POLITECNICO DI MILANO

Facoltà di Ingegneria Industriale

Corso di Laurea in
Ingegneria Aeronautica

Algebraic models for aerodynamic coefficients calculation during the atmospheric re-entry

Relatore:	Prof. Alberto GUARDONE
Co-relatore:	Ing. Francesco CASTELLINI
Co-relatore:	Ing. Dominic DIRKX

Tesi di Laurea di:

Francesco VILLA: Matr. 725451

Anno Accademico 2010/2011

Abstract

The present thesis work introduces an algorithm for the approximate solution of hypersonic viscous flows, under the hypothesis of thin shock layer. This hypothesis has been used widely in hypersonic calculations, since it poses the bases to a wide range of approximate methods for calculating inviscid pressure distribution over bodies of arbitrary shape. These approximate solutions provide accurate results with a very low calculation effort, if compared to CFD hypersonic solutors, hence providing a quick aerodynamic tool for solving attitude problems and design optimization analyses. The aim of the present work is to extend this solving approach with a viscous characterization of the vehicle moving in an hypersonic flowfield. Approximate viscous methods and correlations developed in hypersonic literature are investigated, and their coupling with the inviscid approximate solution is discussed.

After discussing the validity of theoretical basis for this kind of approach, an algorithm for the approximate calculation of inviscid and viscous flowfield properties is proposed: local surface inclination methods are used to solve the inviscid flow, while shear stress and heat flow are calculated using approximate reference enthalpy relations. An estimated velocity field is proposed and integrated, to obtain reference streamlines on which approximate viscous relations are applied. In order to calculate wall temperature, thermal equilibrium between the convective heat flow and the heat flow emitted by the vehicle is imposed, under adiabatic wall and non-radiative gas hypotheses. Thermal analysis is conducted together with viscous calculations, for the strong dependance existing between shear stress and heat flow distribution, exemplified by the Reynolds analogy. Particular care is used for the accurate modelling of the gas mixture through the imposition of thermochemical equilibrium, since in hypersonic flow, as we will demonstrate, high temperature effects cannot be neglected.

The algorithm is applicable to an arbitrary vehicle shape, under the LaWGS format, calculating thermodynamic properties, shear stress and heat flow on the vehicle surface, given the freestream flow. At the current state of the work, the algorithm is implemented partially in C++ and partially in Matlab. The algorithm is developed with the aim to extend the Re-entry Aerodynamic Module (RAM) of the Space Trajectory Analysis software, an open-source space mission analysis environment developed by the European Space Agency, in order to make it capable of viscous and thermal preliminary design calculations. The inviscid solver is in fact directly borrowed from the RAM v1.0 module, which analyzes an arbitrary vehicle shape and reconstructs the inviscid pressure field via approximated methods, while a Matlab algorithm is developed for the streamline reconstruction and for viscous and thermal analysis. RAM v1.0 code classes have been modified in order to build objects apt to receive the viscous and thermal solutor in the near future.

To conclude the work, a comparison between the results of our viscous approximate solver and more sophisticated hypersonic flows CFD simulations is performed, in order to give a first esteem of the accuracy which can be reached with this kind of approach.

Abstract - Italian

Questo lavoro di tesi presenta un algoritmo per la soluzione approssimata dei flussi viscosi ipersonici, sotto l'ipotesi di thin shock layer. Questa ipotesi è stata ampiamente utilizzata nei software di calcolo ipersonici, in quanto pone le basi per una vasta gamma di metodi approssimati per il calcolo della distribuzione di pressione sui corpi di forma arbitraria. Queste soluzioni approssimate possono fornire risultati accurati con uno sforzo di calcolo molto basso, se confrontato con solutori CFD ipersonici, e quindi sono in grado di fornire uno strumento di calcolo aerodinamico rapido, per risolvere per esempio problemi di assetto o di ottimizzazione in fase di design preliminare. Lo scopo del presente lavoro è quello di estendere questo approccio risolutivo con una caratterizzazione viscosa del veicolo in movimento in un campo di moto ipersonico. I metodi viscosi e le correlazioni approssimate sviluppate nella letteratura ipersonica sono indagati, e il loro accoppiamento con la soluzione approssimata non viscosa è proposto.

Dopo aver discusso la validitá della base teorica per questo tipo di approccio, un algoritmo per il calcolo delle proprietà di un campo di moto ipersonico è sviluppato: metodi di inclinazione locali sono utilizzati per risolvere il flusso viscoso, mentre le sollecitazione degli sforzi di taglio e il flusso termico sono calcolati utilizzando dei metodi di entalpia di riferimento per caratterizzare lo stato energetico dello strato limite. Un campo di velocità stimato è proposto ed integrato, per ottenere un campo di streamlines di riferimento sul quale applicare le correlazioni viscose approssimate. Per calcolare la temperatura di parete, l'equilibrio termico tra il flusso di calore convettivo e il flusso di calore emesso dal veicolo è imposto, sotto le ipotesi di parete adiabatica e gas non radiativi. L'analisi termica è condotta contemporaneamente al modello viscoso, per la forte dipendenza esistente tra lo sforzo di taglio e la distribuzione del flusso termico sulla superficie, esemplificato dall'analogia di Reynolds. Particolare cura viene utilizzata per la modellazione accurata della miscela di gas attraverso l'imposizione dell' equilibrio termochimico, poichè in un flusso ipersonico, come vedremo in seguito, gli effetti della temperatura elevata non possono essere trascurato.

L'algoritmo è applicabile ad una geometria arbitraria, secondo il formato

LaWGS, calcolando le proprietà termodinamiche, lo sforzo di taglio e il flusso di calore sulla superficie del veicolo, data la corrente asintotica. Allo stato attuale del lavoro, l'algoritmo è implementato parzialmente in $C++$ e parzialmente in Matlab. L'algoritmo è stato sviluppato con l'obiettivo di estendere il Re-entry Aerodynamic Module (RAM) dello *Space Trajectory Analysis* software, un ambiente open-source per l'analisi di una missione spaziale, sviluppato dall'Agenzia Spaziale Europea, al fine di renderlo capace di calcoli preliminari della parte viscosa e dei flussi termici. Il solutore non viscoso è infatti preso in prestito direttamente dal modulo RAM v1.0, che analizza una forma arbitraria del veicolo e ricostruisce il campo di pressioni senza attrito attraverso metodi approssimati, mentre un algoritmo Matlab è stato sviluppato per la ricostruzione delle streamline e per l'analisi viscosa e termica. Le classi del codice RAM v1.0 sono state modificate al fine di costruire oggetti idonei a ricevere il solutore termico e viscoso in un prossimo futuro.

Per concludere il lavoro, è presentato un confronto tra i risultati del nostro solutore viscoso approssimato con più accurate simulazioni CFD ipersoniche, in modo da fornire una prima stima della precisione che può essere raggiunta con questo tipo di approccio.

Contents

List of Figures

List of Tables

List of Symbols

Letters

- ▶ α - angle of attack angle $[deg]$
- ▶ β - sideslip or shock deflection angle $[deg]$
- ▶ δ - boundary layer thickness $[m]$
- ▶ δ^*- boundary layer displacement thickness $[m]$
- ▶ γ - specific heats ratio
- ▶ θ - surface to free-stream angle, wedge angle$[deg]$
- ▶ $\bar{\chi}$ - viscous interaction similarity parameter
- ▶ c_p - constant pressure specific heat coefficient $\left[\frac{kJ}{kg\,K}\right]$
- ▶ c_v- constant volume specific heat coefficient $\left[\frac{kJ}{kg\,K}\right]$
- ▶ M - Mach number
- ▶ $\hat{\mathbf{n}}$ - panel normal vector
- ▶ p - pressure
- ▶ R - gas constant $\left[\frac{kJ}{kg\,K}\right]$
- ▶ Re - Reynolds number
- ▶ $\mathbf{V}$ - velocity vector

Subscripts

- $_\infty$ - with respect to freestream condition
- $_e$ - with respect to external flow
- $_x$ - relative to distance x, usually referred to curvilinear streamline coordinate for defining Re_x

Part I

Introduction and theoretical study

Chapter 1

Introduction

1.1 Research motivation and goals

Computational fluid dynamics have reached by present day a significant evolution. The accurate modelling of Navier-Stokes equations has attained a significant degree of confidence, even in the analysis of hypersonic flows, which requires a satisfying treatment of compressibility and thermochemical effects on the gas mixture. However, this classic approach in the field of numeric aerodynamics has some disadvantages when the number of solutions required is great. The solving of Navier - Stokes equations, even with turbulence models, is an expensive process in terms of required time and computational effort, and gives an extremely detailed description of the flow on the surface of a vehicle and in a wide surrounding region. For these reasons, the classic CFD approach is unsuitable in solving problems which require fast and repeated calculation of the aerodynamics properties on a body, such as vehicle shape design and optimization or trajectory analysis. For this kind of studies, a high number of solutions is needed, while the wide spread of information and the fine accuracy provided by a CFD code is often unnecessary. The aim of an approximate solutor like the one we are proposing here is then to obtain a fast tool to have quick estimates of the flowfield properties over a body at an incidence, in order to calculate the aerodynamic characterization of a vehicle at a given attitude in seconds or minutes, instead of the hours needed to an accurate hypersonic solver. In this way aerodynamic forces can be quickly obtained to solve an attitude problem, for example the trajectory propagation of a re-entering spacecraft; or an attitude-Mach matrix can be obtained in order to perform optimization analysis, which would be prohibitive if we would be using only CFD tools.

There is another reason to develop an approximate solver in the hypersonic field, and that reason is the fact that hypersonic aerodynamics lends itself to approximation. The particular nature of the flow in this extreme region of aerodynamics makes possible to describe highly non-linear relations with very simplified hypotheses. These assumptions will be discussed in detail in the following sections, but we can say that, thanks to these hypotheses, it will be possible to calculate some flow properties, such as pressure and velocity vectors on the surface, locally - i.e. knowing only the freestream flow and the local inclination of the vehicle surface. This will lead to a fast and simplified inviscid solution, on which we will reconstruct the viscous effect. These effects can't be described locally, because they depend on the flow history, but they can be reconstructed once the inviscid solution is known. This approach is typical of other approximate hypersonic solvers, as will it be discussed when considering the state of the art of this branch of aerodynamics.

We can now begin to understand why the construction of this kind of algorithm is justifiable, and for what kind of purposes it is suitable to use this estimated analysis. This thesis work was born indeed in the context of a mission analysis tool developed at the European Space Agency, named Space Trajectory Analysis. STA is a development project for an open source astrodynamics suite, which involves an huge number of universities and research institutions, including Politecnico di Milano, and its coordinated by the European Space Agency ESTEC center. For what concerns aerodynamics, an important partner of this project is also the Von Karman Institute, which provided its support also for this specific work. The idea underlying this program is to develop a mission analysis tool involving professors, researchers, PhD and M. Sc. students from renowned European institutions and universites; promoting in this way the exchange of scientific and technical ideas in the area of astrodynamics and space exploration. STA is currently developed under an open source format, although its availability has been recently restricted to ESTEC internal use. The project is written in C++ language, using the object oriented programming in order to create a modular software architecture, which permits the simultaneous development of different software features by independent groups or individuals. A professional Graphic User Interface is developed, using the free Nokia Qt IDE, such that the final user finds an intuitive and easy to learn environment in which he can plan and analyze space missions. The main idea underlying STA architecture is to create a space scenario, in which active participants, such as satellites, ground stations, spacecrafts are inserted, and for every player particular actions can be performed. For example, one can characterize a satellite or a spacecraft with a mission arc, i.e. a loitering orbit, a space manouver or a re-entry and landing path. Other actions, such as, e.g., transmission simulation between ground stations and satellites can be activated, depending on the nature of the actors.

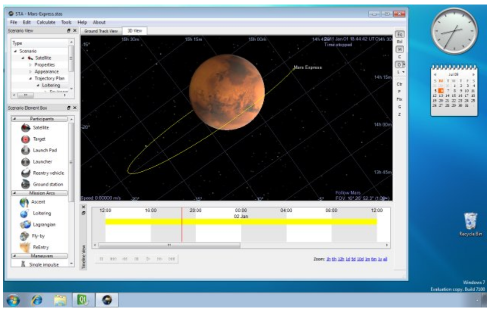


Figure 1.1: STA main view, including object selection and mission arcs windows (left), propagation bar (bottom) and trajectory view (central)

The application of our analysis to STA is located in the re-entry trajectory analysis module, which needed an aerodynamic module, named Re-entry Aerodynamic Module, or RAM, for the trajectory and attitude propagation during the re-entry phase of a space mission. In particular, an inviscid approximate solver exploiting the particular features of the hypersonic flowfield was already been developed, during the M.Sc. thesis work of Dominic Dirkx - for more informations see [1] - and will be called here RAM v1.0. The present work illustrates the concepts underlying this kind of solver, and extend, both theoretically and practically, the aerodynamic solver elaborated in the STA project to the viscous characterization of the flow. This extension will include the determination of approximate relations for calculating the shear stress pattern on the vehicle surface, and the heat flow transfer calculation in the hypothesis of equilibrium between the convective heat flow in the fluid and the thermal radiation emitted by the vehicle surface, considering the wall as adiabatic and the fluid as non-radiative. The goal of this work is then to provide a tool to quickly esteem the hypersonic flow over an arbitrary vehicle shape, reconstructing pressure and shear stress distribution on the surface, in order to calculate the aerodynamic coefficients for a given freestream flow and attitude. A calculation of the heat flow distribution is also performed, both for thermal analysis, and because its determination is mandatory for the viscous characterization of the flow. A comparison between the algorithm's results and CFD calculations over different shapes of re-entering space vehicles will be presented. Even if our algorithm is still in the development phase, and it is still partially written in Matlab, we will refer to it as the RAM v2.0, or viscous Re-entry Aerodynamics Module, since its main destination is to provide a viscous and thermal aerodynamic analysis tool to STA software. Obviously, this is not the only possible application of the algorithm that is outlined in this work, since its architecture can be exported to other environments, due to its open source nature. For example, it can be proposed as a stand-alone aerodynamic solver, or can be integrated, for example, in optimization softwares which want to include viscous aerodynamics or heating analysis in the design of launchers and re-entry vehicles.

1.2 Thesis outline

1.2.1 General

The present thesis work is subdivided in three parts, the first one introducing the work and presenting all the teorethical bases which underlie to the inviscid and viscous approximate analysis. In the second part, an algorithm for the calculation of shear stress and heat transfer, based on theoretical conclusions from part I, is exposed. The third part deals with RAM v2.0 algorithm results and comparison with high accuracy hypersonic CFD simulations, and proposes development guidelines to improve results quality and algorithm's performances.

1.2.2 Part I

In the current chapter, a discussion on re-entry and hypersonic aerodynamics' main peculiarities is dealt with, presenting the influence of the particular phenomena which occur in this extreme region of gas dynamics. A brief overview on the state-of-the art hypersonic approximate solvers is presented. Chapter 2 deals with the definition of inviscid local surface inclination approximate solutions methods, to obtain pressure distribution over a body of arbitray shape. These methods are the backbone of RAM v1.0 algorithm, and are the basis on which we will build our viscous analysis solver. In chapter 3, the search for approximate viscous solutions in the hypersonic flowfield is faced, by looking at new advancements in the reference enthalpy methods, and the thermal analysis on the vehicle surface is sketched, since viscous and thermal properties are so interconnected in the hypersonic regime that they are almost the same phenomena, and together they have to be dealt with. In chapter 4, a thermochemical equilibrium model of the gas mixture is presented, since the flowfield thermodynamic properties cannot be determined without enormous error from the common perfect gas hypothesis, completely wrong in hypersonic gas mixture flowfields.

1.2.3 Part II

Part two is divided in two chapters, the first one, chapter 5, presenting the algorithm outline, including a summary of the previous RAM v1.0 version, and the general approximation of the flow field, including surface velocity field determination and streamline calculation and integration, stagnation point analysis and separation. A viscous solution is proposed, obtained from the iteration on wall temperature, by imposing a heat flow balance between convective heat from the flow and radiation emitted by the vehicle surface.

Chapter 6 presents the geometrical definition of the problem, and exposes the numerical tools adopted to calculate the relations proposed in chapter 5. These features include Runge Kutta integrator and quadrilateral interpolation for streamline calculation over a rough geometry.

1.2.4 Part III

Part three deals with the comparison of the results obtained by RAM v2.0, presented in chapter 7, against some high accuracy CFD simulations over actual re-entry capsules and test geometries, such as IXV and Expert. An in-depth comparison with the DART ESA test geometry, which is used for CFD codes validation, is performed.

Chapter 8 summarize the results achieved and the already defined routes to carry on the implementation, to improve both the accuracy and the efficiency of RAM v2.0 algorithm.

A final appendix presents a tutorial for the use of RAM v1.0 and v2.0 as standalone analysis, and shows in more detail the current implementation of the algorithm.

1.3 Re-entry aerodynamics

Planetary re-entry is probably the most critical phase of a space mission, expecially for manned flights. High velocities and atmospheric gas rarefaction during the aerodynamic braking of the vehicle from orbit to conditions proper to atmospheric glide or parachute opening lead to a very severe heating load on the vehicle surface, so that the thermal protection design and the shape optimization for reducing heat become key factors in the development of the project. The crucial importance of this mission phase has been tragically stressed by the Space Shuttle Columbia accident of February 1st, 2003, when a failure in the reinforced carbon-carbon thermal protection shield on the left wing leading edge, due to a foam debris impact in the launch phase, led to the loss of the vehicle during the re-entry phase, causing the death of all the seven crewmembers flying the mission. It is significant that the crack of the TPS was located at the leading edge, because in this region the flow stagnates, thus leading to a greater increase of the after-shock pressure and temperature, which results in a dramatic peak of the heat flow.

Figure 1.2: IXV during re-entry phase - artist concept

The determination of the flow properties must also take into account the various phenomena that occur in the hypersonic regime, which is the most trying condition that the vehicle will encounter through its way back home. In this region, the intense temperature rise behind the shock wave, combined with the rarefaction of the freestream flow, leads to phenomena such as molecular dissociation and ionization, and radiative heath transfer from the high temperature gas to the surface.

Viscous friction, which is closely linked to heating by Reynold's analogy, is also crucial for re-entry trajectory determination. The contribution of viscous drag can be small compared to the huge pressure drag induced by the shock and the flow impact on the surface, but it becomes significant when its contribution is integrated on an entire arch of re-entry.

1.4 Hypersonic flows

How to define an hypersonic flow? The answer is not trivial, since there isn't a change in the flow behaviour from a supersonic to a hypersonic flow so abrupt as it can be between subsonic and supersonic flow. In this paragraph we will try to describe the most important features of an hypersonic flow, i.e. the features that will characterize the difference between an hypersonic flow and a simply supersonic flow. We stress again the fact that there isn't a definite boundary between these two types of flow, so we will describe features that will increase with Mach number, and when these features become predominant, we can talk of an hypersonic flow. Let us describe these properties and their involvment in our study. It is important to notice, in the following subsections, what hypotheses of all the hypersonic flow properties are considered to design our algorithm. For an in depth description of these features, we recommend the reading of [2].

1.4.1 Thin shock layer

One of the properties of an hypersonic flow is that, when the flow encounters a body, it produces a shock which is very close to the body. If we call the region between the shock and the body a shock layer, we can say that hypersonic flows are characterized by a thin shock layer. This phenomenon happens as a prosecution of the oblique shock layer theory, which states that if we increase the freestream Mach number, we obtain a shock which lies closer to the body. This can be easily verified by considering the relation between the wedge angle and the shock angle, given by the equation

$$tan\,(\theta) = 2\,cot\,(\beta)\left[\frac{M_1^2 sin^2\,(\beta) - 1}{M_1^2\,(\gamma + cos\,(2\beta)) + 2}\right] \tag{1.1}$$

which lead to the well-known β-θ diagram:

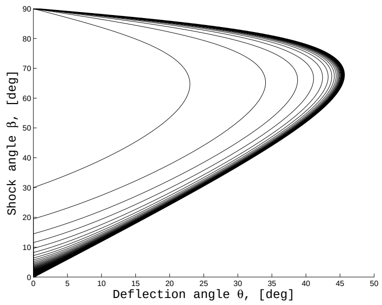


Figure 1.3: β - θ diagram for $M = [2,\, \infty]$

where we can predict that the shock tends to be closer to the body as M tends to infinity, as one can see also in the following image, where the oblique shock relation is resolved for a wedge at $\theta = 25°$ and increasing Mach number:

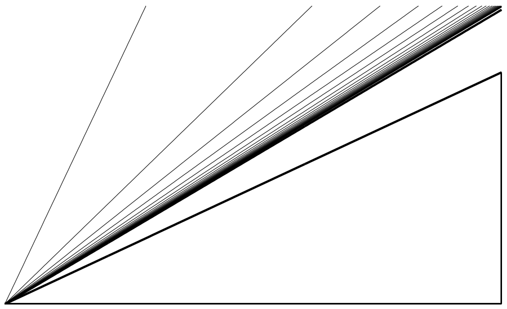


Figure 1.4: Oblique shock over a 25° fence at $M = [2,\, \infty]$

This behaviour leads to one of the key hypotheses of our algorithm, that is the fact that the flows tends to change almost instantaneously its direction from the freestream orientation to a direction tangential to the surface. This consideration leads to a very simplifying approximate definition of the velocity vector over the body surface, which is found by considering the velocity on the body as the tangential component of the freestream velocity:

$$\mathbf{V}_{\text{body}} = \mathbf{V}_{\parallel} = \mathbf{V}_{\infty} - \mathbf{V}_{\perp} = \mathbf{V}_{\infty} - \hat{\mathbf{n}} \cdot \mathbf{V}_{\infty} \cdot \hat{\mathbf{n}} \tag{1.2}$$

This hypothesis permits us to achieve two fundamental results:

- The first fact is that we have the velocity field on the vehicle surface, which will be indispensable for calculating body streamlines in order to characterize the viscous flow, as we will se in the following chapters.
- The second fact is that we will obtain the inviscid pressure on the vehicle surface, simply by considering the loss of normal momentum in the almost instantaneous change of flow direction from normal to tangential. This is the fundamental hypothesis of Newton method, which will be discussed in the next chapter.

This hypothesis is used in professional hypersonic approximate solvers such as CBAERO, which will be described in sec. 1.5.

1.4.2 Entropy layer

This phenomenon is a direct consequence of the existence of a thin shock layer. We can say, by knowing that the shock intensity decreases with its inclination, that a normal shock produces a higher entropy increase rather than an oblique shock. In hypersonic flows, if we have a blunt body, we have a normal shock near the stagnation region which lies very near to the body, due to the fact that the shock layer is thin. After the blunt nose, the shock becomes quickly inclined. We have then an abrupt change of shock intensity near the body surface, thus leading to strong gradients of entropy in the proximity of the wall. We can then identify this zone of near-body, fast-changing entropy as an entropy layer, a concept similar to the boundary layer, but involving entropy rather than velocity. In hypersonic flows, the entropy layer is usually comparable to the boundary layer, so the boundary layer starts its growth in this variable entropy region, that for the Crocco's theorem is also a strong vorticity region. This fact leads to some problems in the correct identification of external flow and internal flow, with respect to the boundary layer. Fortunately, this effect is significant only if we perform an exact calculation of the boundary layer flowfield, for example by solving the boundary layer equations. Since we are planning to construct an approximate solver of the hypersonic flow which won't calculate the exact boundary layer solution, the entropy layer does not affect our algorithm, so it will be ignored.

1.4.3 Viscous interaction

In hypersonic flows, the boundary layer is dragged by an external flow which contains a large amount of kinetic energy. The subsequent viscous dissipation is necessarily large, resulting in an increase of temperature within the boundary layer. Such an increase of temperature produces an increase of viscosity, which results in a thicker boundary layer. Remembering also the state equation $P = \rho RT$, an increase of temperature at a fixed pressure - since the normal gradient through the layer is zero - results in a decreased density. To sustain the required mass flow, the boundary layer has to grow even thicker. The combination of these two effects results in an increased boundary layer growth in hypersonic flows. In fact, for a uncompressible flat plate flow we have that the boundary layer thickness δ is, for a laminar flow:

$$\delta \propto \frac{1}{\sqrt{Re_x}} \tag{1.3}$$

while for a compressible flat plate flow, this length goes as, according to [2]

$$\delta \propto \frac{M_\infty^2}{\sqrt{Re_x}} \tag{1.4}$$

thus becoming significantly increased for high Mach numbers.

The consequence of this faster increase of δ is felt by the external flow as an increased displacement effect, so that the inviscid flow sees a much thicker body with respect to the incompressible flow. The change in the external flow provides a further modification of boundary layer growht. This two-way interaction between external and boundary flow is called viscous interaction. The effects of viscous interaction are signficantly larger in the nose region, and consist mainly in a modification of the surface pressure distribution, affecting lift and momentum, and of the shear stress and heat flow distribution. In particular, we have a general rise of pressure in the nose region, for the augmented displacement effect, and a general increase of skin friction and heat transfer. Unfortunately, no local approximate method which can reconstruct pressure rise due to viscous interaction can be reconstructed from here, so we will have to neglect viscous interaction in the development of our algorithm. This seems an unsuccesful conclusion, but as we will see in section 1.5, every professional hypersonic approximate solver neglects this effect as well. Indeed, comparisons between our algorithm and professional hypersonic CFD resulsts show no sensible influence of the viscous interaction on the pressure field, up to $M_\infty = 15$, as can be seen in chapter 7.

1.4.4 High temperature effects

Another key concept of hypersonics is the fact that the flow can reach extremely high temperatures - up to 11000 K for Apollo re-entry, almost twice the temperature on the surface of the Sun. In these severe conditions, the hypotesis of air, or other extra-terrestrial atmosphere gas mixtures, as a calorically perfect gas must be abandoned. If we take the gas mixture known as air, and we look at its properties as temperature rises, we can see that:

- below 800 K, the mixture can be considered as a calorically perfect gas, i.e. with constant specific heats. Specific heat ratio $\gamma = 1.4$ and mixture constant $R = 286.9 \left[\frac{kJ}{kg\,K}\right]$ are then constants.
- from 800 K to 2000 K, the vibrational degrees of freedom of the molecules composing the mixture are excited. As a result of this fact, specific heats begin to show a relation with temperature and pressure: $c_p = c_p\,(T,\,p)$ and $c_v = c_v\,(T,\,p)$. As a consequence, we will have also $\gamma = \gamma\,(T,\,p)$ and $R = R\,(T,\,p)$.
- Above 2000 K, the mixture becomes chemically active and the molecules of different species begin to dissociate. Mixture composition starts to change, with the creation of new species and the progressive disappearance of initial species. The new molecular and atomic composition bring to a non-linear variation of gas properties - i.e. the dependence of c_p and c_v from temperature varies not only from excitation, but also from the variation of the species molar fraction in the mixture.
- Above 9000 K, ions are formed, and the gas mixture becomes a partially ionized plasma, the model being similar to a dissociated mixture, but considering also free electrons and ions as species.

A chemically active mixture can be described following two different assumptions: in the first case, we suppose that chemical reaction take place almost instantaneously, so that the chemical composition is directly dependant over the temperature of the gas. This is the case of reactions happening in a lapse of time much shorter than the timestep needed for a fluid particle to travel from one point to another, and it is called *thermochemical equilibrium.* The second case is when reaction time and fluid-dynamic time are comparable. In this case, chemical composition depends on the temperature and on the travel history of a particle. This kind of flow fall under the name of *thermochemical non-equilibrium*, and is far more difficult to represent than the thermochemical equilibrium. In counterposition to these two teories on chemically reacting flow, a low temperature flowfield with no reaction at all falls under the definition of *frozen composition* or *frozen equilibrium flow.* In our algorithm we will use the hypothesis of thermochemical equilibrium, which is described in detail in chapter 4 to calculate the flow across the shock. We notice that often this high temperature behavior of the gas is referred as *real gas effect*. This definition is ambiguous, since a real gas model is a modelling of a single-species gas with intermolecular forces, and not a description of a chemically reacting mixture with no intermolecular forces as in our case. These effects will be then addressed as *high temperature effects* or simply as *thermochemical equilibrium*. In the development of the model, we found evidence that a thermochemical equilibrium representation is mandatory in order to obtain correct values behind the shock, and we noticed that the pressure and temperature rise behind the shock wave was excessively overestimated if a simple frozen equilibrium, constant $\gamma = 1.4$ model was used. On the other hand, after the shock, the differences between the two models become small again, so to save calculation time a frozen equilibrium composition hypothesis on the vehicle external flow can be used, obviously by imposing the specific heats, γ and composition of the gas mixture after the shock - we surely won't have anything with a 1.4 γ specific heat ratio after an hypersonic normal shock.

Another important effect which arises from the high temperature of the mixture is the surface heating caused by the high energy flow. There are two main forms of heating, the first is the classical *convective heating*, which is generated by the hot fluid flowing on the surface. The second kind of heating, which gain more importance when the flow temperatures rise to severely high levels, is the *radiative heating* from the hot fluid to the surface, that must not be confused with the radiation created by the vehicle surface which radiates to the external flow. This kind of radiative heating exist when the fluid particles reach extremely high temperatures (above 9000 K) so that they start to emit considerable amount of thermal radiation. In our modelling of the hot gas, convective heating is considered, since it is the main heating source on a re-entering space vehicle, while the description of the gas radiative heating is omitted, since it is less relevant, almost until very high temperatures are reached, mainly because of the extreme difficulty of modelization of a radiative gas surrounding a body. An esteem of the termal flow over the vehicle surface is then obtained, in our algorithm, by imposing an equilibrium between the convective flow granted by the fluid on the surface, and the radiation emitted by the surface, considered as adiabatic, i.e. without an heat sink.

There can be other complex phenomena near the vehicle wall, such as molecular recombination in the boundary layer, or combination with ablative material chemical products, which produces complex chemical reactions near the wall. For the degree of approximation required by our model these kind of phenomena can be safely neglected.

1.5 State of the art

1.5.1 The S/HABP software

The first software which presented the concept of approximate hypersonic software was the Supersonic/Hypersonic Arbitrary Body Program - S/HABP - developed by Douglas Aircraft company in 1964, and modified various times until a final version was released in 1980. We present the key concepts of this algorithm, a detailed description of it being present in reference [3]. The main concept of this software is to define an arbitrary geometry using quadrilaterals - or *panels* - and to determine with approximate methods the inviscid flow properties - i.e. the inviscid pressure distribution. This set of methods, which are known as Local Surface Inclination methods, or LSI, are used to esteem the inviscid pressure on a panel knowing only the local angle between the panel and the freestream flow, and obviously the freestream flow properties. Such methods, which are derived from different theoretical assumptions, but sharing the main concept of thin shock layers presented in 1.4.1, will be presented in detail in the next chapter.

When the inviscid flowfield is resolved, streamlines are calculated and a viscous approximation based on this information is attempted. The approximate viscous methods used are however quite obsolete, if compared to the methods presented in this work. Another limitation of S/HABP is that a real consideration of high temperature effects is missing, and these phenomena are reconstructed using only experimental correlations, instead of performing a thermochemical equilibrium calculation, thus reconstructing the actual properties of the mixture. The inviscid solver maintains however a high degree of confidence on the results while performing only light calculations, due to the fact that LSI methods provide quite accurate results with extremely low information. An interesting comparison between the S/HABP Mark IV inviscid solution and actual experiment data presented in [4] shows the good level of agreement of these inviscid results on a variety of shapes. The other main advantage of S/HABP is that - at present day - it is released as free software, Fortran 77 source code included. For these reasons it is still used today as a preliminary design and optimization tool.

1.5.2 The CBAERO software

The Configuration Based Aerodynamic tool, or CBAERO, represents the present day evolution of S/HABP software. Developed at the Ames Research center at the beginnig of this century, it springs from the same intuitions of S/HABP - i.e. using LSI methods in combination with streamline calculation as a basis for the viscous and thermal analysis - but with present day implementation concepts. The mesh is a triangular unstructred mesh, which can provide better accuracy in describing complex shapes with respect to a panel, and can be directly linked to other solvers parametrization. The advantages are that an immediate comparison with CFD solution can be easily made, and it is also possible, for example, to link an Euler equations inviscid solver to the aerothermal viscous analyzer of CBAERO, bypassing the LSI methods and using a more accurate viscous solution. This solution will not be adopted in our program, since we need to support the LaWGS for compatibility with STA software, which is a quadrilateral ordered mesh similar to S/HABP panelization.

Other interesting concept, that will be used in our work are the C++ object oriented implementation, which is present in our algorithm as a skeleton for future STA integration and the use of a topologic analysis for the determination of stagnation points and attachment lines on the vehicle surface. Another interesting concept in CBAERO is the definition of search trees to order the mesh points in order to perform faster searches of panel points in the integration of the velocity field on the surface. These concept are explained very clearly in the program presentation by Kinney [5]. In this work the concepts of surface velocity field obtained by imposing tangential flow on the panel and calculating the surface heat flow by imposing equilibrium between convective heat flow and adiabatic wall emitted radiation are also present. The reference enthalpy method, which is the approximate viscous calculation method that will be used also in this work, although in a more advanced version with respect to CBAERO, is also presented.

CBAERO is surely a modern hypersonic preliminary analysis tool, but it still shows some limitations. The first and most evident one is that the program is not freely available to the public, and also its reperibility for purchase is very limited. The other fact is that the viscous solution doesn't take into account the most recent developments of the reference enthalpy method for shear stress and heat flow calculation. The discussion about these advancements, that are used by our algorithm, will be the core of chapter 3.

1.6 The LaWGS format

1.6.1 General description

We introduce here the LaWGS format, defined as the standard geometry vehicle description format from Langley Research Center, and used in STA and RAM v1.0 to identify space vehicles shapes. The LaWGS files, characterized by the *.wgs extension, will be the first input for the inviscid calculations performed by RAM v1.0 algorithm. A more accurate description can be found in

Dirkx RAM v1.0 description in [1] or in the Nasa Langley report [6].

Defining the global coordinate system, the x axis is to be set along the main axis of the vehicle, with the minimum x value at the nose and the maximum x at the back. z axis is chosen from the bottom to the top, and y is to be set in order to form a right handed cartesian basis. This setting is fundamental to a proper outcome of the calculation. To obtain proper physical description, all the points coordinates have to be expressed in meters.

