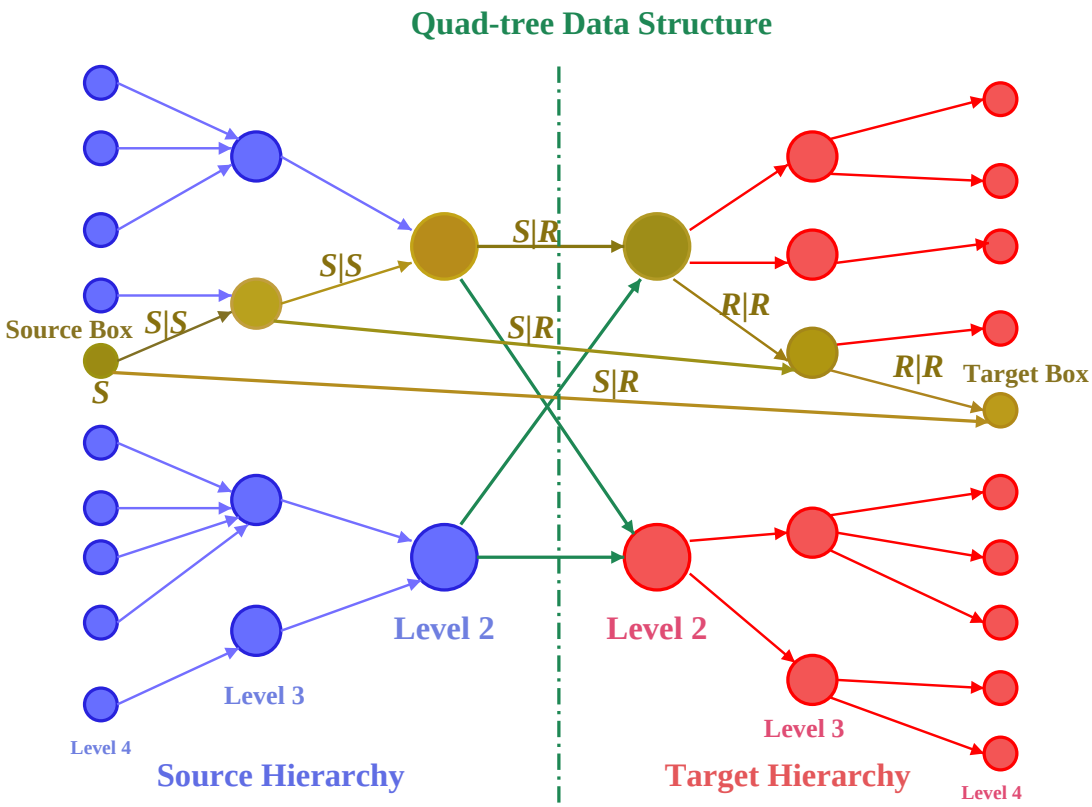



Figure 3.2: A quad-tree data structure and the flow chart of S, S|S, S|R and R|R expansions in FMM

From the above expansion, the truncated error can be bounded as

$$\begin{aligned} Error &= \sum_{k=p+1}^{\infty} \frac{(z_i - z_A)^k}{(z - z_A)^{k+1}} \\ &< \frac{1}{R} \sum_{k=p+1}^{\infty} \left(\frac{1}{2}\right)^{k+1} \\ &< \frac{1}{R} \left(\frac{1}{2}\right)^{p+1} < \epsilon. \end{aligned} \tag{3.27}$$

From equation (3.27), the truncation number $p$ can be computed after a specified error $\epsilon$ is determined.

### $S|S$ expansion

If the $S$ expansion of $F(z)$ at the center of a box A, $z_A$, is given by (3.26), then the expansion of $F(z)$ at the center of A's parent P, $z_P$, can be expressed with $S|S$ translation as

$$S_n(z - z_P) = \sum_{m=0}^{\infty} (S|S)_{mn}(t) S_m(z - z_A), t = z_A - z_P, \tag{3.28}$$

where

$$\begin{aligned} S_n(z - z_P) &= (z - z_P)^{-n-1} \\ &= (z - z_A)^{-n-1} \left[1 - \frac{z_P - z_A}{z - z_A}\right]^{-n-1} \\ &= \sum_{m=0}^{\infty} \frac{(-1)^m (m+n)!}{m!n!} (z_A - z_P)^m (z - z_A)^{-n-m-1} \\ &= \sum_{m=n}^{\infty} \frac{(-1)^{m-n} m!}{n!(m-n)!} t^{m-n} S_m(z - z_A). \end{aligned} \tag{3.29}$$

So the $S|S$ translation operator takes the following form

$$(S|R)_{mn}(t) = \begin{cases} 0, & m < n \\ \frac{(-1)^m (m+n)!}{m!n!t^{m+n+1}}, & m \geq n \end{cases}$$

### $S|R$ expansion

If the $S$ expansion (3.26) of $F(z)$ at the center of a box A, $z_A$, is given, the expansion of $F(z)$ at the center of a box $z_B$ which is in A's $E3$ neighborhood can be calculated from the $S|R$ translation:

$$R_n(z - z_B) = \sum_{m=0}^{\infty} (S|R)_{mn}(t) S_m(z - z_A), t = z_A - z_P. \tag{3.30}$$

The $S|R$ translation operator is

$$(S|R)_{mn}(t) = \frac{(-1)^m (m+n)!}{m! n! t^{m+n+1}}, \qquad m, n = 0, 1, 2, \cdots, p. \tag{3.31}$$

### $R|R$ expansion

If the $R$ expansion of $F(z)$ at the center of a box B, $z_B$, is given, the expansion of $F(z)$ at the center of A's children C, $z_C$, can be calculated from the $R|R$ translation:

$$R_n(z - z_C) = \sum_{m=0}^{\infty} (R|R)_{mn}(t) R_m(z - z_B), t = z_B - z_C, \tag{3.32}$$

where

$$\begin{aligned} R_n(z - z_C) &= (z - z_C)^n \\ &= (z - z_B)^n \left[ 1 - \frac{z_C - z_B}{z - z_B} \right]^n \\ &= \sum_{m=0}^{\infty} \frac{n!}{m!(n-m)!} (z_B - z_C)^{n-m} (z - z_B)^m \\ &= \sum_{m=0}^{\infty} \frac{n!}{m!(n-m)!} t^{m-n} R_m(z - z_B). \end{aligned} \tag{3.33}$$

The $R|R$ translation operator is:

$$(R|R)_{mn}(t) = \begin{cases} 0, & m > n \\ \frac{n!}{m!(n-m)!} t^{m-n}, & m \leq n \end{cases}$$

### 3.3.2 Truncation number, grouping/clustering parameter and optimization

There are two main objectives in the optimization of the FMM. One is cost, the other is error. Assume there are $N$ source points and $M$ evaluation points. $L$ is the finest level, $p$ is the truncation number of the $S$ expansions, $d$ is the dimensionality, and $P_4(d)$ and $P_2(d)$ are the powers of $E4$ and $E2$ neighborhoods. The clustering parameter is $s$, which represents the maximum number of source points contained in the box at the finest level.

The cost of the FMM can be evaluated according to the following formula [42]:

$$\text{Cost} = (M+N)p + 2^d(P_4(d)+2)\frac{N}{s}p^2 + P_2(d)sM. \tag{3.34}$$

The clustering parameter $s$ can be optimized to minimize the cost of FMM:

$$\frac{\partial \text{Cost}}{\partial s} = 0. \tag{3.35}$$

The optimal $s_{opt}$ then can be expressed:

$$\begin{aligned} s_{opt} &\sim \left[\frac{2^d(P_4(d)+2)\alpha^2 \log N}{P_2(d)M}\right]^{\frac{1}{2}} \\ &\sim \left[\frac{N \log^2 N}{M}\right]^{\frac{1}{2}} \\ &\sim \log N. \end{aligned} \tag{3.36}$$

So, the cost of FMM can be optimized by choosing the optimal clustering parameter: $s_{opt} \sim \log N$. The optimal FMM cost is proportional to $O(N \log N)$.

Because $S|S$ and $R|R$ translations have not introduced any error, then the error is completely due to the $S$ expansion and $S|R$ translation. The total error

is bounded by (see [42]):

$$\text{Error} < \frac{3N}{\rho}\left(\frac{r}{r+\rho}\right)^p, \tag{3.37}$$

where $r$ and $\rho$ are geometrical parameters related to the maximum level of the quad-tree structure. In 2D applications $d = 2$, and [42] shows that $\rho = (2 - \sqrt{2})2^{-L}$, $r = 2^{-L-1}\sqrt{2}$, where $L$ is the maximum level of the quad-tree. The relation between the finest level $L$, grouping parameter $s$, and the number of source points $N$ is

$$L = \log_2 \sqrt{\frac{N}{s}}.$$

Thus,

$$\begin{aligned} \text{Error} &< 3\frac{2^L N}{2-\sqrt{2}}\left(\frac{\sqrt{2}}{4-\sqrt{2}}\right)^p \\ &< 5.2N(0.6)^p 2^L \\ &= 5.2N(0.6)^p\sqrt{\frac{N}{s}} < \epsilon. \end{aligned} \tag{3.38}$$

In order to satisfy the specified error $\epsilon$, the truncation number $p$ must satisfy

$$p > \frac{1}{\log(1/0.6)} \log \frac{5.2N^{3/2}}{\epsilon\sqrt{s}}. \tag{3.39}$$

Since the clustering parameter $s$ and the maximum level $L$ are optimized to minimize the cost of the FMM, the truncation number $p$ therefore can be completely determined from (3.39).

### 3.3.3 Comparisons of error and CPU time between FMM and direct summation

Figure 3.3 shows how the errors vary with the truncation number $p$ in the case $N = 1000$, and Max Level = 4. Figure 3.4 shows how the errors vary with the

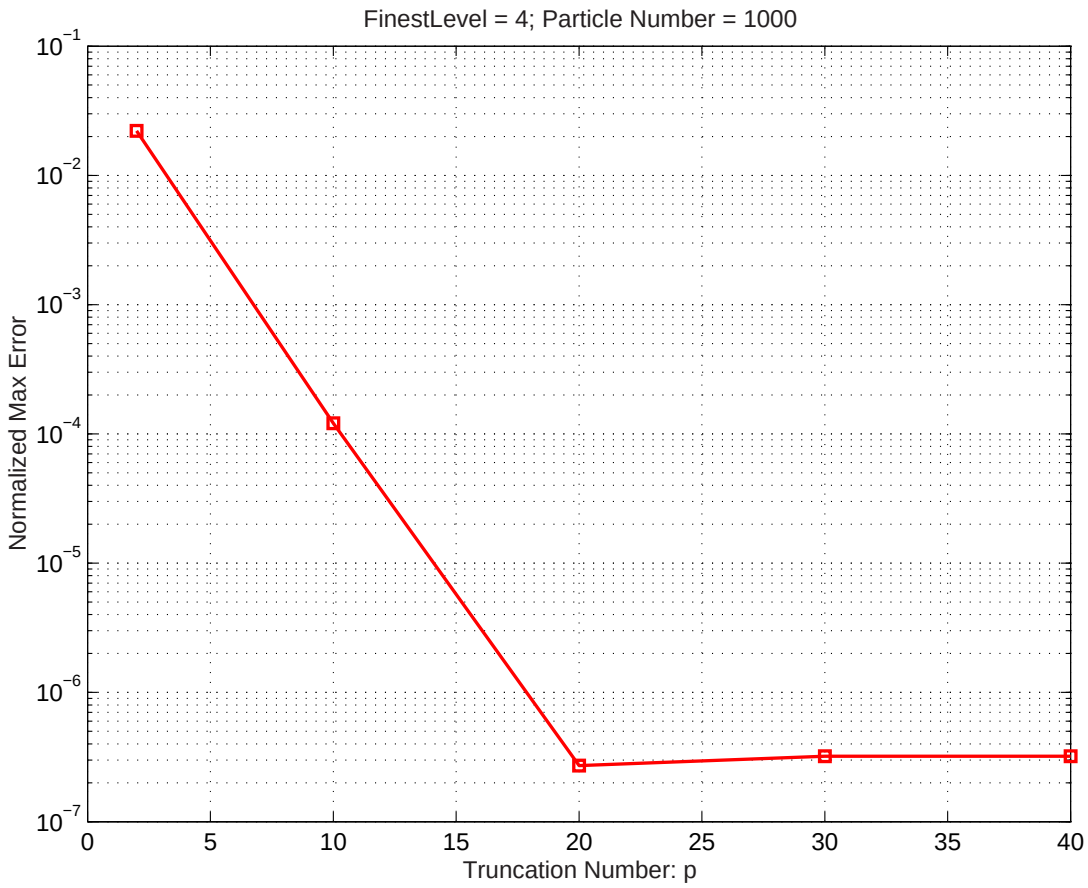



Figure 3.3: The dependence of the errors of the FMM on the truncation number $p$ at $N = 1000$, Max Level $= 4$.

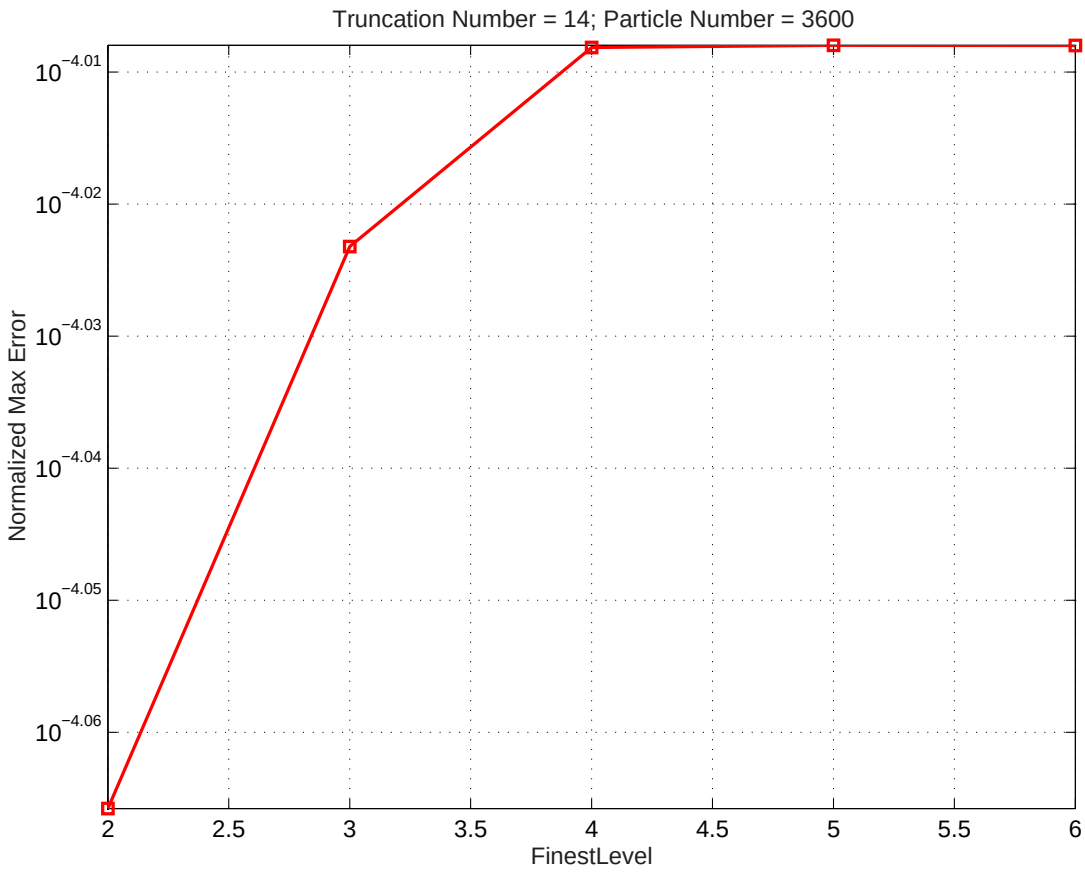



Figure 3.4: The dependence of the errors of the FMM on the maximum level $L$ at $N = 3600, p = 14$.

maximum level $L$ in the case $N = 3600$ and with the truncation number at the maximum level $L$ at $p = 14$. From equation (3.39), the error will decrease with the increase of the truncation number $p$ and the decrease of the maximum level $L$. Figures 3.3 and 3.4 both agree well with this prediction. When $d = 2, N = 1000$, the optimal max level is 4. If the specified error $\epsilon = 1.0 \times 10^{-4}$, the truncation number must satisfy $p > 40$ from (3.39). Figure 3.4 shows the error is already less than $1.0 \times 10^{-4}$ when $p > 20$. So the error is bounded as expected.

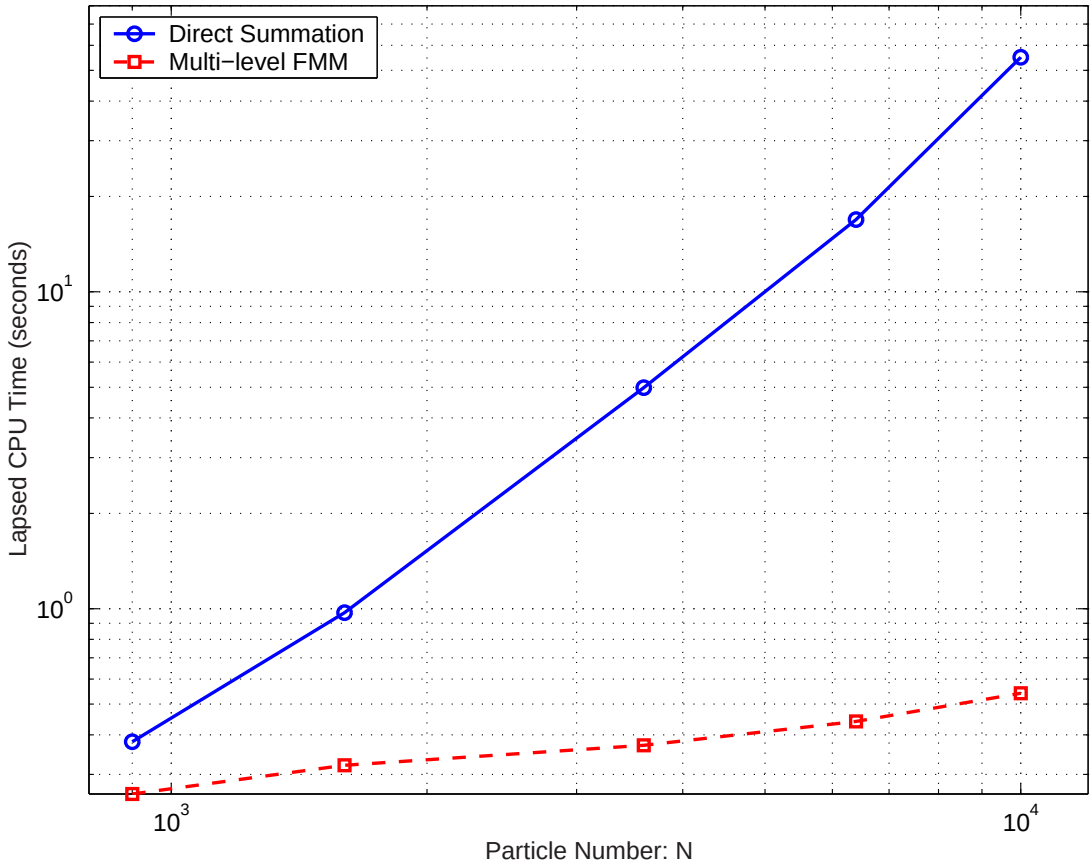



Figure 3.5: The comparison of the CPU time between direct summation and the FMM with $N$ from $900 \sim 10,000$.

Fig. 3.5 shows the comparison of the CPU time between direct summation

and the FMM (Max Level = 4) with $N$ from 900 to $10,000$. The cost of direct summation is proportional to $O(N^2)$, while the cost of the FMM is much less. According to Fig. 3.5, the cost of the FMM is even less than $O(N \log N)$. The reason for this is the vectorized operations in the code, since the cost of the FMM does not take the quad-tree data structure preparation into account.

Since the distribution of dilatation and vorticity in a practical computation will be highly uneven, the particles that carry dilatation and vorticity can be clustered in some isolated regions. For this reason an adaptive FMM is desirable to get rid of the empty boxes from the quad-tree structure and further reduce the cost of computation.

## 3.4 Correction of boundary velocity condition

Thus far, the discussion has considered the recovery of the velocity field in 2D unbounded compressible flow. In order to apply this method in a bounded domain, a potential velocity component $\mathbf{V_3}$ has to be added into equation (3.12).

As noted above, the velocities $\mathbf{V_1}, \mathbf{V_2}$ given by (3.17) and (3.18) respectively, will not generally satisfy velocity boundary conditions on a domain with boundaries. A potential velocity field $\mathbf{V_3}$ should be added to force $\mathbf{V}$ to satisfy the velocity boundary condition, namely (3.11). The potential velocity component $\mathbf{V_3}$ is used to satisfy given boundary conditions for the normal velocity on bounding walls. By definition

$$\mathbf{V_3} = \mathbf{V} - (\mathbf{V_1} + \mathbf{V_2}) \tag{3.40}$$

From equations (3.9-3.11) and (3.13-3.14), it follows that

$$\nabla \cdot \mathbf{V_3} = \nabla \cdot \mathbf{V} - \nabla \cdot \mathbf{V_1} = 0 \tag{3.41}$$

$$\nabla \times \mathbf{V_3} = \nabla \times \mathbf{V} - \nabla \times \mathbf{V_2} = 0 \tag{3.42}$$

$$\mathbf{V_3} \cdot \mathbf{n} = \mathbf{V_b} \cdot n - (\mathbf{V_1} + \mathbf{V_2}) \cdot \mathbf{n}, \tag{3.43}$$

and there exists a potential $\varphi$ such that $\mathbf{V_3} = \nabla\varphi$, and by (3.41)

$$\nabla^2\varphi = 0. \tag{3.44}$$

For fixed solid walls, the non-penetration velocity boundary condition applies and the normal velocity on the wall is zero. From equation (3.43), it follows that

$$\frac{\partial\varphi}{\partial\mathbf{n}} = \nabla\varphi \cdot \mathbf{n} = -(\mathbf{V_1} + \mathbf{V_2}) \cdot \mathbf{n}. \tag{3.45}$$

It is well known that Laplace's equation (3.44) with Neumman type boundary condition (3.45) can be solved in the form of an integral over the bounding surface. This supplies the basis for the boundary element method [43]. In this, the boundary is discretized into a number of boundary elements. A source or sink is put at the center of each boundary element. The center of the $i^{\text{th}}$ element is located at $(x_i, y_i)$. The induced velocities of a unit source or sink at the center of the $j^{\text{th}}$ element are given by (see [43])

$$\begin{aligned} u_i &= \frac{1}{2\pi}\int_{-L_i}^{L_i} \frac{x_j - x}{(x_j - x)^2 + y_j^2} dx \\ &= \frac{1}{2\pi} \ln \frac{(x_j + L_i)^2 + y_j^2}{(x_j - L_i)^2 + y_j^2}, \end{aligned} \tag{3.46}$$

$$\begin{aligned} v_i &= \frac{1}{\pi}\int_{-L_i}^{L_i} \frac{y_j}{(x_j - x)^2 + y_j^2} dx \\ &= \frac{1}{\pi}(tan^{-1}\frac{x_j + L_i}{y_j} - tan^{-1}\frac{x_j - L_i}{y_j}), \end{aligned} \tag{3.47}$$

where, $L_i$ is half of the length of the $i^{\text{th}}$ element.

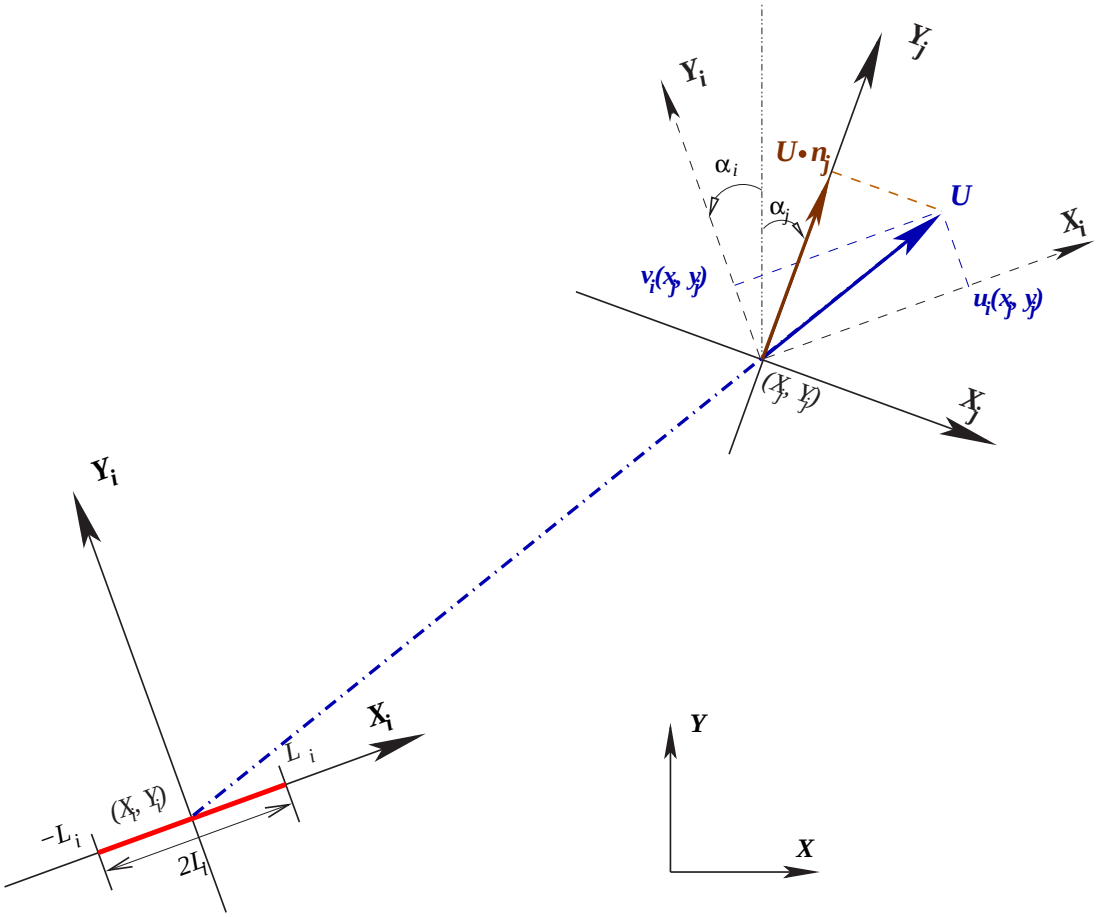



Figure 3.6: The $i^{th}$ boundary element induced velocities at the center of the $j^{th}$ element.

Figure 3.6 shows the $i^{\text{th}}$ element induced velocities $u_i, v_i$ at the center of the $j^{\text{th}}$ element in the $X_i, Y_i$ local coordinates. $\alpha_i, \alpha_j$ are the angles of the $i^{th}$ and $j^{th}$ elements to the global $X$ axis. Because each source or sink has a direct contribution to the normal velocities of the others, the normal velocity at a single element is a sum of the contributions from all the sources. The total normal velocity at the boundary due to sources or sinks should cancel out the normal velocity component from dilatation and vorticity field as given in (3.45). The

strength of each source or sink can be determined by solving the linear system.

$$\mathrm{A}_{ij}\delta_j = \frac{\partial \varphi_i}{\partial n} = -(\mathbf{V_1} + \mathbf{V_2})_i \cdot \mathbf{n}_i, \tag{3.48}$$

where $\delta_j$ are the strength of sources or sinks,

$$\mathrm{A}_{ij} = u_i \sin(\alpha_j - \alpha_i) + v_i \cos(\alpha_j - \alpha_i).$$

Any standard linear system solver can be used to solve (3.48) and thus calculate the strengths of the sources.

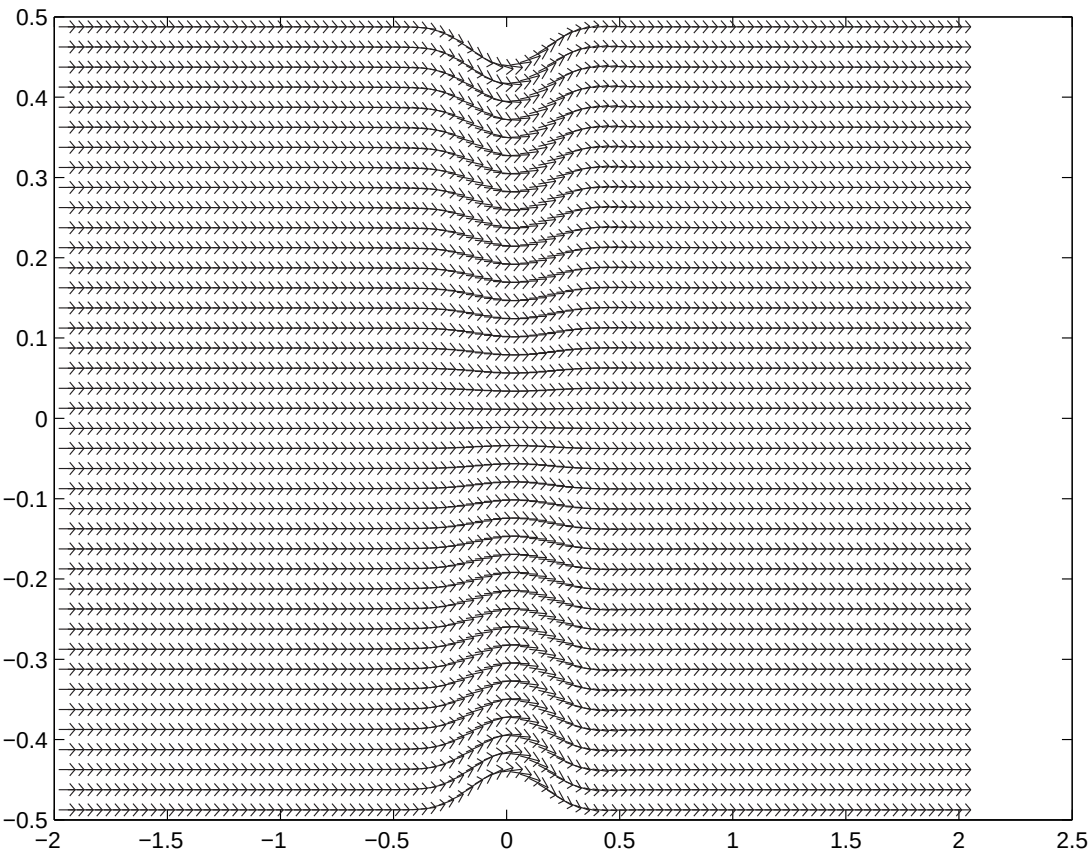



Figure 3.7: The stream lines of the potential flow field in a nozzle with $V_\infty = 0.4$.

As a test of the method the potential flow in a duct has been computed using the boundary element code and compared to the exact solution. In this, the

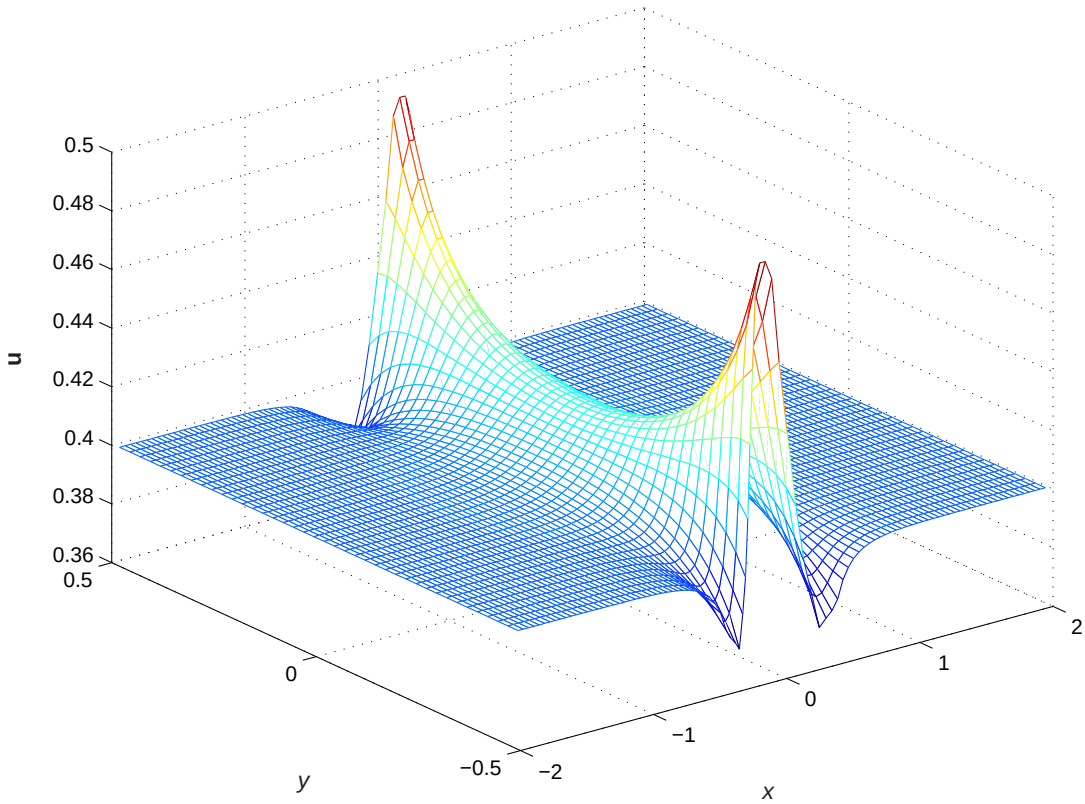



Figure 3.8: The potential velocity component $u$ in a nozzle with $V_\infty = 0.4$.

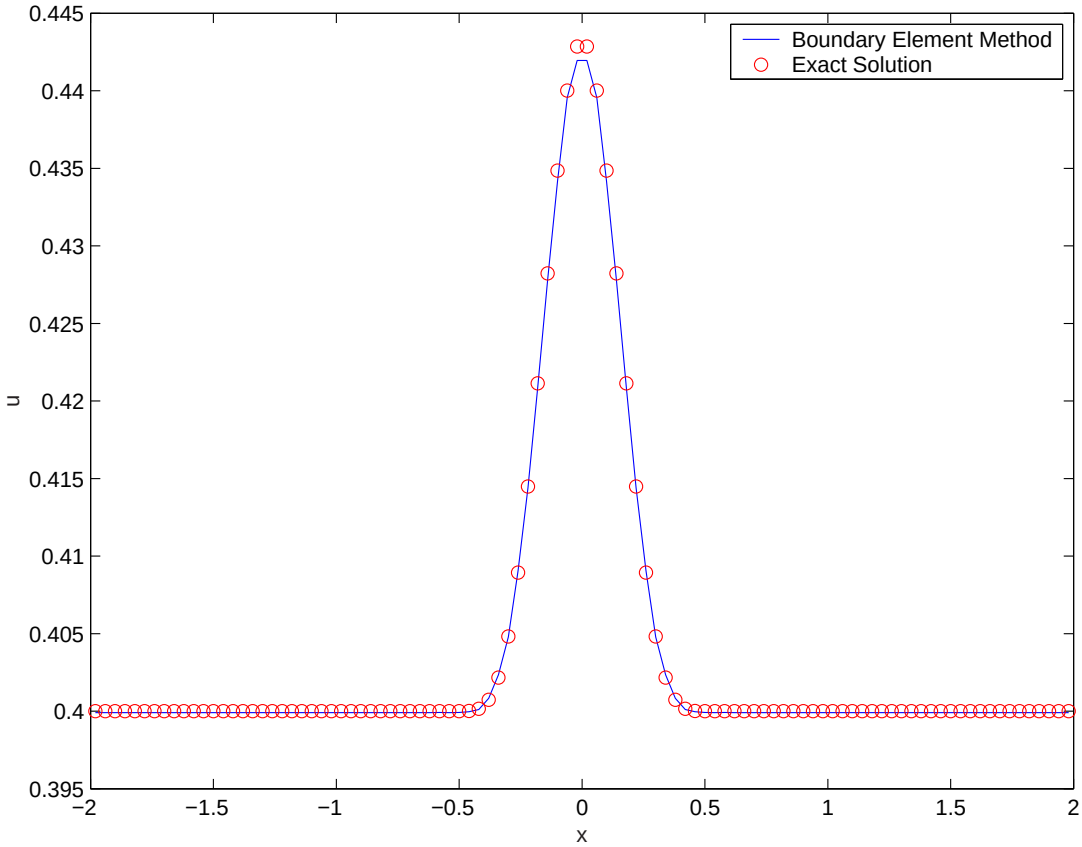



Figure 3.9: The comparison of the $u$ distribution in a nozzle with $V_\infty = 0.4$. (—): the cross-stream averaged $u$ from boundary element method; ($o$): the exact quasi-1D solution.

nozzle walls have been discretized into 240 source elements. The vector distribution of potential flow field calculated by using the boundary element method is shown in Figure 3.7. Figure 3.8 shows the $u$ component of the potential velocity distribution in a nozzle with upstream velocity $V = 0.4$. As seen in Figure 3.9, the results of the boundary element method agree well with the 1D exact solution obtained by integrating across the span. The boundary element method is thus assured to be working correctly.

For viscous flow, besides the requirement that the normal velocities at the wall are zero, the tangential velocities should also be zero at the wall. This condition can be implemented by deploying vortex sheets along the boundary. The velocities induced by vortex sheets are set to cancel the tangential velocities recovered from the dilatation and vorticity fields. This is similar to the procedure of eliminating the tangential velocities used in incompressible vortex methods.

# Chapter 4

# Evaluation of Spatial Derivatives

The approximation of continuous derivatives from discrete data is the key to any numerical scheme. When the computational domain is discretized on a fixed grid, finite differences can be used to approximate derivatives. In addition, the finite element method or the finite volume method can be used to approximate the integrals of differential operators over discrete computational elements. For vortex methods or grid-free Lagrangian methods, the computational domain is discretized into Lagrangian particles that are convected with the fluid motion. There are no fixed spatial configurations between particles. Thus traditional methods based on fixed grids cannot be used for applications of the Lagrangian dilatation element method.

The moving least-square method uses a sliding window to collect discrete data around its center and solves for a polynomial fit on the interpolation window based on minimization of the errors using the least-square criteria. It can be used to easily treat any arbitrary spatial derivatives. In this chapter the moving least-square method as a general approach to approximating spatial derivatives

in particle methods is discussed.

## 4.1 Overview of spatial derivative treatments for grid-free methods

Many grid-free schemes have recently emerged as alternatives to traditional grid based methods. Gingold and Monaghan's smooth particle hydrodynamics (SPH) [38] uses the reproducing kernel method to compute derivatives. Belytschko's element free Galerkin method (EFG) [5] evaluates the spatial derivatives on the background mesh using the Galerkin method. Duarte and Oden [26] and Babuska and Melenk [51] treat the derivatives using a partition of unity. Lohner [49] and Onate et. al. [58] [59] use moving least-square methods to treat the derivatives. Marshall and Grant [50] use a weighted least square method to evaluate the derivatives in a vortex method simulation of axisymmetric flows with and without swirl. In that study a quadratic polynomial was used and the scale of the Gaussian weight function depended on the particle spacing. Strickland [64] and Nitsche [56] use moving least-square fitting to compute derivatives in the context of a Lagrangian dilatation element method. Eldredge [27] [29] extended the particle strength exchange (PSE) to treat spatial derivatives in which global cumulative integration is used to approximate local derivatives in computing co-rotating and leap-frogging vortices in compressible flow.

Basically all of these methods employ either reproducing kernel methods, a partition of unity, or moving least square methods to construct interpolation functions of unknown variables. The reproducing kernel method and the moving least square method share a lot of properties and are very similar [1] [6]. Whatever

grid-free method is used, the spatial derivatives have to be computed by first constructing some kind of interpolation function.

According to Beltyschko [6], Marshall and Grant [50] and Fan and Gijbels [31], the moving least square fitting can give more accurate results than other least square methods. Consequently the moving least-square method is employed to construct the interpolation function and it will be used in this research to evaluate all the derivatives encountered in the governing equations.

## 4.2 Moving least square method

Suppose we want to evaluate the derivative $du/dx$ based on the discrete data $(x_i, u_i)$. First define an interpolation window $\mathbf{W}_i$ which is used to collect $n$ points. Then the interpolation function within the interpolation window is constructed by the moving least square method. It is constructed by the summation of a series of polynomials multiplying unknown coefficients that are solved for by minimizing a localized least-square error. The derivatives are obtained by differentiating the interpolation function. Assume the interpolation function $u^h(x)$ can be expressed by:

$$u^h(x) = \sum_{i=1}^{m} p_i(x)\alpha_i(x) \equiv \mathbf{p}^{\mathbf{T}}(\mathbf{x})\tilde{\alpha}(\mathbf{x}), \tag{4.1}$$

where $p_i(x)$ are chosen basis functions, $\alpha_i(x)$ are the corresponding unknown coefficients and $m$ is the number of basis functions. The basis functions are taken as polynomials consistent with the Taylor series expansion to which any function can be expanded.

