A TWO-DIMENSIONAL TIME-DEPENDENT EULER SOLVER FOR MOVING BOUNDARIES IN CARTESIAN GRIDS APPLIED TO INJECTION DRIVEN INTERNAL FLOWS

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

KEREM PEKKAN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
IN
THE DEPARTMENT OF MECHANICAL ENGINEERING

NOVEMBER 2000

Approval of the Graduate School of Natural and Applied Scineces

Prof. Dr. Tayfur ÖZTÜRK
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Ediz PAYKOÇ
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Doctor of Philosophy.

Prof. Dr. Ahmet Ş. ÜÇER
Supervisor

Examining Committee Members

Prof. Dr. Ahmet Ş. ÜÇER

Prof. Dr. Cahit ERALP

Prof. Dr. Nafiz ALEMDAROĞLU

Prof. Dr. Haluk AKSEL

Prof. Dr. Ercan ATAER

ABSTRACT

A TWO-DIMENSIONAL TIME-DEPENDENT EULER SOLVER FOR MOVING BOUNDARIES IN CARTESIAN GRIDS APPLIED TO INJECTION DRIVEN INTERNAL FLOWS

Pekkan, Kerem

Ph.D., Department of Mechanical Engineering

Supervisor: Prof. Dr. Ahmet Ş. ÜÇER

November 2000, 167 pages

A Cartesian grid Euler solver is developed for 2D applications involving arbitrary combinations of stationary and moving boundaries. Some standard Cartesian grid procedures are revised. The systematic handling of degenerate cells, which is the requirement for a practical moving boundary Cartesian solver is accomplished. Boundary offsetting in Cartesian grids with examples from special applications is presented. Flux calculation procedures in finite volume solvers, for stationary-grid, moving-grid and moving solid boundaries are reviewed utilizing the fundamental Godunov method with an exact Riemann solver. Selected examples illustrate independently complex geometry and boundary movement handling features. The solution method is progressively developed, leading to applications in multi-dimensions and applied to axisymmetric internal flows in Solid Propellant Rocket Motors.

Keywords: Cartesian grid methods, Euler solutions, Finite volume, Moving boundary problems, Offsetting, Grain Burnback, Internal Flows, Solid Propellant Rocket Motors, Axisymmetric. Riemann Solvers

ÖZ

HAREKET EDEN SINIRLARDA İKİ BOYUTLU ZAMANA BAĞLI KARTEZYEN AĞ EULER CÖZÜCÜSÜ VE GAZ ENJEKSİYONUNA DAYALI İÇ AKIŞLARA UYGULANMASI

Pekkan, Kerem

Doktora, Makina Mühendisliği Bölümü

Tez Danışmanı: Prof. Dr. Ahmet ÜÇER

Kasım 2000, 167 sayfa

Hareket eden ve sabit sınırların herhangi değişik geometrilerdeki kombinasyonunu kapsayan iki boyutlu uygulamalara yönelik Kartezyen çözüm ağlarında çalışan bir Euler çözücüsü geliştirilmiştir. Standart Kartezyen çözüm ağı prosedürleri gözden geçirilerek değiştirilmiş. Hareket eden sınır problemlerine yönelik tasarlanmış, pratik bir Kartezyen çözücü için gerekli olduğu görülen, dejenere elemanların sistematik takibi sağlanmıştır. Offset eden sınırların, özel uygulamalardan alınmış örnekleri sunulmuştur. Sabit-ağ, hareketli-ağ ve hareket eden katı sınırları için akı hesap yöntemleri, temel kesin bir Riemann çözücülü Godunov metodu açısından ele alınarak derlenmiştir. Geliştirilen metodun sınır hareketli ve karmaşık geometrilere uygunluğu seçilen örneklerle ayrı ayrı gösterilmiştir. Çözüm metodu adım adım geliştirilerek, çok boyutlu uygulamalarda ve son olarak bir katı yakıtlı roket motorunun eksenel iç akış çözümünde kullanılmıştır.

Anahtar Kelimeler: Kartezyen çözüm ağı metodları , Euler çözümleri, Sonlu hacim, Hareketli sınır problemleri, Ofsetleme, Çekirdek geriye yanma, İç akışlar, Katı yakıtlı roket motorları , Eksenel simetrik. Riemann çözücüleri

To Eser and Sevda

ACKNOWLEDGEMENTS

I express my sincere appreciation to my supervisor Prof. Dr. Ahmet ÜÇER for his guidance, suggestions, support and foresight throughout my research. His unique quality rich attitude and understanding has made this thesis meaningful and our meetings dearly memorable.

It is my pleasure to convey my deep gratitude to Prof. Dr. Cahit ERALP, my esteemed tutor in experimentation, and Prof. Dr. Nafiz ALEMDAROĞLU for their crucial advices and short cuts as patient and understanding members of my thesis advisory committee.

I would like to thank, to all members, past and present, of Propulsion Design Group, especially our head Dr. Tugrul TINAZTEPE, Mehmet COŞKUN, Mine E. YUMUŞAK, Dr. Olcay OYMAK, Suzan KOÇ and specially Başar SEÇKİN, and staff at Engineering Development Department of ROKETSAN, during different phases of this thesis.

In the name of my friend Ugras BARAN, my thanks extend to all I have met before, who are mad for fluid dynamics, as they are mad to live.

My special thanks goes to my dive buddies and sailor friends for their approval of my devotion to this thesis. Although for many long years they had to go out to sea without me, they have kindly kept asking my attendance.

Finally, I would like to thank to my family, to whom this thesis is devoted, for their support, understanding, patience and unshakable faith in me. Without them nothing would have been made.

TABLE OF CONTENTS

CHAPTER I

INTRODUCTION

I.1 THE PURPOSE OF THIS STUDY AND COURSE OF WORK

In this thesis, "a general 2-D time-accurate Euler solver for internal flow problems involving complex moving boundaries" is aimed. Cartesian grid approach is chosen mainly because of its versality in the geometric definition when complex geometries are concerned. Cartesian grid methods, unsteady and moving boundary problems will be reviewed in this chapter, which are currently the attractive Computational Fluid Dynamics (CFD) research areas.

Prediction of internal flow in a solid propellant rocket motor is the primary application of this type of study where complex geometries, moving boundaries and transient operation modes coexist. Besides all these features, the surface combustion of the propellant must be modeled sufficiently complex so that the developed solver can be efficiently used in design environment. The solver should also cover axisymmetric geometries, if it is intended to be used as a design tool for solid propellant rocket motors.

The core of this work is a finite volume solver. Fluxes from cell faces are calculated by the Godunov method, which utilizes an exact Riemann solver that is generic in terms of the code structure and its functions. The exact Riemann core solver can easily be replaced by higher order flux calculation schemes (Variations of Total Variation Diminishing (TVD) and Weighted Average Flux (WAF) schemes, Monotone Upstream Schemes for Conservation Laws (MUSCL)-Hancock methods) that are available from the solver database [1], coded in the form of C language classes.

Developing the Riemann solver and utilizing it in the 1D Euler code was the start of the study. This 1D code is extended further to problems with moving boundaries, with arbitrary source terms and solid propellant burning models. Being a transient code, its usage in combustion instability research is demonstrated. Moving boundary handling and injection models are verified and tested with 1D code for several applications. For the 2D extension of this study, complete geometric algorithms are developed for Cartesian grid approach and tested in moving boundary problems that demonstrate the initial objective. Same solid propellant injection models are used in one- and two- dimensional solvers without changing the basic method of approach; an axisymmetric version of the code is developed and applied to a solid propellant motor test case with injecting grain surfaces.

I.2 FINITE VOLUME FORMULATIONS OF CONSERVATION LAWS

Computational fluid dynamics algorithms are based on the differential or integral formulations of conservation laws. These formulations lead to finite difference and finite volume algorithms, respectively. When it is desired to compute flows over

complex configurations and internal flows, as in this thesis, finite volume approach has advantages in mass and geometry conservation and treatment of singular points over the finite difference approach.

The grid definition is different in the two approaches. In finite difference, grid points represent the vertices of the control volume whereas in finite volume the cell center value can be approximated from the vertices, which affects the practical handling of boundary conditions, singularities and zonal procedures. The review article [2] is comparing the two formulations in all aspects. For considerations on grid methods that are employed [3] should be referred.

I.3 CARTESIAN GRID METHODS

For computational gas dynamics applications, involving complex 2D bodies, Cartesian Grid approach is an alternative to unstructured grid solvers. By both methods automation is brought to the fluid flow simulations. Other attractive features of Cartesian grids are, simplified data structures and possibility of obtaining grid shape independent solutions [4]. Recently there is a growing interest in using Cartesian grids for complex geometries. Quirk [5] defines the building blocks and solver related requirements for such an approach. External flows around stationary bodies were considered. Unlike [6] where various in cell boundary approximations are compared, Quirk has kept the exact orientations of the cut portions of solid walls relative to Cartesian control volumes. Cartesian grid generation algorithms are readily applicable to modern hybrid grid methods. Cartesian elements utilized in mixed grids can be found in references [7], [8] and [9]. An up to date study is [10], where quadrilateral grids, defined on the solid boundaries are overlapped with

adaptive Cartesian grids. By this combination it is made possible to use Cartesian grids in a viscous application. Since in viscous boundary layer type flows, the essential directional boundary adaptation, is not probable when Cartesian meshes are used. In parallel, the resulting randomly distributed surface cell areas are not suited well for the Navier-Stokes equations especially in three dimensions. A remedy taken by [11] is to use a coupled Euler/Integral boundary layer formulation. Another interesting feature of [10] which is designed for steady flow applications is the elimination of tiny cell combinations by utilizing systematic local time-stepping methods. However this short cut in grid algorithms is applicable only to steady flows.

When refined solutions of the Euler and Navier-Stokes equations are needed, a Cartesian cell based grid adaptation technique is required. In [12] the viscous and compressible detectors are used for cell sub division. Refined cells are stored in a suitable structured database. Binary tree data structures provide and store cell-to-cell connectivity information in a logical way. Adaptive mesh refinement is also utilized in studies [4], [5] and [13]. Cartesian grid methods can be extended to three dimensions. One practical approach suggested by [6] is to perform Cartesian griding on section planes that are parallel to two Cartesian directions. Similar idea is used in [13] where a multi-grid solution scheme in Cartesian grids is developed, in addition, for complex configurations.

All the studies noted in this survey of Cartesian cell-based approach are for stationary boundaries. To the author's knowledge, the study in this thesis, is the first moving boundary handling attempt in Cartesian grids.

I.4 MOVING BOUNDARY PROBLEMS

Interface movement is the typical feature of moving boundary problems. Across the interfaces flow features, compositions, phases, material properties can vary rapidly. Interfaces move under the influence of flow field and in turn affect the behavior of the flow. Besides the momentum, heat and mass transport, formation, evolution and dynamics of the interface is also crucial. Near-surface thermochemical phenomena is reviewed in [14] from a CFD perspective, with particular emphasis on computational boundary conditions for surface mass and energy transfer.

There are many technological applications where moving bodies play a major role. In fluid machinery there are systems where an externally driven rigid boundary interacts with flow. Aeroelasticity is a typical example to problems where boundary movement is coupled directly to flow properties. In solidification and flame-propagation, the interface is a part of the continuous media and evolves with flow. In liquid-gas systems, the formation and coalescence of droplets and other surface instabilities present challenges in tracking multiple interactive interfaces [15]. Mechanisms of ablation, pyrolysis, melt flow spallation and icing are other areas that involve a moving boundary.

Moving boundary problems can be handled by Fixed Grid, Boundary Conforming [15] or Level Set Methods [16]. Each method has its own advantages. The fixed grid finite volume approach is especially suitable for solvers utilizing Cartesian Grids and favored in this study, since the correct representation of characteristic waves generated by the moving boundaries requires the accurate orientation of moving

boundary relative to the control volume [17]. Wall boundary conditions are applied with respect to the moving boundary. Moreover satisfying an extra geometric conservation law [18],[19] is not required for every cell in the solution domain.

Boundary conforming methods are suitable for highly automated unstructured grids since remeshing is generally required after a certain solution step. When cell faces are allowed to move, numerical flux calculation schemes should take this movement into account. In some cases, complete derivations may be required.

I.5 INTERNAL FLOW MODELLING OF SOLID PROPELLANTS

In this study, moving boundary techniques are applied to internal flow in solid propellant motors where solid propellant surface regression is coupled to flow properties. Complex surface reactions are present and boundary movement is not uniform due to physical processes such as erosion. Moreover, design requirements force complex initial grain geometries.

Prediction of solid propellant motor internal flow field using CFD methods is a challenging task. Main reason is due to the different flow regimes existing in a single problem. Present CFD methods are being developed for a specific flow regime thus the method of approach is considerably different for low speed and high-speed flows. Contrary to the nozzle divergence region, where the flow field is highly supersonic, gas flow behaves like an incompressible fluid inside the motor chamber. Motor chamber flow velocities are very low and pressure level is high resulting Mach numbers in the order of 0.07.

In the literature, the general approach is to solve the motor internal flow field at the choked steady state for fixed internal grain geometry. Vuillot [20][21] presents solutions for stability applications by superposing the disturbances on a solution obtained for steady state. Schley et. al. [22], used an explicit finite volume code to solve 3D, unsteady N/S equations for a liquid motor chamber. This study comments on using compressible schemes in the large subsonic part of the combustion chamber. In [23], Vortex shedding and unsteady flow inside combustion chambers of solid propellant motors are analyzed with 3D Euler and 2D axisymetric N/S equations on unstructured meshes. [24] Developed two computer programs based on the implicit ADI scheme of Beam and Warming and the explicit Runge-Kutta multi stage scheme to simulate unsteady flow fields in solid propellant motor environments, without reaction. Other attempts to solve cold flow inside solid propellant motors are described in [25], [26], [27]. In [28], a simple solid propellant chemical kinetics is included to a 2D N/S formulation and a semi-implicit four stage Runge-Kutta scheme is applied. Another branch of modeling studies in literature focuses on injection induced flows, which is crucial for understanding the details of the internal flow field of solid propellant motor chambers. To be complete, the following references will be given related to this area [29],[30],[31],[32],[33],[34].

Efforts to predict full motor internal solution with the grain boundaries modeled as moving surfaces are relatively rare. To the author's knowledge, there are two recent studies. In [35] with a body-fitting unstructured grid, the performance of a cylindrical grain motor is predicted. Euler equations are solved on a moving grid employing the flux-difference scheme of Roe. Numerical prediction of internal flow details was not presented but a pressure time history of a single motor is compared. An erosive

burning rate model is applied. Complex motor grain geometries are studied by [36], where at each time step the combustion chamber geometry is computed by a separate surface burnback code and triangular mesh is automatically generated. Prediction of combustion instabilities is the aim. Multi-block and hybrid grids are possible and this feature in [36] is used to generate nozzle grid separately using the specialized techniques.

I.6 CONTENTS OF THE THESIS

The contents of this thesis are grouped into five main chapters. The governing equations are stated in Chapter II. Together with Appendix E, a fairly general set of equations are provided. In Chapter III, an exact Riemann solver is developed and tested for problems that are available from literature. Most of the mathematical groundwork of this study, which is required in the development phase of a numerical scheme for conservation equations, is sited in the Appendix F in order to maintain continuity in the main body of the thesis. Chapter IV is devoted to one-dimensional problems. Topics like discretization of the governing equations with moving boundaries, handling moving geometries in Cartesian grids, and solid propellant surface injection models are covered in this chapter, explaining their extensions to multi-dimensional problems. Chapter V is on the Cartesian grid methods, algorithms, and solver. Test cases are selected from real life problems that demonstrate the possibility of moving boundaries in Cartesian grids. In Chapter VI, two-dimensional method of approach is extended to cover axisymmetric geometries. Features of an axisymmetric formulation are completely presented and the numerical experience gained during the implementation of various equation forms are documented. Results obtained for a steady full solid propellant motor test case are discussed.

CHAPTER II

GOVERNING EQUATIONS

Governing equations for reacting, three-dimensional axisymmetric, swirling flows with dispersed solid particles in a non-inertial coordinate system are discussed from a physical point of view and presented in Appendix E. These equations are derived in integral form, from which corresponding differential equations in conservative form follows naturally.

For future possible applications and research studies, Appendix E is intended to be a reference and a starting point for the discretization. For this reason, it covers a wide range of fluid flow phenomena in a fairly extended form. Terms that govern these extensions can be collected under a general source term. The remaining terms are identical to homogeneous Euler equations, which will be presented in the following section. In these form, equations become suitable for numerical methods that are developed for Euler equations with no source terms. Thus, with the same code, without changing the flux calculation procedure, a wide range of phenomena can be studied.

Reactive source terms may require the solution of stiff ordinary differential equations (ODE). Since for stiff ODE, in literature, there exists powerful numerical methods, numerical solution of problems with reactive source terms is clear-cut. Geometric source terms requires special treatments, like in many cases, an extra geometric conservation law have to be solved. Mass generation and axisymmetric source terms are studied in this thesis. These two are respectively reactive and geometric in nature. For this reason, problems encountered with both types are discussed in the related chapters of this work.

II.1 EULER EQUATIONS

Time dependent, three-dimensional Euler equations govern the flow of a compressible fluid for which the effects of body forces, viscous stresses and heat flux are neglected. In terms of the conserved variable vector $\vec{U}$ and flux vectors $\vec{F}(\vec{U}), \vec{G}(\vec{U})$ and $\vec{H}(\vec{U})$ in the three spatial directions with no source terms, they are

$$\vec{U}_t + \vec{F}(\vec{U})_x + \vec{G}(\vec{U})_y + \vec{H}(\vec{U})_z = \vec{0} \tag{2.1}$$

$$\vec{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ E \end{bmatrix}, \vec{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(E+p) \end{bmatrix}, \vec{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(E+p) \end{bmatrix}, \vec{H} = \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(E+p) \end{bmatrix} \tag{2.2}$$

With the total energy per unit volume, given by

$$E = \rho\left(\frac{1}{2}(u^2 + v^2 + w^2) + e\right) \tag{2.3}$$

Specific internal energy e, can be related to the other flow variables using the ideal gas equation of state and the equations are closed via Equation (2.4).

$$e = p/(\rho(\gamma - 1)) \tag{2.4}$$

In the form given in Equation (2.1), the system is homogeneous. When terms that model injection of mass, momentum and energy, body forces, chemical reactions and other phenomena are included as discussed in Appendix E, inhomogeneous form of the equations with the source term symbolized as $\vec{S}(\vec{U})$ is obtained.

$$\vec{U}_t + \vec{F}(\vec{U})_x + \vec{G}(\vec{U})_y + \vec{H}(\vec{U})_z = \vec{S}(\vec{U}) \tag{2.5}$$

The components of the source term $\vec{S}(\vec{U})$ that are relevant and used to model the phenomena covered in this thesis are presented in the next section for a generalized one-dimensional flow.

II.2 GOVERNING EQUATIONS FOR QUASI-ONE-DIMENSIONAL COMPRESSIBLE GENERALIZED FLOWS

Quasi-one-dimensional simulations provide quite detailed information about transient gas dynamics [57]. Moreover, a broad range of applications can be covered with a relatively simple set of equations. Such as [58], where unsteady gas /particle flows, and [59], magneto-gas dynamics with varying are terms are studied. Classic analysis and a complete set of equations are well documented [60], with area change, heat transfer, friction and mass injection. These equations will be presented here by extending them to include a generalized spatial/temporal area change and mass injection terms for solid propellant motors [61]. For unsteady flows with friction, the methodology in [62], which completes the equation set with boundary-layer momentum and mean-flow kinetic energy integral equations, should be preferred instead of using a simple friction factor.

Dropping multi-dimensional, non-inertial, particle-fluid and species interaction, phase change, radiative and conduction terms, and representing viscous terms by a friction factor, Equations (E.1), (E.4), (E.24) and (E.34), in Appendix E, can be reduced to Equation (2.6). With the addition of two extra conservation equations for the velocity components, v and w components of velocity vector, whose purpose is related to the chosen numerical method of extending a one-dimensional solver to multi-dimensions. This numerical method of extension will be discussed in the related sections of Chapter III and IV. The variables v, w are treated as passively advected quantities, similar to the species term q_i, in this formulation.

$$\vec{U}_t + \vec{F}\left(\vec{U}\right)_x = \vec{S}(\vec{U}) \tag{2.6a}$$

where,

$$\vec{U} = \begin{bmatrix} \rho \\ \rho u \\ E \\ \rho v \\ \rho w \\ \rho q_i \end{bmatrix}, \qquad \vec{F} = \begin{bmatrix} \rho u \\ \rho u^2 + P \\ u(E+P) \\ \rho uv \\ \rho uw \\ \rho u q_i \end{bmatrix} \tag{2.6b}$$

In Equation (2.6b), the conserved variable E represents the total energy per unit volume as defined previously in Equation (2.3)

and the source term,

$$\vec{S} = -\frac{1}{A}\begin{bmatrix} A.\dfrac{\delta \rho u}{x} + \rho A_t - \dfrac{dm_i}{dx} \\ Au.\dfrac{\delta \rho u}{x} + \rho u A_t + \dfrac{\rho u^2}{2}\dfrac{4f}{D}A - u_{ix}\dfrac{dm_i}{dx} \\ A.\dfrac{\delta u(E+P)}{x} + E A_t - H_{in}\dfrac{dm_i}{dx} - \dfrac{dQ}{dx} \\ A.\dfrac{\delta \rho uv}{x} + \rho v A_t - u_{iy}\dfrac{dm_i}{dx} \\ A.\dfrac{\delta \rho uw}{x} + \rho w A_t - u_{iz}\dfrac{dm_i}{dx} \\ A.\dfrac{\delta \rho u q_i}{x} + \rho q_i A_t - q_{i(inj)}\dfrac{dm_i}{dx} \end{bmatrix} \tag{2.6c}$$

By Equation (2.6), governing equations are switched to the vector form of representation. Since it will be more straightforward to discuss the mathematical properties and develop computational schemes in this form.

In the general source term (2.6c), m_i is injection mass flow rate. u_{ix} represents the x-component of injection velocity. and f is the Fanning friction factor, which is included to facilitate partial modeling in possible viscous applications.

For solid propellants the rate of mass injection can be represented in the equations via Equation (2.7)

$$dm_i / dx = r_b \rho_p \frac{\partial S}{\partial x} \tag{2.7}$$

In this equation, S is the propellant surface area. ρ_p and r_b are propellant density and burn rate, respectively. For completeness a simple burning rate law can be used as $r_b = a.P^n$. Where a and n are empirical constants of the propellant.

The total enthalpy per unit mass H_{in} in (J/kg), which is the input to the flow by injection is ($H_{in} = h_i + (u_{ix}^2 + u_{iy}^2 + u_{iz}^2)/2$) Also it is assumed that, the mass dm_i, that is added to the main stream flow mixes completely.

In the quasi-one-dimensional flow, in generally used coordinate systems can be revealed by selecting the parameter δ, from the following list.

$\delta = 0$	for cartesian coordinates
$\delta = 1$	for cylindirical coordinates
$\delta = 2$	for spherical coordinates
$\delta = xA_x / A$	for quasi-one-dimensional flow

A_x and A_t are the rate of change of normal area with respect to x-component and time.

CHAPTER III

THE RIEMANN SOLVER

III.1 MULTIDIMENSIONAL EULER EQUATIONS IN X-SPLIT FORM

Engineering applications of fluid mechanics are generally multi-dimensional and the boundaries of the solution domain are not aligned with the rectangular Cartesian coordinates. Transforming the governing equations to curvilinear coordinates and using body-fitted orthogonal grids is a classical approach for handling such problems [51]. The flux and velocity vectors are expressed by their contravariant components [50] and the governing equation system can still be expressed in the form given in Equation (2.1) of Chapter II.

Another approach to the problem is possible which is called dimensional splitting in CFD literature [63]. Along with the discussion of this topic, no change will be made in the notation of velocity components; in a general curvilinear system, u, v, w now representing contravariant velocity components, where, v and w are the velocity components perpendicular to u. For any orthogonal control volume in this system that is spanned by the unitary basis vectors, one-dimensional Euler equations are solved for each of the three orthogonal directions successively. In these solutions,

u represents the primary solution direction and other velocity components are treated as simple advected quantities. These directions, where 1D solutions are performed will be termed as "x-split" directions.

A flux-based approach of the method is more practical, if a one-dimensional flux stencil, i.e. a coded numerical scheme to calculate flux vector of conservation laws, is available. By the flux-based dimensional splitting approach, the flux vector through each arbitrarily oriented cell face can be calculated, using the contravariant velocity components in the flux stencil. The calculated fluxes, after they are back transformed to the Cartesian direction can be used in the control volume balance equation.

In this thesis, the latter method is practiced where an exact Riemann solver is utilized to calculate one-dimensional fluxes. This Riemann solver is the topic of this Chapter.

The solution of x-split Riemann initial value problem is needed which is stated in Equation (3.1). In order to cover splitting in three dimensions and allow for species terms, variables v, w and q_i also take part as advected quantities along the splitting direction. In Equation (3.1), $\vec{U}(x,0)$ defines the initial conditions of the problem which is a discontinuity at $x=0$, with the given left and right states $\vec{U}_L$ and $\vec{U}_R$.

$$\left.\begin{aligned} \vec{U}_t + \vec{F}(\vec{U})_x &= \vec{0} \\ \vec{U}(x,0) = \vec{U}^{(0)}(x) &= \begin{cases} \vec{U}_L & if \quad x<0 \\ \vec{U}_R & if \quad x>0 \end{cases} \end{aligned}\right\} \qquad (3.1)$$

where

$$\vec{U} = \begin{bmatrix} \rho \\ \rho u \\ E \\ \rho v \\ \rho w \end{bmatrix}, \quad \vec{F}(\vec{U}) = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ u(E+p) \\ \rho uv \\ \rho uw \end{bmatrix}$$

The structure of the time dependent solution is sketched in Figure III-1, on the x-t plane. Four distinct states are developed from the given initial left and right states. The boundaries of these states are defined by the characteristic waves in the x-t plane. $(u-a)$ and $(u+a)$ characteristic fields are genuinely nonlinear and associated with rarefactions or shock waves. Across these waves, regardless of their type, the tangential velocity components v and w remain constant. The remaining three characteristic fields that are associated with three coincident characteristic speeds (u,u,u) are two shear waves where v and w change discontinuously, and a contact wave with a jump in density [39]. Across the middle wave (star region) both normal particle velocity u and pressure remains constant.

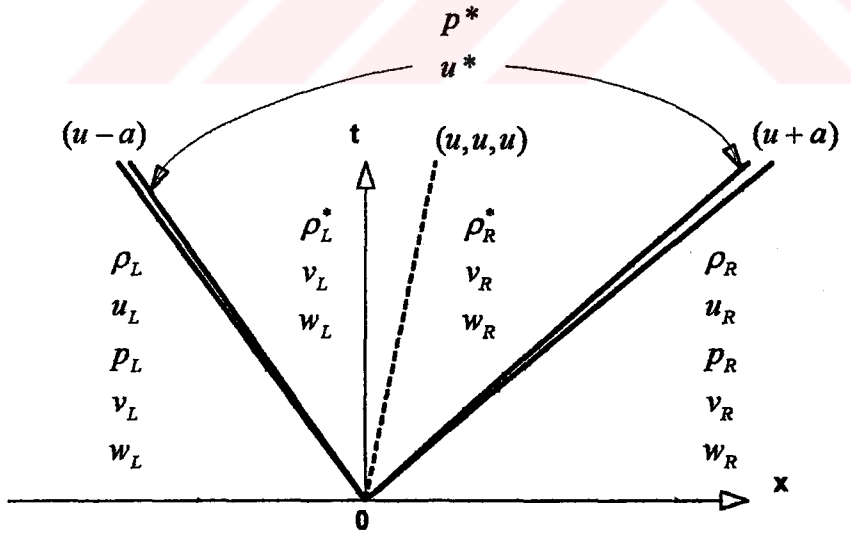


Figure III-1: Solution structure of the three dimensional x-split Riemann problem.

Equation (3.1) is a one-dimensional nonlinear hyperbolic conservation law, with properties that are presented in Appendix F, Section F.2.

III.2 EXACT RIEMANN SOLVER FOR A GENERAL X-SPLIT EULER EQUATION

The Riemann problem is the generalization (for arbitrary left and right states) of the classical shock-tube problem in Gas Dynamics [43]. Starting with Godunov [40], its exact solution formed the foundation of the current upwind methods. An approximate solution may be utilized to reduce the computation time in some numerical methods. Especially when the problems are steady and very fine solution details are not desired. A detailed error analysis to gain confidence in the approximate solver is needed in that case [52]. Moreover, incomplete modeling of the tangential velocity components in the x-split form can be undesirable and approximate solvers are not preferred for Navier-Stokes equations [39]. The Riemann problem solution, exact or approximate, is used locally in the method of Godunov and high-order extensions of it. Such as the higher order schemes like WAF [54] and MUSCL [53]. Additionally, when testing/verifying the numerical methods, analytical solutions are generally obtained using the exact Riemann solutions. Other popular numerical methods that use Riemann solution are Random Choice Methods (RCM), Roe's and Osher's Riemann Solvers, HLL and HLLC solvers.

Since there is no closed solution to the Riemann problem, an iterative scheme should be developed. The equation of state influences the mathematical character of the equations. The following formulation for the exact Riemann solver that is developed

in this study, is based on the procedure given in [39] assumes calorically perfect ideal gases. For gases with covolume equation of state where molecular volume is taken in to account is generally studied in literature, in [42] with an RCM type scheme and in [52] with a two-rarefaction approximate Riemann solver.

Since v and p are constant across the contact discontinuity, it is often easier to work with primitive variables $\vec{W}$ rather than conserved variables $\vec{U}$ [38]. $\vec{W}$ is given below in Equation (3.2).

$$\vec{W} = \begin{bmatrix} \rho \\ u \\ p \\ v \\ w \end{bmatrix} \tag{3.2}$$

Exact solution of the Riemann problem is a two-step process. First pressure and u-velocity at the star region is determined, Figure III-1. The calculation of states separated by the contact wave is the final step, where for shock waves, only density and shock speed, for rarefaction waves full solution inside the rarefaction fan is needed.

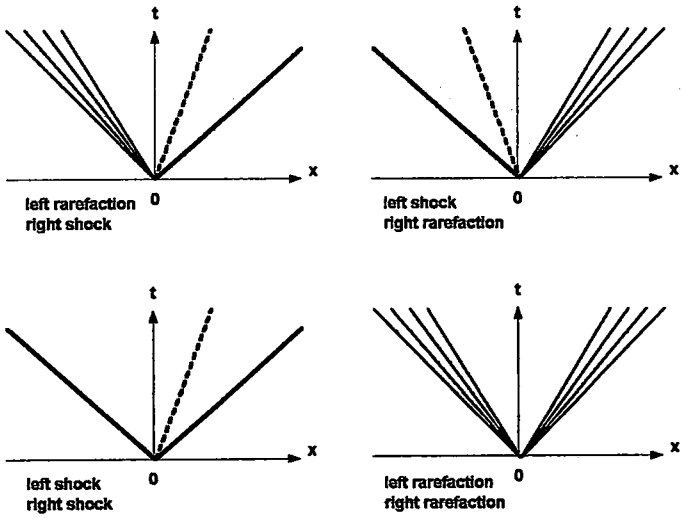


Figure III-2: Possible Wave Patterns.

III.2.1 SOLUTION FOR u^* AND p^*

For each possible wave pattern plotted in Figure III-2, the left or right data state will be related to the unknown u^*. In these relations, terms, that depend on the other unknown p^*, will be identified with special functions; $f_L(p^*,\vec{W}_L)$ and $f_R(p^*,\vec{W}_R)$.

If left wave is a shock wave, function f_L will be derived as follows. With pre-shock values ρ_L, u_L, p_L and post-shock values $\rho_L{}^*$, u^*, p^* and with the relative velocities given below

$$\hat{u}_L = u_L - S_L,\ \hat{u}^* = u^* - S_L \tag{3.3}$$

Mass, momentum and energy across the discontinuity are as follows;

$$\rho_L \hat{u}_L = \rho_L{}^* \hat{u}^* \tag{3.4}$$

$$\rho_L \hat{u}_L^2 + p_L = \rho_L{}^* \hat{u}_L{}^{*2} + p^* \tag{3.5}$$

$$\hat{u}_L(\hat{E}_L + p_L) = \hat{u}^*(\hat{E}_L{}^* + p^*) \tag{3.6}$$

Using mass and momentum equations and defining mass flux Q_L, the relation for u^* is obtained as

$$u^* = u_L - \frac{(p^* - p_L)}{Q_L} \tag{3.7}$$

From momentum relation (3.5) and the definition of mass flux in Equation (3.8),

$$\hat{u}_L = \frac{Q_L}{\rho_L},\ \hat{u}^* = \frac{Q_L}{\rho^*{}_L} \tag{3.8}$$

Q_L as a function of p^* and $\rho^*{}_L$ is obtained as

$$Q_L^2 = -\frac{p^* - p_L}{\frac{1}{\rho^*{}_L} - \frac{1}{\rho_L}} \tag{3.9}$$

Using the energy relation (3.6) and caloric equation of state the density behind the shock is related to p^* by Equation (3.10)

$$\rho^*_L = \rho_L \left[\frac{\left(\frac{\gamma-1}{\gamma+1}\right)+\left(\frac{p^*}{p_L}\right)}{\left(\frac{\gamma-1}{\gamma+1}\right)\cdot\left(\frac{p^*}{p_L}\right)+1} \right] \tag{3.10}$$

Substituting (3.9) and (3.10) into (3.7) produces the desired relation

$$u^* = u_L - f_L(p^*, \vec{W}_L) \tag{3.11}$$

where,

$$f_L(p^*, \vec{W}_L) = (p^* - p_L)\left[\frac{A_L}{p^* + B_L}\right]^{1/2} \tag{3.12}$$

with,

$$A_L = \frac{2}{(\gamma+1)\rho_L},\ B_L = \frac{(\gamma-1)}{(\gamma+1)}p_L \tag{3.13}$$

If the left wave is rarefaction, then a different f_L is derived using the isentropic relation and the generalized Riemann invariants for the left wave.

The isentropic relation;

$$\rho^*_L = \rho_L\left(\frac{p^*}{p_L}\right)^{\frac{1}{\gamma}} \tag{3.14}$$

is substituted to the definition of speed of sound leading

$$a^*_L = a_L\left(\frac{p^*}{p_L}\right)^{\frac{\gamma-1}{2\gamma}} \tag{3.15}$$

Across the left rarefaction the Generalized Riemann Invariant is constant

$$u_L + \frac{2a_L}{\gamma-1} = u^* + \frac{2a^*_L}{\gamma-1} \tag{3.16}$$

Combining (3.15) and (3.16) results the required relation (3.17).

$$u^* = u_L - f_L(p^*, \vec{W}_L) \tag{3.17}$$

with

$$f_L(p^*, \vec{W}_L) = \frac{2a_L}{(\gamma - 1)}\left[\left(\frac{p^*}{p_L}\right)^{\frac{\gamma-1}{2\gamma}} - 1\right] \tag{3.18}$$

The function $f_R(p^*, \vec{W}_R)$ for the right wave is derived similarly. The complete derivation can be found in [39] and [41].

For a right shock wave

$$u^* = u_R + f_R(p^*, \vec{W}_R) \tag{3.19}$$

with,

$$f_R(p^*, \vec{W}_R) = (p^* - p_R)\left[\frac{A_R}{p^* + B_R}\right]^{1/2} \tag{3.20}$$

And for a right rarefaction wave

$$f_R(p^*, \vec{W}_R) = \frac{2a_R}{(\gamma - 1)}\left[\left(\frac{p^*}{p_R}\right)^{\frac{\gamma-1}{2\gamma}} - 1\right] \tag{3.21}$$

Eliminating u^* from (3.12), (3.18), (3.20) and (3.21), Equation (3.22), for p^*, which covers all possible wave structures are obtained.

$$f(p^*, \vec{W}_L, \vec{W}_R) = f_L(p^*, \vec{W}_L) + f_R(p^*, \vec{W}_R) + \Delta u = 0 \tag{3.22}$$

The type of wave, whether it is shock or rarefaction, is determined by comparing magnitudes of p^* with the related right or left states, p_R and p_L. For example if the right wave is a shock then $(p^* > p_R)$.

After p^* is determined from (3.22), u* can be obtained from (3.11), (3.17), (3.19) depending on the wave type in the problem.

With the given initial states and $\Delta u = u_R - u_L$, a vacuum state is possible in the solution of the Riemann problem. Vacuum is generated by the non-linear waves if the pressure positivity condition (3.23) is violated [39].

$$(\Delta u)_{crit} \equiv \frac{2a_L}{\gamma - 1} + \frac{2a_R}{\gamma - 1} > u_R - u_L \tag{3.23}$$

III.2.2 NUMERICAL SOLUTION FOR p^*

An iterative solution is needed to calculate p^* from Equation (3.22). Given left and right states the pressure function (3.22) is concave down and monotone increasing [39]. Then the Newton-Raphson iteration scheme, given in Equation (3.24) can be safely employed in which,

$$p^*_{(k)} = p^*_{(k-1)} - \frac{f(p^*_{(k-1)})}{f'(p^*_{(k-1)})} \tag{3.24}$$

Where k is the iteration level. It is important to have the initial guess for the p^* value to be good to increase the rate of convergence. There are some approximations for p^* that can be good starting points. One such approximation is so called two-rarefaction approximation, which is given as

$$p_{TR} = \left[\frac{a_L + a_R - 1/2(\gamma - 1)(u_R - u_L)}{a_L / p_L^{\frac{\gamma-1}{2}} + a_R / p_R^{\frac{\gamma-1}{2}}} \right] \tag{3.25}$$

Another possible way for fixing the initial value for the iteration, which can be computed by Equation (3.26), is found from the linearized solution based on primitive variables.

$$\left.\begin{aligned} p_0 &= \max\left(CHECK\ , p_{PV}\right) \\ p_{PV} &= 1/2(p_L + p_R) - 1/8(u_R - u_L)(\rho_R - \rho_L)(a_R - a_L) \end{aligned}\right\} \tag{3.26}$$

A third guess value alternative is given by two-shock approximation as:

$$\left.\begin{aligned} p_0 &= \max(CHECK, p_{TS}) \\ p_{TS} &= \frac{g_L(\hat{p})p_L + g_R(\hat{p})p_R + \Delta u}{g_L(\hat{p}) + g_R(\hat{p})} \\ g_K(\hat{p}) &= \left(\frac{A_K}{p + B_K}\right) \end{aligned}\right\} \tag{3.27}$$

In Equation (3.27), A_K and B_K has been defined previously in (3.13), $\hat{p}$ is an initial estimate for the solution. Taking $\hat{p} = p_{TR}$ works well. *CHECK* is a non-negative value chosen to prevent negative pressures that may be produced by the formulas.

A fourth guess is the arithmetic mean of the left and right data:

$$p_0 = \frac{1}{2}(p_L + p_R) \tag{3.28}$$

There are studies comparing the success of these estimates [39]. A hybrid method is used in the present code based on [42].

III.2.3 COMPLETE SOLUTION IN THE STAR REGION

Left and Right density values in the star region is still unknown. More over for shock waves the shock speed, for rarefaction waves head and tail wave speeds must be found. There are four possibilities as given in Figure III-2.