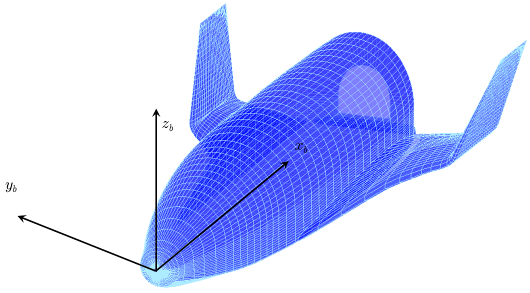


Figure 1.5: Global coordinate system

A *.wgs file is substantially composed by objects, or parts, which for example represent a fuselage, a wing, a flap. It is important to have separated objects dividing a fusiform body - i.e. a fuselage or a generic body which has the y direction comparable to the z direction - from a non fusiform body - e.g. a wing, a flap, or a body wich is developed mainly in the y direction. At the present moment, the algorithm is not able to treat vertical tails, although a simple rotation of the coordinates should fix this problem.

Every object is defined by *contours*, so points are subdivided in *contour lines*. For fusiform shapes, the first line defines the points standing at the fuselage front ordinate, while the last line contains the points corresponding to the last ordinate. For non fusiform bodies, the lines substantially correspond to the wing ribs, even if is not mandatory that the y coordinates are the same. In the following figure contour lines are presented, notice that contours on a fusiform body are oriented as fuselage ordinates, while on a non fusiform body they are parallel to wing ribs.

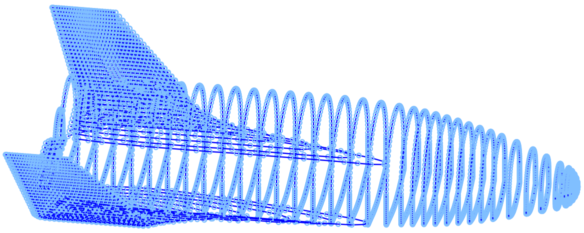


Figure 1.6: Points (light cyan) subdivided in contour lines (blue)

On every line, points are arranged in a clockwise manner. This subdivision in contour lines is very important, since it will be used to develop several routines during the algorithm, which will depend on the point grouping in contour lines. We will use the general convention, during our exposition to denote contour lines by the index i, or I, and clockwise indexing along a single line with the index j, or J. Obviously, j index is coherent between different lines, so that, e.g., point with index $j = 31$ on line $i = 15$ will be adjacent to point $j = 31$ on line $i = 16$, and on line$i = 14$ as well.

1.6.2 *.wgs file buildup

The file is composed by a first line containing the vehicle name, enclosed in single quotes:

- line 1 : 'Vehicle Name'

Then the description for every object is written, following the subsequent format

- line i : 'Part Name'
- line i+1 : Charachterization array
- line i+2, i+3, ... : Point coordinates

The charachterization array contain the information about the number of lines of which the part is composed, the number of points for every line, the rotation angles and the translation coordinates from a local coordinate system, the local and global symmetry identificators, and the scaling factor. All the numbers are put in one line, separated by a single space, as in the following scheme:

$$ID\ nLines\ nPoints\ locSym\ \theta_x\ \theta_y\ \theta_z\ \Delta x\ \Delta y\ \Delta z\ zoomX\ zoomY\ zoomZ\ globSym$$

where

- *ID* is the part number identification
- *nLines* is the number of contour lines composing the part
- *nPoints* is the number of points for every contour line
- *locSym* is the local symmetry identificator, i.e. if a part is self-symmetrical with respect of a plane of the local coordinate system
- θ_x θ_y θ_z are the rotation angles, in radiants, between the local and the global coordinate system
- Δx Δy Δz are the translation coordinates between the local and the global coordinate system
- *zoomX zoomY zoomZ* are the scaling factors of the part in the three directions, default is 1
- *globSym* is the local symmetry identificator, i.e. if a part is self-symmetrical with respect of a plane of the local coordinate system

Point coordinates are ordered in numbers separated from spaces, first come the x-coordinate of the first point of the first line, then the y-coordinate, then the z-coordinate; then we have the x-coord of the second point of the first line and so on. Points are to be put in an ordered way from the first point of the first line to the last point of the first line, then introducing points of the second line and so on. A newline can be put after each point or contour line definition for better readability.

As for the symmetry, value of the identifiers are provided here:

- 0: not symmetrical
- 1: symmetrical about X-Z plane
- 2: symmetrical about X-Y plane
- 3: symmetrical about Y-Z plane

Symmetry option makes you able to define, for example, the right wing of a space plane by defining only the points of the left one and then choosing a global symmetry about the X-Z plane, or the points of a fuselage by defining only the right half, and mirroring them around the local symmetry plane. These operations will be performed automatically by the RAM v1.0 algorithm, in the pre-processing procedure, which uses the stand-alone program WgsReader, based on Dirkx STA RAM v1.0 C++ classes, that contain all the functions and routines apt to perform the mirroring, translations, rotations and scaling defined in a *.wgs file, as will be explained in sec. A.3.1.

1.6.3 Centroids mesh

To succesfully calculate pressure distribution and other quantities, such as velocity vectors, normal vectors, and associated areas, have to be defined. In order to characterize these quantities over the mesh generated by the LaWGS format, we will use the originary points mesh $\mathbf{x}_{p,ij}$, as generated by the *.wgs file processing, to define panels, to be used to define local areas, normals, velocites. Each panel will be formed joining four adjacent point

$$\mathbf{x}_{p,\,ij},\ \mathbf{x}_{p,\,i+1,j},\ \mathbf{x}_{p,\,i+1,j+1},\ \mathbf{x}_{p,\,i,j+1}$$

The centroid will be calculated as the center of mass of the panel:

$$\mathbf{x}_{c,ij} = \frac{\mathbf{x}_{p,\,ij} + \mathbf{x}_{p,\,i+1,j} + \mathbf{x}_{p,\,i+1,j+1} + \mathbf{x}_{p,\,i,j+1}}{4} \tag{1.5}$$

Hence the panel diagonals will be formed:

$$\mathbf{d}_1 = \mathbf{x}_{p,\,i+1,j+1} - \mathbf{x}_{p,\,ij} \tag{1.6}$$

$$\mathbf{d}_2 = \mathbf{x}_{p,\,i+1,j} - \mathbf{x}_{p,\,i,j+1} \tag{1.7}$$

From diagonals, we can define the local centroid normal versor

$$\hat{\mathbf{n}}_{ij} = \frac{\mathbf{d}_2 \times \mathbf{d}_1}{\|\mathbf{d}_2 \times \mathbf{d}_1\|} \tag{1.8}$$

The versor just obtained is checked, in order to obtain a versor pointing outward the part surface.

The area of the panel, which will be associated to every centroid, is obtained from:

$$A_{ij} = \frac{1}{2}\,\|\mathbf{d}_2 \times \mathbf{d}_1\| \tag{1.9}$$

Tangential versors $\hat{\mathbf{t}}_{ij}$ will be obtained once the velocity surface will be defined, following the hypotheses in sec. 5.3.

The obtained centroids $\mathbf{x}_{c,ij}$ mantain the same ordering in contour lines, as can be seen in the next figure, where centroids mesh lines are plotted against the old point mesh lines. From now on, the centroid mesh will be the algorithm main mesh, and we will refer to it, since in the centroids we have the geometric characterization, i.e. $\hat{\mathbf{n}}_{ij}$,$\hat{\mathbf{t}}_{ij}$, A_{ij}, which will be used in our calculations.

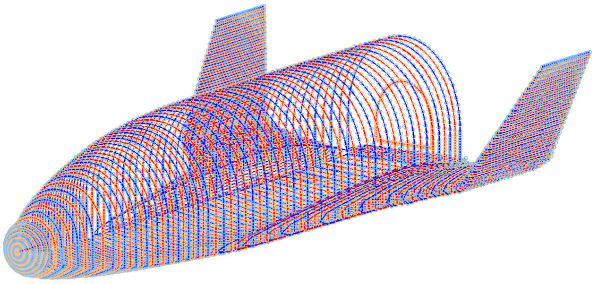


Figure 1.7: New centroid contour lines (red) plotted against old point contour lines (blue)

1.7 Development reference case - ESA Dart CFD simulations

We present here the reference case which has been used during the development of the algorithm, to confront the results of the different models, choosing the best options to shape RAM v2.0 solver in the most general and accurate way. The algorithm results, and in particular the viscous and heating models, were compared with the results provided by the ESA ESTEC Aerothermodynamic Division team. The data, courtesy of mr. Louis Walpot from the ESTEC Aerothermodynamic team, come from a CFD simulation using the Lore code, which is a Navier-Stokes solutor with high temperature, chemical non-equilibrium capabilities, under laminar hypotheses. The simulation has been conducted over a DART geometry, which is a testbed configuration for ESTEC CFD validations and experimental measurements. As we can see in the following figure, DART consist in a blunt nose, followed by a cylindrical region and a flare.

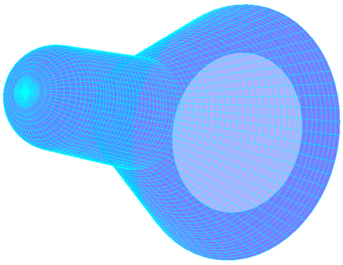


Figure 1.8: ESA Dart geometry

The reference case has been chosen at the following attitude and freestream conditions:

Table 1.1: Dart LORE CFD Freestream parameters

parameter	*value*
α	$0°$
β	$0°$
V_∞	$3146.94 \ \left[\frac{m}{s}\right]$
M_∞	10.4077
Pr_∞	0.75
P_∞	$1162.4854 \ [Pa]$
ρ_∞	$1.7801e^{-2} \ \left[\frac{kg}{m^3}\right]$
T_∞	$226.734 \ [K]$

The simulation, which includes a nitrogen and oxygen non-equilibrium thermochemical state reconstructor, provides several output databases, such as pressure coefficient c_p, friction coefficient c_f distribution, surface heat flow q_w, wall temperature T_w and some external flow values, such as M_e, ρ_e, T_e, V_e. We have then all the interesting quantities to be confronted with our program output, to verify the validity of the models tested during the development phase. We report here the solution for the surface distribution of c_p, c_f, q_w and T_w, which will be used in the following chapter as a reference for the validation of the algorithm and to justify the selection of a particular model among different choices.

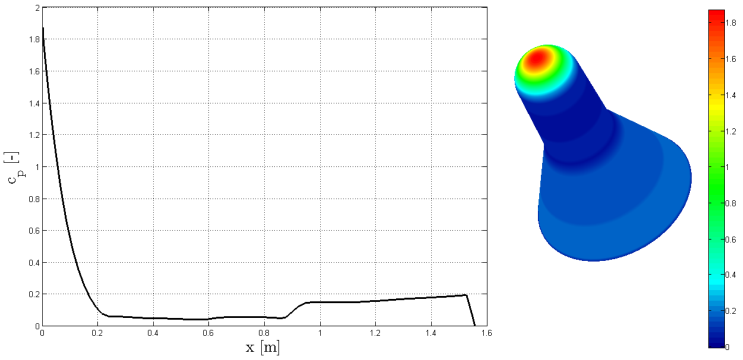


Figure 1.9: c_p distribution from ESA DART CFD laminar simulation - $M_\infty = 10.4077$, $\alpha = 0°$, $\beta = 0°$

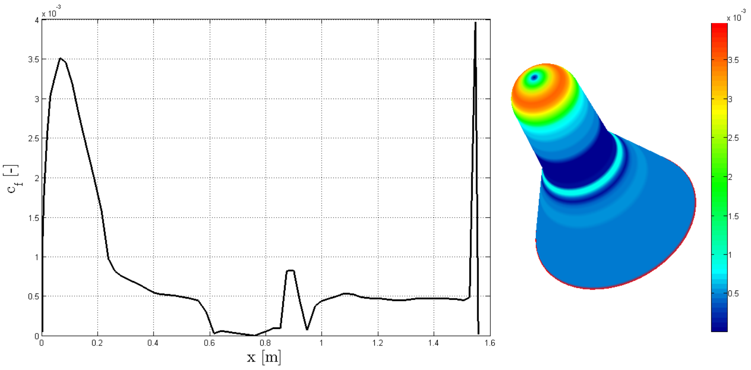


Figure 1.10: c_f distribution from ESA DART CFD laminar simulation - $M_\infty = 10.4077$, $\alpha = 0°$, $\beta = 0°$

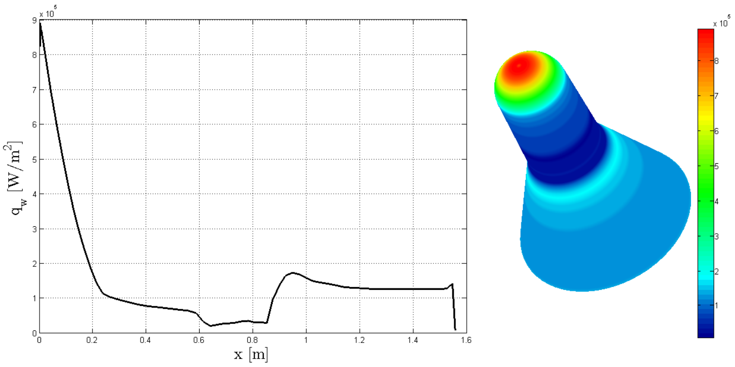


Figure 1.11: q_w distribution from ESA DART CFD laminar simulation - $M_\infty = 10.4077$, $\alpha = 0°$, $\beta = 0°$

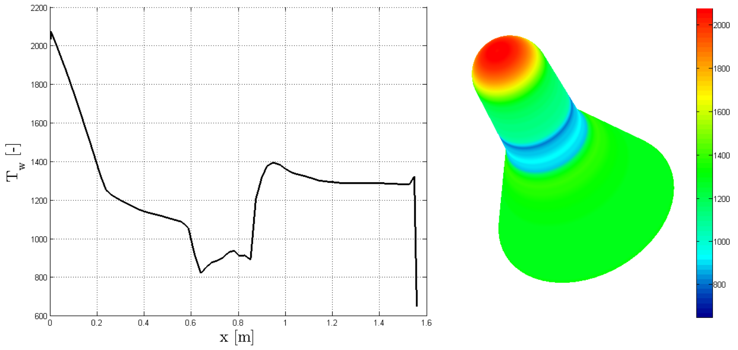


Figure 1.12: T_w distribution from ESA DART CFD laminar simulation - $M_\infty = 10.4077$, $\alpha = 0°$, $\beta = 0°$

Chapter 2

Models for inviscid hypersonic flow

2.1 Local surface inclination methods

This chapter is dedicated to the description of the approximate methods used to obtain the inviscid external flow, which will be used as the boundary condition for the viscous characterization of the surface. Even if we call it *external,* it will be solved only on the proximity of the wall. No calculations on the flow field surrounding the body will be made, as the local methods presented here need only surface geometrical data in order to describe the inviscid wall flow. As we have anticipated during the introduction, all of these methods rely on the assumption of thin shock layer, i.e. the consideration that, due to the proximity of the oblique shock to the vehicle surface, we can suppose that the flow directon changes istantaneously from the freestream heading to the tangential-to-freestream component. This hypothesis has been presented in 1.4.1 and its utilization is justified in CBAERO presentation by Kinney [5].

These approximated analyses are known as *Local Surface Inclination* methods, or LSI, because they rely only on the orientation of the panel with respect to the freestream direction, in order to obtain pressure distribution on the body, and have the enormous advantage that they don't require the solution of the flowfield surrounding the vehicle, and that they are completely local, so the solution on a single panel doesn't depend on the solution on other panels. The use of LSI analysis is typical of all the solvers which we've been discussing in the first chapter. In fact, all these tools are presented in the S/HAPB documentation [3], as well as in classical hypersonic textbooks [2]. Even the first development of the STA aerodynamic solver - denominated Re-entry Aerodynamic Module, or RAM - includes a brilliant implementation of these methods, as one can see in the release technical note [1]. We will present these methods in the following sub-sections, including a discussion on the theoretical justifications. We will divide these methods in two branches: the *compression methods*, for determining the pression on the windward side, where the fluid is compressed by the shock wave, and the *expansion methods*, which are suitable for the description of pressure distribution on the leeward side of the craft, where the gas expands.

2.2 Newton Methods

2.2.1 Newton method

The origin of the Newton method is based upon an amazing coincidence between the hypotheses that sir Isaac Newton formulated when considering a flat plate immersed in a freestream low-speed flow - which were confirmed to be wrong for this case - and the conditions that arise when considering an hypersonic flow. Newton described the freestream fluid as a stream of particles travelling togheter at the same speed, with no other kind of fluctuation or motion. He proposed that every particle, when impacting the front surface, would loose all of its normal momentum, conserving instead all the tangential momentum, like a bowling ball which enters in the side channel. As the particles are supposed to move straight and to change their direction only when impacting a surface, on the back side of a plate there would be no impact at all - i.e. the particles are not able to turn around the plate or body to reach its back.

This description has been proved to be too simplified for low speed aerodynamics, by misinterpreting the actual flowfield shape and totally neglecting the viscous effects. However, when Mach number rises, the shock moves close to the body, constraining the fluid to change its direction from freestream to tangential in a very narrow space, as one can see in the following figure, displaying streamlines over a 15° wedge at $M = 15$:

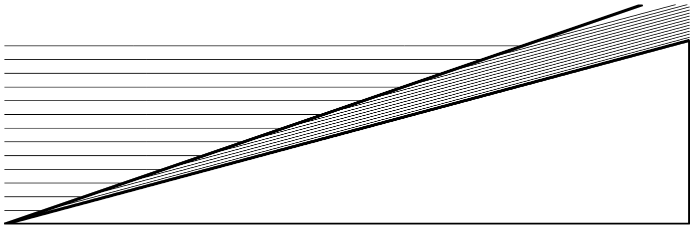


Figure 2.1: Streamlines over a 15°, $15\,M$ wedge

This kind of motion is very similar to the one proposed by Newton. We can then describe the flow against an inclined flat plate as a current of parallel particles with freestream velocity V_∞. The flat plate has an area A and an inclination θ with respect to the freestream flow.

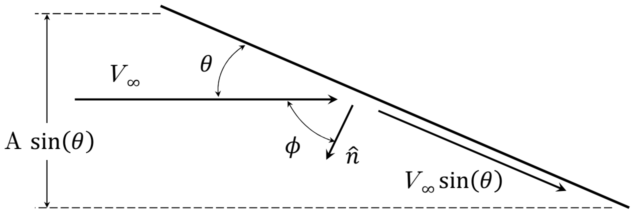


Figure 2.2: Inclined flat plate

The mass flux over the surface is then

$$\dot{m} = \rho_\infty V_\infty A \sin(\theta) \tag{2.1}$$

and, after the impact, the flow looses its normal velocity, experiencing a reduction of

$$\Delta V = V_\infty \sin(\theta) \tag{2.2}$$

so the time variation of the flow momentum becomes

$$\frac{dQ}{dt} = \dot{m}\Delta V = \rho_\infty V_\infty^2 A \sin^2(\theta) \tag{2.3}$$

by the second principle of dynamics, the time variation of momentum equals the force exerted on the body:

$$F = \frac{dQ}{dt} = \rho_\infty V_\infty^2 A \sin^2(\theta) \tag{2.4}$$

or

$$\frac{F}{A} = \rho_\infty V_\infty^2 \sin^2(\theta) \tag{2.5}$$

We have at this point the definition of a pressure on the body. But to which pressure we are referring? As we have previously said, by describing this flow we considered the fluid particles moving in a rectilinear direction, with no other kind of motion. The random molecular motion, which gives birth to the static pressure of the fluid, is then neglected. The pressure which we found from this flow is so not the absolute pressure, but the difference between the absolute pressure and the freestream static pressure, which was not considered before. We have then:

$$\frac{F}{A} = p - p_\infty \tag{2.6}$$

which considering eq. 2.5, becomes

$$p - p_\infty = \rho_\infty V_\infty^2 \sin^2(\theta) \tag{2.7}$$

or

$$\frac{p - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2} = 2\sin^2(\theta) \tag{2.8}$$

So, Newtonian theory defines the pressure coefficient on a body in a very simple way, using only the surface inclination:

$$C_p = 2\sin^2(\theta) \tag{2.9}$$

This relation is valid only for the region of the body which does directly impact the hypersonic flow, i.e. the windward side of an hypersonic vehicle, is then a *compression* formula. The simplyfied model of the flow we presented here describes the flow as an impact of molecules. Particles cannot then turn behind the body and impact the shadow region, or the leeward side of the vehicle, depicted for the IXV geometry in the next figure.

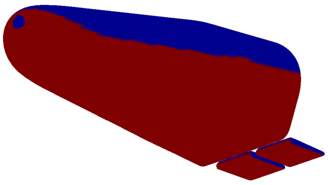


Figure 2.3: Shadow region over an IXV geometry for a 45° AOA reentry

In this region, we have no impact, hence no modification of freestream pressure is possible. On the leeward side we have then the following *expansion* model

$$C_p = 0 \tag{2.10}$$

A small consideration on the definition of inclination angle θ: if the surface panel has a normal $\hat{n}$, and the angle of the normal vector to the freestream velocity vector is ϕ, we obtain that the surface inclination is given by, as can be seen in fig. 2.2:

$$\theta = \pi - \phi \tag{2.11}$$

The leeward region, or shadow region, is hence defined by:

$$\theta < 0 \tag{2.12}$$

2.2.2 Corrected Newton method

Pure Newtonian theory returns accurate results only for $\gamma \to 1$, $M \to \infty$, which is an hypersonic limit (see chap 3.5 of [2]). For hypersonic flows which don't reach exactly these limits, Newton method tends to overestimate stagnation pressure values after the normal shock in the front region of blunt bodies. At the same moment, the shadow region imposition of $C_p = 0$ can underestimate the value in the back, since the value of freestream pressure is reached after a certain distance from the nose. A correction of this method is proposed by Lester Lees, and reconstruct the right pressure value behind the shock, exploiting the normal shock relations. The Newton law sinus squared law is replaced by

$$C_p = C_{p,\max} \sin^2(\theta) \tag{2.13}$$

where $C_{p,\max}$ is the value of the pressure coefficient between freestream condition and nose, after-shock stagnation pressure:

$$C_{p,\max} = \frac{p_{0,2} - p_\infty}{\frac{1}{2}\rho_\infty V_\infty^2} \tag{2.14}$$

the value of total stagnation pressure behind a normal shock is given, *for a perfect ideal gas*, by the Rayleigh pitot tube formula:

$$\frac{p_{0,2}}{p_\infty} = \left[\frac{(\gamma+1)^2 M_\infty^2}{4M_\infty^2 - 2(\gamma-1)}\right]^{\frac{\gamma}{\gamma-1}} \left[\frac{1-\gamma+2\gamma M_\infty^2}{\gamma+1}\right] \tag{2.15}$$

By expressing $\frac{1}{2}\rho_\infty V_\infty^2$ as $\frac{\gamma}{2}p_\infty M_\infty^2$ we obtain:

$$C_{p,\max} = \frac{2}{\gamma M_\infty^2}\left[\frac{p_{0,2}}{p_\infty} - 1\right] \tag{2.16}$$

which becomes:

$$C_{p,\max} = \frac{2}{\gamma M_\infty^2}\left\{\left[\frac{(\gamma+1)^2 M_\infty^2}{4M_\infty^2 - 2(\gamma-1)}\right]^{\frac{\gamma}{\gamma-1}} \left[\frac{1-\gamma+2\gamma M_\infty^2}{\gamma+1}\right] - 1\right\} \tag{2.17}$$

Notice that eqn. (2.17) will not be used to determine $C_{p,\max}$ in our algorithm, and its reported only for the sake of completness. This formulation doesn't hold anymore when considering *high temperature effects* in the solving of the hypersonic flow. The determination of after-shock properties such as $C_{p,\max}$ performed with perfect gas hypotheses, such as in eqn. (2.17), or pressure in (2.15), leads to completely wrong results for high gas temperatures, since the huge distortion of gas properties due to dissociation effects, which are largely present at these temperatures, is not taken into account. The determination of after shock properties for the viscous solutor part will be then performed used a chemically reacting gas mixture model, combined with shock relations, as we will see in more detail in chapter 4. Currently only Newton-modified pressure distribution are used, but this consideration must be applied also to the other shock relation based methods that we will see in this chapter, when applying shock based formulas like, for example, (2.20).

2.3 Tangent wedge/cone method

We present here a classical *compression* method used for determining pressure over slender, pointed nose bodies immersed in hypersonic flows, with zero angle of attack. The method is referred as tangent wedge method when is used over a bi-dimensional shape, or tangent cone if it is used for three-dimensional bodies. As said before, the requirements of this methods are that the body considered is slender and pointed, and we add the fact that the local surface angle θ_i has to be less than the maximum deflection angle for the local flow θ_{max}. Assuming these hypotheses, we present at first the tangent wedge method for 2D shapes - i.e. wing leading edges or similar:

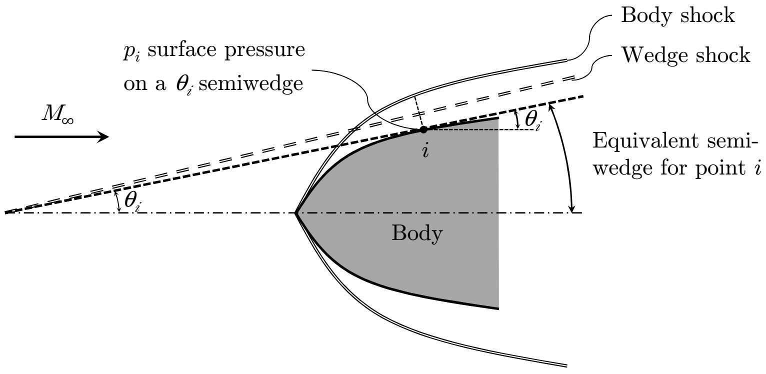


Figure 2.4: Tangent wedge method

Let us consider a point i on the body surface, on which we want to know the pressure. In order to find the pressure value, we imagine a semiwedge, tangent to the body in the point i, which form an angle θ_i with the freestream flow, and we calculate the pressure from the supersonic oblique-shock exact relations for a wedge of angle θ_i. This is valid for a bi-dimensional body. If we are dealing with a three-dimensional shape, we can extend the argumentation, using cone shock relations instead of oblique-shock relations. Such kind of relations can be found tabled, for example, in [7]. The tangent cone method is exactly the analogous of the tangent wedge, but we will consider equivalent cones at point i, instead of wedges.

Let us consider now the theoretical founding of these two methods. In fact, these methods don't stem directly from a model of the hypersonic flow, as it was in the Newton method, but they come from pure intuition. The results have proven to be quite accurate, although, and we will try to understand why. Consider the following figure, relative to the tangent wedge method, magnifying the region surrounding the point i:

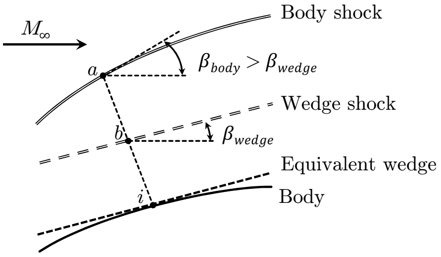


Figure 2.5: Tangent cone method - point i

In an hypersonic flow around an oblique shock over a slender body, we have that the v component of the flow is changed more strongly than the u component. In fact, in an hypersonic current, the shock are very inclined with respect to the free-stream, if θ is small, and since only the velocity component normal to the shock is modified, we have that the u component is only slightly modified. It is possible to demonstrate that, in an hypersonic flow:

$$\frac{\Delta u}{V_\infty} = \mathcal{O}\left(\theta^2\right)$$

$$\frac{\Delta v}{V_\infty} = \mathcal{O}\left(\theta\right)$$

so, for small angles of deflection we have a predominant velocity change in the direction normal to the shock. This implies, by recalling Euler equation $dp = -\rho V\, dV$, that the pressure gradient is almost aligned with the normal to the body direction. Looking at figure 2.5 we can then understand that the pressure in i depends mainly from the pressure value behind the shock in a. The pressure in i should be less than in a, since we have centrifugal effects due to the body curvature, so $p_i < p_a$. If we consider the wedge-shock instead of the actual body shock shape, we obtain a less inclined shock: $\beta_{wedge} < \beta_{shock}$. We have then, by considering the equivalent wedge, a less inclined, thus less intense, shock. This results in a lower pressure behind a wedge shock: $p_b < p_a$. This is coherent with the presence of the centrifugal effects, and gives the rule-of-thumb justification for these two methods, partially explaining why they have proven to be accurate, even with no true theoretical foundation.

We conclude with a simple observation, which is very trivial but must be taken into account, on the applicability of these methods. The fact is that the tangent wedge/cone method provide accurate results only for zero angle of attack bodies, so if we have a pointed-nose body with incidence, we can use these methods to calculate the zero AOA pressure value, then correct it by adding the contribution of the α angle, using the Newtonian theory model, as proposed in paragraph 4.8 of [2]:

$$C_p = C_{p,\alpha=0^\circ} + 2sin^2(\alpha) \tag{2.18}$$

If α is small, however, it is possible that the global θ_i is still less than the maximum deflection angle for the given freestream flow, so that a wedge/cone solution is applicable on the whole surface. Another consideration which has to be made is that these methods are obviously inaccurate for blunt bodies, but they become accurate at a certain distance from the nose, where the shock and the flowfield tend to be more similar to those of a sharp body, so that these tangent methods can be applied with reasonably limited error.

2.4 Prandtl-Meyer Expansion method

The Prandtl-Meyer Expansion method is based upon almost the same hypotheses as the tangent wedge/cone method, because it is based on a very similar concept, although its theoretical bases are more solid. It requires an attached shock, which is proper of sharp-nosed bodies, and a local angle less than the maximum deflection angle, and this means that this method is valid only for low or null angle of attack. The theory which describes this method is very straightforward: it simply calculates the flow properties after the shock and consider the subsequent expansion by mean of the exact theory.

- The Mach number and pressure just after the nose shock, M_n and p_n are calculated from the exact oblique shock theory.

- We want to calculate the pressure at a generic downstream point i on the body surface. We assume then a Prandtl-Meyer expansion to calculate the Mach number at point i:

$$\Delta\theta = \sqrt{\frac{\gamma+1}{\gamma-1}}\left[atan\left(\sqrt{\frac{\gamma-1}{\gamma+1}}\left(M_n^2-1\right)\right) - atan\left(\sqrt{\frac{\gamma-1}{\gamma+1}}\left(M_i^2-1\right)\right)\right]$$

$$-\left[atan\left(\sqrt{(M_n^2-1)}\right)-atan\left(\sqrt{(M_i^2-1)}\right)\right] \tag{2.19}$$

where $\Delta\theta = \theta_n - \theta_i$, which implies $\theta_i < \theta_n$ as an additional hypothesis.

M_i is found by the numerical solving of the equation for assigned $\Delta\theta$ and M_n, e.g. with the application of a tangent Newton Rhapson algorithm (we are referring here to the method for the numerical solving of a non-linear function, not to the just presented pressure determination Newton method!).

- Once M_i is known, we can then easily calculate pressure at point i via the exact isoentropic expansion formula, from $p_0 = const.$ after the shock

$$\frac{p_i}{p_n} = \left[\frac{1+\frac{\gamma-1}{2}M_n^2}{1+\frac{\gamma-1}{2}M_i^2}\right]^{\frac{\gamma}{\gamma-1}} \tag{2.20}$$

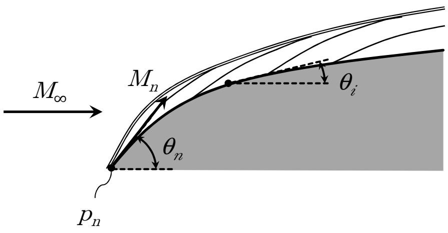


Figure 2.6: Prantdl-Meyer Expansion Method

Let us consider now the validity of this method. As already stated before, one of the requirements is that the angle of attack is zero or small, because of the maximum deflection angle constraint for an attached shock. Another fact that must be stressed is that the Prandtl-Meyer expansion gives correct results only in the case of hypersonic flows. In our approach we have in fact neglected the evidence that, when the expansions characteristics reach the shock, they are reflected as compression characteristics, thus modyfing the flow field. The hypothesis of a perfect, iso-entropic expansion is then not valid on the surface when these compression characteristics reach the surface again. However, if we consider the flow as hypersonic, the shock wave is very close to the body. The expansion characteristics travel then for a long distance before being reflected, and the subsequent compression characteristics hit the body wall very far from the nose. Evidence from [2] shows that in the nose region of a sharp-nosed profile the Shock/Prandtl-Meyer expansion method grants very accurate results, which slowly move away from the exact solution as we walk away from the nose. This combined *compression/expansion* method is derivated over an exact bi-dimensional solution, so it is particularly suitable for sharp-edged wings.

For bodies in incidence, the Prandtl-Meyer expansion can be combined with the Newton method or Modified Newton method in order to treat wider angles of attack, as already did in 2.18 for the tangent cone/wege method. We can use also the Modified Newton for calculating the pressure value in the nose region of a blunt body, and then apply a Prandtl-Meyer expansion downstream, thus resolving even blunt bodies with isoentropic exact expansion, always remembering the fact that this consideration is made possible by the fact that we have a thin shock layer. This method is called *Modified Newtonian/Prandtl-Meyer* expansion. We stress the fact that isoentropic relations can be used only with *frozen mixture* hypothesis, and will grant acceptable results only if a correct γ value is adopted. This value can be obtained by treating the shock with a chemial equilibrium reacting mixture model.

2.5 High Mach number base pressure

We present here another semi-empirical methods to esteem the value of pressure in the leeward region of bodies, as can be seen in [1]. Summing up to this point, we have seen the generic Newton method leeward free-stream pressure hypothesis - i.e. $C_p = 0$ - and the Prandtl-Meyer expansion for sharp edged bi-dimensional bodies. This other *expansion* method comes from a correction of the *vacuum pressure coefficient,* where the vacuum hypothesis is assumed to describe expansion after a steep change in the geometry, such as there can be in a flat back of a missile - with rocket boosters off - or a capsule.

Remembering the C_p expression in terms of Mach number:

$$C_p = \frac{p - p_\infty}{{}^1\!/\!{}_2\rho_\infty V_\infty^2} = \frac{p - p_\infty}{{}^1\!/\!{}_2\gamma p_\infty M_\infty^2} \tag{2.21}$$

we have, in the void ($p = 0$), the *vacuum expansion formula:*

$$C_p = -\frac{2}{\gamma M_\infty^2} \tag{2.22}$$

Empirical correlations have shown that, in an high hypersonic flow with viscous effects, such as recirculation at the vehicle rear, this relation can be corrected in what is known as the *high Mach number base pressure formula:*

$$C_p = -\frac{1}{M_\infty^2} \tag{2.23}$$

This method can be used to describe the rear of blunt vehicles.