For 1D problems, a quadratic polynomial basis is enough to allow the interpolation function to be $C^2$ continuous. However, according to Fan and Gijbels

[31], a cubic basis is recommended to being better than others. Thus a cubic basis has been used for all the 1D applications in this research:

$$\mathbf{p^T} = (1, x, x^2, x^3). \tag{4.2}$$

For 2D problem, the results of a cubic basis function for some circumstances are good but instable for others. A quadratic polynomial basis is instead used in consideration of the efficiency and stability of the scheme:

$$\mathbf{p^T} = (1, x, y, x^2, xy, y^2). \tag{4.3}$$

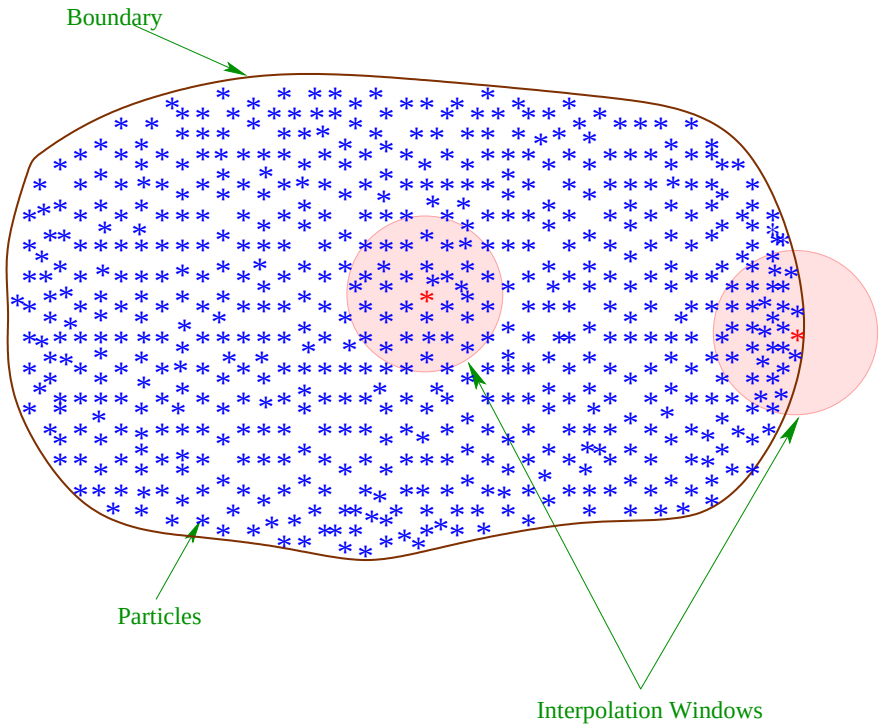



Figure 4.1: Moving Least Square Fitting, * particles center in the shadowed interpolation windows.

The basic idea of the moving least square method is that an interpolation function should be obtained in each interpolation window that will be only used

to evaluate the derivative at its central point (see Fig. 4.1). An interpolation window is made for every particle in order to get the derivatives of all particles.

In the interpolation window $\mathbf{W}_i$, the coefficients $\alpha_i(x)$ are obtained by performing a weighted least-square fitting, which means minimizing the quadratic function:

$$J = \sum_{i=1}^{n} w(x_i - x^*) \left( \sum_{j=1}^{m} p_j(x_i)\alpha_j - u_i \right)^2, \tag{4.4}$$

where $x^*$ is the position of the central particle, $x_i$ are the positions of the particles collected into the interpolation window, $u_i$ are known properties carried by each particle, $p_j$ are the selected basis functions, $\alpha_j$ are the unknown coefficients to be solved for, and $w(x_i - x^*)$ are weighting functions with compact support.

According to [5] (p.232), the weighting functions play an important role in the performance of the method. These should be relatively large in magnitude for $x_i$ close to $x^*$ and relatively small for particles far away. Therefore, weighting functions are typically built to depend on the distance between the discrete particles and the central particles. In the present methods, the exponential weighting function:

$$w(x_i - x^*) = e^{\frac{-(x_i - x^*)^2}{\sigma}} \tag{4.5}$$

is used in the computations. Here $\sigma$ is a shape parameter which is used to control the shape of the weighting function. It is clear that $\sigma$ is a critical parameter which influences the success of the method. Some insights into how to determine $\sigma$ have emerged from 1D and 2D test cases. It can be chosen according to a window size that allows for a given number of particles to be used for the least square fit. Meanwhile, the window size can control the accuracy of the computed derivatives. The influence of the shaping parameter $\sigma$ and the window size on the accuracy of the moving least square method will be discussed in the next section.

The target quadratic function (4.4) to be minimized can be rewritten in the following matrix form:

$$J = (\mathbf{P}\tilde{\alpha} - \mathbf{u})^{\mathbf{T}}\mathbf{W}(\mathbf{x})(\mathbf{P}\tilde{\alpha} - \mathbf{u}), \tag{4.6}$$

where

$$\tilde{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_n)^T,$$

$$\mathbf{u}^T = (u_1, u_2, \cdots, u_n),$$

$$\mathbf{P} = \begin{pmatrix} p_1(x_1) & p_2(x_1) & \cdots & p_m(x_1) \\ p_1(x_2) & p_2(x_2) & \cdots & p_m(x_2) \\ \vdots & \vdots & \ddots & \vdots \\ p_1(x_n) & p_2(x_n) & \cdots & p_m(x_n) \end{pmatrix},$$

and

$$\mathbf{W}(x) = \begin{pmatrix} w(x^* - x_1) & 0 & \cdots & 0 \\ 0 & w(x^* - x_2) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w(x^* - x_n) \end{pmatrix}.$$

Then the weighted least squares problem (4.4) can be written as:

$$\frac{\partial J}{\partial \alpha_i} = \frac{\partial(\mathbf{P}\tilde{\alpha} - \mathbf{u})^{\mathbf{T}}\mathbf{W}(\mathbf{P}\tilde{\alpha} - \mathbf{u})}{\partial \alpha_i} = 0, \tag{4.7}$$

In which case the coefficient vector $\tilde{\alpha}$ is given by:

$$\tilde{\alpha} = (\mathbf{P}^{\mathbf{T}}\mathbf{W}\mathbf{P})^{-\mathbf{1}}\mathbf{P}^{\mathbf{T}}\mathbf{W}\mathbf{u}. \tag{4.8}$$

After getting the interpolation function, the derivatives at the center of the interpolation window are computed by differentiating the interpolation function. The

moving least square method requires that the center of the interpolation window be placed over each particle in order to get the derivatives. Consequently for $N$ particles $N$ least square fits are required at each time step which may be very expensive in terms of computational cost.

Another issue which is worthy of mentioning here is the inconsistency of the moving least square fitting. When the interpolation window moves close to the boundary, a part of the window will be empty and not filled by particles (see Figure 4.1). In this case the least square method tends to be a one-side biased fitting which deteriorates the accuracy of the method. In order to resolve this inconsistency, it is necessary to create artificial particles to fill the vacancy of the interpolation window. The values of these artificial particles are decided by the boundary conditions. For the inflow and outflow boundaries, the values of artificial particles can be determined by using the characteristic waves moving across the inlet and outlet. For the solid wall boundary condition, if the property that particles carry has a fixed value or has a zero gradient at the wall, then different reflection methods are needed to create the artificial particles to enforce these boundary conditions. The detailed implementations of how to create artificial particles associated with different boundary conditions will be presented in the next chapter with examples.

## 4.3 Examples

An example has been used to evaluate the accuracy of the moving least square fitting with different weighting shape parameters $\sigma$ and interpolation window size. The accuracy of the approximated derivatives computed from moving least

square fitting can be measured by varying the number of particles and the inter-particle spaces. The domain is respectively approximated by $50 \times 50$, $75 \times 75$, $100 \times 100$, $150 \times 150$, and $200 \times 200$ evenly spaced particles in the $x, y$ directions. The example considers the function

$$f(x, y) = \cos x \sin y, \tag{4.9}$$

where $x, y \in [0, 2\pi]$. The derivatives of $f(x, y)$ are

$$\begin{aligned} \frac{\partial f}{\partial x} &= -\sin x \sin y, \quad \tfrac{\partial f}{\partial y} = \cos x \cos y, \\ \frac{\partial^2 f}{\partial x^2} &= -\cos x \sin y, \quad \tfrac{\partial^2 f}{\partial y^2} = -\cos x \sin y. \end{aligned} \tag{4.10}$$

The second-order polynomial (4.3) is used as the base function for the moving least square fitting. Equation (4.5) is the weighting function associated with the fitting.

Figures 4.2 - 4.4 show how the errors of derivatives computed by moving least-square fitting change with the shape factor $\sigma$. A circle of a radius $\sqrt{0.06}$ is used as the interpolation window for all the above cases. There appears to exist an optimal $\sigma$ to minimize the errors of computed derivatives associated with each amount of particles. The errors may be significant if the chosen shape factor is very much larger or less than the optimal value. From figures 4.2 - 4.4 the optimal shape factor decreases with the increase of the number of particles. Therefore, the shape factor $\sigma$ might be related to the average spacing between particles. From the results of these figures, the optimal $\sigma$ is proportional to the distance between particles.

$$\sigma_{op} \propto s^2 \approx C\big((\delta x)^2 + (\delta y)^2\big), \tag{4.11}$$

where $C$ is a constant depending on the function $f(x, y)$ and $\approx 1/3$ for this example.

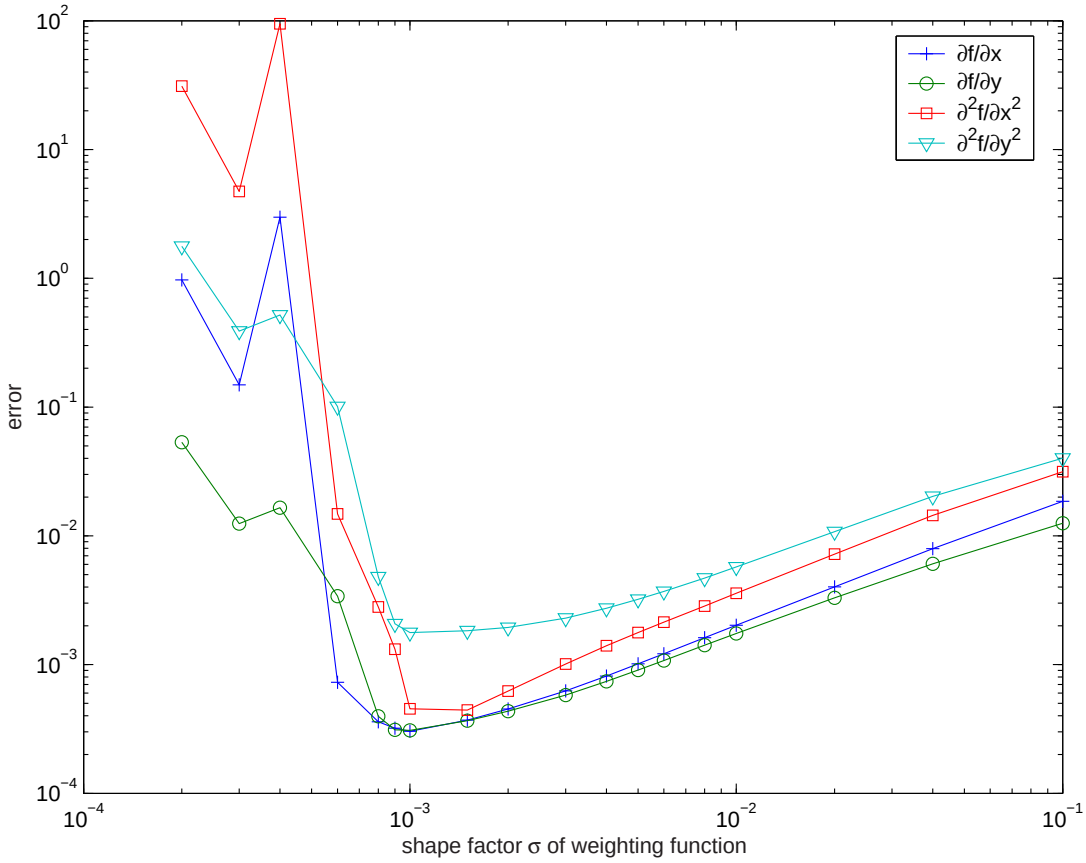



Figure 4.2: Dependence of the errors of moving least square fitting on the shape parameter $\sigma$ of the weighting function ($50 \times 50$ particles).

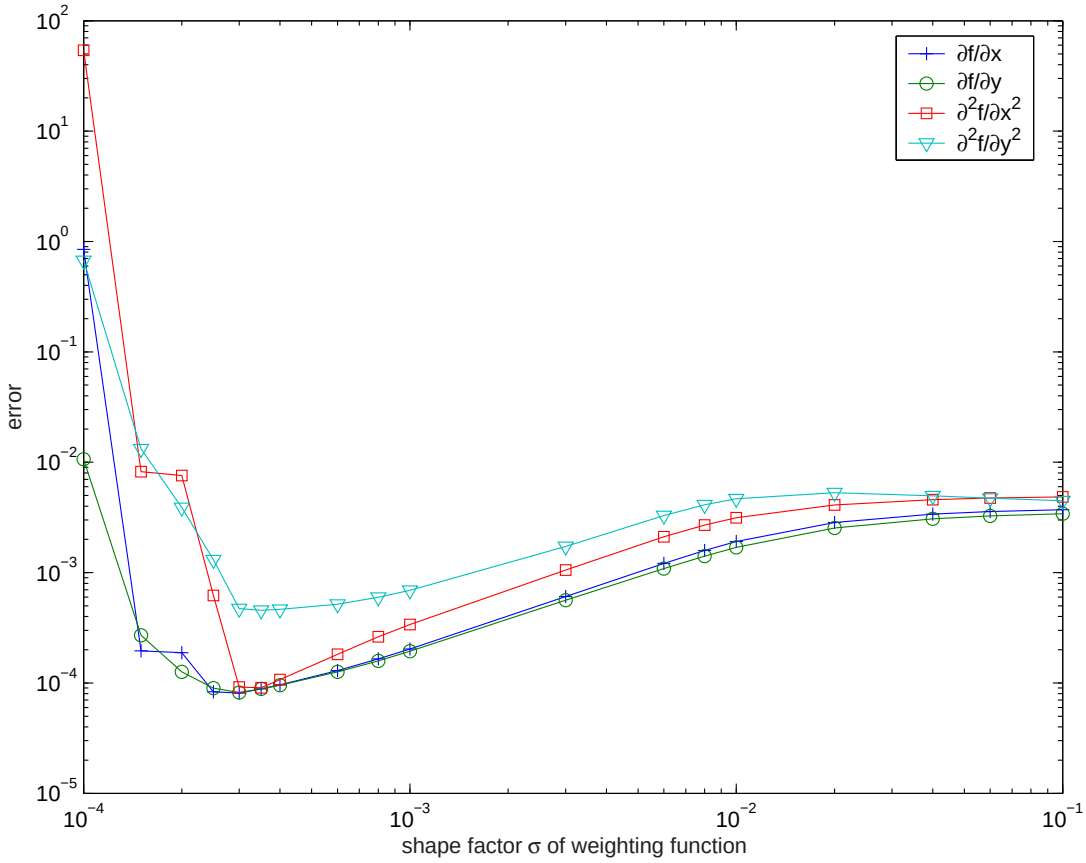



Figure 4.3: Dependence of the errors of moving least square fitting on the shape parameter $\sigma$ of the weighting function ($100 \times 100$ particles).

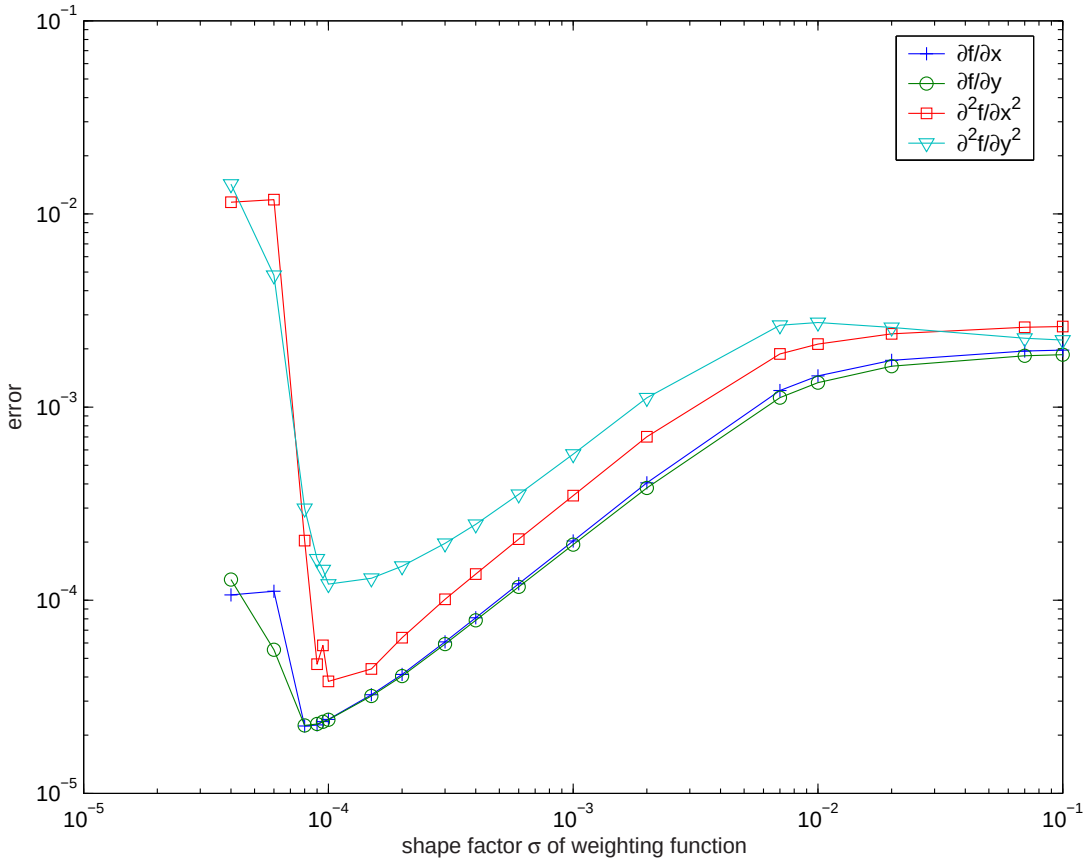



Figure 4.4: Dependence of the errors of moving least square fitting on the shape parameter $\sigma$ of the weighting function ($200 \times 200$ particles).

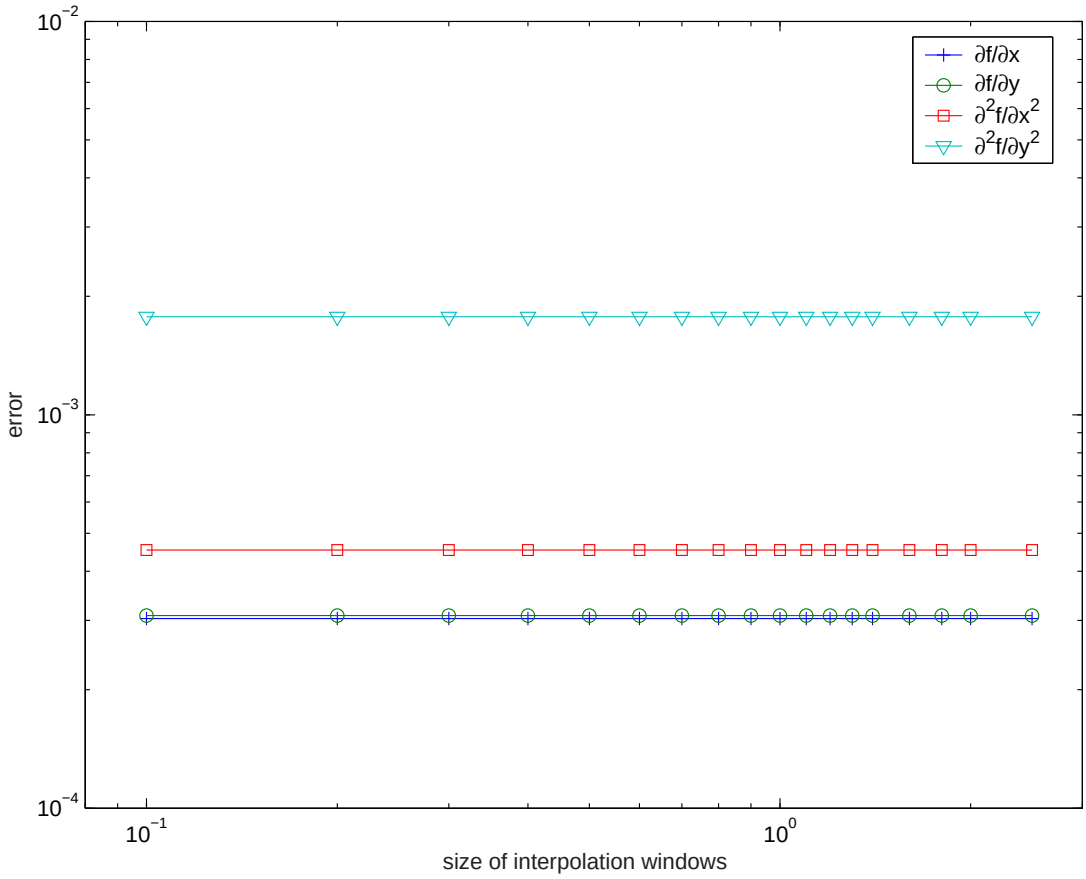



Figure 4.5: Variation of the errors of moving least square fitting with the interpolation window size ($50 \times 50$ particles).

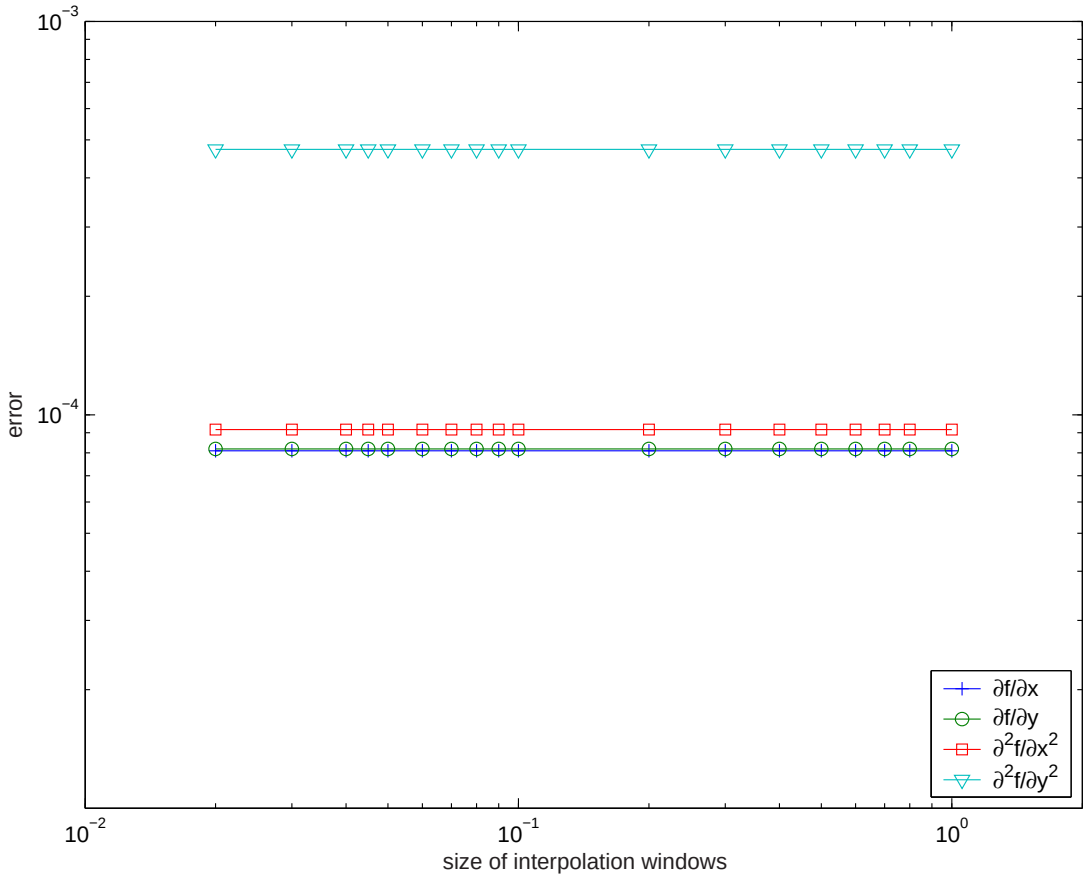



Figure 4.6: Variation of the errors of moving least square fitting with the interpolation window size ($100 \times 100$ particles).

Figures 4.5 and 4.6 show that the errors of moving least fitting are insensitive to the interpolation window size as long as the number of particles collected in the window are sufficient enough for the least-square fitting. The interpolation window sizes used in figures 4.5 and 4.6 are $r^2$, where $r$ is the radius of the interpolation circle. The optimal shape factors are used to evaluate the errors with the change of interpolation window size. The particles closer to the center of the interpolation window (evaluation position) weight more in the least-square fitting than those further away through the use of the weighting function. The particles distant from the evaluation position do not have too much effect on the MLS. Therefore the size of interpolation window tends to play a small role in determining the accuracy of MLS.

Table 4.1: How the average error changes with the number of particles

| *Number of Particles* | *Average Error* | | | |
|---|---|---|---|---|
| | $\frac{\partial f}{\partial x}$ | $\frac{\partial f}{\partial y}$ | $\frac{\partial^2 f}{\partial x^2}$ | $\frac{\partial^2 f}{\partial y^2}$ |
| 50×50 | 2.9880E-004 | 3.0770E-004 | 2.8752E-004 | 1.7519E-003 |
| 75×75 | 1.3922E-004 | 1.4130E-004 | 1.3822E-004 | 7.9841E-004 |
| 100×100 | 8.1216E-005 | 8.1609E-005 | 1.0611E-004 | 4.5200E-004 |
| 150×150 | 4.6475E-005 | 4.6027E-005 | 5.7378E-005 | 2.1229E-004 |
| 200×200 | 2.2788E-005 | 2.2802E-005 | 6.6786E-005 | 1.5387E-004 |

Table 4.1 shows how the average error changes with the number of particles. The average errors decrease with an increase in the number of particles. Table 4.2 shows how the maximum error changes with the number of particles. Note here that the maximum errors with $150 \times 150$ particles is less than that for $200 \times 200$. The boundary condition has not been treated in this test example. Therefore

Table 4.2: How the maximum error changes with the number of particles.

| *Number of Particles* | *Maximum Error* | | | |
|---|---|---|---|---|
| | $\frac{\partial f}{\partial x}$ | $\frac{\partial f}{\partial y}$ | $\frac{\partial^2 f}{\partial x^2}$ | $\frac{\partial^2 f}{\partial y^2}$ |
| 50×50 | 7.3046E-004 | 1.3500E-003 | 8.1615E-003 | 6.4878E-002 |
| 75×75 | 3.4202E-004 | 6.2400E-004 | 7.0388E-003 | 4.5510E-002 |
| 100×100 | 2.1730E-004 | 4.5087E-004 | 1.4104E-002 | 3.6218E-002 |
| 150×150 | 1.1452E-004 | 1.5032E-004 | 2.1054E-003 | 2.1447E-002 |
| 200×200 | 3.3680E-004 | 3.9885E-004 | 4.2746E-002 | 5.1224E-002 |

the accuracy of MLS falls when the interpolation window moves close to the boundary with part of the window left blank. The maximum errors happen on those particles closest to the boundary. The average and maximum errors decrease with the increase of particle numbers when the particle numbers are less than $150 \times 150$ due to the increase of spatial resolution of MLS. However when the particle numbers increase above the threshold $150 \times 150$, the increase of maximum errors due to the decrease of distance to boundary surpass the effect of increased spatial resolution. Therefore the maximum errors with $200 \times 200$ are greater.

Figure 4.7 shows that the average error of $\frac{\partial f}{\partial x}$ varies with the distance between particles. It is clear from the figure that the second-order polynomial basis function offers essentially second order accuracy in the computed derivatives.

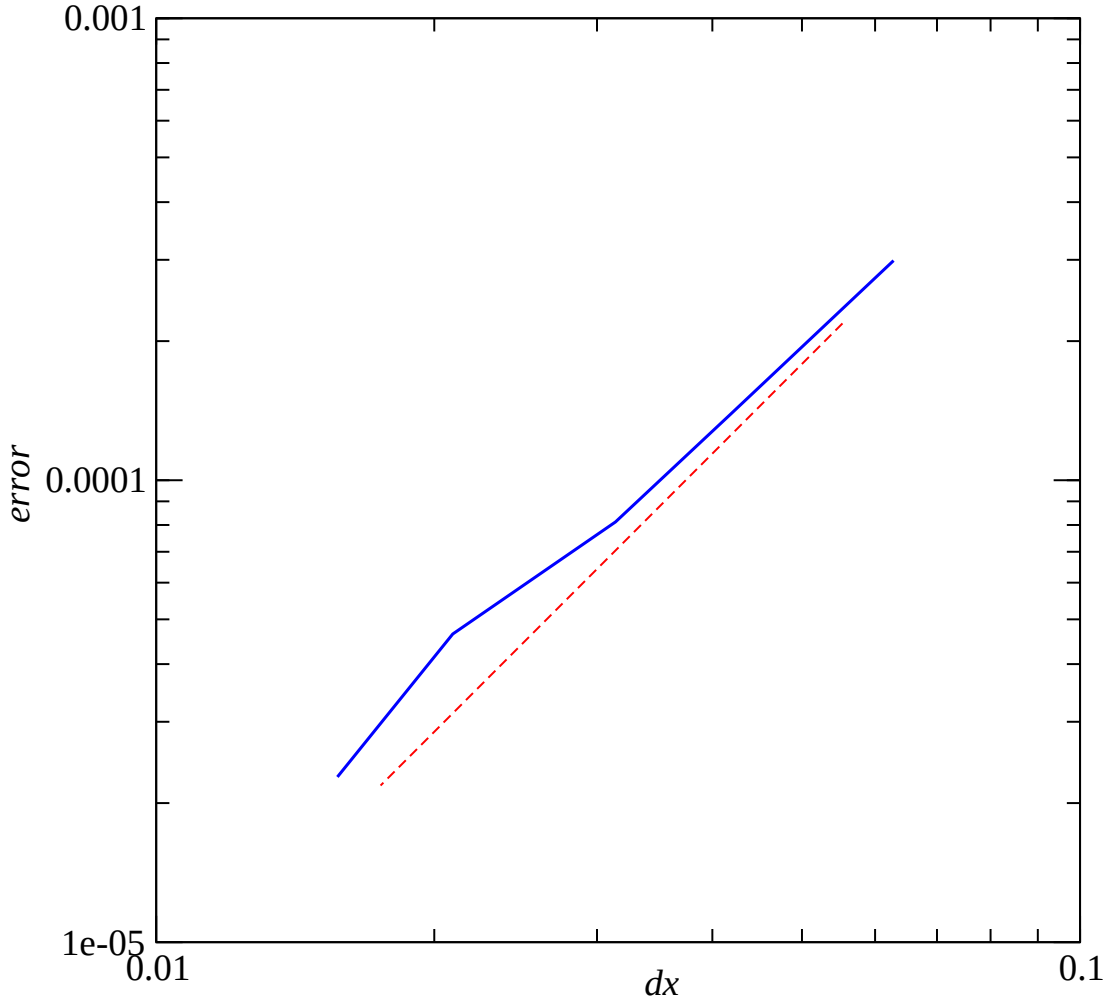



Figure 4.7: The accuracy of computed $\frac{\partial f}{\partial x}$: (—) compared with a 2nd order line: (- -).

# Chapter 5

# Implementation Details

The Lagrangian dilatation element method introduced in Chapter 2 is further developed into having the capability of solving real gas dynamics problems in this chapter. The details of applying the method to a subsonic nozzle flow are presented. Generalization of the approach into other shock free compressible flow problems is straightforward. First, the time integration of the Lagrangian governing equations of isentropic compressible flow is investigated. The implementations of wall boundary conditions and the inlet and outlet boundary conditions are next discussed. One feature of Lagrangian methods is that computational elements are convected with the local flow velocities. The computational domain will be changing with time due to the movement of the Lagrangian computational elements unless the elements closest to the inlet and outlet boundaries are modified to reflect the boundary changes. Here the elements closest to the inlet are extended and the elements closest to the outlet are trimmed in such a way as to keep the computational domain fixed. In addition, elements at the inlet are subdivided when they grow beyond a threshold, while elements at the outlet are

merged with others when they are small. The subdivision and merger process of Lagrangian computational elements is discussed at the end of this chapter.

## 5.1 Time integration

In this chapter the Lagrangian dilatation element method is developed into solving an important simplified problem, namely, a two-dimensional subsonic shock-free nozzle flow (see Figure 5.2(a)). By assuming adiabatic walls, the flow is isentropic. For the present work, there is an incoming potential flow with uniform velocity $u_0$. Conceptually the problem of interest may be viewed as an initially potential flow that is suddenly turned into a compressible flow at time $t = 0$. Equivalently, it may be regarded as an initial uniform flow in a straight duct that is suddenly bent. Because the flow is initially potential and free of vorticity, the flow subsequently is vorticity-free too. The flow in the nozzle is governed by the non-dimensional 2D isentropic equations (2.33) and (2.34) plus the element motion equation (2.41) and the volume governing equation (2.43).

Initially the computational domain is fully covered by a set of $N$ Lagrangian elements. The first and last columns of elements are aligned with the inlet and outlet boundaries respectively. All the elements are located entirely within the computational domain. In fact, due to the nature of Lagrangian methods, these two conditions would not be automatically satisfied unless they are forced to be true at the end of each time iteration. The details of how to enforce these conditions is presented in the next sections. The governing equations of the $i^{th}$ computational element ($i = 1, ..., N$) take the following forms from the previously

derived equations:

$$\frac{D\mathbf{X_i}}{Dt} = \mathbf{U_i} \tag{5.1}$$

$$\frac{D\theta_i}{Dt} = u_i\frac{\partial\theta_i}{\partial x} + v_i\frac{\partial\theta_i}{\partial y} - \nabla^2\left(\frac{u_i^2+v_i^2}{2} + \frac{1}{\gamma-1}T_i\right) \tag{5.2}$$

$$\frac{DT_i}{Dt} = -(\gamma-1)T_i\theta_i \tag{5.3}$$

$$\frac{DV_i}{Dt} = V_i\theta_i, \tag{5.4}$$

where $\mathbf{X_i}$ are the $i^{th}$ element position vectors, $\mathbf{U_i} = (u_i, v_i)$ are the velocity vectors, $\theta_i$ and $T_i$ are the dilatation and temperature the element carries, and $V_i$ is its volume.

The governing equations (5.1) - (5.4) are partial differential equations and the solutions can be achieved using a numerical integration scheme. Generally an implicit integration scheme is too difficult to be used in this case because the equations are highly coupled with implicit variables such as velocities that need an extra step to being recovered from the primary variable $\theta$. On the other hand, the results given by a first-order explicit Euler scheme tend to be unstable, based on numerical experimentation. A second order Runge-Kutta scheme is thus used to complete the numerical integration in the present work. A fourth order Runge-Kutta algorithm was also tested and yielded similar outcomes. The increase of accuracy is not of sufficient consequence to justify the increase of computational work related to higher order schemes.

The time integration scheme was found to be insensitive to an increase of the number of elements used and the decrease of time step. Unlike traditional grid-based schemes, there is no evidence that a critical CFL number exists affecting the numerical time integration. Theoretical studies of the convergence and stability of the Lagrangian scheme are not feasible at the present time as

they await a breakthrough in the computational theories of coupled nonlinear Lagrangian differential equations. Therefore, all the discussions here are given without strictly mathematical proof.

### 5.1.1 First Runge-Kutta step

At the outset of any numerical scheme for solving (5.1) - (5.4), a procedure is required for evaluating velocities $u_i^n, v_i^n$ and spatial derivatives

$$\partial^2 \left( \frac{(u_i^n)^2 + (v_i^n)^2)}{2} + \frac{T_i^n}{\gamma - 1} \right) / \partial x^2,$$

and

$$\partial^2 \left( \frac{(u_i^n)^2 + (v_i^n)^2)}{2} + \frac{T_i^n}{\gamma - 1} \right) / \partial y^2.$$

The velocity can be recovered from the dilatation field as discussed in Chapter 3. The spatial derivatives for the interior elements can be computed by using the moving least square fits as shown in the discussion of Chapter 4.

When computational elements are located in the vicinities of the nozzle wall and inlet or outlet boundaries, the moving least square interpolation window is partially filled with just the data that originates from inside of the computational domain. Ignoring the information lying outside the inlet and outlet is equivalent to ignoring the characteristic waves existing in the compressible flow, which would give unsatisfactory results. A part of the window sitting outside of the computational domain will be left blank unless some boundary treatment is done to create the necessary information.

In this research, a particle reflection technique is used to create data to fill up the interpolation windows for the region near the wall. Meanwhile, the Riemann invariants of characteristic waves in compressible flow are used to create the

necessary data on outward moving waves to fill up the interpolation windows near the inlet and outlet boundaries. These problems will be discussed respectively in detail in the following two sections. It is assumed here that the derivatives are all computed successfully. So it is then possible to advance through the first step of a second order Runge-Kutta scheme given by:

$$\begin{aligned}
\mathbf{X_i}^* &= \mathbf{X_i}^n + dt\mathbf{U_i}^n && (5.5)\\
\theta_i^* &= \theta_i^n - dt\left(u_i^n\frac{\partial\theta_i^n}{\partial x} + v_i^n\frac{\partial\theta_i^n}{\partial y} - \nabla^2\left(\frac{(u_i^n)^2+(v_i^n)^2}{2} + \frac{1}{\gamma-1}T_i^n\right)\right) && (5.6)\\
T_i^* &= T_i^n - dt(\gamma-1)T_i^n\theta_i^n && (5.7)\\
V_i^* &= V_i^n + dtV_i^n\theta_i^n, && (5.8)
\end{aligned}$$

yielding the provisional values $X_i^*, \theta_i^*, u_i^*, v_i^*, T_i^*$ and area $V_i^*$, of the $i^{th}$ element $(i = 1, ..., N)$ for the solution at $t + dt$.

### 5.1.2 Second Runge-Kutta step

As part of the second step of the Runge-Kutta scheme it is necessary to evaluate velocities $u_i^*, v_i^*$ and spatial derivatives

$$\partial^2\left(\frac{(u_i^*)^2+(v_i^*)^2)}{2} + \frac{T_i^*}{\gamma-1}\right)/\partial x^2,$$

and

$$\partial^2\left(\frac{(u_i^*)^2+(v_i^*)^2)}{2} + \frac{T_i^*}{\gamma-1}\right)/\partial y^2.$$

using the provisional solutions obtained from the first step of the Runge-Kutta scheme (5.5 - 5.8). As before, the evaluation of spatial derivatives encounters problems near the boundaries. The data outside of the domain can be supplemented by using either element reflection near the wall or local wave structures near the inlet and outlet boundaries. The details are discussed in the next two

sections. The velocities can be recovered from the provisional dilatation field similar to the first step.

Then, the stencil of the second step of the Runge-Kutta scheme is:

$$\mathbf{X_i}^{n+1} = \mathbf{X_i}^{n} + \frac{dt}{2}(\mathbf{U_i}^{n} + \mathbf{U_i}^{*}) \tag{5.9}$$

$$\begin{aligned}\theta_i^{n+1} &= \theta_i^n - \frac{dt}{2}\Bigg(\Big(u_i^n\frac{\partial\theta_i^n}{\partial x} + v_i^n\frac{\partial\theta_i^n}{\partial y} \\ &\quad -\nabla^2\big(\frac{(u_i^n)^2+(v_i^n)^2}{2} + \frac{1}{\gamma-1}T_i^n\big)\Big) + \Big(u_i^*\frac{\partial\theta_i^*}{\partial x} \\ &\quad +v_i^*\frac{\partial\theta_i^*}{\partial y} - \nabla^2\big(\frac{(u_i^*)^2+(v_i^*)^2}{2} + \frac{1}{\gamma-1}T_i^*\big)\Big)\Bigg)\end{aligned} \tag{5.10}$$

$$T_i^{n+1} = T_i^n - \frac{dt}{2}\left((\gamma-1)T_i^n\theta_i^n + (\gamma-1)T_i^*\theta_i^*\right) \tag{5.11}$$

$$V_i^{n+1} = V_i^n + \frac{dt}{2}\left(V_i^n\theta_i^n + V_i^*\theta_i^*\right), \tag{5.12}$$

that produces the solution at $t_{n+1}$.

## 5.2 Wall boundary conditions

To complete the time integration, it is necessary to enforce the boundary conditions in each time iteration cycle. For inviscid compressible flow, non-penetration at the solid walls is the usual velocity boundary condition. The temperature gradient at the adiabatic wall should be zero as can be derived from the momentum equation. The inlet and outlet boundary conditions need to be treated carefully by allowing the wave properties to pass through the boundaries without reflecting waves back into the computational domain in compressible flow.