For a left shock wave, with condition $p^* > p_L$, the density is found from Equation (3.10). The shock speed given by Equation (3.29) follows from the relations in (3.3).

$$S_L = u_L - a_L\left[\frac{\gamma+1}{2\gamma}\frac{p^*}{p_L}+\frac{\gamma-1}{2\gamma}\right]^{\frac{1}{2}} \tag{3.29}$$

For left rarefaction wave, with condition $p^* \le p_L$, density follows from Equation (3.14). The head and tail characteristic speeds of the wave are obtained found from Equation (3.30) [43]

$$S_{HL} = u_L - a_L,\ S_{TL} = u^* - a_L^* \tag{3.30}$$

For a characteristic starting from origin (0, 0) and a general point (x, t) inside the rarefaction fan, primitive variables inside the fan can be calculated from Equation (3.31).

$$\vec{W}_{Lfan} = \begin{bmatrix} \rho = \rho_L\left[\frac{2}{(\gamma+1)}+\frac{(\gamma-1)}{(\gamma+1)a_L}(u_L-\frac{x}{t})\right]^{\frac{2}{\gamma-1}} \\ u = \frac{2}{(\gamma+1)}\left[a_L+\frac{(\gamma-1)}{2}u_L+\frac{x}{t}\right] \\ p = p_L\left[\frac{2}{(\gamma+1)}+\frac{(\gamma-1)}{(\gamma+1)a_L}(u_L-\frac{x}{t})\right]^{\frac{2\gamma}{\gamma-1}} \end{bmatrix} \tag{3.31}$$

For a right shock wave, with condition $p^* > p_R$, the density is found from,

$$\rho^*_R = \rho_R \left[\frac{\left(\frac{\gamma-1}{\gamma+1}\right)+\left(\frac{p^*}{p_R}\right)}{\left(\frac{\gamma-1}{\gamma+1}\right)\cdot\left(\frac{p^*}{p_R}\right)+1} \right] \tag{3.32}$$

The shock speed is computed by Equation (3.33).

$$S_R = u_R + a_R \left[\frac{\gamma+1}{2\gamma}\frac{p^*}{p_R} + \frac{\gamma-1}{2\gamma} \right]^{\frac{1}{2}} \tag{3.33}$$

For right rarefaction wave, with condition $p^* \le p_R$, density

$$\rho_R{}^* = \rho_R \left(\frac{p^*}{p_R} \right)^{\frac{1}{\gamma}} \tag{3.34}$$

The speeds of head and tail characteristics and solution inside the right rarefaction fan can be calculated from the Equations (3.35) and (3.36) respectively.

$$S_{HR} = u_R + a_R, \; S_{TL} = u^* + a_R{}^* \tag{3.35}$$

$$\vec{W}_{Rfan} = \begin{bmatrix} \rho = \rho_R \left[\frac{2}{(\gamma+1)} - \frac{(\gamma-1)}{(\gamma+1)a_R}(u_R - \frac{x}{t}) \right]^{\frac{2}{\gamma-1}} \\ u = \frac{2}{(\gamma+1)} \left[-a_R + \frac{(\gamma-1)}{2}u_R + \frac{x}{t} \right] \\ p = p_R \left[\frac{2}{(\gamma+1)} - \frac{(\gamma-1)}{(\gamma+1)a_R}(u_R - \frac{x}{t}) \right]^{\frac{2\gamma}{\gamma-1}} \end{bmatrix} \tag{3.36}$$

For each passively advected quantity $q = v, w, q_i, \ldots$ an extra conservation law can be derived;

$$(\rho q)_t + (u\rho q)_x + (v\rho q)_y + (w\rho q)_z = 0 \tag{3.37}$$

The corresponding solution of the one-dimensional Riemann problem is given as

$$q(x,t) = \begin{cases} q_L & if \ \frac{x}{t} < u^* \\ q_R & if \ \frac{x}{t} > u^* \end{cases} \tag{3.38}$$

The remaining variables (v,w) of the x-split Riemann problem (1), behave similar to a passively advected quantity q_i.

For combustion problems, q_i represents the species that take part in the chemical reactions. In that case, the extra source terms will then appear in the equations as presented in detail in Appendix E.

III.2.4 EXACT RIEMANN CODE AND SOLUTION SAMPLING

The solution to a general Riemann problem is self-similar. By observing the Equations (3.31), (3.36), (3.38), solution at a general point (x,t) depends the speed $S = x/t$. Comparing this speed S with the speed of existing waves, appropriate equations can be selected and used. These equations are plotted schematically in Figure III-3 for the left side of the contact wave.

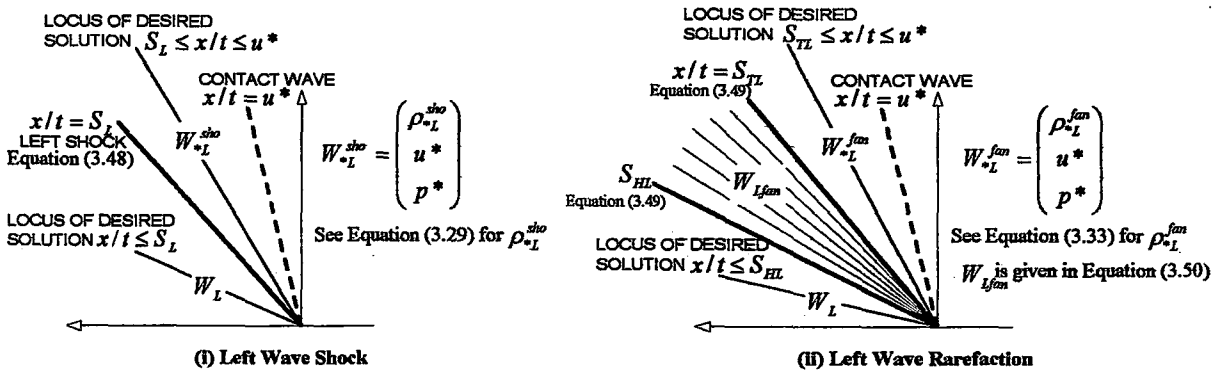


Figure III-3: Solution sampling for the left side of the contact wave.

If the desired solution falls to the right side of the contact wave i.e. $S = x/t \ge u^*$, and if the right wave is shock with speed S_R then the complete solution is given in Equation (3.39).

$$W(x,t) = \begin{cases} W_{*R}^{sho} & if\ u^* \le x/t \le S_R \\ W_R & if\ x/t \ge S_R \end{cases} \tag{3.39}$$

In this equation, $W_{*R}^{sho} = \begin{pmatrix} \rho_{*R}^{sho} \\ u^* \\ p^* \end{pmatrix}$ where ρ_{*R}^{sho} is calculated by Equation (3.32).

If the right wave is rarefaction with speeds given in Equation (3.35), then the complete solution will be sampled according to Equation (3.40).

$$W(x,t) = \begin{cases} W_{*R}^{fan} & if\ u^* \le x/t \le S_{TR} \\ W_{Rfan} & if\ S_{TR} \le x/t \le S_{HR} \\ W_R & if\ x/t \le S_{HR} \end{cases} \tag{3.40}$$

In the above sampling procedure the solutions, for v, w and q are not presented. These three variables are treated as simple advected quantities in the context of splitting and their solution is given in (3.38) previously.

The first output code of this study is the exact Riemann solver. The source code is given with the program name "`riemann`". Given the initial left and right states and

the desired solution point (x,t), this program calculates the state vector $\vec{W}$, in primitive variables. The solution is calculated for a general x-split state vector, with 6 components, together with a single chemical species term. This code is verified using the test cases that are presented in the next chapter.

Subroutines of this code are used exactly as building blocks in the preceding computer programs that are developed for one- and two- dimensional applications of gas dynamics.

III.3 TEST CASES

To test the performance of the Riemann solver four initial value problems are selected from the literature. These test cases are generally used to test numerical schemes in one-dimension. However in this study, their exact solutions will be developed. These results are obtained using the code `riemann` with 40 grid points. The position x varies from –0.5 to 0.5 m. After the solution in the star region is obtained, at each grid point x/t is calculated for the desired solution time and the appropriate equation set is used. Specific heat ratio is γ=1.4 in all test cases.

For Test Cases, which are summarized below, results obtained using the Riemann code are in exact accordance with the corresponding solutions available from the literature. When over plotted the curves become coincident in the Figures given below.

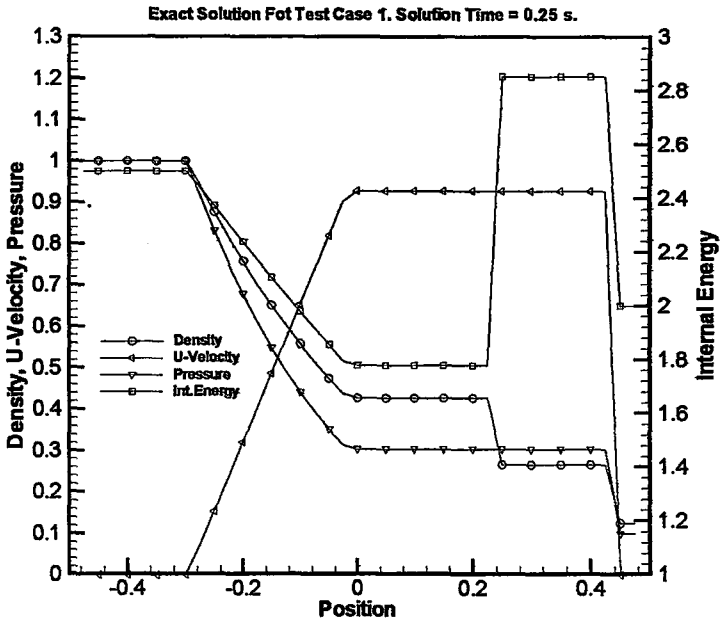


Figure III-4: Exact Solution for Sod's Test Problem. Test Case 1.

The first problem is Sod's Test Problem [55], Solution consists of left rarefaction, a contact and a right shock. The solution at t=0.25 s which is presented in Figure III-4 is found using the exact Riemann solver subroutine, with the initial conditions listed in Table III-1.

Table III-1: Initial conditions for Test Case 1.

	LEFT STATE	**RIGHT STATE**
Density (kg/m^3)	1.0	0.125
U-Velocity (m/s)	0.0	0.0
Pressure (Pa)	1.0	0.1

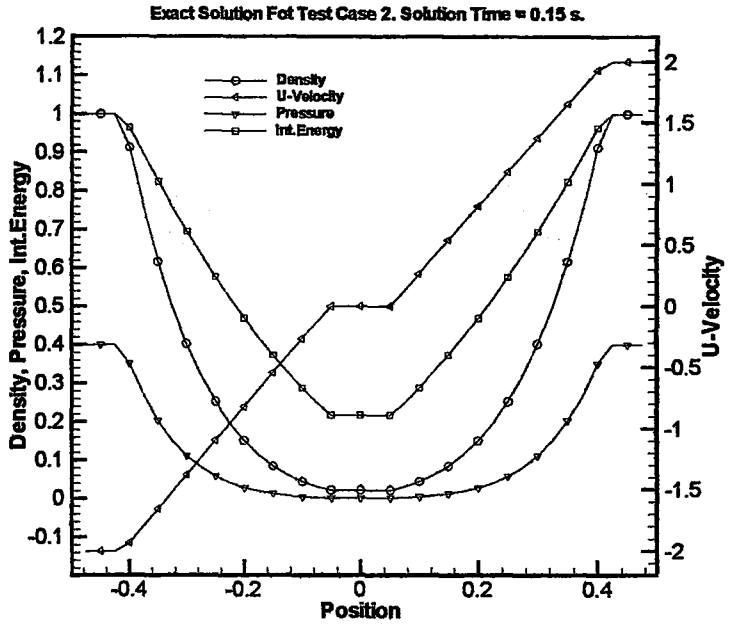


Figure III-5: Exact solution for Test Case 2.

Test 2 is referred as 123 Problem [56] in literature. Its useful in assessing the performance of numerical methods for low density flows [39]. Solution consists of two strong rarefactions and a stationary contact discontinuity. The solution at t=0.15 s is presented in Figure III-5 with the initial conditions of Table III-2.

Table III-2: Initial conditions for Test Case 2.

	LEFT STATE	**RIGHT STATE**
Density (kg/m^3)	1.0	1.0
U-Velocity (m/s)	-2.0	2.0
Pressure (Pa)	0.4	0.4

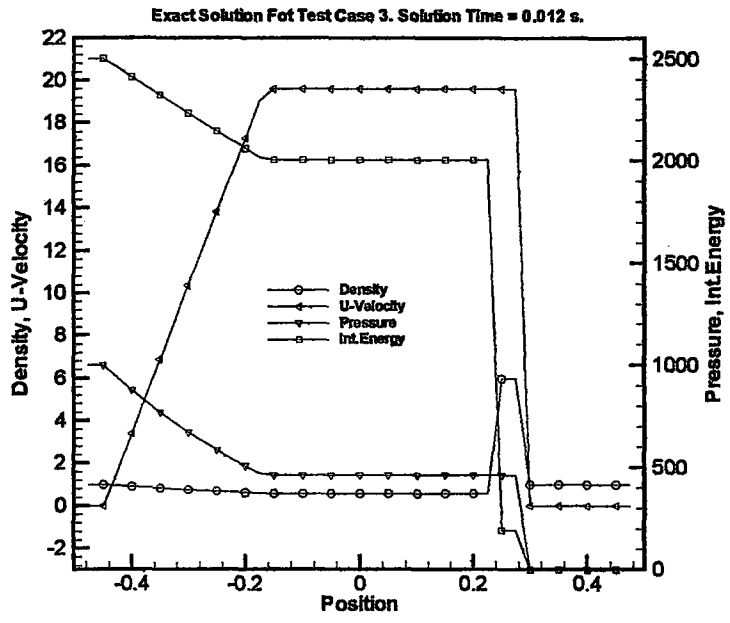


Figure III-6: Blast-wave problem. Exact solution for Test Case 3.

In Test 3 a one-dimensional blast-wave is generated due to the high initial pressure level. A left rarefaction, a contact discontinuity and a right shock is developed. Solution at t=0.012 s. is presented in Figure III-6. The initial conditions for this Riemann problem are presented in Table III-3.

Table III-3: Initial conditions for Test Case 3.

	LEFT STATE	**RIGHT STATE**
Density (kg/m^3)	1.0	1.0
U-Velocity (m/s)	0.0	0.0
Pressure (Pa)	1000.	0.01

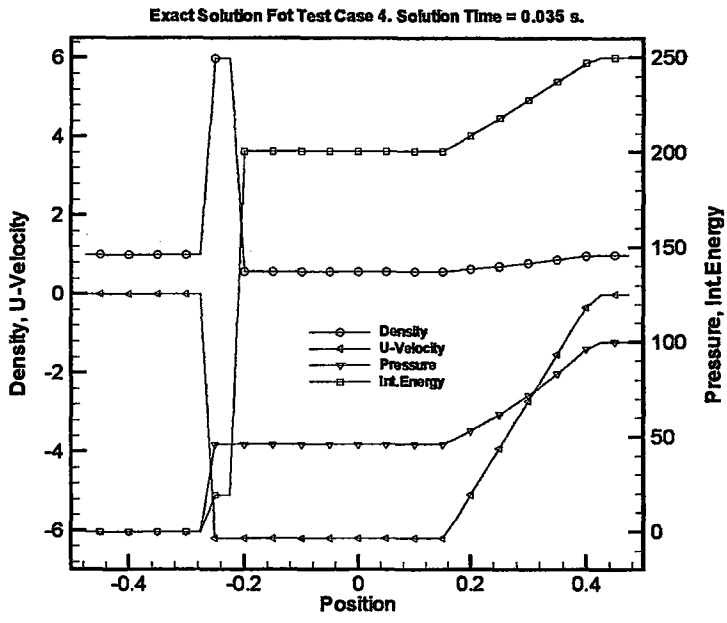


Figure III-7: Exact Solution for Test Case 4.

Test 4 is the reverse of Test Case 3 with a slightly lower pressure ratio. A left shock, a contact discontinuity and a right rarefaction is developed. Solution at t=0.035 s. is presented in Figure III-7. The initial conditions for this Riemann problem are presented in Table III-4 below.

Table III-4: Initial conditions for Test Case 4.

	LEFT STATE	**RIGHT STATE**
Density (kg/m^3)	1.0	1.0
U-Velocity (m/s)	0.0	0.0
Pressure (Pa)	0.01	100.

CHAPTER IV

ONE-DIMENSIONAL FORMULATION

IV.1 MOTIVATION

Unsteady gas dynamics and prediction of unstable burning in solid propellant test motors that are used in combustion instability research is the basic motivation. A full transient solution with injecting and moving propellant boundaries is needed. Apart from this motivation, a one-dimensional study is essential before extending a numerical technique to higher dimensions. Likewise, in this thesis, the proposed retarding surface model is first verified using the developed transient moving boundary one-dimensional code. Also for design purposes, one-dimensional codes are practical in terms of both time and accuracy.

IV.2 INTRODUCTION

In this chapter, after the general solution procedure is stated, the building blocks of the 1D transient, moving boundary Euler code will be presented. Although the applications are selected from the field of propulsion, depending on the problem at hand, any type of source term can be implemented. The core part of this study is the Riemann solver that is developed by the methods discussed in Chapter III.

IV.3 SOLVER

IV.3.1 GEOMETRY

A one-dimensional solution domain is plotted in Figure IV-1 for two consecutive times t and t+Δt. The solution domain models a pipe with moving left and right boundaries. In Figure IV-1, left is an advancing boundary while right is retarding.

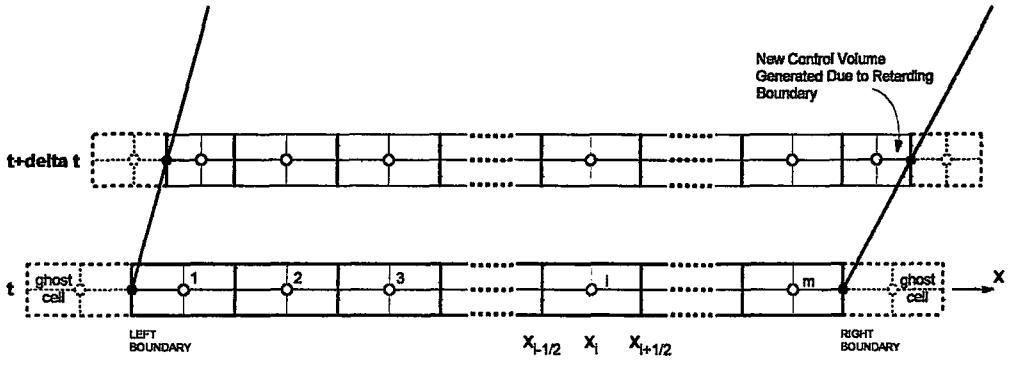


Figure IV-1: Solution domain with moving ends for two consecutive time steps.

The spatial domain [0,L] is discritised into m finite volumes. Except at the moving boundaries the control volume size is regular; $\Delta x = x_{i+1/2} - x_{i-1/2} = L/m$. The location of cell center and cell boundaries are given by

$$x_{i-1/2} = (i-1)\Delta x,\ x_i = (i-1/2)\Delta x,\ x_{i+1/2} = i\Delta x \qquad (4.1)$$

IV.3.2 GOVERNING EQUATIONS

Equation (2.6) of Chapter II that is derived for a general one-dimensional flow is simplified to govern only mass injection, spatial and temporal area variations. By replacing the notation used for mass injection per unit volume, dm_i/dx, by S_p, the source term with reduced terms is repeated below, in Equation (4.2).

$$\vec{S} = -\frac{1}{A}\begin{bmatrix} A_x.\rho u + \rho A_t - S_P \\ A_x.u.\rho u + \rho u A_t - u_{ix} S_P \\ A_x.\, u(E+P) + E A_t - H_{in} S_P \\ A_x.\rho uv + \rho v A_t - u_{iy} S_P \\ A_x.\rho uw + \rho w A_t - u_{iz} S_P \\ A_x.\rho u q_i + \rho q_i A_t - q_{i(inj)} S_P \end{bmatrix} \tag{4.2}$$

IV.3.3 SOURCE TERM SPLITTING

Unlike Equation (2.6) of Chapter II, the Riemann solver in Chapter III is developed for homogeneous Euler Equations, in the form that is given by Equation (4.3)

$$\vec{U}_t + \vec{F}\left(\vec{U}\right)_x = 0 \tag{4.3}$$

In this study, the source terms will be treated by the method of splitting. Apart from upwinding of source terms, the general approach in numerical studies is the source term splitting method [39]. Splitting methods are also used to manage multidimensional and diffusive type source terms. Although different variations of the algorithm can be implemented, the most basic form is utilized.

For the solution at time t+Δt, the homogeneous partial differential equation (4.3) is solved with the given initial conditions, at the first round. The initial condition is the state vector $\vec{U}^n$ at time t. This solution produces the intermediate solution $\overline{\vec{U}}^{n+1}$. This intermediate solution $\overline{\vec{U}}^{n+1}$ will then be the initial condition for the ordinary differential equation

$$\frac{d}{dt}\vec{U} = \vec{S}(\vec{U}) \tag{4.4}$$

Equation (4.4) can be then solved via explicit, stiff-explicit or implicit methods depending on the stability limit of the source term. In this study, an explicit forth order Runge-Kutta method is used to obtain the desired solution $\vec{U}^{n+1}$. In operator notation this method can be symbolized by Equation (4.5).

$$\vec{U}^{n+1} = S^{(\Delta t)} C^{(\Delta t)} (\vec{U}^n) \tag{4.5}$$

Other variations are available in the literature. A second order accurate version is

$$\vec{U}^{n+1} = S^{(\frac{1}{2}\Delta t)} C^{(\Delta t)} S^{(\frac{1}{2}\Delta t)} (\vec{U}^n) \tag{4.6}$$

IV.3.4 GODUNOV SCHEME [40]

Given the initial data $\tilde{\vec{U}}(x,t^n)$ (which may be continuous), to evolve the solution to a new time t+Δt, a piecewise constant distribution of the data over each control volume is needed. The piece-wise constant distribution is obtained by defining cell averages

$$\vec{U}_i^n = \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} \tilde{\vec{U}}(x,t^n)\, dx \tag{4.7}$$

At each inter-cell boundary, the given initial data is now discontinuous, defined by the control volume averaged state vectors of the left and right cells. These local discontinuities form different Riemann Problems (RP) whose solution can be evaluated by the methods of Chapter III.

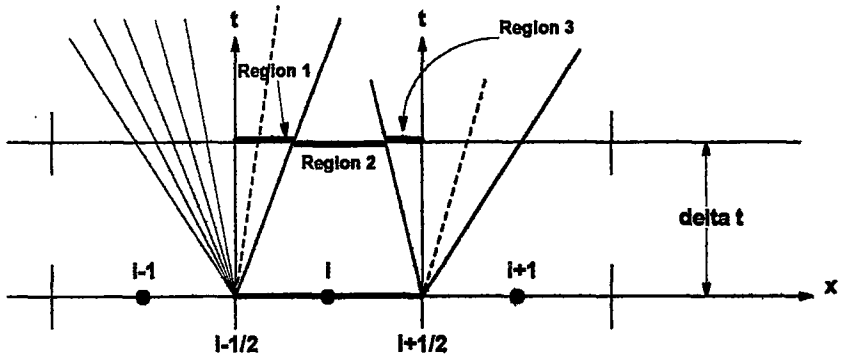


Figure IV-2: Wave patterns of local Riemann problems at the cell boundaries and regions affected.

For the i^{th} control volume the solutions of two local Riemann problems $RP(\vec{U}_{i-1}^n, \vec{U}_i^n)$ and $RP(\vec{U}_i^n, \vec{U}_{i+1}^n)$ will be used to calculate the state vector at new time t+Δt. Due to these discontinuities, wave patterns emerge from the cell boundaries. See Figure IV-2. The solution at new time t+Δt of i^{th} cell consists of three parts that are valid over three regions of i^{th} cell. Region 1, is determined by the Riemann Problem $RP(\vec{U}_{i-1}^n, \vec{U}_i^n)$, Region 2 is by $RP(\vec{U}_i^n, \vec{U}_{i+1}^n)$. Since emerging waves has not yet propagated into Region 3, its state vector is same as the initial data $\vec{U}_i^n$. By integrating the two Riemann problem solutions $\vec{U}_{i-\frac{1}{2}}(\frac{x}{\Delta t})$ and $\vec{U}_{i+\frac{1}{2}}(\frac{x}{\Delta t})$ over the control volume, solution at new time t+Δt of i^{th} cell is obtained in (4.8)

$$\vec{U}_i^{n+1} = \frac{1}{\Delta x}\int_0^{\frac{1}{2}\Delta x} \vec{U}_{i-\frac{1}{2}}(\frac{x}{\Delta t})dx + \frac{1}{\Delta x}\int_{-\frac{1}{2}\Delta x}^{0} \vec{U}_{i+\frac{1}{2}}(\frac{x}{\Delta t})dx \tag{4.8}$$

These Riemann problem solutions are correct if there is no wave interaction within the control volume, which can be controlled by limiting the size of time increment Δt.

$$\Delta t \le \frac{1/2\Delta x}{S_{\max}^n} \tag{4.9}$$

A simple estimate for maximum wave speed is used

$$S_{\max}^n = \max\{u_i^n + a_i^n\} \tag{4.10}$$

where a_i^n is the speed of sound of i^{th} cell.

For practical implementation of the Godunov scheme, it can be shown that in conservative form [39]

$$\vec{U}_i^{n+1} = \vec{U}_i^n + \frac{\Delta t}{\Delta x}\left[\vec{F}_{i-1/2} - \vec{F}_{i+1/2}\right] \tag{4.11}$$

with intercell numerical flux given by

$$\vec{F}_{i+1/2} = \vec{F}(\vec{U}_{i+\frac{1}{2}}(0)) \tag{4.12}$$

Required that the time step Δt satisfies the condition

$$\Delta t \leq \frac{\Delta x}{S_{\max}^n} \tag{4.13}$$

Introducing the CFL condition (0<CFL≤1), (4.8) can be rewritten as

$$\Delta t = \frac{CFL\ \Delta x}{S_{\max}^n} \tag{4.14}$$

IV.3.5 BOUNDARY CONDITIONS

The boundary conditions are specified by assigning appropriate state vectors to the fictitious cells, which are assumed to exist at the left and right ends. See Figure IV-1. Four types of boundary conditions will be discussed. These are inflow/outflow, transmissive, reflective and moving wall boundary conditions, from [64] [41].

Transmissive boundary condition is equivalent to a zeroth order extrapolation. All fictitious flow variables are assigned boundary cell values as given in Equation(4.15)

$$\begin{aligned} \rho_{m+1} &= \rho_m \\ u_{m+1} &= u_m \\ P_{m+1} &= P_m \\ v_{m+1} &= v_m \\ w_{m+1} &= w_m \\ q_{i_{m+1}} &= q_{i_m} \end{aligned} \tag{4.15}$$

Similiarly a reflective boundary condition, along the axial direction, is given in Equation (4.16) as,

$$\begin{aligned} \rho_{m+1} &= \rho_m \\ u_{m+1} &= -u_m \\ P_{m+1} &= P_m \\ v_{m+1} &= v_m \\ w_{m+1} &= w_m \\ q_{i_{m+1}} &= q_{i_m} \end{aligned} \tag{4.16}$$

The moving wall boundary condition is derived with respect to the observer moving with the quasi-steady boundary velocity. For the right end it is given below.

$$\begin{aligned} \rho_{m+1} &= \rho_m \\ u_{m+1} &= -u_m + 2 \cdot (u_{wall}) \\ P_{m+1} &= P_m \\ v_{m+1} &= v_m \\ w_{m+1} &= w_m \\ q_{i_{m+1}} &= q_{i_m} \end{aligned} \tag{4.17}$$

The exact solution of the Riemann problem containing a moving boundary is symmetric about the path of the moving wall and consists of either two shocks or two rarefactions with the contact wave coinciding with the moving wall [39]. See Figure IV-3.

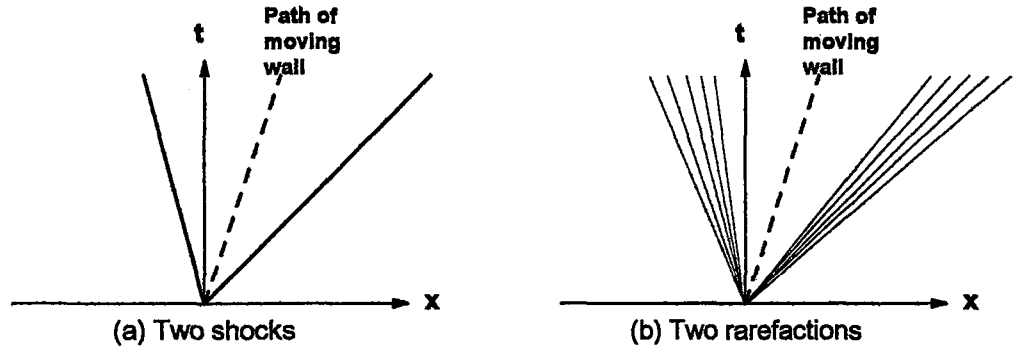


Figure IV-3: Wave patterns generated by the moving boundary. (a) Left-end advancing (b) Right-end retarding

Since the boundaries have no mass, Equation (4.17) is valid, not only for a constant boundary velocity, but also for boundaries moving with an arbitrary acceleration.

For supersonic outflow, all of the three fictitious flow variables are extrapolated from the solution domain. For subsonic outflow, the pressure is the only variable that is specified. For subsonic inflow, atmospheric pressure and density are specified for fictitious cell states. Whereas for supersonic inflow, all variables are specified from outside. Single neighbor cell states are used in the extrapolation routine.

IV.3.6 CONSERVATION LAWS IN A DEFORMING CONTROL VOLUME

The discussion in this chapter is applicable also to multi dimensions. For simplicity, related illustrations are made for a one-dimensional geometry. For the physical meaning of x-splitting, material presented in Section III.1 should be reviewed.

The homogeneous governing Equation (4.3) can be written in the most basic integral form as given below,

$$\frac{D}{Dt}\int_{\forall}\vec{U}\,d\forall = \frac{d}{dt}\int_{\forall}\vec{U}\,d\forall + \int_{S}\vec{U}(\vec{V}-\vec{V}_S)\cdot\hat{n}\,dS \qquad (4.18)$$

The material derivative in (4.18) can be combined with the flux terms, leading to Equation (4.19).

$$\frac{d}{dt}\int_{\forall}\vec{U}\,d\forall + \int_{S}\mathrm{H}^{M}\cdot\hat{n}\,dS = 0 \qquad (4.19)$$

where $\mathrm{H}^{M} = (\vec{F}^{M}, \vec{G}^{M}, \vec{H}^{M})$ is the tensor of fluxes. In (4.19) letter "M" stands for "moving" and implies that flux terms are compatible with the one in Equation (4.18), i.e. with $\int_{S}\vec{U}(\vec{V}-\vec{V}_S)\cdot\hat{n}\,dS$.

In moving grid finite volume solvers, these flux terms, $(\vec{F}^M, \vec{G}^M, \vec{H}^M)$, is computed using the same flux stencils for fixed grids by taking normal velocity vector relative to the moving interface [7]. In these equations, $\vec{V}_S$ is the cell boundary velocity.

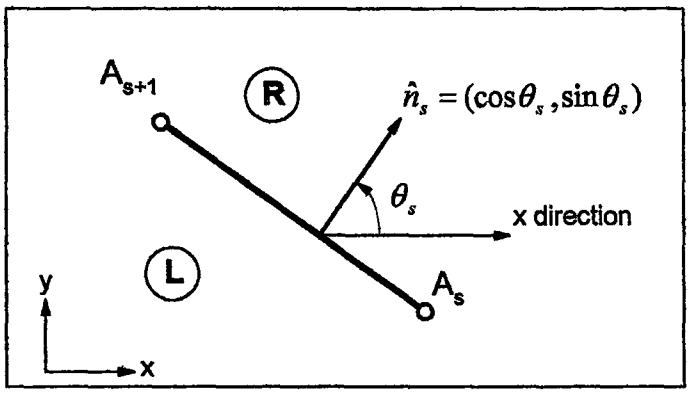


Figure IV-4: Geometric conventions of intercell boundary of a finite volume cell.

Refering to Figure IV-4, for two-dimensional intercell boundaries, the second term in (4.19) can be written as

$$\int_S \mathrm{H}^M \cdot \hat{n}\, dS = \sum_{N\ sides} \int_{A_s}^{A_{s+1}} \mathrm{H}^M \cdot \hat{n}_s\, dA = \sum_{N\ sides} \int_{A_s}^{A_{s+1}} \left[\cos\theta_s \vec{F}^M(\vec{U}) + \sin\theta_s \vec{G}^M(\vec{U})\right] dA \qquad (4.20)$$

For one-dimensional x-split form last term of (4.20) can be written in a compact form with a single flux term

$$\sum_{N\ sides} \int_{A_s}^{A_{s+1}} \left[\cos\theta_s \vec{F}^M(\vec{U}) + \sin\theta_s \vec{G}^M(\vec{U})\right] dA = \sum_{N\ sides} \vec{F}^M(\vec{U}) \qquad (4.21)$$

The first integral in (4.18) assuming averaged properties in the control volume can be taken as

$$\frac{d}{dt} \int_{\forall} \vec{U}\, d\forall = \frac{d}{dt}\left(\vec{U}|\forall|\right) \qquad (4.22)$$

From (4.18), (4.20), (4.21) the properties vector $\vec{U}$ is advanced to the next time step by discritization of the total derivative as follows.

$$\vec{U}^{n+1}\forall^{n+1} - \vec{U}^{n}\forall^{n} = -\Delta t \sum_{N\,sides} \vec{F}^{M}(\vec{U}) \tag{4.23}$$

Where $\vec{F}^{M}$ is any selected flux stencil for moving cell sides. The "stationary control volume" i.e. constant volume version of equation (4.23) is

$$\vec{U}^{n+1} = \vec{U}^{n} - \frac{\Delta t}{\forall^{n}} \sum_{N\,sides} \vec{F}(\vec{U}) \tag{4.24}$$

Since $\vec{U}^{n}\forall^{n}$ needs to be approximated for cells that are initially solid and becoming flow cells in Cartesian grids with moving boundaries, the balance equation (4.23), which in literature, is generally used with adaptively moving meshes, becomes impractical.

In the present method, Cartesian sides of the control volumes are stationary. Only moving sides are the arbitrarily oriented sides of moving boundaries. Besides if the boundary movement is restricted to pure translating motions, it will appear as a contact surface (Figure IV-3) in the x-split direction. Then the same control volume balance equation, both for fixed and moving grids can be used following the method shown below.

The total derivative on the left-hand side of (4.18), by Leibnitz rule can be expressed in two terms;

$$\frac{d}{dt}\int_{\forall} \vec{U}\, d\forall = \int_{\forall} \frac{\partial \vec{U}}{\partial t} d\forall + \int_{S} \vec{U}\vec{V}_{s} \cdot \hat{n}\, dS \tag{4.25}$$

The first change term on the right-hand side of (4.25), which is the local part, provides the solution for constant cell volume. The state vector for this undeformed

control volume can be calculated via the basic balance equation (4.24) by the procedure given above. Where fluxes for the moving cell boundaries are calculated using the left and right states without any transformation. Standard flux stencils mentioned in III.1 are used.

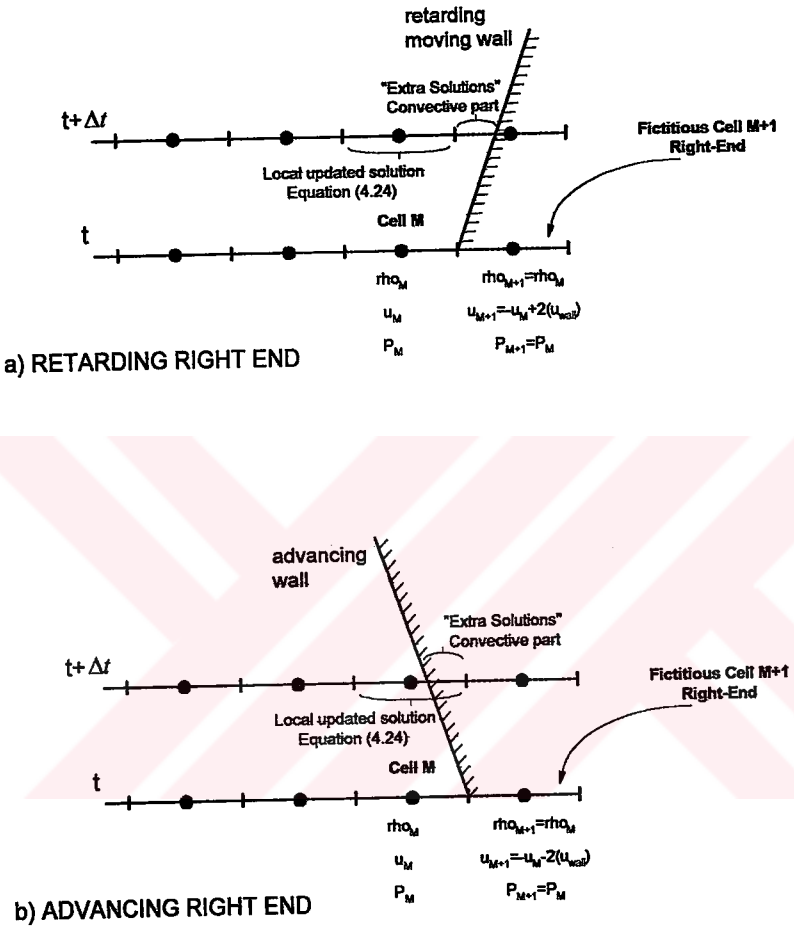


Figure IV-5: Solution update with advancing/retarding wall motion.

The second change term in (4.18), when multiplied by the time step, corresponds to the solution in the deformed part. It is calculated exactly by averaging discrete Riemann solutions over the retarding region. The initial left and right states are again obtained from the wall boundary conditions. In this study, these solutions will be defined as "extra solutions"

This approach is sketched in Figure IV-5. Local and extra solutions are combined by area averaging and new cell center is calculated. Depending on the size at time t+Δt of the advancing/retarding cell a new control volume may be added to the solution domain. These boundary cell size adjustments will be illustrated in the next section.

IV.3.7 MOVING BOUNDARY HANDLING (CELL SIZE ADJUSTMENTS)

As the walls of boundary cells (first and last cell of the solution domain) move, their size increases or decreases from the nominal cell size. Since fixed grids are used and to be compatible with boundary handling procedures in 2D Cartesian code, boundary cell size adjustments are performed in the following way (Figure IV-6):

- For retarding walls if the boundary cell size exceeds 1.5 times the nominal cell size, two new control volumes are defined one with the nominal cell size. The other cell is formed by the remaining part.
- For advancing walls when the boundary cell size is smaller than the half of the nominal cell size, it is combined to its neighbor cell by area averaging of the state vectors.

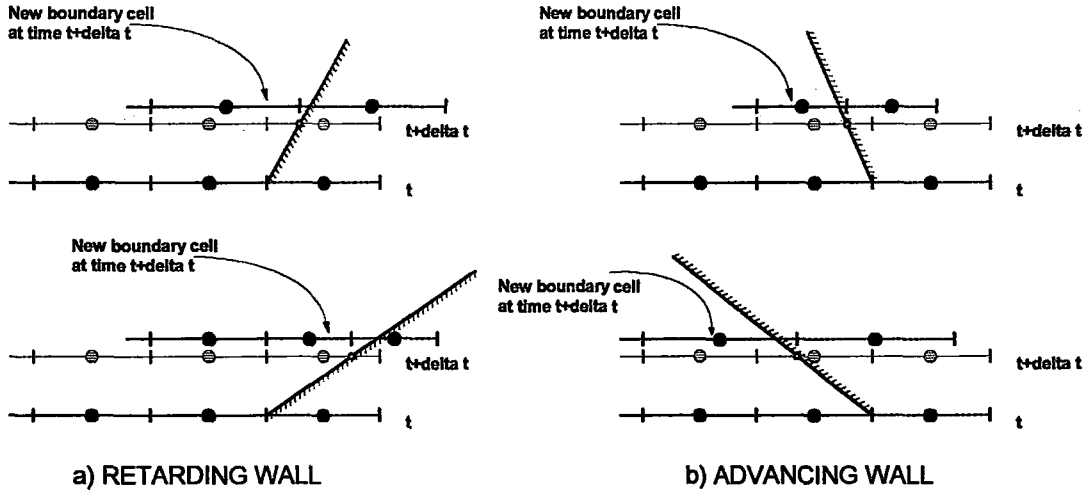


Figure IV-6: Boundary cell sizes for different wall velocities

IV.4 INJECTING INTERFACES AND SOURCE TERM MODELLING

IV.4.1 THE DISPLACEMENT OF INTERFACES

Accurate modeling of a reacting/regressing solid interface requires a detailed study of the kinematics of a moving solid-gas interface. Due to physical and chemical processes that take place at the interface, boundary movement will be observed. The nomenclature for velocities with respect to different observers is as follows:

r_b: Displacement velocity of the condensed phase with reference to the absolute frame of reference.

v'_w: Gas phase velocity at the boundary with respect to the interface

v_{inj}: Gas phase velocity at the boundary with respect to the absolute frame of reference.