Chapter 3

Models for viscous hypersonic flow

3.1 Reference enthalpy methods

At this moment, we have determined in an approximate way the inviscid flow over the vehicle surface. To calculate the effects of viscosity we need to characterize in some way the properties of the boundary layer surrounding the body. The mathematical solution of the compressible boundary layer has been studied in depth, at least for the laminar flow. Exact solutions for the self-similar, laminar flow have been derived from the governing boundary layer equations, either for the flat plate and the stagnation region case. Algorithms for the calculation of a non self-similar current have been investigated, and finite difference solutions have been tested. For an in depth look at these subjects, a good presentation can be found in chapter 6 of [2]. These methods are however too sophisticate for the philosophy of our algorithm, so we have to find a way to define the influence of the boundary layer on our flow without solving the exact equations, which would be prohibitive in terms of computational cost and time, considering the aims of our program.

Fortunately, engineering empirical correlations and approximate calculations linked to the description of the viscous properties of an hypersonic or supersonic body, knowing only the external inviscid flowfield, have been already developed in literature. The first complete analysis of this kind had been elaborated in the Fifties by Eckert [8], then in more recent days Meador and Smart [9], and Simeonides [10],[11], proposed new studies and theoretical justifications on the work of Eckert. The main idea underlying the studies of these authors is to link the compressible boundary layer analysis to the theoretic relations obtained for the incompressible laminar and turbulent flows. The charactheristic numbers and formulas used for the description of the boundary layer, e.g. the friction coefficient c_f calculation for the shear stress, or the Stanton number C_H for the heat transfer, will have the same formulation as in uncompressible flows. So in which way compressible effects are included in the description?

The key of these methods is that the whole of the coefficents, characteristic numbers, thermodinamic variables, and other values that identify the boundary layer state are calculated at a *certain enthalpy*, representative of the internal energy of the compressible boundary layer, taken somewhere inside the layer. For this reason these methods are gathered under the name of *reference enthalpy methods*. Obviously, these approximate methods are intended for preliminar, quick design calculations, for the great simplicity they grant, since they are based on quasi-one-dimensional analysis and on averages of the thermodinamic properties across the boundary layer. These methods show a great dependence over Re_x , and this is the main reason which justifies streamline calculation, which is then mandatory for obtaining the correct curvilinear coordinate x.

We present here the formulations proposed by these three authors, from the simplest and oldest idea to the most generalized and recent one. In the implementation of our algorithm, we have first used the Meador - Smart reference method, which is based only on the nature of the flow and not on the geometry, and then some relations from Simeonides, to take account of the geometry of our vehicle, expecially when considering the stagnation region. Eckert method has been implemented as a part of Simeonides method. We will present it as first to give an idea of how a reference enthalpy method is composed.

3.2 Eckert's reference temperature method

Eckert was the first to propose a method based on the concept of a reference temperature indicative of the energy status inside the boundary layer. We describe here his work as it is presented in [2].

First, we define the formulas for the friction coefficients and Stanton number, which are taken from the flat plate hypoteses.

Table 3.1: Coefficients calculation for Eckert method

Incompressible Flat Plate	*Compressible Flat Plate*
laminar	
$c_f = \frac{0.664}{\sqrt{Re_x}}$	$c_f^* = \frac{0.664}{\sqrt{Re_x^*}}$
$C_f = \frac{1.328}{\sqrt{Re_c}}$	$C_f^* = \frac{1.328}{\sqrt{Re_c^*}}$
$C_H = \frac{0.332}{\sqrt{Re_x}} Pr^{-2/3}$	$C_H^* = \frac{0.332}{\sqrt{Re_x^*}} Pr^{*-2/3}$
turbulent	
$c_f = \frac{0.0592}{(Re_x)^{0.2}}$	$c_f^* = \frac{0.0592}{(Re_x^*)^{0.2}}$
$C_H = \frac{c_f}{2s}$	$C_H^* = \frac{c_f^*}{2s^*}$

As one can see, the formulation doesn't change between the compressible and incompressible case. The only difference is that the values marked with an asterisk are calculated at a given reference temperature T^*, i.e.:

$$c_f^* = \frac{\tau_w}{\frac{1}{2}\rho^* u_e^2} \tag{3.1}$$

$$C_f^* = \frac{D_f}{\frac{1}{2}\rho^* u_e^2 S} \tag{3.2}$$

$$C_h^* = \frac{q_w^*}{\rho^* u_e \left(h_{aw}^* - h_w^*\right)} \tag{3.3}$$

$$Re_x^* = \frac{\rho^* u_e x}{\mu^*} \tag{3.4}$$

$$Re_c^* = \frac{\rho^* u_e c}{\mu^*} \tag{3.5}$$

$$Pr^* = \frac{\mu^* c_P^*}{k^*} \tag{3.6}$$

with $\rho^* = \rho\left(T^*\right)$, $\mu^* = \mu\left(T^*\right)$ and so on. u_e is the inviscid velocity calculated outside the boundary layer. From now on, with the e subscript we denote the values calculated from inviscid theory just outside the BL.

We need then a relation for the reference temperature, that is obtained from empirical correlation, in Eckert's work:

$$\frac{T^*}{T_e} = 1 + 0.032\, M_e^2 + 0.58\left(\frac{T_w}{T_e} - 1\right) \tag{3.7}$$

Given the external flow with T_e, M_e and the wall temperature of the vehicle T_w we are in the condition to calculate T^* and all the reference variables, obtaining the compressible friction coefficient c_f^* and the compressible Stanton number C_H^* in every point of the surface.

We remember here that these approximations are valid for a flat plate. For conical geometries and laminar flows, [2] proposes to multiply the local skin-friction coefficient and Stanton number by the Mangler fraction $\sqrt{3}$, in order to take account of the rise in the skin friction and heat transfer due to the thinning of the boundary layer in a three dimensional geometry.

3.3 Meador Smart reference enthalpy method

In 2005, fifty years later than the work of Eckert, William E. Meador and Michael K. Smart extended the concept of reference temperature [9], taking as the main parameter to characterize the boundary layer the *reference enthalpy ratio*:

$$g^* = \frac{h^*}{h_e} \tag{3.8}$$

With respect to Eckert's work, this study poses the selection of the reference enthalpy formulas on a theoretical ground, elaborating a definition of the reference enthalpy which is not based anymore on empirical correlations, but on analytical elaborations of the exact boundary layer equations. For the laminar flows, the enthalpy ratio is obtained from the Van Driest enthalpy profile $g(f)$, where f is the non-dimensionalized velocity profile in the BL equations; and from Whitfield and High's first-order theory for turbulent boundary layers. Basing their work on these assumptions, Meador and Smart propose an average of the local enthalpy ratio over the boundary layer velocity profile. Two different formulas for the reference enthalpy, one for laminar and another for turbulent flows, are obtained. For the laminar we obtain:

$$\frac{T^*}{T_e} = 0.45 + 0.55\,\frac{T_w}{T_e} + 0.16\,r\left(\frac{\gamma-1}{2}\right)M_e^2 \tag{3.9}$$

while for a turbulent flow we have:

$$\frac{T^*}{T_e} = 0.5\left(1 + \frac{T_w}{T_e}\right) + 0.16\,r\left(\frac{\gamma-1}{2}\right)M_e^2 \tag{3.10}$$

with $r = \sqrt{Pr^*}$ as the adiabatic recovery factor.

Obviously these formulas are intended for a perfect gas with costant γ and c_p. For the laminar flow, formulas for obtaining shear stress are the same as Eckert's work, while for turbulent flows, the authors propose the following:

$$c_f^* = \frac{0.02296}{(Re_x^*)^{0.139}} \left(\frac{\rho^*}{\rho_e}\right)^{0.861} \left(\frac{\mu^*}{\mu_e}\right)^{0.139} \tag{3.11}$$

On the other hand, when presenting this method, Anderson [2] proposes the following expressions for the local and global (flat plate) skin-friction coefficents:

$$c_f^* = \frac{0.02296}{(Re_x^*)^{0.139}} \tag{3.12}$$

$$C_f^* = \frac{0.02667}{(Re_c^*)^{0.139}} \tag{3.13}$$

For the determination of the compressible turbulent Stanton number C_H^*, it should be obtained from the skin-friction coefficient using the Reynolds analogy. We refer to section 3.6 for this discussion. However, knowing c_f^* and obtaining C_H^* from Reynolds analogy, we have analyzed the viscous flow over the surface.

The preceding equations 3.9-3.11 have been used in a previous implementation of the algorithm, even if they rely on the hypothesis of perfect gas. As the authors already proposed in their work, the reference enthalpy can be generalized by replacing perfect gas γ and Pr with their reference corrispectives, calculated with a gas mixture chemical equilibrium solutor, as we do in our work:

$$\gamma^* = \frac{C_p\left(T^*\right)}{C_v\left(T^*\right)} \tag{3.14}$$

$$Pr^* = \frac{C_p\left(T^*\right)\ \mu\left(T^*\right)}{k\left(T^*\right)} \tag{3.15}$$

The equations 3.9, 3.10 become, in a more generalized presentation, relating them to laminar and turbulent flow respectively:

$$g^* = 0.45 + 0.55\,\phi + 0.16\,r\left(\frac{\gamma^*-1}{2}\right) M_e^2 \tag{3.16}$$

$$g^* = 0.5\left(1+\phi\right) + 0.16\,r\left(\frac{\gamma^*-1}{2}\right) M_e^2 \tag{3.17}$$

with $\phi = {}^{h_w^*}\!/_{h_e}$. This formulation is used in order to treat chemical reacting mixtures.

Meador Smart method gives slight enhancement in the results with respect to Eckert work, but its main virtue is to pose theoretical basis to the reference enthalpy formulation. We remember that the authors underline that their formulas are valid only in an engineering approximation sense, and we underline the fact that this method is presented only for the flat plate approximation, hence introducing further exstimation error in our model. To quantify in some way the approximation introduced by a reference enthalpy method, we present here the comparison of the friction factor and the Stanton number of Eckert and Meador Smart compared with numerical solutions over a flat plate, as is presented in [9].

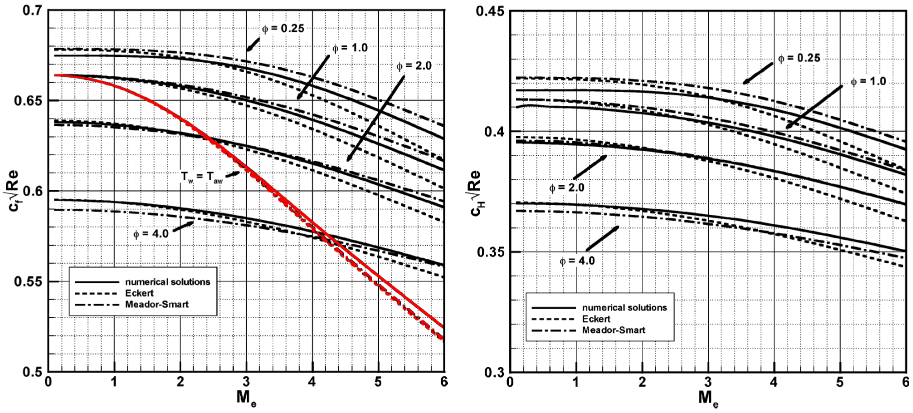


Figure 3.1: Laminar C_f and C_H prediction comparison with numerical solutions

In these two graphs we can see the comparison over numerical laminar flat plate results. We want to stress two facts that emerge from the analysis of these results. The first one is evident in the left image, which represents the confront between $c_f\sqrt{Re}$ at different Mach numbers. If we impose a value of ϕ, we can see that the numerical solutions tend to lie between Eckert and Meador Smart methods. But if we focus on the adiabatic solution - i.e. the solution imposed in our algorithm - which is the red line in the graph, we can see that both these reference enthalpy methods *underestimate* friction factor, and hence Stanton number, as the Mach number grows. A certain *underestimation* of these coefficients at high Mach numbers typical of the hypersonic flow is to be expected.

We conclude the Meador Smart method presentation by stressing the fact that this method has been developed under *flat plate* hypoteses, which have been used for g determination. We will present in the next section a more generalized method, that is currently used by RAM v2.0 viscous solutor, which takes into account different geometrical shapes, two and three-dimensional fields, junctions and other features.

3.4 Simeonides generalized reference enthalpy method

3.4.1 Reference enthalpy method

The main limitation of the methods presented until this moment is that they are referred at the flat plate case. This is a good initial approximation, which has been proven quite accurate for the estimation of friction factor in regions which are far enough from the stagnation region. In the stagnation region, or in other regions of the vehicle which are characterized by different geometries, such as wings leading edges, the relations which produce the friction factor or the Stanton heating coefficient can be signficantly different. The effect of three-dimensional flow fields has also to be taken into account.

In 1996 a collection of different empirical correlations, including Eckert's work as well as other similar enthalpy correlations for different geometries was assembled by Simeonides ([10],[11]), which elaborated a joint algorithm to describe friction and heating based on these formulations. In his work, Simeonides found different geometric and fluid-dynamic criteria in order to choose a proper formulation for the part of the flow which was to be analyzed.

To obtain a better description of the boundary layer physics, the author proposed also a characterization of the running length of the flowfield based on the effective boundary layer growth length. In these cases, Simeonides proposes a very intuitive method to adjust the boundary layer growth over vehicle junctions, which is to redefine the current boundary layer growth, length, i.e. the reference length L, to the *effective boundary layer origin*. L becomes then the *effective boundary layer growth length*, i.e. the length the boundary layer should have if the boundary layer origin had been placed as if the boundary layer was growing over the same generic surface from its effective origin. At the present state of the work, this Simeonides' proposal has not been included in our algorithm, but the RAM v2.0 structure is suitable for this extension in future work, so we will present only its generalization of reference enthalpy methods to generic geometries and flow regimes. A junction method description can be found in [10], and an extended exposition is presented in [11].

The core of Simeonides method is the definition of an unique adimensional formulation for the skin-friction coefficient and for the Stanton number, where the appropriate correlations for the particular flow nature or geometry are triggered by appropriate coefficients. We recall the definition of some coefficients in order to present a clear definition of Simeonides main relation.

At first we define the *local skin-friction coefficient* c_f, which is the main indicator of the flow viscous behaviour

$$c_f = \frac{\tau_w}{{}^1\!/\!_2\,\rho_\infty u_\infty^2} \tag{3.18}$$

where the subscript ∞ refers to freestream conditions, which in our algorithm are the inviscid results, and the subscript w refers to wall conditions. This coefficient must not be confused with the very similar *friction factor*, which is used in Simeonides' exposition:

$$f = \frac{\tau_w}{\rho_\infty u_\infty^2} \tag{3.19}$$

The heat flow characterization is obtained instead by the definition of the *Stanton number*:

$$St = \frac{q_w}{\rho_\infty u_\infty \left(h_{rec} - h_w\right)} \tag{3.20}$$

where h_{rec} is the *adiabatic recovery enthalpy,* sometimes presented in literature as h_{aw}, where the subscript aw is for *adiabatic wall.* Notice the fact that in Simeonides method, in contrast with the C_H Stanton coefficient definition proposed in sec. 3.2 for Eckert method, Stanton number is non-dimensionalized using *freestream quantities* instead of *external quantities.* The adiabatic recovery enthalpy,

$$h_{rec} = h_e + r\frac{u_e^2}{2} \tag{3.21}$$

which is also presented as

$$h_{rec} = \left(h_0 - \frac{u_e^2}{2}\right) + r\frac{u_e^2}{2} \tag{3.22}$$

that, if considering the fact that $h_0 = h_e + \frac{1}{2}u_e^2$, results as equivalent to (3.21).

The *recovery factor* r is defined as

$$r = \frac{h_{rec} - h_e}{h_0 - h_e} \tag{3.23}$$

and is proved that , even in the hypersonic case, it can be approximated as

$$r = \sqrt{Pr} \tag{3.24}$$

for the laminar case, or

$$r = Pr^{1/3} \tag{3.25}$$

for the turbulent flow, as it can be found in [10] and in [2], page 299.

We introduce here the *Reynolds analogy*, which will be presented in a exaustive way in sec. 3.6, which can be expressed, for a laminar flow, as:

$$St = f\ Pr^{-2/3} \tag{3.26}$$

or, alternatively, as

$$St = \frac{c_f}{2s} \tag{3.27}$$

where s is the Reynolds analogy factor, which for a laminar case is $s = Pr^{2/3}$.

Simeonides defines a common generalized expression for the friction factor and the Stanton number, which is reported as:

$$f = \frac{\tau_w}{\rho_\infty u_\infty u_{main}} = St\ Pr^{2/3} = \frac{q_w\ Pr^{2/3}}{\rho_\infty u_\infty \left(h_{rec} - h_w\right)} =$$

$$= C\left(\frac{\rho^*}{\rho_\infty}\right)^{(1-n)}\left(\frac{\mu^*}{\mu_\infty}\right)^{n}\left(\frac{u_{main}}{u_\infty}\right)^{(1-2n)}\left(u_{grad}\frac{L}{u_\infty}\right)^{n} Re_{\infty,L}^{(-n)} \tag{3.28}$$

This formulation can be reported in an adimensional form by multiplying for $Re_{\infty,L}$:

$$\frac{\tau_w L}{\mu_\infty u_{main}} = \frac{q_w\ L\ Pr^{2/3}}{\mu_\infty \left(h_{rec} - h_w\right)} =$$

$$= C\left(\frac{\rho^*}{\rho_\infty}\right)^{(1-n)}\left(\frac{\mu^*}{\mu_\infty}\right)^{n}\left(\frac{u_{main}}{u_\infty}\right)^{(1-2n)}\left(u_{grad}\frac{L}{u_\infty}\right)^{n} Re_{\infty,L}^{(1-n)} \tag{3.29}$$

This formulation does not assume the calorically perfect hypothesis, so it is perfect for the use with the equilibrium hypothesis adopted in our algorithm. Simeonides presents other formulations for perfect gases, which we will not present here for this reason, taking (3.29) as the main reference. We note that eqn. (3.29) depends on a series of coefficients and definitions which are determined by the flow nature and by the particular geometry of the flowfield region. We report here these definitions in two tables, one for the laminar and the other for the turbulent flow:

Table 3.2: Simeonides coefficients and variables definition - Laminar flow

		Laminar Flow (n = 0.5)				
		2D stag. region	Axis. stag. region	Infinite sweep l.e.	Planar b.l.	Conical b.l.
Constant	C	0.570	0.763	0.570	0.332	0.576
Reference density	ρ^*	ρ_e or $(\rho_e^{0.8}\rho_w^{0.2})$			@ Eckert's h^* and P_e	
Reference viscosity	μ^*	μ_e or $(\mu_e^{0.8}\mu_w^{0.2})$			@ Eckert's h^* and P_e	
Main b.l. edge velocity	u_{main}	Term drops out of the RHS in laminar flow, u_e on the LHS remains for shear stress				
Local velocity gradient	u_{grad}	$\frac{du_e}{ds}$ normal to attachment line			$\frac{u_e}{x}$	
Length scale	L	Leading edge / nose radius of curvature R			Surface length, L	

Table 3.3: Simeonides coefficients and variables definition - Turbulent flow

Turbulent Flow (n = 0.2)

		2D stag. region	Axis. stag. region	Infinite sweep l.e. *(n = 0.21)*	Planar b.l.	Conical b.l.
Constant	C	0.040	0.042	0.0345	0.0296	0.0348
Reference density	ρ^*	ρ_e or $(\rho_e^{0.8}\rho_w^{0.2})$		@ Poll's h^* and P_e	@ Eckert's h^* and P_e	
Reference viscosity	μ^*	μ_e or $(\mu_e^{0.8}\mu_w^{0.2})$		@ Poll's h^* and P_e	@ Eckert's h^* and P_e	
Main b.l. edge velocity	u_{main}	u_e along local flow direction		v_e along attachment line	u_e along local flow direction	
Local velocity gradient	u_{grad}	$\frac{du_e}{ds}$ normal to attachment line			$\frac{u_e}{x}$	
Length scale	L	Leading edge / nose radius of curvature R			Surface length, L	

We note that Simeonides proposes different expressions of the reference quantities. For example, the stagnation region density and viscosity values are often taken as the external values, or, after the analysis of Fay & Riddel (1958), as a weighted product of the wall and external values: $\rho^* = \rho_e^{0.8}\rho_w^{0.2}$ and $\mu^* = \mu_e^{0.8}\mu_w^{0.2}$, while in the turbulent case, we have the differentiation between Eckert classical formulation for the reference enthalpy, which is

$$h^* = 0.28\,h_e + 0.50\,h_w + 0.22\,h_{rec} \tag{3.30}$$

and the formulation proposed by Poll in 1992, which is considered more accurate in the infinite swept cylindircal leading edge, which more weighted recovery enthalpy, and lessens the wall influence:

$$h^* = 0.30\,h_e + 0.10\,h_w + 0.60\,h_{rec} \tag{3.31}$$

Another quantity definition which has to be cleared is the velocity gradient in the stagnation region. Simeonides proposes an approximated determination of the velocity gradient based on the Bernoulli equation:

$$u_{grad} = \frac{du_e}{ds} = \sqrt{\frac{2\left(p_e - p_\infty\right)}{\rho_e}} \tag{3.32}$$

whereas, as we will see in section 6.3 a numerical determination of u_{grad} is possible. Another approximated formulation for the stagnation region is given by:

$$u_{grad} = \frac{du_e}{ds} = \frac{1}{R} 414.67 \left(\frac{h_{tot} - h_w}{R_0 T_0}\right) \tag{3.33}$$

Another consideration to remember is that , when applying this method to swept leading edges, the relevant freestream velocity is

$$u_{\infty,swept} = u_\infty cos\left(\Lambda\right) \tag{3.34}$$

This method can then be applied on the whole vehicle, by taking care to choose the correct approximate formulation corresponding to the appropiate region of the body. In the validation of our algorithm, only a distinction between stagnation region and downstream region has been made, considering the stagnation region as axisymmetric for fusiform bodies and as two-dimensional for wings and non-fusiform bodies. For the rest of the vehicle, the considered correlation is the planar boundary layer relation. Dirkx [1] elaborated an automated algorithm to select and split into different parts the vehicle region, defining blunt and non-blunt regions, and curved and non-curved regions. At the present stage of the work a direct software integration has not been developed, and the blunt/slender differentiation is made manually by choosing the appropriate station between the stagnation region and the downstream part of the body.

We would like to clarify the indexes of (3.29) referring to our geometry. For the downstream part of the vehicle, i.e. the region downstream the stagnation region, surface length L is calculated as the *actual running length curvilinear coordinate* x in order to obtain the correct friction factor given by the actual distance travelled by fluid particles. In the downstream region the vehicle surface is hence considered as a single flat plate (or a curved plate for conical surfaces), where all the flow properties are calculated *along the streamline*, as well as the length L which is the distance from the stagnation point measured along the streamline curvilinear coordinate x, so in our formulation we will take

$$L = x \tag{3.35}$$

when we are dealing with planar or conical boundary layers.

The last consideration that we will stress in Simeonides work is about the contribution of viscous interaction. In his study Simeonides clearly states that these correlations mantain their validity since the viscous interaction effects are negligible. We can agree with this, since our algorithm is not intended to study external flow-field modifications due to increased boundary layer growth. However, we would like to rembember that, since Simeonides method is based upon Eckert's method, and that Meador and Smart showed that a certain consideration of viscous heating influence is included in the quadratic growth of the reference enthalpy ratio (see sec. 3.3) both for their method and for Eckert's, it is not completely true that all the high enthalpy flow viscosity effects are not represented.

3.5 Transition

Since the beginning of this chapter, we have proposed formulations and corrections which depended in a strong way from the nature of the flow. Indeed, laminar and turbulent flows carry significant differences which motivate the different correlations which have to be found in order to describe them properly, both in a numerical simulation or in an approximate solver like ours. And here we come to the front line on which aerodynamic research is holding from over a century. Ludwig Prandtl, while studying the subject of boundary layer in the beginning of the XX century, put his trust on the powerful mathematical tools discovered in those years, that in a few decades the problem of aerodynamic transition would be solved. By present day, we can definetively say that his prevision was too optimistic. As one of the most experienced hypersonic aerodynamcists, John D. Anderson Jr., says in our era (2006): *"Turbulence, and transition to turbulence, is one of the unsolved problems in basic physics"[2].*

We can certainly agree with that, and we add the consideration that, while for fully developed turbulent flows literature has developed several empirical or theoretical correlations to describe its behaviour, we can't say the same thing for turbulent transition. Transition keeps itself elusive, expecially in the hypersonic flowfield, where to the challenging study of fluid stability, other influences are added, since transition depends strongly over Mach number and thermo-fluid-dynamic properties of the flow. However, in order to take into account the development of a turbulent flow, a transition point guess, in an engineering approximation sense, has to be made. We propose here two correlations, as presented in [2], to provide an approximate criteria of transition. We would like to stress again the fact that they are provided just to make a very rough esteem of the transition point on our vehicle, as the only way to obtain reliable transition values nowadays is through experimentation or direct numerical simulation - which for our geometries and flow regimes is absolutely unaffordable. For a discussion on the current state of the art on transition, we recommend the reading of [12].

The main concept of the two formulations presented here is to define an appropriate Reynolds number, the *transition Reynolds number*, which is associated with a proper lenght x_T which identifies transition, along the curvilinear coordinate x.

$$Re_T = \frac{\rho_e V_e x_T}{\mu_e} \tag{3.36}$$

The first method to be presented here, which has found some use in hypersonic vehicle design - e.g. Space Shuttle transition determination - is based on the definition on a transition Reynolds number based on the boundary layer momentum thickness:

$$Re_{\theta_T} = \frac{\rho_e V_e \theta_T}{\mu_e} \tag{3.37}$$

where θ_T is the *transition boundary layer momentum thickness,* the boundary layer momentum thickness being defined as:

$$\theta = \int_0^e \frac{\rho(y)\, u(y)}{\rho_e V_e} \left(1 - \frac{u(y)}{V_e}\right) dy \tag{3.38}$$

Transition is reached when:

$$\frac{Re_{\theta_T}}{M_e} = 100 \tag{3.39}$$

This formulation is particularly unsuitable for our case, since that it requires an approximate calculation of the momentum thickness, since we don't solve the boundary layer. Following this road adds further approximation to an already raw formulation, and it requires an accurate identification of the flowfield region nature in order to determine properly θ_T. We will propose here another correlation, based on the study of hypersonic waveriders, which gives the Reynolds numer directly in terms of transition lenght s_T:

$$log_{10}(Re_T) = 6.421 \exp\left(1.209 \times 10^{-4}\, M_e^{2.641}\right) \tag{3.40}$$

This formulation is more straightforward and requires less approximated hypotheses on the boundary layer, so it will be used in our algorithm.

In the description of his generalized method [10], Simeonides proposes an interesting consideration, which leads to an accurate description of heat flow during the transition, with a correct match of experimental data. His proposal is to set the transition point at the location where a heating flux peak is achieved - obviously not the stagnation peak, but a downstream peak on the body. This consideration has been confirmed through Simeonides method validation against experimental wind tunnel data over the Halis Axisymmetric Configuration - HAC - which is the axisymmetric analogous of the Shuttle windward centerline. We see the comparison in the following picture which confronts heat fluxes from Simeonides, CFD code and experimental

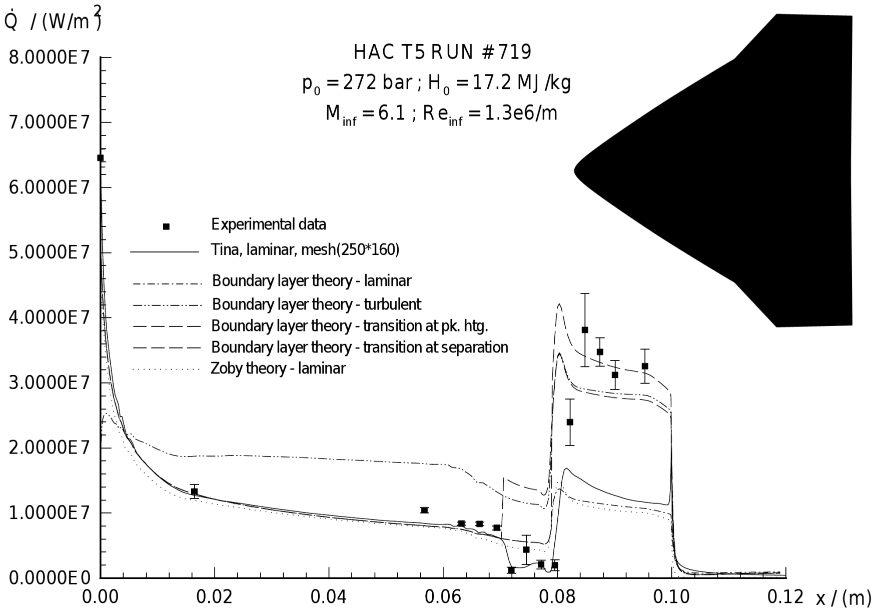


Figure 3.2: Heat flux over HAC configuration - prediction and measurements

As we can see, Simeonides method with transition at the heating peak, dashed line in fig. 3.2, matches the experimental data with an high level of accuracy for the level of approximation of this method. A word of caution on these results is to be said: Simeonides results are confronted on an axisymmetric, flared configuration, as the HAC is, and the peak produced by the presence of a flare is very clear. The validity of this method should be tested even on less characteristic geometries.

We end this chapter with the recommendation to use an high level of attention on these results, in particular on eqn.s (3.39) and (3.40), since they are extracted from very general correlations, and do not give completely reason of an highly non-linear and not fully understood process, as it is the transition phenomenon. We suggest hence to find, depending on the particular analyzed case and on the availability of experimental or numerical data, an appropriate value or correlation for Re_T, and insert it into the algorithm.

3.6 Heat flux calculation - Reynolds analogy

3.6.1 Theoretical Reynolds analogy

In the treatment of uncompressible viscous flows, there is a direct correlation between the friction coefficient and the Stanton number, which takes the name of *Reynolds analogy,* and it is of the form:

$$St = \frac{c_f}{2s} \tag{3.41}$$

where s is called the *analogy factor*, and for a laminar flat plate flow is simply defined as:

$$s = Pr^{2/3} \tag{3.42}$$

This results implies the fact that there is a direct link between the convective heat flow in the current and its internal dissipation due to friction. This is confirmed by the fact that Prandtl number:

$$Pr = \frac{c_p \mu}{k} \tag{3.43}$$

express the ratio between the *frictional dissipation* and the *thermal conduction* inside the fluid.

For a turbulent flow, such an exact relationship does not exist. Reasonable values can be taken from the work of Van Driest [13], which analyzes, both experimentally and theoretically, the behaviour of s with M_e and Re_x. Considering the behaviour of s with M_e for a flat plate, it seem from the data that s tends to a value of 0.815 for $M_e > 5$. Unfortunately, due to the fact that the experimentations were carried out more than 60 years ago, we don't have data for higher Mach numbers. As for the Re_x dependancy, there is only one analysis conducted at $M_e = 0$, which shows that s slowly increases with Re_x. However, this dependence is weaker than the one on M_e, i.e. we have an increase of s of 1.2% between $Re_x = 10^6$ and $Re_x = 10^8$.

According to the considerations stated before, and knowing that this is an hazardous guess, we take as an analogy factor for the turbulent flow in hypersonic conditions

$$s = 0.815 \tag{3.44}$$

thus obtaining the following Reynolds analogy for turbulent flows:

$$S_t = \frac{c_f}{1.63} \tag{3.45}$$

The analogy is presented here for the incompressible, flat plate flow. However, we can see that, in sec. 6.5.1 of [2], which deals with viscous flow solutions over a flat plate, that the validity of these results extend with little approximation even to the hypersonic flow regime. We can then simply extend the Reynolds analogy in the reference enthalpy sense, as Simeonides already did (see 3.4.1):

$$S_t{}^* = \frac{c_f^*}{2s^*} \tag{3.46}$$

where $s^* = Pr^{*2/3}$ for laminar flows, and $s^* = 0.815$ for turbulent flows. Simeonides proposes the $Pr^{*2/3}$ value for both formulations. We will choose the best correlation with our experimental data.

3.6.2 Reynolds analogy in RAM v2.0

In this paragraph, we would like to present some evidences about the practical use of Reynolds analogy in heat flow determination, and how the heat flow is currently calculated in our algorithm. From the comparison of our results with the ones provided by ESTEC CFD Dart calculations, see sec. 1.7, we have noticed indeed that the heat flow distribution followed a different pattern with respect to the friction factor coefficient. Since Reynolds analogy gives a linear relation between St and c_f, if we consider Pr almost constant - indeed it does not vary excessively even in the thermochemical equilibrium model - the heat flow distribution pattern has the same trend as the friction factor curve, in contraddiction with CFD data, as can be seen in the following figure:

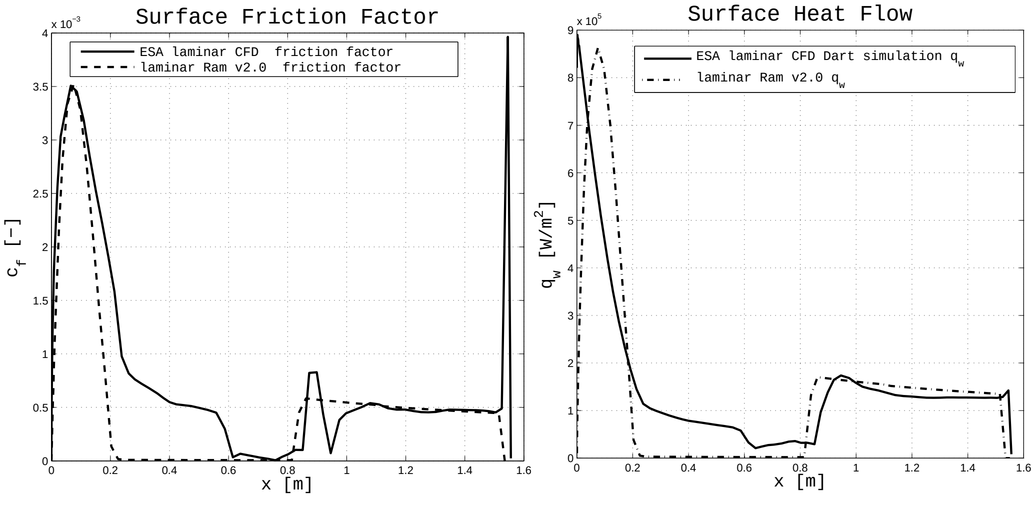


Figure 3.3: Friction factor and heat flow distribution over Dart windward centerline - 0° α and β - $M = 10.41$

Notice that c_f has the same trend between our model and ESA CFD simulations, while ESA q_w tends to resemble flat plate distribution, in contrary to our heat distribution, which, coherently with Reynolds analogy, mantains the same c_f curve trend. This is a limitation of our algorithm, but we accept it noticing that a very good agreement with the CFD values is obtained over a great part of the geometry, the integral error being thus limited. We notice the fact that, if the data are used for a preliminary heat flow study, in the stagnation region the heat peak value from RAM v2.0 should be used in place of the current RAM v2.0 stagnation value, since it would provide a more accurate description of the nose region.