The boundary conditions can be generalized into three categories: non-penetration wall, zero temperature gradient wall, and inlet or outlet conditions. Implementation of the boundary conditions is not transparent due to the nature of grid-free

methods. The two types of wall boundary conditions are discussed in this section, while the inlet and outlet boundary conditions are discussed in the next section. The no-slip wall will not be addressed here because the scope of this research is limited to inviscid flow.

### 5.2.1 Non-penetration wall

The non-penetration wall

$$\mathbf{V} \cdot \mathbf{n} = 0, \text{at the wall}$$

is a very common velocity type boundary condition in the computation of a bounded domain. The velocity directly recovered from the dilatation distribution generally does not satisfy the no-penetration condition at the wall (see Chapter 3 for details). It can be enforced by super-imposing a potential velocity field to cancel the normal velocity components from the contribution of dilatation field. The *Boundary Element Method* is used to produce the potential velocity field. The details are illustrated in Chapter 3.

### 5.2.2 Zero-gradient wall

Another usual boundary condition in compressible flow is a zero gradient at the wall,

$$\frac{\partial \phi}{\partial n} = 0, \text{at wall.} \tag{5.13}$$

As mentioned earlier, the derivatives are evaluated by using moving least square fitting (MLS) in this research, so when the interpolation window moves near the boundary, MLS will be biased toward the interior elements unless a special treatment (see Section 4.2) is done.

*Particle reflection* is used to make the interpolation window full and enforce the zero gradient condition at wall. Particle reflection can be easily implemented on straight boundaries as the sides of a rectangle.

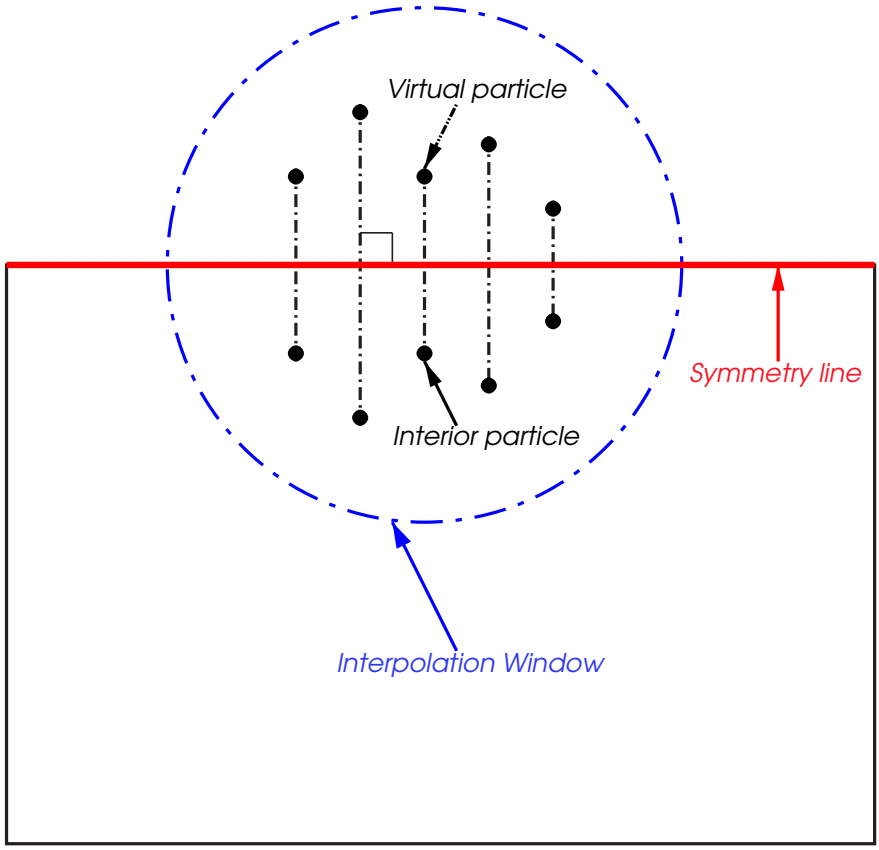



Figure 5.1: Particle reflection on a straight boundary.

In Figure 5.1, the zero gradient wall has been set as the symmetry line $y = y_{wall}$ and the interior elements $x_i, y_i$ are reflected outside to positions $x_i^* = x_i, y_i^* = 2y_{wall} - y_i$. The values of reflected virtual elements are set to be equal to their corresponding interior elements $f_i^* = f_i$ to satisfy the zero normal gradient condition at wall. In the simpler case of 1D compressible flow, for an interior element $(x_i, f_i)$ near the boundary $x = x_{wall}$, the virtual reflected element can be created as $x_i^* = 2x_{wall} - x_i, f_i^* = f_i$ to enforce the zero gradient boundary

condition at the wall.

In the case of curved boundaries, a coordinate transformation is used first to transform the *physical plane* containing Lagrangian elements to a rectangular space called the *computational plane* as shown in Figure 5.2.

The coordinate variables $(x, y)$ in the physical plane are transformed into a new set of coordinates $(\xi, \eta)$ in the computational space. The transformation is

$$\xi = \xi(x, y), \tag{5.14}$$

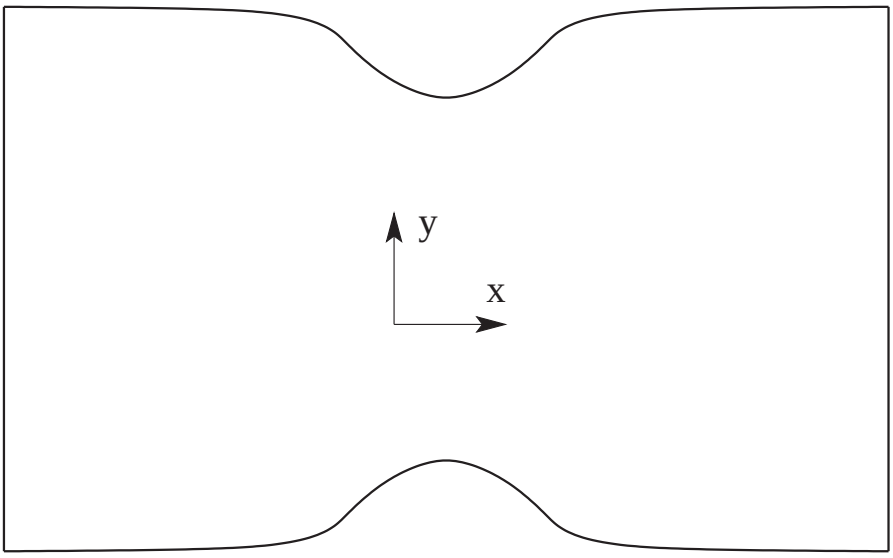
$$\eta = \eta(x, y). \tag{5.15}$$

with the reverse transformation

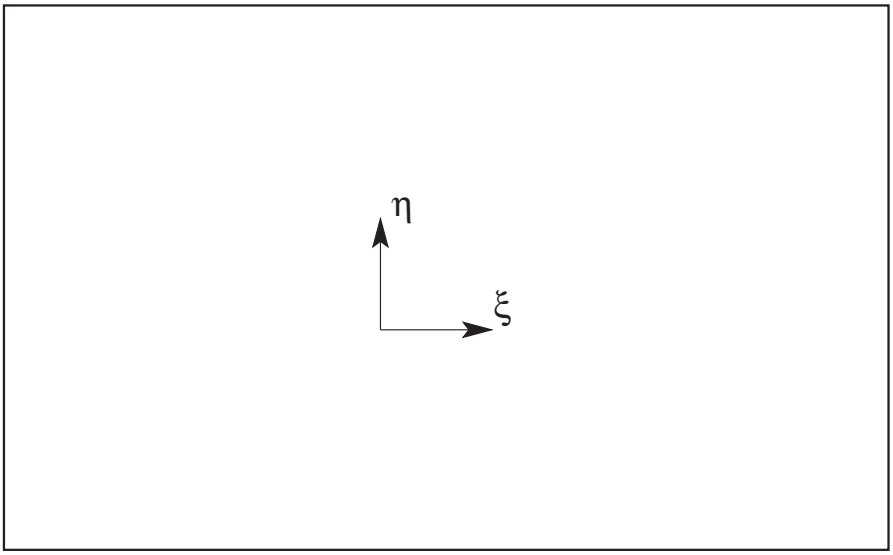
$$x = x(\xi, \eta), \tag{5.16}$$

$$y = y(\xi, \eta). \tag{5.17}$$

The above transformations (5.14) - (5.17) build an exact one-to-one conformal mapping between the physical plane and the computational plane. The physical plane is transformed into an appropriate, local orthogonal rectangular domain. Therefore, the boundary conditions, equation (5.13) can be easily enforced in the computational plane by using *particle reflection* as shown in Fig. 5.2. The derivatives are evaluated in the computational plane $(\xi, \eta)$. However the governing equations are defined in the physical plane. A set of transformation derived from the chain rule is needed to transform the derivatives from $(\xi, \eta)$ space to




(a) physical plane




(b) computational plane

Figure 5.2: Coordinate transformation.

$(x, y)$ space to march the time integration of the original governing equations.

$$\frac{\partial}{\partial x} = \frac{\partial}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial}{\partial \eta}\frac{\partial \eta}{\partial x} \tag{5.18}$$

$$\frac{\partial}{\partial y} = \frac{\partial}{\partial \xi}\frac{\partial \xi}{\partial y} + \frac{\partial}{\partial \eta}\frac{\partial \eta}{\partial y} \tag{5.19}$$

$$\begin{aligned}\frac{\partial^2}{\partial x^2} &= \frac{\partial}{\partial \xi}\frac{\partial^2 \xi}{\partial x^2} + \frac{\partial}{\partial \eta}\frac{\partial^2 \eta}{\partial x^2} + \frac{\partial^2}{\partial \xi^2}(\frac{\partial \xi}{\partial x})^2 \\ &\quad + \frac{\partial^2}{\partial \eta^2}(\frac{\partial \eta}{\partial x})^2 + 2\frac{\partial^2}{\partial \eta \partial \xi}\frac{\partial \eta}{\partial x}\frac{\partial \xi}{\partial x}\end{aligned} \tag{5.20}$$

$$\begin{aligned}\frac{\partial^2}{\partial y^2} &= \frac{\partial}{\partial \xi}\frac{\partial^2 \xi}{\partial y^2} + \frac{\partial}{\partial \eta}\frac{\partial^2 \eta}{\partial y^2} + \frac{\partial^2}{\partial \xi^2}(\frac{\partial \xi}{\partial y})^2 \\ &\quad + \frac{\partial^2}{\partial \eta^2}(\frac{\partial \eta}{\partial y})^2 + 2\frac{\partial^2}{\partial \eta \partial \xi}\frac{\partial \eta}{\partial y}\frac{\partial \xi}{\partial y}\end{aligned} \tag{5.21}$$

The above derivative transformations depend on having values for terms such as $\partial\xi/\partial x$, $\partial\xi/\partial y$, $\partial\eta/\partial x$, $\partial\eta/\partial y$. These terms are called *metrics*. For the flow of interest in this study an orthogonal boundary fitted coordinate system is readily found in the system of potential and streamlines belonging to the potential flow solution for the identical geometry. Lines of constant $\xi(x, y)$ in the computational plane denote streamlines and lines of constant $\eta(x, y)$ are potential lines. The top and bottom walls of the curved physical domain are clearly streamlines so they will coincide with lines of constant $\xi$.

Thus, letting $(\hat{u}, \hat{v})$ denote the potential flow velocities in the physical plane, then the following relationships exist for the constant potential and streamlines:

$$\frac{\partial \xi(x,y)}{\partial x} = -\hat{v}, \qquad \frac{\partial \xi(x,y)}{\partial y} = \hat{u}, \tag{5.22}$$

$$\frac{\partial \eta(x,y)}{\partial x} = \hat{u}, \qquad \frac{\partial \eta(x,y)}{\partial y} = \hat{v}. \tag{5.23}$$

Since the potential flow velocities are divergence free and vorticity free along

with (5.22) and (5.23), it leads to:

$$\frac{\partial^2 \xi}{\partial x^2} + \frac{\partial^2 \xi}{\partial y^2} = 0 \tag{5.24}$$

$$\frac{\partial^2 \eta}{\partial x^2} + \frac{\partial^2 \eta}{\partial y^2} = 0 \tag{5.25}$$

The transformations between velocity components $(u_i, v_i)$ in the physical $x-y$ coordinates and $(\tilde{u_i}, \tilde{v_i})$ in the computational $\xi - \eta$ coordinates are:

$$\begin{pmatrix} \tilde{u_i} \\ \tilde{v_i} \end{pmatrix} = \frac{1}{q_i} \begin{pmatrix} \hat{u_i} & \hat{v_i} \\ -\hat{v_i} & \hat{u_i} \end{pmatrix} \begin{pmatrix} u_i \\ v_i \end{pmatrix}, \begin{pmatrix} u_i \\ v_i \end{pmatrix} = \begin{pmatrix} \hat{u_i} & -\hat{v_i} \\ \hat{v_i} & \hat{u_i} \end{pmatrix} \begin{pmatrix} \tilde{u_i} \\ \tilde{v_i} \end{pmatrix} \tag{5.26}$$

where $q_i \equiv \hat{u_i}^2 + \hat{v_i}^2$. In the event that $(u_i, v_i) = (\hat{u_i}, \hat{v_i})$ then clearly $\tilde{u_i} = 1$ and $\tilde{v_i} = 0$.

Meanwhile the dilatation definition $\theta = \partial u/\partial x + \partial v/\partial y$ maps into

$$\frac{\partial \tilde{u}}{\partial \eta} + \frac{\partial \tilde{v}}{\partial \xi} = \frac{\theta}{q}. \tag{5.27}$$

From (5.18) - (5.27), the dilatation equation (5.2) is transformed to the $\xi - \eta$ coordinate system:

$$\frac{D\theta_i}{Dt} = q_i \left( \tilde{u_i} \frac{\partial \theta_i}{\partial \eta} + \tilde{v_i} \frac{\partial \theta_i}{\partial \xi} \right) - q_i \left( \frac{\partial^2 H_i}{\partial \xi^2} + \frac{\partial^2 H_i}{\partial \eta^2} \right) \tag{5.28}$$

where $H_i = \left( \frac{u_i^2 + v_i^2}{2} + \frac{1}{\gamma - 1} T_i \right)$.

The transformed dilatation equation (5.28) is solved in the computational rectangular domain. The derivative terms involving $\theta$ and $H$ can be evaluated via MLS with the particle reflection on the straight boundaries. The coordinates $(\xi_i, \eta_i)$ in the computational plane are calculated by numerically integrating the equations (5.22) and (5.23). The potential velocity components $(\hat{u_i}, \hat{v_i})$ are given by the boundary element methods in the physical plane.

## 5.3 Inlet and outlet conditions

The Lagrangian elements are deployed in the computational domain. For elements close to the inlet and outlet a complete set of data with which to make the least square fits are not available from the elements lying in the computational domain. Thus a special treatment of the inlet and outlet boundary conditions is necessary. To get a collection of symmetrically placed data for fits at inlet and outlet it is necessary to use data outside the computational domain. A means of acquiring such data is offered by the characteristic wave structures of compressible flow.

In compressible flow, there are right and left traveling waves heading downstream and upstream, respectively. When they move across the inlet and outlet boundaries, flow properties such as: dilatation, vorticity, density and temperature outside the computational domain are stimulated away from their quiescent states as part of the effects of traveling waves. The velocity, temperature, density, and pressure near the inlet and outlet change from the values of the upstream potential flow when the outward traveling waves move through.

When outward traveling waves move across the inlet and outlet boundaries, they bring the dilatation out of the computational domain. This is of special concern for the present scheme since for the velocity evaluation it is necessary to develop a technique that takes into account the presence of outside dilatation despite our interest lying only in the computational region. More importantly is that the velocity computation must be done without explicitly providing elements to keep track of this part of the dilatation field. Otherwise there would be no end to the extent of the computational domain. This problem is similar in principle to that faced in incompressible vortex methods when vorticity lying outside

the region of interest must nevertheless be accounted for when determining the velocity field [8].

Similar to traditional grid-based numerical schemes, the inlet and outlet boundary conditions are constructed by using the properties of the moving waves. The outward moving waves passing through the boundaries should not reflect nonphysical waves back into the computational region. For the derivative evaluations, the outward moving wave properties are used to create data outside of the computational domain to fill up the interpolation windows of the moving least square method. A successful technique has been developed for acquiring the necessary data. It is first illustrated for the quasi-1D case and then the 2D case is discussed.

In fact, a means of acquiring such data is offered by the wave structure of solutions near the inlet and outlet where the nozzle area is a constant so standard results apply from the one-dimensional gas dynamics equations. In one-dimensional inviscid compressible flow, the hyperbolic system has right and left traveling characteristic families, $x^+$ and $x^-$, respectively, determined as the solution to

$$\frac{dx^+}{dt} = u + c \tag{5.29}$$

$$\frac{dx^-}{dt} = u - c, \tag{5.30}$$

where $c$ is the sound speed. The Riemann invariants of the characteristic waves are

$$R^+ \equiv \frac{u}{2} + \frac{c}{\gamma - 1} \tag{5.31}$$

$$R^- \equiv \frac{u}{2} - \frac{c}{\gamma - 1}. \tag{5.32}$$

The Riemann invariant $R^+$ is constant on the characteristic waves $x^+$ while $R^-$

stays constant on $x^-$.

In the application to subsonic nozzle flow, the sound speed $c$ is equivalent to $\sqrt{T}$ using the scalings in the governing equations (5.1)-(5.4). In the vicinity of the outlet $x = x_r$, the characteristics moving toward the duct contraction (i.e., in the negative x direction) are in the $x^-$ family. Each of these originates from flow far upstream with velocity $u = u_0$ and temperature $T = 1$, so they have the same value of $R^-$, where

$$R^- = \frac{u_0}{2} - \frac{1}{\gamma - 1}. \tag{5.33}$$

The velocity and temperature in this region are thus related via

$$\frac{u}{2} - \frac{\sqrt{T}}{\gamma - 1} = \frac{u_0}{2} - \frac{1}{\gamma - 1}. \tag{5.34}$$

Equation (5.34) provides a means of obtaining the temperature at the outlet, $T_r$ once the velocity $u_r$ is computed. In fact,

$$T_r = \left(1 + \frac{(u_r - u_0)(\gamma - 1)}{2}\right)^2. \tag{5.35}$$

The fact that $R^+$ is constant along right moving characteristics proceeding away from the contraction, together with the validity of equation (5.34) implies that velocity and temperature are constants on the $x^+$ families leaving from the outlet at any given time. These characteristic lines are therefore straight with slope given by $u_r + \sqrt{T_r}$. They provide a means for recreating the temperature distribution outside of the computational domain $x > x_r$ for purposes of supplying the data needed for the moving least square fit of $T$ used in computing $\partial^2 T_i^n / x^2$.

The procedure used in the quasi-1D flow is illustrated in Figure 5.3 where it is seen that the $x^+$ characteristic family originating at $x_r$ at earlier times $t_{n-1}, t_{n-2}, ...$ and so forth are extended forward in time, where they occupy locations $x_r^{n,k} = x_r + (n-k)dt(u_r^k + \sqrt{T_r^k}), k = n, (n-1), ...$ and have velocity

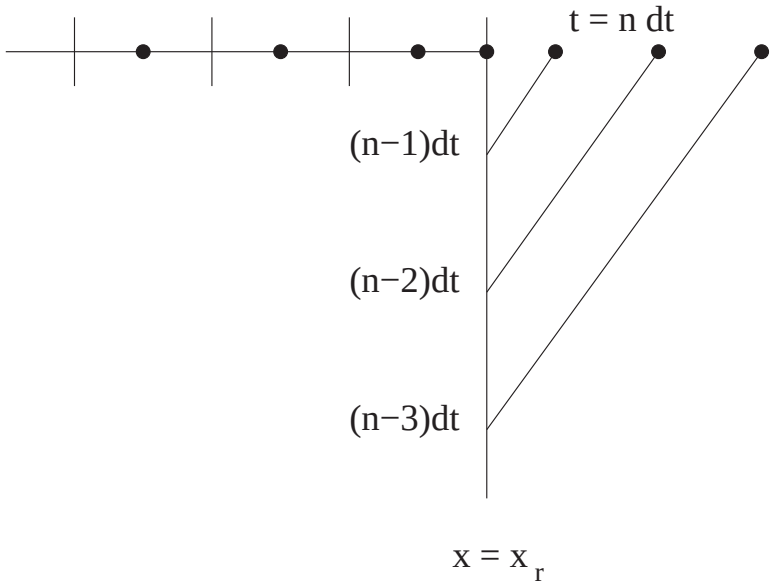



Figure 5.3: Filled circles denote the locations of $T$ values that form the basis for a least-square fit of $T$ that near the outlet boundary. Characteristic lines emanating from $x = x_r$ at times preceding $t_n$ are shown as diagonal lines.

$u_r^k$, temperatures $T_r^k$, and dilatation $\theta_r^k$. For a typical time step $dt$ used in the time integration, the points $x_r^{n,k}$ are more closely spaced than the Lagrangian elements, so the actual data needed for the least-square fits are obtained from the data pairs $(x_r^{n,k}, T_r^{n,k})$, $(x_r^{n,k}, u_r^{n,k})$, $(x_r^{n,k}, \theta_r^{n,k})$ by interpolation over points spaced equivalent to that of the initial element distribution.

Without giving the details it may be shown that similar considerations apply to obtaining data for the moving least square method ahead of the inlet boundary. In this case, analogously to (5.35) there is

$$T_l = (1 - \frac{(u_l - u_0)(\gamma - 1)}{2})^2, \tag{5.36}$$

a relation that may be used after $u_l$ is first computed. Moreover, the least square fits near the inlet $x = x_l$ are aided by data obtained by interpolation from temperature, velocity and dilatation associated with the locations

$x_l^{n,k} = x_l + (n-k)dt(u_l^k - \sqrt{T_l^k}), k = n, (n-1), ...$

Finally, it should be noted that during the startup phase of the calculation, before disturbances created by the constriction reach the boundaries, it is assumed that

$$u_l^k = u_r^k = u_0, \tag{5.37}$$

$$T_l^k = T_r^k = 1, \tag{5.38}$$

$$\theta_l^k = \theta_r^k = 0, \tag{5.39}$$

for $k = 0, -1, -2, ...$ to as early a time as is necessary for the least square fit. Once $\frac{\partial^2 T_i^n}{\partial x^2}$ is computed, the solution to the quasi-1D nozzle flow can be advanced through the first step of the second order Runge-Kutta scheme (similar to (5.5-5.8) in 2D) given by

$$X_i^* = X_i^n + dtu_i^n \tag{5.40}$$

$$\theta_i^* = \theta_i^n - dt\left((\theta_i^n)^2 + \frac{1}{\gamma-1}\frac{\partial^2 T_i^n}{\partial x^2}\right) \tag{5.41}$$

$$T_i^* = T_i^n - dt(\gamma-1)T_i^n(\theta_i^n + u_i^n\alpha_i^n) \tag{5.42}$$

$$h_i^* = h_i^n + dth_i^n\theta_i^n, \tag{5.43}$$

yielding the provisional values $X_i^*, \theta_i^*, u_i^*, T_i^*$ and area $h_i^*$, of the $i^{th}$ element ($i = 1, ..., N$) for the solution at $t + dt$.

As part of the second step of the Runge-Kutta scheme it is necessary to evaluate $u_i^*$, $\alpha_i^*$ and $\frac{\partial^2 T_i^*}{\partial x^2}$ using the provisional solution obtained from (5.40) - (5.43). The computation of $\alpha_i^*$ is straightforward using the values of $X_i^*$. In addition, $\frac{\partial^2 T_i^*}{\partial x^2}$ may be computed using the least-square approach applied to the data points $(X_i^*, T_i^*)$. Near the boundary this data is supplemented by information about $T$ outside the domain that is reconstructed from the local wave structure.

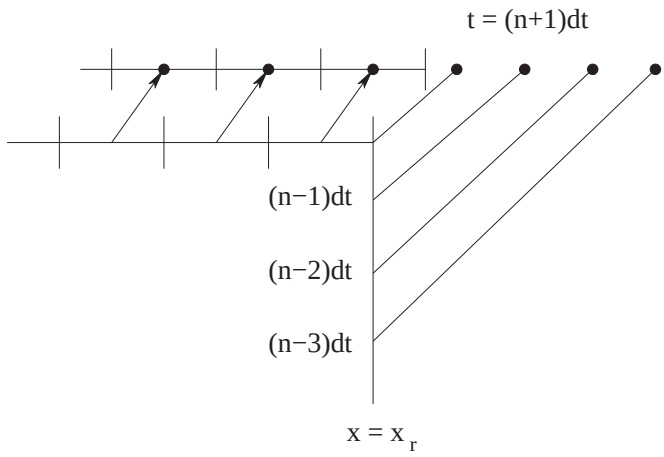



Figure 5.4: Filled circles denote the locations of $T$ values that form the basis for a least-square fit of $T$ near the outlet boundary at the start of the second step of the Runge-Kutta scheme. Movement of the dilatation elements to positions $X_i^*$ is also indicated.

In particular, as depicted in Figure 5.4 for the outlet boundary, $T$ values can be established at locations determined by the characteristics departing from $x_l$ and $x_r$ at earlier times and extended until $t_{n+1}$. The discussion of the computation of $u_i^*$ is deferred to the end of this section.

After having the derivatives of the necessary provisional variables from the first step of Runge-Kutta scheme, it is now possible to effect the second step of the Runge-Kutta algorithm:

$$X_i^{n+1} = X_i^n + \frac{dt}{2}(U_i^n + U_i^*) \tag{5.44}$$

$$\theta_i^{n+1} = \theta_i^n - \frac{dt}{2}\Big((\theta_i^n)^2 + (\theta_i^*)^2 + \frac{1}{\gamma - 1}\big(\frac{\partial^2 T_i^n}{\partial x^2} + \frac{\partial^2 T_i^*}{\partial x^2}\big)\Big) \tag{5.45}$$

$$T_i^{n+1} = T_i^n - \frac{dt(\gamma - 1)}{2}\left(T_i^n(\theta_i^n + u_i^n \alpha_i^n) + T_i^*(\theta_i^* + u_i^* \alpha_i^*)\right) \tag{5.46}$$

$$h_i^{n+1} = h_i^n + \frac{dt}{2}\left(h_i^n \theta_i^n + h_i^* \theta_i^*\right), \tag{5.47}$$

that produces the solution at $t_{n+1}$.

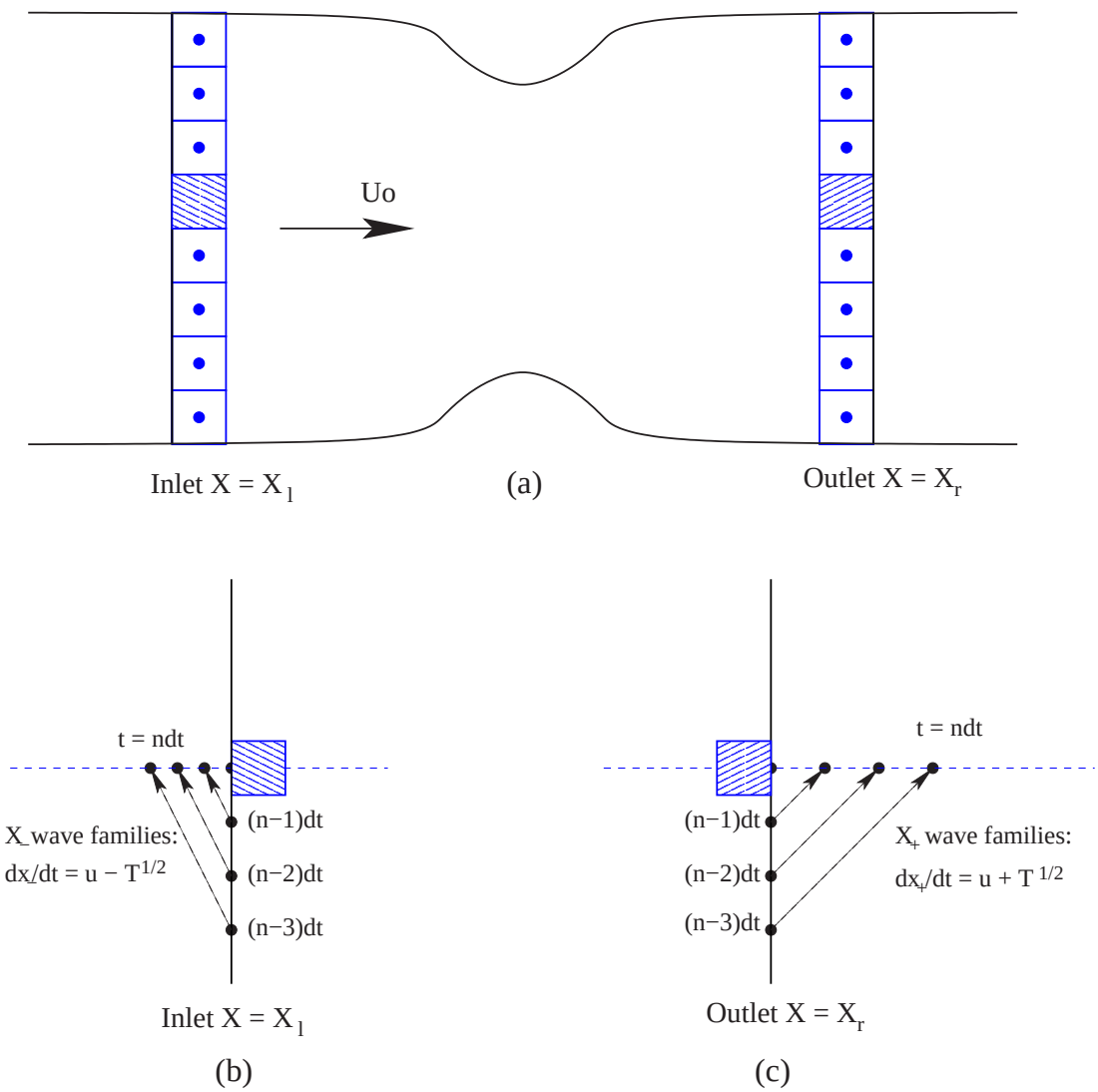



Figure 5.5: The properties of waves at $t = ndt$ originated from the inlet and outlet boundaries at previous time steps, (a) discretization of inlet and outlet planes, (b) wave families at inlet, (c) wave families at outlet.

The generalization of the above quasi-1D procedure to 2D is illustrated in Figure 5.5. The outlet is broken up into a set of small pieces each of which has constant velocity and temperature. The one-dimensional assumption is made on each small piece. For example, the $x^+$ characteristic family originating at $x_r$ at earlier times $t_{n-1}, t_{n-2}, ...$ and so forth are extended forward in time, where they occupy locations $x_r^{n,k} = x_r + (n-k)dt(u_r^k + \sqrt{T_r^k}), k = n, (n-1), ...$ and have velocity $u_r^k$, temperatures $T_r^k$, and dilatation $\theta_r^k$. The details of this and other aspects of the 2D procedure match those discussed previously for the quasi-1D case and will not be repeated.

As in the case of the quasi-1D flow, during the startup phase of the calculation, before disturbances created by the constriction reach the boundaries, it is assumed that $u$, $\theta$ and $T$ stay at their initial states (see equations (5.37) - 5.39)).

As previously mentioned, for the computation of $u_i^*$, the dilatation carried outside of the computational domain by the characteristic waves must be taken into account. This calculation must be sensitive to the effect of the motion of the elements and characteristics that has taken place during the first step of the Runge-Kutta scheme. We first discuss a means of accommodating this for the quasi-1D case. Two oberservations need to be made:

1. While $\Delta_r^n$, for example, represents the integral of $\theta$ over the region $x \geq x_r$ at $t_n$, it also represents the integral of $\theta$ over the region $x \geq x_r + dt(u_r^n + \sqrt{T_r^n})$ at time $t_{n+1}$. This follows from the constancy of $u$ on the right moving characteristics leaving from $x_r$ and the definitions in (3.4) and (3.8). A similar observation applies at the left boundary: thus $\Delta_l^n$ represents the integral of $\theta$ in $x \leq x_l$ at $t_n$ and also $x \leq x_l + dt(u_l^n - \sqrt{T_l^n})$ at time $t_{n+1}$.

2. The computational elements are no longer flush with $x_l$ and $x_r$ after the

first Runge-Kutta step. In particular, the $N^{th}$ element now lies so that it extends past the boundary at $x_r$ (i.e. $X_N^* + h_N^*/2 > x_r$) as seen in Figure 5.6. Similarly, a gap has formed between $x_l$ and $X_1^* - h_1^*/2$.

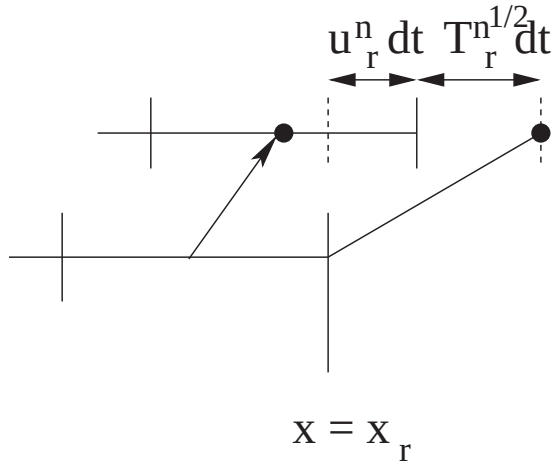



Figure 5.6: The $N^{th}$ element has moved past $x_r$ a distance given approximately by $u_r^n dt$. The characteristic departing from $x_r$ arrives at $x_r + dt(u_r^n + \sqrt{T_r^n})$ after time $dt$.

A simple way of proceeding at this point is to recognize that the formalism leading to (3.6)-(3.8) is equally valid if it is applied to a temporary computational domain encompassing the left and rightmost points of the dilatation elements at positions $X_i^*$. In this scheme, quantities $\Delta_l^*$ and $\Delta_r^*$ are required that provide for dilatation lying in $x \leq X_1^* - h_1^*/2$ and $x \geq X_N^* + h_N^*/2$, respectively, at time $t_{n+1}$. In view of the first observation above, these quantities are readily computed from $\Delta_l$ and $\Delta_r$, by adding appropriate corrections that approximate $\int \theta dx$ over the gap regions that form between $X_N^* + h_N^*/2$ and $x_r + dt(u_r^n + \sqrt{T_r^n})$ at the outflow boundary and $x_l + dt(u_l^n - \sqrt{T_l^n})$ and $X_1^* - h_1^*/2$ at the inflow boundary. The situation at the outlet is illustrated in Figure 5.6. Since we want $\Delta_r^*$ to equal

$\int \theta dx$ for $x \geq X_N^* + h_N^*/2$, it is clear that an approximate to this is

$$\Delta_r^* = \Delta_r^n + (dt\sqrt{T_r^n})\theta^*|_{x=x_r+u_r^n dt+\sqrt{T_r^n}dt/2}, \tag{5.48}$$

where $\sqrt{T_r^n}dt$ is the size of the gap between the end of the $N^{th}$ element and the characteristic that left from $x_r$ at time $t_n$. A similar consideration yields the relation

$$\Delta_l^* = \Delta_l^n + (dt\sqrt{T_l^n})\theta^*|_{x=x_l+u_l^n dt-\sqrt{T_l^n}dt/2}, \tag{5.49}$$

at the left boundary. The required values of $\theta^*$ in (5.48) and (5.49) are found via interpolation using the set of $\theta_i^*$ at $X_i^*$ together with $\theta$ values outside the computational domain. The latter are easily computed from the simple finite difference approximations

$$\theta_l^{n+1,k} \approx \frac{u_l^k - u_l^{k-1}}{\Delta x_l^{n+1,k}} \tag{5.50}$$

and

$$\theta_r^{n+1,k} \approx \frac{u_r^{k-1} - u_r^k}{\Delta x_r^{n+1,k}} \tag{5.51}$$

for $k = n, n-1, ...$ where $\Delta x_l^{n+1,k} \equiv x_l^{n+1,k} - x_l^{n+1,k-1}$ and $\Delta x_r^{n+1,k} \equiv x_r^{n+1,k-1} - x_r^{n+1,k}$. Equation (5.50) applies at the points $x_l^{n+1,k} + \Delta x_l^{n+1,k}/2$ while (5.51) is at the points $x_r^{n+1,k} + \Delta x_r^{n+1,k}/2$. The accuracy of these approximations is aided by the small magnitudes of $\Delta x_l^{n+1,k}$ and $\Delta x_r^{n+1,k}$.

For higher dimensions, the velocity field computed from the procedure illustrated in Chapter 3 needs a small change to account for the dilatations outside of the computational domain to satisfy the velocity boundary conditions at the inlet and outlet. That is essentially to add or subtract a constant, which is the average difference between the inlet and outlet velocity boundary conditions and the computed velocities at the inlet and outlet boundaries from the interior dilatations, to the previous velocity field similar to the quasi-1D case. However, as

the solution converges towards the steady state, the constant decreases to zero.

## 5.4 Modification, subdivision and merging of Lagrangian elements

The computational domain is initially fully filled with elements (see Figure 5.8(a)). Due to the nature of Lagrangian methods, all elements are convected downstream by the local flow velocities. At the end of one time iteration, as shown in Figure 5.8(b), the first group of elements right next to the inlet are slightly away from the boundary. Small gaps appear between these elements and the inlet boundary. The dot-dash lines in Figure 5.8(b) are the new left bounds of the convected Lagrangian elements initially aligned with the inlet. Meanwhile the group of elements immediately next to the outlet are not flush with the outlet boundary either. Part of these elements are convected outside of the domain. As shown in Figure 5.8(b), the dot-dash lines are the new right bounds of those elements convected outside.

In order to keep the entire computational domain covered by elements at the beginning of each time iteration, it is necessary to adjust the volumes and positions of those elements closest to the inlet and outlet boundaries at the end of each time iteration cycle so they are flush with the boundaries. The details for the quasi-1D case are first illustrated and then the 2D case is discussed.

In the quasi-1D case, after the completion of the Runge-Kutta integration, it is necessary to compute $\Delta_l^{n+1}$ and $\Delta_r^{n+1}$ similar to the previous approach in calculating $\Delta_l^*$ and $\Delta_r^*$. Only this time the assumption is in effect that the ends of elements 1 and $N$ are exactly at $x_l$ and $x_r$, respectively, since they will be

so by the end of the time step. According to our first observation made in the computation of $\Delta_l^*$ and $\Delta_r^*$, $\Delta_l^{n+1}$ and $\Delta_r^{n+1}$ are computed by adding to $\Delta_l^n$ and $\Delta_r^n$, respectively, the amount of $\theta$ filling the region between the boundaries and the characteristics that moved out during the intervals $dt$. Using interpolation the following approximations are then suggested:

$$\Delta_l^{n+1} = \Delta_l^n + (dt\sqrt{T_l^n})\theta^{n+1}|_{x=x_l+u_l^n dt-\sqrt{T_l^n}dt/2} - (dtu_l^n)\theta^{n+1}|_{x=x_l+u_l^n dt/2}, \quad (5.52)$$

and

$$\Delta_r^{n+1} = \Delta_r^n + (dt\sqrt{T_r^n})\theta^{n+1}|_{x=x_r+u_r^n dt+\sqrt{T_r^n}dt/2} + (dtu_r^n)\theta^{n+1}|_{x=x_r+u_r^n dt/2}. \quad (5.53)$$

It now follows that $u_l^{n+1}$ and $u_r^{n+1}$ can be obtained from (3.7)-(3.8), respectively, while $T_r^{n+1}$ and $T_l^{n+1}$ are computed from (5.35) - (5.36).


Figure 5.7: At each time step the element closest to the inlet is lengthened so that its leftmost point is at the inlet: (a) after element moves, (b) revised element.

The simple adjustments of $X_1^{n+1}$ and $h_1^{n+1}$ illustrated in Figure 5.7 are done to enforce the condition $X_1 - h_1/2 = x_l$. This then necessitates a slight modification to $T_1^{n+1}$ and $\theta_1^{n+1}$ so that they correspond to the new position of the element center. The former is found from cubic interpolation over the updated temperature field. $\theta_1^{n+1}$ on the other hand is calculated in such a way as to preserve the total

integrated dilatation within the region between $x_l$ and $X_1 + h_1/2$. This leads to the equation

$$[\theta_1^{n+1} h_1^{n+1}]_{new} = [\theta_1^{n+1} h_1^{n+1}]_{old} + (u_l^n dt)\theta^{n+1}|_{x=(x_l+u_l^n dt/2)}, \tag{5.54}$$

which is solved for $\theta_1^{n+1}$ on the left-hand side. The second term on the right-hand side accounts for dilatation in the gap created by the first element moving to the right.

At the outlet boundary an opposite procedure is performed. Here, element $N$ is truncated so that its downstream end is at $x_r$. $X_N^{n+1}$ and $h_N^{n+1}$ are modified in an obvious way. $T_N^{n+1}$ is found by interpolation, while similar to (5.54) the velocity difference across the $N^{th}$ element is maintained before and after adjustment via the relation

$$[\theta_N^{n+1} h_N^{n+1}]_{new} = [\theta_N^{n+1} h_N^{n+1}]_{old} - (u_r^n dt)\theta^{n+1}|_{x=(x_r+u_r^n dt/2)}, \tag{5.55}$$

which is once again used to get $\theta^{n+1}$ on the left-hand side. The use of (5.54) and (5.55) or equivalent appears to be necessary. For example, determining the updated $\theta$ by interpolation in this instance can lead to unacceptable oscillations in the solution.