Density of gas phase and condense phase at the interface will be denoted by ρ_{inj} and ρ_P, respectively. Mass conservation at the interface gives Equation (4.26).

$$\rho_P r_b = \rho_{inj} v'_w = \rho_{inj}(v_{inj} + r_b) \tag{4.26}$$

In most cases, the solid density is much greater than the gas density and the following approximations can be made $\rho_P >> \rho_{inj}$, $r_b << v_{inj}$.

Then Equation (4.27) is obtained.

$$\rho_P r_b \approx \rho_{inj}(v_{inj}) \tag{4.27}$$

Under this assumption, the so-called "quasi-stationary" treatment can be used, as it is the general strategy in solid propellant motor internal flow studies [65] [66]. That is, coordinate fixed to the moving interface can be treated as stationary, coinciding with the absolute frame

of reference. However in some cases, such as under very high pressure, when ρ_{inj} and ρ_P are of the same magnitude, the quasi-stationary treatment will be inappropriate.

IV.4.2 A SOURCE TERM MODEL FOR SOLID PROPELLANTS

The source term given in Equation (4.2) requires the specification of the following terms related to the energetic phenomena occurring on the propellant surface.

Mass injected per unit volume, S_P

Injection velocity components, u_{ix}, u_{iy} and u_{iz}

Total enthalpy per unit mass of the injected gas, H_{in} (J/kg)

To specify these terms, a model is required. The combustion on the propellant surface will be assumed to occur very fast and confined to a zero thick region. After the combustion, the propellant gas at its flame or stagnation state is expanded to the injection state. These flame properties are found using thermochemical codes [48] [49].

Burning rate r_b is a function of pressure given by Equation (4.28) in the simplest form.

$$r_b = a.P^n \tag{4.28}$$

with a and n are empirical constants of the propellant. This burning rate equation can be modified with extra terms to account for instable burning, erosion [46] and transient burning. There may also be spatial variations in the burning rate due to manufacturing processes. This topic is more severe in end burning motors, which results non-uniform burning patterns in the grain [67]. The corresponding models are available from literature [35] [36] and may be replaced by Equation (4.28).

The injection velocity, since it is in general subsonic, is assumed to be controlled by the local internal motor pressure. For an expansion process from the stagnation to the injection state, the injection velocity can be calculated from Equation (4.27). The only unknown in this equation is the density of injecting gas and can be solved from the energy equation as,

$$\rho_{inj} = \frac{\left(\frac{kP}{k-1}\right) + \sqrt{\left(\frac{kP}{k-1}\right)^2 + 2C_P T_f (\rho_P r_b)^2}}{2C_P T_f} \tag{4.29}$$

IV.5 APPLICATIONS

Some test cases that are used during the verification of 1D solver will be presented in the order of growing complexity. The test cases are gradually developed so that at the end of the study 1D transient prediction of Euler flow in a solid propellant rocket motor with moving boundaries has become possible.

IV.5.1 1D SHOCK TUBE PROBLEM

Time evolution of the initial discontinuity will be predicted for a classical shock tube problem. Numerical results, obtained using the quasi-one-dimensional solver are compared with the corresponding exact solution and with the Test Case 1 of Reference [68].

Exact solution, which is used to compare the numerical results for this test case is obtained using the Riemann solver code, which is developed in this thesis, discussed in Chapter III with 40 grid points and in [68] the exact solution is computed by the method of characteristics.

Transmissive boundary conditions are used at both ends where pipe length is 1 meter and the initial conditions are presented in Table IV-1.

Table IV-1: Initial conditions for 1D test case I.

	LEFT STATE	RIGHT STATE
P (Pa)	5	1
ρ (kg/m^3)	5	1
U (m/s)	0	0

Number of grids in x-direction is 200, with the CFL number used is 0.1 throughout the solution and the solution period is from 0 to 0.5 sec. In Figure IV-7 and IV-8, density and velocities are plotted for selected time steps. In Figure IV-7 at time=0.242044 s, the corresponding exact solution is also plotted for comparison. The steady velocity behind the shock wave is calculated as 0.6800 m/s with the present method. The corresponding speed is given as 0.6862 m/s in [68]. No difference is detected in the plotting resolution of Figure IV-8, the small difference may be due to the numerical dissipation of the present solver.

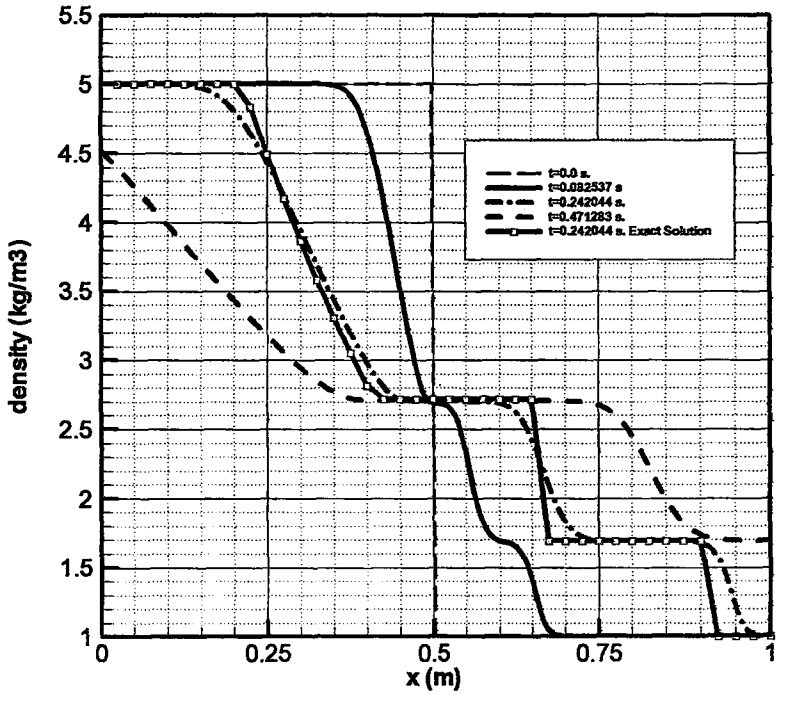


Figure IV-7: Density values for test case 1

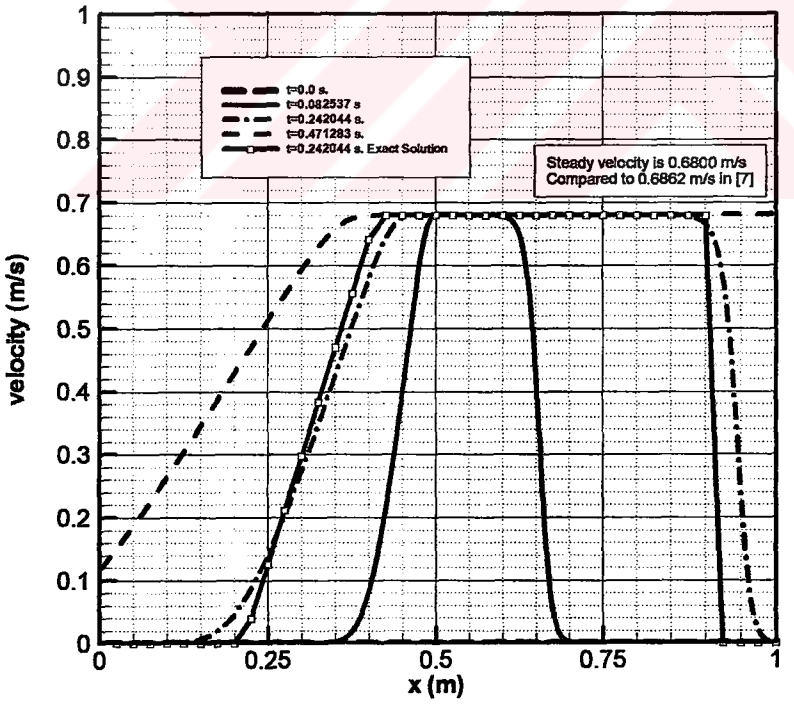


Figure IV-8: Velocity values for test case 1.

IV.5.2 FLOW IN A PIPE WITH RETARDING/ADVANCING WALLS

The evolution of waves generated by a moving piston in a one-dimensional pipe is going to be studied. For an impulsively started piston, which is moving with constant velocity, analytical solution is available [69]. Solutions for arbitrary piston speeds can be obtained by the method of characteristics.

Moving wall boundary conditions specified at left and right ends of the one-dimensional solution domain and wall boundary clipping routine is checked using the following test cases.

IV.5.2.1 CENTERED EXPANSION WAVE

As the piston at the left pipe end is withdrawn, an expansion wave starts to propagate. It continuously flattens as time passes. See Figure IV-9. The front of the wave moves with the speed of sound of the undisturbed fluid in the direction opposite to the piston and fluid motion. The subscript 3 and 4 will denote the states at the head of the piston and of the undisturbed fluid, respectively. Piston velocity, u_P is 4m/s.

Before the piston movement the initial state in the pipe is,

$$\begin{bmatrix} \rho_4 \\ u_4 \\ P_4 \end{bmatrix} = \begin{bmatrix} 12\,kg/m^3 \\ 0.0\,m/s \\ 1013000\,Pa \end{bmatrix} \tag{4.30}$$

The pressure and density ratios can be calculated using the isentropic relations. In Table IV-2, these ratios are compared with the analytically obtained values [69], which shows a successful agreement.

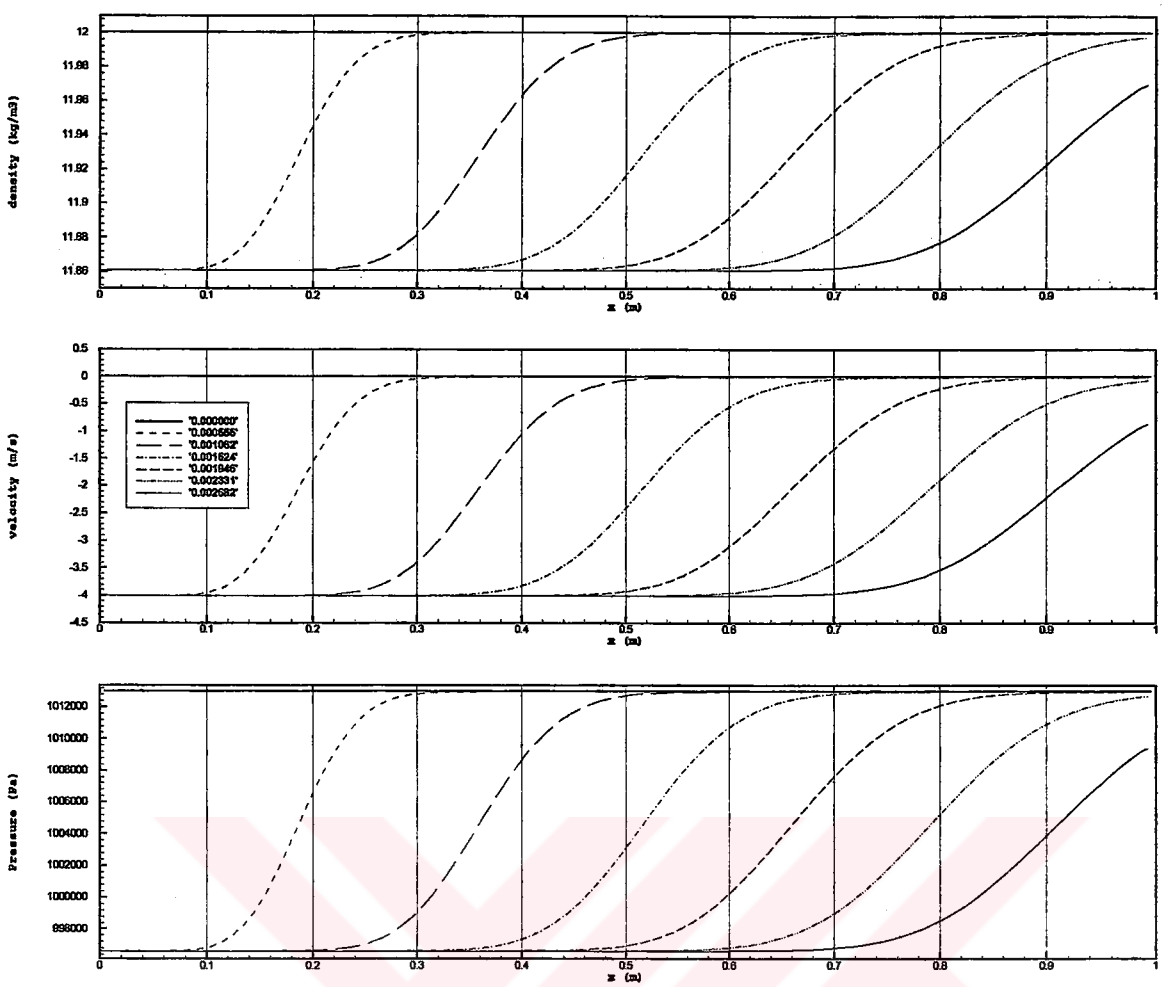


Figure IV-9: Centered expansion wave at consecutive times. Wave is generated by a retarding piston at left pipe end. Piston speed is 4m/s.

Table IV-2: Comparision of pressure and density ratios for a retarding wall.

Ratio	Analytical	Computed
P_3 / P_4	0.9838247	0.9838223
ρ_3 / ρ_4	0.9884193	0.9884167

IV.5.2.2 PROPAGATING SHOCK WAVE

A shock wave is generated by a piston advancing in a pipe. Similar to IV.5.2.1 the piston is impulsively started at t=0 s. Initial conditions are same with the values given in (4.30). The states at piston head and of the undisturbed fluid will be denoted by subscripts 1 and 2 respectively. Piston velocity u_p is 4m/s.

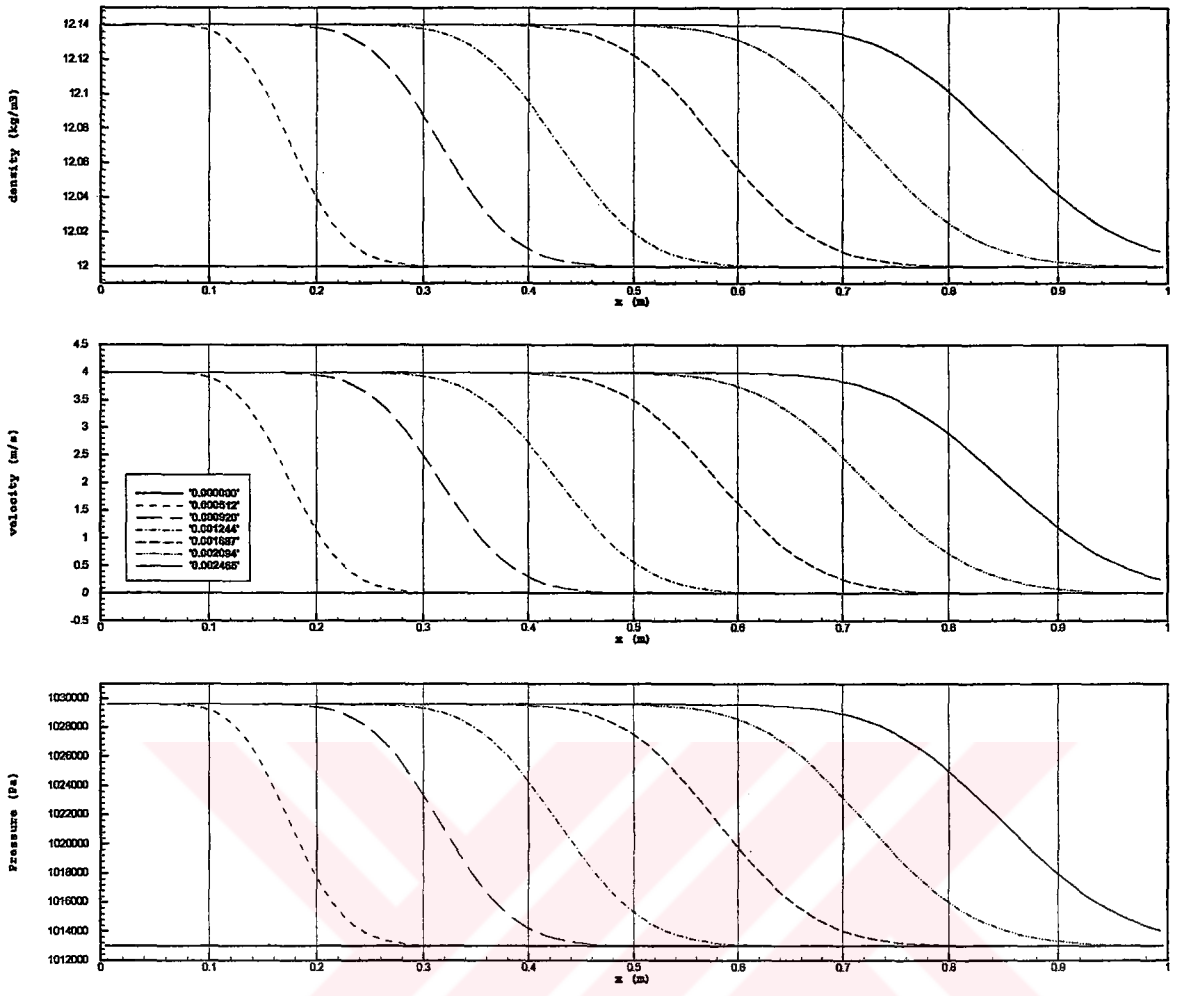


Figure IV-10: Propagating shock wave at consecutive times. Wave is generated by an advancing piston at left pipe end. Piston speed is 4m/s.

Table IV-3: Comparision of pressure and density ratios for an advancing wall.

Ratio	Analytical	Computed
P_2 / P_1	1.0164026	1.0164067
ρ_2 / ρ_1	1.0116888	1.0116833

The pressure and density ratio in Table IV-3 is obtained from 1D gas dynamic equations for a given suddenly accelerated piston. It is worthwhile to note that the pressure and density ratios predicted well while the spatial form of the shock is

smeared over an appreciable distance. This is because of using low number of grids in x-direction (50 grids) and very low cfl number (0.02). The shock wave is generated at the first node, and since at each time step its effect is averaged over the control volume, its sharpness disappears, unlike the real situation where its development takes an infinitesimal time.

IV.5.2.3 PRESSURE INCREASE IN A PIPE DUE TO COMPRESSING WALLS

Transient flow in a closed pipe developed as its two boundaries approach each other will be presented here. The pipe volume constantly decreases and pressure and density level increases due to this advancing movement.

Since the size of the solution domain is constantly decreasing the boundary clipping features of the one-dimensional code is demonstrated with this problem. With fixed grids there was no re-meshing needed at each time step. This reduces the spatial accuracy, as the walls approach each other, because the remaining grids are relatively coarser for the decreased pipe length. At the start of the problem there are 100 grids. For the solution at time=0.38899 s in Figure IV-11 there are approximately 30 grids. Pipe length is 1 meter and both left and right wall which are accelerated suddenly, approaches each other with 1 m/s.

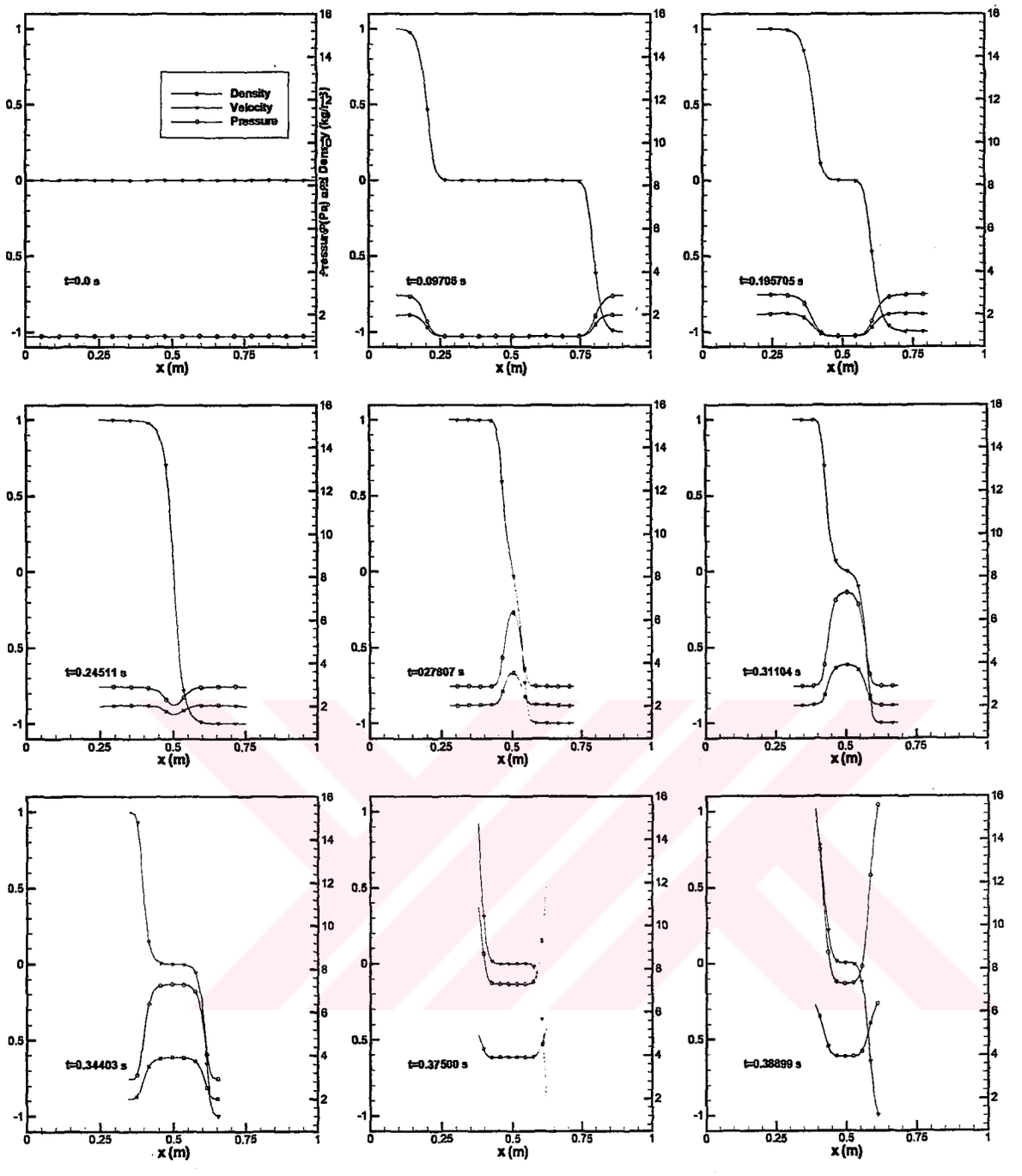


Figure IV-11: Pressure build-up in a pipe with advancing walls.

Pressure increase is observed gradually, as the waves generated by the walls are reflected. In Figure IV-11, this process is plotted in sequence, starting from the upper left plot, where pressure and density are at their initial values and there is zero particle velocity everywhere. When the generated shock waves from both boundaries

met at the middle of the closed pipe, they reflect each other and resulting a step increase in the pipe pressure. When the reflected waves reaches back to the moving pipe walls, another reflection occurs, which increases pipe pressure further. Wave reflections at the wall boundaries are always smooth and no oscillations are observed. This is associated with the numerical method used and one-dimensionality of the problem. No extra precautions need to be taken except specifying the appropriate states for fictitious cells.

IV.5.3 DISCHARGE FROM AN OPEN END PIPE

Another step towards the full transient motor prediction is the study of transients in a relatively simple test case. A natural selection is the discharge from an open end pipe to atmospheric pressure. This problem is chosen because of its geometric simplicity and its close relation with fundamental solid propellant combustion instability studies that are discussed in the next section. Moreover, the same transient inflow/outflow boundary condition scheme is used exactly in full motor test cases that are presented in sections IV.5.5 and IV.5.6.

IV.5.3.1 HELMOLTZ RESONATOR TYPE TEST-SET UPS FOR COMBUSTION INSTABILITY RESEARCH

A brief survey will be presented to reveal an application area of the one-dimensional constituents of this study. To measure the unsteady component of the propellant burning rate, small test motors with simple geometry, usually having an end burning grain configuration is being used. Since with these devices propellant burning rate transients cannot be measured directly, non-intrusive methods are becoming popular in current research and replacing them gradually [70] [44].

A propellant in cylindrical or end burning grain configuration located at the two far ends of the pipe defines a basic T-burner configuration. Oscillations occur at the natural acoustic frequency of the T-burner, which is a function of the burner length and speed of sound, which depends on the temperature of the combustion gases. Various configurations of this device exist and it can be used effectively for frequencies up to 300 Hz [71]. A rotating valve, located at the nozzle throat can also generate disturbances in the motor chamber. This is known as rotating throat valve experiment and can be used down to 100 Hz [72]. Helmoltz resonator burners are devices for very low frequencies, down to few Hertz. Propellant grain in the form of a disk is located at the end of a primary chamber with known volume. This chamber is connected by a pipe to a surge chamber where the pressure is regulated at constant the set value. By changing chamber values different frequencies can be experimented [73].

In these devices, effective frequency is limited by the physical length of the set up. For example, for an experiment at 300Hz a T-burner length of 2m is required. (Speed of sound=1200m/s, f=300 Hz) Fortunately for the numerical experiments, there is no limitation on geometry.

IV.5.3.2 DISCHARGE FROM AN OPEN END PIPE (TRANSIENTS)

The capability of the one-dimensional code is tested in a full transient problem. Some numerical experiments are performed to explore the flow details. By way of this study a confidence about the behavior of the code in similar type of problems is gained. A preliminary frequency analysis is performed which is necessary if the code is intended for combustion instability predictions.

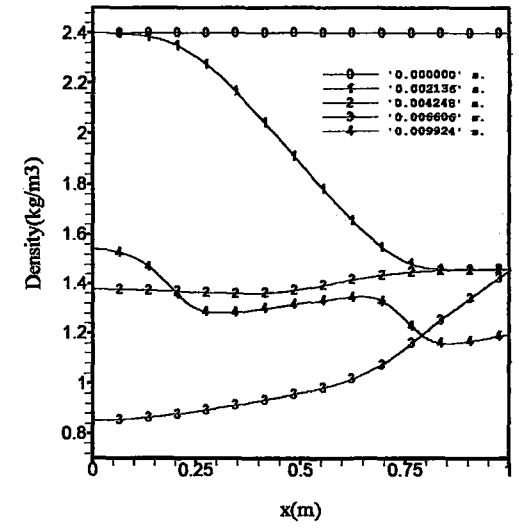
In this investigation, the pipe length is 1 m. This length is divided into 100 grids. Left end of the pipe is closed, which is modeled by the reflective boundary condition. Right pipe end is open to atmospheric pressure, with density and pressure is taken as 1.2 kg/m^3 and 101300 Pa respectively. At this open end, inflow and outflow boundary conditions given in section IV.3.5 are specified. These boundary conditions also represent discharge to a surge tank. Working fluid is air with a specific heat ratio gamma 1.4. Initially the surge tank is separated from the pipe by a diaphragm and is pressurized to several values above the surge tank pressure. Due to an isothermal pressurization process density in the pipe is increased in the same ratio as pressure.

Table IV-4: Pressure dependence of exit flow behavior for the six test runs.

Run	Pressure Ratio	Exit Choked
1	2	No
2	4	Yes
3	6	Yes
4	8	Yes
5	10	Yes
6	12	Yes

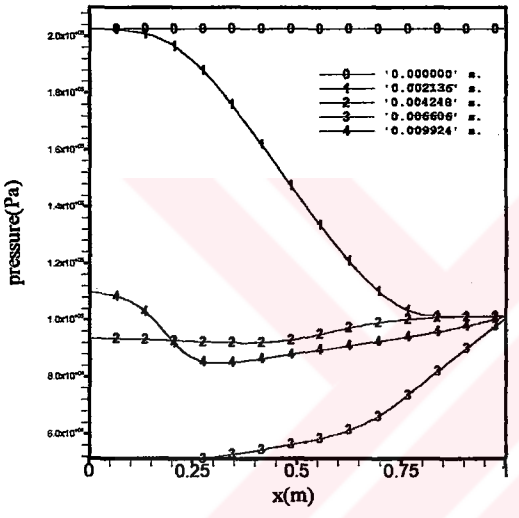
Six runs are made with increasing chamber to discharge pressure ratios as given in Table IV-4. These solutions are performed until the oscillatory steady state is reached.

Depending on the initial pressure ratio two different transient behaviors are observed. Referring to the solution plot for Run1, in Figure IV-12, for low-pressure ratios, at right end cell, pressure reaches to the atmospheric value immediately after the diagram at right end is opened. The maximum Mach number reached is around 0.5 in that case.



Density, Pressure and Mach Number
Variation for Run1
from t=0. to t=0.01 seconds.

Discharge to Chamber Pressure Ratio is 2.



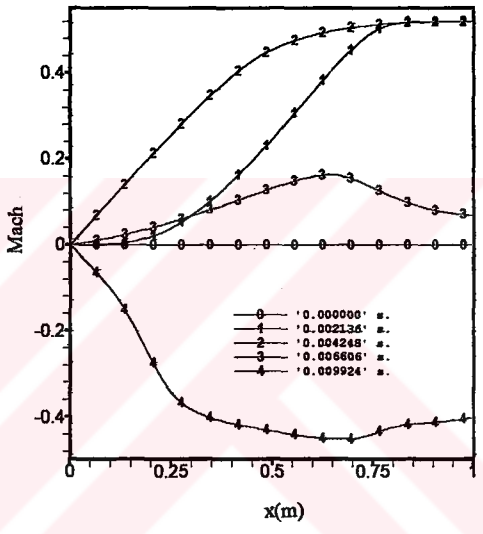
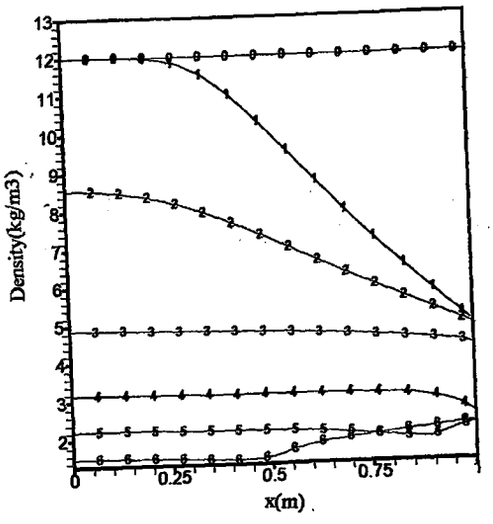


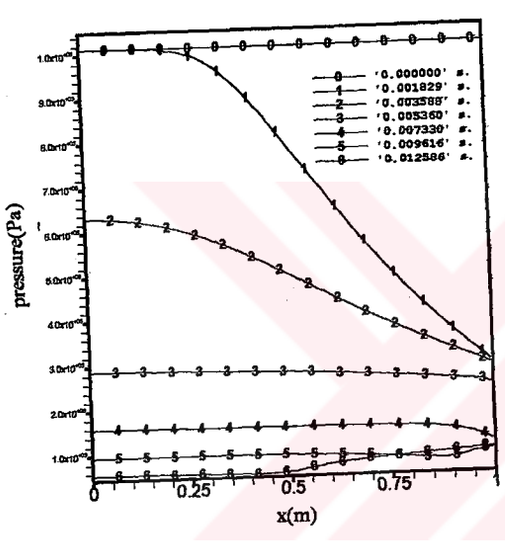
Figure IV-12: Behavior at low pressure ratios. Run 1.

For higher pressure ratios, such as the sample solution for Run5 given in Figure IV-13, before reaching to the atmospheric exit state, a temporary critical state is reached where the pressure at the open end is higher than the atmospheric value. The Mach number, during this time where pressure and density at the last cell remains constant, is very close to 1. As expected increasing the initial pressure ratio keeps this sonic condition and increases the temporary critical density and pressure values.



Density, Pressure and Mach Number
Variation for Run5
from t=0. to t=0.013 seconds.

Discharge to Chamber Pressure Ratio is10.



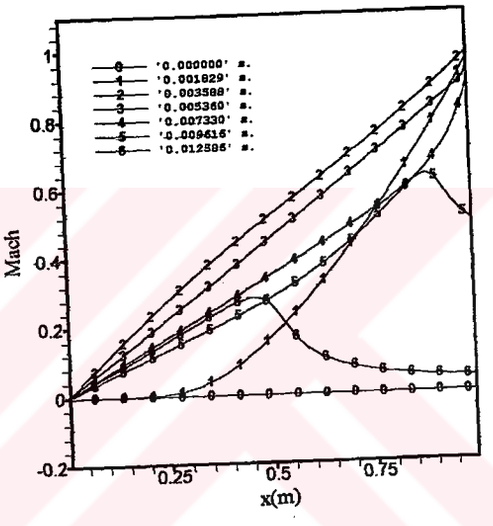


Figure IV-13: Behavior at high pressure ratios. Run 5.

In Figure IV-14, time dependent pressure and Mach number variation at the first and last cell (at open end) is plotted. The discussed behavior as a function of pressure ratio can be seen more clearly. After these transients are over the remaining solutions are oscillatory. The start of this oscillatory behavior can also be observed in Figure IV-14 for runs with low-pressure ratios.

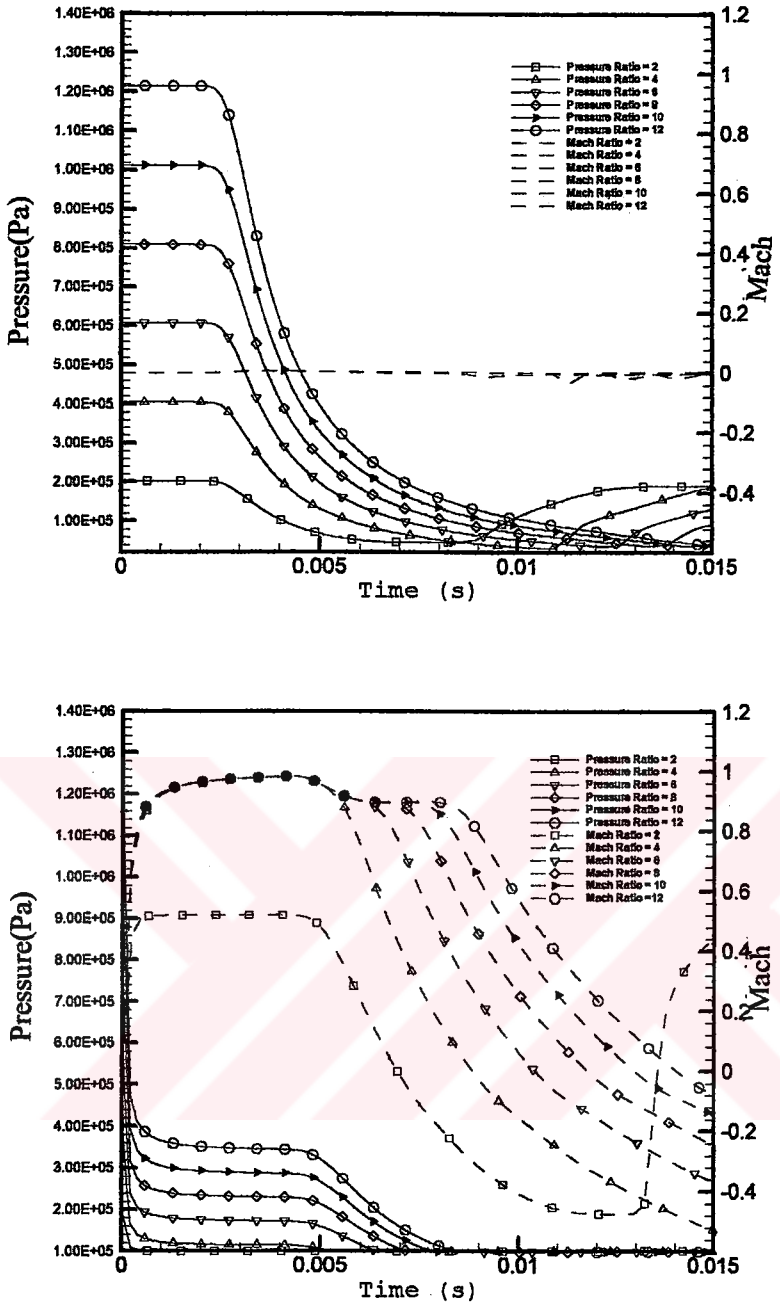


Figure IV-14: Pressure and Mach number variation at head-end cell (top) and last cell (bottom) of the open-end pipe, plotted on left and right respectively. Initial transients. (0-0.015 s.)

The frequency spectrum of this time variation will be analyzed. To demonstrate the possibility and contend of such an analysis, the carpet plot of pressure signal for the third run is plotted in Figure IV-15. Spectrum data analysis is performed for overlapped time intervals. Since the time-step generated by 1D-code depends on the wave structure in the

solution domain time intervals were not regular. They are made uniform by interpolation. The time step is chosen considerably low, so that it does not interfere with the expected frequency. Record length is 2^7 bits with 80% of the data are non-overlapping. A hanning data window is used. Sampling frequency is 6000Hz with half of this value is used as cut-off frequency. For comparison, the acoustic longitudinal frequency (the natural frequency of the organ pipe) of the chamber can be estimated as 140 Hz. For P_{ave}=1.5x10^{5}Pa and ρ_{ave}=2.5kg/m^3.

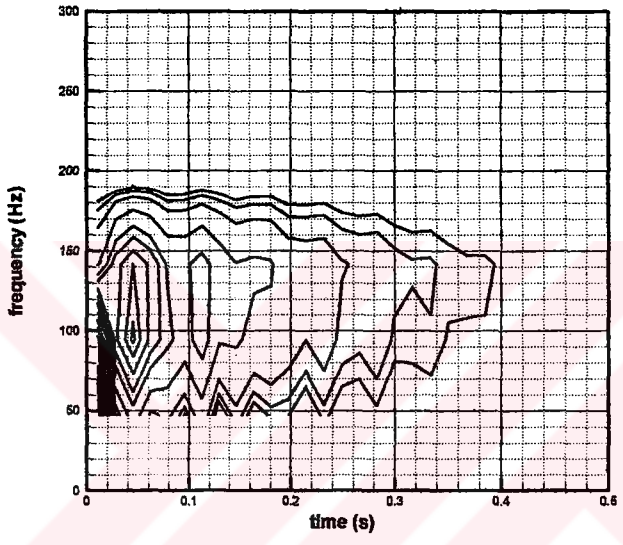


Figure IV-15: Carpet plot for frequency vs. time for pressure at first cell (head-end) node. Run 3 (0-0.5 seconds)

IV.5.4 1D STEADY NOZZLE TEST CASES

To verify source term calculations, a diverging nozzle test case is selected [51] Characteristic inflow/outflow boundary conditions are applied for two different flow conditions, supersonic/supersonic and supersonic/subsonic for the inflow/outflow boundaries, respectively. The nozzle area variation is given by the equation below

where x is in meters and $A(x)$ is in m^2. Nozzle shape is plotted in Figure IV-16. The nozzle length is 3.048 m.

$$A(x) = 0.0929 \cdot (1.398 - 0.347 \cdot \tanh(4 - 2.6248 \cdot x)) \quad (4.31a)$$

$$A'(x) = 1.21555 \cdot (1/\cosh(4 - 2.6248 \cdot x))^2 \quad (4.31b)$$

For both test cases, initially uniform flow is specified in the nozzle. When the steady state is reached, the flow field is compared with the analytical solution. The quasi-one dimensional solution is obtained for each case.