3.7 Viscosity relation

In order to describe correctly the viscous behaviour of the flow, we need a relation to describe properly the dependence of μ from temperature T. Anderson [2] proposes the classical Sutherland's law, even if it is valid only for non-reacting gases, justifying himself by stating that this relation is valid for air in a range of several thousand of degrees. What happens in the event of molecular dissociation at over 2000 K (for atomspheric air) is not clear. However, Anderson assures that this formulation is appropriate for hypersonic flow calculations, so we will report it here:

$$\mu\left(T\right) = \mu_0 \left(\frac{T}{T_0}\right)^{3/2} \frac{T_0 + S}{T + S} \tag{3.47}$$

where, for air, we have $\mu_0 = 1.789\ 10^{-5}\left[\frac{kg}{m\,s}\right]$, $T_0 = 288\,[K]$ and $S = 110.4\,[K]$. Obviously, this correlation is valid only for terrestrial air. In the hypothesis of analysis of an extra-terrestrial re-entry path, appropriate coefficients μ_0 and S have to be found. However, we will notice that, for high temperatures, coefficient S looses importance.

3.47 can be written for air then, by substituting the previous constants, as:

$$\mu\left(T\right) = C_1 \frac{T^{3/2}}{T + C_2} \tag{3.48}$$

with:

- $C_1 = 1.458e^{-6}\ \left[\frac{kg}{m\,s\,K^{1/2}}\right]$
- $C_2 = 110.4\ [K]$

In real gas and mixtures, however, viscosity tends to follow a power law, like

$$\mu\left(T\right)=\mu_{\infty}\left(\frac{T}{T_{\infty}}\right)^{\alpha} \tag{3.49}$$

with $\alpha = 0.5 \rightarrow 1$.

Simeonides, in his generalized reference enthalpy analysis, finds that the following term from equation 3.29

$$\left(\frac{\rho^{*}}{\rho_{\infty}}\right)^{(1-n)}\left(\frac{\mu^{*}}{\mu_{\infty}}\right)^{n} \tag{3.50}$$

which, in perfect gas hypotesis is proportional to

$$\left(\frac{T_{\infty}}{T^{*}}\right)^{(1-n)}\left(\frac{\mu^{*}}{\mu_{\infty}}\right)^{n}=\left(\frac{T_{\infty}}{T^{*}}\right)^{(1-2n)}\left(\frac{T_{\infty}}{T^{*}}\frac{\mu^{*}}{\mu_{\infty}}\right)^{n}=\left(\frac{T_{\infty}}{T^{*}}\right)^{(1-2n)}C^{*n} \tag{3.51}$$

thus leading to the following relation for viscosity:

$$\mu^{*}=\mu_{\infty}C^{*}\frac{T^{*}}{T_{\infty}} \tag{3.52}$$

In his study, Simeonides find that C^{*}, while globally depending on $^{T^{*}}/_{T_{\infty}}$, is locally linear. A local relation for μ can be found, for example by using the *Chapman-Rubesin* relation for viscosity:

$$\frac{\mu^{*}}{\mu_{\infty}}=C^{*}\left(T_{w},T_{\infty}\right)\frac{T^{*}}{T_{\infty}} \tag{3.53}$$

where C^{*} is chosen to satisfy Sutherland's relation at the wall:

$$C^{*}\left(T_{w},T_{\infty}\right)=\left(\frac{T_{w}}{T_{\infty}}\right)^{1/2}\frac{T_{\infty}+S}{T_{w}+S} \tag{3.54}$$

Unfortunately, T_w is not available before an appropriate iterative solution of the termal problem on the surface, which can be obtained only after the calculation of the external flow. We could include this viscosity formulation inside the termal problem, but this solution could increase dramatically the non-linearity of an already non linear problem in T_w, as we will see in chapter 5, so we don't reccommend this option.

Another choice is to couple the viscosity with the thermochemical solver for the gas thermodynamic properties, which will be presented in chapter 4, along with a coherent model for viscosity determination in an equilibrium mixture. This choice is immediate and accurate, but, as we will see, the use of the thermochemical solver is quite expensive, and it is mandatory to obtain correct results only in the presence of a shock. We won't use it in every point of our vehicle, so viscosity can be obtained only where the solver is actually used. In all the other regions, the standard Sutherland relation 3.47 is used.

Chapter 4

Thermochemical equilibrium

4.1 Introduction

When considering thermodynamic problems dealing with very high temperatures, the commonly used perfect gas hypothesis looses its validity. Shock calculations made under perfect gas hypotheses, such as those presented in eqn.s (2.15), (2.17), based on the hypothesis of constant specific heats, i.e. $c_P,\ c_v = const.$ and $\gamma = const.$, produce incorrect results. As temperatures and re-entry velocities rise, vibrational excitation first, and molecular dissociation then, change the gas properties, as we have already seen in sec. 1.4.4. The temperature and pressure increase within the shock wave is absorbed, with respect to the perfect gas shock, in part by the activation of vibrational degrees of freedom, and in part by the variation of the gas mixture composition, resulting in modified global gas properties - i.e. the variation of the specific gas constant R and of the gas specific heats. These effects tend to produce a reduced increase of temperature through the shock. Calculations made under the unrealistic hypothesis of perfect gas overextimate the after-shock values of pressure and temperature, leading to completely wrong results when analyzing classical high-energy, high-Mach re-entry problems. An appropriate reconstruction of the gas thermodinamic state in these conditions has then to be adopted.

Some preliminary concepts will be introduced here to facilitate the theoretical description of high energy gas behaviour. When temperature rises over the level of molecular dissociation - over 2000 K for air - the gas composition starts to change. This modification dominates the thermodynamic properties of the gas, which now reveals its true nature as a *mixture* of different elements. This is the typical situation encountered in hypersonic flows, where the high temperatures reached after a shock wave produce such dissociation. In this process, all the gas properties must be determined from point to point depending on the mixture composition, by assigning the number of *moles* for each species. From the definition of the gas composition the other gas properties can be determined by treating every species as a single perfect gas, and combining the various species properties via the definition of the global mixture constants.

At this point, we can divide the description of chemically reacting mixtures in two categories, depending on the assumptions we can make over the particular flow. If the speed of chemical reactions is very fast, such that the variation of gas chemical composition can be supposed as *instantaneous*, we can say that the system we are considering is in *thermochemical equilibrium.* This means that the composition variation, which needs a finite amount of time to take place, has enough time to settle itself to a point of equilibrium. If equilibrium hypothesis can be used, the composition of the mixture is instantaneously determined by the choiche of *two variables of state,* so that we can describe the properties of the gas by determining these two variables, e.g.:

$$h = h\,(T, p)$$

$$e = e\,(T, p)$$

$$c_p = c_p\,(T, p)$$

$$c_v = c_v\,(T, p)$$

If this assumption is not met, i.e. chemical reactions take place at a rate comparable to fluid convection time, the actual composition of the gas mixture does not only depend on the thermodynamic two-variables state, but also on the flow *hystory.* In this condition, gas properties and energies depend not only on the state variables, but also on the actual composition, which is the result of the chemical and fluid dynamic time evolution of the flow. This condition is referred to as *thermochemical nonequilibrium.* For what we said before, in this case we have:

$$h = h\,(T, N_1, N_2, ..., N_n)$$

$$e = e\,(T, N_1, N_2, ..., N_n)$$

$$c_p = c_p\,(T, N_1, N_2, ..., N_n)$$

$$c_v = c_v\,(T, N_1, N_2, ..., N_n)$$

where N_i is the i-th species number of moles. To describe this kind of gas, the assumption that chemical reactions take place at a finite rate leads to the definition of chemical rates through the definition of appropriate *rate equations*, depending on the chemical reactions which take place in the flow.

It is interesting to note that, in both cases, the perfect gas equation of state still holds:

$$pv = RT \tag{4.1}$$

However, R is not a constant anymore, since it depends on the gas mixture composition:

$$R = \sum_i \frac{\rho_i}{\rho} R_i = \sum_i \frac{N_i}{N} \frac{\mathcal{M}_i}{\mathcal{M}} R_i \tag{4.2}$$

where:
$c_i = \frac{\rho_i}{\rho}$ is the *mass fraction* of a species, while
$X_i = \frac{N_i}{N}$ is the *mole fraction*
and their relation is expressed by:

$$c_i = X_i \left(\frac{\mathcal{M}_i}{\mathcal{M}} \right) \tag{4.3}$$

where $\mathcal{M}_i$ is the *molecular weight* of the single species, and $\mathcal{M}$ is the molecular weight of the mixture.

Since our problem deal with thermochemical reacting mixtures of gas, we will use this formulation for the description of the gas thermodynamic state. In particular, we will use the assumption of *thermochemical equilirium.* This decision is the consequence of some facts: the first is that, since our solution technique is based on the approximate description of the flowfield on the vehicle surface, we are not interested in the accurate reconstruction of the flowfield in other regions. Evidence in the assembly of our algorithm has been found that the major impact on the thermodinamic reconstruction accuracy is located in the correct description of the flow *after the frontal shock wave.* This analysis requires to use a chemically reacting gas model, since the huge temperature increase through the shock lead to dissociation in hypersonic flows. However, non-equilibrium effects are felt only in the immediate surroundings of the shock, as the gas composition tends to reach equilibrium before the vehicle surface. So the assumption of equilibrium flow is suitable if our aim is to reconstruct the correct thermodynamic state at the body wall.

Another reason to prefer an equilibrium analysis is that the infinite rate of chemical reactions leads to a very immediate formulation for the mixture composition; while the determination of the mixture composition for a non-equilibrium flow requires the thermodinamic state equations to be accompanied at every point of the calculation by a whole system of equations, describing finite rate chemical reactions and accounting for the present composition of the mixture. This approach tends to slow excessively computational times, and since the increment of accuracy in the flow description through and just behind the shock wave doesn't affect the accuracy on the vehicle body, this additional effort is not justifiable. In the following subsections, accurate theoretical description of the thermochemical equilibrium gas mixture applied to a shock wave and its implementation are afforded.

4.2 Thermochemical equilibrium through a shock wave

4.2.1 Thermochemical equilibrium via equilibrium constant method

Introduction In this section, we will present the thermodinamic description of an equilibrium mixture, and how to impose the thermodinamic equilibrium by Gibbs energy minimization and equilibrium entropy definition. This is not the actual implementation of thermochemical equilibrium adopted in our algorithm, but it introduces some basilar concepts that will be used in the formulation of the currently used equilibrium iterative method, that will be presented in sec. . We will introduce some basilar concepts and definitions in order to describe correctly the equilibrium definition. In the following section, capital letters are used for extensive quantities, small letters for mass specific quantities and small hat letters for molar specific quantities. For example

$$H\ [J]$$

is the extensive enthalpy,

$$h = \frac{H}{m}\ [\frac{J}{kg}]$$

is the specific enthalpy, and

$$\hat{h} = \frac{H}{N}\ [\frac{J}{mol}]$$

is the molar specific enthalpy.

In a non-reacting gas mixture, or a *frozen mixture*, the internal energy of the fluid depends only on two state variables, the *entropy* S and the *volume* V:

$$E = E\left(S, V\right) \tag{4.4}$$

When we are considering the effects of chemical reactions, however, a dependance on mixture composition is present, as we stated before:

$$E = E\left(S, V, N_i\right) \tag{4.5}$$

Hence, for a reacting mixture, the energy differential becomes:

$$E = \left.\frac{\partial E}{\partial S}\right|_{V,N_i} dS + \left.\frac{\partial E}{\partial V}\right|_{S,N_i} dV + \sum_{i=1}^{N_{sp}} \left.\frac{\partial E}{\partial N_i}\right|_{S,V,N_{j\neq i}} dN_i \tag{4.6}$$

thus, recalling that

$$\left.\frac{\partial E}{\partial S}\right|_{V,N_i} = T \tag{4.7}$$

and

$$\left.\frac{\partial E}{\partial V}\right|_{S,N_i} = -p \tag{4.8}$$

we can express internal energy into the subsequent formulation:

$$dE = TdS - pdV + \sum_{i=1}^{N_{sp}} \hat{\mu}_i dN_i \tag{4.9}$$

We underline the fact that the two first terms of the right hand side of 4.9 are one of the most classical forms of the first principle of thermodynamics for a non-reacting gas. In addition we have the contribution of composition change:

$$\sum_{i=1}^{N_{sp}} \hat{\mu}_i dN_i$$

where

$$\hat{\mu}_i = \left.\frac{\partial E}{\partial N_i}\right|_{S,V,N_{j\neq i}} \tag{4.10}$$

is called the *molar electrochemical potential.*

Gibbs Free Energy We define now Gibbs energy G as the Legendre transformation of internal energy *E,* with the substitution of (S, V) with(T, P) as variables of state:

$$G = G\left(T, P, N_i\right) = E\left(S, V, N_i\right) - \left.\frac{\partial E}{\partial S}\right|_{V,N_i} S - \left.\frac{\partial E}{\partial V}\right|_{S,N_i} V \tag{4.11}$$

that, for eqn. (4.7), (4.8) becomes:

$$G = E - TS + pV \tag{4.12}$$

Expressing Gibbs energy differential leads to:

$$dG = dE - TdS - SdT + pdV + Vdp \tag{4.13}$$

which becomes, by substituting eqn. (4.9):

$$dG = -SdT + Vdp + \sum_{i=1}^{N_{sp}} \hat{\mu}_i dN_i \tag{4.14}$$

One of the properties of Gibbs energy is to be *linear with respect to an arbitrary parameter, λ.* We can then write:

$$G\left(S, V, \lambda N_i\right) = \lambda G\left(S, V, N_i\right) \tag{4.15}$$

Gibbs energy is also *addictive*, hence leading to:

$$G\left(S, V, \sum N_i\right) = G\left(S, V, N_1\right) + G\left(S, V, N_2\right) + ... + G\left(S, V, N_{N_{sp}}\right) \tag{4.16}$$

By combining these two properties if G, one can write:

$$G\left(S, V, \sum N_i\right) = \sum_{i=1}^{N_{sp}} \frac{\partial G}{\partial \lambda} = \sum_{i=1}^{N_{sp}} \frac{\partial G}{\partial\left(\lambda N_i\right)} \frac{\partial\left(\lambda N_i\right)}{\partial \lambda} = \sum_{i=1}^{N_{sp}} \frac{\partial G}{\partial\left(\lambda N_i\right)} N_i \tag{4.17}$$

which becomes, for the arbitrariness of λ:

$$G = \sum_i \frac{\partial G}{\partial N_i} N_i \tag{4.18}$$

for $\lambda = 1$.

Now, recalling the definition of the electrochemical potential (4.10), and of the internal and Gibbs energy differentials, respectively (4.9) and (4.14), we can write:

$$\left.\frac{\partial E}{\partial N_i}\right|_{S,V,N_{j\neq i}} = \left.\frac{\partial G}{\partial N_i}\right|_{T,P,N_{j\neq i}} = \hat{\mu}_i \tag{4.19}$$

thus demonstrating that the *electrochemical molar potential* coincides with the *molar gibbs energy for the single species:*

$$\left.\frac{\partial G}{\partial N_i}\right|_{T,P,N_{j\neq i}} = \hat{g}_i = \hat{\mu}_i \tag{4.20}$$

Eqn. (4.18) can be then rewritten as:

$$G = \sum_i \frac{\partial G}{\partial N_i} N_i = \sum_i \hat{g}_i N_i = \sum_i \hat{\mu}_i N_i \tag{4.21}$$

Thermochemical Equilibrium by Minimization of Gibbs Free Energy We can now define thermochemical equilibrium by using the definition of entropy. Considering:

$$dS = dS_{rev} + dS_{irr} \tag{4.22}$$

and remembering that

$$dE = TdS - pdV \tag{4.23}$$

we have that

$$TdS_{rev} = dE + pdV + TdS_{irr} \tag{4.24}$$

which, confronted to eqn. (4.9), leads to the consideration that irreversible entropy is generated via species modification and mixing:

$$TdS_{irr} = -\sum_i \hat{\mu}_i dN_i \tag{4.25}$$

If we introduce now the hypothesis of chemical equilibrium, we must observe the fact that, if chemical reactions are balanced in both directions with an instantaneous rate, this mean that the production of irreversible entropy is null, hence:

$$dS_{irr} = 0 \tag{4.26}$$

which leads to

$$\sum_i \hat{\mu}_i dN_i = 0 \tag{4.27}$$

which is called the *algebraic equilibrium condition.*

We need now to correlate the species composition of the mixture to the actual chemical reactions which happens in high temperature mixtures. A generic chemical reaction inside the mixture can be expressed as:

$$\nu_i A_i \rightleftharpoons \nu_j B_j \tag{4.28}$$

where ν is the stoichiometric coefficient.

We can then define the *reaction rate* ξ as the rate of creation and destruction of new species, as

$$dN_i = \nu_i d\xi \tag{4.29}$$

with $\nu_i < 0$ if the species i is a reactant, and $\nu_i > 0$ if the species i is a product. The algebraic equilibrium condition becomes then

$$\sum_i \hat{\mu}_i \nu_i d\xi = 0 \tag{4.30}$$

If we assume chemical equilibrium, we have to suppose that chemical reactions take place istantaneusly at every moment, hence $d\xi \neq 0$ at any moment. The algebraic equilibrium then becomes

$$\sum_i \hat{\mu}_i \nu_i = 0 \tag{4.31}$$

Let us underline the fact that the imposition of chemical equilibrium coincides with the minimization of Gibbs energy function:

$$dG = \sum_i \hat{\mu}_i dN_i = \sum_i \hat{\mu}_i \nu_i d\xi = 0 \tag{4.32}$$

If we know the chemical reactions that occur in a particular mixture, i.e. the stoichiometric coefficients, the only thing that is left to find in order to impose thermochemical equilibrium is to determine the molar electrochemical potential $\hat{\mu}_i$. Let us describe the following molar quantities for the *i-th* species:

$$\hat{e}_i = \int\limits_{T_{ref}}^{T} \hat{c}_{v,i} dT + \hat{e}_{ref,i} \tag{4.33}$$

$$\hat{h}_i = \int\limits_{T_{ref}}^{T} \hat{c}_{p,i} dT + \hat{h}_{ref,i} \tag{4.34}$$

For a single species, the Dalton's law holds:

$$p = \sum_i p_i \tag{4.35}$$

remembering that:

$$\frac{p_i}{p} = \frac{N_i}{N} \equiv X_i \tag{4.36}$$

The relation for entropy for the single species then becomes:

$$d\hat{s}_i = \frac{1}{T}\left(d\hat{e}_i + p_i d\hat{v}_i\right) = \frac{1}{T}\left(d\hat{h}_i - \hat{v}_i dp_i\right) = \frac{1}{T}\left(\hat{c}_{p,i}dT - \hat{v}_i dp_i\right) \tag{4.37}$$

having identified the *molar volume* as $\hat{v}_i = \frac{V}{N_i} = \frac{\mathcal{R}T}{p_i}$.

Integration of (4.37) leads to:

$$\hat{s}_i = \int\limits_{T_{ref}}^{T} \hat{c}_{p,i}\frac{dT}{T} - \mathcal{R}\int\limits_{P_{ref}}^{P_i} \frac{dp_i}{p_i} + \hat{s}_{ref,i} \tag{4.38}$$

$$\hat{s}_i = \int\limits_{T_{ref}}^{T} \hat{c}_{p,i}\frac{dT}{T} - \mathcal{R}\ln\left(\frac{p_i}{p_{ref}}\right) + \hat{s}_{ref,i} \tag{4.39}$$

Remembering that the electrochemical potential is equivalent to Gibbs molar energy, we have

$$\hat{\mu}_i = \hat{g}_i = \hat{e}_i - T\hat{s}_i + p_i\hat{v}_i = \hat{h}_i - T\hat{s}_i \tag{4.40}$$

which becomes

$$\hat{\mu}_i = \hat{h}_i - T\hat{s}_i = \int\limits_{T_{ref}}^{T} \hat{c}_{p,i}dT + \hat{h}_{ref,i} - T\int\limits_{T_{ref}}^{T} \hat{c}_{p,i}\frac{dT}{T} + T\mathcal{R}\ln\left(\frac{p_i}{p_{ref}}\right) - T\hat{s}_{ref,i} \tag{4.41}$$

Let us divide the contribution due to the offset from initial conditions from the potential depending from the partial pressure of the mixture, which is proportional to gas composition:

$$\hat{\mu}_i = \hat{\mu}_{0,i}\left(T\right) + T\mathcal{R}\ln\left(p_i\right) \tag{4.42}$$

where

$$\hat{\mu}_{0,i}\left(T\right) = \int\limits_{T_{ref}}^{T} \hat{c}_{p,i}dT - T\int\limits_{T_{ref}}^{T} \hat{c}_{p,i}\frac{dT}{T} + \hat{h}_{ref,i} - T\mathcal{R}\ln\left(p_{ref}\right) - T\hat{s}_{ref,i} \tag{4.43}$$

is the single species electrochemical potential calculated from reference conditions, neglecting the potential variation due to mixing.

We can then express the equilibrium condition (4.31) as:

$$\sum_i \hat{\mu}_i\nu_i = \sum_i \left(\hat{\mu}_{0,i}\left(T\right) + T\mathcal{R}\ln\left(p_i\right)\right)\nu_i = 0 \tag{4.44}$$

which leads to:

$$\sum_{i=1}^{nSP} ln\left(p_i\right)\nu_i = \frac{\sum_{i=1}^{nSP}\hat{\mu}_{0,i}\left(T\right)\nu_i}{T\mathcal{R}} \tag{4.45}$$

which can be rewritten as

$$\prod_{i=1}^{nSP} p_i^{\nu_i} = e^{\frac{\sum_{i=1}^{nSP}\hat{\mu}_{0,i}(T)\nu_i}{T\mathcal{R}}} \equiv K_p\left(T\right) \tag{4.46}$$

$K_P\left(T\right)$ is called the mixture *equilibrium constant* and it can be calculated once $\hat{\mu}_{0,i}\left(T\right)$ is known. The molar electrochemical potentials$\hat{\mu}_{0,i}\left(T\right)$ can be obtained in two ways: the most common way is trough tabulated experimental variables, such as NASA tabulations contained in [14]. In these tables, reference enthalpies and Gibbs energies are tabulated against T, in the form

$$\{\hat{h}_0\left(T\right) - \hat{h}_0\left(0\right)\} \quad \left[\frac{KJ}{mol}\right]$$

for enthalpies, and

$$\{-\frac{\hat{g}_0\left(T\right) - \hat{h}_0\left(0\right)}{T}\} \quad \left[\frac{J}{mol\,K}\right]$$

for Gibbs energies - remembering that $\hat{\mu}_0\left(T\right) = \hat{g}_0\left(T\right)$, from eqn. (4.40). Similarly to the definition of $\hat{\mu}_0\left(T\right)$, we can express molar enthalpy, molar entropy and molar gibbs energy for the single species without the mixing contribution, referred to a *standard state* reference point pressure $p_{ref} = 1\,bar$, thus obtaining:

$$\hat{h}_{0,i}\left(T\right) = \int_{T_{ref}}^{T} \hat{c}_{p,i}dT + \hat{h}_{ref,i} \tag{4.47}$$

and

$$\hat{s}_{0,i}\left(T\right) = \int_{T_{ref}}^{T} \hat{c}_{p,i}\frac{dT}{T} + \hat{s}_{ref,i} + \mathcal{R}\ln\left(p_{ref}\right) \tag{4.48}$$

which results in a definition for Gibbs energy, from eqn. (4.40):

$$\hat{g}_{0,i} = \hat{h}_{0,i} - T\hat{s}_{0,i} \tag{4.49}$$

which is equivalent to (4.43).

The *standard state* reference point temperature is chosen at $\hat{h}_0\left(298.15\right)=0$, i.e. to have null molar enthalpy, with respect to the standard state, at $T_{ref}=298.15\,K$, remembering also $p_{ref}=1\,bar$. We notice that, as can be immediatly seen from eqn. (4.40), that at *0 K* we have $\hat{g}_0\left(0\right)=\hat{h}_0\left(0\right)$, and $\hat{s}_0\left(0\right)=0$.

Obviously, these reference quantities are tabulated relatively to elements and species in their most common compositon at the standard state. For composite species which are created by high temperature reactions, the species *enthalpy of formation* contribution from the reference state has to be calculated from the balance of products minus reactants, taken at the most common standard state form:

$$\Delta_f\hat{h}_{0,comp.species}\left(T\right)=\sum_{i=1}^{nSP}\nu_i\hat{h}_{0,i}\left(T\right) \tag{4.50}$$

where, as stated before $\nu_i<0$ for reactants and $\nu_i>0$ for products are the stoichiometric coefficients.

Electrochemical potentials can be directly calculated from tabulated values of Gibbs molar energies, noting that for the generic reaction through the gas mixture we can write:

$$\Delta_f\hat{g}_0\left(T\right)=\sum_{i=1}^{nSP}\nu_i\hat{g}_{0,i}\left(T\right)=\sum_{i=1}^{nSP}\nu_i\hat{\mu}_{0,i}\left(T\right) \tag{4.51}$$

Hence, equilibrium constants can be expressed as

$$K_p\left(T\right)=e^{\frac{\sum_{i=1}^{nSP}\hat{\mu}_{0,i}(T)\nu_i}{T\mathcal{R}}}=e^{\frac{\Delta_f\hat{g}_0(T)}{T\mathcal{R}}} \tag{4.52}$$

We can now write, for every reaction which is present in our mixture, an equation of the form (4.46), hence obtaining a description of the species composition in form of partial pressures. This introduces an unknown in pressure for every species present in the mixture, so obviously we will have more degrees of freedom than the number of reactions, hence making it necessary to introduce new equations. One could think that the way to close the system would be to add more chemical reactions to the system. However, we are forgetting that thermochemical equilibrium must satisfy two physical constraints. The first is the Dalton law of partial pressures, eqn. (4.35), while the second constraint is the conservation of atomic nuclei in the composition variation, which can be written as

$$\sum_{j=1}^{nSP}\tilde{\nu}_{i,j}N_j-\mathcal{N}_{0,i}=0\quad i=1:nEL \tag{4.53}$$

where $\tilde{\nu}_{i,j}$ is the ratio of the atomic mole number of element i on the species j mole number (e.g., if we are considering the number of oxigen atoms in the species O_2, hence we have $\tilde{\nu}_{O,O_2} = 2$) and $\mathcal{N}_{0,i}$ is the initial number of atomic nuclei for a given element; *nSP* is the total number of species, while *nEL* is the total number of elements. Notice the difference between $\tilde{\nu}_{i,j}$, which is always positive and counts not the stoichiometric value but the times an element appears in a species, and the stoichiometric coefficient ν_i, which depends of whether we are considering products or reactants and counts the ratio of species in a reaction.

We can then express (4.53) in terms of partial pressures as:

$$\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} p_j - \mathcal{N}_{0,i} \frac{V}{\mathcal{R}T} = 0 \quad i = 1 : nEL \tag{4.54}$$

which is immediately obtained from the molar form of the state equation (4.1)

$$p_i V = N_i \mathcal{R} T \tag{4.55}$$

We can obtain a closed form equation for partial pressures if we express eqn. (4.54) by expressing it as a ratio between different elements:

$$\frac{\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} p_j - \mathcal{N}_{0,i}}{\sum_{j=1}^{nSP} \tilde{\nu}_{l,j} p_j - \mathcal{N}_{0,l}} = 0 \tag{4.56}$$

Obviously in this form we can write $nEL-1$ equations, if we close the mass balance system with equation (4.35) we have nEL equations in *P,* equivalent to system (4.53). In a thermochemical equilibrium system, which is composed by $nCHEM$ chemical reaction equations of the form (4.46), of $nEL - 1$ mass conservation ratios defined as in eqn. (4.56), and of *1* Dalton sum of partial pressures (4.35), the balance of these equation could be not uniquely defined . One will chose the correct balance of reaction equations and mass equations depending on the reactions which actually take place and on the unknowns number, which depends on species and on the number of elements for each single species.

4.2.2 Thermochemical equilibrium via constrained minimization of Gibbs energy (CEA)

Introduction In this subsection, we will show in detail the current implementation of thermochemical equilibrium. An algorithm of the form presented in sec. 4.2.1 is very versatile and intuitive, but has to be implemented and validated for every new composition which has to be analyzed. In the last 40 years, NASA Lewis Research Center has devoted some of his best energies and efforts in defining a complete tool for thermochemical equilibrium and high temperature gas mixture properties determination. This tool, called *Chemical Equilibrium with Applications,* or CEA, is already validated by the decade-long NASA experience in high temperature gas dynamics, and is presented to the public in a open-source, completely free format.

Its main source code is composed by an unique Fortran 77 main file and by some thermodynamics external databases, which reproduce in a digital way the data contained in [14]. In the present form of our aerodynamic module, a modified version with respect to standard CEA distribution, which can be found in [15], is used. In particular, the input and output patterns of NASA standard version have been adjusted in order to produce a compiled version which is now fully automated inside a Matlab script. An optimization of the output has been made, so that only the variables of interest are returned to the script, written in the correct format. A quick re-compilation for C++ integration, under the same philosophy of defining an external executable optimized to operate with the current language will be followed, modifying the output in a proper way.

Restrained Minimization of Gibbs Energy We will analyze in this paragraph the formulation which underlies the implementation of CEA software, its analogies and differences from the method presented in sec. 4.2.1, and the iteration forms of the theoretical equations used for actual calculations. The concept underlying CEA software is again the minimization of Gibbs energy, which as we have seen in eqn. (4.32), coincides with thermochemical equilibrium by imposing to zero the generation of irreversible entropy. In the previous formulation we have used the molar specific form of energies and quantities, here we will use the specific formulations. Hence we define the mole per kilogram of mixture as

$$n_i = \frac{N_i}{M} \tag{4.57}$$

where M is the total mass of the mixture. The total moles per kilogram can be defined as:

$$n = \sum_{i=1}^{nSP} n_i \tag{4.58}$$

The difference between the use of N_i and n_i is that, in the first case, we obtain from the molar specific species properties the *extensive* properties; for example for Gibbs energy we have:

$$G = \sum_{i=1}^{nSP} \hat{g}_i N_i \tag{4.59}$$

If we use mole per kilogram expression, we obtain instead the *specific* quantity of Gibbs energy, i.e.:

$$g = \sum_{i=1}^{nSP} \hat{g}_i n_i \tag{4.60}$$

Recalling eq. (4.20), we have that:

$$\hat{g}_i = \hat{\mu}_i = \left.\frac{\partial G}{\partial N_i}\right|_{T,P,i\neq j} = \left.\frac{\partial g}{\partial n_i}\right|_{T,P,i\neq j} \tag{4.61}$$

hence

$$g = \sum_{i=1}^{nSP} \hat{\mu}_i n_i \tag{4.62}$$

As in the previous section, we want to find the minimization of Gibbs free energy, but we will be looking for it while satisfying the mass-balance constraint, expressed by eqn. (4.53), which can be rewritten, by dividing for the mixture mass, as:

$$\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i} = 0 \quad i = 1 : nEL \tag{4.63}$$

where

$$\mathfrak{n}_{0,i} = \frac{\mathcal{N}_{0,i}}{M} \tag{4.64}$$

is the number of atomic nuclei for element i per unit of mixture mass.

Introducing constraint (4.63) in eqn. (4.62) through Lagrange multipliers, we obtain:

$$\bar{g} = g + \sum_{i=1}^{nEL} \left(\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i} \right) \lambda_i = \sum_{j=1}^{nSP} \hat{\mu}_j n_j + \sum_{i=1}^{nEL} \left(\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i} \right) \lambda_i \tag{4.65}$$

The condition of equilibrium comes from the imposition of stationarity for the constrained potential $\bar{g}$:

$$\delta\bar{g} = \sum_{j=1}^{nSP} \left(\hat{\mu}_j + \sum_{i=1}^{nEL} \lambda_i \tilde{\nu}_{i,j} \right) \delta n_j + \sum_{i=1}^{nEL} \left(\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i} \right) \delta\lambda_i = 0 \tag{4.66}$$

Treating variations as independent leads to:

$$\hat{\mu}_j + \sum_{i=1}^{nEL} \lambda_i \tilde{\nu}_{i,j} = 0 \quad j = 1 : nSP \tag{4.67}$$

from the variation δn_j, and to the mass balance equation (4.63) from the variation $\delta\lambda_i$.

We can then begin to solve the equilibrium system defined by (4.67) and (4.63), by recalling that the electrochemical potential can be expressed as in eqn. (4.42). We can notice that the term

$$T\mathcal{R}\ln(p_i)$$

can be rewritten, since $p_i = X_i P$ as:

$$T\mathcal{R}\ln(p_i) = T\mathcal{R}\ln(X_i p) = T\mathcal{R}\ln(X_i) + T\mathcal{R}\ln(p)$$

which, noting that $X_i = \frac{N_i}{N} = \frac{n_i}{n}$, let us rewrite eqn. (4.42) as

$$\hat{\mu}_i = \hat{\mu}_{0,i}(T) + T\mathcal{R}\ln\left(\frac{n_i}{n}\right) + T\mathcal{R}\ln(p) \tag{4.68}$$

where $\hat{\mu}_{0,i}(T) = \hat{g}_{0,i}(T)$ can be obtained from tabulated data using the procedure explained in the preceding subsection.

Once we have defined the equilibrium system, we must select the two assigned termodynamic variables which characterize the state at the equilibrium point. For our problem, which is a shock problem, we will use as state variables temperature T and pressure p. Other state variables can be choosen, the possible alternative pairs being *(h,p)* or *(s,p)* for the formulation in Gibbs free energy. For other state descriptions, an Helmolthz formulation is needed. For state formulation different from *(T,p)* we recommend the complete CEA algorithm analysis as reported in [16], in this chapter we will analyze only the impositon of equilibrium in pressure and temperature.