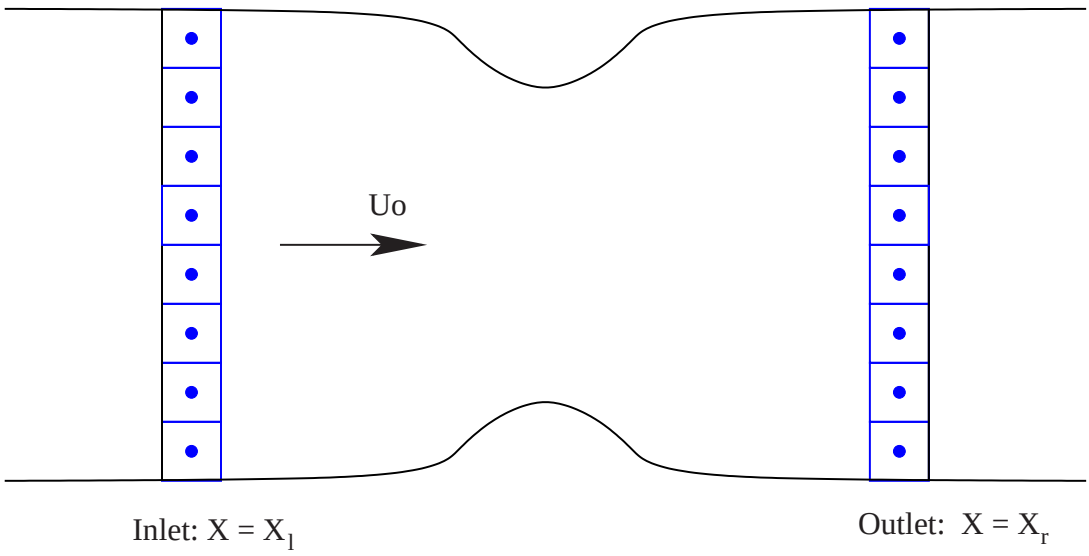
Since the element at the inlet boundary grows slightly in length after each time step it is necessary to periodically subdivide it when it increases beyond a given length. Similarly, the element at the outlet shrinks after each time step and it must occasionally be merged with its nearest neighbor when it is too short. The performance of the algorithm proves to be for the most part insensitive to the exact criterion used to determine when it is time for subdivision and merger. After element merger or division the position, length, temperature and dilatation of the new elements are determined very much the same way they are during

the boundary adjustment at each time step. At this point the integration step is complete and the calculation of a new time step can be initiated.

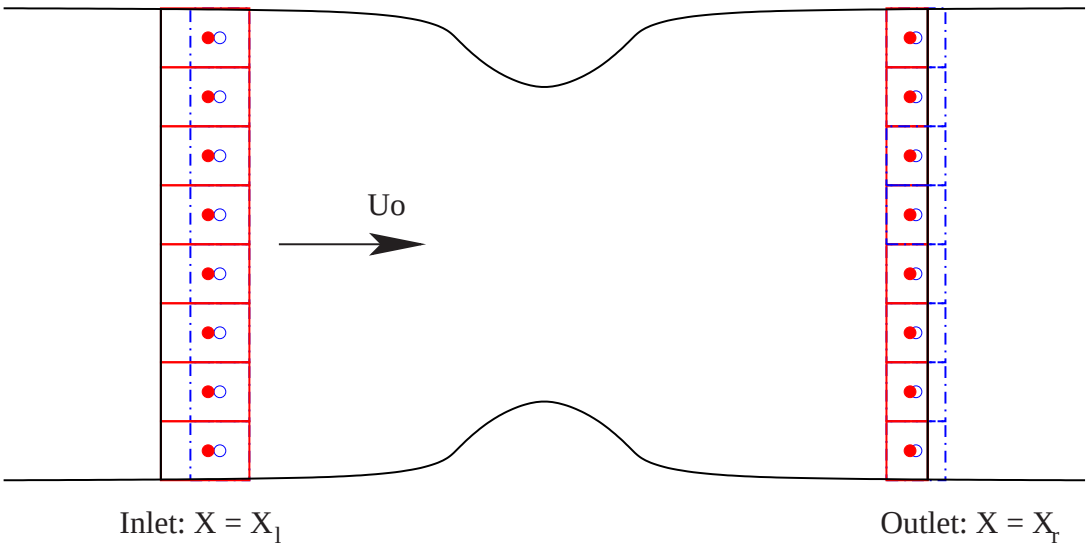
The generalization of the above quasi-1D procedure to the 2D case is shown below. The adjustment of elements closest to the inlet boundary, illustrated in Figure 5.8(b), is done to enforce the condition $X - dx/2 = x_l$ so the left (upstream) ends of these elements are extended to align with the inlet. The solid red lines in Figure 5.8(b) are the bounds of the new extended elements at the inlet. At the outlet boundary an opposite procedure is performed. The elements closest to the outlet boundary are truncated so that their right-ends (downstream) are exactly at the outlet to enforce the condition $X + dx/2 = x_r$. The solid red boxes at the outlet in Figure 5.8(b) show the new elements after trimming. The volumes and centers of these new modified elements are easily determined.

There are generally two means of assigning the temperature and dilatation to the modified elements. One choice is to slightly modify the temperature and dilatation they carry so that the new values correspond to the new positions of the element centers. The new $T$ and $\theta$ may be found from an interpolation over the updated temperature and dilatation within the computational domain plus the data given on the outward moving waves. This makes sure that the new assigned values continuously change with their neighbors and can eliminate the chance of bringing in any unnecessary oscillations.

Another way of deciding the properties of the modified elements that has been examined is the simpler method of keeping the same temperature and dilatation in the elements as they had before modification. Compared with the first method, it is found to introduce small oscillations due to small modifications in positions of the elements. On the other hand, in the case of the 2D calculation it is found to




(a) particles closest to the inlet and outlet boundaries at the begining of time step ndt.



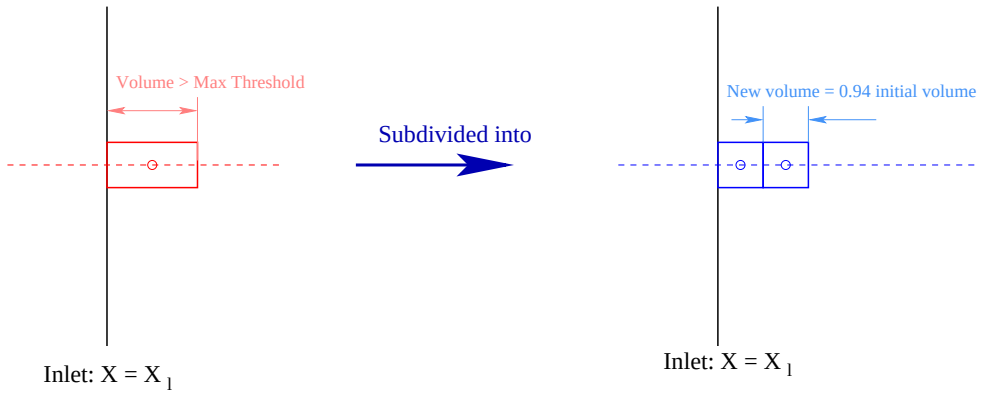

(b) modification of particles closest to the inlet and outlet boundaries at the end of time step ndt.

Figure 5.8: The modification of elements closest to the inlet and outlet boundaries.

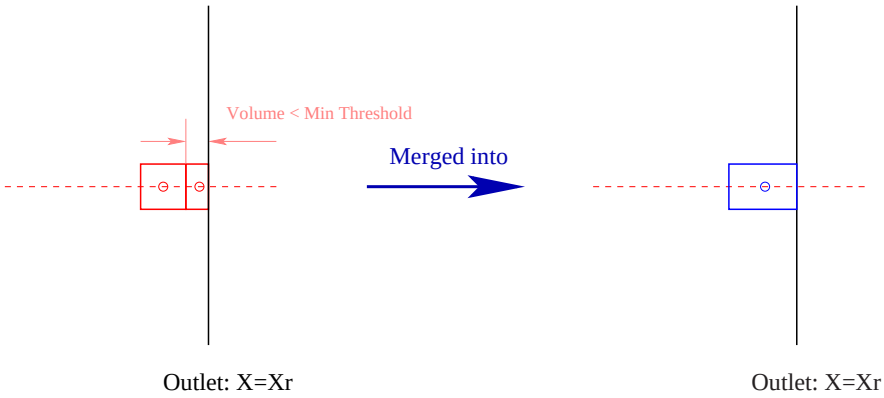
be very stable and the oscillations subside with the progress of the time iteration. In fact, the later method is used in the present research in 2D since it ultimately gives better results and conserves the total dilatation better than the first method depending on a cubic interpolation scheme.

When the gaps created between the boundary and the left-ends of the elements closest to the inlet are merged at the end of each time step, the element volumes grow slightly. It is thus necessary to periodically subdivide them when they grow beyond a given threshold volume in order to prevent the scheme from losing resolution. Similarly, the elements closest to the outlet shrink after each time step due to the modification procedure. They must occasionally be merged with their nearest neighbors when their volumes are too small, that is lower than a minimum threshold value.

Figure 5.9 shows the procedure of subdividing and merging of elements closest to the inlet and outlet boundaries. When an element grows too big and needs to be subdivided, a new element is added into the computational domain (Figure 5.9(a)). On the contrary, an element has to be merged with its nearest neighbor and deleted out of the computational domain when it shrinks too small (Figure 5.9(b)). After the element division or merger, the position, volume, temperature and dilatation of the elements involved can be determined by almost the same means as is used for the element modification at each time step. Thus, as before they can be found by using interpolation over elements and the outside data provided by the characteristic waves. However, it is found for the 2D case that this interpolation can lead to undesirable oscillations at the inlet and outlet boundaries. An alternative procedure that is more stable is to retain the values of temperature and dilatation when an element is divided in two. When an element




(a) subdivision of particles closest to the inlet boundary




(b) merger of particles closest to the outlet boundary

Figure 5.9: The subdivision and merging of elements closest to the inlet and outlet boundaries.

merges with a neighboring element, the temperature and dilatation of the larger element had before merger are assigned to the new element. In this work, the maximum threshold volume is taken to be 1.5 times the initial volume and the minimum is 0.6 times. When an element is subdivided into two elements at the inlet boundary, the volume of the downstream element is 0.94 times that of the initial volume while the remainder is taken by the other new element, as shown in Figure 5.9.

# Chapter 6

# Results

The success of applying the Lagrangian dilatation element method to the problem of the viscous Burgers equation was demonstrated in Chapter 2. How well this method can work in more complicated gas dynamics problems is investigated in this chapter. The treatment of boundary conditions plays a vital role in the performance of the numerical scheme. Besides inflow and outflow boundaries, solid walls are typically encountered in compressible flow problems. An enclosed tube with oscillating waves is first investigated to examine the wall boundary condition. In addition, the results of a study of the compressible flow in a subsonic nozzle are shown in this chapter. The two computational examples demonstrate the capabilities of the Lagrangian dilatation element method. It will be seen to be able to handle different types of boundary conditions and dynamically adjust the Lagrangian particles. The application of this method in capturing shock waves is given a cursory discussion at the end of the chapter. Future work to further develop the method so it can cover all compressible flows is also discussed at the end.

## 6.1 Oscillating waves in an enclosed tube

An oscillating wave in an enclosed tube is an interesting problem with which to examine the wall boundary conditions. Both the one and two-dimensional models of this flow are done using the dilatation element method. Both results are compared with grid-based methods.

### 6.1.1 1D results

Consider a unit length enclosed domain $[0, 1]$ filled initially with a still perfect gas with gas constant $R = 0.29$, and $\gamma = 1.4$. Assume it is stirred up suddenly by applying a density change $\rho(x, 0) = \cos(2\pi x) + 10$. The initial conditions of the flow are:

$$u(x, 0) = 0 \tag{6.1}$$

$$\theta(x, 0) = 0 \tag{6.2}$$

$$\rho(x, 0) = \cos(2\pi x) + 10 \tag{6.3}$$

$$T(x, 0) = 290. \tag{6.4}$$

We have computed the subsequent dynamics of this flow by using the grid-free dilatation element method with $N = 101$ Lagrangian particles and time step $\Delta t = 10^{-3}$. The governing equations (2.24) - (2.26) are integrated with the explicit Euler scheme from $t = 0$ to 0.072. The velocity recovery is accomplished similar to the viscous Burgers equation presented in Chapter 2. The moving least square fitting is used to compute the derivatives. Particle reflection developed in Chapters 2 and 5 is used to enforce the solid wall boundary conditions. Basically, the particles close to the boundaries are reflected outside and take the same

values as their symmetrically placed particles inside the domain. In this way the derivatives $T_x$ and $\rho_x$ are enforced to zero at the left and right boundaries.

In order to verify the results of the Lagrangian dilatation element method, the same problem is solved using the Godunov scheme [63]. The Eulerian governing equations for this problem are:

$$\frac{\partial \rho}{\partial t} + (\rho u)_x = 0 \tag{6.5}$$

$$\frac{\partial \rho u}{\partial t} + (\rho u^2 + p)_x = 0 \tag{6.6}$$

$$\frac{\partial E}{\partial t} + (u(E + p))_x = 0, \tag{6.7}$$

and the same initial conditions (6.1) - (6.4) are used. The domain is evenly divided into 200 fixed cells each with a length of 1/200. The time step $\Delta t = 10^{-4}$.

Figures 6.1 - 6.4 show comparisons of the results of velocity, density, temperature and pressure computed by the Lagrangian particle method and Godunov scheme. The predications of $u, \rho, T$ and $p$ by the particle method agree well with the Godunov scheme although the time step and element size used in the particle method are larger than those in the Godunov scheme.

### 6.1.2 2D results

The previous section showed that the dilatation element method can successfully solve the one-dimensional oscillating wave problem. Now we consider how it performs in higher dimensions. For this purpose the 1D oscillating wave problem is modified to the case of a circular domain centered at the origin $(0, 0)$ and having a radius $R = 0.5$. A perfect gas fills the circular domain having gas constants $R = 0.29$ and $\gamma = 1.4$. The gas is assumed to be at rest at the beginning and then suddenly is subjected to a density change. The initial conditions are similar

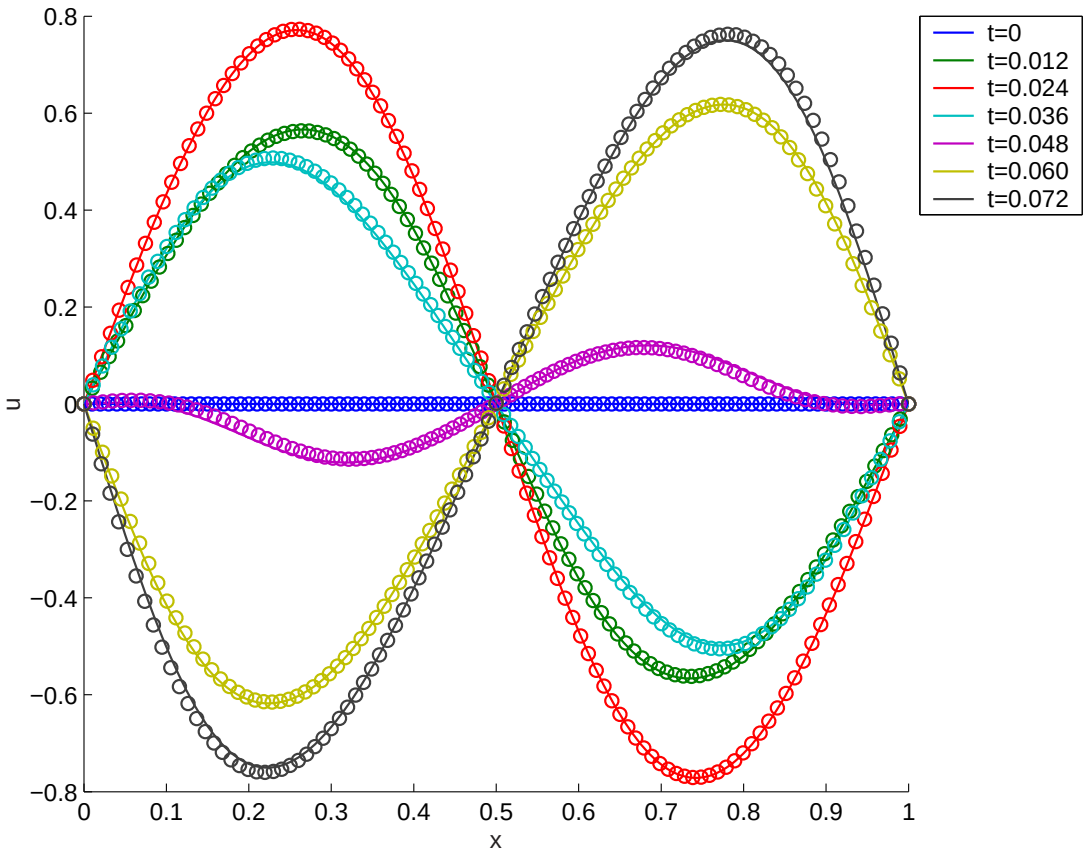



Figure 6.1: Comparison of velocity $u$ at different times; Godunov scheme(—), particle method (o).

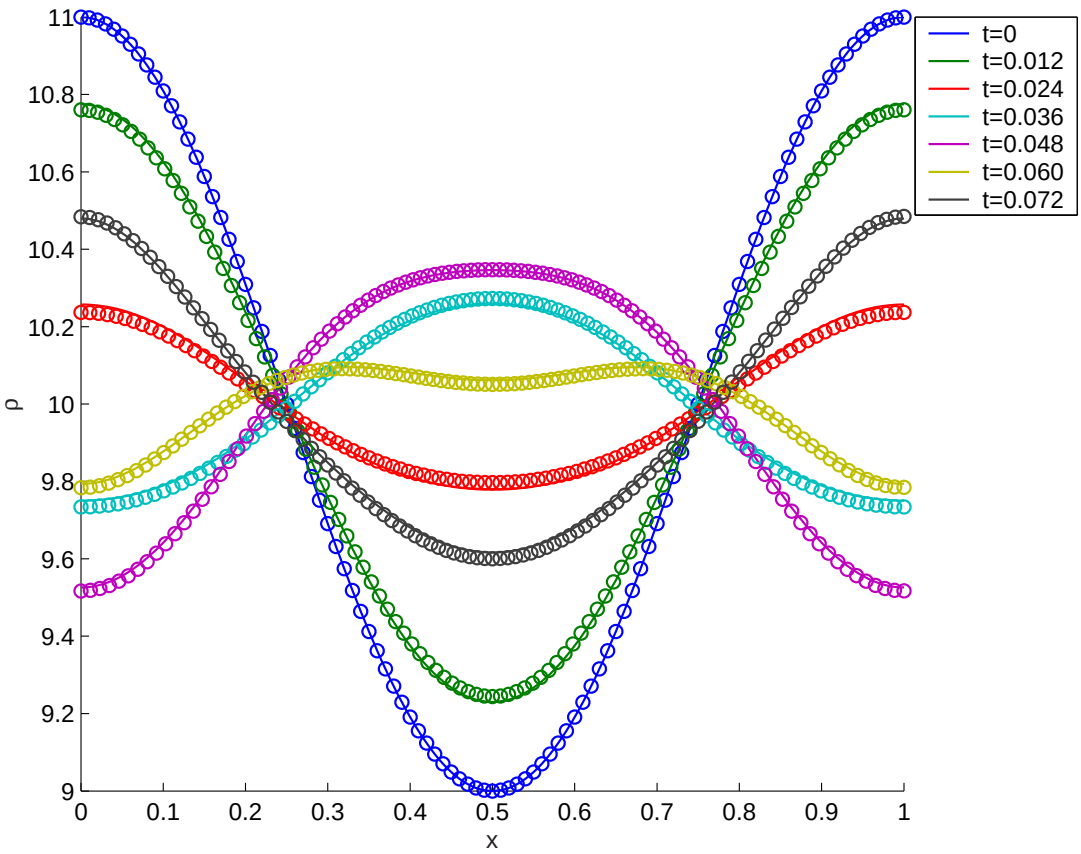



Figure 6.2: Comparison of density $\rho$ at different times; Godunov scheme(—), particle method (o).

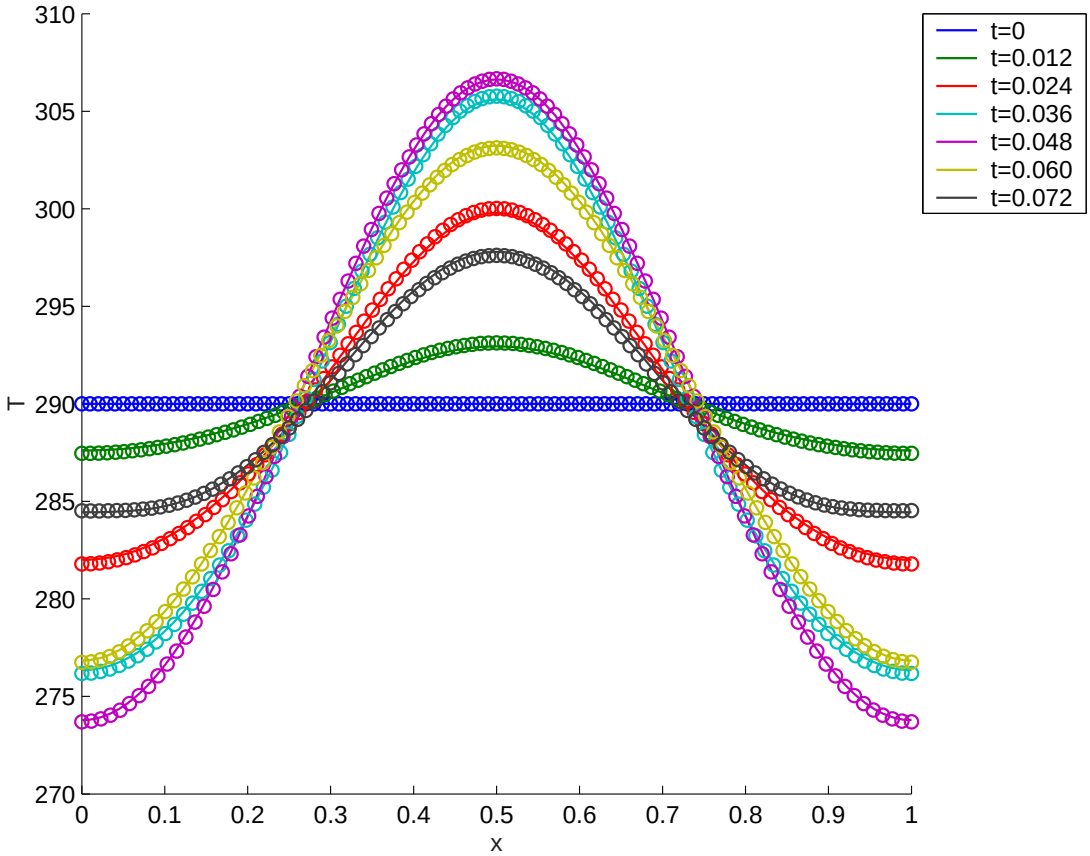



Figure 6.3: Comparison of temperature $T$ at different times; Godunov scheme(—), particle method (o).

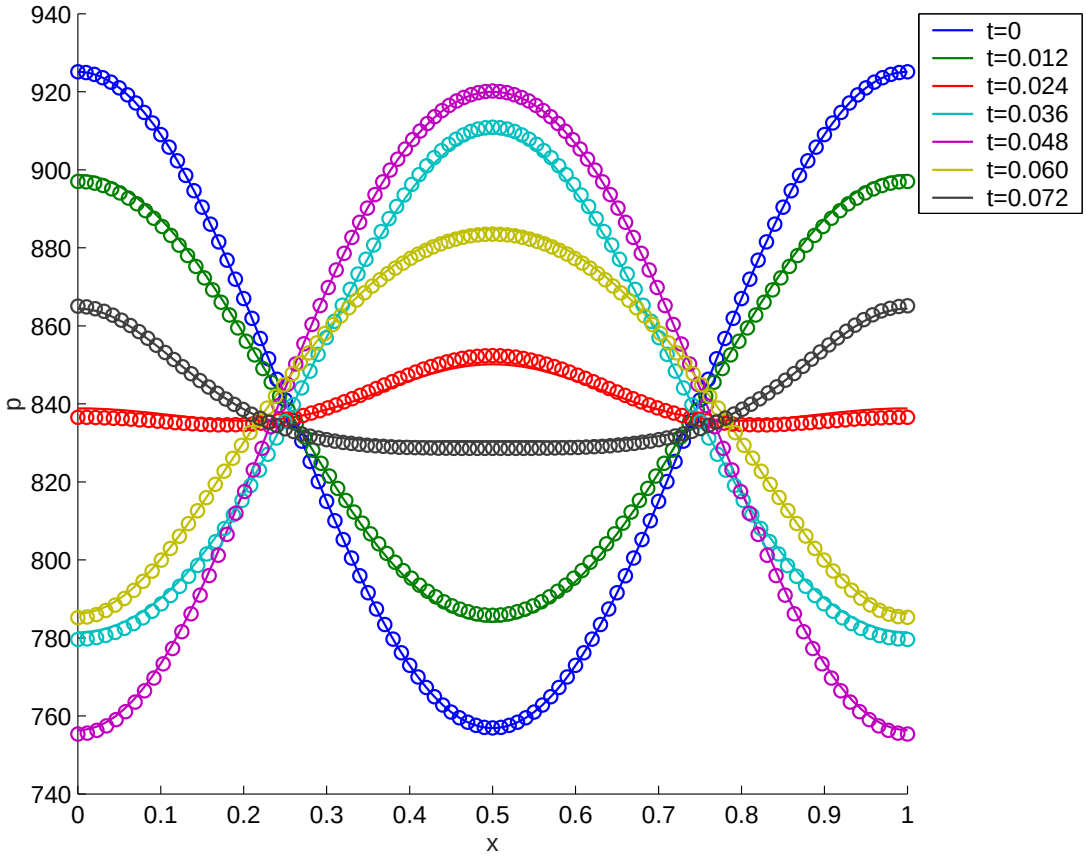



Figure 6.4: Comparison of pressure $p$ at different times; Godunov scheme(—), particle method (o).

to that of the previous 1D problem:

$$u(x,y) = 0 \tag{6.8}$$

$$v(x,y) = 0 \tag{6.9}$$

$$\theta(x,y) = 0 \tag{6.10}$$

$$\rho(x,y) = \cos(2\pi\sqrt{x^2+y^2}) + 10 \tag{6.11}$$

$$T(x,0) = 290, \tag{6.12}$$

in the region $\sqrt{x^2+y^2} \leq 0.5$.

This problem is solved using standard cartesian coordinates, so that the governing equations (2.29), (2.30), (2.32) apply. The circular domain is initially divided into $50 \times 60$ elements along the radial and angular directions respectively. Spatial derivatives are computed using moving least square fits as developed in Chapter 4. *Particle Reflection* is used to create virtual particles lying outside the domain in order to enforce the zero gradient condition of temperature and density. Assuming $(x_i, y_i)$ are the Lagrangian element centers in the computational domain, then their corresponding reflected element centers $(x_i^*, y_i^*)$ can be described as

$$x_i^* = \frac{x_i * (1.0 - \sqrt{x_i^2 + y_i^2})}{\sqrt{x_i^2 + y_i^2}}, \tag{6.13}$$

$$y_i^* = \frac{y_i * (1.0 - \sqrt{x_i^2 + y_i^2})}{\sqrt{x_i^2 + y_i^2}}. \tag{6.14}$$

The temperatures and densities $(T_i^*, \rho_i^*)$ at the centers of these image elements are set equal to the values $(T_i, \rho_i)$ at the centers of elements at$(x_i, y_i)$.

Since the derivatives of velocities appear in the dilatation governing equation (2.30), these also have to be computed by least-square fits, and image values have to be created. When *Particle Reflection* is used to compute the velocity

derivatives near the wall, the velocities $(u_i^*, v_i^*)$ at the reflected virtual positions are given the values:

$$u_i^* = -u_i; v_i^* = -v_i.$$

Note that these conditions assume radially symmetric flow. More general conditions are treated below in the case of the duct flow with constriction.

A 2nd order Runge-Kutta scheme is used to integrate the Lagrangian governing equations (2.29), (2.30), (2.32). The time step is set to $10^{-3}$. The particle positions are updated according to their local flow velocities, which are recovered by the FMM code developed in Chapter 3.

Figures 6.6 - 6.8 show the distributions of density, temperature, dilatation and velocity in a 2D enclosed circular tube from $t = 0.012$ to 0.072 computed by the grid-free dilatation element method. This same problem is also computed using a traditional grid-based Godunov scheme. In this case it is much more convenient to use cylindrical coordinates to describe the circular domain instead of cartesian coordinates. When cylindrical symmetry is considered, the angular direction can be ignored, and the governing equations become

$$\frac{\partial \rho}{\partial t} + (\rho u)_r = \frac{-\rho u}{r} \tag{6.15}$$

$$\frac{\partial \rho u}{\partial t} + (\rho u^2 + p)_r = \frac{-\rho u^2}{r} \tag{6.16}$$

$$\frac{\partial E}{\partial t} + (u(E+p))_r = \frac{-u(E+p)}{r}. \tag{6.17}$$

Note the above equations are different than the one-dimensional governing equations shown in the previous section. In particular, the right side of the equations have terms that account for the two-dimensional effects. The time step $\Delta t = 10^{-5}$ and the domain $-0.5 \leq r \leq 0.5$ is evenly divided into 320 cells having lengths 1/320.

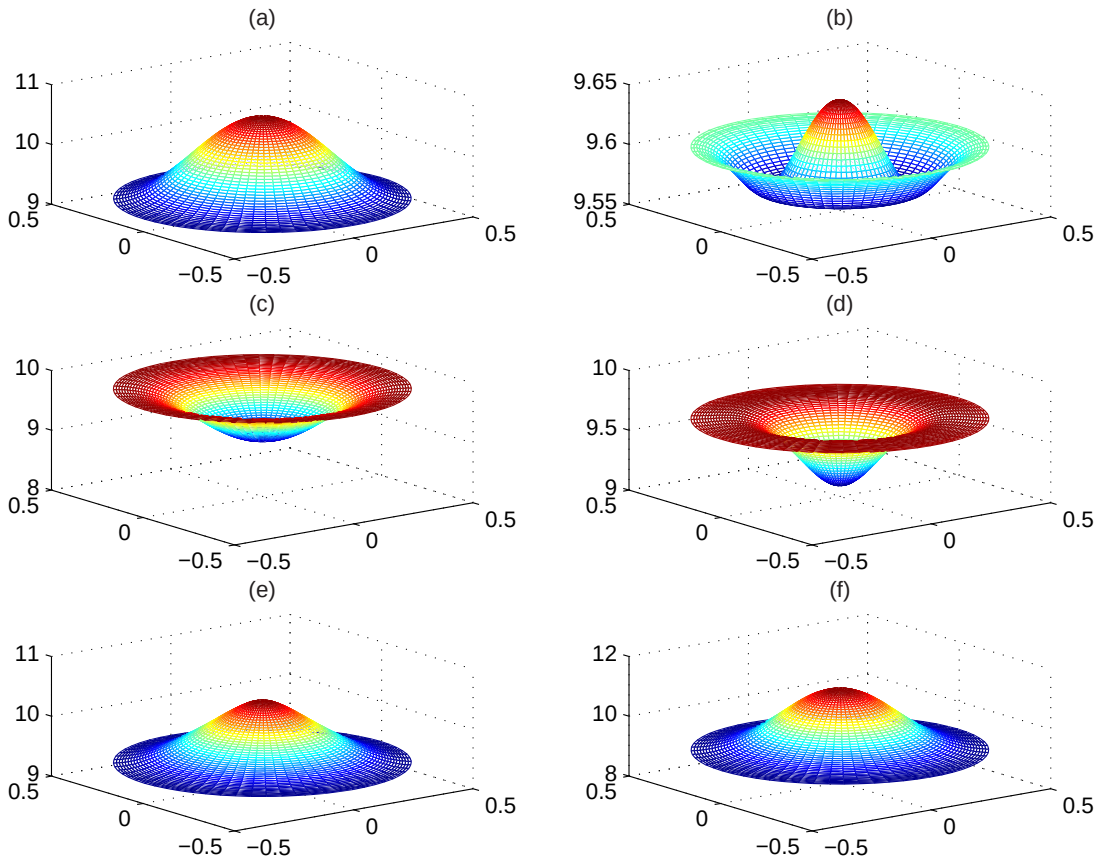



Figure 6.5: Density distributions in a circular tube from t = 0.012 to 0.072 with an increment of 0.012 from (a)-(f), computed by the dilatation element method.

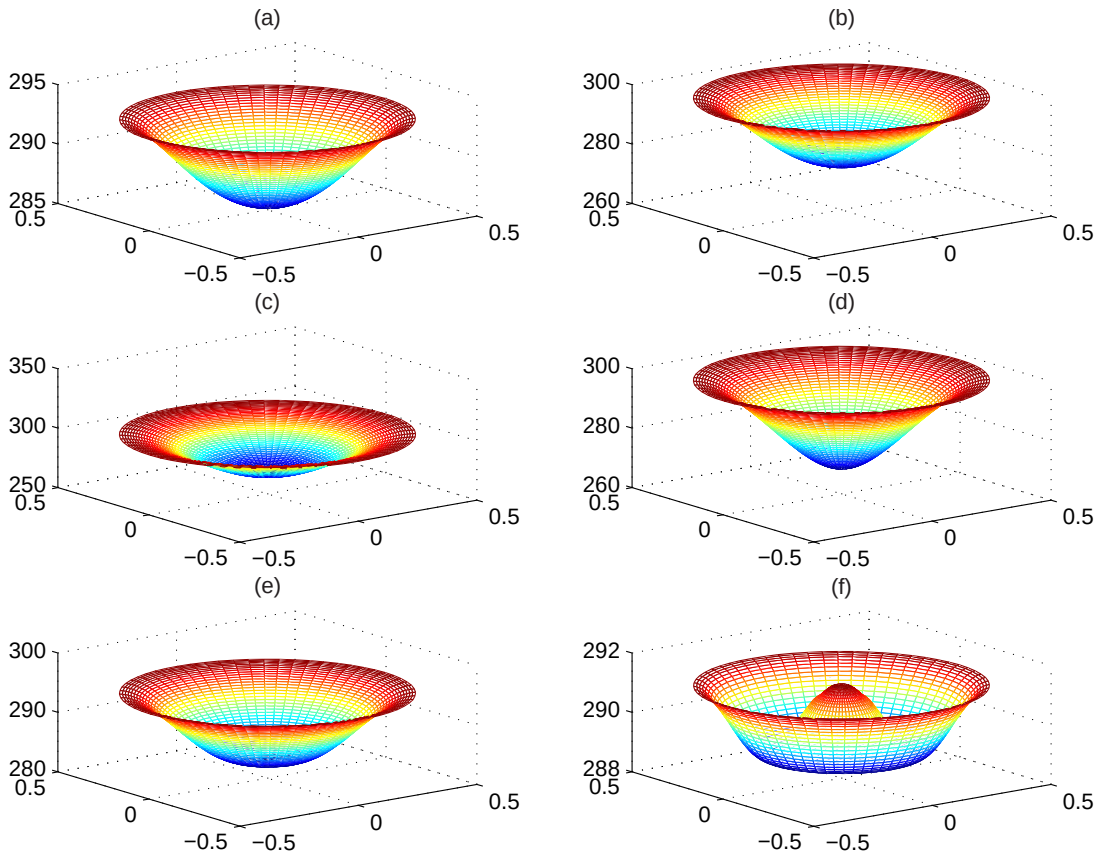



Figure 6.6: Temperature distributions in a circular tube from t = 0.012 to 0.072 with an increment of 0.012 from (a)-(f), computed by the dilatation element method.

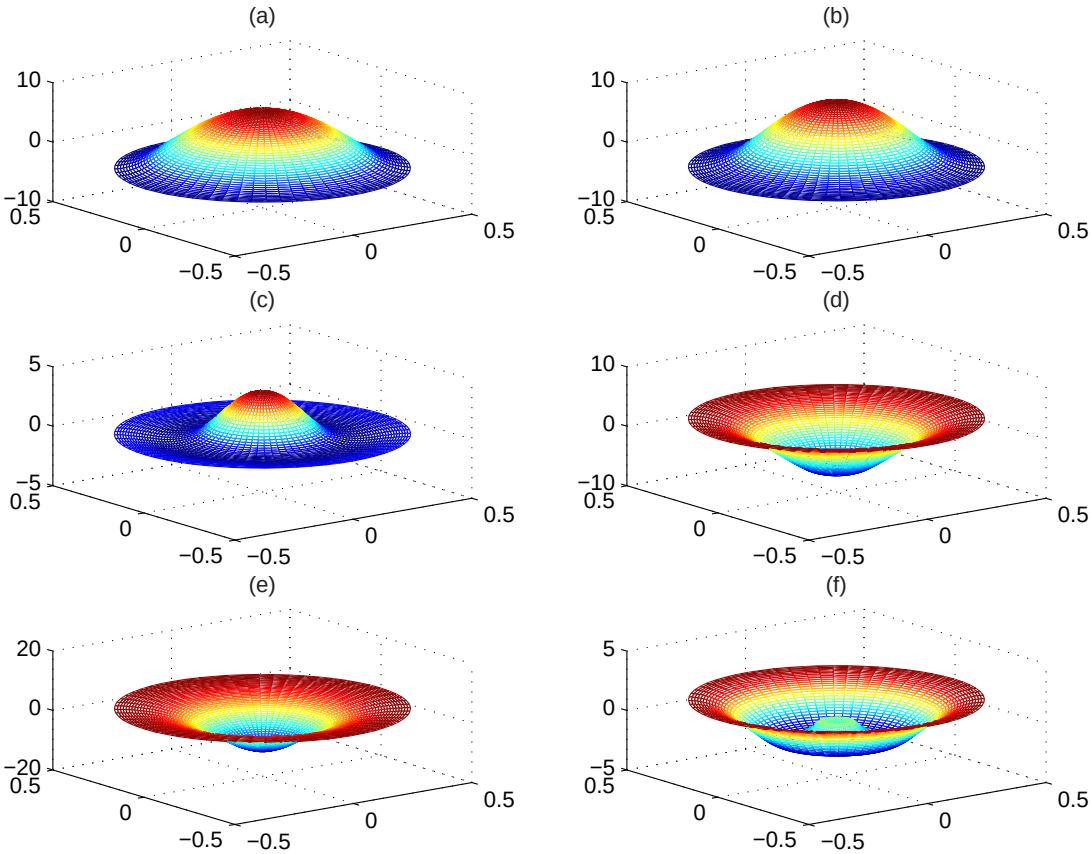



Figure 6.7: Dilatation distributions in a circular tube from t = 0.012 to 0.072 with an increment of 0.012 from (a)-(f), computed by the dilatation element method.