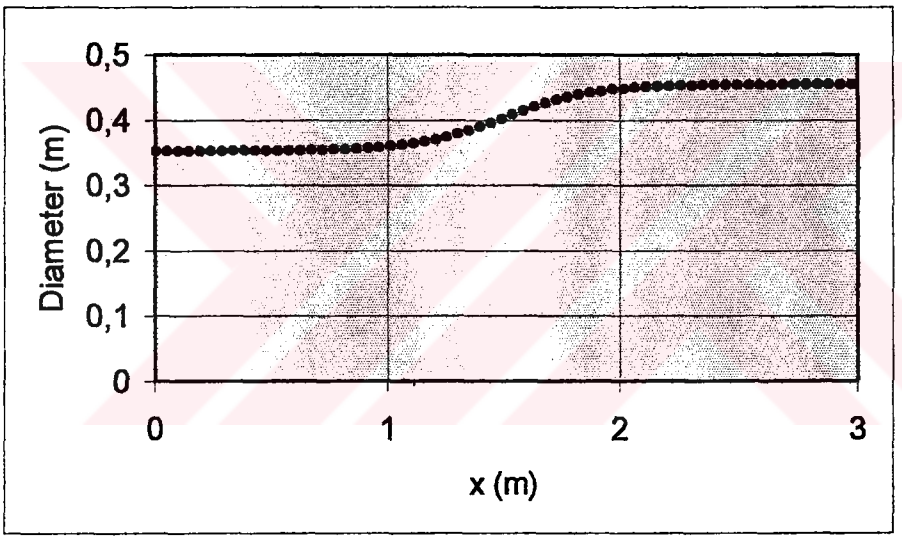


Figure IV-16: Nozzle shape used in steady nozzle test cases.

Gas properties are $\gamma = 1.4$ and $R = 286.9$ J/kg.K. The CFL number used in both test cases is 0.3. For both test cases, number of grids used is 100.

1D SUPERSONIC INFLOW, SUPERSONIC OUTFLOW

The supersonic flow at the inlet is specified by (4.32). Inlet Mach number is 1.5.

$$\begin{bmatrix} \rho_0 \\ u_0 \\ P_0 \end{bmatrix} = \begin{bmatrix} 1.2215\,kg/m^3 \\ 351\,m/s \\ 47880\,Pa \end{bmatrix} \tag{4.32}$$

All the variables at the exit fictitious cell are extrapolated, identical to a transmissive boundary condition. The analytical solution to this problem is obtained using gas dynamics tables. Comparison of computed and analytical results is shown in Figure IV-17.

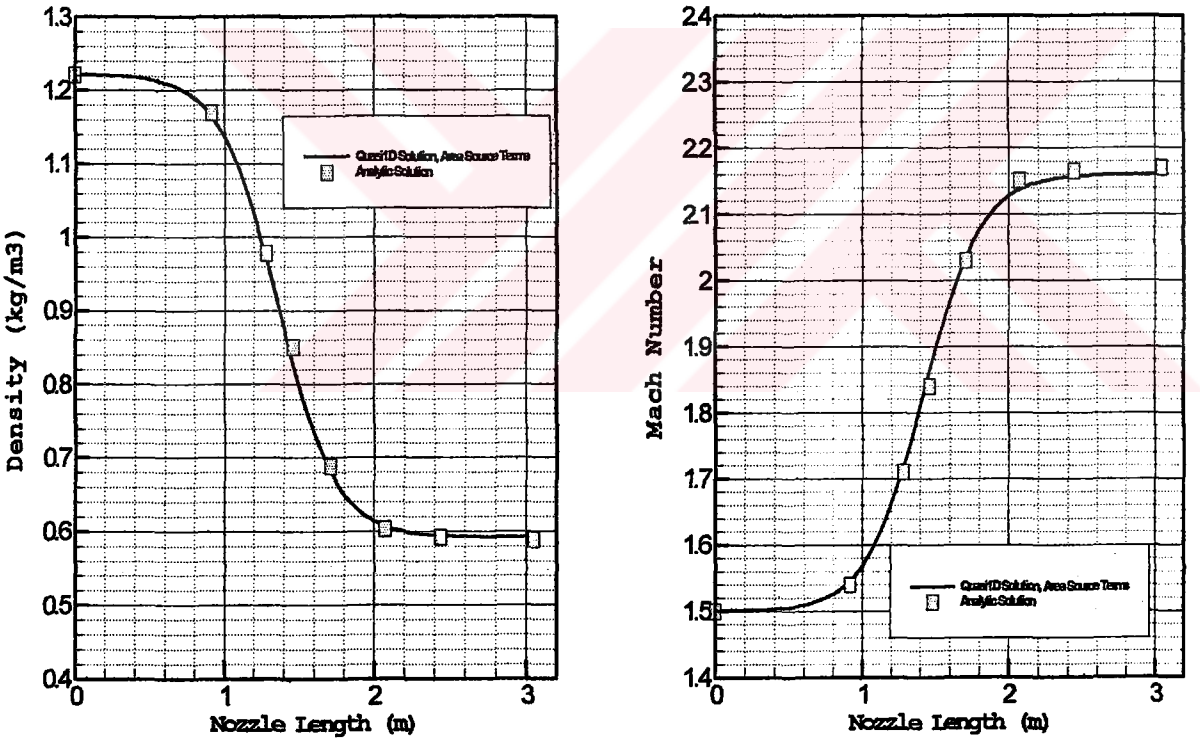


Figure IV-17: Comparisons of density/Mach number distributions for the steady state of supersonic inflow/outflow divergent nozzle test case.

1D SUPERSONIC INFLOW, SUBSONIC OUTFLOW

The solution of this test case contains, a normal stationary shock wave at x=1.524 m. To accelerate convergence to steady state, initial states given below in Equation (4.33) are used in the nozzle [51].

$$\begin{bmatrix} \rho_0 \\ u_0 \\ P_0 \end{bmatrix} = \begin{bmatrix} 1.2215\,kg/m^3 \\ 351\,m/s \\ 47880\,Pa \end{bmatrix} \quad x \le 0.8534 \tag{4.33a}$$

$$\begin{bmatrix} \rho_0 \\ u_0 \\ P_0 \end{bmatrix} = \begin{bmatrix} 1.2215\,kg/m^3 \\ 119.1\,m/s \\ 47880\,Pa \end{bmatrix} \quad x > 0.8534 \tag{4.33b}$$

For the subsonic outflow exit boundary, only the velocity is specified as 119.1 m/s. Results are plotted in Figure IV-18. There is agreement with the analytical calculations and no shock smearing is observed.

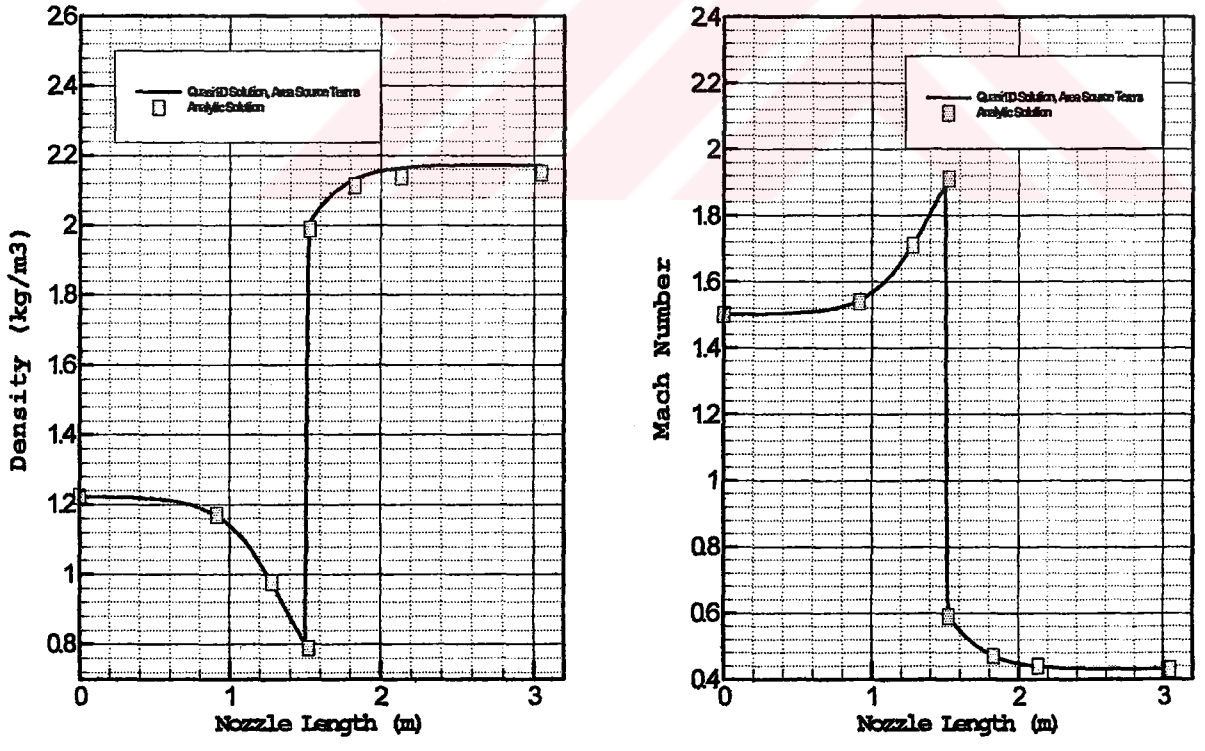


Figure IV-18: Comparisons of density/Mach number distributions for the steady state of supersonic inflow, subsonic outflow divergent nozzle test case.

High resolution of the exit shock wave is evident; this is due to a high number of grids in a relatively short domain and the numerical method used. Which is an encouraging result for the further studies.

This is a steady problem solved by a transient code. Since the code is intended for time-dependent problems a convergence history is not a direct output. However based on the total change in pressure, approximately 3000 iterations are needed for convergence for the CFL number and grid size used.

IV.5.5 1D MOTOR TEST CASE I (CYLINDRICAL TEST MOTOR)

A test motor, whose pressure and thrust vs. time histories are available from static tests, is selected as the first test case. It has a cylindrical grain. The predicted pressure-time histories are plotted and compared with static firings. Also the steady state pressure level is compared with the value obtained from a zero dimensional mass balance calculation.

IV.5.5.1 GEOMETRY

The assembly drawing of the cylindrical grain test motor, abbreviated as 2x4, is given in Figure IV-19.

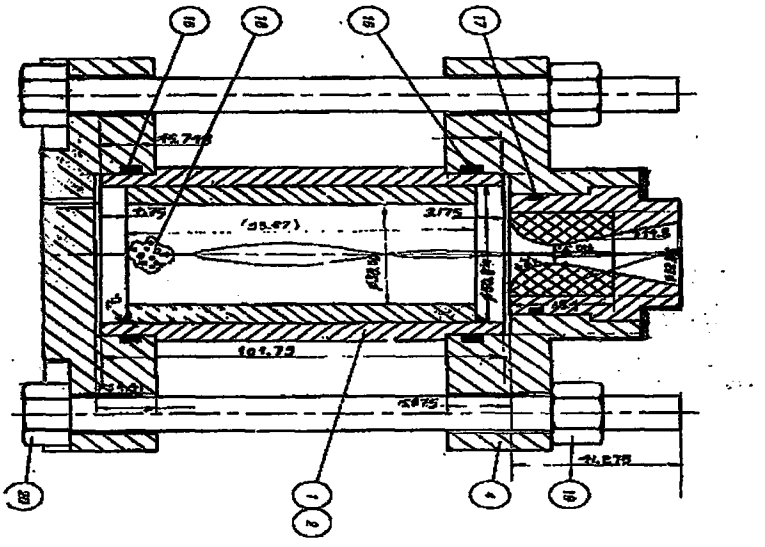


Figure IV-19: 2x4 Test motor. 1D Motor test case I.

To represent area burned correctly for 1D calculations, an equivalent length calculation is performed keeping the port area constant. Corresponding one-dimensional geometry is presented in Figure IV-20. Along the cylindrical grain average port diameter is used. The nozzle region is modeled by the area functions of Table IV-5.

Table IV-5: Geometry of the nozzle.

Momenclature	Description	Value
D_t	Throat Diameter	7.24 mm
D_i	Port Diameter = Nozzle Inlet Diameter	45.09 mm
m_l	Motor Length up to Nozzle	104.7 mm
x_{end}	Total Motor Length	209.4 mm
ch	Length of Convergent Section	$2\cdot(x_{end}-m_l)+m_l$
A	Constant	$D_i/2-(D_i+D_t)/4$
B	Constant	$(D_i+D_t)/4$
x_p	Position along x-axis	
θ	x_p in radians	$3\cdot(x_p-m_l)\cdot\pi/(2\cdot(x_{end}-m_l))$
D	Nozzle Diameter	$2\cdot a\cdot\cos\theta+2\cdot b$
A	Nozzle Port Area	$\pi\cdot D^2/4$
dA/dx	First Derivative of Nozzle Port Area	$\frac{2\cdot\pi\cdot D}{4}\cdot(-a\sin\theta\cdot 3\pi/(x_{end}-m_l))$

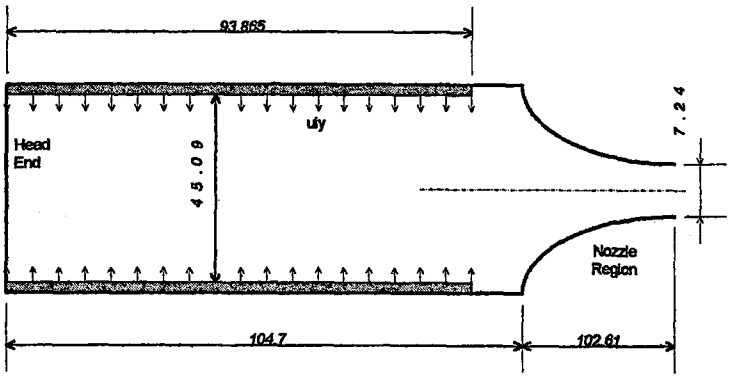


Figure IV-20: Simplified one-dimensional motor geometry of test case 1.

Thermochemistry codes are used to determine gas properties of the propellant. The propellant batch that is used in this test case has the following thermochemical properties at 2250 psi; R=320.31 J/kgK γ=1.2111 T_{flame}=3012.5 K. The total enthalpy at flame conditions can be calculated assuming constant specific heats, giving H_{in}=5535.9 kJ/kg. The speed of sound and characteristic exhaust velocity values can be calculated using these thermochemical properties as a=1081 m/s and C*=1509.67 m/s respectively.

Empirical pressure dependent linear burning rate law corresponding to the Equation (4.28) is given below as,

$$r_b[m/s] = 14.5177284e-6 \cdot P[Pa]^{0.44} \quad (4.34)$$

In the governing equations, momentum source term is zero since mass is injected normal to the motor axis.

IV.5.5.2 STATIC FIRINGS FOR COMPARISION

Two firings of the same motor with similar geometric dimensions are used for comparison. Some parameters that affect the final pressure-time history and performance are presented in Table IV-6. In this table results of the two predictions that are made using the quasi-one-dimensional code. One prediction is made with the nominal motor dimensions, taken from the technical drawing. Other prediction is made specifically for firing 1293, whose pressure-time history at motor head-end is given in Figure IV-19 over plotted with the corresponding test values. As seen from the table, small variations in burn area, port length, throat diameter and burning rate affect the average test pressure obtained.

Table IV-6: Geometric and Steady State Pressures for Static Firings and Predictions.

Firing Number	Propellant Batch	Burning Rate @ Pref and Tref (mm/s)	Average Area Burn (m^2)	Average Port Diameter (mm)	L Port (mm)	Average Pressure Test (Mpa)	Average Pressure Predicted From Zero D (Mpa)	Throat Dia (mm)
1293	XX-X-003	14.82	0.0127616	44.50	91.288	14.58	14.26	7.747
1D Prediction of Firing 1293	XX-X-003	14.82	0.0127616	44.50	91.288	**14.42**	14.26	7.747
1292	XX-X-003	14.54	0.0128633	44.62	91.770	16.10	16.05	7.525
1D Prediction with nominal motor dimensions	XX-X-003	**14.82**	**0.0132964**	**45.09**	**93.865**	**19.98**	**19.54**	**7.240**

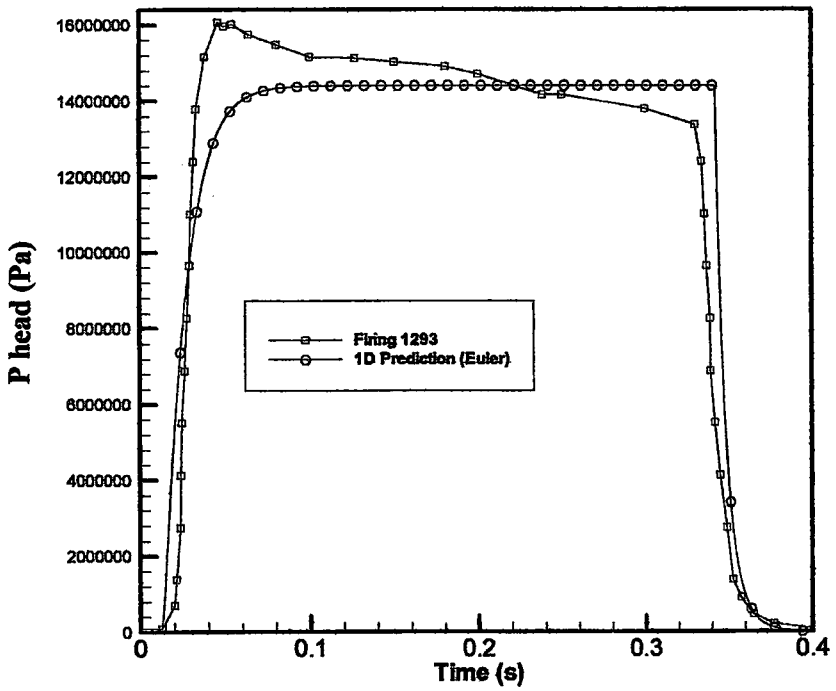


Figure IV-21: Head End Pressure (Comparison with Static Firing 1293)

As seen in Figure IV-21, agreement of motor chamber pressure with the experiment is good. The initial peak in firing 1293 may be due to the start-up erosion/volumetric change, which affects the experimentally determined burning rate. Chamber pressure further decreases due to the erosion of graphite nozzle insert. This throat erosion is absent in the computational model. In the one-dimensional prediction, propellant disappears in a zero time interval. However in the real case some sliver may remain in the motor that effect the shape of the tail-off region. However, general shape of the motor depressurization curves is similar.

IV.5.6 1D MOTOR TEST CASE II (END BURNING TEST MOTOR)

This test motor is relatively larger and has a cylindrical planar end-burning grain. The complete assembly drawing is given in Figure IV-22. The basic motor and nozzle dimensions are given in Table IV-7. Nozzle area functions are same with the previous test case and they are available from Table IV-5.

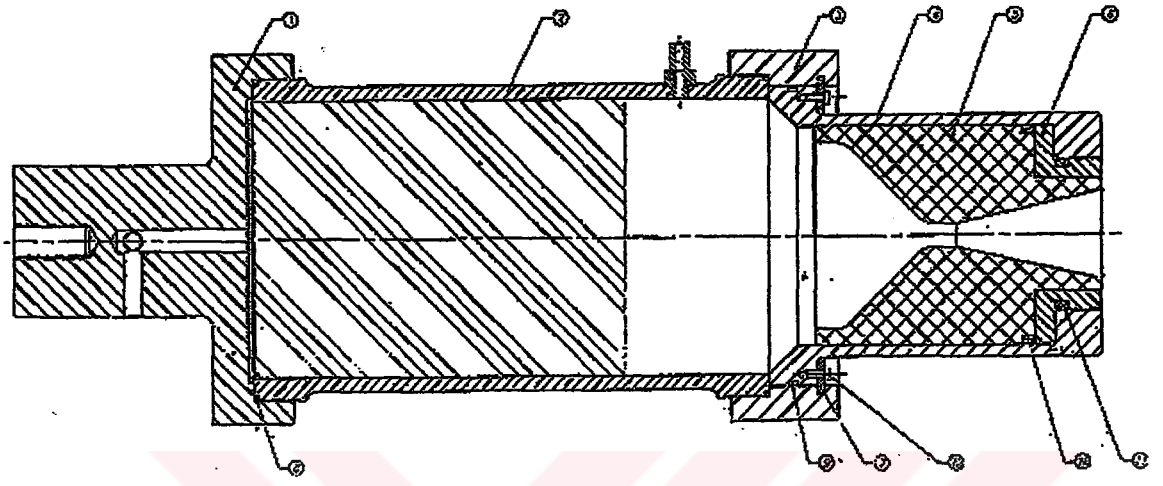


Figure IV-22: 6C4 Test Motor in End Burning Configuration. (1D Motor Test Case II)

Continuity and energy source terms are identical to the previous test motor, whereas momentum source, due to the end-burning configuration, has a non-zero source term in this test case.

Table IV-7: Geometry for the Second Motor Test Case.

Initial Motor Length	255.8 mm
Initial Port Length	82.2 mm
Final Motor Length	463.3 mm
Predicted Burning Time	30. s
Throat Diameter	12.95 mm
Motor Inside Diameter	152.4 mm

Predicted pressure rise at the propellant surface is plotted in Figure IV-23. The solution is performed until the steady state is reached. This steady state pressure level

can be calculated from a zeroth order mass balance. Equilibrium between choking nozzle mass flow rate and injected mass flow rate leads to the Equation (4.35) given below.

$$P_{steady} = \left(\frac{C^* A_{burn} \rho_{propellant} a}{A_{throat}} \right)^{\frac{1}{1-n}} \tag{4.35}$$

With the given dimensions and propellant properties, $P_{steady} = 4.308$ MPa is obtained. and from the full transient 1D moving boundary code P_{steady} is predicted as 4.385 Mpa.

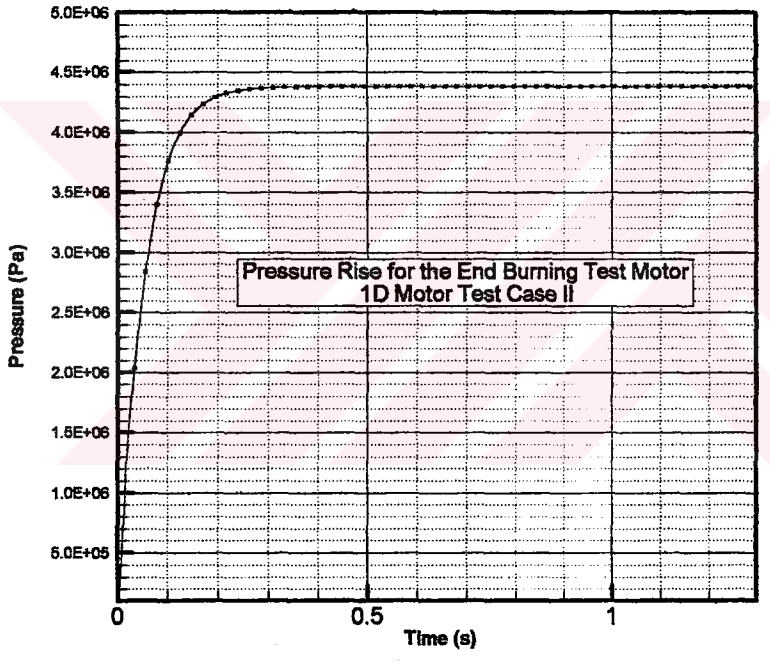


Figure IV-23: Predicted Pressure-Time History for the End Burning Test Motor.

In Figure IV-24, Mach number distributions inside the motor are plotted at three consecutive times, showing a traveling shock in the nozzle, which is developed during the starting period.

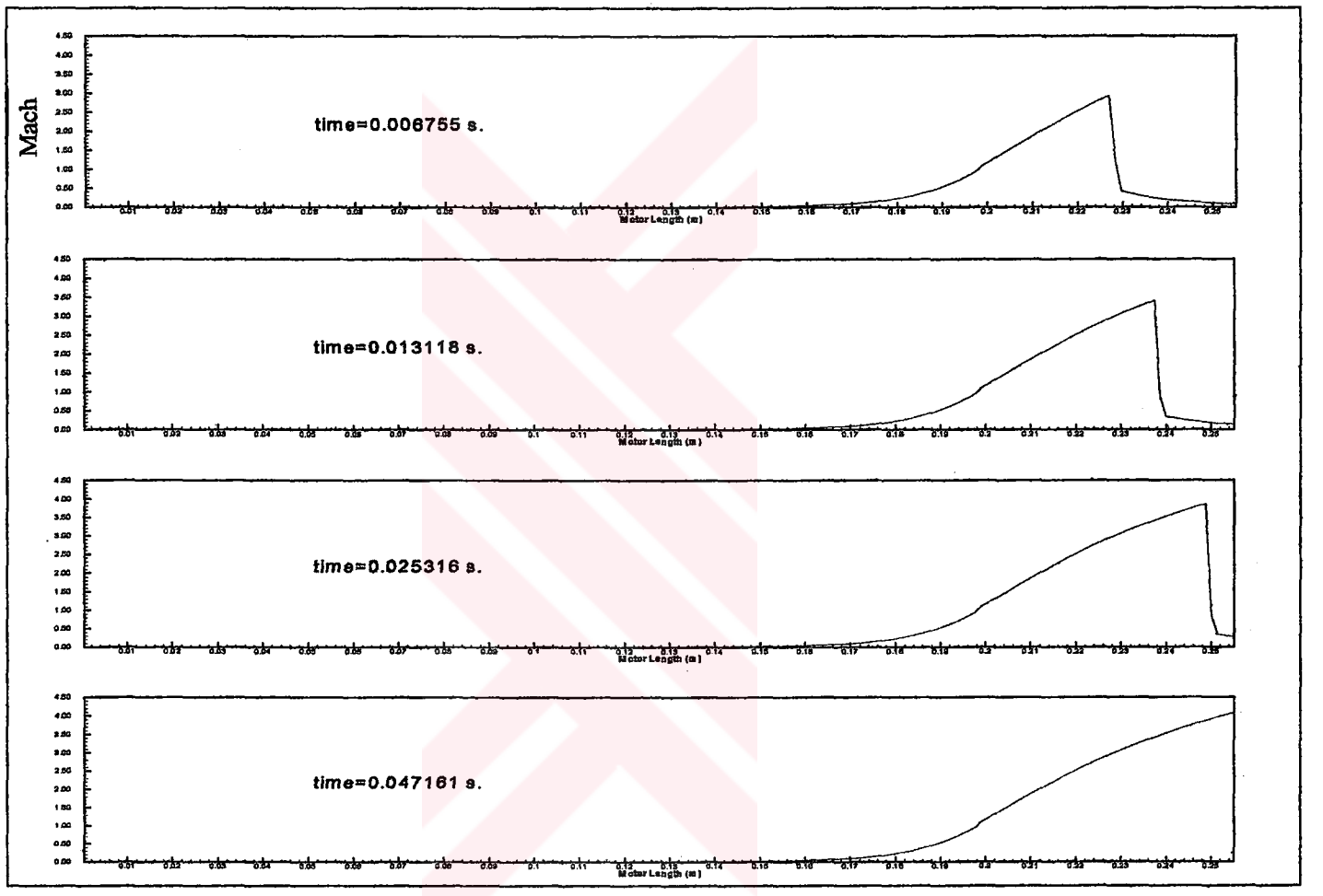


Figure IV-24: Mach Number Distributions of the End-Burning Test Motor during Start-up.

CHAPTER V

TWO-DIMENSIONAL CARTESIAN GRID AND SOLVER

V.1 INTRODUCTION

In this chapter a two-dimensional Cartesian grid Euler solver will be explained. The solver is intended for complex geometries, with any number and combination of solid and moving boundaries. Cartesian grid approach is chosen to facilitate an efficient and rapid geometry definition. As will be demonstrated, Cartesian grid intersections depends on the local topological character of the boundaries. For this reason the end user of the code does not have to perform a special grid generation process for the given application. The geometry dependent problems of classical grid generation are eliminated. The generated Cartesian grids are rectangular and may not be in the best possible quality for the given application. However it is general, and with a single program satisfactory results are obtained for different type of applications. For boundaries with sharp corners, this approach is another alternative to triangulation and multi block structured grids.

The Cartesian grids became unstructured near the boundaries. The triangular, quadrilateral and pentagonal elements may coexist even in a fairly simple problem. Since the geometric possibilities are close to unlimited a systematic approach is a

requisite throughout the study. The structured programming features, CAD algorithms and approaches from constructive solid geometry are indispensable for the fulfillment of this highly geometric task. Including the axisymmetric and solid propellant regression models that will be discussed in the next chapter, the lines of code sum up to 10267.

V.2 DEFINITIONS AND TERMINOLOGY

Before starting, a terminology should be developed to discuss Cartesian grid approach. Since the literature about Cartesian Grids known to the author's knowledge, up to now, does not enfold a complete moving boundary treatment, some definitions are original to this work.

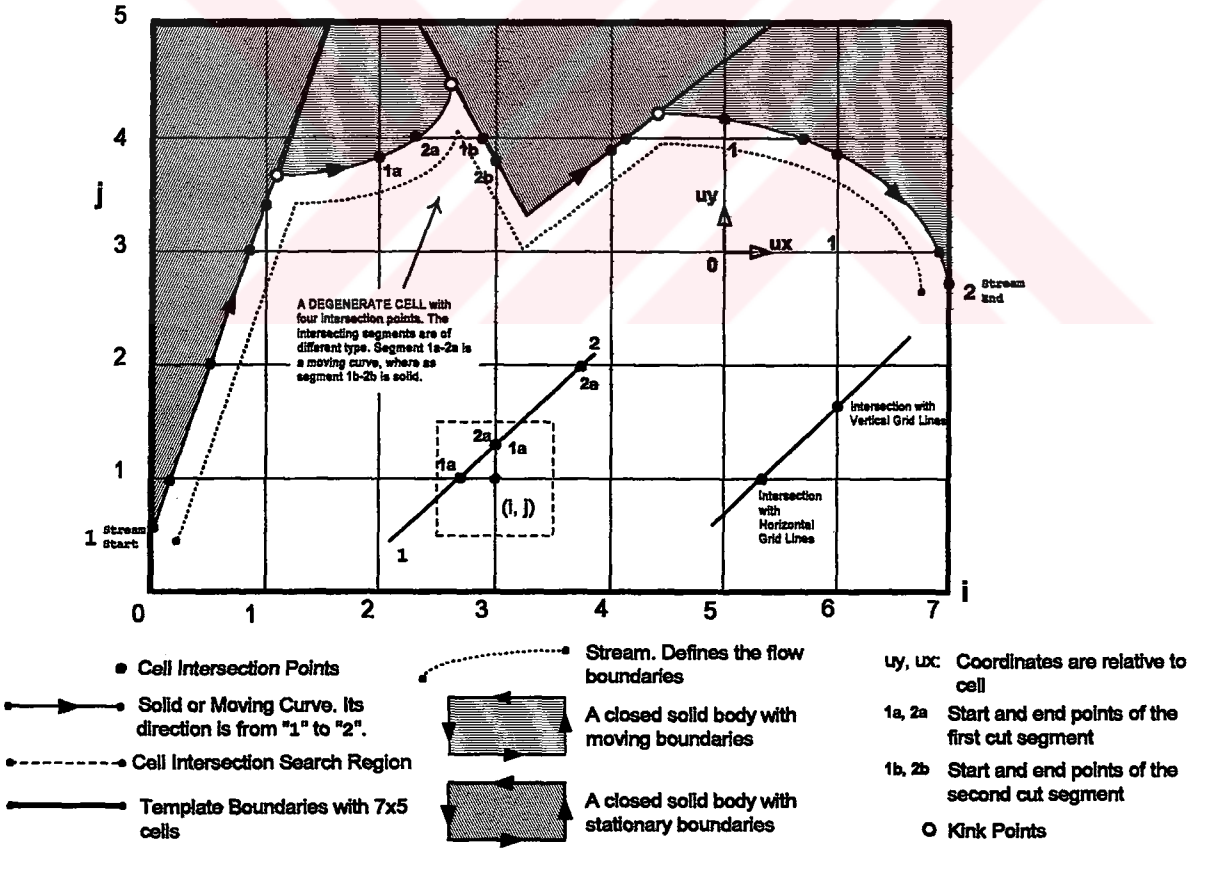


Figure V-1: Terminology for cell intersections.

The solution domain is rectangular. Given its length and width, square grids are generated inside. The rectangular boundaries and square grids form the solution "template". Square grids are termed as "cells". In the template plotted in Figure V-1. there are seven and five cells, in the x- and y- directions, respectively. Template boundary sides are EAST, WEST, SOUTH and NORTH, where inflow/outflow, reflective, transmissive, injecting, moving wall boundary conditions can be specified. Also for the side of any cell inside the template, a boundary condition can be assigned. This feature may be useful in some specific problems and increases the versatility of the code.

In applications, solid and moving walls are bordering the flow field of interest. "Curves" that form the boundaries of solid/moving walls are specified as line segments, in an order so that when the curve parameter increases, solid bodies are always on the left.

In a general problem there may be moving and solid boundaries connected in series as shown in Figure V-1. These series of curves are termed as "streams." In streams intersection points of moving and solid curves are labeled as "kink points". A stream can be composed of a single moving or solid curve. Streams are allowed to loop, forming voids or closed bodies, with coinciding start and end points.

A boundary curve, passing through a template cell, intersects its sides in two points. Tracing the curve with the solid region being on the left-hand side, first intersection is denoted as "point 1a" and second as "point 2a." Cell intersection points are stored with respect to coordinates relative to the cell.

If number of intersection points in a cell exceeds two, cell types that are not covered by the solvers' cell-type-domain may appear. These cells need special treatment and named as "degenerate cells" in the Cartesian literature [5]. Extra intersection points will be labeled and stored as "point 1b" and "point 2b." A simple degenerate cell example is given in Figure V-1.

V.3 SOLUTION PROCEDURE

The main algorithm consists of the steps that are listed in Figure V-2.

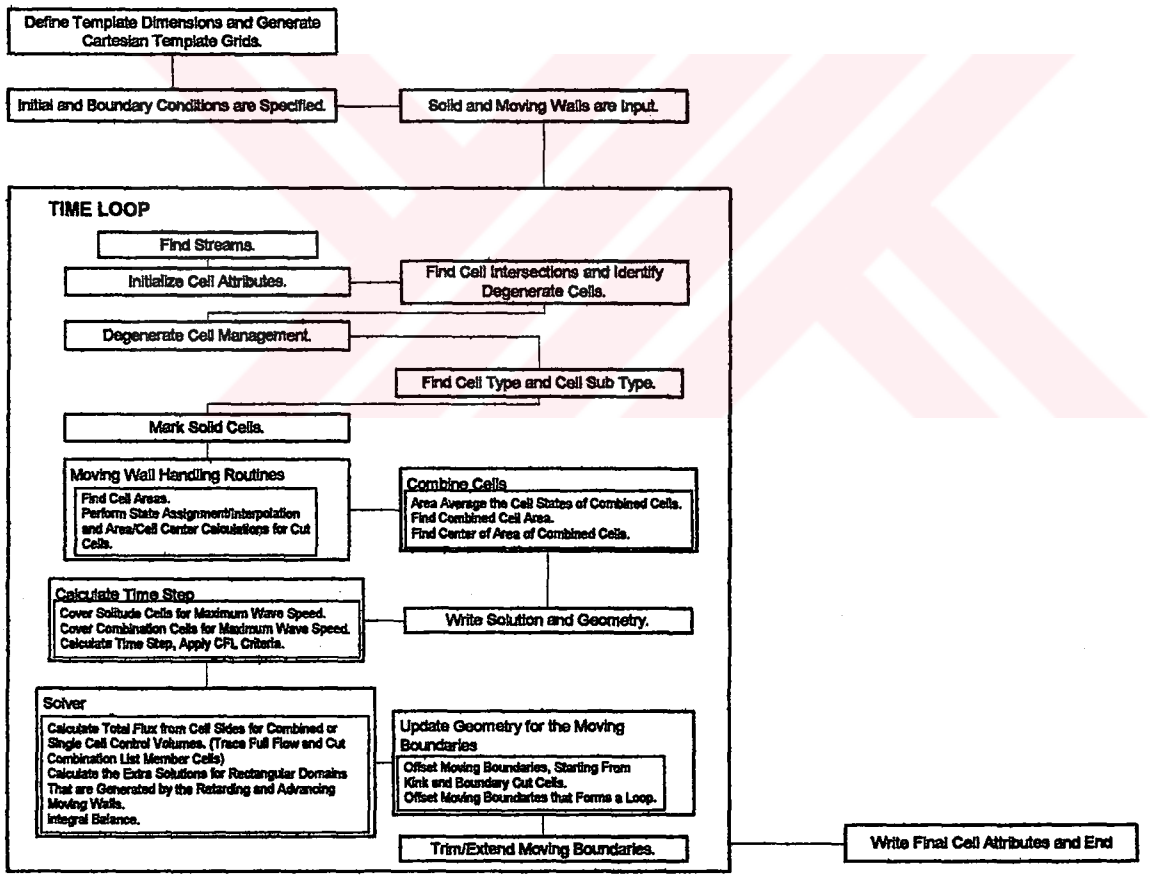


Figure V-2: Solution algorithm and time loop.

V.3.1 CARETSIAN GRID GENERATION ROUTINES

Cartesian grid solvers does not require a separate grid generator programme. However, when the aim is arbitrary complex geometries, generation of the template grids and finding local intesections becomes a complicated geometric task. An addition to this is the complexity brought by the boundary movement in the solution domain.

Unlike boundary-conforming grids where global remeshing, at each time step, alters the positions of grid points [35] [58], Cartesian grid points are stationary. Thus for Cartesian control volumes that are not cut by a moving boundary, any geometric conservation law or a Jaeobian transformation that cares for the time dependent grid movement is not applied.

V.3.1.1 INPUT FILES AND MODELLING OF SOLID/MOVING WALL BOUNDARIES

Data specifying boundary conditions, template dimensions, gas properties and geometry of solid/moving curves is input to the code via the related data files. The contents of these files are presented in Appendix B.

Curves that form the boundaries of solid or moving walls are approximated with line segments. A line segment is the lowest object of the stream structure. It is defined by the parametric equation of straight line. Parametric representation of line segments is found useful especially in finding cell intersections. For this type of representation, position vectors of the start and end points of the straight line are needed. A single

curve parameter defines the position of any point on the line segment. In the code, the line segments are defined using the following C++ structure;

```
struct line {
            double p0[2] ; /* start vector */
            double p1[2] ; /* end vector */
            double n[2] ;  /* unit vector in offset direction */
            double rb ;    /* offsetting velocity */
            };
```

Besides start and end position vectors, segment center velocity and its magnitude are also stored to allow for variable offsetting along a moving type curve.

The curve structure used in the code composed of the line segments, number of line segments that make up the curve and type of curve, which may be moving or solid.