Iteration Equations for Restrained Gibbs Energy Once the electrochemical potential definition is cleared, we can begin to write the iteration equations for obtaining thermodynamic state in *(T,p)*. The three equations that will be used to define chemical equilibrium are the constrained chemical equilibrium equation (4.67), the specific mass conservation equation (4.63) and the specific molar composition conservation (4.58). The system is hence:

$$\begin{cases} \hat{\mu}_j + \sum_{i=1}^{nEL} \lambda_i \tilde{\nu}_{i,j} = 0 & j = 1 : nSP \\ \sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i} = 0 & i = 1 : nEL \\ n = \sum_{i=1}^{nSP} n_i \end{cases} \tag{4.69}$$

with the imposed thermodynamic state at equilibrium:

$$\begin{cases} T = T_e \\ p = p_e \end{cases} \tag{4.70}$$

We proceed to discretize the system through Newton-Rhapson algorithm. The increment independent variables which will be used are $\Delta \ln (n_j)$,, $\Delta \ln (n)$, for $j = 1 : nSP$, $\Delta \ln (T)$ and $\pi_i = -\frac{\lambda_i}{\mathcal{R}T}$. π_i is not marked as an increment, since because the system is strictly linear in λ_i, we will start the linearization from $\lambda_i = 0$. These variables are chosen in order to obtain simpler derivatives of the first function of system (4.69), which can be rewritten as, remembering eqn. (4.68):

$$\hat{\mu}_j + \sum_{i=1}^{nEL} \lambda_i \tilde{\nu}_{i,j} = \hat{\mu}_{0,j}(T) + T\mathcal{R} \ln\left(\frac{n_j}{n}\right) + T\mathcal{R}\ln(p) + \sum_{i=1}^{nEL} \lambda_i \tilde{\nu}_{i,j} = 0 \quad j = 1 : nSP \tag{4.71}$$

Rewriting the previous equation in non-dimensional form leads to:

$$\frac{\hat{\mu}_j}{\mathcal{R}T} - \sum_{i=1}^{nEL} \pi_i \tilde{\nu}_{i,j} = \frac{\hat{\mu}_{0,j}(T)}{\mathcal{R}T} + \ln\left(\frac{n_j}{n}\right) + \ln(p) - \sum_{i=1}^{nEL} \pi_i \tilde{\nu}_{i,j} = 0 \quad j = 1 : nSP \tag{4.72}$$

If we linearize the previous equation around the equilibrium state, in our independent variables $\ln (n_j)$ for $j = 1 : nSP$, $\ln (n)$, $\ln (T)$, with null lagrange multipliers, we obtain:

$$\frac{\hat{\mu}_j}{\mathcal{R}T} =$$

$$= \left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e + \frac{\partial}{\partial \ln T}\left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e \Delta \ln(T) + \frac{\partial}{\partial \ln n_j}\left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e \Delta \ln(n_j) + \frac{\partial}{\partial \ln n}\left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e \Delta \ln(n) \tag{4.73}$$

Recalling eqn. (4.68) and eqn. (4.40), we have that:

$$\frac{\partial}{\partial \ln T}\left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e = \frac{\partial}{\partial \ln T}\left(\frac{\hat{\mu}_{0,j}(T)}{\mathcal{R}T}\right)_e = \frac{\partial}{\partial \ln T}\left(\frac{\hat{g}_{0,j}(T)}{\mathcal{R}T}\right)_e \tag{4.74}$$

since $\hat{\mu}_{0,j}(T)$ is the only temperature dependent term of $\hat{\mu}_j$.

We have that

$$\frac{\partial}{\partial \ln T}(\cdot) = T\frac{\partial}{\partial T}(\cdot) \tag{4.75}$$

hence eqn. (4.74) becomes:

$$\frac{\partial}{\partial \ln T}\left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e = T\frac{\partial}{\partial T}\left(\frac{\hat{\mu}_{0,j}\left(T\right)}{\mathcal{R}T}\right)_e \tag{4.76}$$

We have that:

$$\frac{\partial}{\partial T}\left(\frac{\hat{g}_{0,j}\left(T\right)}{\mathcal{R}T}\right)_e = \frac{\mathcal{R}T\frac{\partial \hat{\mu}_{0,j}(T)}{\partial T} - \hat{\mu}_{0,j}\left(T\right)\mathcal{R}}{\left(\mathcal{R}T\right)^2} \tag{4.77}$$

Considering (4.43) and (4.48), we have that:

$$\frac{\partial \hat{\mu}_{0,j}\left(T\right)}{\partial T} = \hat{c}_{p,j} - \int\limits_{T_{ref}}^{T} \hat{c}_{p,j}\frac{dT}{T} - \hat{c}_{p,j} - \mathcal{R}\ln P_{ref} - \hat{s}_{ref,j} = -\hat{s}_{0,j}\left(T\right) \tag{4.78}$$

hence, recalling (4.49):

$$\frac{\partial}{\partial T}\left(\frac{\hat{g}_{0,j}\left(T\right)}{\mathcal{R}T}\right)_e = \frac{-\mathcal{R}T\hat{s}_{0,j}\left(T\right) - \left(\hat{h}_{0,j}\left(T\right) - T\hat{s}_{0,j}\left(T\right)\right)\mathcal{R}}{\left(\mathcal{R}T\right)^2} = -\frac{\hat{h}_{0,j}\left(T\right)}{\mathcal{R}T^2} \tag{4.79}$$

which leads to

$$\frac{\partial}{\partial \ln T}\left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e = -\frac{\hat{h}_{0,j}\left(T\right)}{\mathcal{R}T^2} \tag{4.80}$$

Let us find the derivatives with respect to the other correction variables: from equation (4.68) we obtain readily

$$\frac{\partial}{\partial \ln n_j}\left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e = 1 \tag{4.81}$$

and

$$\frac{\partial}{\partial \ln n}\left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e = -1 \tag{4.82}$$

The increment becomes then:

$$\frac{\hat{\mu}_j}{\mathcal{R}T} = \left(\frac{\hat{\mu}_j}{\mathcal{R}T}\right)_e - \frac{\hat{h}_{0,j}}{\mathcal{R}T}\Delta\ln\left(T\right) + \Delta\ln\left(n_j\right) - \Delta\ln\left(n\right) \tag{4.83}$$

The iterative equation can be derived then, by substituting the increment in eqn. (4.72):

$$\Delta \ln(n_j) - \Delta \ln(n) - \frac{\hat{h}_{0,j}}{\mathcal{R}T}\Delta \ln(T) - \sum_{i=1}^{nEL} \pi_i \tilde{\nu}_{i,j} = -\frac{\hat{\mu}_j}{\mathcal{R}T} \quad j = 1 : nSP \tag{4.84}$$

recalling that π_i is theoretically an increment, but starting from zero as discussed before. The reference to an equilibrium point is omitted, since we are considering the left hand side as the increment calculated from the previous iteration, which is then the right hand side.

Notice the fact that pressure doesn't appear as an increment, as one would think when looking at eqn. (4.68); this is because we are searching for the equilibrium condition by imposing state through conditions (4.70). In fact, when imposing temperature, even the variation $\Delta \ln(T)$ disappears; we have kept it just because the Gibbs approach presented here, which always require the imposition of pressure as a thermodynamic state variable, accepts as the other state variable *T, h* or *s*. In the latter two cases, a variation in T is required, hence we are proposing it in eqn. (4.84).

The second and third equation of system (4.69) are written in n and n_j, their increments have to be expressed in terms of $\Delta \ln(n_j)$ and $\Delta \ln(n)$. The second equation is written as:

$$\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i} = 0 \quad i = 1 : nEL \tag{4.85}$$

Its variation, considering the presence of the single iteration variable $\Delta \ln(n_j)$, near the equilibrium state, becomes:

$$\left(\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i}\right)_e + \frac{\partial}{\partial \ln(n_j)}\left(\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i}\right)_e \Delta \ln(n_j) = 0 \tag{4.86}$$

Since $d\ln(n_j) = {}^1\!/_{n_j} dn_j$ the logarithmic derivative can be expressed as:

$$\frac{\partial n_j}{\partial \ln(n_j)} = n_j \frac{\partial n_j}{\partial n_j} = n_j \tag{4.87}$$

hence obtaining a variation of the form:

$$\left(\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j - \mathfrak{n}_{0,i}\right)_e + \left(\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j\right)_e \Delta \ln(n_j) = 0 \tag{4.88}$$

The reference to an equilibrium point can be omitted, considering, as we did with eqn. (4.84), the stationary term in eqn. (4.88) as the previous iteration variables and the increment term as the correction term, in the Newton-Rhapson algorithm sense; the equation can be then rewritten in the form:

$$\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j \Delta \ln(n_j) = \mathfrak{n}_{0,i} - \sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j \quad i = 1 : nEL \tag{4.89}$$

Proceeding in the same way for the third equation:

$$n - \sum_{j=1}^{nSP} n_j = 0 \tag{4.90}$$

we obtain the increment

$$\left(n - \sum_{j=1}^{nSP} n_j\right)_e + \frac{\partial}{\partial \ln(n_j)} \left(n - \sum_{j=1}^{nSP} n_j\right)_e \Delta \ln(n_j) + \frac{\partial}{\partial \ln(n)} \left(n - \sum_{j=1}^{nSP} n_j\right)_e \Delta \ln(n) = 0 \tag{4.91}$$

which leads to the iteration step:

$$\sum_{j=1}^{nSP} n_j \Delta \ln(n_j) - n \Delta \ln(n) = n - \sum_{j=1}^{nSP} n_j \tag{4.92}$$

The iteration system becomes then, combining (4.84),(4.89) and (4.92):

$$\begin{cases} \Delta \ln(n_j) - \Delta \ln(n) - \frac{\hat{h}_{0,j}}{\mathcal{R}T} \Delta \ln(T) - \sum_{i=1}^{nEL} \pi_i \tilde{\nu}_{i,j} = -\frac{\hat{\mu}_j}{\mathcal{R}T} \ , \\ j = 1 : nSP \\ \\ \sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j \Delta \ln(n_j) = \mathfrak{n}_{0,i} - \sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j \ , \quad i = 1 : nEL \\ \\ \sum_{j=1}^{nSP} n_j \Delta \ln(n_j) - n \Delta \ln(n) = n - \sum_{j=1}^{nSP} n_j \end{cases} \tag{4.93}$$

The system which we present here is apt to solve only the thermochemical equilibrium with assigned state in *(T,p)*. To define equilibrium in *(h,p)*, an iteration equation on enthaply must be written, and similarly, an equilibrium in *(s,p)* requires an iteration equation on entropy. Iteration variables are $\Delta \ln(n)$, $\Delta \ln(n_j)$, and π_j, since temperature is assigned, and $\Delta \ln(T)$ is reported here only for the sake of completness with respect to entropy and enthalpy formulations.

Considering system (4.93), another reduction of variables is suggested, since the resolution of a high number of simultaneous equations is an expensive process, expecially if the number of considered species grows. We perform then an algebraic substitution on the system, by obtaining $\Delta \ln(n_j)$ from the first equation of (4.93) and substituting it in the other two, hence obtaining

$$\sum_{k=1}^{nEL}\sum_{j=1}^{nSP} n_j\tilde{\nu}_{k,j}\tilde{\nu}_{i,j}\pi_i + \sum_{j=1}^{nSP}\tilde{\nu}_{k,j}n_j\Delta\ln(n) + \sum_{j=1}^{nSP}\tilde{\nu}_{k,j}n_j\frac{\hat{h}_{0,j}}{\mathcal{R}T}\Delta\ln(T) =$$

$$= \mathfrak{n}_{0,k} - \sum_{j=1}^{nSP}\tilde{\nu}_{k,j}n_j + \sum_{j=1}^{nSP}\tilde{\nu}_{k,j}n_j\frac{\hat{\mu}_j}{\mathcal{R}T} \tag{4.94}$$

for $k = 1 : nEL$ from the second iteration equation, and

$$\sum_{i=1}^{nEL}\sum_{j=1}^{nSP}\pi_i\tilde{\nu}_{i,j}n_j + \left(\sum_{j=1}^{nSP}n_j - n\right)\Delta\ln(n) + \sum_{j=1}^{nSP}n_j\frac{\hat{h}_{0,j}}{\mathcal{R}T}\Delta\ln(T) =$$

$$= n - \sum_{j=1}^{nSP}n_j + \sum_{j=1}^{nSP}n_j\frac{\hat{\mu}_j}{\mathcal{R}T} \tag{4.95}$$

These two equations form a new iteration system in the variables π_i, $\Delta\ln(n)$ and Δn_j. The last variable is the correction on n_j, in fact after the iteration of the two main equations, the correction for mixture composition Δn_j is obtained from $\Delta\ln(n_j)$, found from equation (4.84), and then substituted to update n_j:

$$n_j = n_j + \Delta n_j \tag{4.96}$$

In this way only two coupled equations are solved from step to step, and the third variable need only to be updated. To extend this formulation to other state variables, the same approach is available in [16], where along with the extended treatment for Gibbs energy equilibrium in p for temperature, entropy and enthalpy, there is also an Helmoltz energy equilibrium formulation, which is supposed to treat the imposition of a thermodynamic state where one of the state variables is volume instead of pressure.

4.2.3 Reconstruction of mixture properties from mixture composition

The reconstruction of mixture properties can be readily performed, once the composition is calculated and since the tabulated values for every species are known. The enthalpy of the mixture is then

$$h = \sum_{j=1}^{nSP} n_j\hat{h}_{0,j}(T) \tag{4.97}$$

where $\hat{h}_{j,0}(T)$ is the enthalpy referred to the standard state $T = 0\ K$, $p = 1\ bar$. This value could be calculated for the single species, by considering the differential expression of enthalpy

$$d\hat{h}_i = \hat{c}_{p,i} dT \tag{4.98}$$

which leads to:

$$\hat{h}_i = \hat{h}_{i,ref} + \int\limits_{T_{ref}}^{T} \hat{c}_{p,i} dT \tag{4.99}$$

where $\hat{c}_p$ could be recovered from the Meyer relation

$$\hat{c}_{p,i} = \hat{c}_{v,i} + \mathcal{R} \tag{4.100}$$

where $\hat{c}_{v,i}$ could be obtained, e.g. from kinetic gas theory - see sec. 11.8 of [2], as:

$$\hat{c}_{v,i} = \begin{cases} \frac{3}{2}\mathcal{R} + \frac{\partial \hat{e}_{el}}{\partial T} & \text{for atomic species} \\ \frac{5}{2}\mathcal{R} + \frac{\left(\frac{\theta_{v,i}}{T}\right)^2 \exp\left(\frac{\theta_{v,i}}{T}\right)}{\left(\exp\left(\frac{\theta_{v,i}}{T}\right)-1\right)^2}\mathcal{R} + \frac{\partial \hat{e}_{el}}{\partial T} & \text{for molecular diatomic species} \end{cases} \tag{4.101}$$

where are cleary visible the translational ($\frac{3}{2}\mathcal{R}$ or $\frac{5}{2}\mathcal{R}$), vibrational $\left(\frac{\left(\frac{\theta_{v,i}}{T}\right)^2 \exp\left(\frac{\theta_{v,i}}{T}\right)}{\left(\exp\left(\frac{\theta_{v,i}}{T}\right)-1\right)^2}\mathcal{R}\right)$ and electronic $\left(\frac{\partial \hat{e}_{el}}{\partial T}\right)$ contributions. By using this method we would calculate $\hat{h}_{0,i}(T)$ as

$$\hat{h}_{0,i}(T) = \hat{h}_{i,ref.} + \int\limits_{T_{ref}}^{T} \hat{c}_{p,i} dT \tag{4.102}$$

However, this is not the approach followed in [16], where $\hat{h}_{0,i}(T)$ is obtained from extrapolation of tabulated data, hence is supposed as a known function for every species included in the analysis. In the same way, entropy is obtained as

$$s = \sum_{j=1}^{nSP} n_j \hat{s}_j \tag{4.103}$$

where the molar entropy contribution can be obtained, from (4.39), as:

$$\hat{s}_j = \hat{s}_{0,j}(T) - \mathcal{R}\ln\left(\frac{n_j}{n}\right) - \mathcal{R}\ln(p) \tag{4.104}$$

where

$$\hat{s}_{0,j}(T) = \hat{s}_{i,ref.} + \int_{T_{ref}}^{T} \hat{c}_{p,i} \frac{dT}{T} \tag{4.105}$$

can be obtained directly from tabulated data.

As one can see, differently from the molar enthalpy for the single species, which doesn't depend from the current mixture composition, molar entropy of the single species contains the term

$$\mathcal{R}\ln(p_j) = \mathcal{R}\ln\left(\frac{n_j}{n}\right) + \mathcal{R}\ln(p) \tag{4.106}$$

which represent the generation of entropy due to the mixing of molecular species, and for this reason term (4.106) is called *mixing entropy term.*

The specific heat can be obtained by the derivation of (4.97), which can be obtained in the following way:

$$\begin{aligned} c_p = \left.\frac{dh}{dT}\right|_p &= \frac{d}{dT}\left(\sum_{j=1}^{nSP} n_j \hat{h}_{0,j}(T)\right)_p = \\ = \sum_{j=1}^{nSP} n_j \left.\frac{d\hat{h}_{0,j}(T)}{dT}\right|_p &+ \sum_{j=1}^{nSP} \left.\frac{dn_j}{dT}\right|_p \hat{h}_{0,j}(T) = \\ = \sum_{j=1}^{nSP} n_j \hat{c}_{p0,j}(T) &+ \sum_{j=1}^{nSP} \left.\frac{dn_j}{dT}\right|_p \hat{h}_{0,j}(T) \end{aligned} \tag{4.107}$$

It is easy to divide in (4.107) the contribution of the frozen mixture and the reaction composition change contribution:

$$c_p = c_{p,f} + c_{p,r} \tag{4.108}$$

where

$$c_{p,f} = \sum_{j=1}^{nSP} n_j \hat{c}_{p0,j}(T) \tag{4.109}$$

with $\hat{c}_{p0,j}(T)$ obtained from tabulated data,
and

$$c_{p,r} = \sum_{j=1}^{nSP} \left.\frac{dn_j}{dT}\right|_p \hat{h}_{0,j}(T) \tag{4.110}$$

We want now to express c_p derivative with respect to logarithmic derivatives, in order to obtain an expression of the specific heat which can be directly linked to the correction variables adopted in our discrete system (4.93).

Since

$$d\ln(T) = \frac{1}{T}dT$$

and

$$d\ln(n_i) = \frac{1}{n_i}dn_i$$

we can rewrite (4.107) as

$$c_p = \sum_{j=1}^{nSP} n_j \hat{c}_{p0,j}(T) + \sum_{j=1}^{nSP} n_j \frac{\hat{h}_{0,j}(T)}{T}\left(\frac{d\ln n_j}{d\ln T}\right)_p \tag{4.111}$$

The derivative $\left(\frac{d\ln n_j}{d\ln T}\right)_p$ must be obtained from our correction variables. Now, if one take a glance at system (4.93) and remebers that it is obtained at imposed pressure, it is immediate to express its derivatives as:

$$\begin{cases} \left(\frac{\partial\ln(n_j)}{\partial\ln(T)}\right)_p - \left(\frac{\partial\ln(n)}{\partial\ln(T)}\right)_p - \sum_{i=1}^{nEL}\tilde{\nu}_{i,j}\left(\frac{\partial\pi_i}{\partial\ln(T)}\right)_p = \frac{\hat{h}_{0,j}}{\mathcal{R}T}, \\ j = 1 : nSP \\ \\ \sum_{j=1}^{nSP}\tilde{\nu}_{i,j}n_j\left(\frac{\partial\ln(n_j)}{\partial\ln(T)}\right)_p + \sum_{j=1}^{nSP}\tilde{\nu}_{i,j}\left(\frac{\partial n_j}{\partial\ln(T)}\right)_p = 0\ , \quad i = 1 : nEL \\ \\ \sum_{j=1}^{nSP} n_j\left(\frac{\partial\ln(n_j)}{\partial\ln(T)}\right)_p - n\left(\frac{\partial\ln(n)}{\partial\ln(T)}\right)_p = 0 \end{cases} \tag{4.112}$$

Exactly as in iteration system (4.93), it is possible to reduce the order of the system by eliminating $\left(\frac{\partial\ln(n_j)}{\partial\ln(T)}\right)_p$ by substituting it in the second and third equation from the first, hence obtaining the reduced system

$$\begin{cases} \sum_{k=1}^{nEL}\sum_{j=1}^{nSP} n_j\tilde{\nu}_{k,j}\tilde{\nu}_{i,j}\left(\frac{\partial\pi_i}{\partial\ln(T)}\right)_p + \sum_{j=1}^{nSP}\tilde{\nu}_{k,j}n_j\left(\frac{\partial\ln(n)}{\partial\ln(T)}\right)_p = -\sum_{j=1}^{nSP}\tilde{\nu}_{k,j}n_j\frac{\hat{h}_{0,j}}{\mathcal{R}T}, \\ i = 1 : nEL \\ \\ \sum_{i=1}^{nEL}\sum_{j=1}^{nSP}\tilde{\nu}_{i,j}n_j\left(\frac{\partial\pi_i}{\partial\ln(T)}\right)_p = -\sum_{j=1}^{nSP} n_j\frac{\hat{h}_{0,j}}{\mathcal{R}T} \end{cases} \tag{4.113}$$

Notice that in the second equation the term $\left(\sum_{j=1}^{nSP} n_j - n\right)\left(\frac{\partial \ln(n)}{\partial \ln(T)}\right)_p$ which should come from derivation doesn't appear, because since we are not evaluating the correction of an increment, the term $\left(\sum_{j=1}^{nSP} n_j - n\right)$ is zero.

Once system (4.113) is solved, we can calculate $\left(\frac{\partial \ln n_j}{\partial \ln T}\right)_p$ from the first equation of (4.112), and then we can reconstruct specific heat from (4.111). Including the substitution of $\left(\frac{\partial \ln n_j}{\partial \ln T}\right)_p$ in the expression of c_p adn dividing by $\mathcal{R}$, we can express the specific heat for the mixture as a direct function of the derivatives obtained from (4.113):

$$\begin{aligned}\frac{c_p}{\mathcal{R}} &= \sum_{j=1}^{nSP} n_j \frac{\hat{c}_{p0,j}(T)}{\mathcal{R}} + \sum_{k=1}^{nEL}\sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j \frac{\hat{h}_{0,j}(T)}{\mathcal{R}T}\left(\frac{\partial \pi_i}{\partial \ln(T)}\right)_p + \\ &+ \sum_{j=1}^{nSP} n_j \frac{\hat{h}_{0,j}(T)}{\mathcal{R}T}\left(\frac{\partial \ln(n)}{\partial \ln(T)}\right)_p + \sum_{j=1}^{nSP} n_j \left(\frac{\hat{h}_{0,j}(T)}{\mathcal{R}T}\right)^2 \end{aligned} \tag{4.114}$$

To obtain other important quantities, such as velocity of sound and c_v, we need the equilibrium system derivatives with respect to pressur. These derivatives can be obtained again by the differentiation of eqn. (4.72), (4.85) and (4.90), leading, following the same approach by which we obtained system (4.112), but considering now T fixed and p variable, thus obtaining:

$$\begin{cases} \left(\frac{\partial \ln(n_j)}{\partial \ln(p)}\right)_T - \left(\frac{\partial \ln(n)}{\partial \ln(p)}\right)_T - \sum_{i=1}^{nEL} \tilde{\nu}_{i,j}\left(\frac{\partial \pi_i}{\partial \ln(p)}\right)_T = -1, \\ j = 1 : nSP \\ \\ \sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j \left(\frac{\partial \ln(n_j)}{\partial \ln(p)}\right)_T + \sum_{j=1}^{nSP} \tilde{\nu}_{i,j}\left(\frac{\partial n_j}{\partial \ln(p)}\right)_T = 0\ , \quad i = 1 : nEL \\ \\ \sum_{j=1}^{nSP} n_j \left(\frac{\partial \ln(n_j)}{\partial \ln(p)}\right)_T - n\left(\frac{\partial \ln(n)}{\partial \ln(p)}\right)_T = 0 \end{cases} \tag{4.115}$$

which can be reduced, by the substitution of $\left(\frac{\partial \ln(n_j)}{\partial \ln(p)}\right)_T$ from the first to the other two derivatives, to the system

$$\begin{cases} \sum_{k=1}^{nEL} \sum_{j=1}^{nSP} n_j \tilde{\nu}_{k,j} \tilde{\nu}_{i,j} \left(\frac{\partial \pi_i}{\partial \ln(p)}\right)_T + \sum_{j=1}^{nSP} \tilde{\nu}_{k,j} n_j \left(\frac{\partial \ln(n)}{\partial \ln(p)}\right)_T = \sum_{j=1}^{nSP} \tilde{\nu}_{k,j} n_j, \\ i = 1 : nEL \\ \\ \sum_{i=1}^{nEL} \sum_{j=1}^{nSP} \tilde{\nu}_{i,j} n_j \left(\frac{\partial \pi_i}{\partial \ln(p)}\right)_T = \sum_{j=1}^{nSP} n_j \end{cases} \tag{4.116}$$

$\left(\frac{\partial \ln(n_j)}{\partial \ln(p)}\right)_T$ can be then obtained by the first equation of (4.115).
From the perfect gas equation of state in the form:

$$pV = nRT \tag{4.117}$$

we obtain the derivatives:

$$\left(\frac{\partial \ln V}{\partial \ln p}\right)_T = -1 + \left(\frac{\partial \ln n}{\partial \ln p}\right)_T \tag{4.118}$$

$$\left(\frac{\partial \ln V}{\partial \ln T}\right)_p = -1 + \left(\frac{\partial \ln n}{\partial \ln T}\right)_p \tag{4.119}$$

which are known from the resolution of systems (4.113) and (4.116).
The sound velocity a can be obtained from its definition

$$a^2 = \left(\frac{\partial p}{\partial \rho}\right)_S = \frac{P}{\rho}\left(\frac{\partial \ln p}{\partial \ln \rho}\right)_S = -\frac{P}{\rho}\left(\frac{\partial \ln p}{\partial \ln V}\right)_S \tag{4.120}$$

where $\left(\frac{\partial \ln p}{\partial \ln V}\right)_S$ can be written, as is reported in [16], as:

$$\left(\frac{\partial \ln p}{\partial \ln V}\right)_S = \frac{c_p}{c_p \left(\frac{\partial \ln V}{\partial \ln p}\right)_T + \frac{PV}{T}\left(\frac{\partial \ln V}{\partial \ln T}\right)_p^2} \tag{4.121}$$

or

$$\left(\frac{\partial \ln p}{\partial \ln V}\right)_S = \frac{c_p}{c_v \left(\frac{\partial \ln V}{\partial \ln p}\right)_T} \tag{4.122}$$

since

$$c_v = \left(\frac{\partial e}{\partial T}\right)_V = c_p + \frac{PV}{T}\frac{\left(\frac{\partial \ln V}{\partial \ln T}\right)_p^2}{\left(\frac{\partial \ln V}{\partial \ln p}\right)_T} \tag{4.123}$$

Now, we can define the specific heats ratio:

$$\gamma = \frac{c_p}{c_v} \tag{4.124}$$

which is not anymore constant, such as in the perfect gas hypothesis, but is dependent from the mixture composition, that in the hypothesis of thermochemical equilibrium is uniquely defined by the imposition of the thermodinamic state *(T,p)*, as we have seen in sec. (4.1), i.e.

$$\gamma = \frac{c_p\left(T,p\right)}{c_v\left(T,p\right)} = \gamma\left(T,p\right) \tag{4.125}$$

We conclude this section by showing a more compact definition of the sound velocity (4.120), by defining

$$\left(\frac{\partial \ln p}{\partial \ln V}\right)_S \equiv \gamma_S \tag{4.126}$$

hence obtaining the classical formulation

$$a^2 = n\mathcal{R}T\gamma_S \tag{4.127}$$

which, since $n = {}^1\!/\!_{\mathcal{M}}$ and $R = {}^{\mathcal{R}}\!/\!_{\mathcal{M}}$, can be written as:

$$a^2 = \gamma_S RT \tag{4.128}$$

4.2.4 Reconstruction of transport properties

Once we have calculated the thermodynamic properties of our mixture, we need to reconstruct its transport properties, especially the viscosity μ - which we will not confuse with the molar electrochemical potential $\hat{\mu}$, and the thermal conductivity k. As viscosity and thermal conductivity for the single species are tabulated as functions of temperature (see chapter 5 of [16]) we have to reconstruct μ and k for the mixture. The methods presented here are *approximate* and not exact as the thermochemical equilibrium determination method presented in sec. (4.2.2), and are derived from various correlations, collected in [16].

The viscosity for the mixture can be obtained from:

$$\mu = \sum_{i=1}^{nSP} \frac{X_i\mu_i}{X_i + \sum_{\substack{j=1 \\ j\neq i}}^{nSP} X_j\phi_{ij}} \tag{4.129}$$

where ϕ_{ij} is called the *viscosity interaction coefficient* between species i and species j, and can be obtained by the correlation:

$$\phi_{ij} = \frac{1}{4}\left[1 + \left(\frac{\mu_i}{\mu_j}\right)^{\frac{1}{2}} \left(\frac{\mathcal{M}_j}{\mathcal{M}_i}\right)^{\frac{1}{4}}\right]^2 \left(\frac{2\mathcal{M}_j}{\mathcal{M}_i + \mathcal{M}_j}\right)^{\frac{1}{2}} \tag{4.130}$$

For some species binary combinations, a *viscosity interaction parameter* μ_{ij} is included in the species data. In these cases, the interaction coefficient can be reconstructed from:

$$\phi_{ij} = \frac{\mu_i}{\mu_{ij}} \frac{2\mathcal{M}_j}{\mathcal{M}_i + \mathcal{M}_j} \tag{4.131}$$

To obtain the thermal conductivity coefficient k, we must reconstruct both the frozen contribution and the equilibrium contribution, in a similar way as we have seen for the c_p in eqn. (4.108):

$$k = k_f + k_r \tag{4.132}$$

The frozen thermal conductivity can be obtained from an interaction correlation form:

$$k_f = \sum_{i=1}^{nSP} \frac{X_i k_i}{X_i + \sum_{\substack{j=1 \\ j \neq i}}^{nSP} X_j \psi_{ij}} \tag{4.133}$$

where

$$\psi_{ij} = \phi_{ij}\left[1 + \frac{2.41\left(\mathcal{M}_i - \mathcal{M}_j\right)\left(\mathcal{M}_i - 0.142\mathcal{M}_j\right)}{\left(\mathcal{M}_i + \mathcal{M}_j\right)^2}\right] \tag{4.134}$$

The method for reaction thermal conductivity is a little less intuitive; the mixture reaction thermal conductivity can be obtained as:

$$k_r = \mathcal{R} \sum_{i=1}^{nREAC} \frac{\Delta_r \hat{h}_{0,i}(T)}{\mathcal{R}T} k_{r,i} \tag{4.135}$$

where $\Delta_r \hat{h}_{0,i}(T)$ are the molar enthalpies of reaction of the $nREAC$ chemical reactions occuring in our mixture, calculated as:

$$\Delta_r \hat{h}_{0,i}(T) = \sum_{k=1}^{nSP} \nu_{ik} \hat{h}_{0,i}(T) \quad i = 1 : nREAC \tag{4.136}$$

where ν_{ik} are the stoichiometric coefficients of the species i for the k-th reaction as defined in form (4.28), with $\nu_{ik} < 0$ for reactants and $\nu_{ik} > 0$ for products.

The reaction thermal conduction coefficients for the single species, $k_{r,i}$, can be obtained by solving the following set of linear equations:

$$\sum_{j=1}^{nREAC} g_{ij} k_{r,j} = \frac{\Delta_r \hat{h}_{0,i}(T)}{\mathcal{R}T} \quad i = 1 : nREAC \tag{4.137}$$

where the coefficients g_{ij} can be obtained by

$$g_{ij} = \sum_{k=1}^{nSP-1} \sum_{l=k+1}^{nSP} \left(\frac{\mathcal{R}T}{PD_{kl}} X_k X_l \right) \left(\frac{\nu_{ik}}{X_k} - \frac{\nu_{il}}{X_l} \right) \left(\frac{\nu_{jk}}{X_k} - \frac{\nu_{jl}}{X_l} \right) \tag{4.138}$$

where

$$\frac{\mathcal{R}T}{PD_{kl}} = \frac{5 \mathcal{M}_k \mathcal{M}_l}{3 A^*_{kl} \mu_{kl} (\mathcal{M}_k + \mathcal{M}_l)} \tag{4.139}$$

with A^*_{kl} factor is a collision cross-section ratio between molecules and μ_{kl} is the viscosity interaction factor just presented. For the extreme complexity to obtain all the terms in eqn. (4.139), the following approximate hypothesis is adopted in CEA, as justified in [16]:

$$\frac{\mathcal{R}T}{PD_{kl}} \cong 1.1 \tag{4.140}$$

We can now reconstruct the thermal conduction coefficient of the whole mixture.

The mixture Prandtl number can be reconstructed as:

$$Pr = \frac{c_p \mu}{k} \tag{4.141}$$

or, recalling eqn. (4.108) and (4.132), as:

$$Pr = Pr_f + Pr_r = \frac{c_{p,f} \mu}{k_f} + \frac{c_{p,r} \mu}{k_r} \tag{4.142}$$

4.2.5 Shock wave solution

We will now apply what we have studied in the previous subsection to the case of a normal shock. This formulation will be used in the analysis of a blunt body, where for blunt bodies we are considering both fusiform bodies and slender bodies, like wings, but with rounded noses or leading edges. For sharp edged bodies, like pointed nose fusiform bodies or sharp leading edges, the inclined shock formulation must be adopted.