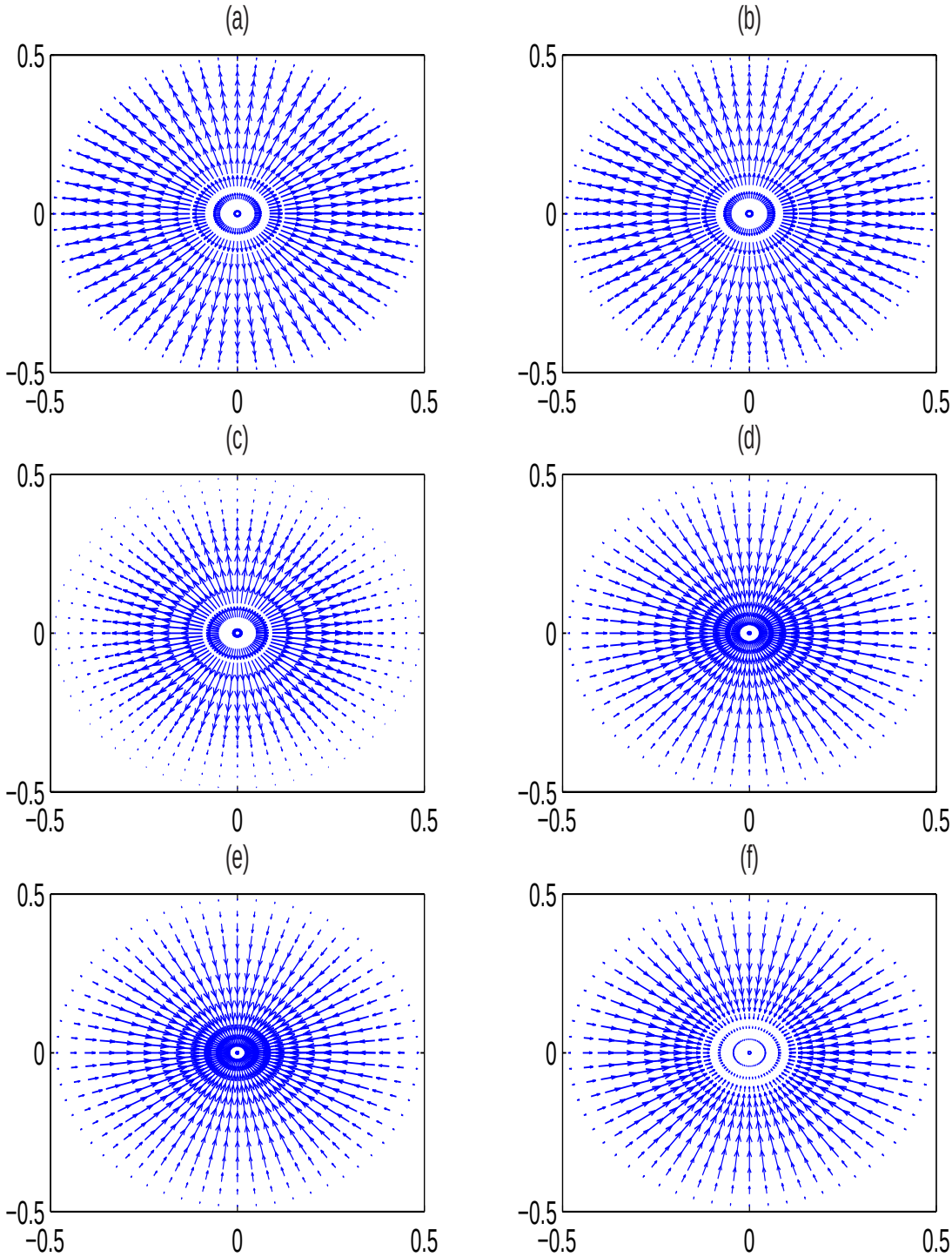



Figure 6.8: Velocity distribution of the flow field in a circular tube from t = 0.012 to 0.072 with an increment of 0.012 from (a)-(f), computed by the dilatation element method.

h

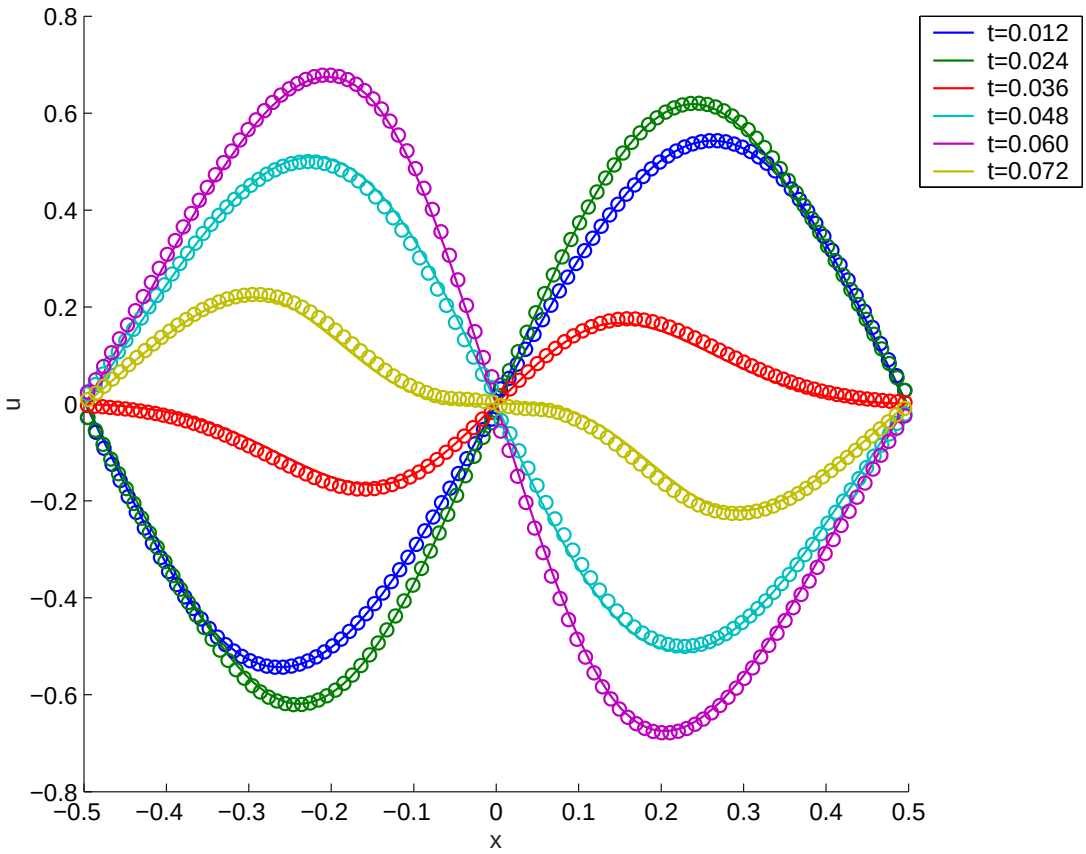



Figure 6.9: Comparison of the velocity $u$ in a 2D tube; Godunov scheme(—), particle method (o).

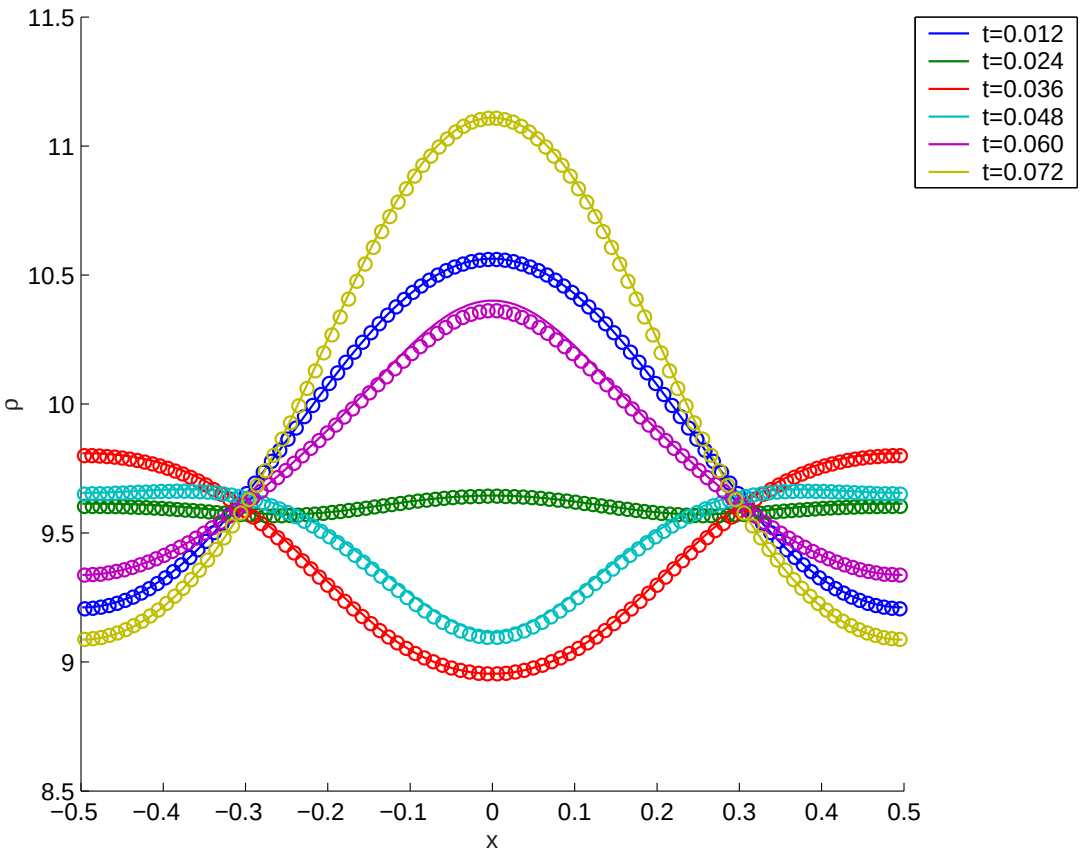



Figure 6.10: Comparison of the density $\rho$ in a 2D tube; Godunov scheme(—), particle method (o).

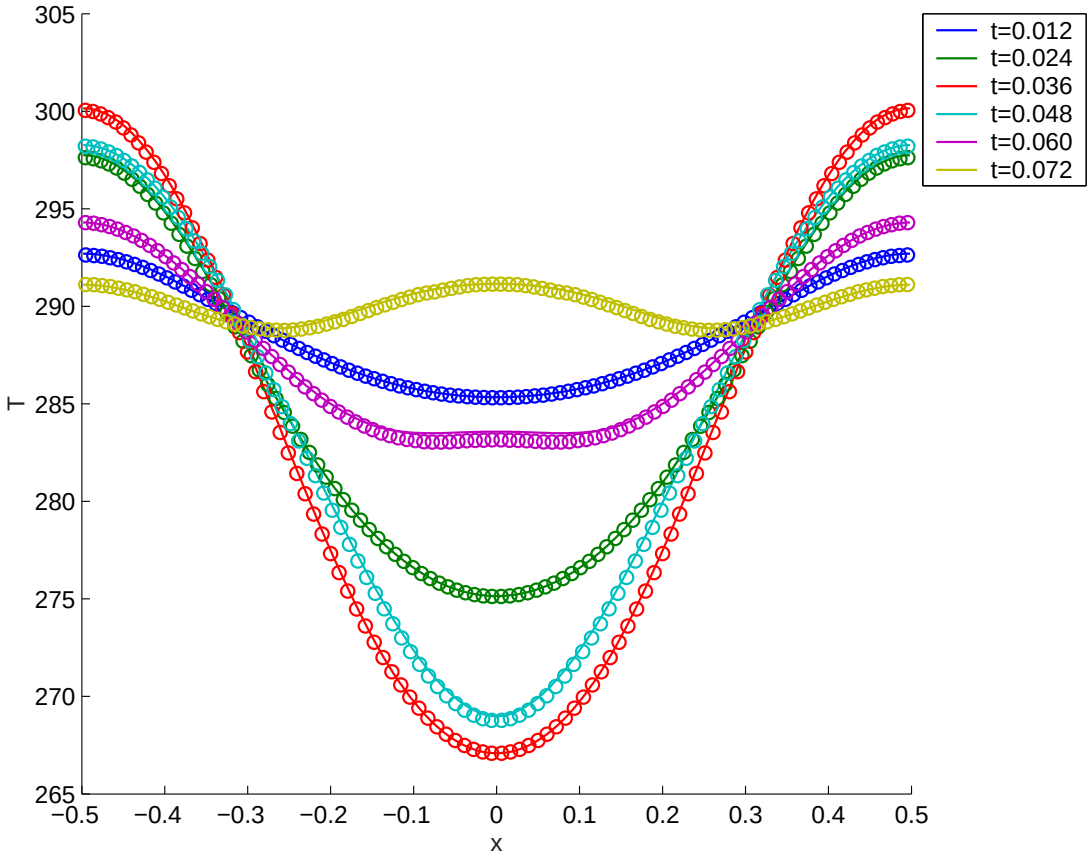



Figure 6.11: Comparison of the temperature $T$ in a 2D tube; Godunov scheme(—), particle method (o).

Figures 6.9 - 6.11 show the comparisons of the results for velocity, density, and temperature computed by the dilatation element method and the Godunov scheme. The results of the dilatation element method are taken along the radial direction as the $x$ axis with $y = 0$ because of the cylindrical symmetry. The predications of $u, \rho, T$ by the particle method agree well with the Godunov scheme.

## 6.2 Quasi-1D nozzle flow

For the quasi-1D nozzle flow, the governing equations (2.39) and (2.40) developed in Chapter 2 plus the equations for element positions and volumes are solved by the Lagrangian dilatation element method following the discussion of Chapter 5. A total of 200 particles are initially used to subdivide the domain, and time integration is done by the second order Runge-Kutta scheme with time step $\Delta t = 5.0 \times 10^{-04}$.

The duct area $A$ in this problem is the function of $x$

$$A(x) = \begin{cases} (1 - \frac{\beta}{4}(\cos(\pi(1+2x)) - 1)^2)^2, & |x| \leq 0.5 \\ 1, & |x| > 0.5 \end{cases} \tag{6.18}$$

where the parameter $\beta = 0.05$ for the results presented here. The maximum constriction occurs at $x = 0$ where the minimum of $A(x)$ is $(1-\beta)^2 = 0.9025$. This is a relatively small constriction. The computational domain is set between $x_l = -1$ and $x_r = 1$.

It is a simple matter to find the quasi-1D steady state solution of the nozzle flow. In fact, for an Eulerian frame of reference in steady flow, the corresponding

governing equations are

$$\frac{du}{dx} = -\alpha \frac{Tu}{T-u^2} \tag{6.19}$$

$$\frac{dT}{dx} = (\gamma-1)\alpha \frac{Tu^2}{T-u^2}, \tag{6.20}$$

where $\alpha \equiv \frac{dA(x)/dx}{A(x)}$ is only non-zero in regions where the duct changes area.

Equations (6.19) and (6.20) may be solved for $u$ and $T$ by Matlab (e.g., using the solver ODE45). For any given upstream condition the local Mach number $M = u/\sqrt{T}$ peaks at the throat except when the downstream flow is supersonic. For the nozzle geometry given by $\beta = 0.05$, a shock wave ($M = 1$) first appears at the throat when the far upstream velocity $u_0 \approx 0.681$. This then is an upper limit to what $u_0$ should be since our main interest is confined to the subsonic regime. Additional work is needed to develop this scheme into handling shock waves. The upstream flow velocity for the results shown here is $u_0 = 0.4$. The quasi-1D steady state solutions computed from (6.19) and (6.20) are used to test the accuracy of the quasi-1D and 2D solutions of the dilatation element method after they reach an equilibrium state.

Figures 6.12 - 6.16 show the transient change of the temperature distribution in the nozzle with time from $t = 0.1$ to 3.0 with a 0.1 increment. Similarly, figures 6.17 - 6.21 present the transient change of the velocity distribution in the nozzle and figures 6.22 - 6.26 show the transient change of the dilatation distribution in the nozzle. The computational elements in the Lagrangian dilatation element method go through a transient from an initial state to an equilibrium flow condition. Equilibrium is achieved when each individual element traveling through the computational domain undergoes a similar history. It is seen in the figures 6.12(a), 6.17(a) and 6.22(a), that the downstream and upstream traveling waves originate in the nozzle region. The following figures (b) to (f) show the

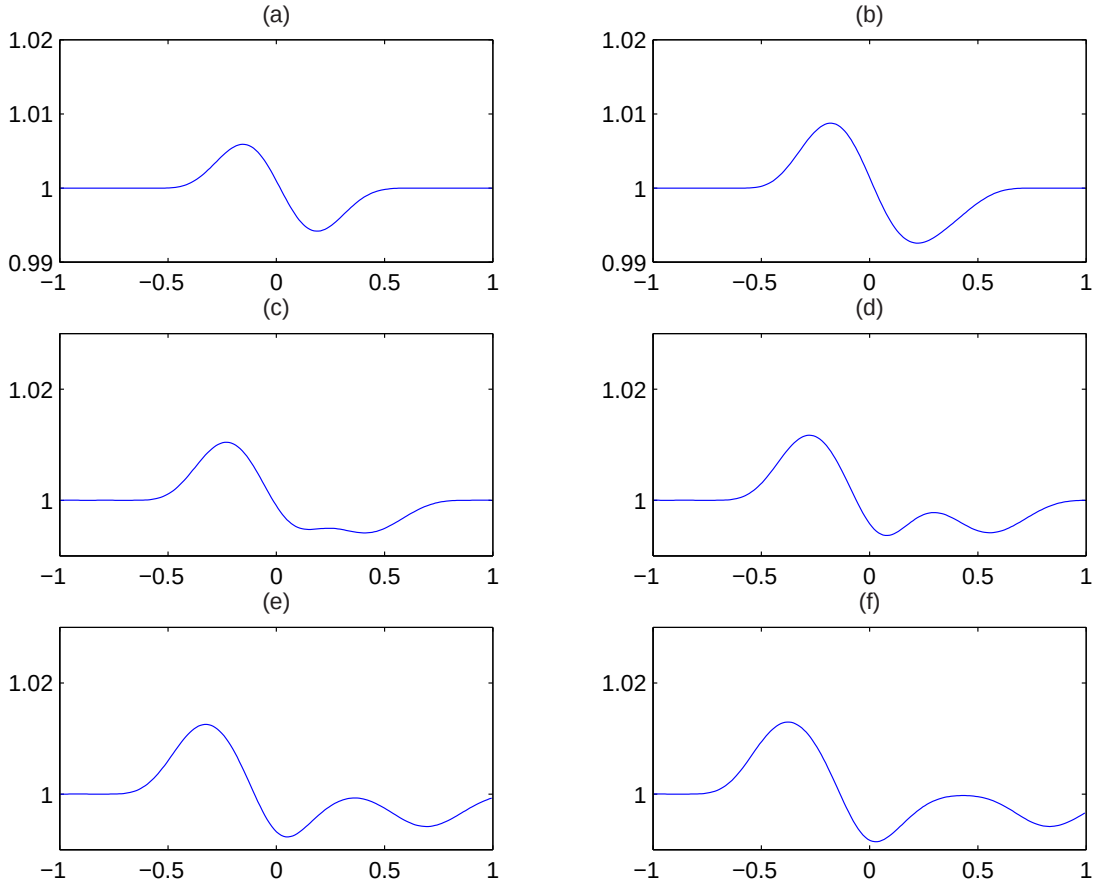



Figure 6.12: Temperature $T$ distribution, t = 0.1 to 0.6 with an increment of 0.1 from (a) to (f).

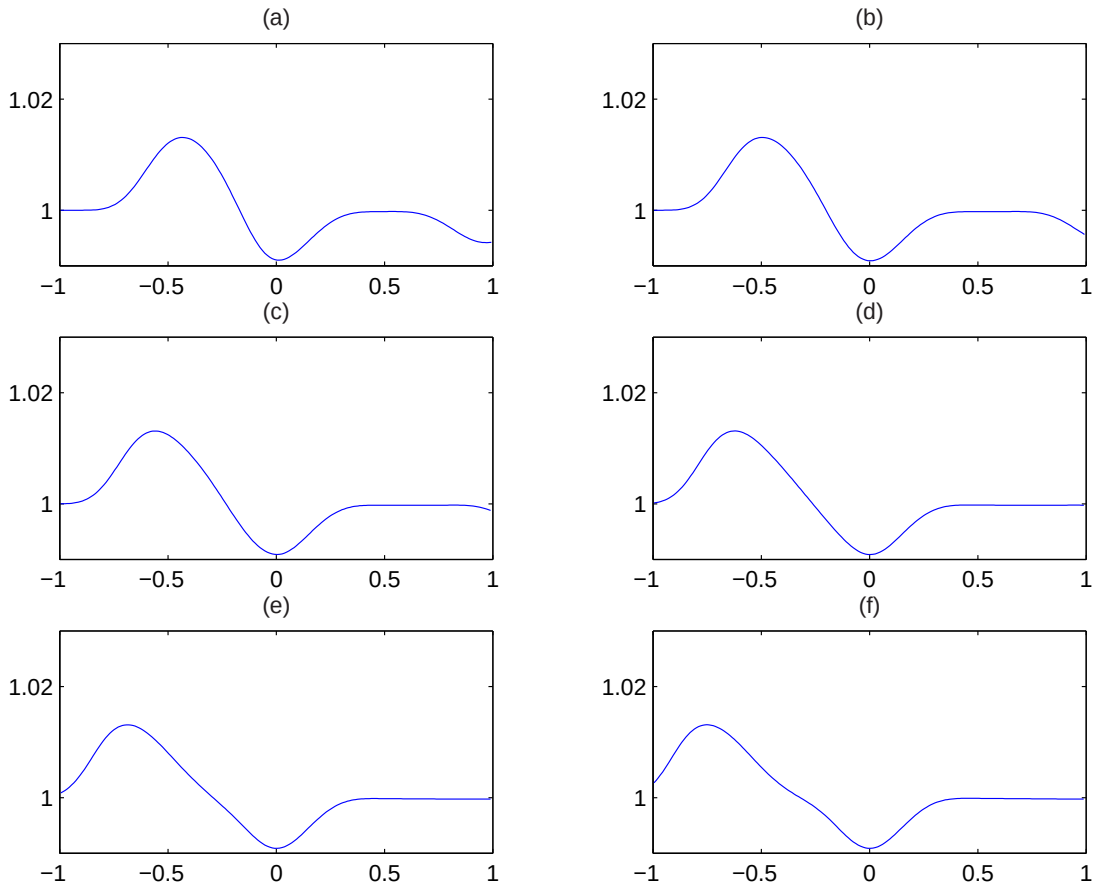



Figure 6.13: Temperature $T$ distribution, t = 0.7 to 1.2 with an increment of 0.1 from (a) to (f).

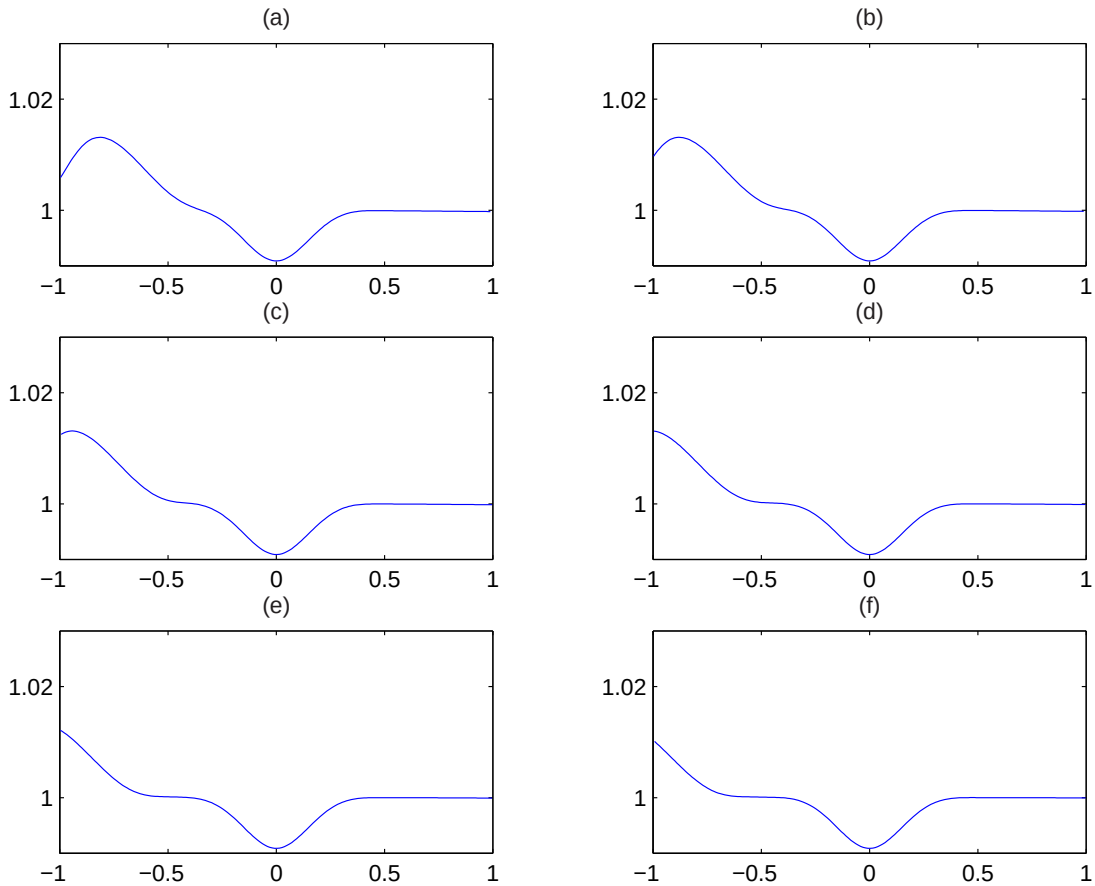



Figure 6.14: Temperature $T$ distribution, t = 1.3 to 1.8 with an increment of 0.1 from (a) to (f).

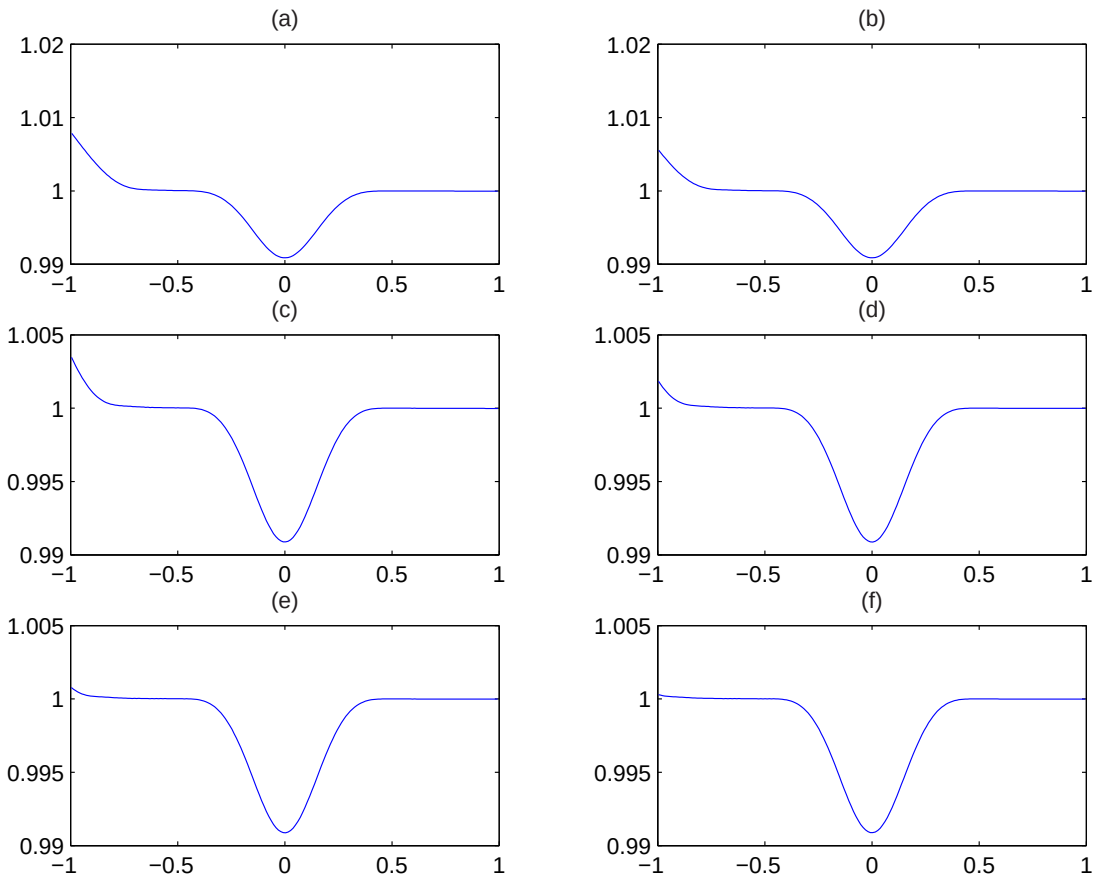



Figure 6.15: Temperature $T$ distribution, t = 1.9 to 2.4 with an increment of 0.1 from (a) to (f).

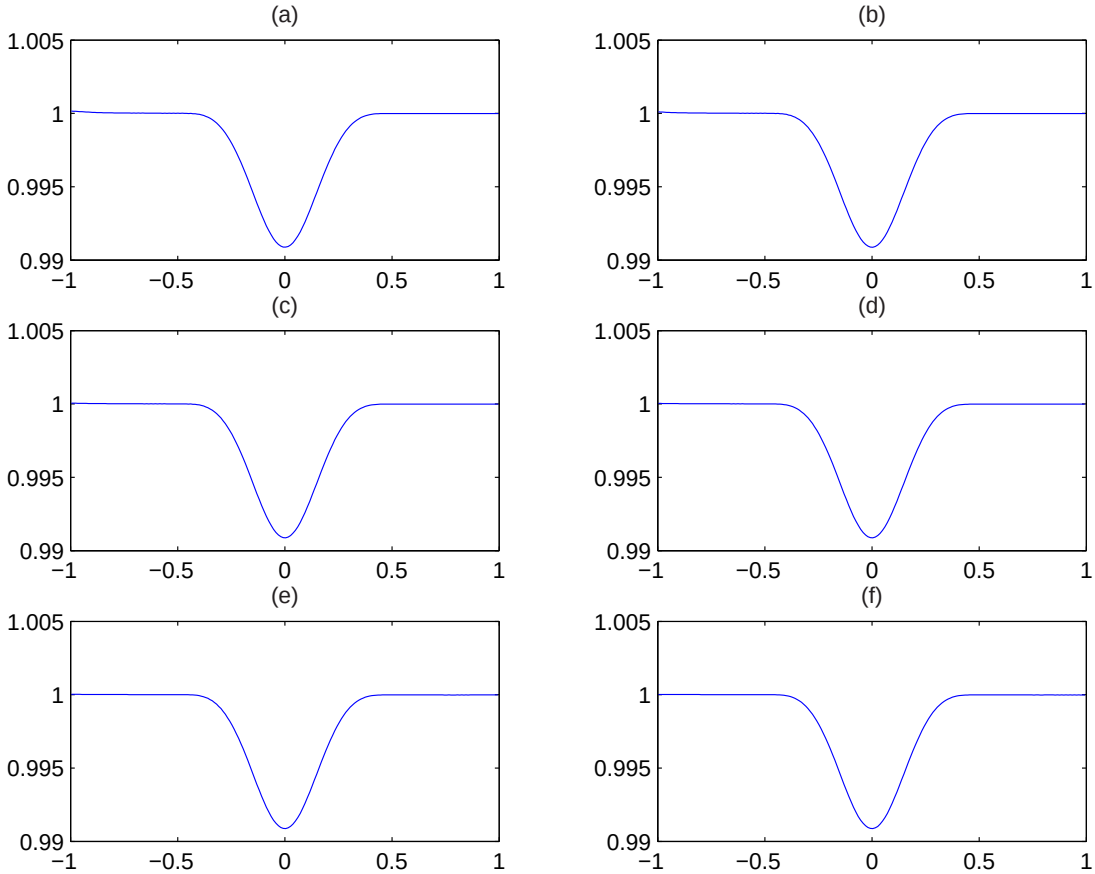



Figure 6.16: Temperature $T$ distribution, t = 2.5 to 3.0 with an increment of 0.1 from (a) to (f).

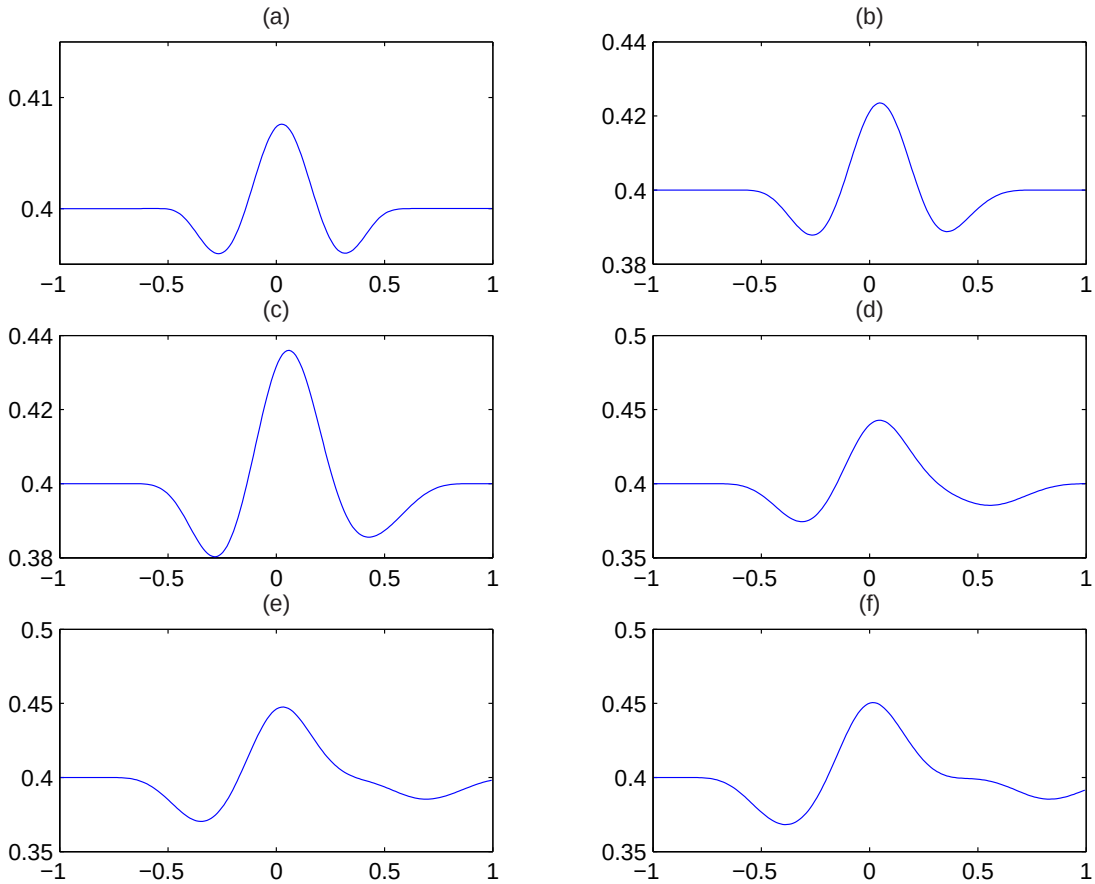



Figure 6.17: Velocity $u$ distribution, t = 0.1 to 0.6 with an increment of 0.1 from (a) to (f).

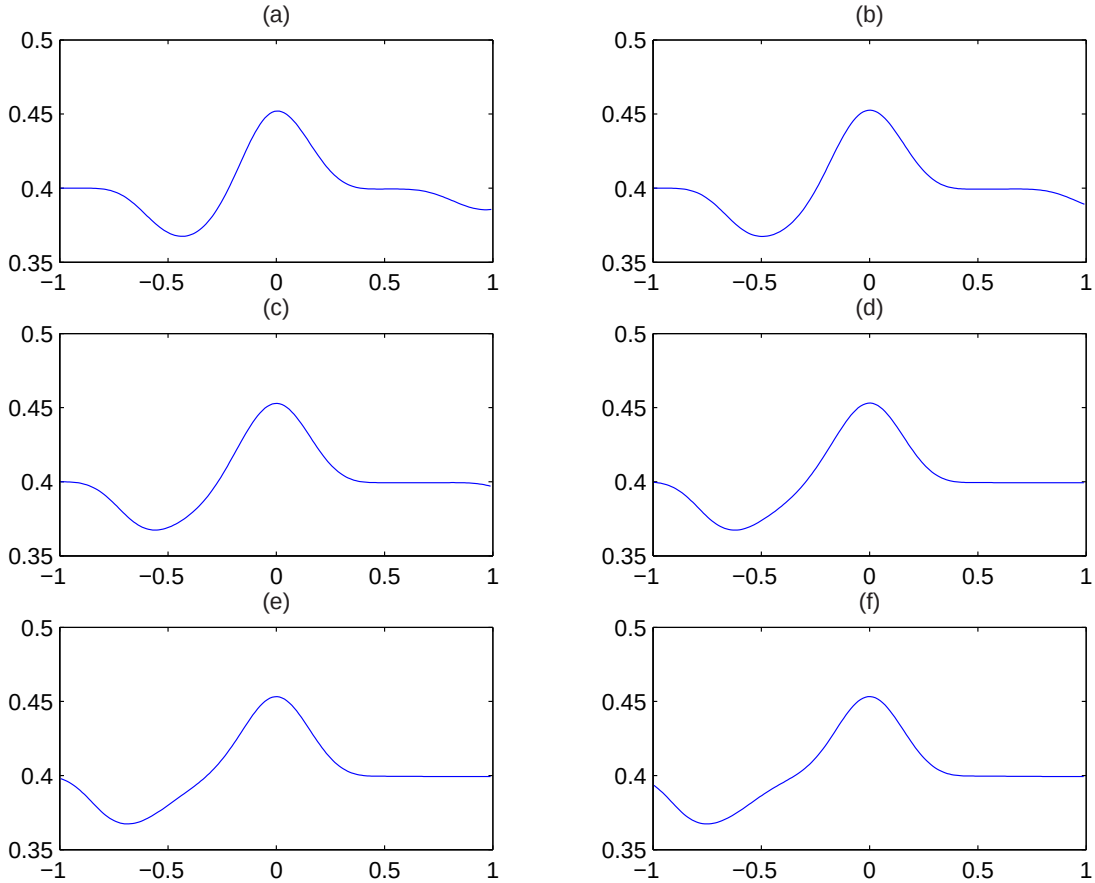



Figure 6.18: Velocity $u$ distribution, t = 0.7 to 1.2 with an increment of 0.1 from (a) to (f).

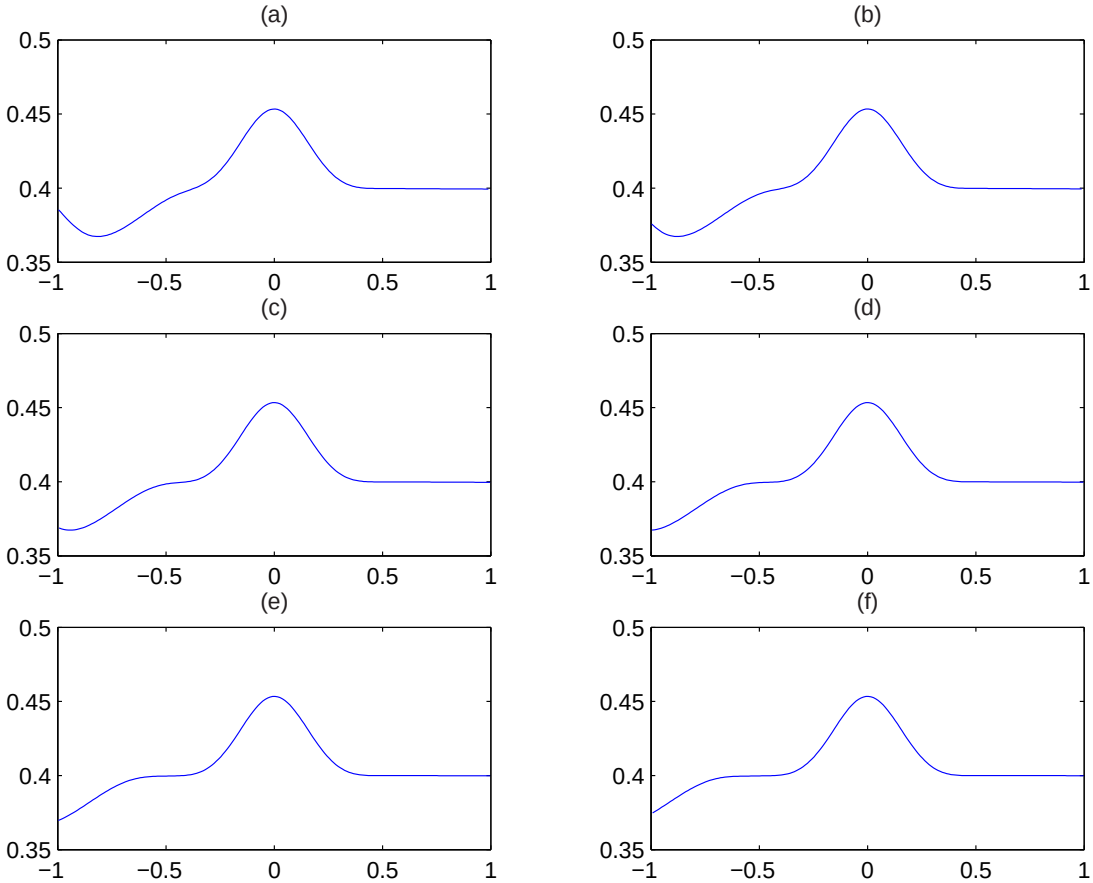



Figure 6.19: Velocity $u$ distribution, t = 1.3 to 1.8 with an increment of 0.1 from (a) to (f).