Line segment sequence of each solid/moving curve is arranged so that solid bodies are always on left. The geometric information of the input curves are kept in the following structure:

```
struct curve {
             struct line seg[N_SEGMT] ;
             short int type ;        /* SOLID or MOVE */
             int n_segmt ;           /* end indis of seg[]
                                        end_ind=(# of segments-1) */
             };
```

V.3.1.2 CELL SIZE AND NUMBER OF CARTESIAN GRID LINES

Since the domain is rectangular and the cells are square, number of grid lines in x- and y- directions are dependent on each other. First the minimum number of grids in each direction is calculated, which is the coarsest possible grid for the given template dimensions. Then, if an extra refinement is required, number of grids are increased in both directions with the same ratio. If the template side dimensions are not whole numbers, then they should be expressed in ratio form, i.e. numerator over

denominator. In that case, to calculate cell size and template grid line locations accurately, integer arithmetic should be performed. Please refer to Appendix B, for the details of calculations and related input file used.

V.3.1.3 FINDING STREAMS

In a given geometry, composed of arbitrary number of solid and moving curves, at the start of the problem, i.e. at initial state, some part of the solid boundaries are covered with moving curves and may not see the flow region. For a propellant type regressing material this situation is shown in Figure V-3. At the later times of the solution, these solid boundaries will expose out and start affecting the flow. In order to obtain the transient solution, at each time step the flow boundaries should be determined. Curves defining flow boundaries or wetted areas are called streams and the corresponding process is stream generation.

In Figure V-3, initially there is one moving curve (A-B) and one solid curve (1-c.c-2). Solid curve is composed of two line segments and the initial stream is made up of the moving curve (A-B) only. As the boundary moves, at a later time step the number of moving curves increases to two. The stream is now made up of three curves (a-b, b-c.c-d, d-e). In a general problem, the number of streams may be more than one.

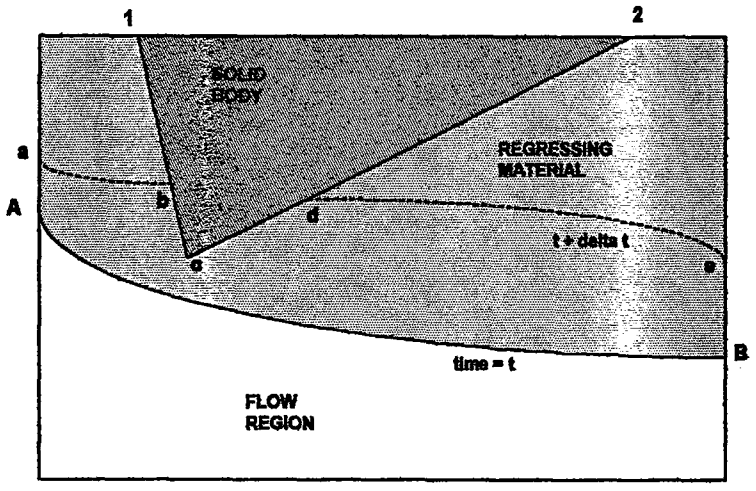


Figure V-3: A Solid Boundary Exposing out Which is Initially Covered with a Regressing Material. (A-B is the Stream at Time t. a-b.b-c.c-d.d-e is the Stream at a Later Time.)

For finding streams, algorithms developed in this study covers situations involving merging and break-up of moving boundaries and also generate loop streams.

If there are no moving walls and all the geometry that defines fluid boundaries are solid, then each solid wall is assigned as a new stream. If there are moving walls, new streams are generated by tracing moving and solid curves alternatively. In this trace, "A stream can form a closed loop or start and end at a template boundary" is the basic rule. Until this rule is satisfied, each trimmed/extended moving curve is traced first in its start direction and then towards its end. During this trace, intersections with other curves will be detected. Each new detected curve during this trace is kept in the order as a member of the generated stream. The stream information is kept in the following structure:

```
struct stream {
              struct curve sw_mv[N_CURVE] ;
              int n_curve ;
              short int cloop ; /* Switch for whether
              the stream is closed or not */
              };
```

V.3.1.4 CARTESIAN CELL INTERSECTIONS

The basic geometry, which defines the variables used in the algorithm, is presented in Figure V-1. For each stream, their line segments are searched for an intersection with vertical and horizontal grid lines. If an intersection is found, intersection point and its type (moving or solid) are stored in the cell structure relative to cell coordinates. The coordinates should be specified relative to square cells because of the accuracy considerations. This algorithm is different than the one proposed in [5]. In that study boundary curves were traced and cell coordinates are defined as integer variables taking discrete values to overcome the accuracy problem.

The intersection routine considers the sense of each line segment and covers different cases including specific orientations of segments that are parallel to Cartesian grid lines. These cases are compiled in Appendix C for future reference.

Two intersections are allowed and typical for each Cartesian cell. If more than two intersections are found, their positions are stored for degenerate cell considerations.

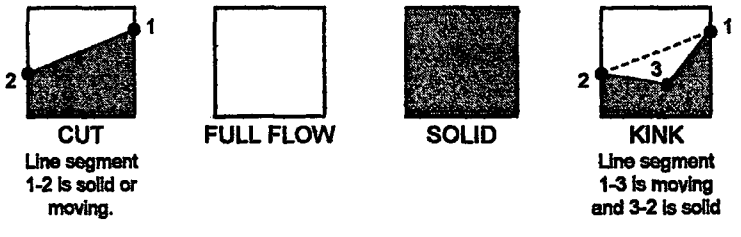


Figure V-4: Basic cell types (SOLID, FULFLOW, CUT-M/S, KINK)

For geometries involving moving and solid boundaries, there are five basic cell types. These are: full flow, solid, cut-solid, cut-move and kink cells as shown in Figure V-4. Cut cells contain a single curve segment, which may be moving or stationary. Kink cells are cut cells where a solid curve ends and a moving curve starts or vice versa. The same convention also holds for segments inside cells: the solid part is on left, in the direction of curve parameter increase.

V.3.1.5 CARTESIAN CELL STRUCTURE

In the solution domain each square formed by the grid lines defines a control volume, named as cell. Number of cells are equal to the number of grid lines. For each cell, besides flow variables the following information is also stored:

- Basic cell type (Figure V-4)
- Cell sub-type (Figure V-5)
- Position vector of first and second intersection points.
- Intersection types of first and second intersections. (Which depends whether the cell is intersected at that point by a solid or a moving curve.)
- Position vector of first and second degenerate intersection points and their types.
- Position vector of the kink point.
- List number that the cell is combined.
- Cell area.
- Position of cell center.
- The boundary condition specified for any of the cell sides.

V.3.1.6 CELL SUB-TYPES

The intersection type and the cell side each intersection point is located determines the cell sub-type. These sub-types are grouped and presented in Figure V-5 in compact form. Depending on the conditions given in Figure V-5, the solver differentiates 48 different cell sub-types. Although only cut-moving/solid (CUT-M/S) cells are drawn, for each cut cell sub-type there is also a corresponding KINK cell with intersection points at the same locations. In that case the kink point, point-3 in Figure V-4, is located at an arbitrary position inside the Cartesian cell.

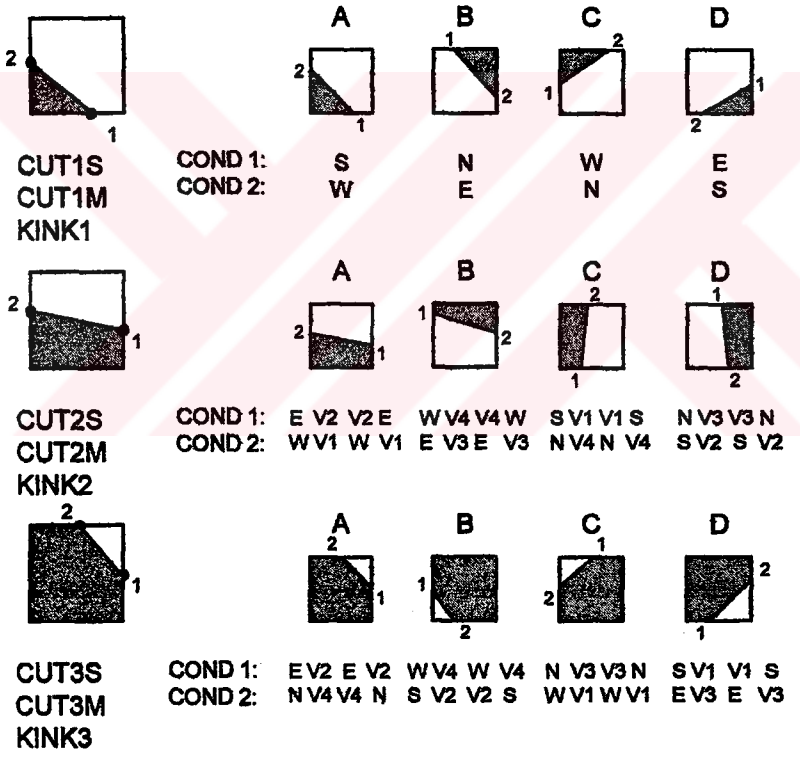


Figure V-5: Cell types; cell sides (E, W, S, N) and four vertices (V1, V2, V3, V4) clockwise, starting from southwest corner are possible locations for the intersection points.

V.3.1.7 DEGENERATE CELLS

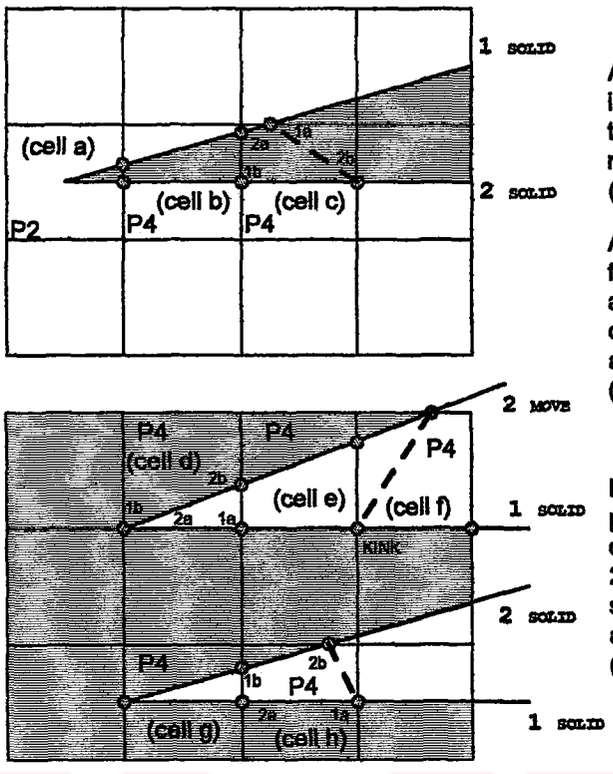
In the solution domain, if arbitrary intersections with the input geometry are allowed, some cells that are not recognized by the Cartesian Solver may appear. These cells are named as Degenerate Cells in the Cartesian literature. (A simple example is a cell with more than two intersections, Figure V-1) By increasing grid size or slightly shifting the input geometry some of the problematic cases can be overcomed.

However such remedies work only for bodies that are not moving. For applications involving continuously changing shapes and offsetting, these problematic geometries must be identified and suitably approximated.

Table V-1: Number of possible unit geometries for each degenerate cell problem type.

PROBLEM TYPE	NUMBER of UNIT GEOMETRIES
P1	*32*
P2	*8*
P3	*64*
$P4_{diagonal}$	*16*
P4	*88*

Depending on the number of intersections in the degenerate Cartesian cell, geometrically possible problem types can be grouped in to four. These problem topologies will be labeled as P1, P2, P3 and P4 cells (With one, two, three and four intersections in a Cartesian cell respectively.) For each problem type, the rotations and symmetries of the basic geometry should be considered, together with the type (moving or solid), of the intersecting boundary. The total number of unit operations that is taken into account for each problem type is given in Table V-1.



All P2 cells, whether their intersecting line segment type is same or different, are marked as FULL FLOW. (cell a)

All P4 cells, that has the following points (1b and 2a) and (1a and 2b) placed in opposite sides are marked as FULL FLOW or SOLID. (cell b, d, e, g)

P4 cells that 1a and 2b points are placed on different sides and 1b and 2a are placed on the same side, make boundary approximation. (cell h, c, f)

Figure V-6: P2 and P4 type degenerate cells. In all figures dashed lines represent the approximated boundary. Both line segments are of same type (Both move and both solids). For the definitions of cell intersection points 1a, 2a, 1b and 2b refer to Figure V-1.

To disclose the scope of degenerate cell work, some examples of the primary geometries and rules that are used in the code are given in the Figures V-6, V-7 and V-8. For degenerate cells with different curve type of intersections, approximated geometry becomes more accurate and results a kink type of cell. Without considering these topologies, an efficient moving body Cartesian solver is not possible.

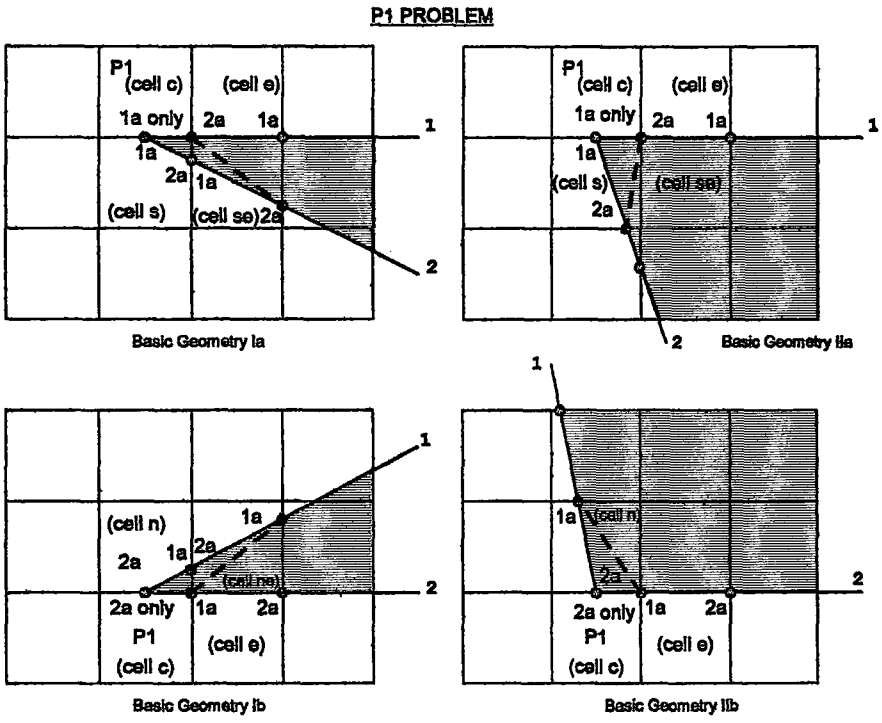


Figure V-7: P1 problem. P1 problem is possible if the segments that make a sharp corner are of the same type and one of the segments coincide with the template grid lines. There are two basic geometry types. (And each type has two variations, a and b) For the basic geometry type 1a (top left figure), to overcome the ambiguity in cell intersections; Mark (cell c) and (cell s) as FULL FLOW. Move Point 1a of (cell se) to Point 2a of (cell e). Each of the four topologies that are plotted above has also four different orientations depending on the position of the cell that is marked 1a/2a-only.

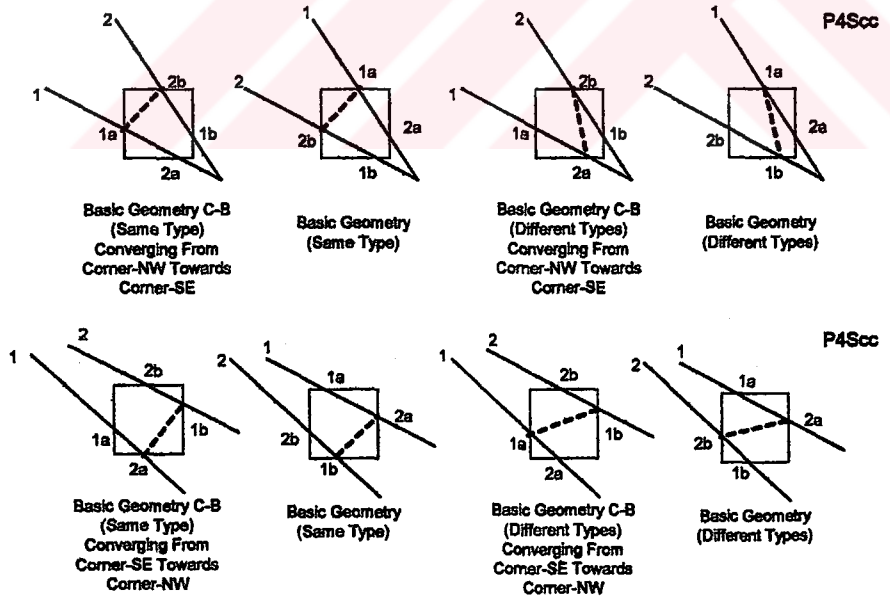


Figure V-8: Diagonal P4 problem. One of its four sub-types. The four cases that are shown on the left are same type of curve intersections and different type intersections are on right.

V.3.1.8 MARKING SOLID CELLS

Following the above approach attributes for each cut cell in the solution domain is now known. The remaining cells are either solid or full flow. For solid cells, no flow solution is needed. Hence they must be distinguished from the full flow cells before starting the solution. Since the geometry is going to change at each time step calculations should be repeated.

The procedure that is developed for identifying full-flow cells is similar to the one used in [5]. All the Cartesian cells are traced first horizontally and then vertically. During each trace, cell type does not changed and marked as either SOLID or FULFLOW, depending on the initial cell type assumption. When a cut cell is reached, during these traces, the marking type is reversed and switched to FULFLOW or SOLID. The tracing proceeds with this marking type afterwards. By taking into account the detected cut cell sub-type, initial cell type assumption and previous type marks are corrected. The algorithm is tested in various complex geometries and found to be working in all cases considered so far.

The solid cell marking procedure of [5] does not take the cut cell sub-type into account. For this reason an extra trace, either in horizontal or vertical direction is needed. Even with this extra trace, ambiguous geometries are still possible. In this study, depending on the cut cell sub-type initial cell type assumption is corrected when the marking type is switched. By that way, the number of horizontal and vertical traces is decreased to two.

V.3.1.9 STATE INTERPOLATION/ASSIGNMENT FOR NEW DOMAINS CREATED BY RETARDING MOVING BOUNDARIES

When the boundary geometry of the problem is updated at each time step due to retarding boundary movement, "new domains" are created inside Cartesian cut cells. These domains whose solution is not yet known, does not exactly overlap with the "extra solutions" that are calculated during the solver phase. ("Extra solutions" have been defined in Section IV.3.6.) These "extra solutions" are valid for rectangular regions that are generated by the cut segments sweeping during the time step.

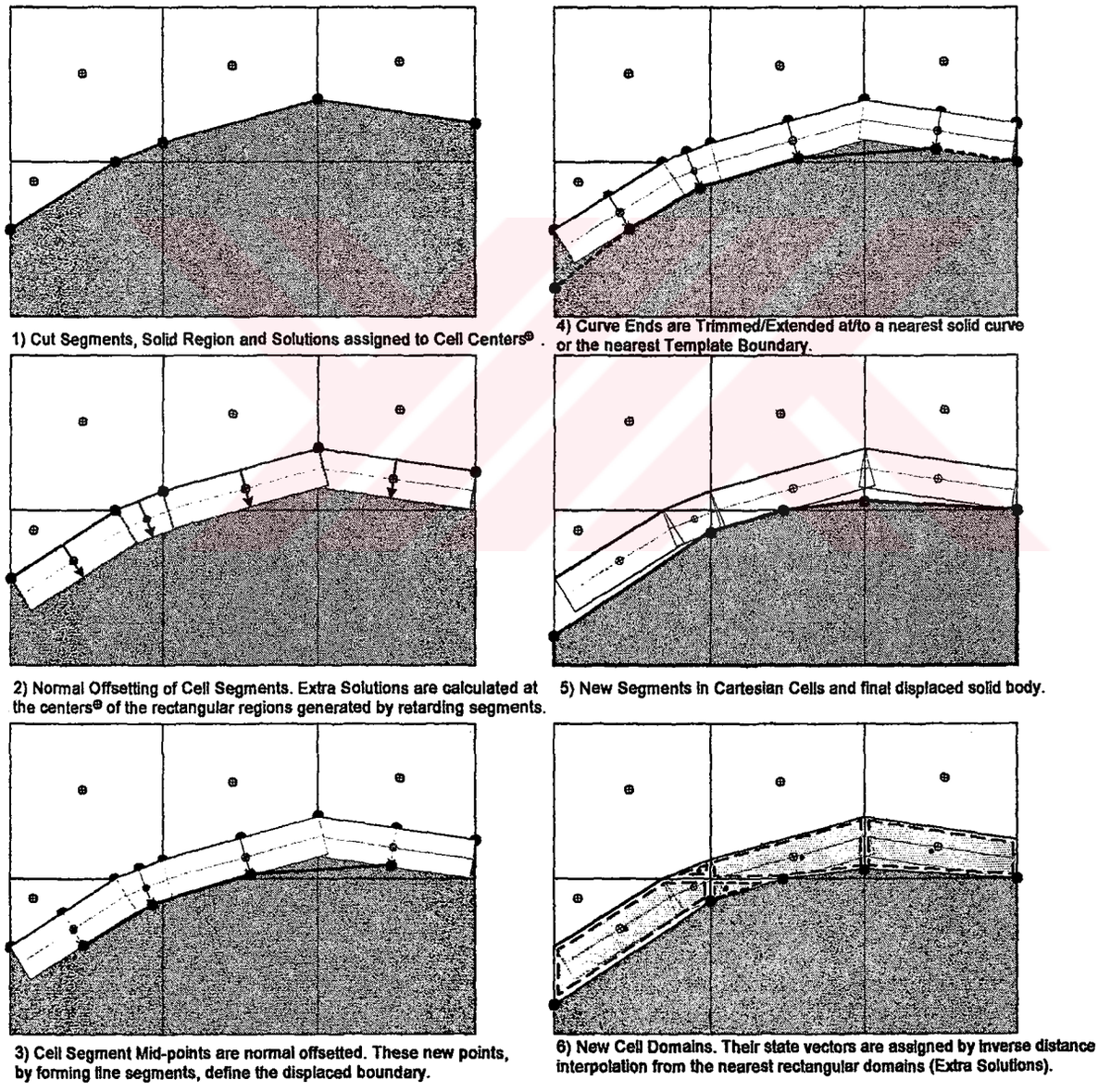


Figure V-9: Steps of boundary offsetting and state vector interpolation/assignment.

New cell center and area of the investigated cell determines if there is a created domain without a solution. The center position of this new solution domain is calculated. Nearest four rectangular extra solutions are then found, their state vectors are inverse-distance approximated and assigned as a solution to the new solution domain of the investigated cell. The visual interpretation of the algorithm is depicted in Figure V-9. This procedure is exact for purely translating boundaries without rotation. Offset boundary movement is a translating motion in the surface normal direction, which occurs in surface reactions, evaporation or agglomeration. Inverse distance interpolation formulas are presented in Appendix C.

V.3.1.10 CELL COMBINATIONS

Intersections of arbitrary line segments may produce tiny cells, which minimize the time step size. If an efficient Cartesian Cell solver is the aim, these tiny cells must be combined and treated as a single control volume. Combined cells become a member of the same list. These combined cell groups will be referred as "combination lists" and the assigned list number will identify them. When finding the cell to be combined the segment mid-point normal rule [5] is practiced here. The cell, which is found by this rule, is known as the best combination.

Around confined regions and for cut cells with neighboring template boundaries, the best possible combination may not exist or the planned combination may produce an undesired size increase that decreases the local spatial accuracy. In order to acquire a consistent control volume size as much as possible, throughout the template, maximum three alternatives of the four neighboring cells are returned to the code in

the order of the best combination possibility. In cases when the best combination cell does not exist other alternatives are considered in order.

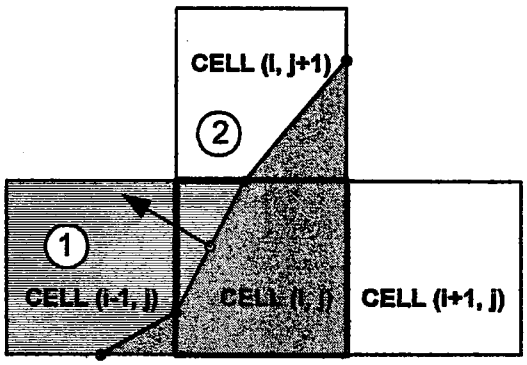


Figure V-10: Cell (i, j) is combined with the first of the two alternatives.

In Figure V-10, the segment mid-point normal marks cell (i-1, j) as the best combination candidate. If grouping with this cell is not possible the second alternative, cell (i, j+1) will be considered. There are no other alternatives since east and south sides of cell (i, j) neighbor solid material.

For all the combined cells in the template, cell states are area-averaged, area and cell center of the combined cells are found.

In Figure V-11, Cartesian Grid Information from various template locations of the letters 'S' and 'A' are presented. These closed-up positions are selected to demonstrate some critical locations and treatment of degenerate cells. Labels F, C and S stand for FULLFLOW, CUT and SOLID respectively and the numbers shown on Cartesian cell centers, represent the combination list that the cell is a member. This cell information is generated by the code for each time step and plotted over automatically on the output file. Intersections of dotted lines are cell centers and solid lines are Cartesian grids.

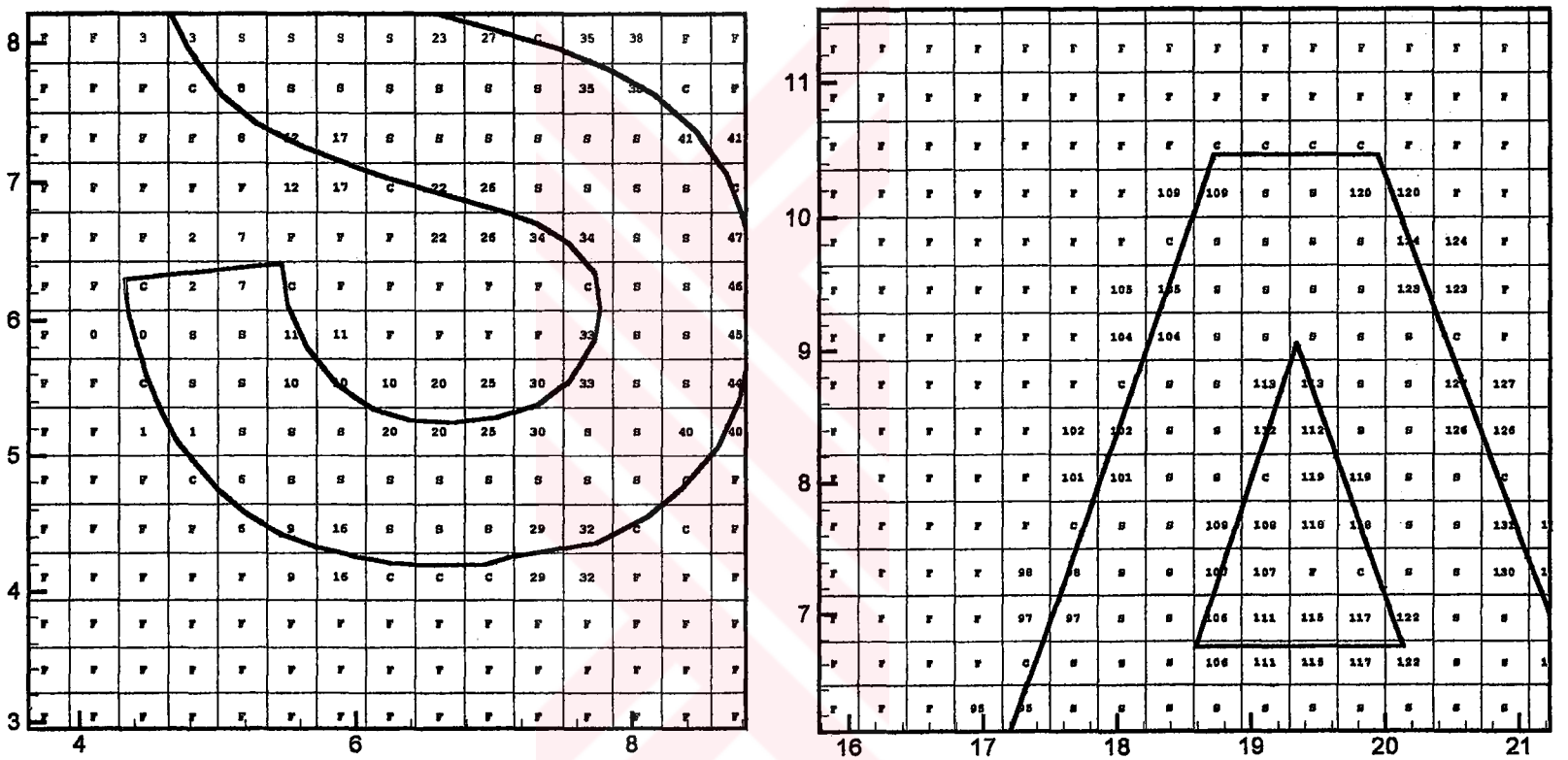


Figure V-11: Cell type and combination information. Lower left corner of letter "S" and top right of letter "A".

V.3.1.11 BOUNDARY OFFSETTING, END TRIMMING/EXTENDING

After the solution at a new time step is obtained, moving curves are offset in the following way; for cells that contain a moving type cut segment, segment center points are displaced in the wall velocity direction. New curve points for the moving wall are generated without loosing the curve parameter sense. Curves that form loops are separately detected and displaced. Moving curve ends are trimmed or extended to a nearest solid wall or a template boundary.

Explanation and geometric details about moving curve trimming/extending algorithms can be found in Appendix C.

At the end of the trimming and extending process, it is established that each moving curve starts from or end at a solid boundary /template side, or form a loop. This then enables the stream generation algorithms that are discussed in Section V.3.1.3, can trace the moving and solid curves without any breaks.

V.3.2 THE SOLVER

In the solver routine, time step size calculation and balance equations for the control volumes of cut and full-flow cells are solved. For conventional solvers, the grid usually consists of cells of one geometry. In Cartesian approach similar to hybrid methods, different element types coexist and the solver must distinguish each type. Moreover cells that are combined must be solved together, and in this study, since orientation and number of combinations is allowed to vary, extra controls are needed to realize this task.

V.3.2.1 DETERMINATION OF TIME STEP

Time step size Δt, that the solution is advanced to the next time level depends on the speed of the fastest wave in the solution domain and cell size. The procedure is discussed in Chapter IV for one-dimensional applications and its extension to higher dimensions is utilized here. For a given Cartesian cell of any type, its sides are traced for the maximum wave speed. For sides that are not aligned with the Cartesian directions, the speed is calculated from the Equation (4.10) of Chapter IV, using the rotated left and right states. Twice the normal distance between the cell center and the cell side is taken as the linear distance, of Equation (4.14) of Chapter IV. The minimum time step is calculated for each cell in the solution domain, and the cell with the minimum value compared to other cells in the domain determines the time step size.

For combined cells, each combination list and each side in the list is traced; the linear distance used in that case is twice the shortest distance between the cell center of the combination list and the cell side that is considered.

V.3.2.2 FLUX TERMS AND BALANCE OF CONSERVATIVE VARIABLES

The integral form of the conservation laws, which are presented in Chapter IV by Equations (4.19) and (4.20), will be repeated here.

$$\frac{d}{dt}\left(\vec{U}|\forall|\right)=\sum_{N\,sides}\int_{A_s}^{A_{s+1}}\left[\cos\theta_s\vec{F}^M(\vec{U})+\sin\theta_s\vec{G}^M(\vec{U})\right]dA \tag{5.1}$$

In which, sides of the control volume $|\forall|$ are inclined to the x-coordinate with arbitrary angle θ_s. By the use of the rotational invariance property, which is discussed in Chapter III, and for a control volume that is not deforming, Equation

(5.1) can be represented in the form given in Equation below. This form of the governing equations is suitable for a x-split method of approach.

$$\frac{d}{dt}\left(\vec{U}\right) = -\frac{1}{|V|}\sum_{N\,sides}\int_{A_s}^{A_{s+1}}\left[T_s^{-1}\vec{F}\left(T_s\vec{U}\right)\right]dA \tag{5.2}$$

T_s is the rotation matrix given in full form in Appendix A and T_s^{-1} is its inverse matrix. $T_s\vec{U}$ is the state vector in the x-split direction. Using this state vector, flux $\vec{F}\left(T_s\vec{U}\right)$ is calculated by a suitable flux stencil, like an exact Riemann solver as in this study. $T_s^{-1}\vec{F}\left(T_s\vec{U}\right)$ is the final flux contribution from side s, that is rotated back to the original direction before summation.

In discritized form, for a two-dimensional planar control volume, with sides s, side length L_s and cell area $A_{i,j}$ the governing equations are;

$$\vec{U}_{i,j}^{n+1} = \vec{U}_{i,j}^{n} - \frac{\Delta t}{A_{i,j}}\sum_{s=1}^{N} L_s\left[T_s^{-1}\vec{F}\left(T_s\vec{U}\right)\right] \tag{5.3}$$

Using Equation (5.3), state vector of cut and full-flow cells is updated at each time step. The flux contributions from cell sides that are aligned with the Cartesian directions are calculated first.

Flux contribution from the cut segment, which may be oriented in any direction is calculated by rotating the state vector of the Cartesian cell to the direction of wall normal. This rotated state will form the left state, of the intercell Riemann problem. Using this left state, moving/stationary wall boundary conditions will be applied and the corresponding fictitious right cell-state is determined. With these left and right states, local Riemann problem will be solved which leads the flux through the cut side. This flux is the component along wall normal and should be rotated back to the original direction before placing it in the balance Equation (5.3).

Cells that are combined are treated as a single control volume. Flux contribution from all sides are calculated via intercell Riemann problems and sides that are common to two cells of the same combination list are ignored.

Details of the algorithm, which is used to calculate fluxes for each cell side is presented in Appendix D.

V.4 MOVING WALL EXAMPLES

To demonstrate the performance of geometry handling and moving wall-offsetting functions two examples are presented. Geometric problems encountered during normal offsetting are discussed in [74]. These are related to the local curvature and offsetting distance. For both examples that are presented wall offset velocity is constant.

Figure V-12 is the burnback [75] of an arbitrary solid propellant grain. Burnback data is the basic input for one-dimensional performance prediction codes of solid propellants rocket motors. This data consists of wetted perimeter and area burned vs. web burned (offset distance) One tenth of the cylindrical shell is plotted. 50x60 Cartesian grids are used in the template of 5/6-length ratio. Due to its high convex curvatures this geometry may cause some problems in geometry dependent burnback codes [76]. Sharp grain corners are tolerably rounded. At the end points of the regressing curve, which represents the moving propellant, to force symmetry, slope is specified.

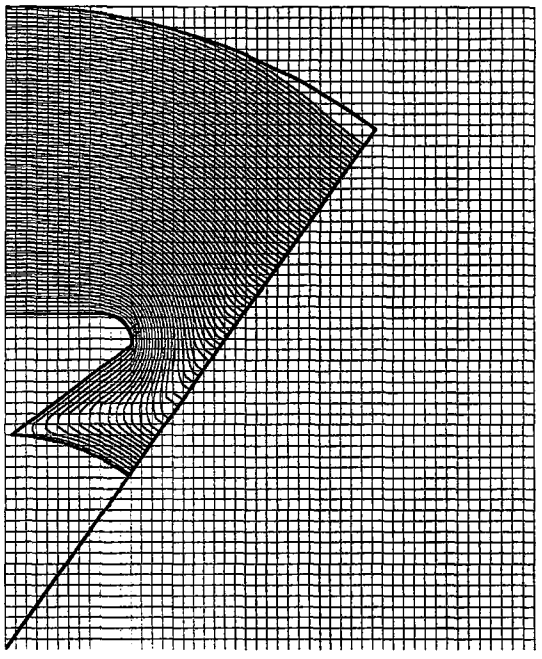


Figure V-12: Grain burnback in a Cartesian template. One of the four consecutive time steps are plotted.

In order to demonstrate the performance of degenerate cells, some simple merging and break up situations can be simulated, no merge/break-up routines are used but the present code detects brake as an integral capability of the algorithm. Break-up region is detected by marking some degenerate cells as full flow. Figure V-13 demonstrates an offsetting sequence leading to break-up. Template is a square 8x8 millimeters. The offset velocity is 2.5mm/s. 46x46 grid is used in the calculation.

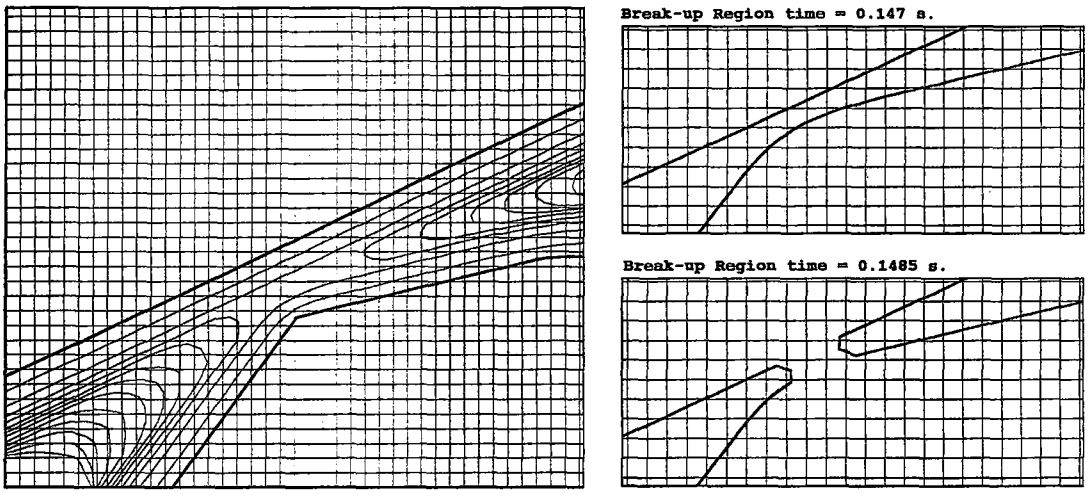


Figure V-13: Break-up of an arbitrary solid body. Demonstrating the degenerate cell handling procedures and stream formation. The initial geometry is drawn using thick lines. Not all the zones are plotted. During break-up, time step is modified.

V.5. APPLICATIONS

V.5.1 2D IMPULSIVE PISTON WITHDRAWAL AND ADVANCE

Gas dynamics of the simple piston problem is a good test case to verify the moving boundary handling routines. The analytic solution obtained via characteristic methods for different piston velocities are available in the literature [69].

In this test case, the chamber length is 10 and width is 4 meters. A relatively high piston velocity, 40 m/s is selected. The moving piston head is placed at the left-hand side (at t=0. s piston is at x=1 m). Open-end boundary condition is specified at right end, all other template boundaries are reflective simulating closed chamber walls. The CFL number is 0.1.

The results are compared in Table V-3 and V-4 for 80x32 and 40x16 grid sizes. For both advanced and withdrawn cases, as the grid size increases results become more close to the corresponding 1D analytic solution. Even for coarse grid, the presented percent errors are acceptable. The sixth column in the table gives the speed in the first cell close to the piston head. These values are not exactly same with the piston speed since they represent the cell-averaged value assigned to the cell center. The sample solutions (density contours) are plotted when the generated wave is approximately at one-third of the solution domain, i.e. before the whole domain reaches to the piston velocity and to piston head state, Figure V-14.

Table V-3: Comparison of results with 1D analytic solution, **piston withdrawn.** (Values of the piston head cell is compared at time = 0.019744 s.)

Grid Size	P3 (Pa)	P3 from 1D (Pa)	Rho3 (kg/m3)	Rho3 from 1D (kg/m3)	U cell (m/s)	% error in Rho (density)
40x16	85897	85907	1.064807	1.066724	-40.0003	0.18
80x32	85900	85907	1.065071	1.066724	-40.0001	0.15

Table V-4: Comparison of results with 1D analytic solution, **piston advanced.** (Values of the piston head cell is compared at time = 0.015269 s.)