This is not a problem, since the formulation presented here mantains its validity for the flow component normal to the shock, while for the tangential component the flow remains unaltered. The after shock properties can be calculated by calculating the discontinuity along the normal direction, using in the equilibrium shock formulation only the normal components, i.e. velocities u_1 and u_2 and normal Mach numbers $M_{n,1} = M_1 \cos\beta$ and $M_{n,2} = M_2 \cos(\beta - \theta)$, while tangential components remain unaltered - i.e. $v_1 = v_2$. Parallel Mach number is however subject to change due to the variation of the sound velocity throughh the shock:

$$M_{2,\|} = \frac{v_2}{a_2} \neq M_2 \sin(\beta - \theta)$$

The following figure explicates the terms and notations used in the oblique shock description:

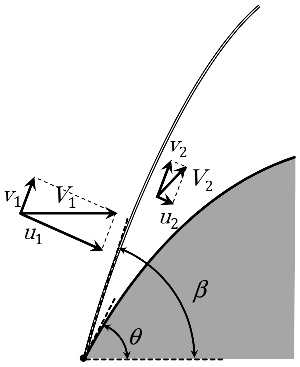


Figure 4.1: Oblique shock scheme with notation

Notice that, although the inclined shock waves are bent in the hypersonic case due to chemical non-equilibrium effects - not included in our model - the description of the classical inclined shock (see sec. 1.4.1 for a brief outlook) still works, because we are interested to the near-surface flowfield, and the shock inclination at the nose is the same with respect to the frozen flow equilibrium angle, i.e. relation (1.1), as can be seen clearly in sec. 15.3 of [2].

We will now present the equations used in CEA for the after-shock properties calculation, underlining the importance of using chemical equilibrium hypothesis, as already discussed in sec. 2.2.2, to obtain correct values of the thermodynamic state after the wave, since a perfect gas treatment would lead to completely wrong values. The flow through a shock wave preservs its mass flow, its momentum and its energy:

$$\begin{cases} \rho_2 u_2 = \rho_1 u_1 \\ p_2 + \rho_2 u_2^2 = p_1 + \rho_1 u_1^2 \\ h_2 + \frac{u_2^2}{2} = h_1 + \frac{u_1^2}{2} \end{cases} \tag{4.143}$$

By substituting u_2 from the first equation and rearranging we can obtain:

$$\begin{cases} \frac{p_2}{p_1} = 1 - \frac{\rho_1 u_1^2}{p_1}\left(\frac{\rho_1}{\rho_2} - 1\right) \\ \\ h_2 = h_1 + \frac{u_1^2}{2}\left[1 - \left(\frac{\rho_1}{\rho_2}\right)^2\right] \end{cases} \tag{4.144}$$

which can be rewritten in the form:

$$\begin{cases} \tilde{p} - \frac{p_2}{p_1} = 0 \\ \\ \frac{\tilde{h} - h_2}{\mathcal{R}} = 0 \end{cases} \tag{4.145}$$

with:

$$\tilde{p} = 1 - \frac{\rho_1 u_1^2}{p_1}\left(\frac{\rho_1}{\rho_2} - 1\right) \tag{4.146}$$

and:

$$\tilde{h} = h_1 + \frac{u_1^2}{2}\left[1 - \left(\frac{\rho_1}{\rho_2}\right)^2\right] \tag{4.147}$$

We will now express the variations of system (4.145), as we did with thermochemical equilibrium relations in the previous subsection, in order to provide a Newton - Rhapson algorithm solutor for the two shock equations. The variation will be expressed in terms of logarithmic derivatives with respect to the jump of *(p, T)* variables across the shock. The iterative system obtained from (4.145) is then:

$$\begin{cases} \frac{\partial\left(\tilde{p} - \frac{p_2}{p_1}\right)}{\partial \ln\left(\frac{p_2}{p_1}\right)} \Delta \ln\left(\frac{p_2}{p_1}\right) + \frac{\partial\left(\tilde{p} - \frac{p_2}{p_1}\right)}{\partial \ln\left(\frac{T_2}{T_1}\right)} \Delta \ln\left(\frac{T_2}{T_1}\right) = \frac{p_2}{p_1} - \tilde{p} \\ \\ \frac{\partial\left(\frac{\tilde{h} - h_2}{\mathcal{R}}\right)}{\partial \ln\left(\frac{p_2}{p_1}\right)} \Delta \ln\left(\frac{p_2}{p_1}\right) + \frac{\partial\left(\frac{\tilde{h} - h_2}{\mathcal{R}}\right)}{\partial \ln\left(\frac{T_2}{T_1}\right)} \Delta \ln\left(\frac{T_2}{T_1}\right) = \frac{h_2 - \tilde{h}}{\mathcal{R}} \end{cases} \tag{4.148}$$

where the iteration step is:

$$\begin{cases} \Delta \ln \left(\frac{p_2}{p_1}\right)_k = \ln \left(\frac{p_2}{p_1}\right)_{k+1} - \ln \left(\frac{p_2}{p_1}\right)_k \\ \\ \Delta \ln \left(\frac{T_2}{T_1}\right)_k = \ln \left(\frac{T_2}{T_1}\right)_{k+1} - \ln \left(\frac{T_2}{T_1}\right)_k \end{cases} \tag{4.149}$$

We can express the partial derivatives in system (4.148) in terms of the thermochemical derivatives $({}^{\partial \ln V}/_{\partial \ln p})_T$ and $({}^{\partial \ln V}/_{\partial \ln p})_T$, obtained in sec. (4.2.3) from (4.118) and (4.119), which are used and calculated in the equilibrium model. From the perfect gas state equation (4.1), considering that $R = {}^{\mathcal{R}}/_{\mathcal{M}}$, we can rewrite (4.146) as:

$$\tilde{p} = 1 - \frac{\mathcal{M}_1 u_1^2}{\mathcal{R} T_1} \left(\frac{\rho_1}{\rho_2} - 1\right) \tag{4.150}$$

From the state equation we have:

$$\frac{\rho_1}{\rho_2} = \frac{p_1}{p_2} \frac{\mathcal{M}_1}{\mathcal{M}_2} \frac{T_2}{T_1}$$

hence we can rewrite the logarithmic derivatives of our system as:

$$\frac{\partial \left(\tilde{p} - \frac{p_2}{p_1}\right)}{\partial \ln \left(\frac{p_2}{p_1}\right)} = -\frac{\rho_1}{\rho_2} \frac{\mathcal{M}_1 u_1^2}{\mathcal{R} T_1} \left(\frac{\partial \ln V}{\partial \ln p}\right)_{T,2} - \frac{p_2}{p_1} \tag{4.151}$$

$$\frac{\partial \left(\tilde{p} - \frac{p_2}{p_1}\right)}{\partial \ln \left(\frac{T_2}{T_1}\right)} = -\frac{\rho_1}{\rho_2} \frac{\mathcal{M}_1 u_1^2}{\mathcal{R} T_1} \left(\frac{\partial \ln V}{\partial \ln T}\right)_{p,2} \tag{4.152}$$

for the pressure equation, and:

$$\frac{\partial \left(\frac{\tilde{h} - h_2}{\mathcal{R}}\right)}{\partial \ln \left(\frac{p_2}{p_1}\right)} = -\frac{u_1^2}{\mathcal{R}} \left(\frac{\rho_1}{\rho_2}\right)^2 \left(\frac{\partial \ln V}{\partial \ln p}\right)_{T,2} + \frac{T_2}{\mathcal{M}_2} \left[\left(\frac{\partial \ln V}{\partial \ln T}\right)_{p,2} - 1\right] \tag{4.153}$$

$$\frac{\partial \left(\frac{\tilde{h} - h_2}{\mathcal{R}}\right)}{\partial \ln \left(\frac{T_2}{T_1}\right)} = -\frac{u_1^2}{\mathcal{R}} \left(\frac{\rho_1}{\rho_2}\right)^2 \left(\frac{\partial \ln V}{\partial \ln T}\right)_{p,2} - \frac{T_2 c_{p,2}}{\mathcal{R}} \tag{4.154}$$

for enthalpy conservation.
The right hand sides can be also rewritten as:

$$\frac{p_2}{p_1} - \tilde{p} = \frac{p_2}{p_1} - 1 + \frac{\mathcal{M}_1 u_1^2}{\mathcal{R} T_1} \left(\frac{\rho_1}{\rho_2} - 1\right) \tag{4.155}$$

from eqn. (4.150), and

$$\frac{h_2-\tilde{h}}{\mathcal{R}}=\frac{h_2-h_1}{\mathcal{R}}-\frac{u_1^2}{2\mathcal{R}}\left[1-\left(\frac{\rho_1}{\rho_2}\right)^2\right] \tag{4.156}$$

Since we have expressed the increments using the same derivatives of the thermochemical equilibrium problem, by means of (4.151), (4.152), (4.153), (4.154), we can now integrate the shock iterative solution system (4.148) with the thermochemical equilibrium iterative system (4.116), with the increment derivatives linked by (4.118) and (4.119). The solution of the thermochemical shock in equilibrium will be then adopted to obtain correct values of the thermodynamic quantities after the discontinuity wave. As we have already reminded in sec. 2.2.2, a thermochemical solution of the shock wave is *mandatory* in order to obtain correct results, since the huge increase of thermodynamic quantities through a hypersonic shock lead to a strong influence of *high temperature effects*, modifying the mixture composition. Anderson [2] made an esteem of shock-layer temperature on the Apollo re-entry vehicle, obtaining a result of 58·128 K with the perfect case hypothesis, against 11·600 K with the thermochemical equilibrium shock solution. Obviously the first result is absolutely unrelated with actual re-entry shock-layer temperatures, while the second is a reasonable esteem. This example clearly shows the fact that a viscous and thermal analysis on re-entering bodies *must* be performed using a chemical reacting gas model to treat the after-shock quantities determination; another proof will be presented in chapter 7, by comparing our results with state-of-the-art thermochemical CFD calculations.

If we consider now the flowfield region after the shock, the use of a thermochemical model of the gas looses in part its importance, since expansion tends to lower temperatures from the huge after-shock peak to more reasonable values, thus granting a gas behaviour closer to a frozen mixture. The idea to use the thermochemical equilibrium only to calculate gas properties after the shock is then tempting, since a full-body solution using equilibrium is quite expensive for the purposes of our algorithm. The current viscous solution is then obtained by calculating a chemical active shock, using equilibrium hypothesis, and then isoentropic perfect gas expansion relations are used, as if we would be considering a frozen mixture, but taking care to adopt the thermodynamic properties of the gas to those of the after-shock mixture obtained witht the chemical equilibrium model. As we will see in the next paragraphs, the loss in accuracy is truly lighter, if compared with the error we have if we calculate the shock without thermochemical equilibrium, thus justifying this approach to the viscous and thermal solution. Obviously one can also activate the thermochemical solutor on the full body of the vehicle, in order to have more accurate results. As we will see in the comparison with CFD data, this increment in accuracy is not excessively significative.

What we would like to clarify here is where actually in our algorithm the thermochemical equilibrium is adopted. We will confront here, over the by now familiar Dart ESA CFD calculations, the heat flow distributions obtained in the three conditions of:

- Thermochemical equilibrium imposed only through the shock wave, followed by a frozen composition gas model using the thermochemical properties calculated after the shock. Isoentropic frozen composition relations are used to obtain external flowfield variables and reference enthalpy variables from P_e and T^*- presented in our confront as *Shock Equilibrium*

- Thermochemical equilibrium imposed through the shock wave and in the determination of the external flowfield variables - presented in our confront as *Shock and External Equilibrium,* the variables for reference enthalpy still obtained by isoentropic frozen composition relations.

- Thermochemical equilibrium everywhere, presented as *Full Equilibrium*

Evidence during the development RAM v2.0 showed that the the use of a *Full* or *Shock and External* Thermochemical models does not improve the accuracy of the method, leading to heat flow values wich are almost identical to the *Shock Equilibrium* model, while the use of perfect gas hypotheses lead to completely wrong results. Since the use of the latter two thermochemical models requires a remarkable increase of computational effort and times, we are talking of 10-15 minutes of increased calculation time for the *Full* and *Shock and External* models with respect to the simple *Shock* equilibrium model, which needs to impose thermochemical equilibrium only downstream the shock, the simple *Shock* Thermochemical equilibrium is then adopted by our algorithm. The use of the *External* and *Full* models can be still easily triggered via appropriate flags while preprocessing the *.wgs file. The use of these models is however discouraged.

To conclude, we underline the fact that the perfect gas hypothesis is the worst possible approximation, confirming the arguments presented just above on the impossibility to use always a perfect gas model in an hypersonic problem. The error in q_w distribution could not seem dramatic, but if we confront results on other quantities, such as the shear stress τ_w, one can easily see that unaccetable error rises from the full perfect ideal gas (PIG) approximation:

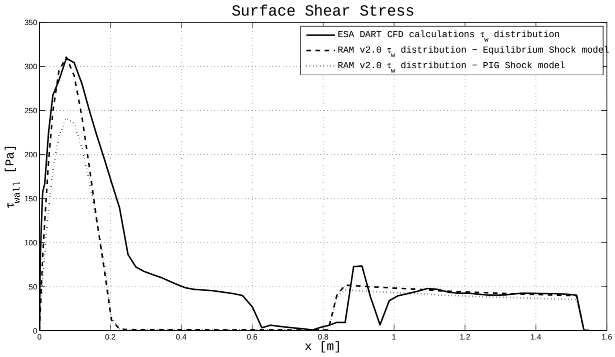


Figure 4.2: Dart τ_w calculations - 0° α and β- $M_\infty = 10.41$ - comparison between Shock with Thermochemical equilibrium and Shock with PIG model.

Part II

Algorithm for shear stress and heat flow calculation

Chapter 5

Approximate solution of hypersonic viscous flows

5.1 RAM v1.0 inviscid algorithm description

5.1.1 General description

The first version of the Re-entry Aerodynamic Module, as developed by Dominic Dirkx and actually implemented in STA Cambrian version, is a powerful tool which implements all the inviscid approximate methods discussed in chapter 2, including all the hypersonic Local Surface Inclination methods and expansion methods. RAM v1.0 algorithm is described in the next subsection, while its current software implementation will be discussed in sec. A.3.1, as Dominic's C++ classes for inviscid pressure field determination have been used as an input to our program, by developing an appropriate stand-alone software which included the RAM v1.0 aerodynamic classes.

In this part we will describe briefly Dominic's work, as the complete description has been given by the author in ref. [1]. RAM v1.0 implemented all the LSI compression and expansion methods presented in chapter 2, with the possibility for the user to choose the appropriate LSI method for compression/windward region - and expansion/leeward region, depending on the region bluntness an geometry. A look at figure 2.3 should remind immediatly what is intended for windward or compression region and leeward or expansion region. The particular method chosen can be differentiated also by considering a different method for the high Mach regime, since it has been proven that LSI method accuracy changes with mach number. Dirkx proposes the following differentiation of methods, based on different studies, considering region geometry an Mach number.

Table 5.1: Approximate inviscid method selections - from RAM v1.0 inviscid solver

Compression Methods	*Low Hypersonic* $M_\infty = 3 \div 8.5$	*High Hypersonic* $M_\infty = 6.5 \div \infty$
Blunt Region stagnation point, leading edge, etc.	Modified Newtonian- Prandtl Meyer expansion	Modified Newtonian- Prandtl Meyer expansion
Low Inclination Flat wing, tail, etc.	Tangent Wedge	Tangent Wedge Empirical
Low Inclination Curved fuselage, nacelle, etc.	Tangent Cone	Tangent Cone

Expansion Methods	*Low Hypersonic* $M_\infty = 3 \div 8.5$	*High Hypersonic* $M_\infty = 6.5 \div \infty$
Blunt Region	Newtonian ($C_p = 0$)	High M base pressure
Low Inclination Flat	ACM	Freestream Prandtl - Meyer
Low Inclination Curved	ACM	Freestream Prandtl - Meyer

As one can see, the range of Mach regimes is overlapping between the low and high hypersonic, as results in this region are bridged with a cubic function. The methods presented here are already discussed in chapter 2, with exception of the Freestream Prandtl - Meyer, which is simply the application of the Prandtl - Meyer relation described in sec. 2.4, but neglecting shock and windward region upstream contribution, thus expanding directly from freestream pressure to the local panel inclination value.

ACM, or *Aerodynamic Configured Missile*, is an empirical method used in missile preliminary design calculations, which extends the High Mach number base pressure formula to low Mach numbers, through the formulation:

$$c_p = max\left(-\frac{1}{M_\infty^2},\, \theta\frac{1}{16\,M_\infty^2}\right) \tag{5.1}$$

where θ_{rad} is the local inclination angle in degrees.

The reason to prefer a method to another sits in the different physical phenomenon it highlights. For example, tangent-wedge method is well representing a low inclination flat surface which is invested by a hypersonic flow, and tangent-cone add to this approach the contribution of a surface which is curved perpendicularly to the flow, thus describing precisely a curved, low inclination region.

Newton modified method describe better the stagnation process, since it emphasizes the concept of flow impact over a surface, as it is described in 2.2.1, and the correction matches the correct value of c_p after a shock. In this part we highlight a flaw in Dirkx's solutor, which is to consider the perfect gas frozen composition model to treat the shock, using perfect gas relations such as 2.17. As we have seen in chapter 4, this approach can lead to completely wrong results. Since RAM v1.0 doesn't include a thermochemical solver, this is the only possible approach, even if Dirkx mitigated it with the choice of a value of $\gamma = 1.2$ which is better related with air at hypersonic speeds and matches quite good with values encountered in our comparisons. If we consider for example von Karman IXV simulations, whose freestream condition can be found at sec. 7.1, we obtain the following values for the after-shock gas:

Table 5.2: After shock γ values from IXV simulations

Test n°	γ	M_∞
1	1.28	6
2	1.16	11
3	1.13	15.04

In the purpose of pressure distribution calculation, which is the main purpose of RAM v1.0 solver, the effect of having an approximated γ value produced however a great accuracy in pressure results. In spite of this, when considering other thermodynamic values determination, what emerged from comparison is a great dependance of the viscous solver, and of the significative viscous related variables such as c_p, T_w, Pr even to small modifications of γ value after the shock, as we will see in chap. 7. In addition to this, we also noticed the fact that in an equilibrium high temperature shock, the flow thermodynamic properties cannot be characterized only by a value of γ, hence requiring a more complex modeling of the flow, like the one we have from our thermochemical equilibrium model, as also Dirkx noticed it in his work, proposing a CEA extension of the gas model, which we have succesfully fulfilled. In conclusion, while relations like (2.17) can hold for the inviscid pressure field determination, with an appropriate choice of approximated γ, the viscous solver needs an equilibrium solution. Dirkx solutor can be then safely used for pressure field determination, as also his validation work on RAM v1.0 suggest, while for other after shock thermodynamic properties calculation, CEA equilibrium, as been presented in chapter 4, has been adopted, since perfect gas results for these properties in an hypersonic flowfield are surely unreliable.

5.1.2 Automatic geometry identification

Another interesting feature of Dominic algorithm, which is already implemented in the STA C++ classes we have used as an origin for inviscid solving of our vehicle, is the automatic selection of the vehicle region properties. As we will see in keen detail in the actual description of the RAM v2.0 implementation, in chapter 6, every vehicle is subdivided into several objects, which will be called parts, which can be, e.g.,wings, tails, fuselages. Dirkx elaborated an algorithm which could identify geometrical properties of different regions of a single part, starting from the indication of a fusiform or planar characterization of the part - the exact meaning of these attributes has been clarified when describing vehicle geometry in sec. 1.6. For example, referring to methods differentiation in table 5.1, Dirkx automated part analyzer can determine if a region of a part can be characterized as blunt or slender, depending on nose panels inclination, and if the planar parts are to be treated as curved or uncurved. The concept underlying the identification is to check if a region of the part has different features with respect to other parts of the vehicle, and if so split the part in two different parts, which can be treated separately by different approximated methods. We will present here only the baselines of Dirkx automated geometry identifier, a complete and detailed description being available in [1], since for the development and validation phase of RAM v2.0 we have not already linked its automated method identificator to our algorithm, preferring manual input for sake of simplicity during our algorithm implementation. Since some of the region characterizations are shared also by Simeonides viscous method, as can be seen in sec. 3.4.1, the automated selector proposed by Dirkx will be surely considered as an extended development of our algorithm.

Blunt part determination and splitting Fusiform parts are defined blunt if the area weighted mean inclination of the nose panels exceed a certain value defined by the user:

$$\bar{\theta}_i = \frac{\sum_{j=1}^{nPoints} \theta_{ij} A_{ij}}{\sum_{j=1}^{nPoints} A_{ij}} \tag{5.2}$$

where the index i is referred to the *i-th centroid contour*, the precise meaning of *centroid contour* being clarified in sec. 1.6, while the index j is related to the *j-th centroid point* over the *i-th* contour. Obviously the θ value presented here is taken at *zero AOA,* since we are dealing with a pure geometrical characterization of the vehicle.

If we refer ourselves to the front contour, i.e. $i = 1$, we can estabilish, if the mean inclination is higher than a certain value, if the body is to be considered blunt or not:

$$\bar{\theta}_1 > C_1 \tag{5.3}$$

For planar parts, blunt regions are typically leading edges of wings and tails. If a wing has an high sweep, it is possible that the leading edge should be not considered as blunt. For this reason, a sufficient part of the wing contour should be blunt enough for the whole wing leading edge being considered as blunt. To determine whether the wing leading edge is blunt or not, the most forward lateral wing *centroid contour* is examined, and the leading edge is said to be blunt if $C_2\%$ of the most lateral contour has a local inclination angle

$$\theta > C_3 \tag{5.4}$$

where the percentage $C_2\%$ is calculated from the areas of the contour, projected on the y-z normal to the vehicle axis plane, so that an high sweep leading edge will result with a decreased area with respect to a straight wing.

At this point we would like to split the blunt region part from the rest of the vehicle. Splitting exactly at the point where criteria (5.3) or (5.4) are met is unwise, since too rough for methods which require an expansion zone, such as the Modified Newton/Prandl-Meyer expansion method. Moving towards the rear from one fusiform contour to another, the following condition is searched:

$$x_{low} = min_x \left(\bar{\delta}_i < C_4 \right) \tag{5.5}$$

i.e. the minimum x position of the *i-th* contour where the contour-averaged inclination falls down a certain value C_4, which will be obviously minor than C_1. Taking a little buffer zone, we split then at

$$x_{split} = C_5 x_{low}$$

A final check is then made on the split location. If there are concavities in the fusiform part which undergo the analysis, a split must be performed including the concavity in the front region, and not performing the split at the concavity, by confusing it with a low inclination region. A concavity can be found if the local inclination is abruptly decreasing while moving from a contour to the next one, towards increasing x. The longitudinal derivative inclination must be considered then along all the centroid contours:

$$\frac{\partial \theta}{\partial x_j} = \frac{\theta_{i+1} - \theta_i}{x_{i+1,j} - x_{i,j}} \tag{5.6}$$

for every ensemble of points j along all contours i. If the concavity is in the rear part of the vehicle, however, it can be ignored. For a concavity to be taken into account, the following condition must be met:

$$x_{conc} < x_0 + C_6 x_{max} \tag{5.7}$$

where x_0 is the front coordinate and x_{max} is the back extreme coordinate. A new value of x_{low} after the concavity must be searched, to see when the body effectively starts to have low inclinated panels, this new value being named $x_{low,c}$. A split after the concavity can be then performed:

$$x_{split} = x_{conc} + C_5 \left(x_{low,c} - x_{conc} \right) \tag{5.8}$$

A condition on the fact that, if split point is almost in the back of the vehicle the split should be avoided, must be included. A warning must be issued, i.e. that x_{split} should be set between a contour and the next one x coordinates, in order not to split a single contour in two.

Curved or uncurved part determination Once all the part are splitted, the automatic identification algorithm defines if the part is curved or not. To do this, we first project the contour lines over the plane perpendicolar to part axis x (plane y-z for a fusifor part, e.g.). Then, for every contour *I,* a good argument to determine whether a part is curved or not is to confront all the angles differences between two points on the same contour, where ψ is the angle of the segment obtained by projecting contour I on the y-z plane with respect to the horizontal.

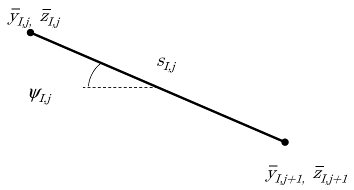

$\bar{y}_{I,j},\ \bar{z}_{I,j}$
$s_{I,j}$
$\psi_{I,j}$
$\bar{y}_{I,j+1},\ \bar{z}_{I,j+1}$

Figure 5.1: Projected contour segment on the y-z plane from contour I and with extremes points j and *j+1*, with angle $\psi_{I,j}$ with the horizontal and projected span $s_{I,j}$

One could think that if the angle between two adjacent segments becomes too big, the part could be considered as curved:

$$max_{I,j} \left| \psi_{I,j+1} - \psi_{I,j} \right| < k_1 \tag{5.9}$$

but an uncurved part, with a very little region of curvature, e.g. a wingtip, could be considered as curved, hence we will define an angle gradient by weighting it on the projected segment effective span, since in this way a curvature concentrated in a little zone of the part can be ignored:

$$\frac{\partial \psi_{I,j}}{\partial S_{I,j}} = \frac{|\psi_{I,j+1} - \psi_{I,j}|}{\frac{1}{2}\left(S'_{I,j} + S'_{I,j+1}\right)} \tag{5.10}$$

where $s^{'}_{I,j}$ is the weighted span of the projected segment $s_{I,j}$ on contour line I of extreme points j and *j+1*. To obtain a weighted span of $S_{I,j}$ over a contour line I, we define:

$$s^{'}_{I,j} = \frac{s_{I,j}}{\sum_{j=1}^{nPoints} s_{I,j}} \tag{5.11}$$

where $\sum_{j=1}^{nPoints} s_{I,j}$ is the perimeter of contour line I.

Once the angle spanwise derivative is defined, we can then considered uncurved a part if a certain percentage k_2% of its contour lines has

$$\frac{\partial \psi_{I,j}}{\partial S_{I,j}} < k_3 \tag{5.12}$$

This is the algorithm presented by Dirkx, some extimation on the user coefficients $C_1 \div C_6$ and $k_1 \div k_3$ has to be done. Dirkx proposes the following table:

Table 5.3: Automatic part identifier coefficients

Parameter	Value	Parameter	Value
C_1	60°	k_1	40°
C_2	40%	k_2	80%
C_3	50°	k_3	180°
C_4	30°	-	-
C_5	2	-	-
C_6	0.6	-	-

However, since the dependance over the particular shape of the vehicle cannot be determined *a priori*, all these values can be set by the user in RAM v1.0.

5.2 RAM v2.0 viscous algorithm description

In this subsection, we collect the results of the theorethical effort presented in the previous chapters, showing how the viscous and heating problem is actually solved, from the *.wgs geometry file to the final thermal and viscous analysis results. The actual implementation consists in a first preprocessing part, actuated by a stand-alone executable program called WgsReader, which performs the inviscid solving of the pressure field, using RAM v1.0 classes and the first part of RAM v2.0 preliminar calculations, including velocity field determination. The program provides a database file for the Matlab script, which completes the analysis with streamline determination and the thermal and viscous iteration on the reference enthalpy method, in order to obtain shear stress and heat flow distribution, as well as other interesting quantities, such as wall temperature and friction coefficient distribution on the surface and other surface and external flow fluid dynamic variables. The usage of WgsReader and the modification of Matlab input in order to use properly the algorithm will be explained in a tutorial over the reference case in appendix A, in this part we focus on the general outline of the algorithm.

The schema of the algorithm is presented here, while the single parts are deepened in the following sections of the chapter. The algorithm is conceptually split into two parts, the first being the inviscid flowfield determination, that, with respect to the pressure field determination provided by RAM v1.0, adds further information by defining an approximated velocity field, with stagnation points and streamlines definition over the whole part surface. Once streamlines are found, we can then afford the calculation of viscous and thermal properties, since the enthalpy reference methods are to be applied along the streamlines, knowing the boundary layer growth length from the stagnation point.

Inviscid calculations

- From WGS reader, the geometry file is processed, and a Matlab database file containing both RAM v1.0 solution - i.e. pressure field and approximated velocity field for the given freestream, incidence and sideslip conditions - is obtained for the single part.

- Velocity derivatives are obtained from quadrilateral interpolation parametrization of the velocity field, see section 6.3, and a topological analysis on the velocity field jacobian eigenvectors is provided, to be used in the stagnation point determination.

- Forced separation is imposed, for those parts of the vehicle with abrupt change of geometry features, on which the hypothesis of an inviscid attached external flow is to be considered absurd - e.g. vehicle flat rear zone.

- Stagnation point are determined, using topological criteria which will be dealt with in sec. 5.5, and the eigenvalue analysis is performed in the velocity field derivation. For fusiform bodies, a single stagnation point is searched, while on planar bodies we have attachment lines, so a stagnation point for every contour is found, thus leading to multiple attachment points.

- Since we have defined a velocity field and identified stagnation points on the surface, we can then calculate streamlines. Search trees are built before the iteration in order to perform fastest searches on the vehicle part points, to find the nearest centroid over a given streamline point. The algorithm to obtain streamlines, with the integration and surface projection routine is presented in sec. 5.6.

- Once the streamlines are defined, the boundary layer running length for every surface panel centroid is then calculated, by defining the curvilinear length along the streamline, from the stagnation point to the nearest point of the nearest streamline to the given centroid. A detailed procedure is presented along with the streamline calculator description in sec. 5.6.

- We have now the necessary inputs for viscous and thermal algorithm.

Viscous and heat flow calculations

- Shock properties are determined via the thermochemical equilibrium shock solver exposed in chapter 4, having imposed the freestream conditions.

- From the thermodynamic properties calculated after the shock, the external flowfield is calculated, assuming isoentropic expansion on the surface. The mixture is considered frozen from this point on, coherently with the results obtained by the end of chapter 4. A chemical equilibrium point per point solution is still selectable via flag setting, even if it not recommendable.

- Once the external thermodynamic state is solved, the iterative viscous and thermal solution is searched. The iteration variable is the wall temperature T_w, which is unknown. The iteration cycle is performed on the viscous and thermal solution at the same time, since they are self - interacting. We report here the sub - cycle of the viscous and thermal iteration: starting from an initial guess of T_w, taken as half of the external temperatureT_e, we iterate, from every centroid $\mathbf{x}_{c,ij}$ - with i the centroid contour index and j the point index - on the wall temperature

$T_{w,k}$ where k is the iteration step on temperature over the point $\mathbf{x}_{c,ij}$. The k-cycle is then:

- ▷ From $T_{w,k}$, the corresponding reference temperature T^* is calculated, via Eckert or Meador Smart relations, using laminar hypothesis.
- ▷ Thermodynamic properties, such as μ^*, ρ^* are calculated from T^* and P_e - due to the external pressure constant gradient through the boundary layer.
- ▷ Re^*_x is then calculated from reference properties and streamline running lengths.
- ▷ Transition control is applied, as stated by eqn. 3.40.
    - ▷ If the flow is turbulent, appropriate Simeonides coefficients for the turbulent flow are selected from table 3.3, depending on the bluntness or slenderness of the part region, and eqn. 3.29 is solved and τ_w is obtained.
    - ▷ If the flow is laminar, laminar coefficients are selected from table 3.2, according also in this case for the bluntness or slenderness of the body. τ_w is calculated from eqn. 3.29.
- ▷ Friction factor is calculated, using *freestream* non-dimensionalization for the actual c_f:

$$c_f = \frac{\tau_w}{\frac{1}{2}\rho_\infty V_\infty^2} \tag{5.13}$$

- ▷ Recovery enthalpy is calculated, as proposed in sec. 3.6.2, as $h_{rec} = h_e + \frac{1}{2} r_e V_e^2$
- ▷ Stanton number and heat flow relative to temperature $T_{w,k}$ are calculated:

$$St = \frac{c_{f.}}{2s} \tag{5.14}$$

  with $s = Pr^{*\frac{2}{3}}$ for laminar flows, or $s = 0.815$ for turbulent flow. The heat flow at iteration k is then

$$q_{w,k} = \rho_\infty V_\infty^2 \, St \, (h_{rec} - c_{p,w} T_{w,k}) \tag{5.15}$$

  Due to the frozen composition hypothesis after the shock, we have, in practice: $c_{p,w} = c_{p,e}$

- ▷ At this point, equilibrium between the convective heat flow from the fluid and the radiation emitted by vehicle surface is imposed, under the hypotheses of adiabatic wall and non radiating gas, the latter hypothesis being acceptable under 11·000 K - i.e. except for superorbital re-entry vehicles:

$$q_{w,k} = q_{w,rad,k} \tag{5.16}$$

with

$$q_{w,rad,k} = \sigma \epsilon_w T_{w,k}^4 \tag{5.17}$$

where $\sigma = 5.67e^{-8}$ is the Stefan-Boltzmann constant, and ϵ_w is the surface emissivity, which has been taken as *0.8* in accordance with the value adopted in VKI IXV CFD simulations. Since the equation is not linear in $T_{w,k}$, a Newton-Rhapson iteration is performed, defining:

$$f_k = q_{w,rad,k} - q_{w,k} = \sigma \epsilon_w T_{w,k}^4 - \rho_\infty V_\infty^2 \, St \, (h_{rec} - c_{p,w} T_{w,k}) = 0 \tag{5.18}$$

and

$$f_k' = \frac{\partial f_k}{\partial T_{w,k}} = 4\sigma \epsilon_w T_{w,k}^4 + \rho_\infty V_\infty^2 \, St \, c_{p,w} \tag{5.19}$$

hence obtaining:

$$T_{w,k+1} = T_{w,k} - \frac{f_k}{f_k'} \tag{5.20}$$

▷ The iteration on k continues until the maximum number of iterations, which has been set to 20, is reached, or the relative difference between two iterations

$$\Delta T_k = \frac{|T_{w,k+1} - T_{w,k}|}{T_e} \tag{5.21}$$

is less than $1e^{-20}$ K.

- Once the viscous - thermal iteration is concluded for every centroid $\mathbf{x}_{c,ij}$, the friction coefficient c_f, the shear stress τ_w and the heat flow q_w, as well as the surface temperature T_w, are determined in every centroid, hence the viscous and thermal problem are solved.

- Once shear stress distribution τ_w is known, its contribution to the vehicle aerodynamic coefficients can be determined, following the procedure in sec. 5.7.