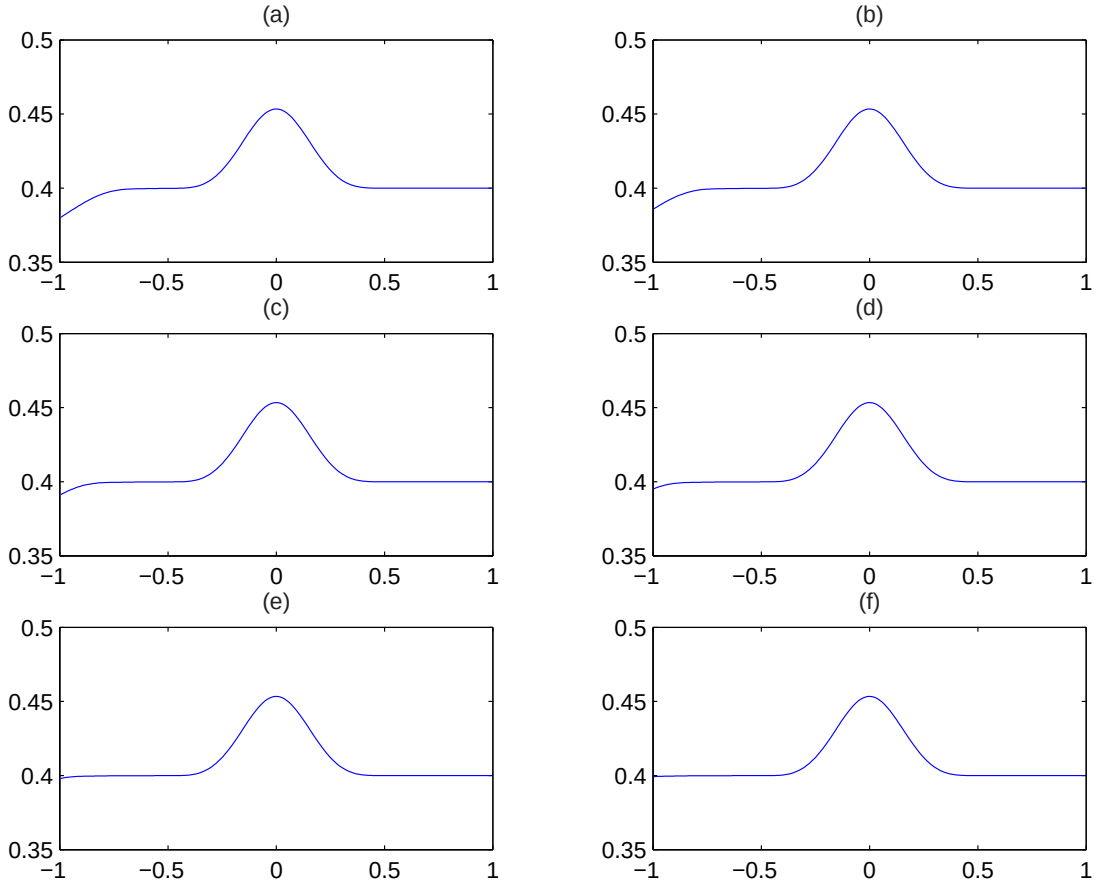



Figure 6.20: Velocity $u$ distribution, t = 1.9 to 2.4 with an increment of 0.1 from (a) to (f).

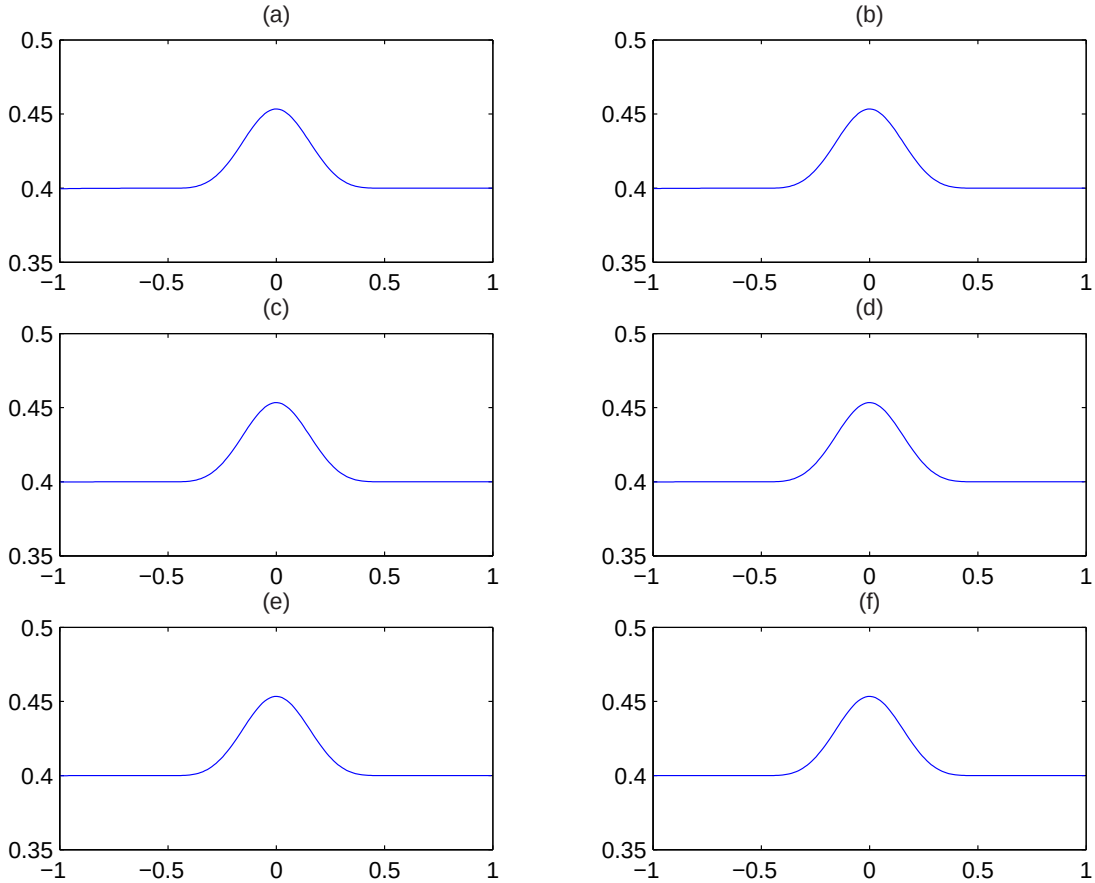



Figure 6.21: Velocity $u$ distribution, t = 2.5 to 3.0 with an increment of 0.1 from (a) to (f).

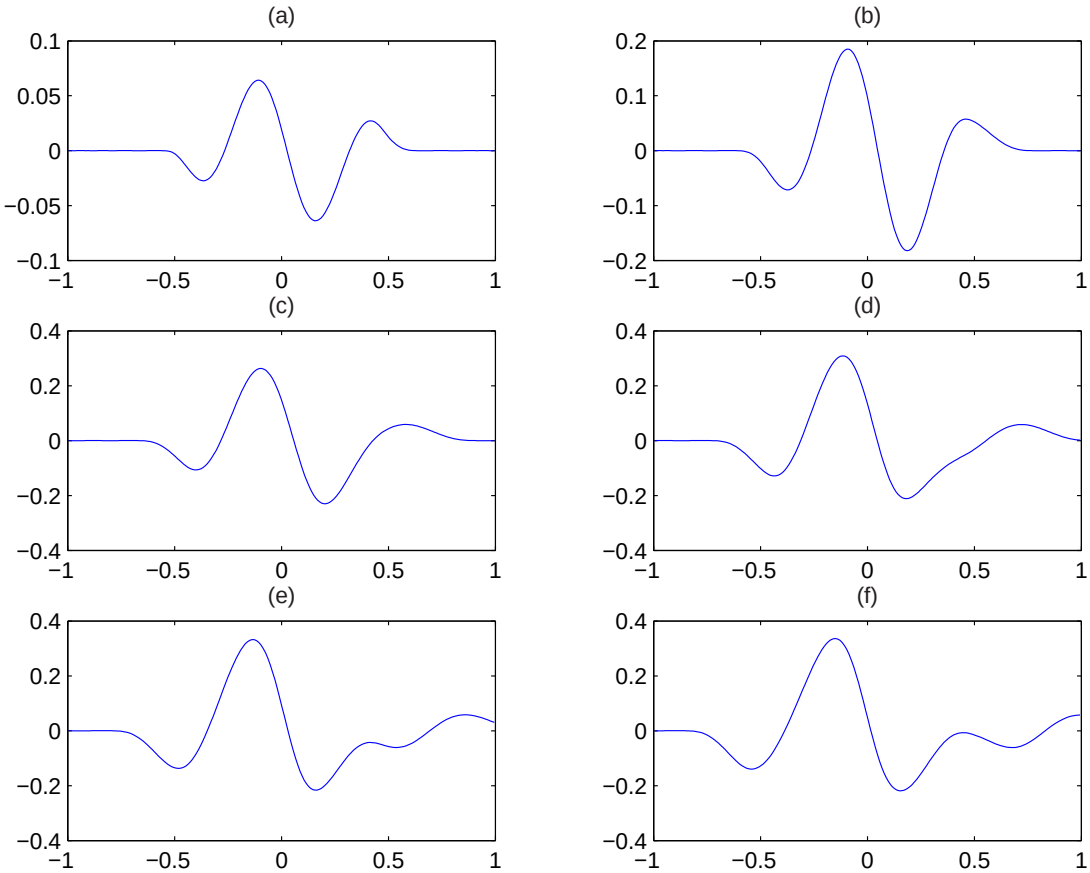



Figure 6.22: Dilatation $\theta$ distribution, t = 0.1 to 0.6 with an increment of 0.1 from (a) to (f).

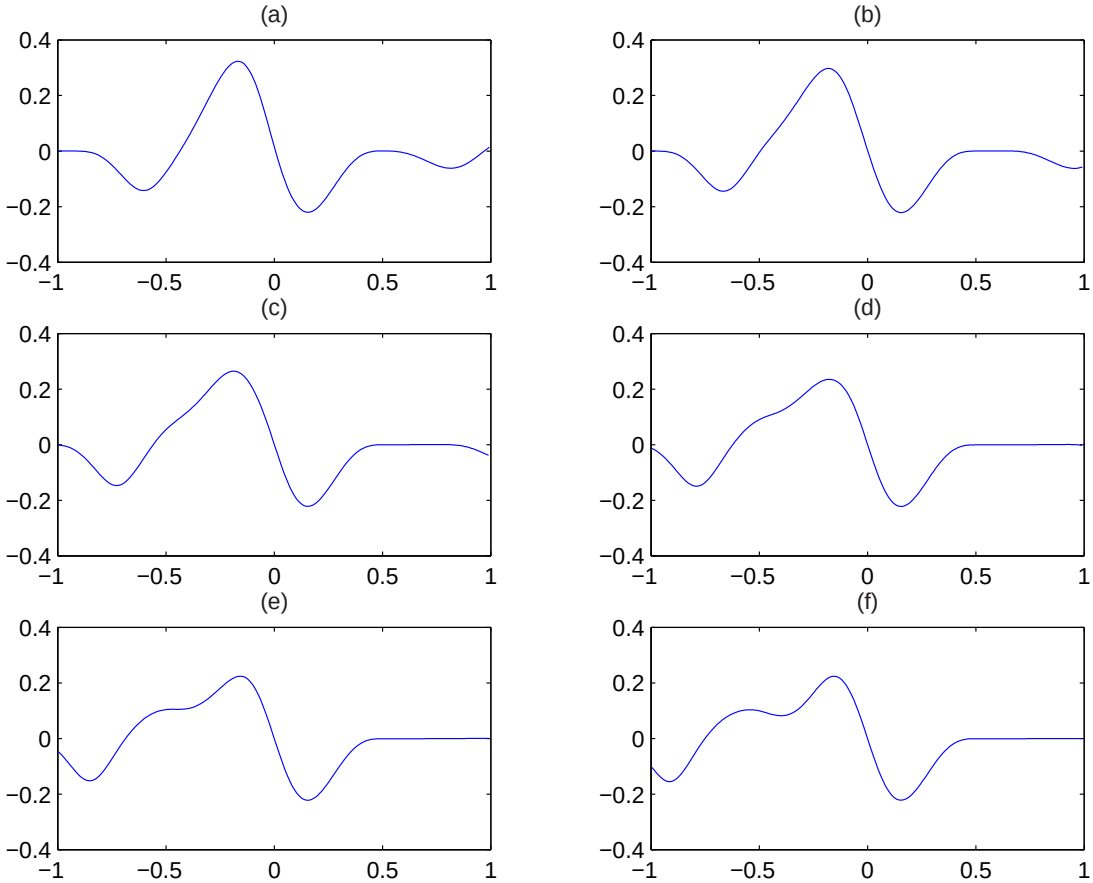



Figure 6.23: Dilatation $\theta$ distribution, t = 0.7 to 1.2 with an increment of 0.1 from (a) to (f).

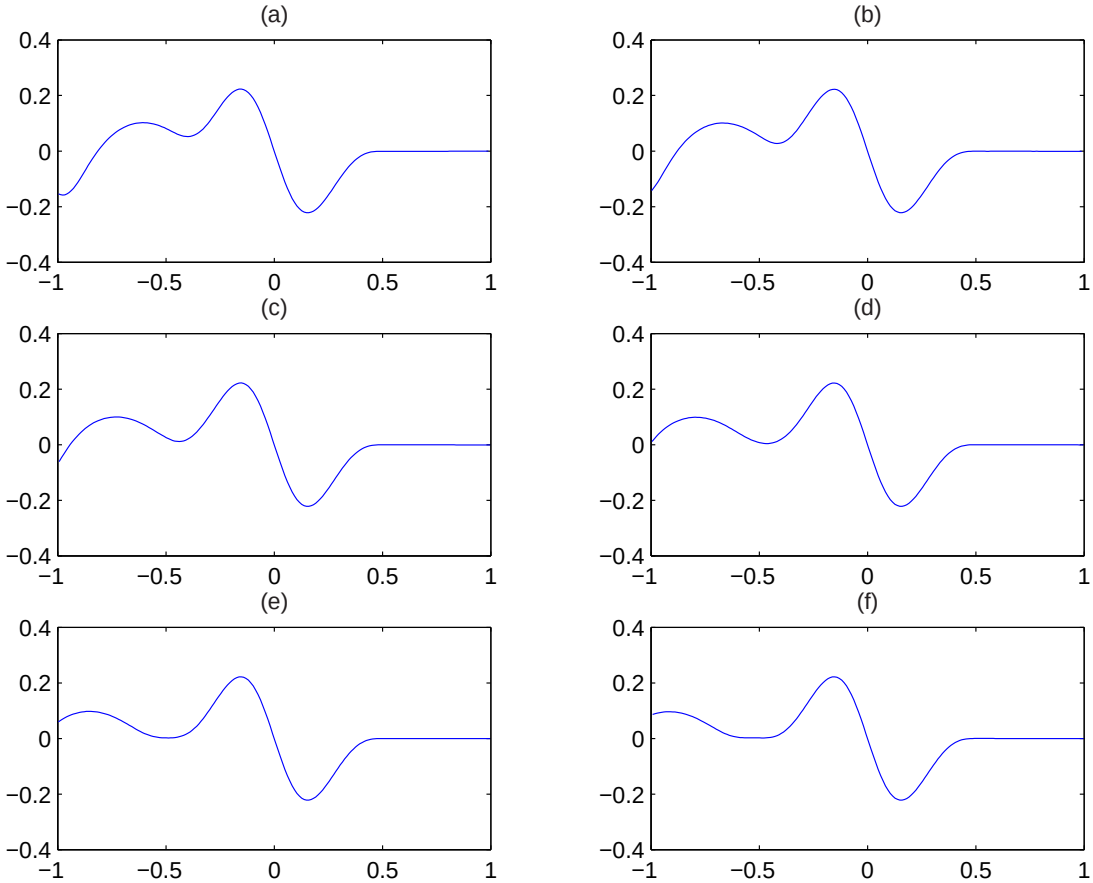



Figure 6.24: Dilatation $\theta$ distribution, t = 1.3 to 1.8 with an increment of 0.1 from (a) to (f).

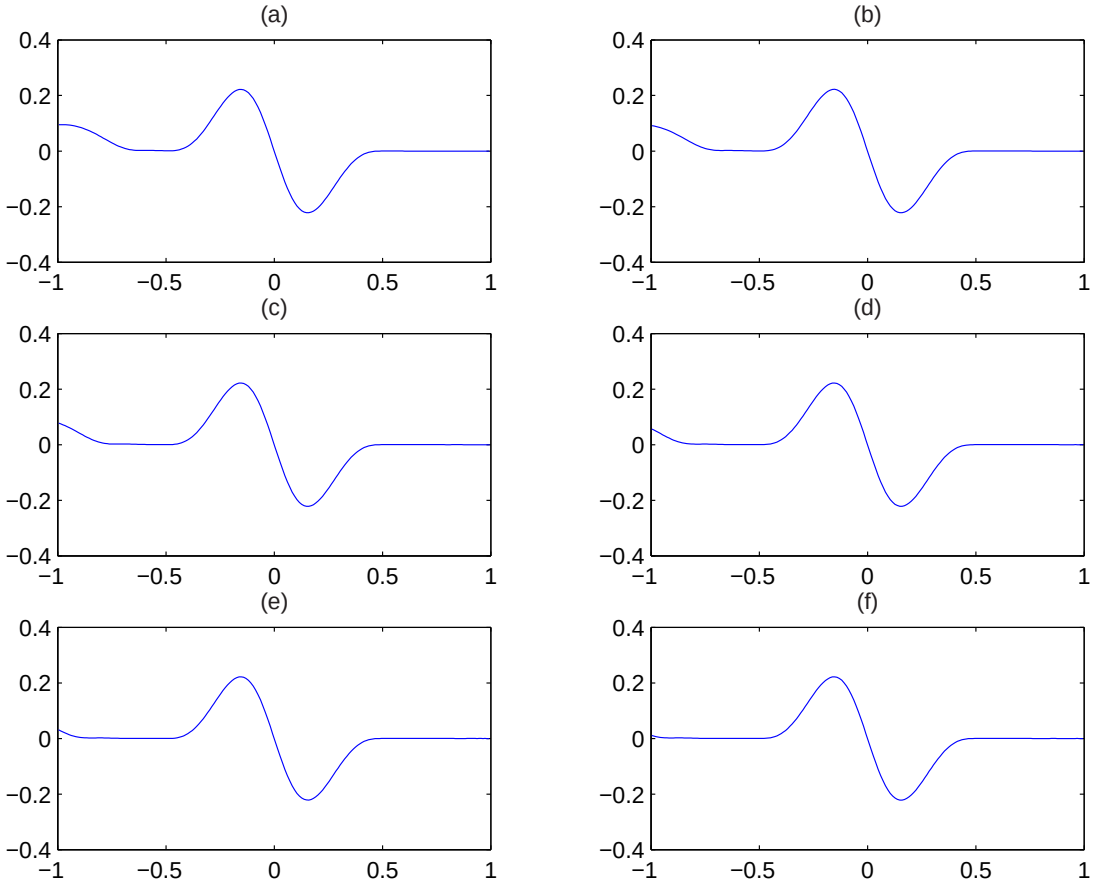



Figure 6.25: Dilatation $\theta$ distribution, t = 1.9 to 2.4 with an increment of 0.1 from (a) to (f).

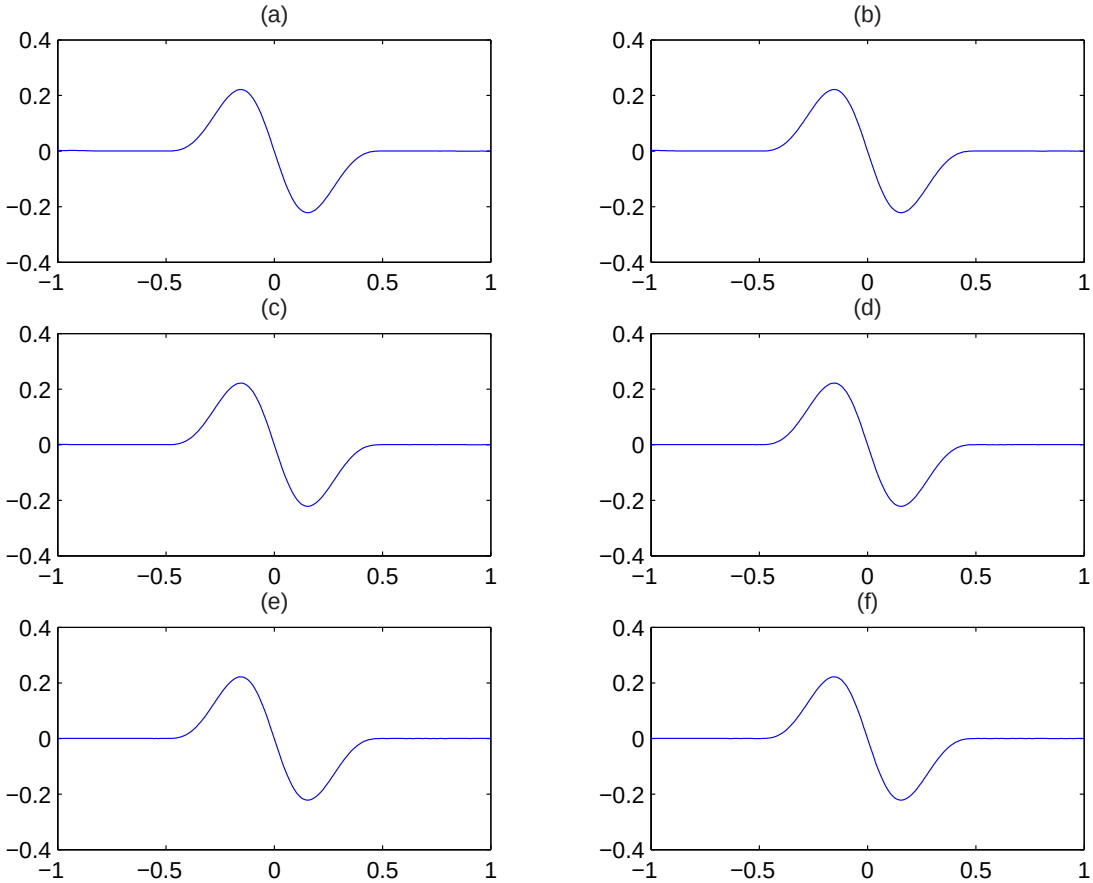



Figure 6.26: Dilatation $\theta$ distribution, t = 2.5 to 3.0 with an increment of 0.1 from (a) to (f).

development of these characteristic waves. The downstream wave moves with a speed consistent with the right moving characteristics given by (5.29) while the upstream wave is consistent with (5.30).

As evident in these figures, the downstream wave reaches the right boundary $x_r$ at $t \approx 1.0$ while the upstream wave reaches the left boundary $x_l$ at $t \approx 2.4$. Both of the waves march through the boundary without noticeable distortion or reflection. In the downstream wave, $T$ and $u$ are below the ambient while $T > 1$ and $u < u_0$ for the upstream wave. After both waves leave the domain all that remains is the equilibrium solution. It suffices to observe that equilibrium is reached for all practical purposes when the plots of temperature, velocity and dilatation do not visibly change in time. For example figures (6.16), (6.21) and (6.26) covering the time from $t = 2.5$ to 3.0 show that all of the elements are experiencing a similar history so equilibrium has been reached.

The results of the Lagrangian dilatation element method at $t = 3.0$ are compared with the exact steady state solution computed from (6.19) and (6.20). Figures 6.27 and 6.28 are the comparisons of computed temperature $T$ and velocity $u$ by the dilatation element method with exact solutions. The correctness of the computation is clearly shown.

For the equilibrium flow in a duct, mass conservation clearly implies that

$$\rho u A = Q, \tag{6.21}$$

where $Q$ is the constant mass flux down the duct. For incompressible flow, the density is a constant. Given the upstream velocity $u_0$, the velocity $u$ can be easily computed from equation (6.21). The dilatation based on (6.21) gives

$$\theta = -u \frac{dA(x)/dx}{A(x)}, \tag{6.22}$$

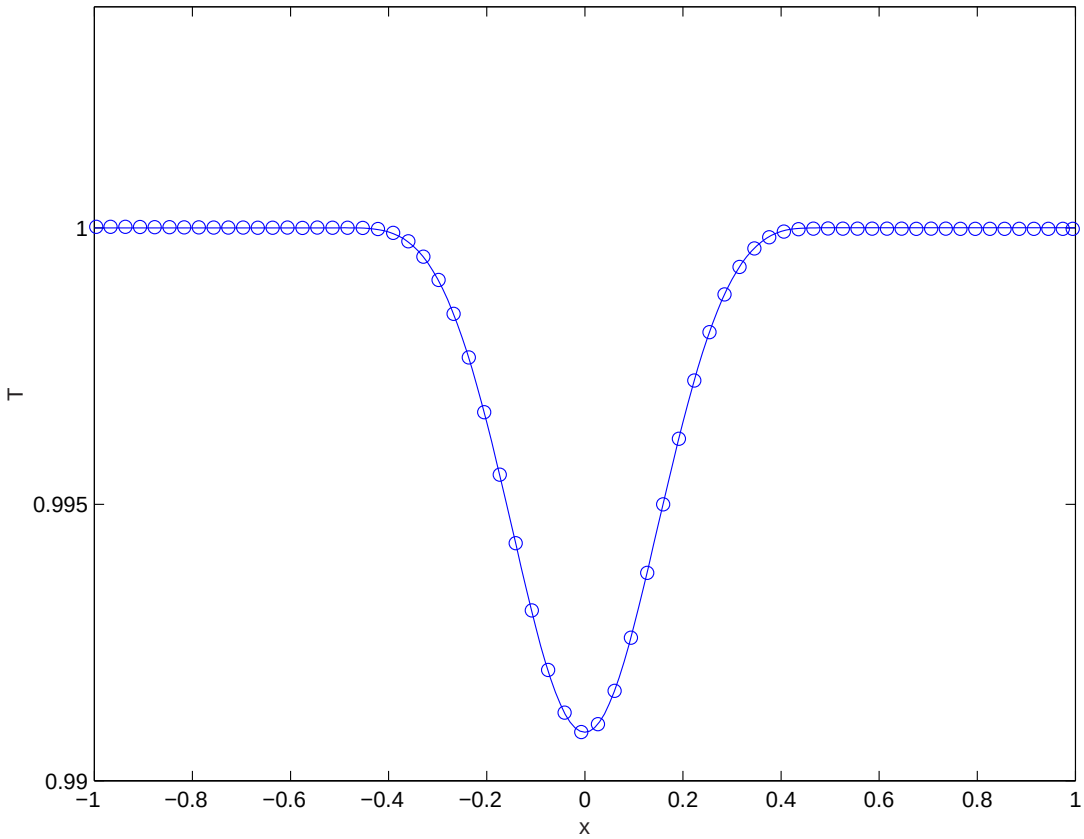



Figure 6.27: Comparison of the computed temperature $T$, (o) with exact solution (—).

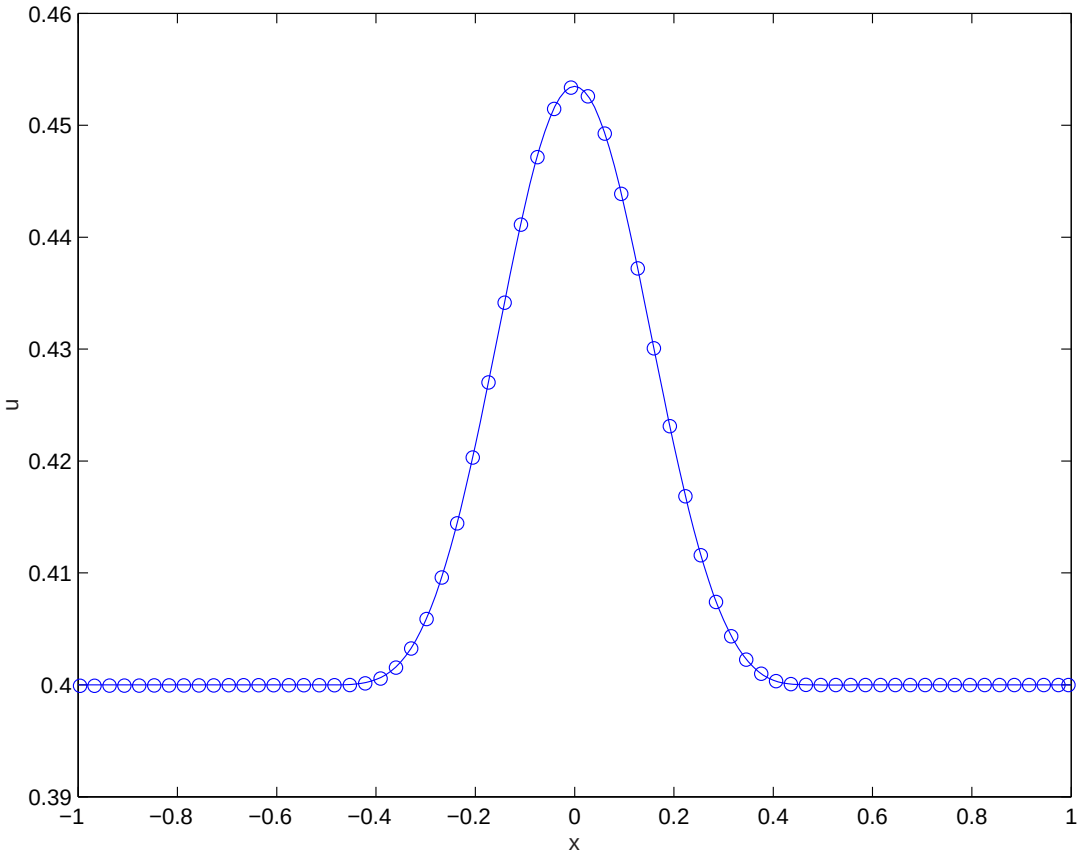



Figure 6.28: Comparison of computed velocity $u$, (o) with exact solution (—).

which means that despite incompressibility, the assumption of one-dimensionality implies that the dilatation is non-zero in regions where the duct changes area.

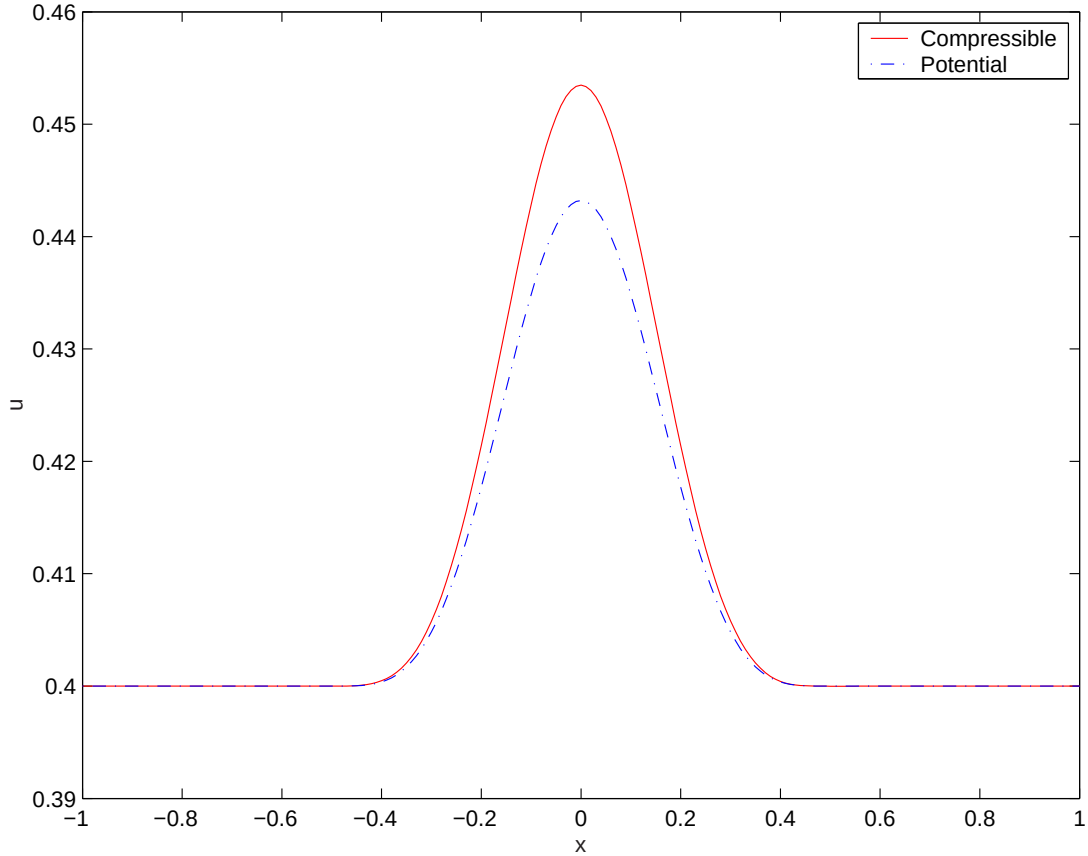



Figure 6.29: The equilibrium velocity distribution compared with the incompressible potential flow.

The equilibrium velocity and dilatation fields of the compressible quasi-1D nozzle are compared to the incompressible potential flow values in figures 6.29 and 6.30. There is clearly a significant additional compression and expansion associated purely with compressibility.

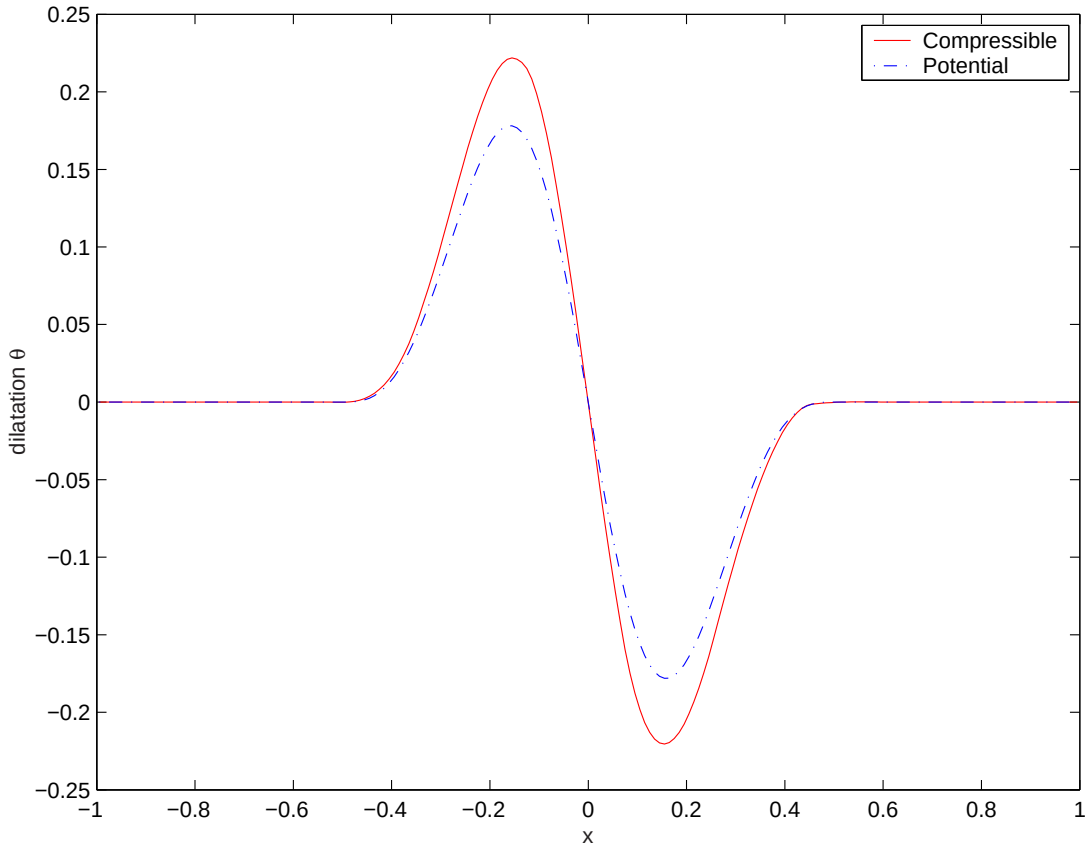



Figure 6.30: The equilibrium dilatation distribution compared with the incompressible potential flow.

## 6.3 2D subsonic nozzle flow with inlet and outlet

2D nozzle flow requires use of the previously discussed algorithm for treating inlet and outlet boundary conditions (see 5.3). Moreover, it is necessary to use the coordinate transformation approach given in section 5.2 in order to accommodate the curved boundaries. The methodology developed in Chapter 5 is applied here in a 2D nozzle flow problem. The nozzle has the same cross-sectional area as the previous quasi-1D nozzle given by (6.18). The inlet and outlet boundaries are $x_l = -1$ and $x_r = 1$, respectively. The upstream velocity $u_0 = 0.4$. A total of $60 \times 30$ Lagrangian computational elements are initially used to simulate the compressible flow in the $x$ and $y$ directions, respectively. The second order Runge-Kutta scheme is used to integrate the equations (5.1) - (5.4) while the spatial derivatives are evaluated by moving least square fitting. The time step $\Delta t = 0.02$. The flow is integrated until $t = 5.0$.

In figures 6.31 - 6.36, the time histories of temperature, mach number and dilatation distribution of the 2D nozzle flow are presented. These results are independent of the number of particles as long as it is not so small as to prevent adequate resolution of the spatial variation in the computed fields. These figures clearly show the development of upstream and downstream traveling waves that originate in the nozzle throat region. Equilibrium is achieved at approximately $t \approx 4.8$, that is, after this time the temperature, mach number and dilatation distributions do not vary beyond what is shown in these plots. The equilibrium state of the Mach number appears to be the first reached while that of the dilatation is the last. This may be due to the fact that $\theta$ experiences larger diffusion

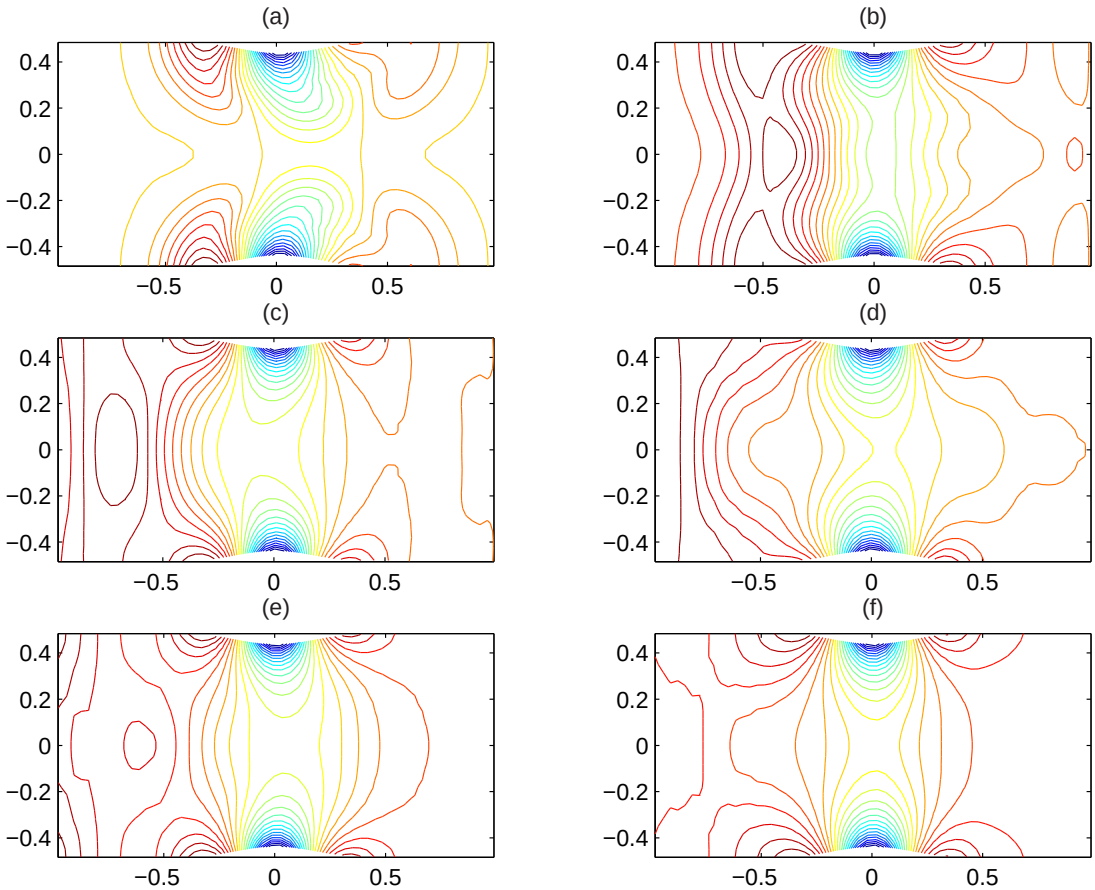



Figure 6.31: The temperature $T$ contour plot, t = 0.4 to 2.4 with increment 0.4 from (a) to (f).