Grid Size	P2 (Pa)	P2 from 1D (Pa)	Rho2 (kg/m3)	Rho2 from 1D (kg/m3)	U cell (m/s)	% error in Rho (density)
40x16	118975	118994	1.343465	1.346068	+39.9997	0.19
80x32	118982	118994	1.343916	1.346068	+39.9999	0.16

For comparison density flood plot for a 45° inclined wall that is withdrawn 40m/s impulsively, perpendicular to the piston surface is also presented in Figure V-14. Although template boundaries are transmissive, generated waves are not parallel to the inclined wall because of the unsymmetrical decrease of retarding wall length. In Figure V-15, velocity vectors are plotted for an advancing 45° inclined wall. The turning of velocity vectors at the top corner of the piston is distinguishable.

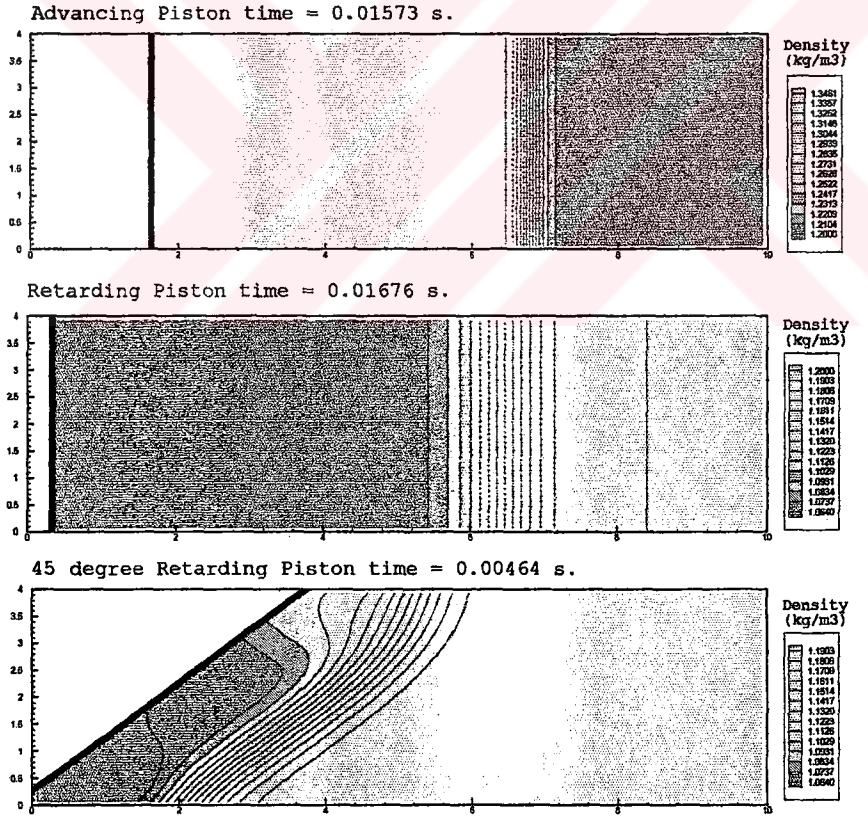


Figure V-14: Density flood plots of impulsively advanced/withdrawn piston for different orientations.

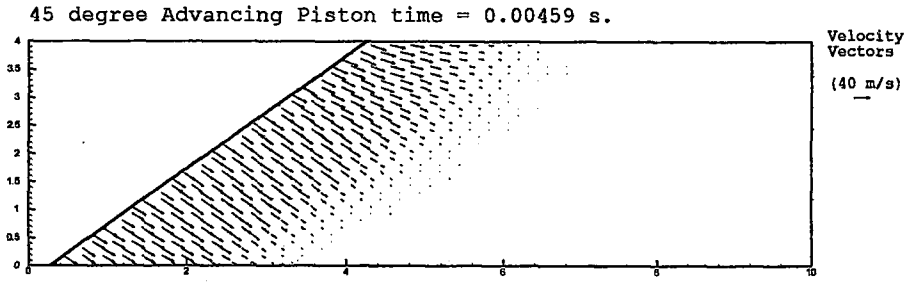


Figure V-15: Velocity vectors for 45 degree inclined advancing piston.

V.5.2 FLOW AROUND THE LETTERS 'S' 'E' 'A'

For the qualitative assessment of the algorithms that has been developed can be accomplished by means of a typical arbitrary complex geometry. In this study the flow around the letters 'S' 'E' 'A' as presented in Figure V-16 and V-17, is selected for this purpose. Letters are occasionally used in CFD literature to typify complex solid geometries [5] [77]. The template grid size is 70x42 and transmissive boundary conditions of Section IV.3.5 are specified for all its sides. A discontinuous initial condition is specified for the flow state. The discontinuity is positioned at X=1 and its left side is high pressure and density. U_L = (12 kg/m3, 0.0 m/s, 1013.0 kPa), U_R = (1.2 kg/m3, 0.0 m/s, 101.3 kPa). All over the solution domain, initial velocity is zero. The cfl number is 0.1 throughout the solution duration, which is 2 s.

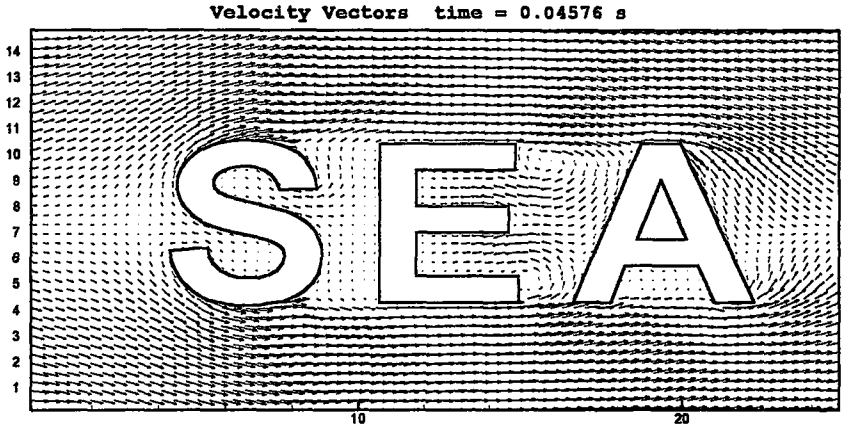


Figure V-16: Velocity vectors around the letters 'S' 'E' 'A'

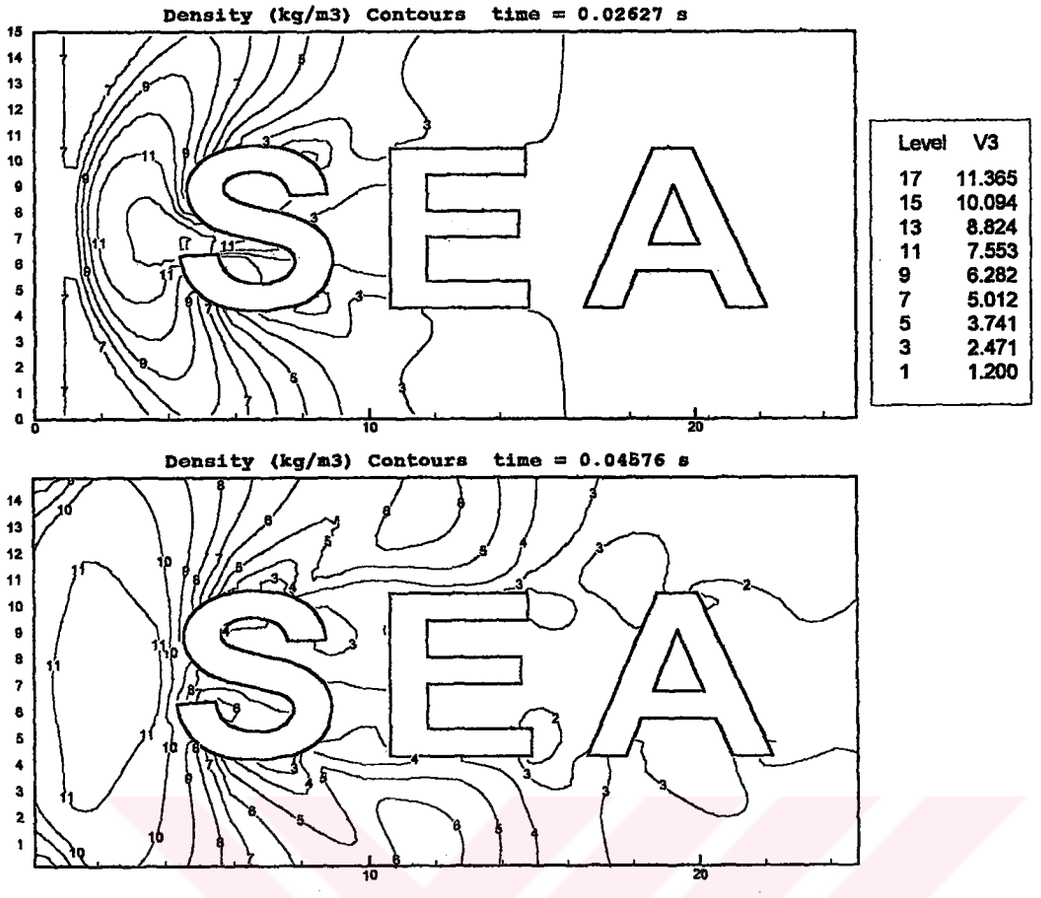


Figure V-17: Density contours at two consecutive times.

CHAPTER VI

AXISYMMETRIC FORMULATION AND SOLID PROPELLANT ROCKET MOTOR SOLUTIONS

VI.1 INTRODUCTION

Axisymmetric formulation is a natural extension of 2D planar codes and essential for many practical problems. Same problems can be solved via the full 3D codes, however this approach becomes impractical especially in a design environment for the following two reasons. The time step limitation, due to small 3D finite volume elements at the center region [78] and large number of control volumes need to be solved due to the extra spatial dimension.

The governing equations, when derived in axisymmetric coordinates, become inhomogeneous with a geometric type source term. This is due to the unbalanced normal pressure in the azimuthal direction. For this reason, the equations, when written for axisymmetric coordinates, cannot be represented in the full conservative form. Alternative forms of the equations are possible, if some part of the geometric source is

covered in flux terms. Besides the discritized equations, the boundary condition treatment, evaluation of flux terms and other practical numerical issues becomes different for each alternative arrangement. In this chapter, the numerical implementation details of three alternative equation forms will be discussed and applied to a steady solid propellant rocket motor internal flow problem.

Since a Jacobian transformation is implicitly involved, both finite difference and finite volume discritizations may have problems in satisfying the Geometric Conservation Law (GCL) [18]. Especially in the r-scaled flux formulation [79], where cell areas implicitly defined in flux calculations may not match with those in the overall discritized balance equation [2]. The free stream test reveals the source term problems and must be conducted for all axisymmetric codes. Another numerical problem of finite difference schemes is the $1/r$ problem at the centerline. This singularity is overcomed by introducing derivatives and L'Hopital rule [80]. $1/r$ Singularity problems are not faced in finite volume schemes.

VI.2 CENTERLINE BOUNDARY CONDITIONS

At the centerline, due to flow symmetry first derivatives of all the primitive variables is zero. Except for radial velocity, which is anti-symmetric at the center, instead of a condition for the first derivative, its second derivative should change sign. Radial velocity becomes zero at the center, since there is no flow through the centerline. Mach number contour plots, is a good evidence for an acceptable centerline treatment since it

involves all four variables. Centerline conditions are similar to a wall boundary condition. Thus for a ghost cell approach, to specify the fictitious cell state, radial velocity is taken with a sign change, and all other variables are zeroth order extrapolated from the boundary cell. Another approach is the direct calculation of flux at the centerline, without using the Riemann solver. Since radial velocity is zero at the center, the radial flux term contains pressure only, which can be interpolated from inside [41]. Both methods have practical advantages depending on the alternative form used.

VI.3. ALTERNATIVE FORMS OF AXISYMMETRIC EULER EQUATIONS

FORMULATION (A)

The first form of the axisymmetric governing equations, which results from direct derivation, is presented in Equation (6.1), including species terms q_i.

$$\vec{U}_t + \vec{F}\left(\vec{U}\right)_x + \vec{G}\left(\vec{U}\right)_y = \vec{S}\left(\vec{U}\right) \tag{6.1}$$

where,

$$\vec{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \\ \rho q_i \end{bmatrix}, \quad \vec{F} = \begin{bmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ u(E+P) \\ \rho u q_i \end{bmatrix}, \quad \vec{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ v(E+P) \\ \rho v q_i \end{bmatrix} \tag{6.2}$$

The source term,

$$\vec{S}(\vec{U}) = -\begin{bmatrix} \rho v/r \\ \rho uv/r \\ \rho v^2/r \\ v(E+P)/r \\ \rho v q_i/r \end{bmatrix} \tag{6.3}$$

In some discretization schemes, this equation structure is prone to the geometric conservation problems since all geometric terms are grouped as a source. Moreover, from numerical point of view, noting that the flux derivations assumes zero source term in the governing equations, loading the source term with many quantities makes our zero source term assumption unrealistic, especially for large radial velocity components. Utilization of source term splitting methods also fails since unlike mass injection or chemical reaction source terms, axisymmetric source terms are geometric in nature and should be discritized synchronously with the flux terms.

However, the form given in Equation (6.1) becomes practically advantageous for control volume methods, especially in internal flow applications, with boundary conditions are specified by the fictitious cell approach. if planner flux stencils and numerical procedures can be used cautiously in axisymmetric problems without any alteration.

FORMULATION (B)

The r-scaled form of the equations can be obtained from Equation (6.1) by scaling conservative variables by the radial coordinate.

$$\frac{\partial y\vec{U}}{\partial t} + \frac{\partial y\vec{F}(\vec{U})}{\partial x} + \frac{\partial y\vec{G}(\vec{U})}{\partial y} = \vec{S}(\vec{U}) \tag{6.4}$$

where,

$$\vec{S}(\vec{U}) = \begin{bmatrix} 0 \\ 0 \\ P \\ 0 \\ 0 \end{bmatrix} \tag{6.5}$$

Although Equation (6.4) is still not completely divergence free due to the pressure term, only source term is in the radial momentum equation. By scaling the conserved variables and fluxes as shown below, same flux stencils can be used [81] [65].

$$\vec{\bar{U}}_t + \vec{\bar{F}}\left(\vec{\bar{U}}\right)_r + \vec{\bar{G}}\left(\vec{\bar{U}}\right)_z = \vec{\bar{S}}\left(\vec{\bar{U}}\right) \tag{6.6}$$

$$\bar{U} = rU, \bar{F} = rF, \bar{G} = rG \tag{6.7}$$

The incompatibility between the geometric properties implicitly defined in the flux calculations and in the overall volumetric balance still exists in this formulation. A remedy is to apply the corrective quantities suggested in [79].

FORMULATION (C)

In this approach, to overcome the drawback stated in Formulation (B), areas and volumes are found from the geometric relations, both in the overall balance and flux calculations. No flux scaling is made but areas associated with the fluxes are taken into account as suggested by [82]. No problems associated with the violation GCL are observed and divergence that is sometimes faced in the above formulations has been disappeared.

In the program developed, the volume of a single cell is calculated according to the first theorem of Pappus-Guldinus, which states that the volume of a body of revolution is equal to the generating area times the distance traveled by the centroid of the area while the body is being generated.

$$\forall_{cell} = 2\pi \hat{y} A \tag{6.8}$$

$\hat{y}$ is the radial coordinate of the centroid of the area A and for quadrilateral cells it is calculated by averaging the radial coordinates of four corners. The surface areas associated with the flux terms are calculated using the second theorem of Pappus-Guldinus, which is given in Equation (6.9).

$$B = \pi (y_1 + y_2) \sqrt{(x_2 - x_1)^2 - (y_2 - y_1)^2} \tag{6.9}$$

Coordinates x and y defines the generating 2D line segment where the fluxes are passing through the generated lateral area.

VI.4 DISCUSSION ON THE DIFFERENT FORMULATIONS

All the three formulations are applied and studied in the context of the developed Cartesian grid method and solver. Since the solution method used in this thesis is an unstructured method in principle, most of the experience that follows may also be applicable to unstructured finite volume solvers.

To preserve the accuracy of the developed solution method, which is due to the usage of an exact Riemann solver in flux calculations, Formulation (B) is selected first. Each state vector is scaled by the radial coordinate of the corresponding cell and the solution is

performed for the new variables. For combined cells, the coordinates of the combined cell center is used. This scaling produced a linear variation in the all the state variables along the radial direction, even though the initial condition of the problem is uniform. This y-scaled initial condition when solved produced an unrealistic flow away from the center. Although all of this flow must be canceled by discritized source term given in (6.5), at the balance step, some residual remained. The magnitude of this residual was tolerable for the interior cells of the solution domain, however at the center and near curved boundaries its magnitude is large enough to deteriorate the desired solution.

The reason for this excess flow is due to the difference between the y-values used in the calculation of fluxes and cell volume. When control volume size approaches zero Equation (6.4) becomes exact, since the y-scale used for all flux terms and radial coordinate of the cell center that defines the control volume becomes identical. However for finite control volumes, fluxes are scaled using the center points of the cell sides and cell volume is defined by the radial coordinate of the of the cell vertex.

Specification of boundary conditions both for wall and center becomes impractical for this formulation. In curved wall, which is modeled as cut segments in Cartesian cells, the calculation of y-values for the fictitious cells are needed. Since the wall boundary conditions are applied in the x-split direction, several rotations and back rotations is required. The combined cell at near wall region also complicates the task considerably. Moreover, since in this study wall boundary movement is also possible, this formulation requires the storage of y-values for the extra solutions generated by retarding walls.

Although for each of these problems a solution is proposed and tested in the application considered. An efficient and acceptable solution was not possible.

At centerline, where the geometric conservation error was the most apparent, the y-scale value for the fictitious cell, which is needed to reflect radial velocity, is undefined. Use of the negative radial coordinate produces negative pressure and density, which are assigned to the fictitious left state. For this reason, when this form is used the center-state is interpolated from inside, similar to a finite difference treatment of the center. Cells neighboring the centerline are not solved and the interpolated values weakly imitate the exact boundary behavior.

Formulation (C), uses the exact areas for the flux terms, from purely geometric formulas. Same areas are used in the definition of the control volume. This approach has solved all the geometric conservation problems that are faced in Formulation (B). However it must be noted that radial flux terms are passing through curved surfaces, although they are generally derived for planner surfaces. For the case of the Riemann solver, to calculate exact flux terms, the Riemann invariants that are derived for cylindrical coordinates should be used. In that case, the Godunov method looses its advantage since Riemann solution will now provide the discrete values along the radial direction of the control volume, but not the flux values through the cell boundaries as for the planar control volumes. These discrete states must be averaged for a wedge shape control volume at each time step. Reference [83] gives extension of Godunov method in radial direction, for one-dimensional formulation. Even the flux terms are calculated

exactly via the proposed method, a practical problem at the center will still exists. At the center, the flow area, for which centerline flux should pass is zero, as the exact areas are used. Thus centerline boundary condition could not be specified via flux. The approach in the previous formulation could be used. However, at center geometric conservation is effected and a weak boundary condition is specified at best. The result of weak boundary conditions is the severe cusps in flow variables near the center.

Finally, formulation (A) is exercised in the steady motor test problem. These forms of equations are utilized in literature in [65] for applications to internal flows. Also following the opinion given in [84], where time step effects the accuracy, the desire to use exact flux terms are suspended. This formulation brought numerous advantages. The most important is the specification of the exact boundary conditions both at wall boundaries and at the center via the fictitious cell approach, which is very practical for Cartesian grids. Also the moving boundary algorithms developed for planner problems are exactly used. The only change is that the states are averaged by volumes instead of areas.

VI.5 APPLICATIONS

VI.5.1 2D AXISYMMETRIC SHOCK-TUBE

This test case is used to check the discritization method for the free stream reproduction condition when the solution contains no solid body in the solution domain. Besides this basic task, it also demonstrates the accuracy of flux calculations in x-direction in a cylindrically symmetric geometry. For the complete treatment and finer details of the

flow in cylindrical shock tubes, including viscosity and thermal effects associated with the inhomogeneity of the test gas in the shock tube refer to [85].

The results presented are obtained with the source vector of Formulation (C) where cell areas in flux terms are calculated directly from geometry without any conservative variable scaling. Initial shock-tube pressure and density ratios are 5/1. Grid size is 160x16. CFL number is 0.02. Centerline passes from y=0.

In Figure VI-1. density contours are plotted for different solution times. The density, pressure and velocity variations along an arbitrary horizontal line are compared with the exact 1D solution, Section IV.5.1. Discontinuities of contact surface and shock wave are distinguishable. The axial variation is similar to the computed 1D solution of IV.5.1.

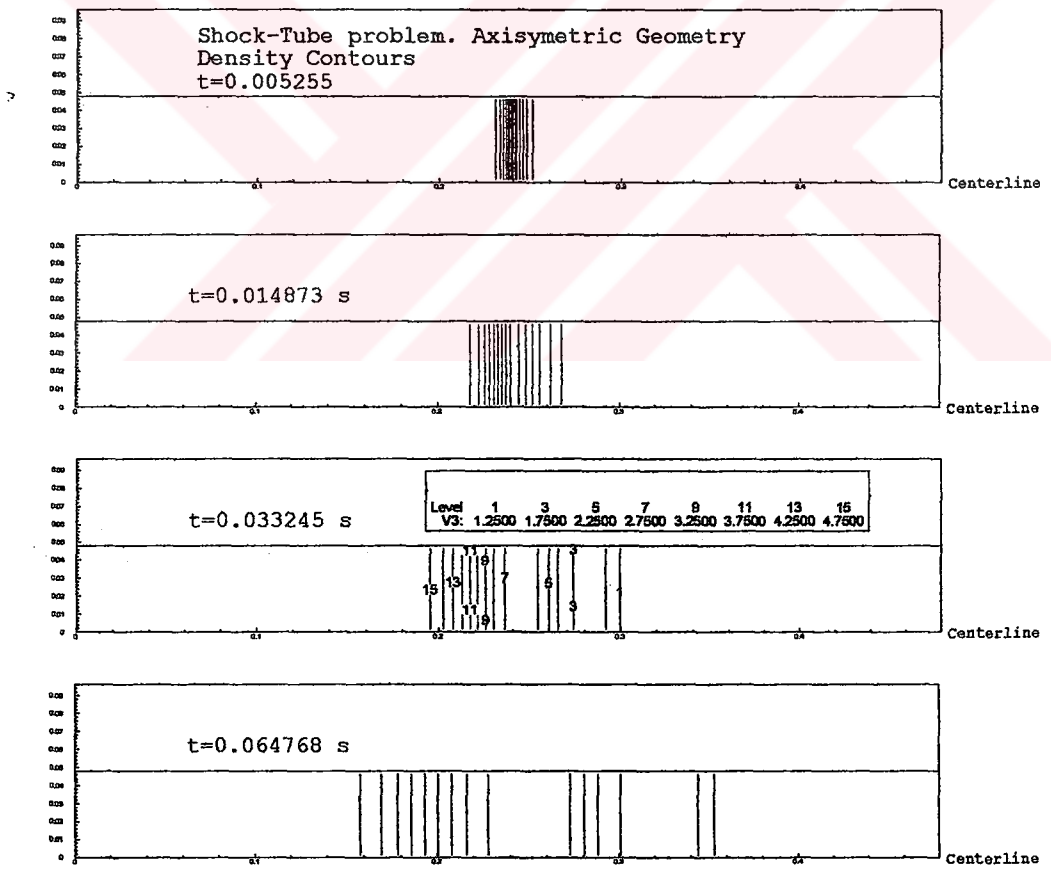


Figure VI-1: Density Contours in a Cylindrical Shock-Tube. Initial State Ratio is 5.

VI.5.2 2D MOTOR TEST CASE: (Steady End-Burning Motor of ONERA [65])

A test case is suggested by [78], for validating codes that are developed solely for the prediction of internal flows in solid propellant rocket motors. The main characteristic of the internal flow in solid propellant rocket motors is the co- existence of very low and very high Mach numbers in the same problem. Mach number ranges from 0.07 to 5., where chamber flow is pressure and nozzle flow is density driven. For this reason, to capture flow details accurately the convergence criteria, in the p^* calculation routine in section III.2.2 of the Riemann solver is decreased.

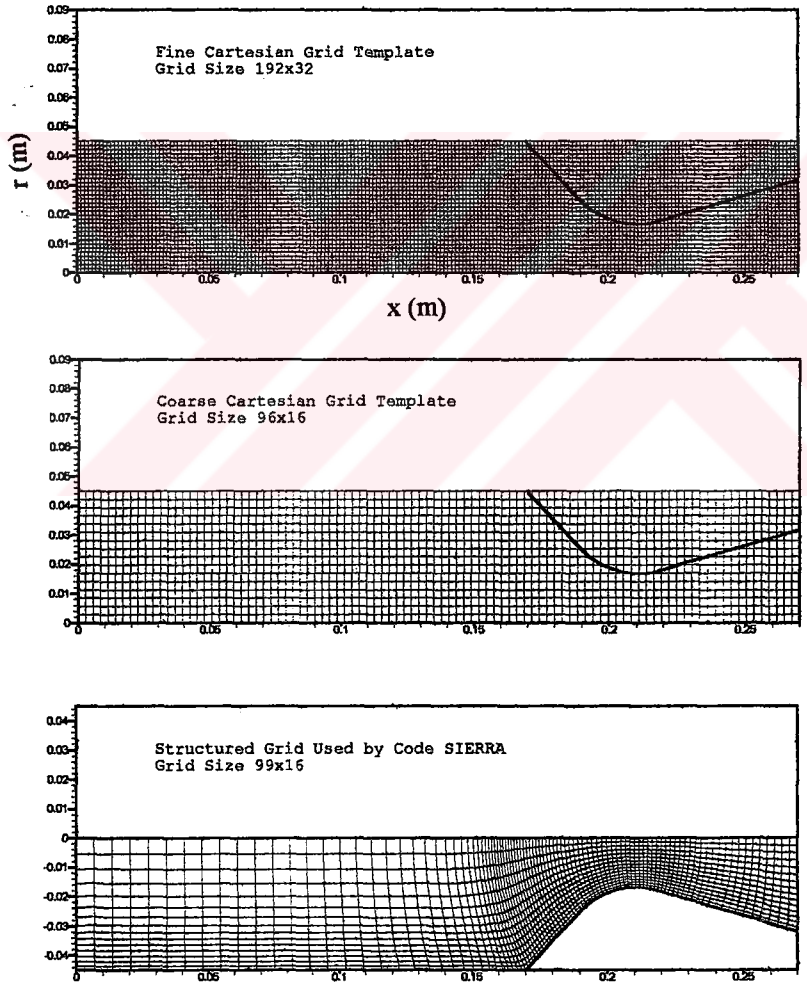


Figure VI-2: Test case grid and Cartesian templates.

Grain geometry of the test case is end-burning. Propellant in the gas phase is injected from the head-end. Results that are obtained by the Cartesian grid solver will be compared, with the converged results of the code SIERRA, which was available in the context of the project [86].

The SIERRA ONERA/FRANCE code is a research code dedicated to internal flows inside solid rocket motors as well as to unsteady regimes that can develop on the motor acoustic modes. It is based on a cell centered, finite volume, explicit predictor-corrector Mac Cormack's scheme, in its original unsplit form (1969). Artificial viscosity can be added in the form of second and fourth order terms that are adapted to the flow by sensors reacting to the pressure and/or the velocity fields (Jameson's type). The code is multi-block and can be applied to a variety of geometries. Several boundary conditions are available and can be chosen by the user.

Besides the differences in the solver and numerical scheme, for this particular problem the grid methodology in SIERRA is completely different compared to the present work. The Cartesian grid results are obtained for two different grid sizes. For comparison grids are plotted in Figure VI-2. The test solution is obtained using a high quality grid, which is continuously refined near the boundaries. In the Cartesian grid method that is employed, nozzle wall region is composed of cut cells. Since many of them are combined cells the effective grid size near the wall is decreased further. Grid size close to wall is important since the streamlines that pass near the nozzle boundaries are being

originated from the stagnation region, which is located at the upstream of the nozzle, away from the flow core.

Table VI-1: Grid Sizes.

Feature	Grid Size	Number of Radial Grids	Number of Grids at the Throat
Structured, Body-Fitted	99x16	16	16
Cartesian, Fine Mesh	192x32	32	12
Cartesian, Coarse Mesh	96x16	16	6

In Table VI-1, grid sizes of the three solutions are presented. The number and density of grids at the throat of the nozzle is also important for the quality of the solution. In fine grids, Cartesian method generates 12 of 32 radial cells, at the throat. whereas SIERRA solution utilizes all 16 grids at the throat due to the structured body fitted feature.

BOUNDARY CONDITION

The head-end injection boundary condition used in the SIERRA solution is different then the propellant injection boundary condition developed in this study, which is discussed in Chapter IV, for 1D applications. The main difference is that in Cartesian grids the injected propellant is introduced to the solution domain as a mass source from the cell center, whereas in SIERRA, injecting gas is specified as a subsonic inflow condition from the head end boundary. In both solutions the energy equation is used to calculate ρ_{inj}, the injection density. which depends on injection mass flux, flame temperature of the propellant, specific heat ratio, gas constant and pressure. The pressure of the injecting gas is assumed to be same with the static pressure of the boundary cell.

Dividing the injecting mass flux by the injection density gives the injection velocity. This velocity is in the axial direction for an end-burning motor.

Supersonic outflow boundary conditions are used. For the fictitious cells at the nozzle exit all variables are first order extrapolated from inside.

CONTOUR PLOTS

The contour plots for Mach number, pressure and density are given in Figures VI-3, VI-4 and VI-5 respectively. They are presented together with the corresponding solutions of the code SIERRA of ONERA/FRANCE.

From the Mach number plot, the acceleration of the low subsonic inlet flow is clearly visible, where near the throat it reaches to the sonic speed. Waves initiated along the wall just down stream of the throat and they propagate toward the axis of symmetry. This is a characteristic of a typical nozzle flow [87]. The difference in the Mach contours at the throat entrance is associated to the small differences in the nozzle contour used in the calculation. Except the nozzle entrance, stagnation region the behavior of all the variables are similar.

Using the relations of isentropic flow of a perfect gas, stagnation pressure and density, inlet and exit Mach numbers can be calculated for comparison. Mach numbers depend

on the inlet and exit area ratio and are useful for the initial verification of the numerical nozzle solution. In Table VI-2, values obtained, for these variables by various test runs are listed, together with the corresponding analytical solution.

The IBSE2D code in Table VI-2 uses a cell vertex, finite difference, explicit multistage Runge-Kutta solution scheme based on the work of Jameson, Schmidt, and Turkel [86].

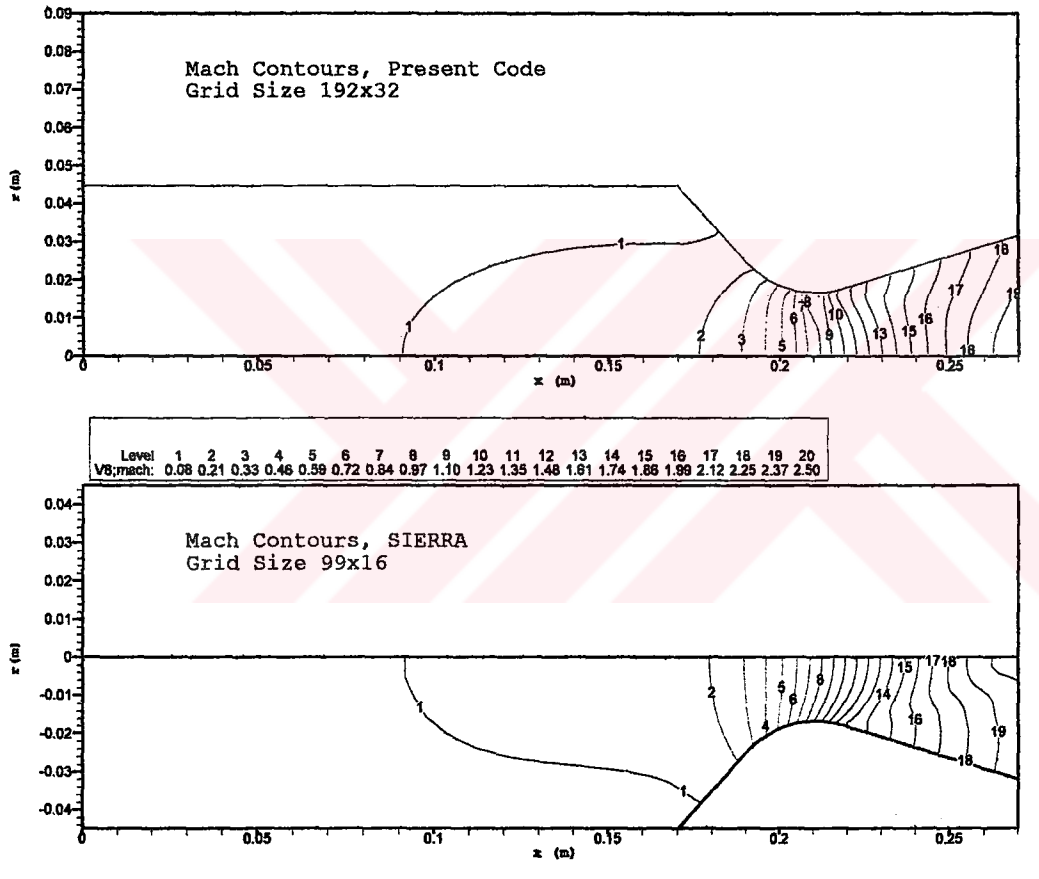


Figure VI-3: Mach contours obtained for the motor test case.

Table VI-2: Comparisons of different axisymmetric codes with the analytically obtained values.

	Inlet Mach Number (center)	Exit Mach Number (center)	Stagnation Pressure (Pa)	Stagnation Density (kg/m3)
Isentropic[88]	0.08314	2.460	129561.	0.12772
SIERRA[65]	0.0828	2.456	129048.	0.12728
IBSE2D[86]	0.0829	2.476	129470.	0.12770
2D Cartesian, Fine Grid	0.0807	2.465	131443	0.12954
2D Cartesian, Coarse Grid	0.0792	2.385	136331	0.13440

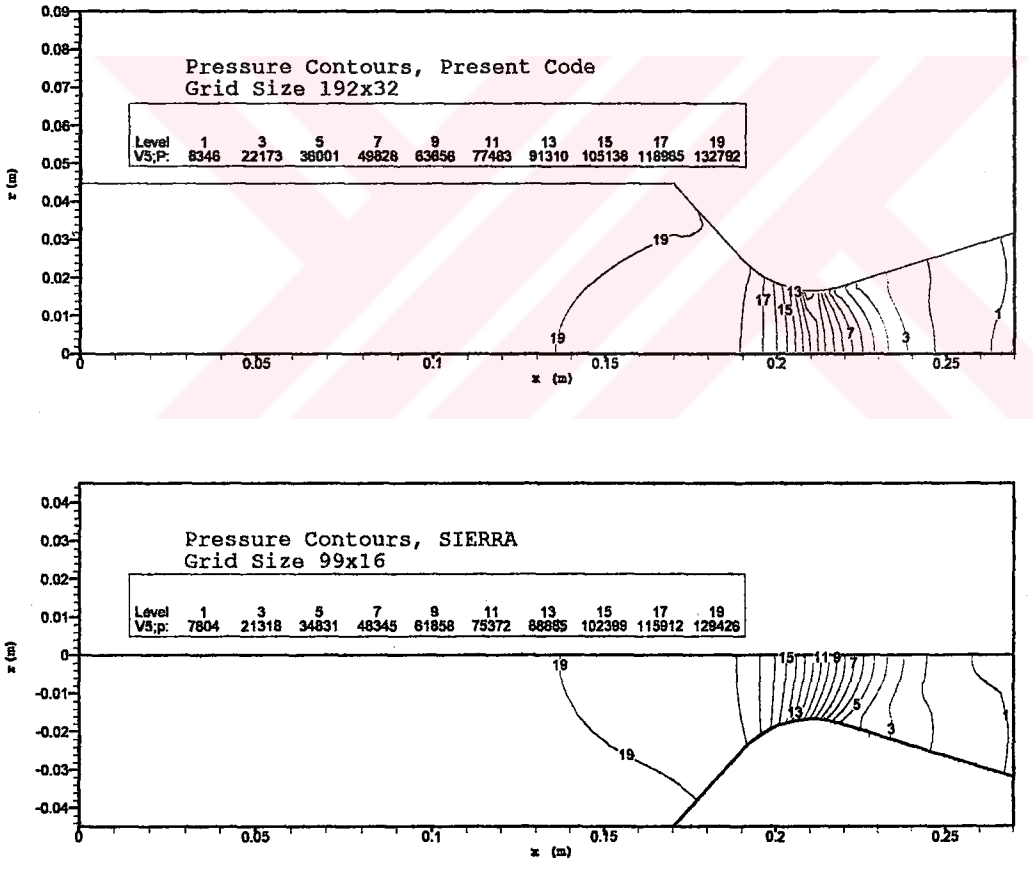


Figure VI-4: Pressure contours for the motor test case.

As seen in Table VI-2, stagnation pressure and density values are larger than the corresponding values of isentropic flow by the two axisymmetric codes. However it is also seen that as the grid size increases both parameters approach to their isentropic flow values.

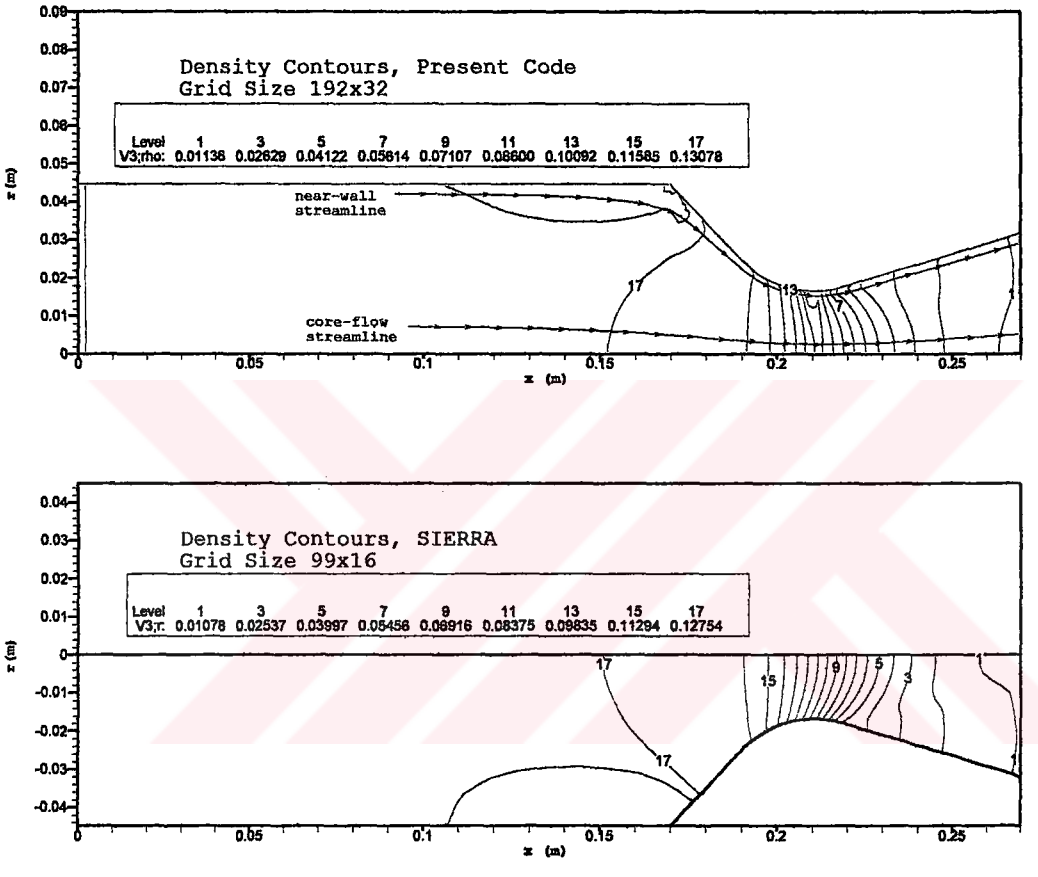


Figure VI-5: Density contours and two streamlines of Cartesian grid solution.