5.3 Velocity field determination

As we have explained in the introductory chapter, in hypersonic fields the hypothesis of thin shock layer is one of the main features characterizing the flow. For this hypothesis, which is also one of the fundamentals of LSI methods such as Newton and Newton modified methods (see chapter 2), we can make a very strong approximation on the surface velocity field, by defining the velocity on the surface as the tangential component of the freestream velocity. This hypothesis, which would surely be very rough in every other aerodynamic regime, becomes justifiable in an hypersonic context. As the shock wave approaches the body, the flowfield tends to change its direction abruptly from freestream direction to tangential, maintaining its tangential momentum while being heavily slowed down in the direction normal to the shock wave, as the oblique shock theory suggest - see fig. 4.1. This consideration leads to the tangential velocity body surface formula, eqn. (1.2), which we report here:

$$\mathbf{V}_{\text{body}} = \mathbf{V}_{\|} = \mathbf{V}_{\infty} - \mathbf{V}_{\perp} = \mathbf{V}_{\infty} - \hat{\mathbf{n}} \cdot \mathbf{V}_{\infty} \cdot \hat{\mathbf{n}} \tag{5.22}$$

This hypothesis permits to obtain almost instantaneously a coherent approximation of the windward velocity field, thus saving ourselves from the great computational effort which is always linked with a numerical solution of a three dimensional flowfield from Euler or Navier-Stokes equations. Such hypothesis is, together with the LSI methods, the backbone of any approximate hypersonic solutor, having been used both by Gentry in the S/HABP [3], and in CBAERO algorithm by Kinney, as is clearly stated in his program presentation reported in [5]. The velocity surface field results provide very good agreement with actual CFD data in the windward region, as the flow in this region fits very well to the thin shock layer model.

On the leeward side, however, such proximity between the shock wave and the vehicle surface gives out, becoming an unsatisfying hypothesis. For the same reason, the LSI methods, which are mainly based on the thin shock layer hypothesis in the *compression* region, have to use other methods when describing the *expansion* region, as can be clearly seen in chapter 2. In particular, the use of tangential velocity leads to an overextimation of leeward velocity, hence suggesting to adopt a different relation in this region of flow. Once the external flowfield is solved, by assuming isoentropic expansion after the shock, and knowing the pressure distribution from the application of a LSI method, external enthalpy h_e can be calculated, as it is done in the external variables determination in our algorithm. Leeward velocity can be then obtained from the conservation of total enthalpy, as:

$$V_{\text{body, leeward}} = \sqrt{2\left(h_0 - h_e\right)}$$

where h_0 is the total enthalpy, calculated at the stagnation point as:

$$h_0 = c_{p,s} T_0$$

with T_0 taken from the isoentropic relations:

$$T_0 = T_s \left[1 + \frac{1}{2} \left(\gamma_s - 1 \right) M_s^2 \right]$$

where the subscript s indicates that the isoentropic arrest has been performed with the frozen gas hypothesis, but with the gas properties calculated after the equilibrium shock, hence considering the mixture change which has taken place through the wave. The velocity direction is always assumed considering the tangential-to-freestream direction, coherently with the work of Kinney.

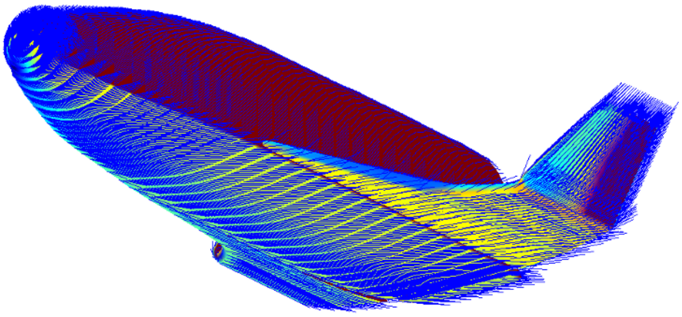


Figure 5.2: Tangential Velocity Surface field - Horus geometry - $\alpha = 45°$ and $\beta = 30°$

The surface velocity field has been then identified over the whole part, and we can proceed with its analysis, defining separation regions after steep angles, stagnation points and streamline calculation.

5.4 Forced separation algorithm

In the study of typical re-entry geometries, we encounter often features like flat back regions, where the flow cannot be obviously attached, and the application of the algorithm leads to unlikely streamlines patterns on the back of the vehicle, as can be seen in the following figure, where streamlines calculation over an IXV geometry at incidence and sideslip is performed.

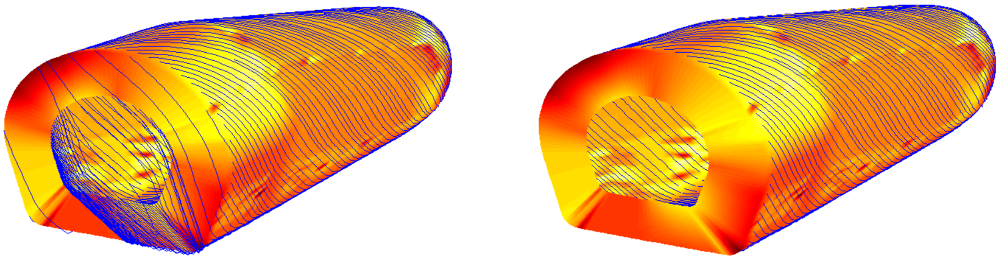


Figure 5.3: Streamlines calculation on IXV geometry without forced separation and with forced separation - $\alpha = 45°$ and $\beta = 30°$

As it can be seen, the streamlines pattern on the back cannot lead to accurate results. After such steep angles, a separation is hence imposed. The problem is to define a convenient identification of these steep geometries, in order to perform separation in the correct region, independently from the flow orientation. A immediate criteria to identify an abrupt geometry change is, on a fusiform part, to calculate the velocity vector angle variation from a contour to the next. On the *i-th* contour we calculate then the angular variation of the velocity vector between contours i and *i+1* (or *i-1* if we are on the last contour), for every point j of the contour, and then we take the mean over the entire contour i:

$$\Delta\theta_{ij} = \arccos\left(\frac{\vec{V}_{i+1,j,\text{body}} \cdot \vec{V}_{i,j,\text{body}}}{\left\|\vec{V}_{i+1,j,\text{body}}\right\| \left\|\vec{V}_{i,j,\text{body}}\right\|}\right) \tag{5.23}$$

Then the mean value is taken over every point j of the contour i, defining one variation value for every contour:

$$\Delta\theta_i = \Delta\bar{\theta}_{ij} \tag{5.24}$$

The detachment location is identified by the contour on which we have the maximum mean angular variation. The actual detachment is imposed two contours before the maximum $\Delta\theta_i$, because otherwise the algorithm would be still propagating streamlines towards the detachment region.

$$i_{\text{detach}} = i_{\max_i \Delta\theta_i} - 2 \tag{5.25}$$

In fact, since the velocity on the body is defined as the freestream velocity component tangential to the surface, an abrupt velocity angular variation coincides with an abrupt geometrical shape variation, so we impose there separation.

5.5 Stagnation point determination

In order to obtain the correct boundary layer effective running length, the calculation of the distance along the streamline must be performed from the stagnation point. Another reason to esteem the position of the stagnation point is to reduce suitably the integrator step in order to reconstruct the stagnation region, in which we have strong velocity gradients mixed with a rough geometry. The choice of a too loose iteration step may produce integration errors, which result in a wrong definition of the running lenghts over the vehicle surface, as can be seen in the following example:

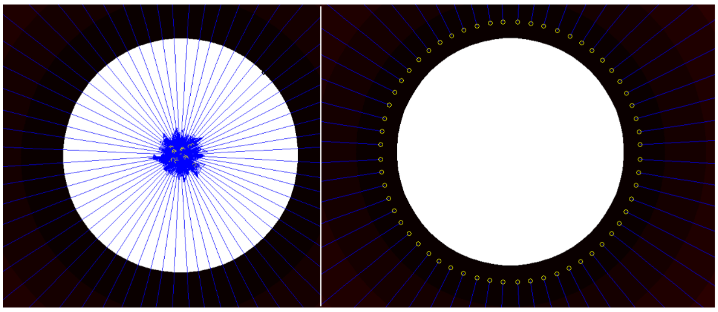


Figure 5.4: Streamlines over Dart geometry at zero incidence - on the left we can see streamlines calculated assuming the same step-size over the vehicle surface leading to a convergence leak on streamlines iteration, on the right a refinement criteria based on the proximity to the calculated stagnation point, is used.

A definition of the stagnation point over the vehicle surface has to be found. An interesting work by Kentwright et al., [17], provides an automatic criterion for identifying singular features in a velocity field, such as separations and attachment points or lines. This criterion is used in other professional hypersonic solvers such CBAERO, as Kinney observes in [5]. In their work, the authors identify a suitable method, which can define the presence of an attachment or detachment point by means of analytical study of the local bi-dimensional surface flowfield. While analyzing two-dimensional velocity fields, the authors noticed that in the proximity of asimptotically converging streamlines regions, i.e. near a singularity point or line, one of the two eigenvectors of the jacboian $\nabla\mathbf{V}$ of the local velocity field $\mathbf{V}$ aimed to be parallel to the velocity field itself. Again, the authors proposed that if the aligning eigenvector is the largest, we have a separation, while if it is the smallest, we are near a flow attachment point. The condition for attachment is then

$$\mathbf{e_2} \times \mathbf{V}_{body} = 0 \tag{5.26}$$

$\mathbf{e_2}$ being the smallest eigenvector of the jacobian $\nabla\mathbf{V}$, while we have detachment for

$$\mathbf{e_1} \times \mathbf{V}_{body} = 0 \tag{5.27}$$

with $\mathbf{e_1}$ the largest eigenvalue.

The attachment or separation criteria is applied only when both the eigenvalues are real, since real eigenvalues are linked to saddle point, or maximum/minimum points in the local velocity field, corresponding to the attachment or detachment point we are looking for.

As we will discuss in chapter 6, streamlines, velocity field and its derivatives are calculated using quadrilateral interpolation between centroid points, in order to obtain converging integration and higher accuracy. The practical definition of the local eigenvalues field will be presented along with the interpolation parametrization in sec. 6.3, since the eigenvalue analysis will be performed over the parametrized element. Our algorithm does not provide an analytical velocity field, since the velocity field definition comes from an approximated hypothesis, and due to the usual roughness of the LaWGS mesh, imposed by STA standard mesh and by computational times limitation, the direct results provided by criteria (5.26) and (5.27) have proven to be unadequate. We have to remember that the authors proposed these criteria as a tool to detect separation and attachment lines over a bi-dimensional flowfield, while we are working over a three dimensional field with approximated derivative tools, as will be exposed in sec. 6.3.

Considering these limitations, a modified criterion has been extrapolated from Kentwright work, and has been proven to be effective over several LaWGS geometries. The identification, derived directly from eqn.s (5.26) and (5.27), resulted more robust and independent from geometrical shape and mesh discontinuities, which are common in LaWGS formats. Eqn.s (5.26) and (5.27) in fact showed an excessive conditioning over mesh irregularities. A topological alignment variable is defined, in every centroid point j on the *i-th* contour , as

$$al_{ij} = \max\left(\left|\mathbf{e_1} \times \mathbf{V}_{ij,\mathrm{body}}\right|, \left|\mathbf{e_2} \times \mathbf{V}_{ij,\mathrm{body}}\right|\right) \tag{5.28}$$

When al_{ij} aims to zero, we are approaching the stagnation point. Imaginary results are excluded from the definition, accordingly with Kentwright work. Hence the minimum alignment coefficient identifies the stagnation centroid, i.e. the nearest centroid to the stagnation point:

$$i_{\mathrm{stag}}, j_{\mathrm{stag}} \longleftrightarrow \min\left(al_{ij}\right)$$

$$\mathbf{x}_{c,\mathrm{stag}} = \mathbf{x}_{c,i_{\mathrm{stag}}j_{\mathrm{stag}}} \tag{5.29}$$

Obviously, the alignment tends to zero even in the detachment region, hence we are considering, in finding the minimum value of al_{ij}, only the region which is not included in the detached area identified in the previous sub section from criterion (5.25).

The point determined in this way is not the actual stagnation point, but the centroid point nearest to stagnation, which can be offset with respect to a stagnation point. This centroid is used by the algorithm to define a proper surrounding area, identified in a 7x7 centroids mesh, in which integration step is reduced in order to avoid the streamline convergence fail presented in figure 5.4. The following figure presents stagnation point identification via the al_{ij} minimum criteria over the Dart geometry at incidence, and the corresponding step reduction area, which leads to streamline convergence as presented in fig. 5.4. On this interval, quadrilateral interpolation is also performed, if the user has selected this kind of velocity interpolation in the stagnation region.

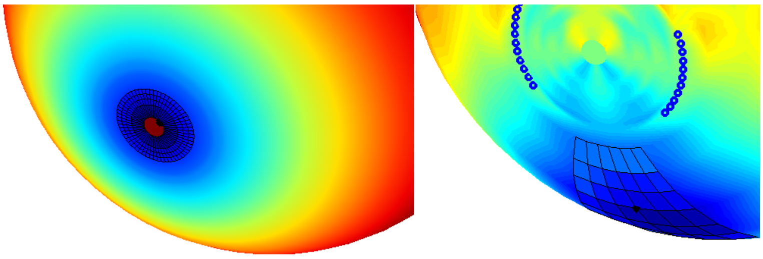


Figure 5.5: Stagnation centroid point definition through jacobian eigenvectors alignment minimization criteria (black diamond) and corresponding integration step refinement area, represented by the square grid. Blue circles represent ignored imaginary alignments, while surface colouring shows calculated al_{ij} value. DART reference geometry is used. If stagnation is in the furthermost lines, the step is refined in those first lines (left), if it is away from the first lines, it is refined over a 7x7 patch (right).

For fusiform bodies, the search of the minimum al_{ij} value is performed over the whole surface, while for non-fusiform bodies, a local stagnation point for every contour line is defined, leading to a stagnation line definition over the leading edge.

Actual stagnation point are obtained on every streamline: when the integration procedure provides a stationary solution, since streamlines are calculated from the detached region towards the stagnation area. Actual stagnation points are represented by yellow points in fig. 5.4. However, this procedure alone would not be sufficient if a proper step refinement would not be adopted near $\mathbf{x}_{c,\text{stag}}$, defined via the eigenvector alignment minimization criterium.

If $\mathbf{x}_{c,\mathrm{stag}}$ is found over the furthermost contour lines, i.e. $i_{\mathbf{x}_{c,stag}} < 3$, choosing the surrounding panels does not lead to an accurate step refinement criteria. In this case the step is refined over the whole first 3 front lines of the fusiform part.

The choice of the number of centroids composing the refinement mesh, such as the 7x7 centroids patch rather than the first 5 lines is linked to the fact that, due to the adaptive step selection which we will expose subsequently, the Runge Kutta integration step does not find a new streamline point beyond two or three adjacents centroids. We have chosen to refine in an area from 5 to 7 adjacent centroids only to guarantee an high robustness to geometry peculiarities.

Another function of knowing the nearest centroid to the stagnation point is to arrest streamline calculation when the true stagnation point is inside the first contour line, a thing which happens often for fusiform geometries at small angles of attack. If the actual stagnation point is circumscribed by the first contour line, we cannot properly interpolate velocity field, since we cannot define properly 4 centroids to define a quadrilateral over which the interpolation can be performed. This is in fact a true hole in our mesh, which lead to a completely wrong streamline calculation in the nose region, as can be seen in fig. 5.4. We could fix this problem by defining a triangular interpolation patch for this region, taking two centroids on the first line as the base segment and the vertex point of the vehicle as the third point. However, since the definition of parametrical quadrilateral derivatives is a great effort, as we will see in sec. 6.3, and some derivation and interpolation features have been specifically found and developed by us specificatedly for this algorithm, this feature has not been implemented yet. An effective and easy to adopt solution is to cut the streamline when it goes past the first line further x coordinate, *when* $\mathbf{x}_{c,stag}$ *is found on the first contour line.* The importance of the calculation of $\mathbf{x}_{c,\mathrm{stag}}$ via the topological study of the surface velocity field is then increased.

This solution leads to an almost perfect streamline calculation when the angle of attack α and the angle of sideslip β are small, as can be seen in fig. 5.4. A little error in the streamline running length calculation, proportional to the distance from the cutting point to the actual stagnation point, will be however introduced. In the comparisons over the DART geometry, as can be seen in the end of chapters 3 and 4, this error showed to be not significantly influencing the nose region c_f or q_w distribution, hence while a satisfying algorithm for the particular mesh conditions of the very fore region of our vehicle. Obviously, this problem is only encountered in fusiform bodies.

To conclude the procedure of $\mathbf{x}_{c,\mathrm{stag}}$ we must notice the last fix one has to perform in order to obtain a correct alignment map a_{ij}. Due to the approximated criteria on which velocity field and eigenvalues analysis is performed, during the algorithm development we have found the possibility to obtain some outlier values, which cause the detection of an erroneous eigenvalues alignment and can lead to wrong stagnation point identification. To correct these values, a blur gaussian filtering has been applied to the a_{ij} matrix, hence smoothing the isolated alignment singularities which come from numerical errors, and enhancing the correct alignment detection.

The gaussian blur has been performed only in the *j-wise* direction, to avoid windowing effects. In that direction, due to the axial geometry of a fusiform body, we can border the matrix with the adjacent j-values, reproducing a periodic infinite signal. The filtering is applied on the bordered matrix, which is then trimmed again, so that the values at the extremes are not displaying any kind of windowing effect. This procedure can not be applied in the *i-wise* direction, since after the last contour and before the first contour we have no data. Filtering by a 2d gaussian blur would hence flatten the local minimum, which identifies stagnation, towards the borders - i.e. the first contour line if the stagnation point is near the aft of the vehicle - due to the presence of windowing. A wrong stagnation point would be then found, leading to an error in $\mathbf{x}_{c,\mathrm{stag}}$ determination. A mono-dimensional filtering along the j direction provides succesful outliers trimming while not modyifing the stagnation region alignment pattern.

5.6 Streamline calculation

5.6.1 Introduction and analytical treatment

Once we have defined the velocity field over the vehicle surface, and the singularities which characterize a typical flow over an hypersonic re-entering body, are succesfully dealt with, the calculation of streamlines can begin. In this section we present the outline of the streamlines calculation process, while some implementative aspects are left to the next chapter, for their importance and extension.

Streamlines can be obtained directly from the integration of the velocity field, which can be done, following the approach given in Gentry S/HABP software manual [3], from the differential definition of velocity:

$$\frac{dx}{dt} = V_x, \quad \frac{dy}{dt} = V_y, \quad \frac{dz}{dt} = V_z \tag{5.30}$$

Since velocity modulus can be defined as:

$$V = \frac{ds}{dt} \tag{5.31}$$

we can express the local increment in terms of the infinitesimal increment step ds in terms of velocity director cosines C_i:

$$\frac{dx}{ds} = \frac{V_x}{V} = C_x, \quad \frac{dy}{ds} = \frac{V_y}{V} = C_y, \quad \frac{dz}{ds} = \frac{V_z}{V} = C_z \tag{5.32}$$

The numerical integration of relations (5.32) will lead to streamlines definition. Actually, these differential relations are integrated using a Runge-Kutta second order method, which will be presented in detail in the next chapter. Usually, in the calculation of streamlines over a given body, only two of these equations are actually integrated, the third equation being a constraint equation describing local geometry of the form

$$z = f(x, y) \tag{5.33}$$

A global definition of such a function is not possible over geometries of the complexity we are treating here, since we would not have an univocal definition of the z coordinate given x and y, so in our algorithm we will integrate over the three equations provided in (5.32). The resulting streamlines are not mathematically forced on the vehicle surface, so at every step we will project conveniently the resulting streamline iteration on the vehicle surface.

During the calculation of streamlines, the main problem we encounter is linked to the fact that a coarse surface meshing is obtained from the LaWGS format geometry description. In fact LaWGS is not truly an aerodynamic mesh of the vehicle, but instead an aerodynamic panelization for LSI, or for other aerodynamics panel methods, if we speak of other regimes than the hypersonic. Due to this fact, streamlines cannot be defined on the coarse LaWGS centroid mesh, but are defined separatedly, as independent points which can move freely following the integration process.

The fact that streamlines geometry is not directly linked to centroid mesh leads to three main issues to be solved. The first one is the fact that, as we are not constrained on the vehicle surface, a projection of the calculated streamline point on the panels identified by the nearest centroids has to be made.

In order to find the surrounding centroids to define a proper *panel* on which perform the projection of the streamline, we must know which of the centroid mesh points is the closest to our streamline calculated point. Hence, a global search on the vehicle LaWGS mesh has to be performed. Such a search proved to be the most expensive computation in the streamline calculation process, hence a certain optimization via *search trees* and other devices will have to be found. Search trees are a hyerarchical structure which can be used to improve nearest-neighbour searches, and are proposed also by Kinney in [5]; more details on this tool will be given in chapter 6.

The last issue linked to the independence of streamlines coordinates from centroid mesh is the fact that, if we are moving through a streamline, and we want to know the value of the velocity there in order to integrate the next step, we need to interpolate the velocity field values from the surrounding centroids to our actual position, since surface velocity vectors have been defined only in centroid points. A quadrilateral linear interpolation will be performed, from the surrounding centroid points, to the actual projected streamline point in order to define an accurate velocity vector for the next integration. Solving the streamline problem without interpolation has hence lead to unsatisfying results. The quadrilateral interpolation tool will be presented togheter with the other implementation tools used in the streamline solutor in chapter 6.

5.6.2 General outline of streamline solutor

We report here the general structure of the streamline integration algorithm, explaining every step in the rest of the section. The algoritm cycles on the *i-th* contour from the aft towards the fore of a fusifuform body, starting after the detached region, and then on the *j-th* point of the *i-th* contour. We are moving from the back of the vehicle towards the front, hence using an inverse propagation, since this solution is more stable. Starting from the nose would lead to The velocity field is then reversed with respect to the one defined in sec. 5.3:

$$\mathbf{V}_{\text{stream}} = -\mathbf{V}_{\text{body}} \tag{5.34}$$

For a non fusiform body the procedure is the same, except that the contours usually start from the root of a wing and go toward the tip, hence streamline calculation follows this order, while mantaining an inverse propagation, since the stagnation points are usually in the front. Before the algorithm starts, search trees of the LaWGS mesh are generated, in order to perform hierarchic searches over the part geometry.

At every rs iteration, being r the line index of the starting point, and s the point index over the r line, the following procedures are fulfilled:

- The starting point of the new iteration is selected as the centroid *rs,* except if another streamline is not already passing nearby. Starting from the last non-detached contour, and moving to the nose, streamlines are calculated taking every centroids point s of the *r-th* contour as a starting point. Before initializing a streamline, however, a check is made, that another streamline generated by an aft point does not pass near the centroid. In fact, if another streamline is already passing nearby the centroid, we can already esteem the running length using that streamline, instead of generating a new one, hence saving a huge quantity of calculation time.

- ▶ The centroid indexes *I,J,IP1,JP1* surrounding centroid rs are calculated, in order to define a proper panel, defined by the four vertex points

  $$\mathbf{x}_{c,\,I,J},\ \mathbf{x}_{c,\,IP1,J}, \mathbf{x}_{c,\,IP1,JP1}, \mathbf{x}_{c,\,I,JP1}$$

  where *I,J,IP1,JP1* are adjusted indexes in order to not exceed part indexing. *I,J,IP1,JP1* are usually chosen as *i, j, i+1, j+1,* except when they are reaching indexing limit. If the contours limits is exceeded, i.e. $i+1 > nLC$, being nLC the numbers of contour lines of the part, indexes are set to have non-exceeding values: for $i = nLC$ we have *I = nLC-1, IP1 = nCL.* The same thing is done with contour lines, having *J = j, JP1 = j+1,* except when we are reaching the last point, in such case we will have *J = j, JP1 = j+1.* We will use also *JM1, IM1* in the rest of the section to indicate the same non-exceeding indexing of *i-1* and *j-1.*

- ▶ The datas of the starting streamline point are saved. To save streamlines data a 4-dimensional array is created, defined as:

  $$\mathbf{X}_{calc}\,(it, d, r, s)$$

  where it is the iteration index, d the data storage index, *r,s* are the indexes of the starting point. Given a starting point $\bar{r}$, $\bar{s}$, at every iteration it six variables are saved in the array indexed by $d = 1 \div 5$:

  - ▷ *x,y,z* coordinates of the current streamline point at iteration *it.*
  - ▷ I and J indexes of the nearest centroid to the actual streamline.

- ▶ Another data saving variable, linked to $\mathbf{X}_{calc}$ array, is used to save the nearest streamline passing near a given centroid $\mathbf{x}_{c,IJ}$. This variable, defined as **saveDist** is structured as:

  $$\textbf{saveDist}\,(I, J, d)$$

  where I and J are the reference indexes of the centroid **saveDist** is referring to, and d is another data array, with $d = 1 \div 4$, which stores:

  - ▷ The current distance of the nearest streamline to $\mathbf{x}_{c,IJ}$ at the present iteration time.
  - ▷ r and s indexes of the starting point of the nearest streamline, hence identifying a single streamline from the $\mathbf{X}_{calc}$ data structure.
  - ▷ The iteration it of the nearest streamline at which the nearest distance from $\mathbf{x}_{c,IJ}$ is reached.

- At this point the iteration over the single streamline is performed, and Runge Kutta iteration procedure begins, being *it* the actual iteration along the streamline:

    - I and J indexes of the nearest centroid to the last streamline point found on the previous iteration, which we will refer as $\mathbf{x}_{\text{stream},it}$, and last streamline point are retrieved from

    $$\mathbf{X}_{calc}\left(it, d, r, s\right)$$

    - The panel surrounding $\mathbf{x}_{\text{stream},it}$ is retrieved having been identified by the four indexes $I, J, IP1, JP1$ found at the previous iteration, which identify the panel vertices $\mathbf{x}_{c,\,I,J}$, $\mathbf{x}_{c,\,I,JP1}$, $\mathbf{x}_{c,\,IP1,JP1}$, $\mathbf{x}_{c,\,IP1,J}$, as can be seen in figure:

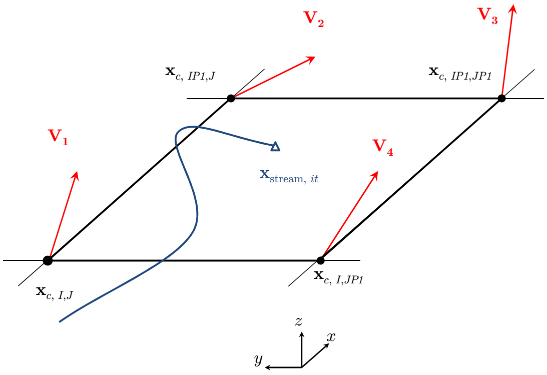


Figure 5.6: Centroid panel surrounding the streamline point $\mathbf{x}_{\text{stream},it}$, with the consequent velocites defined on every centroid as in sec. 5.3, which will be interpolated over the panel and used for the Runge-Kutta step.

    - Runge Kutta iteration step is calculated, by taking the minimum distance h_k between centroid $\mathbf{x}_{c,\,I,J}$ and its eight surrounding centroids, and dividing it for a coefficient which depends on the nature of the analyzed part and on the proximity to the stagnation nearest centroid $\mathbf{x}_{c,stag}$:

    $$h = \frac{\min\left(h_k\right)}{\Delta_h} \quad , k = 1 \div 8 \tag{5.35}$$

    where Δ_h assumes the values of

    $$\begin{cases} 1 \text{ if we are outside the stagnation step refinement region, as defined in 5.5} \\ 10 \text{ if we are over a fusiform part in the stagnation step refinement region} \\ 5 \;\; \text{if we are over a non-fusiform part in the stagnation step refinement region} \end{cases}$$

- ▷ Having found the correct iteration step h, a Runge Kutta second order iteration is performed over the panel surrounding $\mathbf{x}_{\text{stream},it}$. Through the iteration, quadrilateral interpolation is applied to find an accurate value of the velocity vector by the streamline point and the intermediate integration points, using the velocity values defined on the corner of the panel. These steps will be deepened in the next chapter, since their analytical formulation is not trivial.
- ▷ The new centroid panel, identified by the indexes *I, IP1, J, JP1,* surrounding the newly found streamline point, is identified over a 7x7 centroid grid surrounding the old nearest centroid $\mathbf{x}_{c,IJ}$, in the following method:
  - ▷ First, the nearest centroid $\mathbf{x}_{c,\,In,Jn}$ to the new streamline point $\mathbf{x}_{\text{stream},it+1}$ is found
  - ▷ Then, the four centroids $\mathbf{x}_{c,\,IP1,JP1}$, $\mathbf{x}_{c,\,IP1,JM1}$, $\mathbf{x}_{c,\,IM1,JP1}$ and $\mathbf{x}_{c,\,IM1,JM1}$ are taken as vertex of a search panel, as showed in fig. 5.7
  - ▷ Over the extended search panel, diagonal vectors $\mathbf{v}_1$, $\mathbf{v}_2$, $\mathbf{v}_3$, $\mathbf{v}_4$ are found. The actual new panel surrounding $\mathbf{x}_{\text{stream},it+1}$ is found by comparing these vectors with the distance vector from the nearest centroid $\mathbf{x}_{c,\,In,Jn}$ to the streamline point $\mathbf{x}_{\text{stream},it+1}$, defined as v_Δ. The panel surrounding the streamline point will be then the one with the diagonal vector $\mathbf{v}_k$ forming the maximum cosine with $\mathbf{v}_\Delta$ - i.e. the most aligned vector. The direction chosen is then the direction k which gives:

    $$k_{max} \leftrightarrow \max_k \left(\mathbf{v}_k \cdot \mathbf{v}_\Delta\right)$$

  - ▷ Once the direction k is found, the corresponding vertices of the new panel surrounding $\mathbf{x}_{\text{stream},it+1}$ are easily found. They are reindexed such as the IJ point is the point with the minimum line and point index i,j , the other poins being defined coherently with the line contour and point contour indexing.

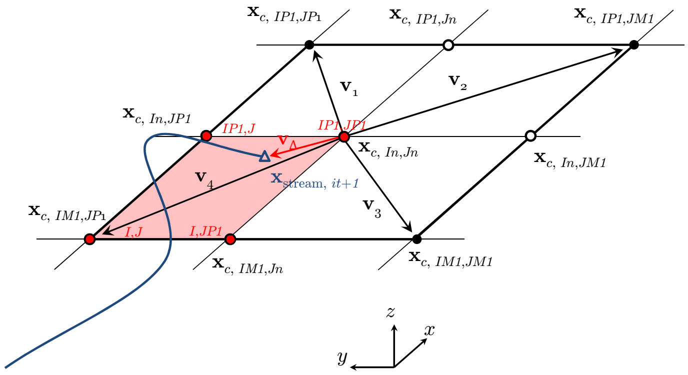

$\mathbf{x}_{c,\ IP1,JP1}$
$\mathbf{x}_{c,\ IP1,Jn}$
$\mathbf{x}_{c,\ IP1,JM1}$
$\mathbf{v}_1$
$\mathbf{v}_2$
$\mathbf{x}_{c,\ In,JP1}$
$IP1,J$
$IP1,JP1$
$\mathbf{v}_\Delta$
$\mathbf{x}_{c,\ In,Jn}$
$\mathbf{x}_{c,\ In,JM1}$
$\mathbf{v}_4$
$\mathbf{x}_{\text{stream},\ it+1}$
$\mathbf{x}_{c,\ IM1,JP1}$
$\mathbf{v}_3$
I,J
$I,JP1$
$\mathbf{x}_{c,\ IM1,Jn}$
$\mathbf{x}_{c,\ IM1,JM1}$
z
x
y

Figure 5.7: New panel search algorithm: the newly calculated streamline point $\mathbf{x}_{\text{stream},it+1}$ being the blue triangle. Centroid points are represented as dots. In this case, the maximum scalar product is between $\mathbf{v}_\Delta$ and $\mathbf{v}_4$. Hence the actual coordinates of the panel surrounding $\mathbf{x}_{\text{stream},it+1}$ will be $(IM1, JP1)$, $(IM1, Jn)$, (In, Jn),$(In, JP1)$, marked in red. They will be re-indexed as the new surrounding panel coordinate, coherently with the red labels in figure.

- ▷ The point is projected over the surrounding panel, since as stated in sec. 5.6.1, since we don't have an analytical defined surface, the calculated point might not adhere to the vehicle surface. To project the point, the normal of the panel is obtained by its vertices:
  - ▷ Diagonals are obtained:

$$\mathbf{d}_1 = \mathbf{x}_{c,\,IP1,JP1} - \mathbf{x}_{c,\,I,J} \tag{5.36}$$

$$\mathbf{d}_2 = \mathbf{x}_{c,\,I,JP1} - \mathbf{x}_{c,\,IP1,J} \tag{5.37}$$

  - ▷ Normal versor is obtained then by:

$$\hat{\mathbf{n}} = \frac{\mathbf{d}_1 \times \mathbf{d}_2}{\|\mathbf{d}_1 \times \mathbf{d}_2\|} \tag{5.38}$$

  - ▷ The center of the panel is then found by

$$\mathbf{x}_{m,IJ} = \left\{ \begin{array}{c} x_m \\ y_m \\ z_m \end{array} \right\}_{IJ} = \frac{1}{4} \left\{ \begin{array}{c} x_{I,J} + x_{I,JP1} + x_{IP1,JP1} + x_{IP1,J} \\ y_{I,J} + y_{I,JP1} + y_{IP1,JP1} + y_{IP1,J} \\ z_{I,J} + z_{I,JP1} + z_{IP1,JP1} + z_{IP1,J} \end{array} \right\} \tag{5.39}$$

The distance vector from the panel midpoint is:

$$\mathbf{x}_0 = \mathbf{x}_{\text{stream},it+1} - \mathbf{x}_{m,IJ} \tag{5.40}$$

The projected point is:

$$\mathbf{x}_0\big|_p = \mathbf{x}_0 - (\mathbf{x}_0 \cdot \hat{\mathbf{n}})\,\hat{\mathbf{n}} \tag{5.41}$$

The projected streamline point is then calculated by going back from the $\mathbf{x}_{m,IJ}$ reference to the global system:

$$\mathbf{x}_{\text{stream},it+1}\big|_p = \mathbf{x}_0\big|_p + \mathbf{x}_{\text{stream},it+1} \tag{5.42}$$

- ▷ The new streamline point coordinates are now on the surface, they can hence be saved in the streamline data array:

  $$\mathbf{X}_{calc}\,(it+1, d, r, s)$$

  Notice that every streamline is identified by its starting points indexes *rs.*

- ▷ At this point, the nearest centroid $\mathbf{x}_{c,In,Jn}$ to the newly calculated streamline point is found using a nearest point search algorithm, and its indexes are saved in $\mathbf{X}_{calc}$

- ▷ The nearest centroid indexes *In, Jn* are saved also in an external map, so that the algorithm can recognize that a streamline is already passing nearby that particular centroid, and it deselect it as a potential starting point. In this way, when the algorithm cycles throug *r* and *s,* the indexes corresponding to a centroid with a nearby passing streamline are avoided. In this way, the number of streamlines to be calculated to associate at least a single streamline to a centroid is greatly reduced, saving a huge amount of computational time

- ▷ For every step, it is verified if the distance between $\mathbf{x}_{\text{stream},it+1}$ and $\mathbf{x}_{c,\,In,Jn}$ identifies the actual streamline as the nearest of all the streamlines passing nearby $\mathbf{x}_{c,\,In,Jn}$. If the condition is met, the current distance, the current iteration *it + 1* and the centroid indexes *r,s* are saved in

  $$\mathbf{saveDist}\,(r, s, d = \{streamDist, r, s, it+1\})$$

  Hence, we can uniquely identificate for every centroid of the mesh what is the closest streamline passing nearby, through its starting point indexing *rs.* This feature will be crucial for boundary layer running length determination.