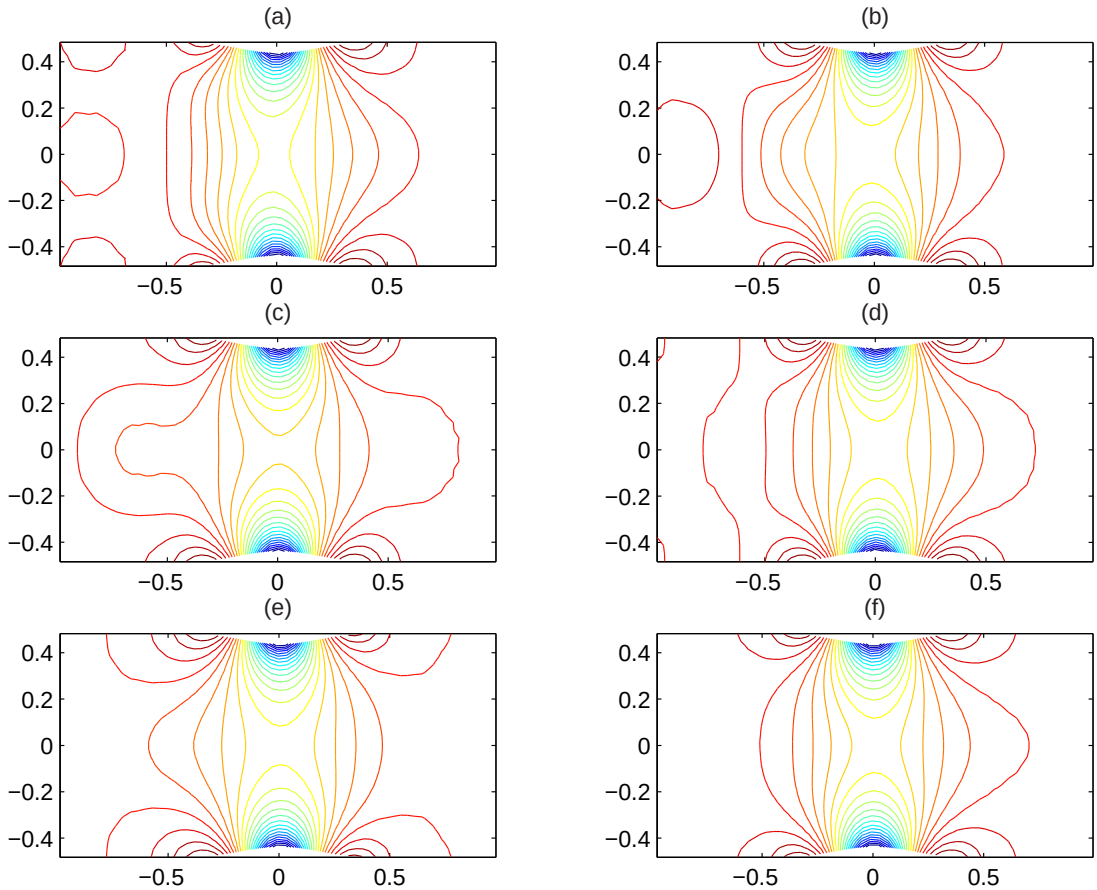



Figure 6.32: The temperature $T$ contour plot, t = 2.8 to 4.8 with increment 0.4 from (a) to (f).

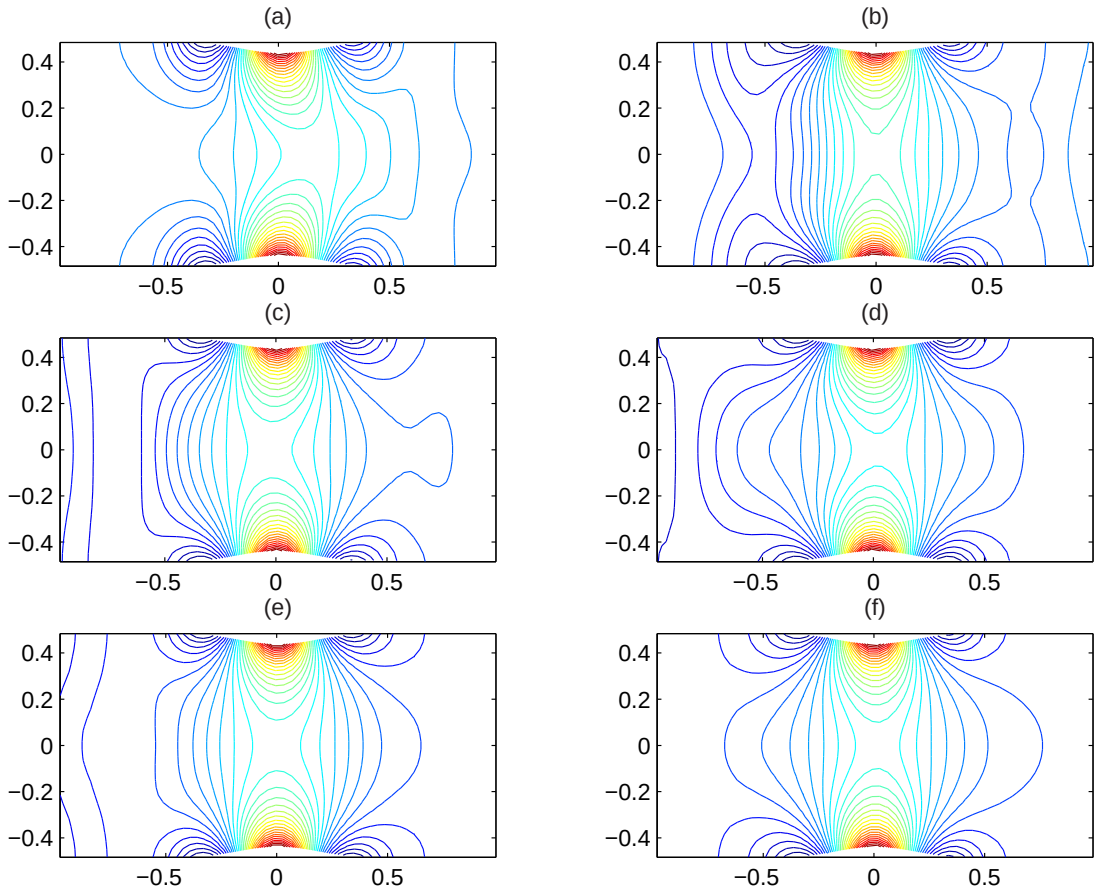



Figure 6.33: The Mach number contour plot, t = 0.4 to 2.4 with increment 0.4 from (a) to (f).

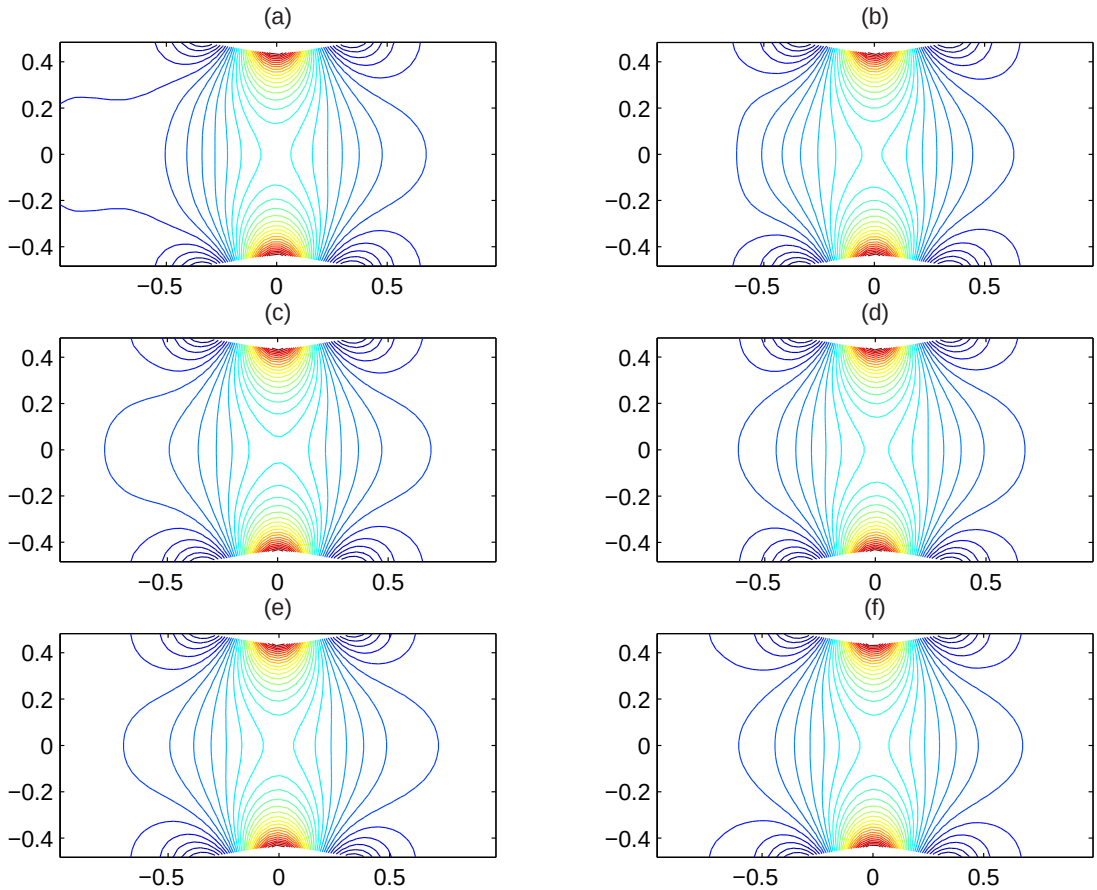



Figure 6.34: The Mach number contour plot, t = 2.8 to 4.8 with increment 0.4 from (a) to (f).

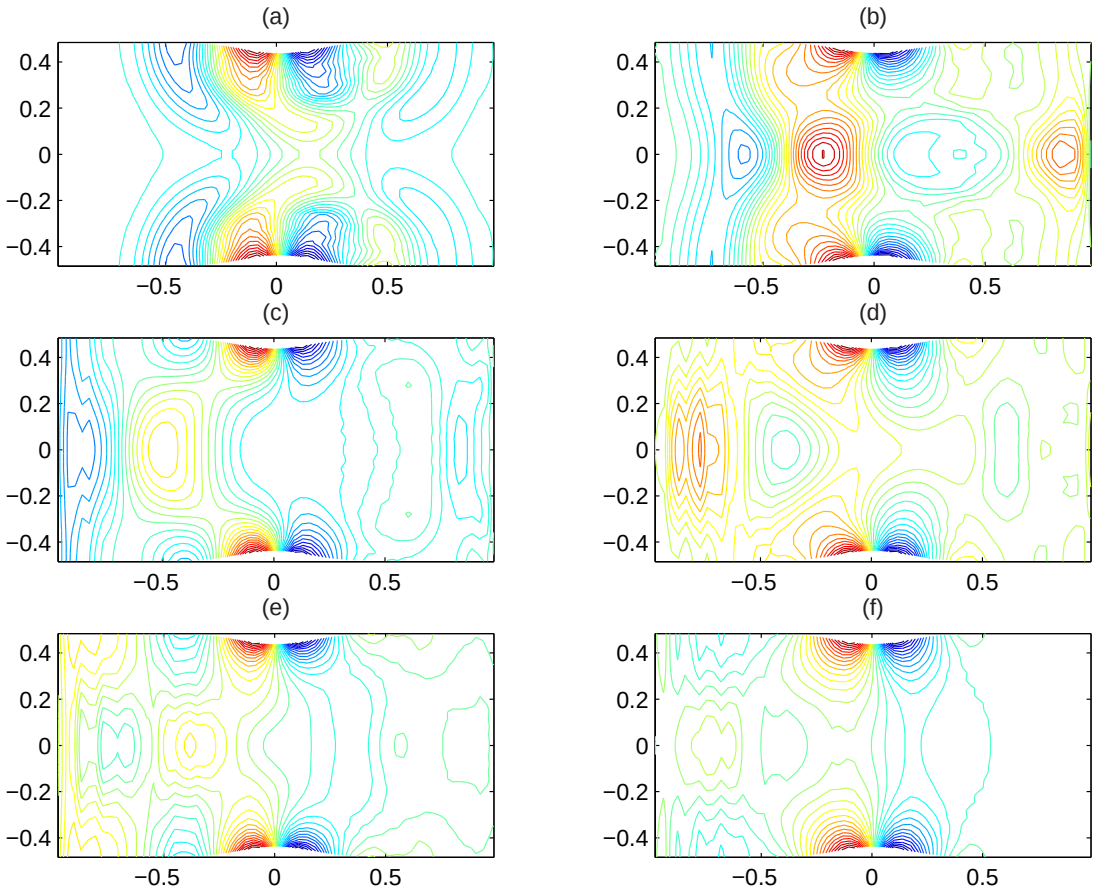



Figure 6.35: The dilatation $\theta$ contour plot, t = 0.4 to 2.4 with increment 0.4 from (a) to (f).

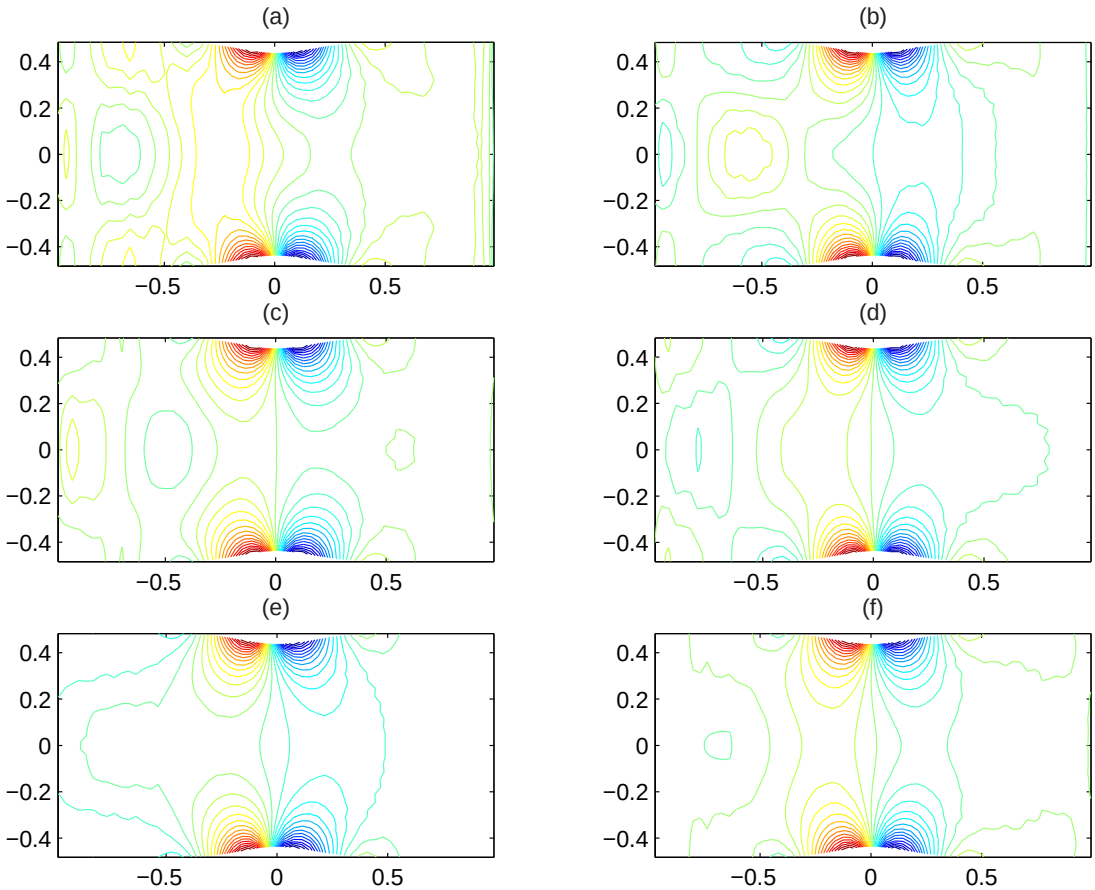



Figure 6.36: The dilatation $\theta$ contour plot, t = 2.8 to 4.8 with increment 0.4 from (a) to (f).

according to its governing equation. The average traveling time for a Lagrangian element released at the inlet and reaching the outlet is approximately

$$t \approx |x_r - x_l|/u_0 = 2/0.4 = 5.$$

This means that the Lagrangian elements initially deployed in the computational domain are fully replaced after $t \approx 5.0$. All the individual elements traveling through the computational domain experience a similar history after the equilibrium is reached at $t \approx 4.8$. The elements in the computational domain at the equilibrium after $t = 5.0$ all entered the domain from the inlet $x_l = -1$ and subsequently evolved through the Lagrangian dilatation element method.

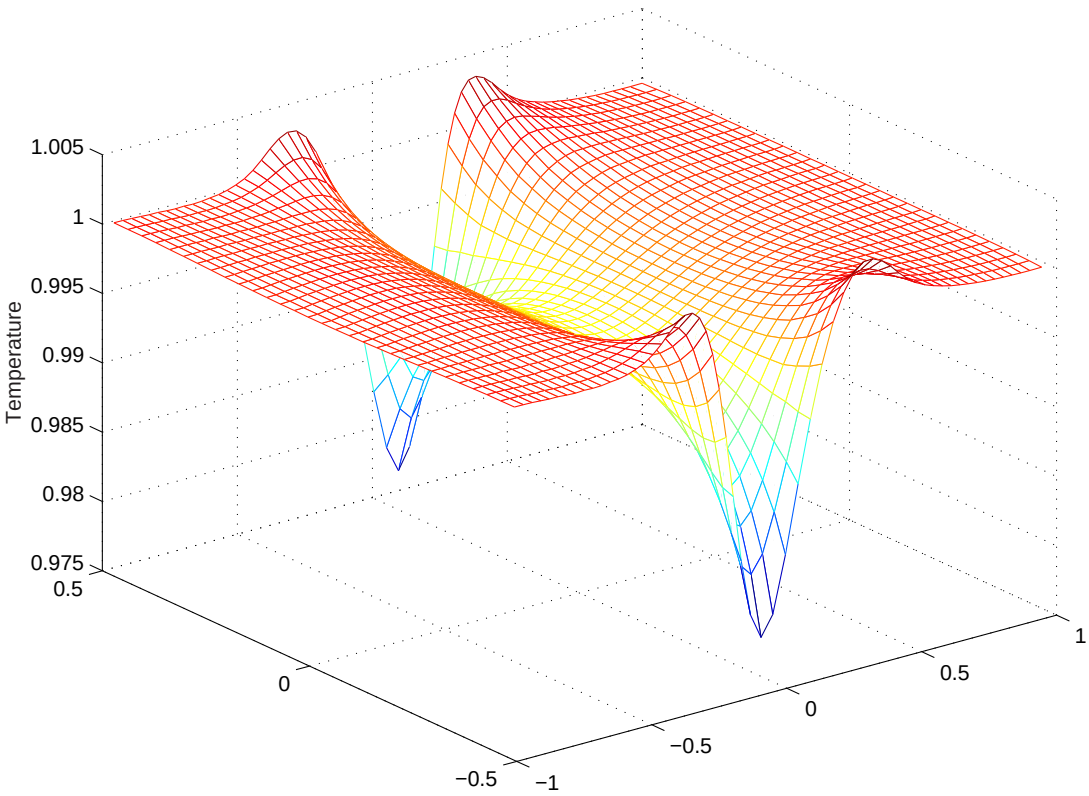



Figure 6.37: The mesh plot of temperature $T$ distribution at t = 5.0.

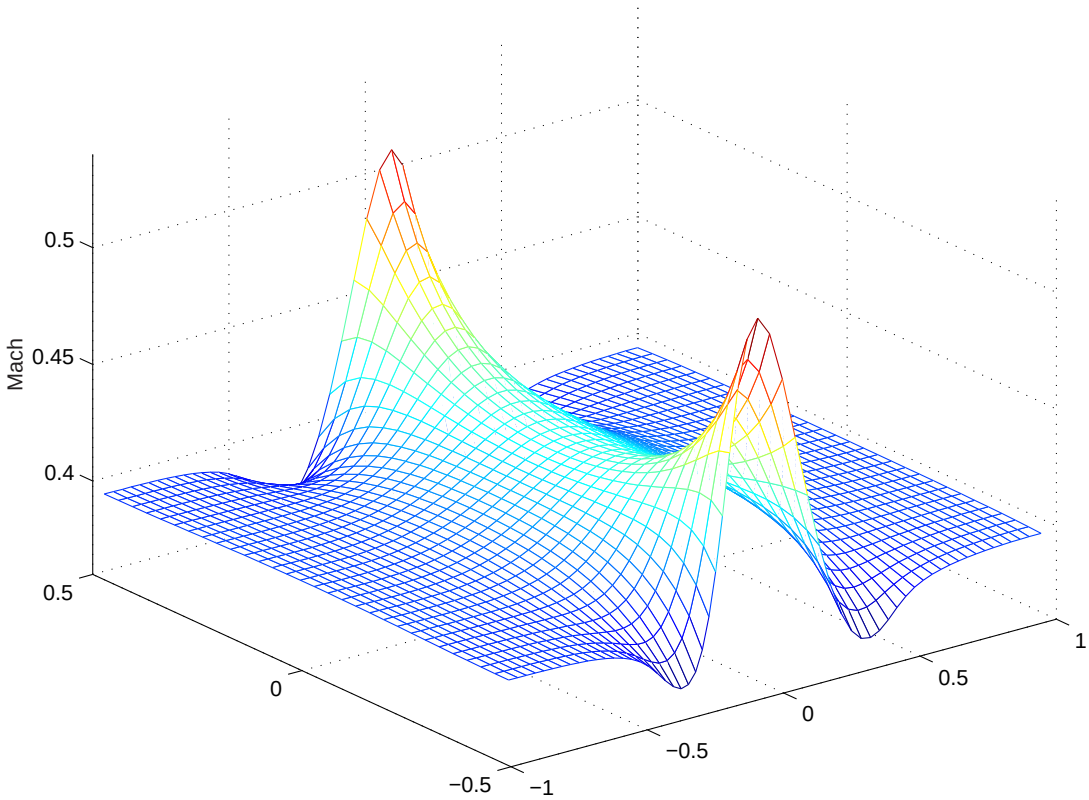



Figure 6.38: The mesh plot of Mach number distribution at t = 5.0.

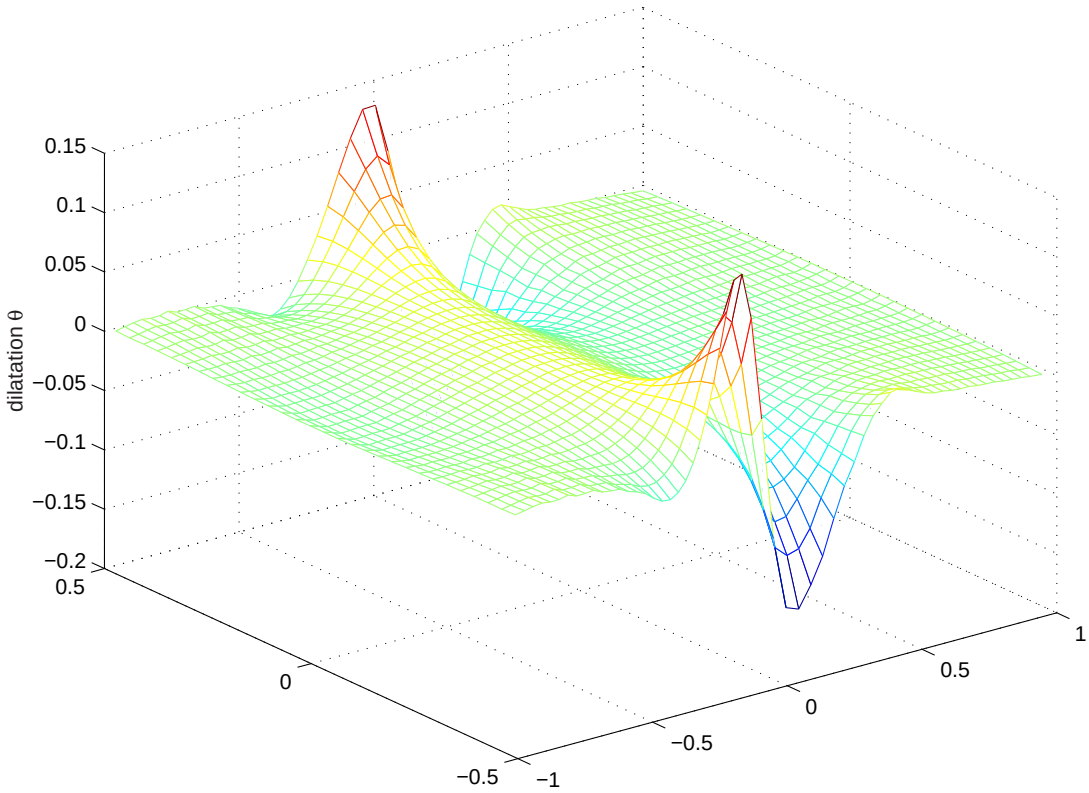



Figure 6.39: The mesh plot of dilatation $\theta$ distribution at t = 5.0.

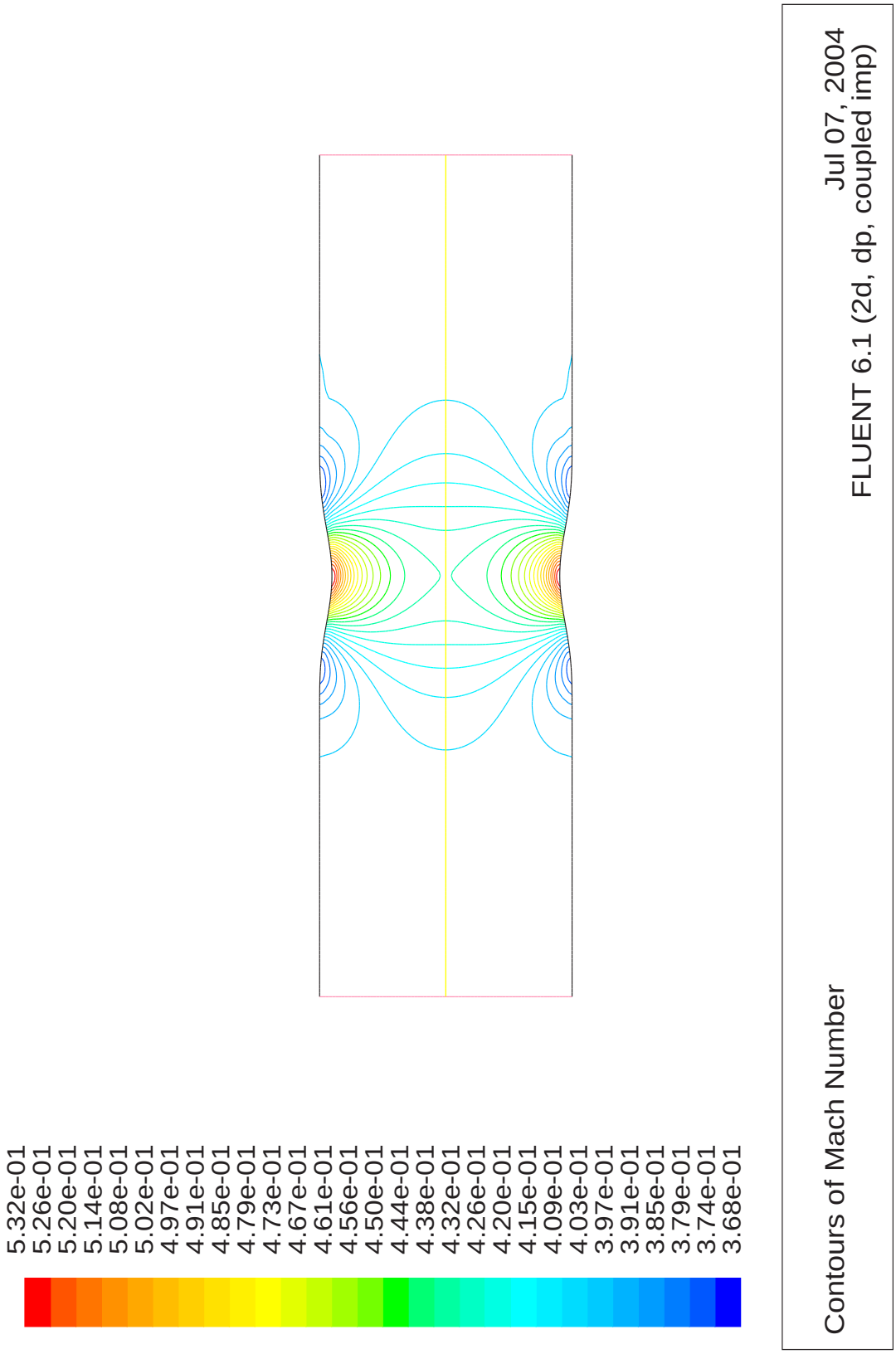



Figure 6.40: The mach contour plot by Fluent.

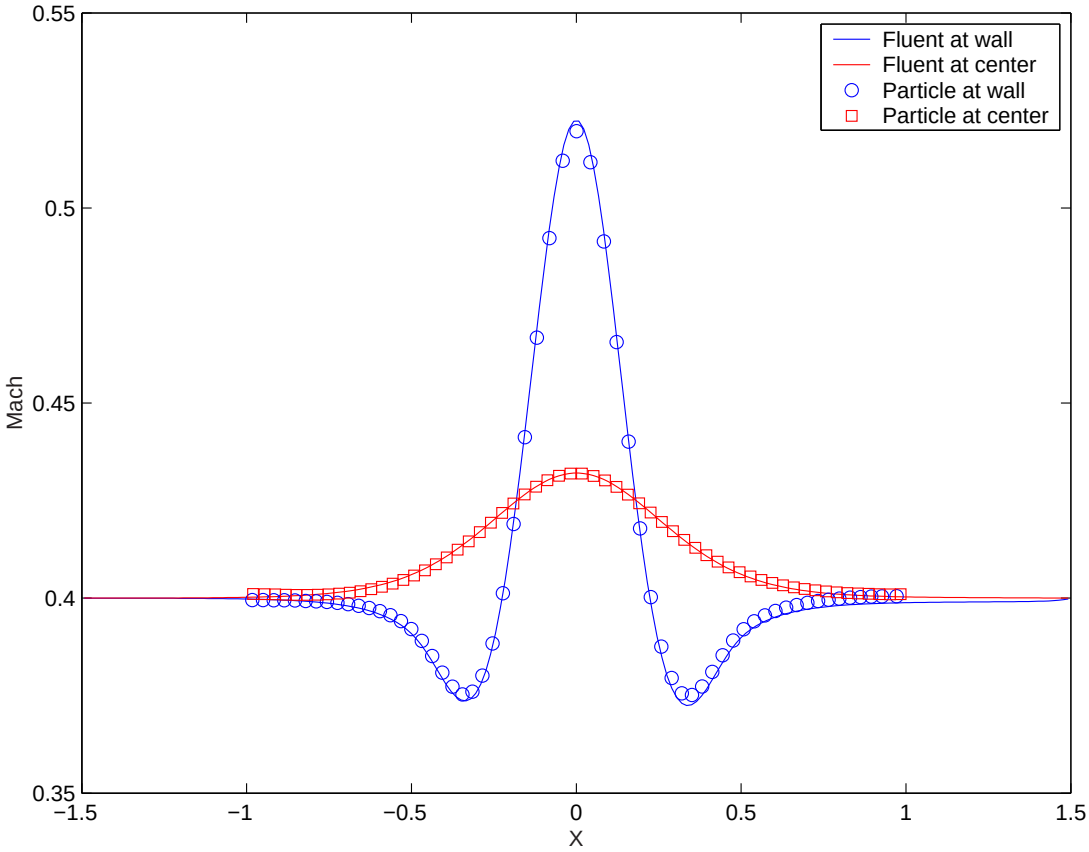



Figure 6.41: Comparison of the computed mach number at the wall and the centerline with the Fluent solution.

Mesh plots of temperature, Mach number and dilatation at $t = 5.0$ are shown in figures 6.37 - 6.39. The same problem is also computed using the commercial package Fluent. In this the x coordinates are set between [-1.5, 1.5]. A 2D axisymmetrical model is used for this case. An evenly spaced grid $200 \times 80$ is used to mesh the domain and just the steady state governing equation is solved by Fluent. The Mach contour plot of the Fluent solution is shown in figure 6.40. Figure 6.41 shows the comparison of the computed Mach number at the wall and the centerline compared with the results of Fluent. It is seen that they agree quite well.

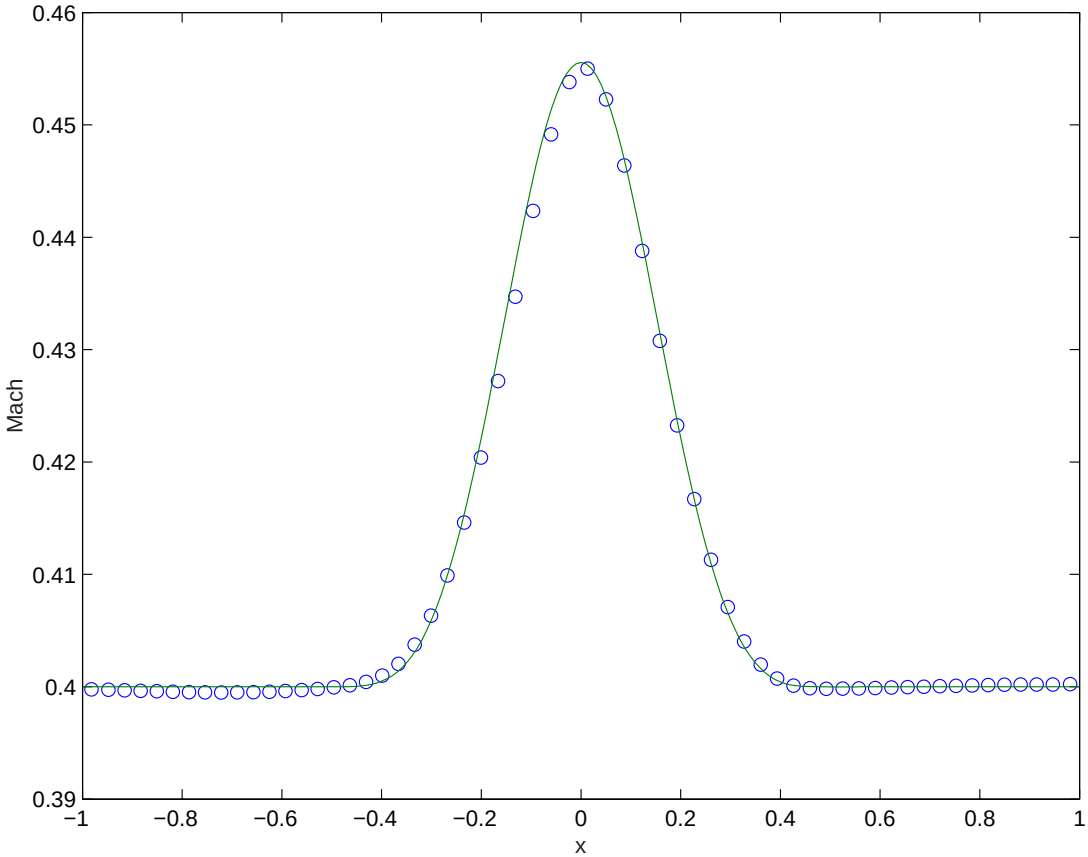



Figure 6.42: Average mach number across the channel from the 2D solution: (o), quasi-1D exact solution: (—).

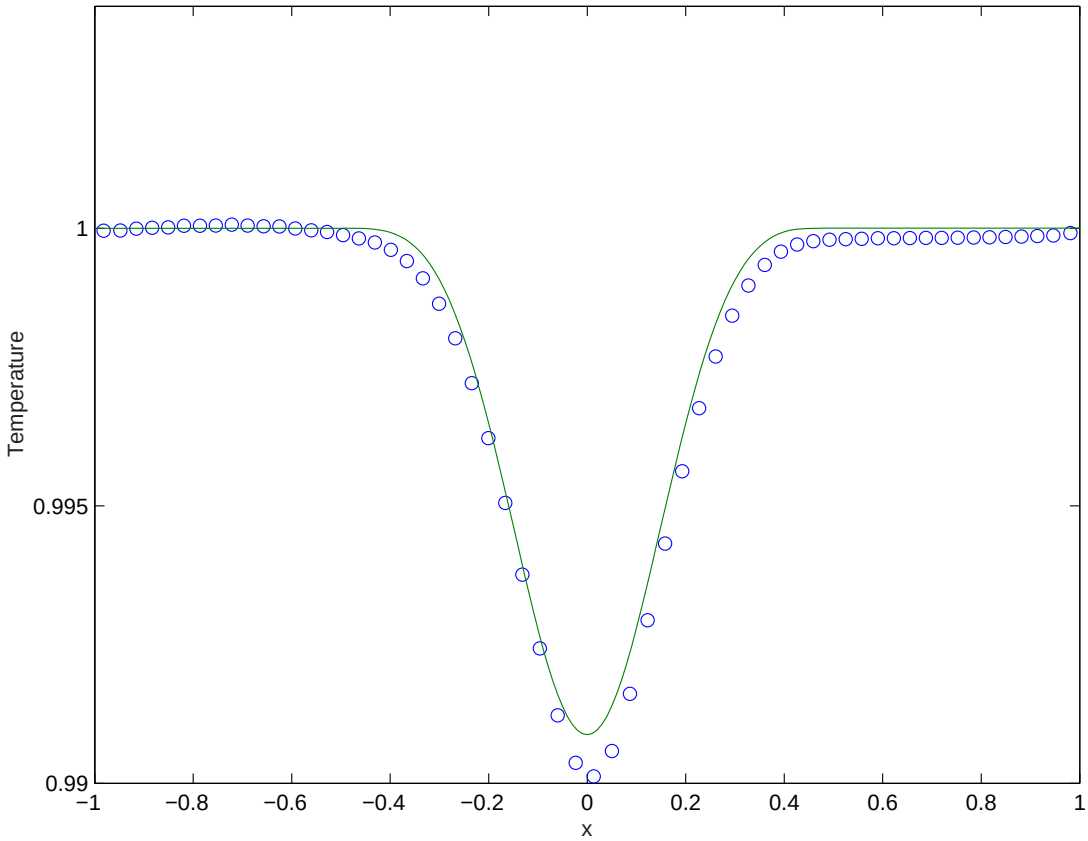



Figure 6.43: Average temperature across the channel from the 2D solution: (o), quasi-1D exact solution: (—).

Figures 6.42 and 6.43 are comparisons of the spanwise averaged mach number and temperature of the 2D solutions with the quasi-1D exact solutions. The agreement is also seen to be quite well.

Figures 6.44 - 6.46 present the convergence histories of the averaged temperature, mach number, and dilatation from $t = 0.2$ to 5.0 with increment 0.2. Characteristic waves as shown in these figures move upstream and downstream with speeds consistent with (5.29) and (5.30) as in the 1D case. The steady state is reached after the waves move out of the domain. The wiggles in these figures

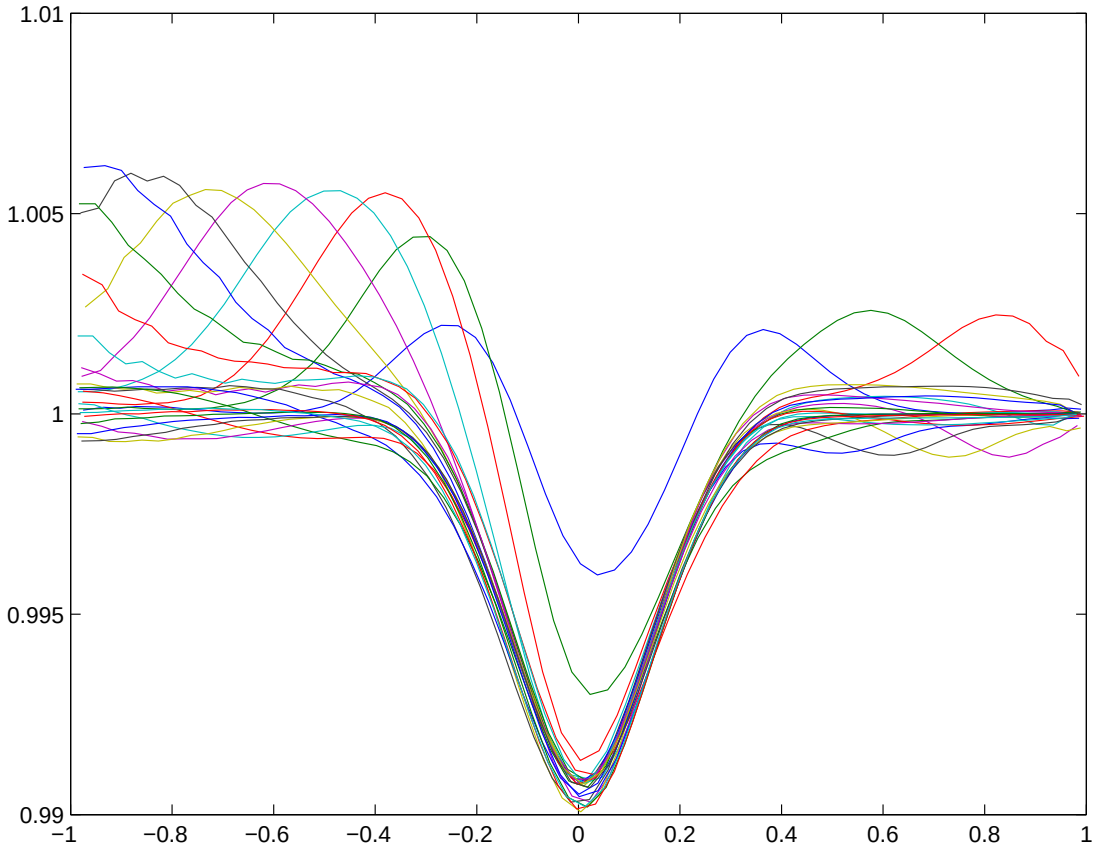



Figure 6.44: Convergence histories of spanwise average temperature from $t = 0.2$ to 5.0 with increment 0.2.