From the contour plots of pressure and density, the shape variance in the nozzle inlet stagnation region is evident. The flow behavior in this low subsonic region is directly related to the near wall nozzle inlet flow. As demonstrated in Figure VI-5 by the streamline emanating from this region. The entire nozzle near wall streamlines pass from

this region before reaching to the nozzle inlet and due to the low subsonic character of the flow, accuracy in the nozzle wall, effects the accuracy in the upstream stagnation region.

The grid quality of Cartesian cells near the nozzle wall due to arbitrary cut and combined cells is evident and discussed previously in detail. As the grid size decreases the region of irregular cells that neighbors the nozzle wall boundaries also decrease which results a corresponding decrease in the size of the upstream region effected.

The two codes IBSE2D and SIERRA, that their results are compared in Table VI-2 are specially designed for internal motor applications. Moreover, exactly the same grid is used in the solutions obtained by them. The grid they utilize is very adequate for nozzle solutions, as this issue is discussed in the previous sections in detail.

Another difference between the methods compared in Table VI-2, is the nozzle contour geometry. In Cartesian grids, the nozzle geometry is generated from the discrete data that is obtained manually. The resulting nozzle contour is linear at the throat along approximately three cell sizes for coarse grids, where as SIERRA nozzle has no linear region. The minimum throat diameter exists at a unique x- location. Due to this constant area cylindrical region, nozzle modeled by the Cartesian grids behaves like a blast-tube, where Mach number becomes unity, by a short delay, at the end of this linear region, from then on, nozzle area starts changing again. This throat shape difference may also contribute to the contour plots obtained and effects the solution.

Mach number values are acceptable and close to the isentropic solution when fine grids are used. This is also evident from the contour plots that are given in the previous section.

CHAPTER VII

CONCLUSION AND FUTURE STUDIES

Moving boundary problems, in Cartesian grids and in solid propellant motor internal flow predictions, is the fundamental subject, where this thesis contributes. Three computer codes are developed. A Riemann solver, quasi-1D moving boundary transient solver and Planner/axisymmetric 2D time dependent moving boundary Cartesian grid solver. Splitting methods to handle source terms and multidimensional terms are applied to unstructured control volumes of Cartesian grids.

Chronologically, the exact Riemann solver is the first output of this study. By itself this 1D solver can generate exact analytical solutions for any given discontinuous initial states. These results can be used for comparing accuracy and performance of numerical schemes that may be designed in the future. The Riemann solver assumes perfect gas with constant specific heats in the equation of state. A modification to the equation of state is possible. However complete derivations will be needed for functions that govern the wave structure.

A large number of one-dimensional applications are studied with the quasi 1D moving boundary code. Finally, it is shown that it can be used to predict internal transient flows in solid propellant motors with moving boundaries. Fine details of the transient behavior are predicted especially at the start-up period of solid propellant rocket motors. Subroutines of the exact Riemann solver are utilized in the context of the Godunov method. Since Godunov method is fundamental to current flux vector difference and flux vector splitting schemes both from the coding and physical point of view, higher order numerical schemes can be implemented easily in future extensions.

Detailed boundary clipping and approximation algorithms are developed so that fast and slowly moving end boundaries can be handled at the same time. Transient pressure increase in a closed vessel is studied. These types of problems can be increased in number and interesting comparisons are possible in thermodynamics. In this context an example problem that can be attacked is the calculation of piston speed for a freely retarding piston where forces acting on the piston is due to the pressure and piston weight.

Although quasi-one dimensional transient code assumes a general source term vector with many terms for possible future applications, in this thesis only mass injection and spatial area variation is demonstrated. One future application area is the combustion instability [47], where to the authors knowledge, in literature, there is no study with moving boundaries yet. Due to the efficiency and satisfactory results of the time dependent one-dimensional code, investigations in this area are possible.

Chemical reactions can be included further since the species terms that are kept in the formulation of the solver can also be utilized in such a study.

Propellant regressing speed is coupled to the flow variables by a suitable model and its time evolution can be predicted numerically. Otherwise specific non-intrusive experiments are needed to measure the regression speed. The present numerical approach is relatively new for solid propellant rocket motor research. In this work, implementation of coupled solution was the priority; therefore injection model is selected to be as simple as possible. More detailed models in any complexity are available in the literature.

Besides the transient solutions, steady applications are possible which are demonstrated by the nozzle test cases. Features related to convergence should be included to the code if steady applications are aimed.

In higher-dimensions, a practical moving boundary 2D Euler code is put to use that is free from any classical meshing considerations. Usage of Cartesian grids is a topic of growing popularity. Moreover, studies involving moving boundary problems in Cartesian grids are relatively few and limited in complexity, which adds to the originality of this study. A systematic degenerate cell consideration is accomplished since moving complex geometries are covered by this thesis.

The governing equations are derived for deforming control volumes in the form suitable for a two-dimensional Cartesian grid solver. By this derivation and due to the nature of x-splitting, problems involving only translating and offsetting boundary movement is allowed.

Some issues of moving boundary problems in fixed Cartesian cells are studied. Each requirement, such as stream generation, solid cell marking and cell combination procedures are extended as much as possible to reduce the number of unambiguous geometries. The geometrical problems and moving boundary-handling procedures are presented with proposed remedies in the domain of the available finite volume solver.

Apart from the Euler solution, the code also generates burnback data (Burn perimeter and port area) for some grain geometries that are difficult to burn with the available burnback tools. Burn perimeter and port area data is required as input for internal balistic flow prediction programs of solid propellant rocket motors. Generation of burnback data is a special topic by itself. In this study its possibility is demonstrated using the already available algorithms that are written originally for Cartesian grid methodology. If required, results can be made more accurate using extra algorithms that are developed solely for grain burnback. Another area of future study is the three dimensional burnback. Consecutive two-dimensional grain sections placed in Cartesian templates may offer a practical solution method.

Adaptive mesh refinement feature worth the fairly involved coding work. Different data structures can be utilized. Mesh refinement should be performed at boundaries and increase spatial resolution of shock waves. Boundary mesh refinement is required if a viscous solution is desired or Cartesian grids are utilized in a hybrid grid structure.

The axisymmetric extension of the developed two-dimensional planer Cartesian grid method has been performed for three discretized formulations of the governing equations. One of the formulations is found to work well for the selected internal flow application. Centerline treatments differ and geometric conservation problems appeared in the remaining formulations. Two axisymmetric test cases are studied. The last test case is a steady end burning solid propellant rocket motor. Due to the rectangular template many Cartesian cells are marked as solid for nozzles with narrow throat diameters. Solid cells allocate memory although for those cells no flow solution is performed.

Another point that should be mentioned about the Cartesian grid method is that in its current state curved boundaries are approximated as line segments whose length is approximately same as the nominal cell size. These line segments produce non-smooth solution boundaries. Whereas in conventional structured grid solvers boundary geometries are continuous up to their second derivative. One remedy to this problem is to store cut cell curvature besides the other cell attributes and to modify the solver accordingly.

Similar to the species terms, the third spatial direction is kept in the formulation in all the routines of the developed solver, for this reason the exact Riemann solver can be tried in a three dimensional problem with extra coding work. The solver being a finite volume solver can also be applied to any type of unstructured grids.

Compressible viscous flows are another direction of extension. Although extra modeling related to the physics of viscous flow is needed, treating diffusive viscous terms as source terms, and applying the source term splitting methods that are designed and already available in this thesis can initiate a study in this area.

REFERENCES

[1] Baran Ö.U., Experimental and numerical analysis of transient liquid slug motion in a voided line. M.Sc. Thesis. Department of Civil Engineering. Middle East Technical University. Turkiye 1999

[2] Vinokur M., "An Analysis of Finite Difference and Finite Volume Formulations of Conservation Laws". Journal of Computational Physics. 1989;(81): 1-52

[3] Venkatakrishnan V., "Perspective on Unstructured Grid Flow Solvers". AIAA Journal. 1996;(34): 533-547

[4] DeZeeuw D., Powell K., "An adaptively refined Cartesian mesh solver for the Euler equations", AIAA Paper No. 91-1542, 1991

[5] Quirk J. J., An alternative to unstructured grids for computing gas dynamic flows around arbitrarily complex two-dimensional bodies, Computers Fluids 1994;23(1):125-142

[6] Lin W., Chen C.J., Automatic grid generation of complex geometries in Cartesian co-ordinates., International Journal for Numerical Methods in Fluids 1998;28:1303-1324

[7] Barth T.J., Deconinck, editors. High-Order Methods for Computational Physics. Lecture Notes in Computational Science and Engineering. RTO. Springer, 1999

[8] Baysal O. Yen G. Kinematic domain decomposition to simulate flows past moving objects. 29th Aerospace Sciences Meeting. AIAA-91-0725. Nevada, 1991

[9] Tuncer I.H., Two-dimensional unsteady Navier-Stokes solution method with moving overset grids. AIAA Journal 1997;35(3):471-476

[10] Wang Z.J., "A fast nested multi-grid viscous solver for adaptive Cartesian/Quad grids" International Journal for Numerical Methods in Fluids 2000;(33):657-680

[11] Fenno C.C. Newman P.A. Hassan HA. Unsteady viscous-invicid interaction procedures for transonic airfoils using Cartesian grids. Journal of Aircraft 1989;26(8):723-730

[12] Coirier WJ., Powell KG., Solution-adaptive Cartesian cell approach for viscous and invicid flows, AIAA Journal 1996;34(5):938-945

[13] Epstein B., Luntz A.L., Nachshon A., Cartesian Euler method for arbitrary aircraft configurations, AIAA Journal 1992;30(3):679-685

[14] Milos F.S., Rasky D.J., "Review of numerical procedures for computational surface thermochemistry". Journal of Thermophysics and Heat Transfer. 1994;(8): 24-34

[15] Shyy W., Udaykumar H.S., Rao M.M., Smith R.W., Computational Fluid Dynamics with Moving Boundaries, Taylor-Francis, 1996

[16] Koren B. Venis A.C.J., A fed back level-set method for moving material-void interfaces. Journal of Computational and Applied Mathematics 1999;101:131-152

[17] Kurbatskii K.A., Tam C.K.W., Cartesian boundary treatment of curved walls for higher-order computational aeroacoustics schemes. AIAA Journal 1997;35(1):133-140

[18] Thomas P.D., Lombard C.K., Geometric conservation law and its applications to flow computations on moving grids. AIAA Journal 1978;17(10):1030-1037

[19] Navaz H.K., Berg R.M., Formulation of Navier-Stokes equations for moving grid and boundary. Journal of Propulsion and Power 1998;15(2):360-362

[20] Lupoglazoff N., Vuillot F., "Two-Dimensional Numerical Simulation of the Stability of a Solid Propellant Rocket Motor", AIAA paper, No: 91-0205, Reno, January 1991

[21] Lupoglazoff N., Vuillot F., "Numerical Computation of Acoustic Boundary Layers in Large Solid Propellant Space Booster", AIAA paper, No: 91-0206, Reno, January 1991

[22] Schley C. A., Hagemann G., Krülle G., Towards an Optimal Concept for Numerical Codes Simulating Thrust Chamber Processes in High Pressure Chemical Propulsion Systems. *Aerospace Science and Technology*, No. 3, 1997, pp. 203-213.

[23] Tissier, P. Y., Godfroy, F., Jacquemin, P., "Simulation of Three Dimensional Flows Inside Solid Propellant Rocket Motors Using a Second Order Finite Volume Method Application to the Study of Unstable Phenomena," AIAA Paper 92-3275, July, 1992.

[24] Yang, V., Hsieh, K., Tseng, J. I. S., "Velocity-Coupled Flow Oscilations in a Simulated Solid-Propellant Rocket Environment," AIAA Paper ,

[25] Majumdar, A. K., Whitesides, R. H., Baccus, D. L., Jenkins, S. L., "Circumferential Flow Analysis at the Aft Field Joint of the Space Shuttle Solid Rocket Motor," AIAA Paper , 1988.

[26] Sabnis, J. S., Gibeling, H. J., McDonald, H., "Navier-Stokes Analysis of Two- and Three- Dimensional Flow Field in Solid Rocket Motors with Segmented Joints," AIAA Paper 87-1804, Dec., 1987

[27] Sabnis, J. S., Gibeling, H. J., McDonald, H., "Calculation of Solid Propellant Rocket Motor Internal Flow Field Using an Implicit Navier-Stokes Procedure," AIAA Paper 85-4625, 1985.

[28] Hsieh, K. C., Yang, V., Tseng, I. S., "Navier-Stokes Calculation of Solid-Propellant Rocket Motor Internal Flowfields," AIAA Paper 88-3182, July, 1988.

[29] Dunlap R. et. al., Internal Flow Field Studies in a Simulated Cylindrical Port Rocket Chamber. Journal of Propulsion and Power, 1990;(6): 690-704

[30] Culick F. E. C., Rotational Axisymmetric Mean Flow and Damping of Acoustic Waves in Solid Propellant Rocket Motors. AIAA Journal, 1966;(4): 1462-1464

[31] Dunnap R., Willoughby P.G. Flow Field in the Combustion Chamber of a Solid Propellant Rocket Motor. AIAA Journal, 1974;(12): 1440-1443

[32] Balakrishnan.G, Linan A., Williams A Rotational Inviscid Flow in Laterally Burning Solid-Propellant Rocket Motors. Journal of Propulsion and Power, 1992;(8): 1167-1176

[33] Liou T., Lien W., Numerical Simulation of Injection-Driven Flows in a Two-Dimensional Nozzleless Solid-Rocket Motor. Journal of Propulsion and Power, 1995;(11): 600-606

[34] Beddini R., A, Injection-Induced Flows in Porous-Walled Ducts. AIAA Journal, 1986;(24): 1766-1773

[35] Karimian S.M.H., Amoli A., Two-dimensional simulation of SRM internal balistics on a moving grid. 35th AIAA/ASME/SAE/ASEE Joint Propulsion Conference. Los Angeles, 1999

[36] Brenton P., Ribéreau D., Godfroy F., SRM performance analysis by coupling bidimensional surface burnback and pressure field computations. 34th AIAA/ASME/SAE/ASEE Joint Propulsion Conference. Cleveland, 1998

[37] Eberle A., Rizzi A., Hirschel E.H., Numerical Solutions of the Euler Equations for Steady Flow Problems, Vieweg, 1992

[38] LeVeque R.J, Numerical Methods for Conservation Laws, Brikhauser Verlag, 1992

[39] Toro E.F., Riemann Solvers and Numerical Methods for Fluid Dynamics, Springer, 1997

[40] Godunov S. K., "A Finite Difference Method for the Computation of Discontinuous Solutions of the Equations of Fluid Dynamics" Mat. Sb., 47:357-393, 1959

[41] Laney C.B., Computational Gas Dynamics, Cambridge University Press, 1998

[42] Toro E.F., A Fast Riemann Solver with Constant Covolume Applied to the Random Choice Method. International Journal for Numerical Methods in Fluids. 1989;(9):1145-1163

[43] Courant R., Friedrichs K.O., Supersonic flow and shock waves. Springer-Verlag, 1985

[44] Culick F.E.C., Some recent results for nonlinear acoustics in combustion chambers, AIAA Journal 1994;32(1)

[45] Mortenson ME., Geometric Modelling,Willey, 1985

[46] Arora R, Wu X., White F.X., Kuo K.K, Erosive Burning of Composite Solid Propellants: Mechanism, Correlation, and Grain Design Applications. Journal of Spacecrafts and Rockets, 1983;20:43-48

[47] Turchi P.J. eds. , Propulsion techniques: action and reaction, Torda P.T.,Notes on problems in combustion instability of rocket motors, AIAA, 1998

[48] Gordon S., McBride B.J., Computer program for the calculation of complex chemical equilibrium compositions, Rocket performance, Incident and reflected shocks, and Chapman-Jougnet detonations., NASA SP-273, 1971

[49] Brinkley S. R., Calculation of the equilibrium composition of systems of many constituents., Journal of Chemical Physics.,1947;15(2):107-111

[50] Reddy J.N., Rasmussen M.L., Advanced Engineering Analysis, Willey, 1982

[51] Hoffmann C., Computational fluid dynamics for engineers, Engineering Education System, 1993

[52] Pike J. Riemann Solvers for Perfect and Near-Perfect Gases. AIAA Journal. 1993;(31):1801-1808

[53] Leer B.V., Upstream-Centered Finite Difference Schemes for Ideal Compressible Flow. Journal of Computational Physics. 1977;(23): 263-275

[54] Toro E. F., The Weighted Average Flux Method Applied to the Time Dependent Euler Equations. Phil. Trans. Roy. Soc. London. 1992;(A341): 499-530

[55] Sod G.A., A Survey of Several Finite Difference Methods for Systems of Nonlinear Hyperbolic Conservation Laws. Journal of Computational Physics. 1978;(27): 1-31

[56] Einfeldt B., Munz C.D., Roe P.L., Sjögreen, On Godunov-Type Methods near Low Densities. Journal of Computational Physics. 1991;(92): 273-295

[57] Doolan C.J., Jacobs P.A., Modelling mass entrainment in a quasi-one-dimensional shock tube code, AIAA Journal, 1997, (34): 1291-1293

[58] Marconi F., Rudman S., Calia V., Numerical study of one-dimensional unsteady particle-laden flows with shocks, AIAA Journal, 1981, (19): 1294-1301

[59] Niewood E.H., Sanchez-Martinez M., Quasi-one-dimensional numerical simulation of magnetoplasmadynamic thrusters. Journal of Propulsion and Power, 1992;(8): 1031-1039

[60] Zucrow M.J., Hoffman J.D., Gas Dynamics , Willey, New York, 1976

[61] Tinaztepe H.T., Akmandor İ.S., Üçer A.Ş., Unsteady internal ballistic calculations of rocket motors, . Journal of Propulsion and Power, 1992;(8): 1125-1128

[62] Swafford T. W., Computation of unsteady supersonic quasi-one-dimensional viscous-invicid interacting internal flowfields, AIAA Journal, 1992, (31): 404-408

[63] Deconinck H., "Analysis of wave propagation properties for the Euler equations in two space dimensions" Lecture Series 1994-05, von Karman Institute for Fluid Dynamics, Belgium, 1994.

[64] Dadone A., Grossman B., Surface boundary conditions for the numerical solution of the Euler equations. AIAA Journal 1994;32(2):285-293

[65] Vuillot F, Lupoglazoff N, "Numerical Simulation of Nonsteady Two-Dimensional Flows in Solid Propellant Rocket Motors" La Recherche Aérospatiale, Vol 2, 1992.

[66] Zhou L., Theory and numerical modeling of turbulent and gas-particle flows and combustion., CRC Press, 1993

[67] Jolley W.H., Hooper J.F., Hilton P.R, Bradfield WA "Studies on coning in end-burning rocket motors" Journal of Propulsion and Power , Vol. 2, No. 3, 223-227, 1986.

[68] Başeşme E, Solution of High Speed Flows Using Three-dimensional Method of Characteristics., Ph. D. Thesis. Department of Mechanical Engineering. Middle East Technical University. Turkiye 1998

[69] Liepmann, Roshko, Elements of Gasdynamics, Wiley, 1957

[70] Cauty F., "Non-Intrusive Measurement Methods Applied to Energetic Material Regression Rate Determination", Second International High Energy Materials Conference and Exibith, India, 1998

[71] Schöyer H. F. R., "The Double Helmoltz Resonator as a Possible Tool for the Determination of the Response Function of Solid Rocket Propellants in the Low Frequency Range", Delft Univ. Dept. Of Aero. Eng. Memorandum M-365, 1980

[72] Brown R.S. et.al., Coupling between acoustic velocity oscillations and solid propellant combustion., Journal of Propulsion and Power 1986;2(5):428-437

[73] Combustion Technology of Solid Propellants, AGARD-CPP-259, 1979

[74] Johnston B.P., Sullivan J.M., Fully automatic two-dimensional mesh generation using normal offsetting. International Journal for Numerical Methods in Engineering 1992;33:425-442

[75] Hejl R.J., Heister S.D., Solid rocket motor grain burnback analysis using adaptive grids. Journal of Propulsion and Power 1995;11(5):1006-1011

[76] Ricciardi A., Generalized geometric analysis of right circular cylindrical star perforated and tapered grains. Journal of Propulsion and Power 1992;8(1):51-58

[77] Kao K., Liou M., Advance in overset grid schemes: from chimera to dragon grids. AIAA Journal 1995;33(10):1809-1815

[78] Private Communication with F. Vuillot, RTO-AVT T-108 Project, "2D Internal Flow Applications for Solid Propellant Rocket Motors"

[79] Orkwis P. D, McRae D. S, "Newton's Method Solver for the Axisymmetric Navier-Stokes Equations" AIAA Journal, Vol 30, No 6, 1992.

[80] Olcay O., "The Method of Lines Solution of Time –Dependent Navier-Stokes Equations for Incompressible Separated Internal Flows", METU Ph.D. Thesis June 1997

[81] Glaister P., "Flux Difference Splitting for the Euler Equations with Axial Symmetry" Journal of Engineering Mathematics 22: 107-121, Kluwer Academic Publishers, 1998

[82] Sheng-Tao Y., "Convenient Method to Convert Two-Dimensional CFD Codes into Axisymmetric Ones" Journal of Propulsion and Power, Vol. 9, No 3, 493-495, 1993.

[83] Ben-Artzi M., Birman A, Falcovitz J., "The GRP treatment of Flow Singularities"

[84] Menikof R., "The Riemann Problem for Fluid Flow of Real Materials" Reviews of Modern Physics 61: 75-130, The American Physical Society, 1989

[85] Sharma P.S., Wilson G.J., "Computations of Axisymmetric Flows in Hypersonic Shock Tubes." Journal of Thermohysics and Heat transfer., Vol. 10, No 1, 493-495, 1996.

[86] Yumusak M. E., Taskinoglu E., "2-D Internal Flow Applications for SPRM", Project Report RTO AVT T-108 TR, 1999

[87] Loth E., Baum J., Lohner R., "Formation of shocks within axisymetric nozzles" AIAA Journal, Vol. 30, No 1, 268-270, 1992.

[88] Aksel M. H., Eralp O. C., "Gas Dynamics", Prentice-Hall, 1994

[89] Chwalowski P., Taylor III A.C., "Use of generalized set of field variables in fluid flow calculations" AIAA Paper 91-0240, January, 1991.

[90] Trépanier J.Y., Reggio M., Paraschivoiu M., Camarero R., Unsteady Euler solutions for arbitrarily moving bodies and boundaries. AIAA Journal 1993;31(10):1869-1876

APPENDIX A

GENERAL ROTATION MATRIX FOR X-SPLIT EQUATIONS WITH SPECIES TERMS IN THREE DIMENSIONS

Mathematical groundwork of the method of splitting for multi-dimensional problems is reviewed in Appendix F, Chapter III and IV. To complete the topic a general rotation matrix, which covers three dimensions and species terms will be presented here. Matrices below are used to rotate state vectors to x-splitting direction and for back rotating the resulting flux terms to Cartesian coordinates. The equations are for a state vector of six components. The third spatial direction and species terms are kept in the formulation and used in the code for possible future studies and extensions.

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \cos(Ty)\cdot\cos(Tz) & 0 & \cos(Ty)\cdot\sin(Tz) & \sin(Ty) & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & -\sin(Tz) & 0 & \cos(Tz) & 0 & 0 \\ 0 & -\sin(Ty)\cdot\cos(Tz) & 0 & -\sin(Ty)\sin(Tz) & \cos(Ty) & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \cos(Ty)\cdot\cos(Tz) & 0 & -\sin(Tz) & -\sin(Ty)\cos(Tz) & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & \cos(Ty)\sin(Tz) & 0 & \cos(Tz) & -\sin(Ty)\cdot\sin(Tz) & 0 \\ 0 & \sin(Ty) & 0 & 0 & \cos(Ty) & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

In the above terms cos(Ty) and cos(Tz) are direction cosines of the outward unit normal vector of a three dimensional control volume side. Complete forms of these equations are coded and available. Although the scope of this study is two-dimensional, the last two components of the state vector, namely w-velocity and mass fraction of species qi are kept in the formulation and take part in the calculations. cos(Ty)=1 and sin(Ty)=0 for two-dimensional flows.

APPENDIX B

GEOMETRY DEFINITION AND INPUT/OUTPUT FILES OF THE 2D CARTESIAN GRID SOLVER

2D Cartesian grid solver "2DECGS" requires the input files of Table B-1.

Table B-1: Input Files of 2D Cartesian Grid Solver "2DECGS"

INPUT FILE	INFORMATION
Template.inp	Template dimensions and grid fineness specification.
solid.inp	Contains geometry of stationary surfaces.
Moving.inp	Contains initial geometry of moving surfaces.
bc.inp	Template boundary conditions.
Property.inp	Gas properties and constants for surface injection/movement models.

File template.inp:

The template dimensions are read from the file `template.inp`. For example a solution domain length 48 mm and width 14.4 mm, with a grid fineness factor of 4 is input in the following format:

```
48      1
144     10
4
1
```

With these dimensions minimum possible grid size is 10x3. With a fineness factor of 4, the grid size used in the solution will be 40x12.

The last switch is for selecting planner or axisymmetric solution which must be –1 and 1 respectively.

Related Functions performing integer arithmetic are:

```
void reduce(struct rational *inrat, struct rational *outrat) ;
void multiply(struct rational *r1, struct rational *r2, struct
rational *r3) ;
```

Files solid.inp and moving.inp:

Curves that form the boundaries of solid walls are input from the file `solid.inp` with the following format: (2 separate solid walls, first one is made up of single segment and second one is made up of 4 segments. The start and end point coordinates of each member segment is follows)

```
2
1
200. 27. 107. 0.
4
91. 22. 121. 22.
121. 22. 121. 32.
121. 32. 91. 32.
91. 32. 91. 22.
```

Moving boundaries are input similarly from file `move.inp`. Solid and Moving curves are input so that solid bodies are always on left.

File bc.inp

The following types of boundary conditions can be specified for the EAST, WEST, SOUTH and NORTH sides of the rectangular template. Also for an arbitrary cell in the template, a boundary condition can be specified, in that case switch which is normally –1 in the input file should be 1, and boundary information for each cell should follow. The available boundary condion types with corresponding switches used in the file bc.inp are presented in Table B-2.

Table B-2: Boundary Conditions

BOUNDARY CONDITION	SWITCH
Inflow/Outflow, x-split version	1
Inflow/Outflow, South boundary	10
Inflow/Outflow, East boundary	11
Inflow/Outflow, North boundary	12
Inflow/Outflow, West boundary	13
Injection from East template boundary	1920
Reflective, x-split version	3
Reflective, North/South boundaries	30
Reflective, East/West boundaries	31
Transmissive	4
Moving Wall	6

Boundary conditions are read from the file bc.inp in the following format:

-1 No cell specific boundary condition is input.
30 SOUTH, reflective
4 EAST, transmissive
30 NORTH, reflective
4 WEST, transmissive

Initial conditions are specified in the function loomings:

```
void loomings(struct net *petek, int i, int j, int *mx, int *ny,
double *celsz) ;
```

File property.inp

The gas and propellant properties are input in the following format. All the properties are specified in SI units.

1.211	Specific Heat Ratio
320.31	Gas Constant (J/kg/K)
-1	Injection Switch (1=with, -1=without injection from moving boundaries)
14.5177e-6	Burning Rate Constant a in ($rb=aP^n$)
0.44	Burning Rate Pressure Exponent n in ($rb=aP^n$)
3012.5	Flame Temperature (K)
15.513e6	Stagnation Pressure (Pa)
1710.	Density of Regressing Material (kg/m^3)

APPENDIX C

NOTES ON SOME SELECTED CARTESIAN GRID ALGORITHMS

C.1 INTERSECTION ROUTINE

Function `intersect`, returns the intersection point location and its type if an intersection point exists for the given two line segments. Intersection point location is relative to the first input line segment, in terms of the line segment parameter u. Line segment parameter u, varies from zero to one. $u = 0.$ corresponds to the start point of the first segment and $u = 1.$ corresponds to the end point of the first segment.

First line segment is defined by the vector: $\vec{p}(u) = \vec{a} + u \cdot \vec{b}$

Second line segment is defined by the vector: $\vec{q}(u) = \vec{c} + w \cdot \vec{d}$

Intersection location is given by [45],

$$u = -\frac{(\vec{c} \times \vec{d}) \cdot \vec{a}}{(\vec{c} \times \vec{d}) \cdot \vec{b}} \text{ and } w = -\frac{(\vec{a} \times \vec{b}) \cdot \vec{c}}{(\vec{a} \times \vec{b}) \cdot \vec{d}} \qquad \text{(C.1)}$$

Four possible orientations of two line segments are shown in Figure C-1. In this figure the segment parameter increases in the direction of arrowhead. In this direction the solid part is on the left side.

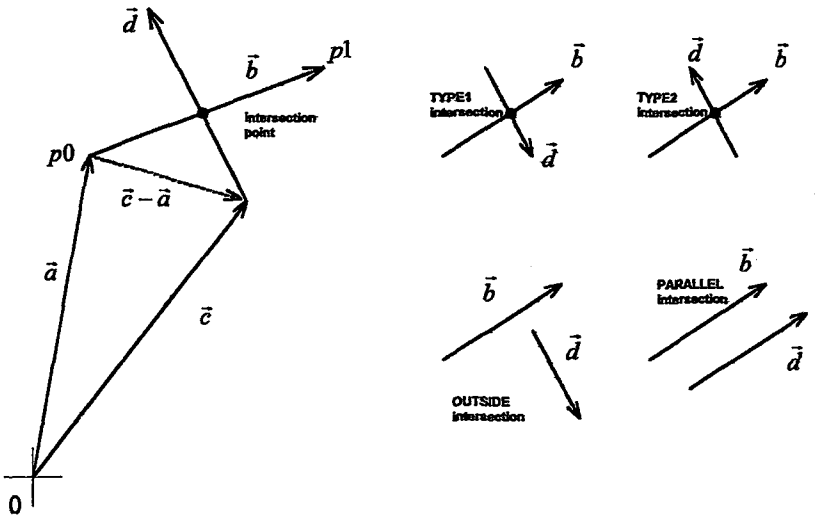


Figure C-1: Four Possible Orientations of Two Line Segments with Sense in 2D.

In the Cartesian grid generation code, for finding cell intersections and degenerate cell considerations different possibilities of parallel segments should be considered separately. These possibilities with corresponding rules are compiled in Figure C-2.

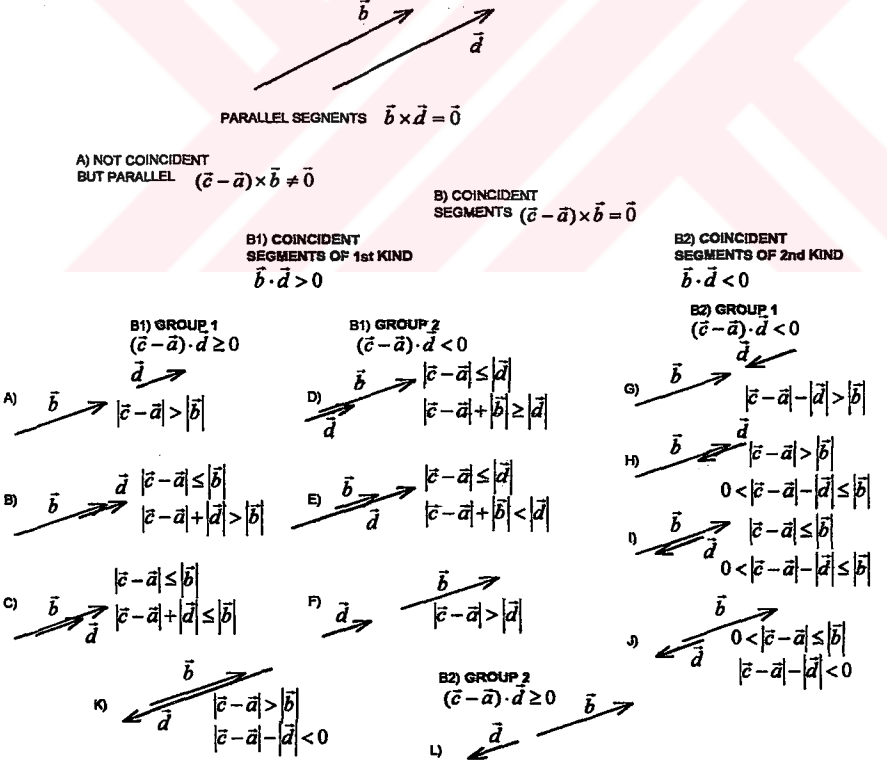


Figure C-2: Parallel segments.

C.2 CELL COMBINATIONS AND LISTS

Two C++ functions are used for this task. The function finder returns three candidate cells for combination. And in function combine the possible combination is made which checks for template boundaries and excessive total combined cell area increase.

```
void finder(struct net *cel, int icel, int jcel,
                    /* Coordinates of cell which seeks a combinatin
*/
        short int *side1, int *icomb1, int *jcomb1,
                    /* The most possible cell combination */
        short int *cside1,
                    /* Side of possible cell */
        short int *side2, int *icomb2, int *jcomb2,
                    /* The second possible cell combination */
        short int *cside2,
                    /* Side of possible cell */
        short int *side3, int *icomb3, int *jcomb3,
                    /* The third possible cell combination */
        short int *cside3          /* Side of possible cell */)


double combine(struct net *cell,
                    /* cell to be combined. Is always SOLITUDE */
                struct net *cel2,
                    /* to the existing/or not list of cel2 */
                short int side1 ,
                    /* which side of cell is combined */
                short int side2 ,
                    /* which side of cel2 is combined */
                struct list1 *list,   /* comblis[listnum] */
                short int *listnum,   /* If cel2 != SOLITUDE, the
*list
                and list number should be same as the one of *cel2
*/)
```

Data structure of the combination lists are as follows.

```
struct oyle {
             struct net *ccel ;  /* pointer to list element */
             double cg[2] ; /* center of list relative to ccel */
             short int side[4] ; /* keeps which boundary of cell i,j
                                    is combined OK or NOK */
             int bi ; /* indis of ccel */
             int bj ; /* y indis of ccel */
             };

struct list1 {
              struct oyle elemans[MAX_COMB] ;
              int n_of_el ; /* number of members in the combination
                               list max is MAX_COMB */
              double toarea ; /* Total area of the combined cells */
              double rout[2] ; /* Center of gravity of list w.r.t.
                                  first elemans i.e [0] */};
```

Cell center coordinates are stored relative to the local coordinates of the first cell in the list. Cell center coordinates and combined cell area are calculated using the following functions.

```
void listaver(struct list1 *comblis)
/* Calculates the average state for the cells that are a member of a
given list */ ;

void thepacific(struct list1 *list, double rout[])
/* Calculates the center of gravity vector of flow area of list */ ;
```

C.3 FINDING STREAMS

A stream is a combination of solid and moving curves. If there are no moving walls and all the geometry that defines fluid boundaries are solid, then each solid wall is assigned as a new stream. If there are moving walls, new streams are generated by tracing moving/solid curves. "A stream can form a closed loop or start and end at a template boundary" is the basic rule. Until this rule is satisfied, each trimmed/extended moving curve is traced first in its start direction and then towards its end. During this trace, intersections with other curves will be detected. Each new detected curve during this trace is kept in the order as a member of the generated stream.

The stream information is kept in the following structure:

```
struct stream {
                struct curve sw_mv[N_CURVE] ;
                int n_curve ;
                short int cloop ; /* Switch for whether the
focused stream is closed or not */
                };
```

Streams are generated by the following C++ functions (algorithms) that are developed:

tracesolid: INPUT: all moving curves and number, all boundaries,
solid curve whose intersection with a boundary or mv is seeked,
start indis of this solid curve, vector of start intersection,
OUTPUT: segment where there is intersection (`end_ind`),
the intersection vector, the intersected moving curve `indis`,
`endtyp`-whether a boundary intersection is detected

tracemove: INPUT: all solid curves and number, all boundaries, moving curve whose intersection with a boundary or sw is seeked,
OUTPUT:
the intersected solid wall indis, the intersection vector, segment where there is intersection (`st_in`), `endtyp`-whether a boundary intersection is detected

tracesolidlop: INPUT: all moving curves and number, the starting loop moving curve,
solid curve whose intersection with a boundary or mv is seeked start indis of this solid curve, vector of start intersection,
OUTPUT: segment where there is intersection (`end_ind`), the intersection vector, the intersected moving curve indis, `endtyp`-whether a boundary intersection is detected

delta:
This function is made up of two main parts. In the first part the stream start point is determined. And in the second part, this stream is generated and stored by tracing moving and solid curves starting from this stream start point. The streams that form loops are also considered. They are two types: streams that are made up of a single moving curve and streams made up of an arbitrary number of solid and moving curves.

Solitude solid walls are not considered in this function but depending the switch `swic2[i]=UNTOUCHED ;` it will be investigated later.

Initialize switches for moving and solid curves. If `TOUCHED` than that curve is a part of a stream. (Will be used in determining solid walls which are embedded in moving material.)

```
swic1[i]=UNTOUCHED ;
swic2[i]=UNTOUCHED ;
```

Initialize the number of streams.

```
strnum = 0 ;
```

Cover moving curves. Skip moving curves that are already considered.

```
if(swic1[i]==TOUCHED) continue ;
```

Loop check. If the moving curve forms a loop, write to stream[strnum]. Increase strnum by one.

Then consider the next moving curve. If this moving curve does not form loop, to find the start of the stream that the this moving curve is a member, do the following:

Initialize start found switch as not found.

```
st_fd=NOK ;
```

Assign the moving curve at hand as hold curve. And mark it as TOUCHED.

```
hldcrv=move[i] ;
```

while(st_fd=NOK) do

Check the intersection of the first segment of the moving curve (hldcrv) with the boundaries.

If an intersection of type 2 is detected. Save the stream start parameters.

```
type=MOVE ;
str_crv=hldcrv ;
i_st=0 ;
rfst[0]=hldcrv.seg[0].p0[0] ;
rfst[1]=hldcrv.seg[0].p0[1] ;
```

And mark st_fd as OK. That is to say, stream start has found.

If st_fd still NOK. Do:

Check the intersection of the first segment of the moving curve (hldcrv) with the segments of solid walls. Trace segments backwards. (Since we are in search of start of the stream.) If an intersection of type 1 (relative to solid segment) is detected. (If no intersection is found error: SW'S AND BOUND'S COVERED, NO INT. MV START END will be printed)

Going backwards from this segment,

Cover last segments of moving curves. Check intersection of type 2. If an intersection is dedected:

First check loop stream condition. If true call function torus. write to stream[strnum]. Increase strnum by one. Take next moving curve. (Loop stream condition is: the first point of this intersected moving curve should coincide with the first point of initial moving curve move[i])

Mark this moving curve as TOUCHED.

Assign new hldcrv as the last segment intersected moving curve.

hldcrv=move[m] ;

Check while loop continuation statement and start all over again with hldcrv=move[m].

Cover Boundaries for the intersection of type 2 with solid segments. If an intersection is detected:

Start stream has found. (Now its solid curve) Save the stream start parameters.

```
type=SOLID ;
str_crv=solid[j] ;
i_st=n ;
rfst[0]=rint[0] ;
rfst[1]=rint[1] ;
```

And mark st_fd as OK. That is to say, stream start has found.

End while loop.

Now the start of the stream has found. The second part of this function generates a new stream. Starting from this start curve (which is found in the first part) the moving and solid curves are traced and written to the current stream.

If stream start is known do:

Initialize stream end found as not found:

entyp=NOK ;

Initialize number of curves in the stream and stream loop switch.

dalyan[strnum].n_curve = 0 ;

dalyan[strnum].cloop = NOK ;

while(entyp =NOK) do

If start curve is SOLID do:

Number of curves in the stream= Number of curves in the stream+1

CALL function tracesolid. (This function traces the input solid curve and finds the first appropriate moving wall or boundary intersection. INPUT: all moving curves and number, all boundaries, solid curve whose intersection with a boundary or mv is seeked, start indis of this solid curve, vector of start intersection. OUTPUT: segment where there is intersection (end_ind), the intersection vector, the intersected moving curve indis, end type(entyp)-whether a boundary intersection is detected.

Write the solid curve (which was the input to the function tracesolid) as a stream member with proper end points.

Assign the moving curve (which is the output of tracesolid) to the new start curve.

str_crv=move[jj] ;

If switch entyp still equals NOK do:

Number of curves in the stream= Number of curves in the stream+1

CALL function tracemove. (This function traces the input moving curve and finds the first appropriate solid wall or boundary intersection. INPUT: all solid curves and number, all boundaries, moving curve whose intersection with a boundary or sw is seeked, OUTPUT: the intersected solid wall indis, the intersection vector, segment where there is intersection (st_in), end type(entyp -whether a boundary intersection is detected.

Write the moving curve (which was the input to the function tracemove) as a stream member with proper end points.

Assign the solid curve (which is the output of tracemove) to the new start curve.

If start curve is SOLID do the same loop which is presented above, but first calling function tracemove then function tracesolid.

End while loop.
End if loop.
Increase strnum by one.
Next moving curve.
Return strnum-1.
End delta.

C.4 BOUNDARY OFFSETTING, END TRIMMING/EXTENDING

The functions devoloped for these tasks are as follows:

route: This function finds the offset point roffs[2] for the input cel which contains a moving segment (roffs[2] is transformed to template coordinates),
Returns the indises of the next cel to be offsetted igo and jgo,
Returns OK, if an end point is found else returns NOK. End point is a cel which contains a kink point or a boundary. Celn' s are neighbour cells starting from SOUTH CCW (1=SOUTH, 2=SE ...)

p_extrapol: Finds the extrapolated start point for the three consecutive points p0....p1
Currently blending functions for four point form are used.
lc=range*celsz, coarsely [0,u] is 3*celsz, then this function r eturns the value at -lc/(3*celsz)

shore: Returns OK if the cel contains a moving segment which starts from a boundary

trim: Updates the end points of the offsetted moving curve

C.5 INVERSE DISTANCE INTERPOLATION

The value required for a given position in space, is found from the values of neighboring points by a weighting function, which depends the inverse of the distance between. For φ_d representing the required data value and φ_s is the value at surrounding points then,

$$\varphi_d = \frac{\sum w_s \varphi_s}{\sum w_s} \tag{C.2}$$

The weighting function used in this thesis is,

$$w_s = D^{3.5} \tag{C.3}$$

Where D is the distance between the two data points.

APPENDIX D

2D UNSTRUCTURED/COMBINED CARTESIAN CELL SOLVER

In this section the solution prosess is presented in algorithm form and when it is appropriate comments are inserted in order to clarify some specific steps. These steps can be followed from the source file "flux2d.cpp" of the 2D Cartesian Euler Code.

Cover All Cells.

If cell is already SOLVED or SOLID, take next cell.

Set Total Flux=0.

If cell is a member of a combination list, cover all the members in this list. Else cell is a SOLITUDE cell then cover only this cell. (Switch `jason` is used for this distinction.)
(In this solver all the member cells of combination lists are solved together by forming a single control volume from the list elements. See Appendix C for averaging the states of the combined cells and calculations for combined cell area.)

Trace cell sides, find right states:

If a cell side is a boundary, call function `ghost` with the appropriate boundary condition type. Function `ghost` returns the corresponding right state for the input boundary condition type (Reflective, Transmissive, Moving Wall, Inflow-Outflow etc)

If the side is not a boundary then there is a neighbor cell. If this neighbor cell is a member of the same list, do nothing and go to the next cell side, since this side is an intercell side of the combined control volume formed by combined cells. Through this side there is no net flux crosses.

If the neighbor cell is not a list member, the right state, then, is the state of this neighbor cell.

Call function `tirbishon` that updates the Total Flux according to cell type. Inputs are Cell Side, Cell Pointer. Right State, Gamma, Cell Size, Delta Time and Total Flux.

End Trace Sides.

Calculate flux along cut segment, call function gocek. Update Total Flux. Inputs to function gocek are: Cell Pointer, Gamma, Cell Size, Delta Time and Total Flux.

End Cover Member Cells.

Function `tirbishon`:

This function updates the total flux vector for vertical and horizontal cell sides. Depending on the type of cut cell and side under consideration, flux from the cell sides are calculated. As an example consider the cell in Figure D. I.

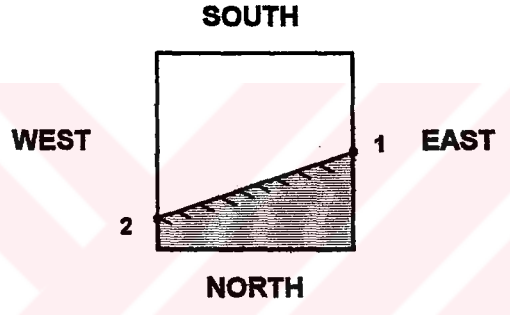


Figure D-1: A Cut Cell of Type CUT2, Sub Type A

Function `tirbishon` returns no contribution from the north side since this side is fully covered with a moving segment. Also note that, flow area for the WEST side is greater than the flow area of the EAST side. These dimensions defining these flow areas are passed to the function `flux2d`, which calculates flux across an arbitrary cut segment, given the right and left cell states. The solution strategy is explained above, here only the steps followed in function `flux2d` will be listed.

The following points should be remembered:

- First point position and second point positions are inputs. (`pnt1[2]` and `pnt2[2]`)
- Cell is on left while going from first to second point.
- Left and right state vectors are input. (`wleft[6]` and `wright[6]`)
- Returns 6 flux components. ($Ls.Ts^{-1}.Fs$)

Procedure:

1) Convert state vectors to conservative variables.

Call function:
```
prim_to_cons( gamma, wleft, uleft) ;
prim_to_cons( gamma, wright, uright) ;
```

2) Rotate conservative state vectors.

Call function:
```
compass( uleft, urolf,  pnt1[0], pnt1[1], pnt2[0], pnt2[1]) ;
compass( uright, urorh, pnt1[0], pnt1[1], pnt2[0], pnt2[1]) ;
```

3) Convert rotated state vectors to primitive variables.

Call function:
```
cons_to_prim( gamma, urolf, wrolf) ;
cons_to_prim( gamma, urorh, wrorh) ;
```

4) Call function `driver` with inputs `wrolf` and `worh`, the rotated state vectors in primitive variables. And obtain rotated flux. Function driver is the same function used in 1D calculations. It calculates the Godunov flux via the exact Riemann solver, the details are covered in Chapter IV.
```
driver( -1/2, 1/2, 3, gamma, deltat, wrolf, wrorh, f) ;
```

5) Rotate flux vector to Cartesian plane.
```
therapin( f, t_1f,      pnt1[0], pnt1[1], pnt2[0], pnt2[1]) ;
```
6) Calculate Length of the segment that flux flows.
```
distance= celsz*denizmili(pnt1[0], pnt1[1], pnt2[0], pnt2[1]);
```
7) Finaly Calculate the desired flux components.
```
for(i=0; i<=5; i++) rflux[i]=distance*t_1f[i] ;
```
8) End function flux2d.

Function `gocek`:

This function calculates the flux for the cut segment. Since the procedure is a little bit different, instead of function `flux2d` it uses function `flux2dw`. The main difference is the calculation of new solution domains which are generated by the retarding walls and specifying the wall velocity for moving walls. Similar to the 1D code, the state of this new solution domain that is created by the expanding wall movement must also be solved. For compressing walls there is not any generated solution domain or space since the wall movement is towards the flow direction. The function `driver2` is utilized which is taken from the 1D code without any change. In the 1D code this function were calculating the missing solution only at the pipe start and end points. Again the exact Riemann problem is solved but this time the aim is not the Godunov flux but the state vector. Which is calculated for any number of specified discrete points and their average is taken as the extra solution. In function gocek, the position of the center of gravity of this extra solution is also calculated. This solution is valid for an inclined rectangle. Sides of this rectangle are defined by the length of the cut segment and wall velocity times deltat.

APPENDIX E

GOVERNING EQUATIONS FOR A GENERAL FLUID FLOW

Governing equations for reacting, three-dimensional axisymmetric, swirling flows with dispersed solid particles in a non-inertial coordinate system are discussed from a physical point of view and presented. These equations are derived in integral form, from which, corresponding differential equations in conservative form follows naturally.

E.1 GOVERNING EQUATIONS FOR LAMINAR FLOWS IN CARTESIAN COORDINATES IN CONSERVATION FORM

E.1.1 CONTINUITY EQUATION (GAS/FLUID PHASE)

Using mass balance and Reynolds transport theorem:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho \mathrm{v}_j\right) = S_{\mathrm{P}}^{m} + S \qquad \text{(E.1)}$$

S_{P}^{m} is the generation of mass per unit volume. This term governs for injecting walls. It is the total species injection and a source term of the gas phase itself.

$$\rho = \sum_s \rho_s \tag{E.2}$$

Density of the gas phase is given in Equation (E.2), where ρ_s is the mass concentration of species s. ($\rho_s = n_s M_s$)

Equation of state;

$$P_s = \rho_s RT / M_s = n_s RT \tag{E.3}$$

M_s is the molecular weight of species s, and n_s is the number of moles of species s per unit volume of the mixture.

S is the particle-fluid/gas interaction term. Total fluid/gas production due to particle phase change. Evaporation of a particle contributes to each species fluid/gas mass. This term is the sum of them.

E.1.2 SPECIES CONSERVATION EQUATIONS

Equations for mixture fraction Y_s, which is mass of species s per total mass of mixture, will be presented. For any scalar quantity ϕ, the advection-diffusion conservation law is given in (E.4).

$$\frac{\partial}{\partial t}(\rho\phi) + \frac{\partial}{\partial x_j}(\rho\phi v_j) = -J_\phi dS + \phi S_p^m + S_p^\phi \tag{E.4}$$

In Equation (E.4);

J_ϕ is the diffusive flux,

ϕS_p^m is the gain due to production of S_p^m per unit volume.

S_p^ϕ is the increase in scalar transport per unit volume due to particle-fluid interaction.

v_j in the above equations are measured relative to non-inertial frame. In the non-inertial frame, position coordinates are represented by x_j.

For $\phi = Y_s$, where $Y_s = \rho_s / \rho$, is the mass fraction of species s. Fick' s law, for the diffusive flux is given in Equation (E.5).

$$J_\phi = J_{sj} = -\rho D_s \frac{\partial Y_s}{\partial x_j} \tag{E.5}$$

In Equation (E.5), J_{sj} is the diffusive flux of species s in j-direction. $D_s = f(Y_1, Y_2, ..., Y_s)$. A simplifying assumption is $D_1 = D_2 = ... = D_s = D$.

Substituting (E.5) to Equation (E.4);

$$\frac{\partial}{\partial t}(\rho Y_s) + \frac{\partial}{\partial x_j}(\rho Y_s v_j) = \rho D_s \frac{\partial Y_s}{\partial x_j} + Y_s^{inj} S_P^m + S_P^{Y_s} - W_s \tag{E.6}$$

$S_P^{Y_s} = \alpha_s S$ is the amount of species s production due to particle reaction or evaporation.

α_s is the fraction of contribution of s species in phase change.

W_s is the reaction term, negative if species s is produced due to chemical kinetics.

$Y_s^{inj} S_P^m$ is the amount of that species generated per unit volume due to the injected mass fraction Y_s^{inj}. S_P^m is generally zero over the solution domain, only when there is mass injection it will be used.

The summation of all species conservation equations gives the continuity equation due to $\sum_s W_s = 0$ and $\sum_s Y_s S_P^m = S_P^m$. So if there are totally "z" species only "z-1" species conservation equations will be needed.

E.1.3 CHEMICAL KINETICS

Stociometric relation of a single step forward chemical reaction;

$$\sum_s \nu_s A_s \rightarrow \sum_s \nu_s' A_s' \tag{E.7}$$

A_s, A_s' are chemical symbols for reactants and products respectively.

ν_s, ν_s' are the corresponding stochiometric coefficients.

W_s is the reaction rate of s species, which is defined as the consumed or produced mass per unit volume per unit time as formulated in Equation (E.8) and (E.9).

$$W_s = \left(-\frac{\partial \rho_s}{\partial t}\right)_{chem} \tag{E.8}$$

$$W_s = k_s \prod_{s=1}^{z} C_s^{\nu_s} \tag{E.9}$$

C_s represents mass concentration ρ_s or molar concentration $n_s M_s$

k_s is the rate coefficient.

z is the total number of species.

$\nu = \nu_1 + \nu_2 + ... = \sum \nu_s$ is the apparent reaction order. For most reactions actual order is not equal to apparent order, that is;E.

$$W_s = k_s \prod_{s=1}^{z} C_s^{m_s} \tag{E.10}$$

m is the actual reaction order. ($m_s \neq \nu_s$ and $m = \sum m_s \neq \nu = \sum \nu_s$)

If there are several simultaneous reactions and the r^{th} reaction is;

$$\sum_{s=1}^{z} \nu_{sr} A_s \rightarrow \sum_{s=1}^{z'} \nu_{sr}' A_s' \tag{E.11}$$

Total reaction rate will be;

$$W_s = \sum_r W_{sr} = \sum_r k_{sr} \prod_{s=1}^{z} C_s^{m_{sr}} \qquad \text{(E.12)}$$

For a reversible reaction;

$$\sum_s \nu_s A_s \leftrightarrow \sum_s \nu_s' A_s' \qquad \text{(E.13)}$$

The forward reaction rate is;

$$W_{s+} = k_s \prod_{s=1}^{z} C_s^{m_s} \qquad \text{(E.14)}$$

The rate of backward reaction;

$$W_{s-} = k_s' \prod_{s=1}^{z'} C_s'^{m_s'} \qquad \text{(E.15)}$$

So the net reaction term rate is given in Equation (E.16).

$$W_s = W_{s+} - W_{s-} = k_s \prod_s C_s^{m_s} - k_s' \prod_s C_s'^{m_s'} \qquad \text{(E.16)}$$

For r simultaneous reversible reactions, the total reaction rate is;

$$W_s = \sum_r W_{sr} = \sum_r (k_{sr} \prod_{s=1}^{z} C_s^{m_{sr}} - k_{sr}' \prod_{s=1}^{z'} C_s'^{m_{sr}'}) \qquad \text{(E.17)}$$

By Arhenius law;

$k_s = k_{0s} \exp(-E / RT)$

E is the activation energy

R is universal gas constant.

$k_{0s} = BT^{0.5}$

B is called pre-exponential function.

In some treatments of combustion problems an equivalent one-step reaction is considered, in that case the global reaction rate is expressed by;

$$W_s = k_{0s}\rho^{\sum m_s} \exp\left(-\frac{E}{RT}\right)\prod_s Y_s^{m_s} \qquad \text{(E.18)}$$

This approach is generally preferred in modeling solid phase pryolisis or evaporation. Where k_{0s}, E, m_s and $\sum m_s$ are pure empirical constants.

E.1.4 MOMENTUM EQUATION

Observing and modeling some types of flows from a non-inertial frame in some cases may be more intuitive, then an inertial frame. The transformations presented in Figure E-1, which will be used in the equations, follow from dynamics. Application areas are turbomachinery flows and rocket motors with arbitrary acceleration.

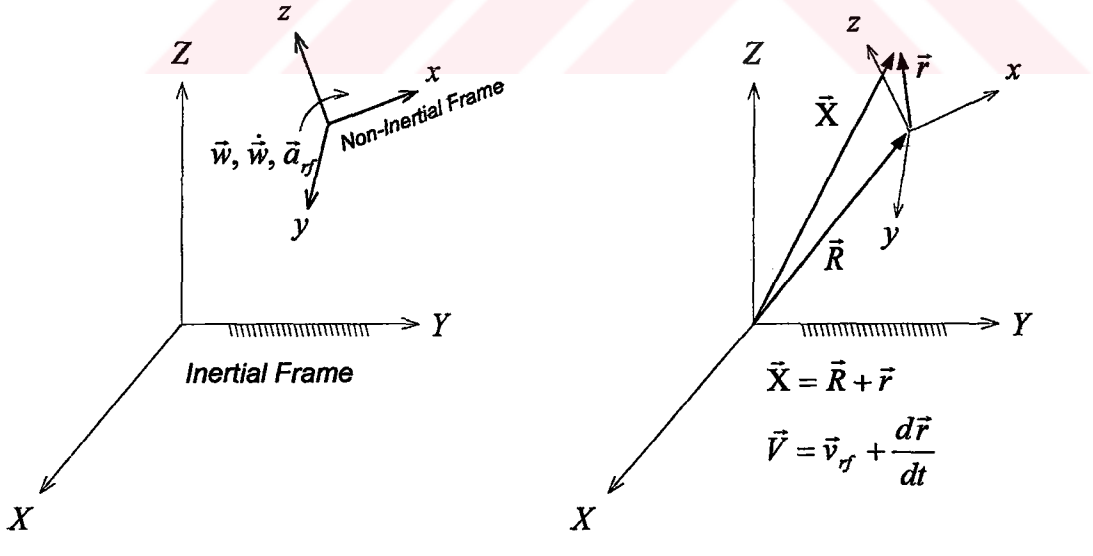


Figure E-1: Definitions and Kinematic Transformations between Inertial and Non-inertial Frames of Reference.

For fluid mechanics, representing the acceleration of the non-inertial frame as a body force (D'Alémbert's force) will be convenient and the momentum equation for a single componènt fluid can be formulated as,

$$\vec{F}_S + \vec{F}_B - \int_{CV}\left[\vec{a}_{rf} + 2\vec{w}\times\vec{V}_{xyz} + \vec{w}\times(\vec{w}\times\vec{r}) + \dot{\vec{w}}\times\vec{r}\right]\rho d\forall = \frac{\partial}{\partial t}\int_{CV}\vec{V}_{xyz}\rho d\forall + \int_{CS}\vec{V}_{xyz}\rho\vec{V}_{xyz}\cdot d\vec{A} \quad \text{(E.19)}$$

Which follows that, in the momentum equation, all velocities will be in terms of $\vec{V}_{xyz}$, velocity measured with respect to the rotating *xyz*-frame. This velocity is represented by $\vec{v}$ in this formulation. However in the energy equation the absolute velocity, $\vec{V}_{XYZ}$ should be used. They are related to each other by kinematics via Equation (E.20).

$$\vec{V}_{XYZ} = \vec{V} = \vec{v}_{rf} + \vec{V}_{xyz} + \vec{w}\times\vec{r} \quad \text{(E.20)}$$

The velocity terms related to the diffusion of species needs some clarification, as given below.

$\vec{V}_s$ represents *s* species velocity with respect to laboratory coordinate. (Inertial frame XYZ)

$\vec{V}_{XYZ}$ ($\vec{V}$) is mixture velocity with respect to laboratory coordinate.

$\vec{V}_s$ is *s* species velocity with reference to mixture motion, caused by diffusion drift of molecular random motion.

$$\rho_s\vec{V}_s = \vec{J}_s \qquad J_{sj} = \left(-\rho D_s\frac{\partial Y_s}{\partial x_j}\right) \quad \text{(E.21)}$$

The relation between these three velocity vectors is given in Equation (E.22).

$$\vec{V}_s = \vec{V}_s - \vec{V}_{XYZ} \quad \text{(E.22)}$$

One more definition is required before presenting the full form of the momentum equation;

$\rho_s \vec{F}_s$ ($\rho_s F_{si}$) is the body force which could be gravitational force ρg_i, the electric force $\rho_e E_i$, and the magnetic force $(\vec{J} \times \vec{B})_i$.

$$\vec{F}_S + \vec{F}_B = \int_{CV} \left(\frac{\partial \sigma_{ij}}{\partial x_j} + \sum \rho_s F_{si} \right) \delta \forall \tag{E.23}$$

Substituting all the terms, Equation (E.24) follows as,

$$\frac{\partial}{\partial t}(\rho V_{xyz\,i}) + \frac{\partial}{\partial x_j}(\rho V_{xyz\,j} V_{xyz\,i}) = \left(\frac{\partial \sigma_{ij}}{\partial x_j} \right) + \sum \rho_s F_{si} - \sum \rho_s \left[\vec{a}_{rf} + 2\vec{w} \times \vec{V}_{s\,xyz} + \vec{w} \times (\vec{w} \times \vec{r}) + \dot{\vec{w}} \times \vec{r} \right]_i + S_P^m V_{xyz\,i}^{inj} + S_{Pi} \tag{E.24}$$

$S_P^m V_{xyz\,i}^{inj}$ represents momentum generated due to production of mass with injection velocity in the control volume.

$\vec{V}_{s\,xyz} = \vec{V}_s + \vec{V}_{xyz}$ is the relation between velocity components.

$S_{P\,i}$ is the increase in *i*-momentum due to particle fluid interaction.

Velocity of mass production term S is zero.

E.1.5 ENERGY EQUATION

Conservation of Energy in integral form with mechanical engineering sign convention for work terms is given in Equation (E.25).

$$\dot{Q} - \dot{W} = \left(\frac{dE}{dt} \right)_{system} \tag{E.25}$$

In this equation, E_{system} is the total energy of the system given by Equation (E.26).

$$E_{system} = \int_{M(system)} e\,dm = \int_{\forall(system)} e \rho\, d\forall \tag{E.26}$$

$$e = u + \frac{V^2}{2}$$

Where $e\rho$, is the energy per unit volume and u is the internal energy. Body force contribution is included in the form of work input.

System and control volume formulations are related by Lagrange equation;

$$\left(\frac{dE}{dt}\right)_{system} = \frac{\partial}{\partial t}\int_{CV} e\rho d\forall + \int_{CS} e\rho \vec{V}\cdot d\vec{A} \qquad \text{(E.27)}$$

and the Stokes relation $\int_{CS}(e\rho\vec{V})\cdot d\vec{A} = \int_{CS}(e\rho\vec{V})\cdot \hat{n}dS = \int_{CV}\nabla\cdot(e\rho\vec{V})d\forall$.

In Equation (E.27), $\nabla\cdot$ represents divergence operator.

(For cylindrical coordinates $div\vec{A} = \frac{1}{r}\frac{\partial}{\partial r}(rA_r) + \frac{1}{r}\frac{\partial A_\theta}{\partial \theta} + \frac{\partial A_z}{\partial z}$)

$$\dot{Q} = \int_{CV}\left(\frac{\partial q_j^r}{\partial x_j} + \frac{\partial}{\partial x_j}\left(k\frac{\partial T}{\partial x_j}\right) + (h^{inj} + \frac{V^{inj\,2}}{2})S_P^m + h^{ph-cg}S + S_P^e + \frac{\partial}{\partial x_j}\left(\sum_s D_s\rho\frac{\partial Y_s}{\partial x_j}h_s\right)\right)d\forall$$

(E.28)

Equation (E.28) is the heat transfer with the following contributions. The first term represents radiative heat transfer, volumetric heating of fluid inside control volume due to the absorption of radiation outside the system, or the local emission of radiation by the fluid itself, if the temperature of the fluid inside the control volume is high enough. The second term is due to conduction Energy generated due to production (by injection) of mass in the control volume or system is represented by the third term. If the flow is chemically reacting it may be tempting to consider energy released or absorbed by such reactions as a volumetric heating term. Fourth and fifth terms represents the enthalpy introduced by phase changing fluid and

increase in energy due to particle fluid interaction. The last contribution is the multi-component diffusion term.

The total rate of work done on the control volume, $\dot{W}$, is composed of the terms given in Equation (E.29).

$$\dot{W} = \dot{W}_{surface} + \dot{W}_{bodyforces} + \dot{W}_{shaft} + \dot{W}_{other} \tag{E.29}$$

In Equations (E.30), (E.31), (E.32) and in (E.33) these terms are presented separately.

$$\dot{W}_{surface} = -\int_{CV} \frac{\partial}{\partial x_j}\left(\sigma_{ij} \cdot \mathrm{V}_i\right)\delta\forall \tag{E.30}$$

In Equation (E.30) the surface stress term is $\sigma_{ij} = -P\delta_{ij} + \tau_{ij}$.

$\dot{W}_{bodyforces}$ includes work done by gravitational force, electric force, magnetic force. For a general body force term it is given by Equation (E.31).

$$\dot{W}_{bodyforces} = -\int_{CV}\sum_{s} \rho_s \vec{F}_s . \vec{\mathrm{V}}_s \delta\forall \tag{E.31}$$

$\vec{\mathrm{V}}_s$ is defined previously as $\vec{\mathrm{V}}_s = \vec{V}_s + \vec{\mathrm{V}}$.

Equation (E.31) can be further expanded to two terms that are presented in Equation (E.32).

$$\int_{CV}\sum_{s} \rho_s \vec{F}_s . \vec{\mathrm{V}}_s \delta\forall = \int_{CV}\left(\vec{\mathrm{V}} \cdot \sum \rho_s \vec{F}_s + \sum \rho_s \vec{F}_s \cdot \vec{V}_s\right)\delta\forall \tag{E.32}$$

$$\dot{W}_{shafts} = -\int_{A(shafts)} \tau \cdot \vec{v}_{sh} dA \tag{E.33}$$

Finally substituting all these to Equation (E.25). The Energy Equation in conservative form is presented in Equation (E.34), below.

$$\frac{\partial}{\partial t}(\rho e)+\frac{\partial}{\partial x_j}\left(e\rho \mathrm{V_j}-k\frac{\partial T}{\partial x_j}-\sum_s D_s\rho\frac{\partial Y_s}{\partial x_j}h_s-q_j^r-\sigma_{ij}\cdot V_i\right) \\ =(h_{inj}+\frac{\mathrm{V}^{inj^2}}{2})S_{\mathrm{P}}^m+h^{ph-cg}S+S_{\mathrm{P}}^e+\left(\bar{\mathrm{V}}\cdot\sum\rho_{\mathrm{s}}\bar{F}_s+\sum\rho_s\bar{F}_s.\bar{V}_s\right)+\dot{W}_{shafts} \tag{E.34}$$

Substituting the stress tensor $\sigma_{ij}=-P\delta_{ij}+\tau_{ij}$ to the flux term of (E.34), two new terms that are given in Equation (E.35) will appear.

$$\frac{\partial}{\partial x_j}\left(\left(-P\delta_{ij}+\tau_{ij}\right)\cdot \mathrm{V}_i\right)=\frac{\partial}{\partial x_j}\left(\sigma_{ij}\cdot \mathrm{V}_i\right) \\ =\mathrm{V}_i\frac{\partial}{\partial x_j}\sigma_{ij}+\sigma_{ij}\frac{\partial \mathrm{V}_i}{\partial x_j} \tag{E.35}$$

In the formulations, the first term of the right hand side of Equation (E.35), $\mathrm{V}_i\frac{\partial \sigma_{ij}}{\partial x_j}$, is sometimes related through mechanical energy equation, i.e. $\bar{\mathrm{V}}\cdot$ momentum equation. However this form will be used in this study.

Besides the source term, in Equation (E.34) there are two additional terms, to the governing equations of non-reacting single component flows;

1. Net enthalpy flux caused by the species diffusion fluxes
2. Work done by the body forces with the diffusion drift velocity

All the velocity components are absolute in the Equation (E.34). F_s is the body force for gravity, electric and magnetic effects. This body force should not be mixed with the fictitious body force representation of extra acceleration terms that are appearing in momentum equation for flows in non-inertial frames.

E.2 PARTICLE-FLUID INTERACTION AND TURBULENCE

Extra equations should be derived to model particle-fluid interaction terms. Depending on the modelling approach, some of these extra equations may be;

- kth particle phase continuity
- mixture continuity
- kth particle phase momentum
- kth particle phase energy
- mixture energy
- mixture momentum

Turbulence modeling, unless algebraic, introduces two more advection and diffusion equations for k and ε. Modeling turbulent flows with particles requires many assumptions and as a result of such assumptions various terms can be omitted. Derivation of equations that will handle turbulence depends on experimental experience. Since these two topics are special research areas, detailed derivations are not covered in this study.

APPENDIX F

MATHEMATICAL PROPERTIES OF EULER EQUATIONS

F.1 TWO–DIMENSIONAL EULER EQUATIONS IN CONSERVATIVE FORM

Euler equations are obtained by neglecting viscosity, heat conduction and body forces for compressible flows. They are hyperbolic conservation laws and their study is an important area of applied mathematics. Some properties of the Euler equations, which form the basis of numerical methods, will be presented in this chapter. To study these properties a model equation is needed. The Euler equations in two space dimensions are selected as the model equation and the mathematical properties will be compiled with reference to them. In a compact state vector form, they are given in Equation (F.1).

$$\vec{U}_t + \vec{F}(\vec{U})_x + \vec{G}(\vec{U})_y = \vec{0}, \tag{F.1}$$

with,

$$\vec{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}, \vec{F} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{bmatrix}, \vec{G} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{bmatrix}$$

Equation (F.1) can also be written in the so-called quasi-linear form;

$$\vec{U}_t + \vec{A}(\vec{U})U_x + \vec{B}(\vec{U})\vec{U}_y = \vec{0} \tag{F.2}$$

$\vec{A}(\vec{U})$ and $\vec{B}(\vec{U})$ are respectively the Jacobian matrices of the fluxes $\vec{F}(\vec{U})$ and $\vec{G}(\vec{U})$. Where $\vec{A}(\vec{U})$, which corresponds to the flux $\vec{F}(\vec{U})$ is given by Equation (F.3).

$$\vec{A}(\vec{U}) = \frac{\partial \vec{F}}{\partial \vec{U}} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -u^2 + \frac{1}{2}(\gamma-1)\vec{V}^2 & (3-\gamma)u & -(\gamma-1)v & \gamma-1 \\ -uv & v & u & 0 \\ u\left[\frac{1}{2}(\gamma-1)V^2 - H\right] & H-(\gamma-1)u^2 & -(\gamma-1)uv & \gamma u \end{bmatrix} \tag{F.3}$$

The eigenvalues of coefficient matrix $\vec{A}$ are,

$$\lambda_1 = u-a,\ \lambda_2 = \lambda_3 = u,\ \lambda_4 = u+a \tag{F.4}$$

with the corresponding right eigenvectors, presented in Equation (F.5).

$$\vec{K}^{(1)} = \begin{bmatrix} 1 \\ u-a \\ v \\ H-au \end{bmatrix},\ \vec{K}^{(2)} = \begin{bmatrix} 1 \\ u \\ v \\ \frac{1}{2}\vec{V}^2 \end{bmatrix},\ \vec{K}^{(3)} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ v \end{bmatrix},\ \vec{K}^{(4)} = \begin{bmatrix} 1 \\ u+a \\ v \\ H+au \end{bmatrix} \tag{F.5}$$

The Homogeneity property of the flux terms of Euler equations is given in Equation (F.6). This property is satisfied for ideal gas equation of state when the conservative variables are used in the formulation [89]. It has an important influence on the accuracy of numerical flux differences [37].

$$\vec{F}(\vec{U}) = \vec{A}(\vec{U})\vec{U} \tag{F.6}$$

The Rotational Invariance property of the Euler equations is crucial for computational purposes when dealing with domain boundaries that are not aligned with the Cartesian directions. The x-split version of the Euler equations, which will be introduced in Section 2., is utilized in multi-dimensional problems via this property.

$$\cos\theta\,\vec{F}(\vec{U}) + \sin\theta\,\vec{G}(\vec{U}) = T^{-1}\,\vec{F}(T\vec{U}) \tag{F.7}$$

Referring to Figure F-1, Equation (F.7) is satisfied for all angles θ, which is measured in the anticlockwise direction, and vectors $\vec{U}$. $T = T(\theta)$ is the rotation matrix and $T^{-1}(\theta)$ is its inverse as presented in Equation (F.8).

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & \sin\theta & 0 \\ 0 & -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad T^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{F.8}$$

The rotational matrices for the most general three-dimensional x-split form with species equations are given in Appendix A.

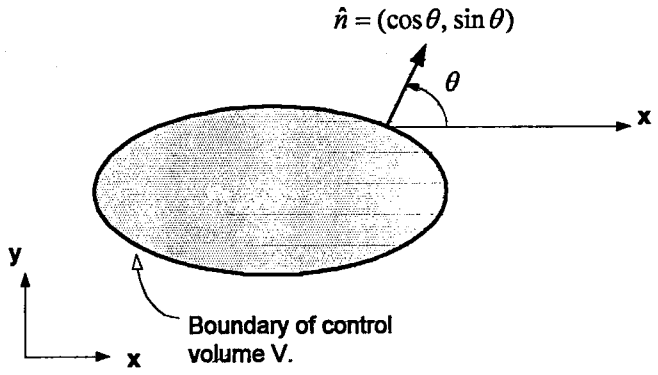


Figure F-1: Control volume V, bounded by the surface with unit normal vector n.

For the equation system given in (F.1), the matrix, in Equation (F.9) is diagonalizable, for all admissible states $\vec{U}$ and real angles θ.

Each eigenvalue $\lambda_i(\vec{U})$, given in Equation (F.4), defines a characteristic field, so called λ_i-field. The types of these fields and their relation with the resulting wave structure will be discussed next.

A λ_i-characteristic field is Linearly Degenerate if

$$\nabla \lambda_i(\vec{U}) \cdot \vec{K}^{(i)}(\vec{U}) = 0 \qquad \text{(F.10)}$$

where, $\vec{U}$ is a real valued state vector with four components. For the Euler equation system given in (F.1) λ_2-field and λ_3-fields are linearly degenerate.

A discontinuity in a linearly degenerate field is called contact discontinuity [38]. In an invicid 1D shock-tube problem the gas initially on one side of the diaphragm never mixes with the gas on the other side. Two gases remain contact along a ray in the x-t plane along which there is a jump in density. Across the waves associated with λ_2- and λ_3- characteristics, pressure and u-velocity is constant. Field-2 is associated with a shock tube analogue contact discontinuity, across which density jumps discontinuously. (With consecutive jumps in the other conserved quantities;

momentum and energy, and in variables that depend on density; temperature, sound speed, entropy) Characteristic field -3 is related to a shear wave across which v-velocity component jumps discontinuously.

A λ_i-characteristic field is Genuinely Nonlinear if

$$\nabla\lambda_i(\vec{U})\cdot\vec{K}^{(i)}(\vec{U})\neq 0 \tag{F.11}$$

where $\vec{U}$ is a real valued state vector. This is an entropy condition that can be applied to a discontinuous weak solution to determine whether the jumps are allowed. For scalar equations, a convex flux function has characteristics that go into the shock and Equation (F.11) is satisfied. The convex behavior of the flux function is related to the monotonicity of the characteristic speed.

For equation system (F.1) the 1- and 4 -characteristic fields are genuinely non-linear thus these fields are associated with shock waves and rarefaction waves.

In the above equations (F.10) and (F.11), the gradient of $\lambda_i(\vec{U})$ is

$$\nabla\lambda_i(\vec{U})=\left(\frac{\partial}{\partial u_1}\lambda_i,\frac{\partial}{\partial u_2}\lambda_i,\ldots,\frac{\partial}{\partial u_m}\lambda_i\right)^T \tag{F.12}$$

with m=4 for two-dimensional Euler equations.

F.2 RELATIONS FOR A ONE-DIMENSIONAL NONLINEAR CONSERVATION LAW

$$\left.\begin{aligned}\vec{U}_t+\vec{F}(\vec{U})_x&=\vec{0}\\ \vec{U}(x,0)=\vec{U}^{(0)}(x)&=\begin{cases}\vec{U}_L & if\quad x<0\\ \vec{U}_R & if\quad x>0\end{cases}\end{aligned}\right\} \tag{F.13}$$

where

$$\vec{U} = \begin{bmatrix} \rho \\ \rho u \\ E \\ \rho v \\ \rho w \end{bmatrix}, \quad \vec{F}(\vec{U}) = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ u(E+p) \\ \rho uv \\ \rho uw \end{bmatrix}$$

Given the hyperbolic conservation law system (F.13), and a discontinuous wave solution of speed S_i associated with the λ_i-characteristic field, the relations in (F.14) hold true across the discontinuous wave,

$$\Delta\vec{F} = S_i \, \Delta\vec{U} \tag{F.14}$$

Where $\vec{F}$ is the flux vector in Equation (F.13), with $\Delta\vec{U} \equiv \vec{U}_R - \vec{U}_L$, $\Delta\vec{F} \equiv \vec{F}_R - \vec{F}_L$, and left and right flux terms, $\vec{F}_L = \vec{F}(\vec{U}_L)$, $\vec{F}_R = \vec{F}(\vec{U}_R)$.

Since the equation system is nonlinear, it is not possible to solve Equation (F.14) for the speed S_i.

Equation system in (F.13), using any suitable set of dependent variables $\vec{W}$, instead of $\vec{U}$, can be written as a quasi-linear hyperbolic system.

$$\vec{W}_t + \vec{A}(\vec{W})\vec{W}_x = \vec{0} \tag{F.15}$$

with,

$$\vec{W} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{bmatrix}$$

The wave is associated with the i-characteristic field of eigenvalue λ_i and corresponding right eigenvector.

$$\vec{K}^{(i)} = \begin{bmatrix} k_1^{(i)} \\ k_2^{(i)} \\ \vdots \\ k_m^{(i)} \end{bmatrix}.$$

The generalized Riemann Invariants are relations that hold true, waves, across the wave structure and lead the following $(m-1)$ ordinary differential equations

$$\frac{dw_1}{k_1^{(i)}} = \frac{dw_2}{k_2^{(i)}} = \frac{dw_3}{k_3^{(i)}} = \cdots = \frac{dw_m}{k_m^{(i)}} \tag{F.16}$$

Relations (F.14) and (F.16) can be applied to the elementary-wave solutions of the Riemann problem.

For a shock wave with two constant states $\vec{U}_L$ and $\vec{U}_R$ the Rankine-Hugoniot Conditions follow from Equation (F.14) are

$$\vec{F}(\vec{U}_R) - \vec{F}(\vec{U}_L) = S_i(\vec{U}_R - \vec{U}_L) \tag{F.17}$$

Satisfying the entropy condition, which says that characteristics are running in to the field as given in Equation (F.18).

$$\lambda_i(\vec{U}_L) > S_i > \lambda_i(\vec{U}_R) \tag{F.18}$$

For a contact wave, a single jump discontinuity with speed S_i separating two data states, $\vec{U}_L$ and $\vec{U}_R$, the conditions in (F.14) are

$$\vec{F}(\vec{U}_R) - \vec{F}(\vec{U}_L) = S_i(\vec{U}_R - \vec{U}_L) \tag{F.19}$$

Generalized Riemann Invariants across the wave follows from Equation (F.16).

The characteristics run parallel to the contact wave as stated in Equation (F.20).

$$\lambda_i(\vec{U}_L) = S_i = \lambda_i(\vec{U}_R) \tag{F.20}$$

Rarefaction waves are connected through a smooth transition with Generalized Riemann Invariants (F.16) and divergence of characteristics is shown by the relation (F.21).

$$\lambda_i(\vec{U}_L) < \lambda_i(\vec{U}_R) \tag{F.21}$$

VITA

Kerem Pekkan was born in Ankara, on August 1970. After graduating from TED Ankara College, he has recieved his B.S degree in Mechanical Engineering from Middle East Technical University in June 1992. He has completed his M.Sc. studies as a teaching and research assistant in the Fluid Dynamics Laboratory of Mechanical Engineering department from 1992 to February 1995. Since then he has worked in the propulsion group of Research and Development department of ROKETSAN Missiles Industries Inc. as an engineer. His main area of interest are Computational and Experimental Fluid Mechanics, Turbomachinery and Mechanical Design.