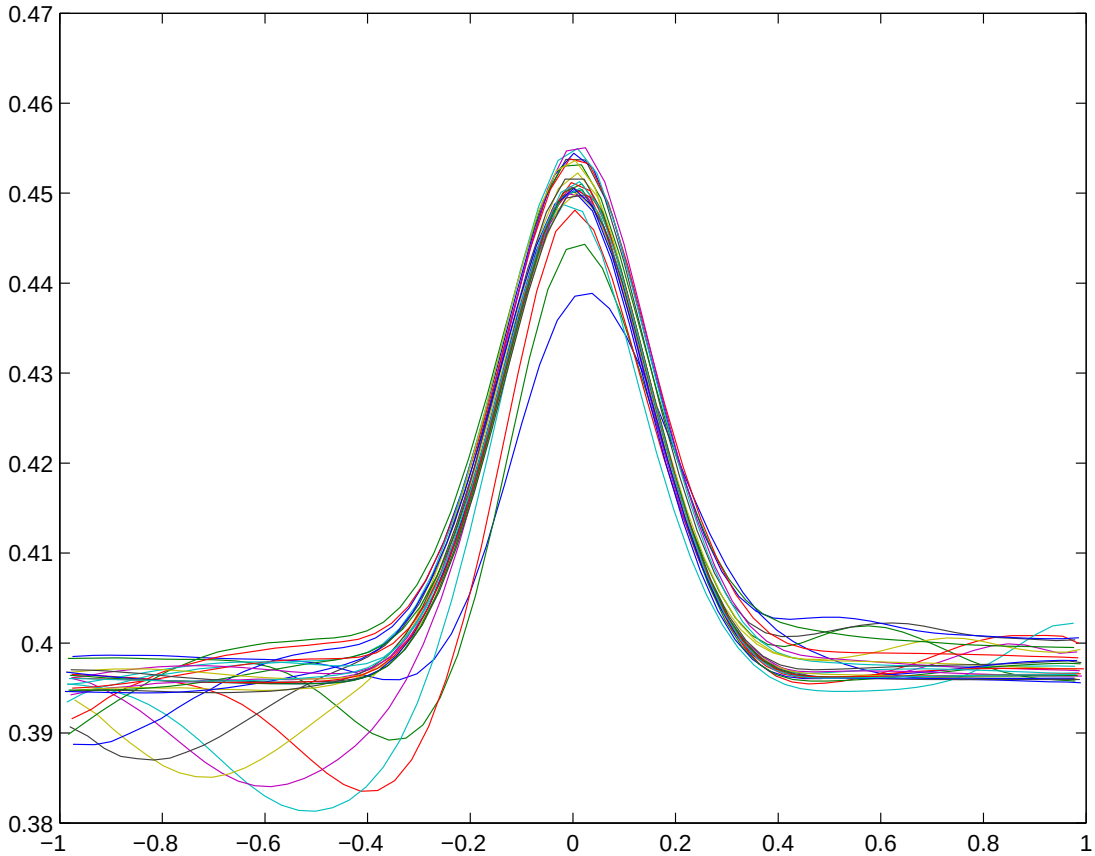



Figure 6.45: Convergence histories of spanwise average mach number from $t = 0.2$ to 5.0 with increment 0.2.

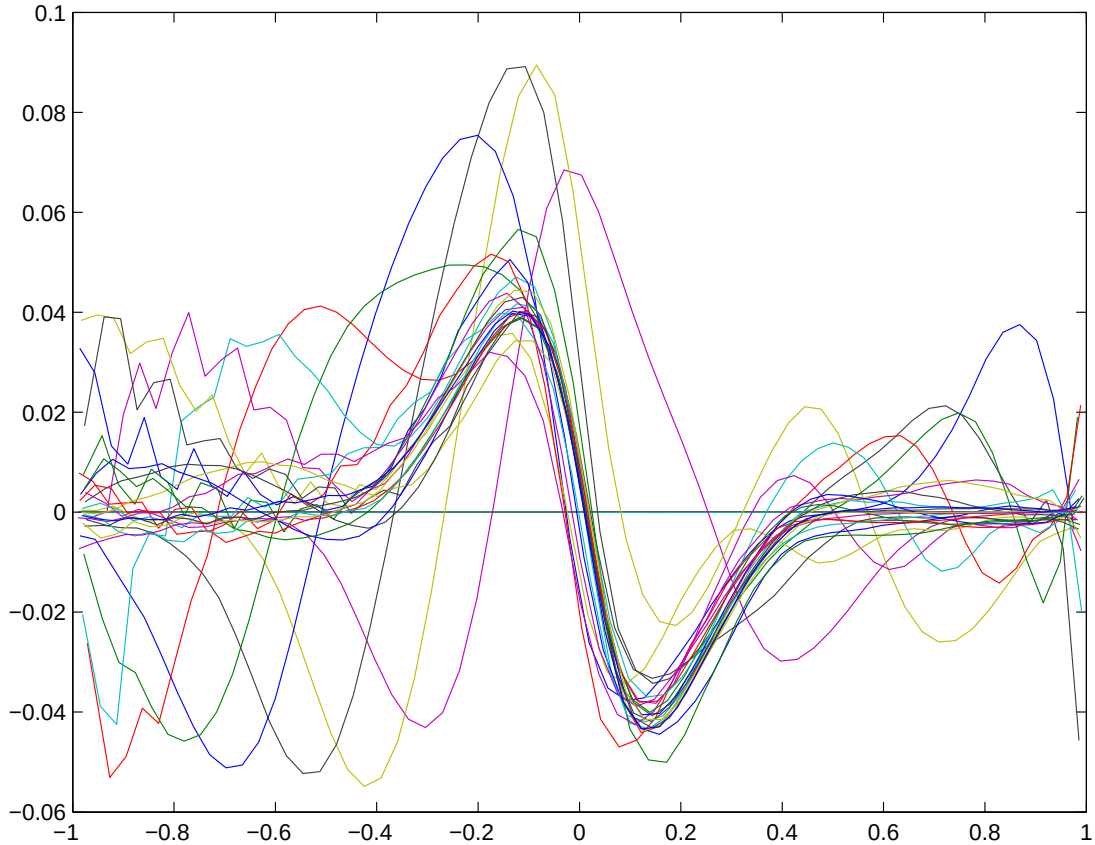



Figure 6.46: Convergence histories of spanwise average dilatation from $t = 0.2$ to 5.0 with increment 0.2.

originate at the inlet and outlet boundaries due to the implementation schemes of the inflow and outflow conditions (see Chapter 5 for details) are implemented. The more chaotic appearance of wiggles in the $\theta$ plot (Fig. 6.46) are due to the fact that by being a differential quantity it is more sensitive to perturbations. The wiggles tend to be stable and do not ruin the convergent steady-state solutions.

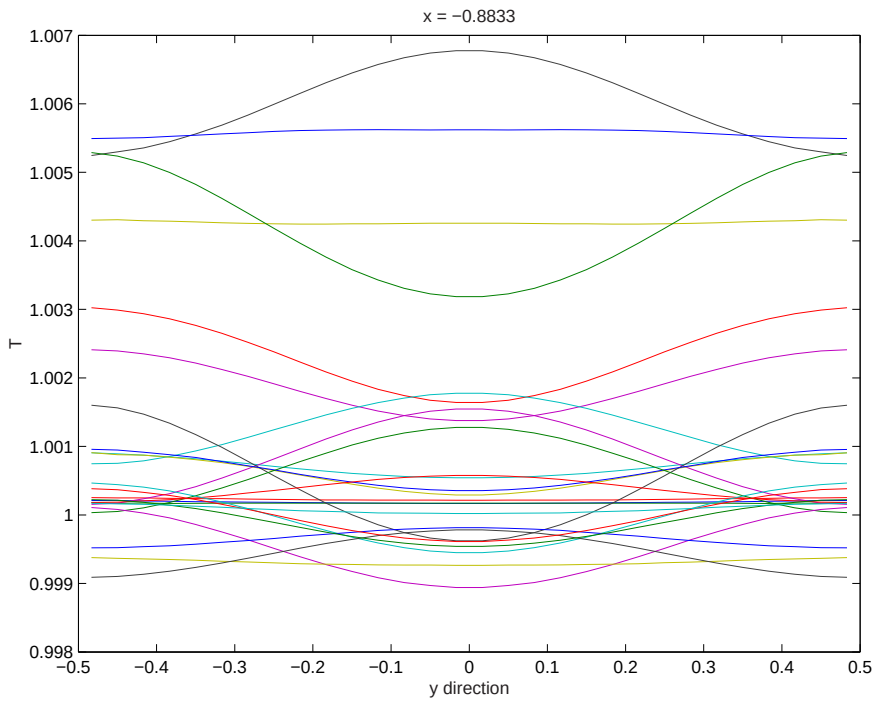



Figure 6.47: Convergence histories of temperature at $x = -0.8833$ along the spanwise direction of nozzle

Figures 6.47 - 6.51 show the spanwise temperature distribution at five different streamwise positions. These figures reveal that there are various waves reflecting between the walls of the nozzle. These type of spanwise waves that exist in higher dimensions are different than the streamwise characteristic waves in Figs 6.44 - 6.46. The spanwise waves appear to reflect up and down between the nozzle walls in the regions where the area change rate is not zero. As shown in Figure 6.49,

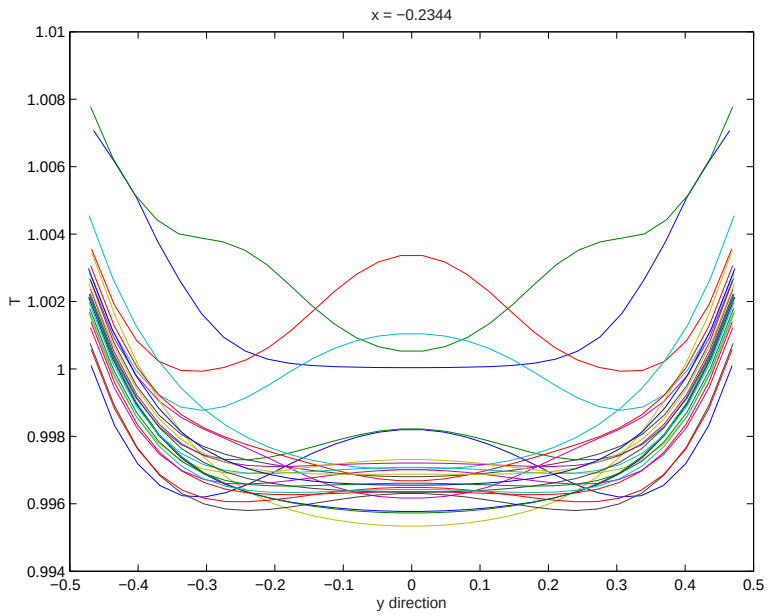



Figure 6.48: Convergence histories of temperature at $x = -0.2344$ along the spanwise direction of nozzle

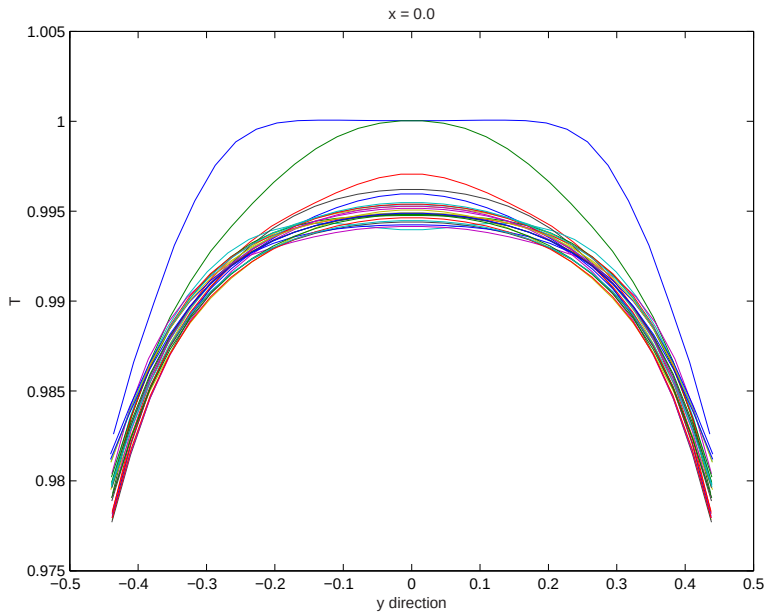



Figure 6.49: Convergence histories of temperature at $x = 0.0$ along the spanwise direction of nozzle

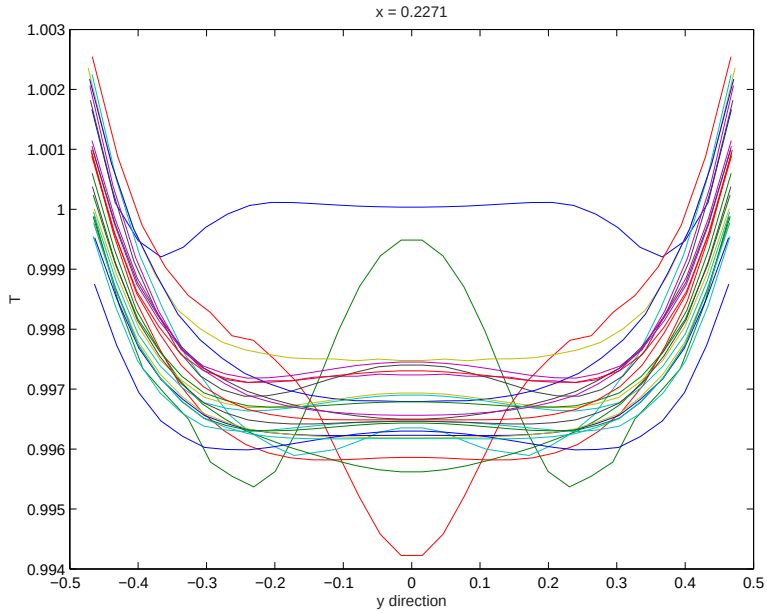



Figure 6.50: Convergence histories of temperature at $x = 0.2271$ along the spanwise direction of nozzle

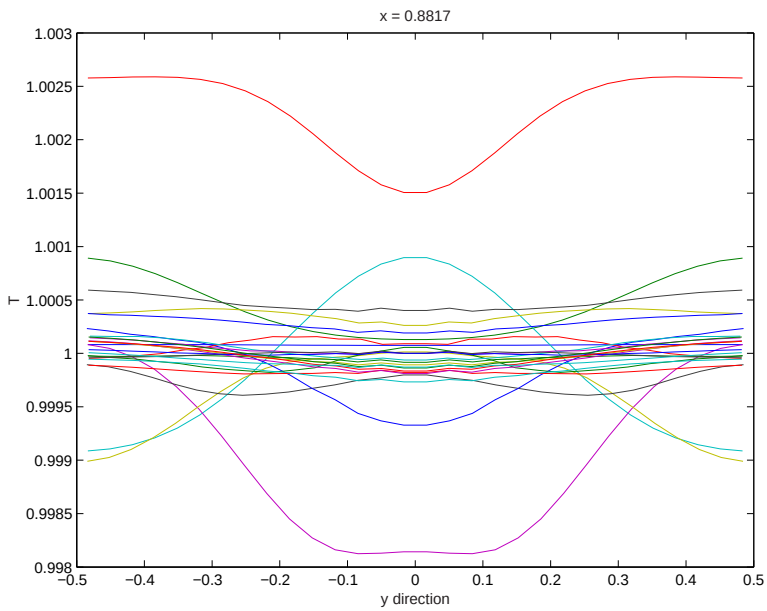



Figure 6.51: Convergence histories of temperature at $x = 0.8817$ along the spanwise direction of nozzle

there is little of such activity at the nozzle throat $x = 0$. This may be because the area change rate is zero at the throat.

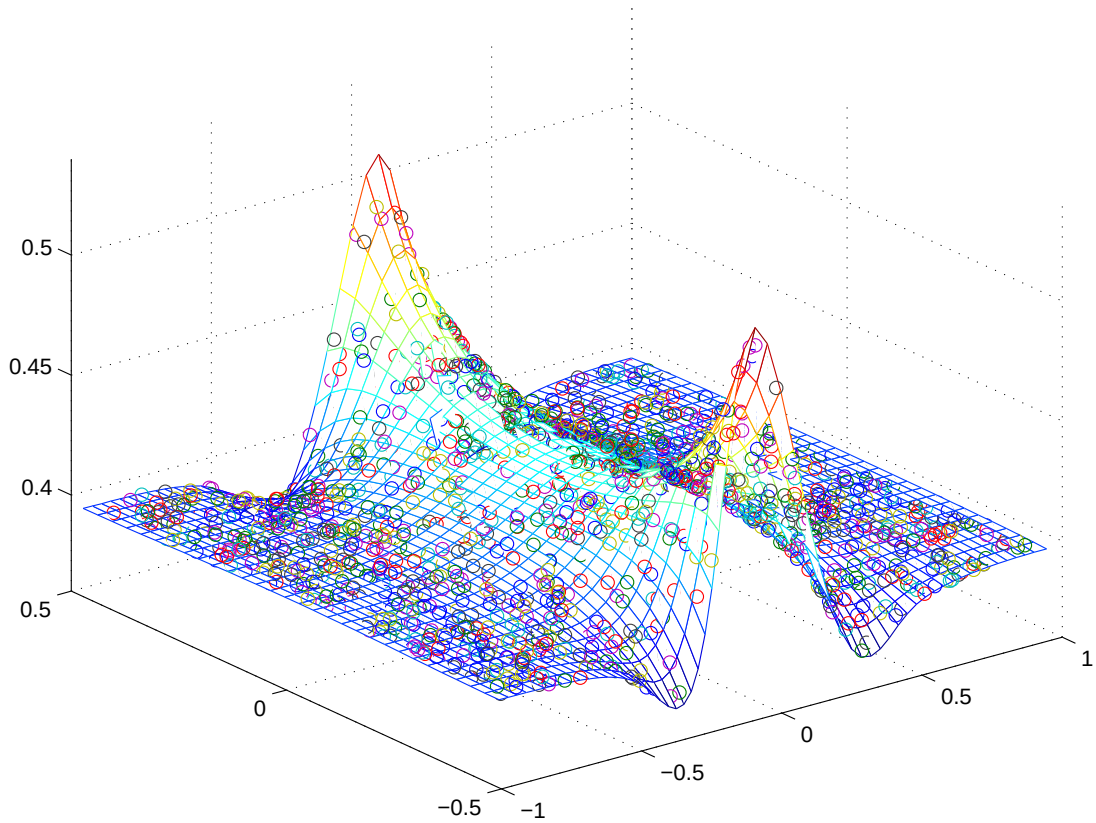



Figure 6.52: The Mach number plot which were interpolated into a fixed 60*30 mesh from a total of $1,800$ random distributed particles (o).

Simulations were also performed with more particles ($80 \times 40$). The results show no significant difference from those in figures 6.37 - 6.39. Computations were also done in which the domain was initially divided into randomly sized elements, and kept this way for all time by randomly varying the size of the incoming elements at the upstream boundary. In fact, the final results as shown in Figure 6.52 are indistinguishable from the previous plots based on more uniformly sized elements. Since each Lagrangian particle is convected on their own and the

connection between Lagrangian particles is built by the various derivatives in the governing equations. It may be concluded that the derivatives computed by the MLS are relatively insensitive to particle positions. This is consistent with the observations of Marshall and Grant in a related context [50] and suggests that the least-square fitting used in this research, after appropriate generalization, may also succeed in the context of three-dimensional turbulent flow.

## 6.4 Discussion

The grid-free Lagrangian dilatation element method for compressible flow has been derived and demonstrated to work well for the oscillating waves in an enclosed domain and subsonic nozzle flow in one and two dimensions. The example flows are inviscid and vorticity free, so the computational elements were limited to carry the dilatation, temperature and density only. The inclusion of viscous effects adds the need for solving the vorticity equation. Technically, there is no reason the present method cannot be generalized to include viscous compressible flow. The diffusion terms in the temperature equation and the viscous terms in the dilatation and vorticity equations only introduce additional spatial derivatives to be evaluated. The moving least square method developed in Chapter 4 should be able to adequately handle them. The no-slip velocity boundary condition can be simulated by adding a layer of vortex sheets on the solid boundary similar to the situation encountered in incompressible viscous flow.

One limitation of the present research is that it is constricted to shock-free subsonic compressible flow problems due to the physics of the Lagrangian governing equations and the treatment of spatial derivatives. Shock waves are a

common and important physical phenomenon that as yet cannot be simulated in the context of the present scheme. How to capture shock waves based on the framework developed in this research is given a brief discussion in the following section.

### 6.4.1 Shock wave capture

The present dilatation element method cannot successfully capture discontinuous solutions like shock waves for two reasons. First, the governing equations are not conservative. Secondly, the treatment of spatial derivatives of discontinuous functions by the moving least square method smooths out sharp jumps and tends to introduce unnecessary numerical oscillations leading possibly to instability. The simplest way to handle this problem is to add an artificial viscous term to the inviscid Lagrangian governing equations to smooth the discontinuity and thus smear the sharp jump.

Here, an isentropic shock tube is used to illustrate how to modify the method to capture shock waves. From the isentropic assumption, there exists the $\gamma$ law

$$p = C\rho^{\gamma},$$

where the constant $C = \frac{a_0^2}{\rho_0}$ and $a_0$ is the local initial sonic speed and $\rho_0$ is the initial density. With this relation, equations (2.24)-(2.26) are simplified to:

$$\frac{D\rho}{Dt} = -\rho\theta \tag{6.23}$$

$$\frac{D\theta}{Dt} = -\theta^2 - C\gamma\rho^{\gamma-3}[\rho\rho_{xx} + (\gamma-2)\rho_x^2] \tag{6.24}$$

The gas constant $C = 1$ and $\gamma = 1.4$ and the computational domain is from 0 to 1. A diaphragm is located at the center $x_0 = 0.5$, and the initial states on the

left and right sides of the diaphragm are:

$$\rho_{left} = 10, \qquad \rho_{right} = 8.0 \tag{6.25}$$

$$u_{left} = 0.0, \qquad u_{right} = 0.0. \tag{6.26}$$

An artificial viscosity term

$$\frac{1}{Re}\theta_{xx}$$

is added to the right-hand side of the dilatation equation (6.24), and the inviscid Euler equations are modified into viscous equations that give stable solutions after tuning the viscosity parameter $Re$. The simulation is run with $N = 101$ evenly displaced Lagrangian elements. the time step $\Delta t = 0.01$ and the time of integration with an explicit Euler scheme is from $t = 0$ to 0.2. After adjusting $Re$ to be between $100 - 200$, the Lagrangian dilatation element method is found to successfully capture the discontinuous solution although sharp jumps are smoothed. Thus, figures 6.53 and 6.54 show that with the addition of the artificial viscosity term, the Lagrangian dilatation element method successfully computes the shock waves.

### 6.4.2 Future work

The above discussion shows that the dilatation element method can accommodate discontinuous solutions with the use of an artificial viscosity term. However, due to its lack of sophistication the discontinuous solutions are greatly smeared and the viscosity needs to be carefully tuned. Therefore, it is clear that a better shock capturing scheme is needed. To do this the moving least square method needs to be modified to better handle the spatial derivatives near the discontinuous solutions. Moreover, it is known from the study of partial differential equations that

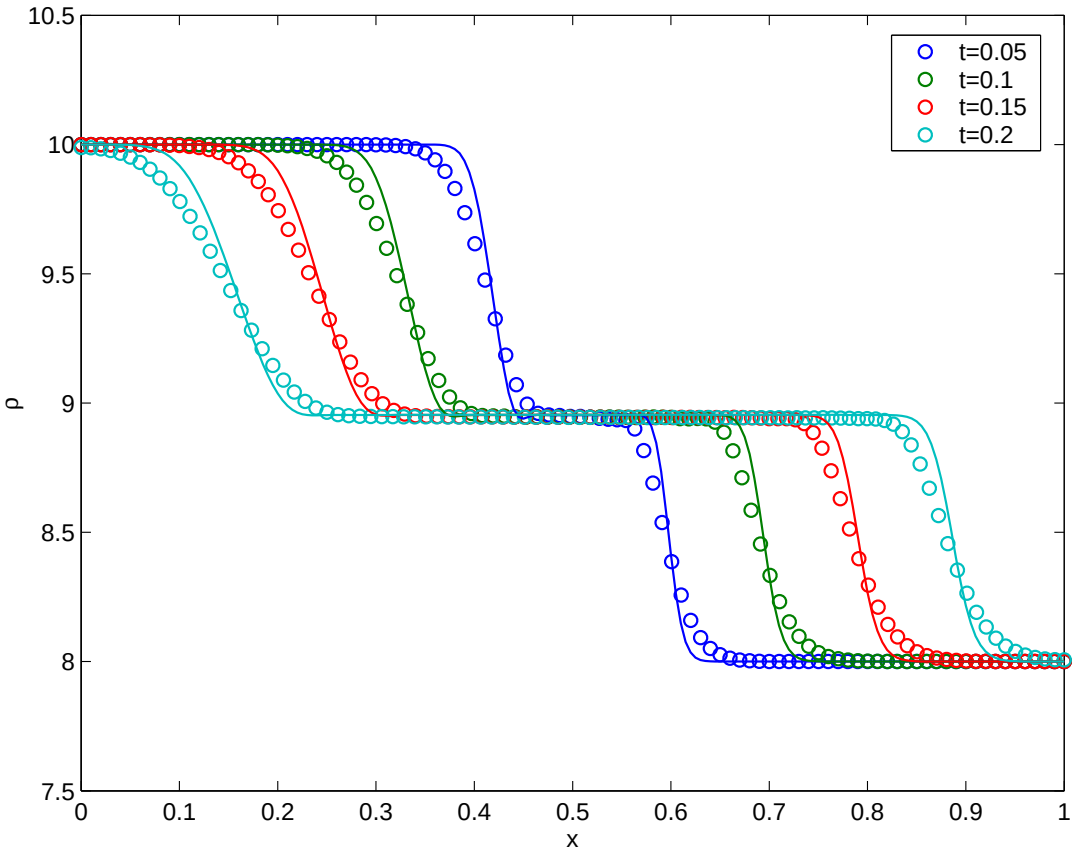



Figure 6.53: The computed density (o) compared with Godunov scheme (—).

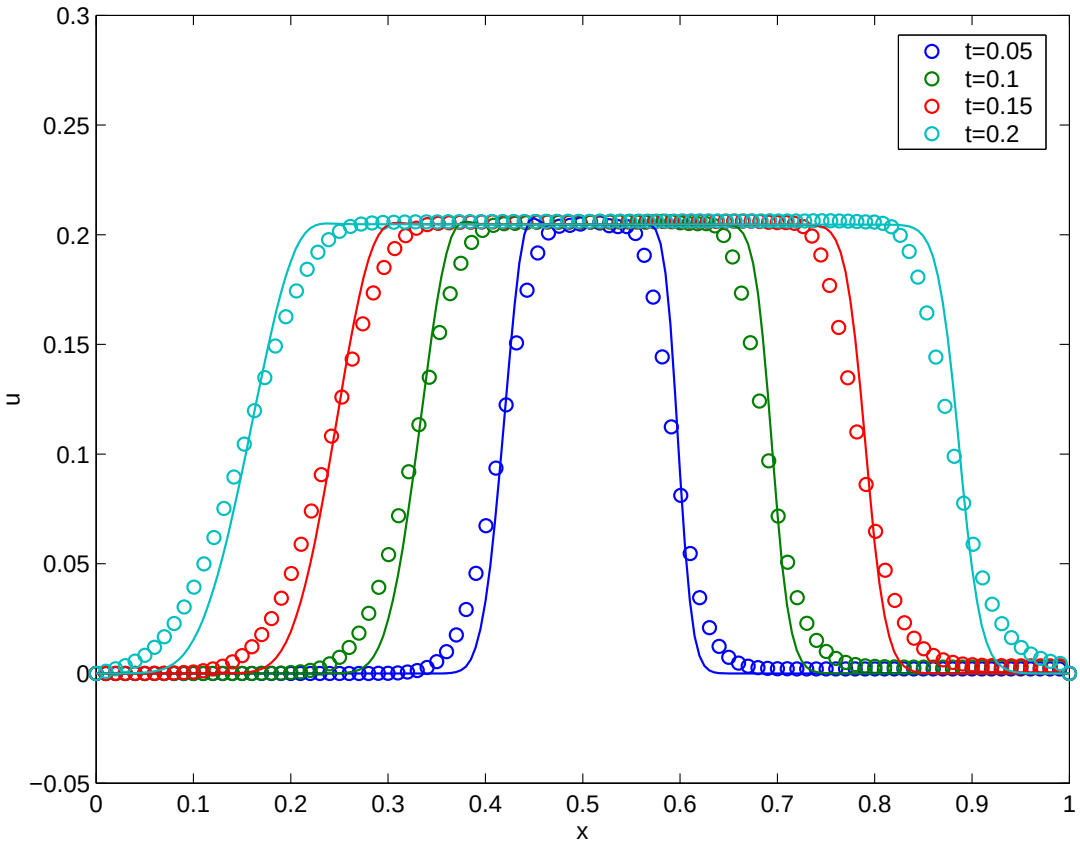



Figure 6.54: The computed velocity (o) compared with Godunov scheme (—).

discontinuous solutions cannot be successfully computed from non-conservative governing equations. It might thus be necessary to reformulate the governing equations to accommodate the discontinuous solutions.

It has been shown that the Lagrangian dilatation element method can successfully predict inviscid compressible flows such as oscillating compressible waves in an enclosed tube and subsonic nozzle flow. This suggests that it can also succeed when generalized for viscous compressible flow. In fact, it can be expected that the viscous terms are not likely to cause instability, and the moving least square fitting should be able to handle the spatial derivative terms appearing in them. The no-slip velocity boundary condition can be enforced in this case by adding vortex sheet layers on the wall boundaries.

Grid-free methods do not depend on a fixed spatial grid system as in traditional grid-based schemes such as finite difference, finite volume, and finite element methods. Moreover it treats problems with very complicated geometries with no greater effort than simplified boundaries. The insensitivity to time step of the integration scheme and also the number of computational elements makes it as efficient as traditional grid-based schemes because it can use a relatively larger time step and a smaller number of particles. Considering the great amount of time spent on building suitable grids for traditional grid-based schemes, the present method may be more efficient in computing complex geometry problems. It should be interesting to investigate how this method can be generalized to viscous compressible flow and thus bring the benefit of being grid-free to this case. Furthermore, deriving a scheme based on this method to simulate compressible turbulent flow should be the end goal of this research. Similar to the simulation of incompressible turbulence, the Lagrangian computational elements can be ex-

pected to have the advantage of clustering in positions having large gradients. In grid-based LES or DNS schemes it is necessary to a priori provide this fine resolution. Thus the grid-free scheme is well suited for modeling compressible turbulence in complex geometry problems which would be very difficult for grid-based LES and DNS. Moreover research on grid-free simulation of compressible turbulent flow should provide a new perspective for understanding turbulence. It may ultimately help make this method a practical alternative to traditional grid-based schemes.

# BIBLIOGRAPHY

[1] Aluru NR, Li G, Finite cloud method: a true meshless technique based on a fixed reproducing kernel approximation, Int. J. Numer. Meth. Engng 2001; 50:2373-2410.

[2] Anderson JD Jr., Computational Fluid Dynamics, McGraw-Hill, New York, 1995.

[3] Babuska I and Melenk JM, The partition of unity method, Int. J. Numer. Meths. Eng. (1997), Vol. 40, p. 727-758.

[4] Batchelor GK, An introduction to fluid dynamics, Cambridge Univerisity Press, 1967.

[5] Belytschko T, Lu YY, Gu L., Element free Galerkin methods, Int. J. Numer. Meth. Engng 1994; 37:229-256.

[6] Belytschko T, Krongauz Y, Organ D, Fleming M, Krysl P, Meshless methods: an overview and recent developments, Computer Methods in Applied Mechanics and Engineering 1996; 139:3-47.

[7] Bernard PS, A deterministic vortex sheet method for boundary layer flow, J. Comput. Phys. (1995), Vol. 117, p. 132-145

[8] Bernard PS, Dimas AA, Lottati I, Vortex method analysis of turbulent flows, Proceedings of The First International Conference on Vortex Method, Nov. 4-5, 1999, Kobe, Japan, p. 79-91.

[9] Bernard PS, Dimas AA, Vortex method modeling of complex, turbulent engineering flows, Proceedings of The Second International Conference on Vortex Method, Sept. 26-28, 2001, Istanbul, Turkey, p. 41-54.

[10] Bernard PS, Wallace, JM, Turbulent flow: analysis, measurement, and prediction, John Wiley, 2002.

[11] Bernard PS, Collins JP, Krispin J, Gridfree simulation of turbulent boundary layers using VorCat, in: 16th AIAA Computational Fluid Dynamics Conference, Orlando, FL, 2003, AIAA Paper No. 2003-3424.

[12] Bernard PS, Potts MA, Krispin J, Studies of turbulent mixing using the VorCat implementation of the 3D vortex method, in: 33rd AIAA Fluid Dynamics Conference, Orlando, FL, 2003, AIAA Paper No. 2003-3599.

[13] Bernard PS, Collins P, Potts M, Prediciton of external flows using the VorCat formulation of the vortex method, AIAA Paper No. 2004-2553.

[14] Bernard PS, Shen J, A grid-free dilatation element method for quasi-one-dimensional gas dynamics, J. Comput. Phys. (2004), Vol. 199, p. 41-65.

[15] Choquin JP, Lucquin-Desreux B, Accuracy of a deterministic particle method for the Navier-Stokes equations, Inter. J. Num. Meth. Fluids (1988), Vol. 8, p. 1439-1458.

[16] Chorin AJ, Numerical study of slightly viscous flow, J. Fluid Mechanics (1973), Vol. 57, p.785-796.

[17] Chorin AJ, Vortex sheet approximation of boundary layers, J. Comput. Phys. (1978), Vol. 27, p.428-442.

[18] Chorin AJ, Vortex models and boundary layers instability, SIAM J. Sci. Stat. Comput. (1980), Vol. 1, p.1-21.

[19] Chorin AJ, The evolution of a turbulent vortex, Commun. Math. Phys. (1982), Vol. 35, p. 517-535

[20] Chorin AJ, Hairpin removal in vortex interaction I, J. Comput. Phys. (1990), Vol. 91, p.1-21.

[21] Chorin AJ, Hairpin removal in vortex interaction II, J. Comput. Phys. (1993), Vol. 107, p.1-9.

[22] Cottet GH, Mas-Gallic S, A particle method to solve the Navier-Stokes system, Numer. Math. (1990), Vol. 57, p. 805-827.

[23] Cottet GH, Koumoutsakos P, Vortex methods: theory and practice, Cambridge University Press, 2002.

[24] Degond P and Mas-Gallic S, The weighted particle method for convection-diffusion equations, Part 1: The case of an isotropic viscosity, Math. Comp., Vol. 188 (53), p.485-507.

[25] Dilts GA, Moving least squares particle hydrodynamics I: consistency and stability, Int. J. Numer. Meth. Engng 1999; 44:1115-1155.

[26] Duarte CA and Oden JT, Hp clouds - a meshless method to solve boundary - value problems, Technical Report 95-05, University of Texas at Austin.

[27] Eldredge JD, A dilatating vortex particle method for compressible flow with application to aeroacoustics, Ph. D thesis, California Institute of Technology, Pasadena, California, 2002.

[28] Eldredge JD, Colonius T, Leonard A, A vortex particle method for two-dimensional compressible flow, J. Comput. Phys. (2002) Vol. 179, p. 371-399.

[29] Eldredge JD, Leonard A, Colonius T, A general deterministic treatment of derivatives in particle methods, J. Comput. Phys. (2002) Vol. 180(2), p. 686C709.

[30] Eldredge JD, Colonius T, Leonard A, A dilating vortex particle method for compressible flow," J. Turbul. (2002) No. 3, Art. 036.

[31] Fan J and Gijbels I, Local polynomail modelling and its applications, Chapman & Hall, 1996

[32] Fishelov D, A new vortex scheme for viscous flows, J. Comput. Phys. (1990), Vol. 86, p. 211-224

[33] Garabedian PR, Partial differential equations, Chelsea, New York, 1986.

[34] Ghoniem AF, Chorin AJ, Oppenheim AK, Numerical modeling of turbulent flow in a combustion tunnel, Phil. Trans. Roy. Soc. (1982) Vol. 304, p. 303-325.

[35] Ghoniem AF, Grid-free simulation of diffusion using random walk methods, J. Comput. Phys. (1985), Vol. 61, p. 1-37.

[36] Ghoniem AF, Heidarinejad G, Krishnan A, Numerical simulation of a thermally stratified shear layer using the vortex element method, J. Comput. Phys. (1988), Vol. 79, p. 135-166.

[37] Gossler AA, Moving least squares: a numerical differentiation method for irregularly spaced calculation points, Sandia National Laboratories Report SAND (2001) 2001-1669.

[38] Gingold RA and Monaghan JJ, Smoothed particle hydrodynamics, theory and application to non-spherical stars, Mon. Not. Roy. Astr. Soc. (1977), Vol. 181, p. 375-389.

[39] Goodman J, Hou TY, Lowengrub J, The convergence of the point vortex method for the 2-D Euler equations, Commun. Pure Appl. Math (1990), Vol. 43, p. 415-430.

[40] Greengard L and Rokhlin V, A fast algorithm for particle simulations, J. Comput. Phys. (1980), Vol. 37(3), p. 289-335.

[41] Griebel M, Schweitzer MA (eds.), Meshfree methods for partial differential equations, Lecture notes in computational science and engineering, Springer 2003.

[42] Gumerov N. and Duraiswami R, Class notes of CMSC878R: Fast Multipole Method, UMIACS, University of Maryland, College Park, 2002.

[43] Hess JL and Smith AMO, Calculation of potential flow about arbitrary bodies, Progress in Aeronautical Sciences, Vol.8 (D. Kuchemann, Ed.), Pergamon Press, 1967.

[44] Leonard A, Vortex methods for flow simulation, J. Comput. Phys. (1980), Vol. 37, p. 289-335.

[45] Leonard A, Computing three dimensional flows with vortex elements, Ann. Rev. Fluid Mech. (1985), Vol. 17, p. 523-559.

[46] Leonard A, Vortex methods for flow simulation, J. Comput. Phys. (1987), Vol. 73(2), p. 325-348.

[47] Liepmann HW, Roshko A, Elements of gasdynamics, John Wiley, 1962.

[48] Liszka T, Orkisz J, Finite difference method at arbitrary irregular grids and its application in applied mechanics, Computers and Structures 1980; 11:83-95.

[49] Lohner R. et al, A finite point method for compressible flow, Int. J. Numer. Meth. Engng 2002; 53:1765-1779.

[50] Marshall JS and Grant JR, A lagrangian vorticity collocation method for viscous, axisymmetric flows with and without swirl, J. Comput. Phys. (1997), Vol. 138, p. 302-330.

[51] Melenk JM and Babuska I, The partition of unity finite element method: basic theory and applications, Comput. Meths. Appl. Mech. Engrg. (1996), Vol. 139, p. 289-314.

[52] Monaghan JJ and Gingold RA, Shock simulation by the particle method SPH, J. Comput. Phys. (1983), Vol. 52, p. 374-389.

[53] Monaghan JJ, Particle methods for hydrodynamics, Comp. Phys. Rep. (1985), Vol. 3, p. 71-124.

[54] Monaghan JJ, On the problem of penetration in particle methods, J. Comput. Phys. (1989), Vol. 82, p. 1-15.

[55] Monaghan JJ, SPH and Riemann solvers, J. Comput. Phys. (1997), Vol. 136, p. 298-307.

[56] Nitsche M, Strickland JH, Extension of the gridless vortex method into the compressible flow regime, J. Turbul. (2002) Vol. 3, Art. No. 001.

[57] Ogami Y and Cheer A, Simulations of unsteady compressible fluid motion by an iteractive cored particle method, SIAM J. Appl. Math. (1995), Vol. 55, p. 1204-1226.

[58] Onate E, Idelsohn S, Zienkiewicz OC, Taylor RL, A finite point method in computational mechanics: Application to convective transport and fluid flow, Int. J. Numer. Meth. Engng 1996; 39:3839-3866.

[59] Onate E, Idelsohn S, Zienkiewicz OC, Taylor RL, Sacco C, A stabilize finite point method for analysis of fluid mechanics problems, Computer Methods in Applied Mechanics and Engineering 1996; 139:315-346.

[60] Puckett EG, Vortex methods: a introduction and survey of selected research topics, Incompressible Computational Fluid Dynamics - Trends and Advances (ed. by M. D. Gunzburger and R. A. Nicolaides), Cambridge University Press (1993), p.335-408

[61] Rosenhead L, The formation of vortices from a surface of discontinuity, Proc. R. Soc. Lon., A (1931) Vol.134, p170-192.

[62] Schweitzer MA, A parallel multilevel partition of unity method for elliptic partial differential equations, Lecture notes in computational science and engineering, Springer 2003.

[63] Sod GA, Review: A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws, Journal of Computational Physics 27, 1-31.

[64] Strickland JH, Gridless compressible flow: a white paper, SANDIA REPORT SAND2001-0527, Sandia National Laboratories, Albuquerque, New Mexico 87185.

[65] Strickland JH, A 3-D fast solver for arbitrary vorton distributions, SANDIA REPORT SAND93-1641, Sandia National Laboratories, Albuquerque, New Mexico 87185.

[66] Toro EF, Riemann solvers and numerical methods for fluid dynamics : a practical introduction (2nd ed.), Spring, New York, 1999. Shock-capturing

[67] Toro EF, Shock-capturing methods for free-surface shallow flows, John Wiley, New York, 2001.

[68] Zerroukat M, Djidjeli K, Charafi A, Explicit and implicit meshless methods for linear advection-diffusion-type partial differential equations, Int. J. Numer. Meth. Engng 2000; 48:19-35.