- ▷ At this point we have two controls which save the so far calculated streamline, cut it and break the cycle and re-intialize it from the next starting point if:

    - ▷ The part is not fusiform and the streamlines exit from the lateral boundary of the part, then the streamline is cut at the part boundary.
    - ▷ The part is fusiform and the stagnation point lies inside the small nose region part not covered by the mesh, as explained in sec. 5.5, hence the streamline must be cut off in order to not leave the body.
  - ▷ The *it* step is concluded, the counter is set as *it = it + 1* and the cycle is re-initialized from a new starting point, until every point on the centroid mesh is cleared or sees a streamline passing by.

- ▶ The streamline solutor delivers then the streamline information, contained in the multidimensional arrays $\mathbf{X}_{calc}\,(it, d, r, s)$ and **saveDist** (In, Jn, d), to the running length determination routine, which will be explained in the following section.

- ▶ Running lenghts can be determined, since after the calculation of all the streamlines needed, we have defined, for every centroid $\mathbf{x}_{c,ij}$, the nearest streamline passing by through the **saveDist** array. We have calculated the streamlines with a backward integration, so from the last point to the stagnation. Distance is obviously taken from stagnation to the last point, so for every streamline array identified by an rs we cycle from the last iteration to the first, measuring the increment in the three coordinates, for k decreasing from the last iteration to the first, as:

  - ▷
$$\Delta x_{rs,k} = \left(x_{\text{stream},k+1} - x_{\text{stream},k}\right)_{rs} \tag{5.43}$$
  - ▷
$$\Delta y_{rs,k} = \left(y_{\text{stream},k+1} - y_{\text{stream},k}\right)_{rs} \tag{5.44}$$
  - ▷
$$\Delta z_{rs,k} = \left(z_{\text{stream},k+1} - z_{\text{stream},k}\right)_{rs} \tag{5.45}$$
  - ▷ The curvilinear step along the streamline is then
$$\Delta s_{rs,k} = \left\| \begin{Bmatrix} \Delta x_{rs,k} \\ \Delta y_{rs,k} \\ \Delta z_{rs,k} \end{Bmatrix} \right\| \tag{5.46}$$
  - ▷ The running length is then
$$r_{rs,k+1} = r_{rs,k} + \Delta s_{rs,k} \tag{5.47}$$

5.7 Aerodynamic coefficients calculation

5.7.1 General

Once we have succesfully run the viscous algorithm over a part, we know, in every centroid of the part, the surface pressure value $p_{e,ij}$ - from RAM v1.0 - and the shear stress at the wall $\tau_{w,ij}$ - from RAM v2.0. We are then able to reconstruct the aerodynamic coefficients' contribution of the part, both for the inviscid and for the viscous components. Forces and moments will be obtained in the *body axes* frame, coherent with the LaWGS definition, see sec. 1.6, and then transformed in the *wind axes* frame, parallel to the velocity vector, in order to obtain classical definition of lift, drag and sideforce, and aerodynamic moments. At this point, aerodynamic coefficients are obtained by the appropriate non - dimensionalization. Notice that the conventions we take as *body axes* or *wind axes* might be not coherent with homonymous literature definitions. Aerodynamic forces and moments will be however obtained, at the end of the transformation, in the most typical flight dynamic convention rule.

5.7.2 Body frame forces and moments definition

Reconstruction of aerodynamic forces and moments in body axes from the pressure and shear stress distribution is quite immediate. The global contribution of pressure distribution over the part surface is:

$$\mathbf{F}^b_{\text{inv}} = \int_A P\tilde{\mathbf{n}}dA = -\int_A P\hat{\mathbf{n}}dA \tag{5.48}$$

being $\tilde{\mathbf{n}}$ the body normal entering the surface, and$\hat{\mathbf{n}}$ the body normal exiting the surface, coerent with our definition of centroid normals $\hat{\mathbf{n}}_{ij}$. The apex b denotes the fact we are measuring forces in body axes. The contribution due to viscous effects is, similarly:

$$\mathbf{F}^b_{\text{visc}} = \int_A \tau_w\hat{\mathbf{t}}dA \tag{5.49}$$

where $\hat{\mathbf{t}}$ is the versor indicating the local tangent to the velocity field. Since the velocity in the centroid points is known, it can be obtained by normalizing the local velocity vector, as being defined in sec. 5.3:

$$\hat{\mathbf{t}}_{ij} = \frac{\mathbf{V}_{ij}}{V_{ij}} \tag{5.50}$$

Referring to our centroids discretized geometry, we can express the individual forces on a single centroid as:

$$\mathbf{f}_{\text{inv},ij} = -A_{ij} P_{e,ij} \hat{\mathbf{n}}_{ij} \tag{5.51}$$

for the invscid contribution, and

$$\mathbf{f}_{\text{visc},ij} = A_{ij} \tau_{w,ij} \hat{\mathbf{t}}_{ij} \tag{5.52}$$

being A_{ij} the LaWGS panel area associated at the centroid $\mathbf{x}_{c,ij}$, as defined in sec. 1.6.

for the viscous contribution. Eqn.s (5.48) and (5.49) become then:

$$\mathbf{F}^b_{\text{inv}} = \sum_{i=1}^{nLC} \sum_{j=1}^{nPC} \mathbf{f}_{\text{inv},ij} = -\sum_{i=1}^{nLC} \sum_{j=1}^{nPC} A_{ij} P_{e,ij} \hat{\mathbf{n}}_{ij} \tag{5.53}$$

$$\mathbf{F}^b_{\text{visc}} = \sum_{i=1}^{nLC} \sum_{j=1}^{nPC} \mathbf{f}_{\text{visc},ij} = \sum_{i=1}^{nLC} \sum_{j=1}^{nPC} A_{ij} \tau_{w,ij} \hat{\mathbf{t}}_{ij} \tag{5.54}$$

The global forces acting on our part, in body axes, are then:

$$\mathbf{F}^b = \mathbf{F}^b_{\text{inv}} + \mathbf{F}^b_{\text{visc}} \tag{5.55}$$

In analogy with forces, moments can be defined as:

$$\mathbf{M}^b_{o,\text{inv}} = \int_A P \left(\mathbf{r_o} \times \tilde{\mathbf{n}}\right) dA = -\int_A P \left(\mathbf{r_o} \times \hat{\mathbf{n}}\right) dA \tag{5.56}$$

$$\mathbf{M}^b_{o,\text{inv}} = \int_A \tau_w \left(\mathbf{r_o} \times \hat{\mathbf{t}}\right) dA \tag{5.57}$$

Using the discretized geometry, we have that the single moment over every centroid is

$$\mathbf{m}_{\text{inv},ij} = -A_{ij} P_{e,ij} \mathbf{r_o}_{ij} \times \hat{\mathbf{n}}_{ij} = \mathbf{r_o}_{ij} \times \mathbf{f}_{\text{inv},ij} \tag{5.58}$$

for the invscid contribution, and

$$\mathbf{m}_{\text{visc},ij} = A_{ij} \tau_{w,ij} \mathbf{r_o}_{ij} \times \hat{\mathbf{t}}_{ij} = \mathbf{r_o}_{ij} \times \mathbf{f}_{\text{visc},ij} \tag{5.59}$$

$\mathbf{r_o}_{ij}$ being the distance between the current centroid and the moment reference point $\mathbf{x}_o$:

$$\mathbf{r_o}_{ij} = \mathbf{x}_{c,ij} - \mathbf{x}_o \tag{5.60}$$

Developing the cross product, we have:

$$\begin{cases} m_{\text{inv}\,x,\,ij} = f_{\text{inv}\,z,\,ij}\; r_{o\,y,\,ij} - f_{\text{inv}\,y,\,ij}\; r_{o\,z,\,ij} \\ m_{\text{inv}\,y,\,ij} = f_{\text{inv}\,x,\,ij}\; r_{o\,z,\,ij} - f_{\text{inv}\,z,\,ij}\; r_{o\,x,\,ij} \\ m_{\text{inv}\,z,\,ij} = f_{\text{inv}\,y,\,ij}\; r_{o\,x,\,ij} - f_{\text{inv}\,x,\,ij}\; r_{o\,y,\,ij} \end{cases} \tag{5.61}$$

from the inviscid centroid forces, and

$$\begin{cases} m_{\text{visc}\,x,\,ij} = f_{\text{visc}\,z,\,ij}\; r_{o\,y,\,ij} - f_{\text{visc}\,y,\,ij}\; r_{o\,z,\,ij} \\ m_{\text{visc}\,y,\,ij} = f_{\text{visc}\,x,\,ij}\; r_{o\,z,\,ij} - f_{\text{visc}\,z,\,ij}\; r_{o\,x,\,ij} \\ m_{\text{visc}\,z,\,ij} = f_{\text{visc}\,y,\,ij}\; r_{o\,x,\,ij} - f_{\text{visc}\,x,\,ij}\; r_{o\,y,\,ij} \end{cases} \tag{5.62}$$

from the viscous centroid forces.
Moments acting on the single part can be defined then as:

$$\mathbf{M}^b_{\text{inv}} = \sum_{i=1}^{nLC} \sum_{j=1}^{nPC} \mathbf{m}_{\text{inv},ij} \tag{5.63}$$

$$\mathbf{M}^b_{\text{visc}} = \sum_{i=1}^{nLC} \sum_{j=1}^{nPC} \mathbf{m}_{\text{visc},ij} \tag{5.64}$$

The global moments acting on the part in body axes are then:

$$\mathbf{M}^b = \mathbf{M}^b_{\text{inv}} + \mathbf{M}^b_{\text{visc}} \tag{5.65}$$

Figure 5.8 shows forces and moment conventions over the body axes and the panel centroids.

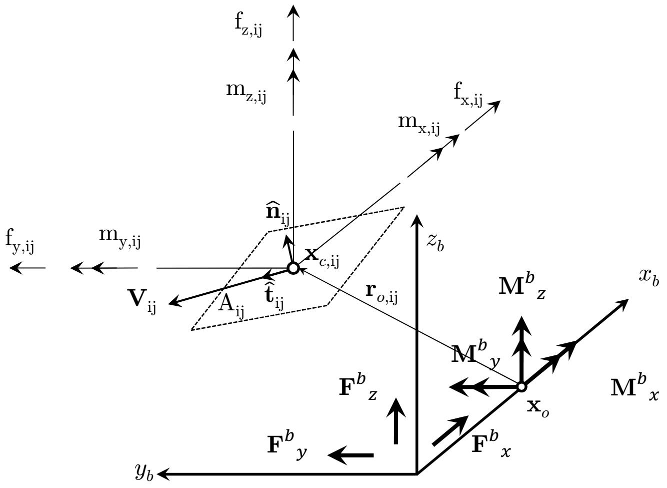


Figure 5.8: Body axes forces and moment definitions and conventions

5.7.3 Wind frame forces and moments transformation

We have obtained forces and moments in the body axes frame. Aerodynamic forces are typically calculated in a different frame, the wind axes frame. The wind axes frame presented here may differ slightly from the most common definitions found in literature, its definition being consistent with the one used in RAM v1.0, as exposed in [1]. x_w axis is aligned against the freestream velocity vector $\mathbf{V}_\infty$, z_w axis is perpendicular to the x_w axis and lying on the vehicle symmetry plane, hence is aligned with the lift force. y_w is chosen in order to form a right-handed frame. To transform the body axes into wind axes, two angles have to be defined, and they are the *angle of attack* α - sometimes defined as AOA - which measures the angle between the vehicle or part longitudinal axis and the freestream velocity parallel to the vehicle symmetry plane $\mathbf{V}_{\infty,||}$ and the *angle of sideslip* β, which measures the angle between the freestream velocity and the symmetry plane of the vehicle. These angles are usually defined in the freestream conditions of the RAM v2.0 solver in the WgsReader program input settings. Notice that our sign convention for β is opposite to the classical flight dynamics conventions, in order to maintain us coherent with the RAM v1.0 definitions.

To obtain forces and moments in the wind frame, a transformation from the body axes has to be performed. First, a rotation of the body frame is performed along α, from the body frame b to an intermediate reference frame i. The rotation can be measured in the frame b with the rotation matrix:

$$\mathbf{R}_1^b = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \tag{5.66}$$

However, as can be seen in fig. 5.9, x and y axes are in the opposite directions. This transformation can be represented by the operator:

$$\mathbf{T} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.67}$$

The two transformation can be joined in a single operator $\tilde{\mathbf{R}}_1^b$:

$$\tilde{\mathbf{R}}_1^b = \mathbf{T}\mathbf{R}_1^b = \begin{bmatrix} -\cos\alpha & 0 & -\sin\alpha \\ 0 & -1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix} \tag{5.68}$$

Another rotation, along angle β, is performed from the intermediate frame i to the final wind axes frame w, which can be measured in the intermediate frame with the rotation matrix:

$$\mathbf{R}_2^i = \begin{bmatrix} \cos\beta & \sin\beta & 0 \\ -\sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5.69}$$

The following figure shows the two trasformations $\tilde{\mathbf{R}}_1^b$ and $\mathbf{R}_2^i$ required to transform the coordinate system from body axes to wind axes.

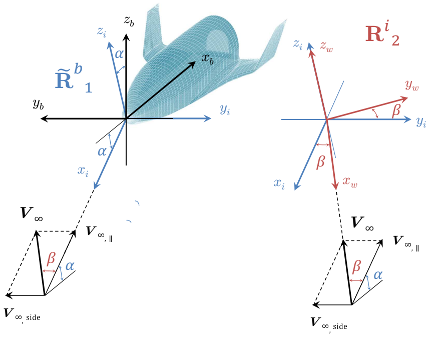


Figure 5.9: Transformations from the body frame b (in black) to the intermediate reference frame i (in blue), and from the intermediate reference frame i to the final wind frame w (in red).

Global forces and moments in the wind axes frame can hence be obtained by the combination of the two transformations:

$$\mathbf{F}^w = \mathbf{R}_2^i \tilde{\mathbf{R}}_1^b\, \mathbf{F}^b \tag{5.70}$$

$$\mathbf{M}^w = \mathbf{R}_2^i \tilde{\mathbf{R}}_1^b\, \mathbf{M}^b \tag{5.71}$$

5.7.4 Aerodynamic frame forces and coefficients definition

The relation of the wind forces with the classic definition of aerodynamic forces, i.e. lift, drag and side force, and the relative aerodynamic moments is defined by referring ourselves to the classic *aerodynamic axes* frame definition, which is strictly related to the wind axes by the relation:

$$\left\{ \begin{array}{c} x^a \\ y^a \\ z^a \end{array} \right\} = \left\{ \begin{array}{c} x^w \\ -y^w \\ -z^w \end{array} \right\}$$

Lift, Drag and Side forces can be then defined as:

$$\left\{ \begin{array}{c} D \\ S \\ L \end{array} \right\} = \left\{ \begin{array}{c} -F_x^a \\ -F_y^a \\ -F_z^a \end{array} \right\} = \left\{ \begin{array}{c} -F_x^w \\ F_y^w \\ F_z^w \end{array} \right\} \tag{5.72}$$

For the moments, body axes are used, since the pitching, roll and yaw moments are usually referred to the vehicle frame. We notice that this convention is not always respected in RAM v1.0 as presented in [1], hence we will replace the moments convention with ours, which respect the classical flight dynamic conventions. To be coherent with the most common definitions of pitch, roll and yaw, some signs are changed from our definition of body frame:

$$\left\{ \begin{array}{c} \mathcal{L} \\ \mathcal{M} \\ \mathcal{N} \end{array} \right\} = \left\{ \begin{array}{c} -M_x^b \\ M_y^b \\ -M_z^b \end{array} \right\} \tag{5.73}$$

We can now define the aerodynamic coefficients; for the forces we have:

$$C_L = \frac{L}{\frac{1}{2}\rho_\infty V_\infty^2 S_{ref}} \tag{5.74}$$

$$C_D = \frac{D}{\frac{1}{2}\rho_\infty V_\infty^2 S_{ref}} \tag{5.75}$$

$$C_S = \frac{S}{\frac{1}{2}\rho_\infty V_\infty^2 S_{ref}} \tag{5.76}$$

and for the moments:

$$C_{\mathcal{L}} = \frac{\mathcal{L}}{\frac{1}{2}\rho_\infty V_\infty^2 S_{ref}\, c_{ref}} \tag{5.77}$$

$$C_{\mathcal{M}} = \frac{\mathcal{M}}{\frac{1}{2}\rho_\infty V_\infty^2 S_{ref}\, c_{ref}} \tag{5.78}$$

$$C_{\mathcal{N}} = \frac{\mathcal{N}}{\frac{1}{2}\rho_\infty V_\infty^2 S_{ref}\, c_{ref}} \tag{5.79}$$

Even the aerodynamic coefficients can be charachterized by the separation between viscous and inviscid contribution, since the aerodynamic forces can be calculated from the separate global forces contributions, which can obtained in body axes from (5.53) and (5.54) for the forces and from (5.63) and (5.64). We can calculate the separate contributions in wind frame for the forces, obtaining:

$$\mathbf{F}^w_{\text{inv}} = \mathbf{R}^i_2 \tilde{\mathbf{R}}^b_1\, \mathbf{F}^b_{\text{inv}} \tag{5.80}$$

$$\mathbf{F}^w_{\text{visc}} = \mathbf{R}^i_2 \tilde{\mathbf{R}}^b_1\, \mathbf{F}^b_{\text{visc}} \tag{5.81}$$

This procedure is not adopted for the moments since they are defined directly from the body axes. We have then:

$$\left\{ \begin{array}{c} D \\ S \\ L \end{array} \right\}_{\text{inv}} = \left\{ \begin{array}{c} -F^w_x \\ F^w_y \\ F^w_z \end{array} \right\}_{\text{inv}} \tag{5.82}$$

$$\left\{ \begin{array}{c} D \\ S \\ L \end{array} \right\}_{\text{visc}} = \left\{ \begin{array}{c} -F^w_x \\ F^w_y \\ F^w_z \end{array} \right\}_{\text{visc}} \tag{5.83}$$

for the forces and

$$\left\{ \begin{array}{c} \mathcal{L} \\ \mathcal{M} \\ \mathcal{N} \end{array} \right\}_{\text{inv}} = \left\{ \begin{array}{c} -M^b_x \\ M^b_y \\ -M^b_z \end{array} \right\}_{\text{inv}} \tag{5.84}$$

$$\left\{ \begin{array}{c} \mathcal{L} \\ \mathcal{M} \\ \mathcal{N} \end{array} \right\}_{\text{visc}} = \left\{ \begin{array}{c} -M^b_x \\ M^b_y \\ -M^b_z \end{array} \right\}_{\text{visc}} \tag{5.85}$$

Hence, separate contribution of the aerodynamic coefficients can be calculated, in order to esteem the viscous contribution to the vehicle dynamics.

Chapter 6

Algorithm implementation

6.1 Introduction

While the last chapter was focused on the definition of the algorithm tasks and routines, explaining how the physical quantities were calculated, in this chapter we will afford those numerical tools we did not deepen previously for the sake of clarity. Some simpler tools, like the Newton - Rhapson iteration for non-linear systems, and all the geometrical definitions algorithms, have been already presented along with the global algorithm presentation, in sec. 5.2 or with the streamline solutor, in sec. 5.6.2. Here, we will affront the complexity of the Runge - Kutta method integration, with the embedded velocity interpolation and derivation over the quadrilateral. The tools for the topological eigenvectors alignment analysis used for the stagnation point identification, adopted in sec. 5.5, will be deduced from the velocity interpolation. A brief presentation of search trees hierarchical data structure for the nearest neighbour search will be also showed. In the end, we will face some considerations about the RAM v2.0 algorithm performances.

6.2 Runge Kutta Integration

To calculate the streamline field, the integration of the differential equations presented in sec. 5.6.1 is needed. To perform the numerical integration of eqn.s (5.32) a Runge Kutta 2nd order scheme will be used. Runge Kutta methods are one-step non-linear methods, which can be easily adapted to adactive step algorithms like our streamline solutor. Once the step h is set as defined in sec. 5.6.2. The currently adopted scheme is the 2nd order RK scheme known as the *trapezoidal rule*, which can be presented also as a multistep method. The increment scheme for the streamline coordinates is:

$$\mathbf{x}_{\text{stream},it+1} = \mathbf{x}_{\text{stream},it} + \frac{h}{2}\left(\mathbf{C}_{it} + \mathbf{C}_{it+1}\right) \tag{6.1}$$

where $\mathbf{C}$ is the vector containing the velocity director cosines in the three directions:

$$\mathbf{C} = \left\{ \begin{array}{c} C_x \\ C_y \\ C_z \end{array} \right\} = \left\{ \begin{array}{c} \frac{V_x}{V} \\ \frac{V_y}{V} \\ \frac{V_z}{V} \end{array} \right\} \tag{6.2}$$

$\mathbf{C}_{it}$ is obtained at the point $\mathbf{x}_{\text{stream},it}$ by evaluating the velocity vector there using the most appropriate velocity interpolation, as will be defined in subsection 6.4.6. Since we don't have a functional definition of the velocity field, which should be used to solve the non linear equation (6.1), we will adopt a different method to calculate $\mathbf{C}_{it+1}$. To define the velocity at the iteration *it+1* a first esteem of the streamline coordinate is obtained with an explicit Euler scheme

$$\mathbf{x}_{\text{stream},it\tilde{+}1} = \mathbf{x}_{\text{stream},it} + h\mathbf{C}_{it} \tag{6.3}$$

the cosines are then approximated in the $\mathbf{x}_{\text{stream},it\tilde{+}1}$, hence obtaining the cosines $\mathbf{C}_{it\tilde{+}1}$, from the velocity vector interpolation, which will be used in eqn. (6.1). Even if it is presented as a single step RK2 method for the sake of simplicity, the version of the trapezoidal rule presented here is actually a prediction-corrector method, the corrected final step being:

$$\mathbf{x}_{\text{stream},it+1} = \mathbf{x}_{\text{stream},it} + \frac{h}{2}\left(\mathbf{C}_{it} + \mathbf{C}_{it\tilde{+}1}\right) \tag{6.4}$$

6.3 Quadrilateral interpolation

6.4 Isoparametric representation

While performing the streamlines integration, we have to interpolate the velocity values in position which are not generally localized in the mesh centroids, since, as we have widely seen in the previous chapter. Hence, we will need to interpolate velocity on every streamline point from the surrounding vertex of the panel, as it has been defined in sec. 5.6.2. Since we are working over quadrilateral panels with generic shape, and variously oriented through the space, an intuitive idea is to use the isoparametric quadrilateral elements, widely used in the finite element theory to characterize panel or plate elements.

An exhaustive reference for isoparametric quadrilateral elements and for finite elements in general is [22]. In this work, however, quadrilateral elements are defined only for bi-dimensional quadrilaterals. We will extend the treatment to the three-dimensional case, and propose also an approximate solution, via the least square method of the inverse problem of obtaining the parametric coordinates once the interpolated point, i. e. the streamline point, is known. This problem has been already faced in [23], but its application is linked to the bi-dimensional case, and it reaches an excessive peak of complication, hence we will propose an alternate way. The solution of this problem occurs practically when we have a streamline point, or an intermediate Runge - Kutta calculation point, and we want to reconstruct the parametric representation of that point in order to interpolate the velocity there.

6.4.1 Coordinates interpolation

If we have a generic quadrilateral patch, for example the quadrilateral panel surrounding a streamline or an intermediate calculation point, we can express any of its internal coordinates by a parametric representation, via the parametric space (ξ, η). The patch is defined on the panel surrounding the old streamline point, in the Runge-Kutta integration process towards the new defined point, which has been defined by the vertices:

$$\mathbf{x}_{c,I,J},\ \mathbf{x}_{c,IP1,J}, \mathbf{x}_{c,IP1,JP1}, \mathbf{x}_{c,I,JP1}$$

To ease the notation of the following sections, a lighter notation is adopted, and the panel vertices become, mantaining the same ordering, respectively:

$$\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$$

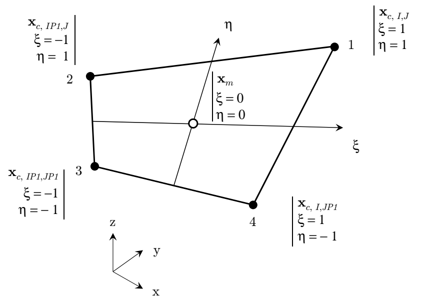


Figure 6.1: Quadrilateral patch in the three-dimensional space

To express the cartesian coordinates in the parametric space, we will use linear shape functions:

$$\left\{\begin{array}{l} N_1 = \frac{1}{4}\left(1+\xi\right)\left(1+\eta\right) \\ N_2 = \frac{1}{4}\left(1-\xi\right)\left(1+\eta\right) \\ N_3 = \frac{1}{4}\left(1-\xi\right)\left(1-\eta\right) \\ N_4 = \frac{1}{4}\left(1+\xi\right)\left(1-\eta\right) \end{array}\right. \tag{6.5}$$

In three dimension, the internal coordinates of the quadrilateral can be expressed, referring to the parametric coordinates and to corner values, as

$$\left\{\begin{array}{c} x\left(\xi,\eta\right) \\ y\left(\xi,\eta\right) \\ z\left(\xi,\eta\right) \end{array}\right\} = \left[\begin{array}{cccc} N_1\mathbf{I} & N_2\mathbf{I} & N_3\mathbf{I} & N_4\mathbf{I} \end{array}\right] \left\{\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \end{array}\right\} \tag{6.6}$$

where

$$\mathbf{I} = \left[\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{array}\right] \tag{6.7}$$

A three-dimensional generic variable, defined on the corner points can be also expressed as:

$$\left\{\begin{array}{c} u\left(\xi,\eta\right) \\ v\left(\xi,\eta\right) \\ w\left(\xi,\eta\right) \end{array}\right\} = \left[\begin{array}{cccc} N_1\left(\xi,\eta\right)\mathbf{I} & N_2\left(\xi,\eta\right)\mathbf{I} & N_3\left(\xi,\eta\right)\mathbf{I} & N_4\left(\xi,\eta\right)\mathbf{I} \end{array}\right] \left\{\begin{array}{c} u_1 \\ u_2 \\ u_3 \\ u_4 \\ v_1 \\ v_2 \\ v_3 \\ v_4 \\ w_1 \\ w_2 \\ w_3 \\ w_4 \end{array}\right\} \tag{6.8}$$

6.4.2 Inverse problem

Definition The solution of the inverse problem will be used to perform with increased accuracy the Runge - Kutta integration step in the routine 5.6.2. We solve here the problem of obtaining ξ, η given an arbitrary point $\{x_i\, y_i\, z_i\}$, which will be in our case the streamline coordinate $\mathbf{x}_{\text{stream},\, it}$ or an intermediate step calculation point, as defined in the previous section. In this point we want to interpolate other quantities defined in the quadrilateral vertex, e.g. in our algorithm the velocity vectors:

$$\left\{ \begin{array}{c} x\,(\xi,\eta) \\ y\,(\xi,\eta) \\ z\,(\xi,\eta) \end{array} \right\} = \left[\begin{array}{cccc} N_1\mathbf{I} & N_2\mathbf{I} & N_3\mathbf{I} & N_4\mathbf{I} \end{array}\right] \left\{ \begin{array}{c} x_1 \\ x_2 \\ x_3 \\ x_4 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ z_1 \\ z_2 \\ z_3 \\ z_4 \end{array} \right\} \tag{6.9}$$

This expression can be rewritten as:

$$\begin{cases} -4x_i\,(\xi,\eta) + (x_1 + x_2 + x_3 + x_4) + (x_1 - x_2 - x_3 + x_4)\,\xi + \\ -4y_i\,(\xi,\eta) + (y_1 + y_2 + y_3 + y_4) + (y_1 - y_2 - y_3 + y_4)\,\xi + \\ -4z_i\,(\xi,\eta) + (z_1 + z_2 + z_3 + z_4) + (z_1 - z_2 - z_3 + z_4)\,\xi + \end{cases}$$

$$\begin{array}{c} +\,(x_1 + x_2 - x_3 - x_4)\,\eta + (x_1 - x_2 + x_3 - x_4)\,\xi\eta = 0 \\ +\,(y_1 + y_2 - y_3 - y_4)\,\eta + (y_1 - y_2 + y_3 - y_4)\,\xi\eta = 0 \\ +\,(z_1 + z_2 - z_3 - z_4)\,\eta + (z_1 - z_2 + z_3 - z_4)\,\xi\eta = 0 \end{array} \tag{6.10}$$

or

$$\begin{cases} f_1\,(\xi,\eta) = 0 \\ f_2\,(\xi,\eta) = 0 \\ f_3\,(\xi,\eta) = 0 \end{cases} \tag{6.11}$$

which is a non-linear system in the ξ, η variables.

Newton Rhapson system solution We solve the non-linear ξ, η system using a Newton - Rhapson algorithm

$$\begin{cases} f_1(\xi,\eta) = f_1(\xi_0,\eta_0) + \left.\frac{\partial f_1}{\partial \xi}\right|_0 \Delta\xi + \left.\frac{\partial f_1}{\partial \eta}\right|_0 \Delta\eta = 0 \\ f_2(\xi,\eta) = f_2(\xi_0,\eta_0) + \left.\frac{\partial f_2}{\partial \xi}\right|_0 \Delta\xi + \left.\frac{\partial f_2}{\partial \eta}\right|_0 \Delta\eta = 0 \\ f_3(\xi,\eta) = f_3(\xi_0,\eta_0) + \left.\frac{\partial f_3}{\partial \xi}\right|_0 \Delta\xi + \left.\frac{\partial f_3}{\partial \eta}\right|_0 \Delta\eta = 0 \end{cases} \tag{6.12}$$

which can be expressed as a linear system:

$$\mathbf{A}\left\{\begin{array}{c}\Delta\xi\\ \Delta\eta\end{array}\right\} = -\left\{\begin{array}{c} f_1(\xi_0,\eta_0)\\ f_2(\xi_0,\eta_0)\\ f_3(\xi_0,\eta_0)\end{array}\right\} \tag{6.13}$$

The increment can be calculated by pseudo inversion in the least squares sense of the matrix:

$$\left\{\begin{array}{c}\Delta\xi\\ \Delta\eta\end{array}\right\} = -\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\left\{\begin{array}{c} f_1(0,0)\\ f_2(0,0)\\ f_3(0,0)\end{array}\right\} \tag{6.14}$$

the increment is then calculated an the step is increased:

$$\xi = \xi_0 + \Delta\xi \tag{6.15}$$

$$\eta = \eta_0 + \Delta\eta \tag{6.16}$$

Evidences have shown that error in the ξ, η determination goes under $1e^{-11}$ after two Newton iterations, for the typical panels defined in our LaWGS geometries. Once ξ, η are known, the parametric identification of a generic point inside the quadrilateral is complete. We will use this inverse procedure in the Runge-Kutta iteration, in order to find the parametric position of a streamline point, or of a Runge-Kutta intermediate calculation point, inside the surrounding panel. When the parametric position of this point is known, parametric interpolation from the known velocity values V_x, V_y and V_z on the panel vertices can be performed, for example using eqn. (6.8). Hence for the next Runge - Kutta initial or intermediate step we have defined the accurate velocity value in the starting point.

6.4.3 Scalar field derivation

We present here the basis of scalar quantity derivation in the parametric space, which will be used to obtain velocity derivatives which will be used in the tangential derivative definition, used in the Simeonides algorithm in defining

u_{grad} for the solution of equation (3.29), and also to define the velocity field eigenvalues problem in order to obtain the alignment matrix al_{ij} presented in sec. 5.5.

Two dimensions If we want to find the derivative of a quantity $u = u\left(\xi, \eta\right) = u\left(x, y\right)$, we can express its derivatives in the parametric space by chain rule derivation:

$$\left\{ \begin{array}{l} \frac{\partial u}{\partial \xi} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \xi} \\ \frac{\partial u}{\partial \eta} = \frac{\partial u}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial u}{\partial y}\frac{\partial y}{\partial \eta} \end{array} \right. \tag{6.17}$$

which can be expressed in matricial form as:

$$\mathbf{J} = \left[\begin{array}{cc} x_{/\xi} & y_{/\xi} \\ x_{/\eta} & y_{/\eta} \end{array} \right] \tag{6.18}$$

$$\left\{ \begin{array}{c} u_{/\xi} \\ u_{/\eta} \end{array} \right\} = \left[\begin{array}{cc} x_{/\xi} & y_{/\xi} \\ x_{/\eta} & y_{/\eta} \end{array} \right] \left\{ \begin{array}{c} u_{/x} \\ u_{/y} \end{array} \right\} = \mathbf{J} \left\{ \begin{array}{c} u_{/x} \\ u_{/y} \end{array} \right\} \tag{6.19}$$

Looking at the single element of the jacobian matrix, the derivative with respect to the parameter can be expressed in terms of derivatives of the shape function:

$$\begin{cases} \frac{\partial x}{\partial \xi} = \sum_1^4 N_{i\,/\xi} x_i \\ \frac{\partial y}{\partial \xi} = \sum_1^4 N_{i\,/\xi} y_i \\ \frac{\partial x}{\partial \eta} = \sum_1^4 N_{i\,/\eta} x_i \\ \frac{\partial y}{\partial \eta} = \sum_1^4 N_{i\,/\eta} y_i \end{cases} \tag{6.20}$$

so that the Jacobian can be expressed as

$$\mathbf{J} = \left[\begin{array}{cc} \sum_1^4 N_{i\,/\xi} x_i & \sum_1^4 N_{i\,/\xi} y_i \\ \sum_1^4 N_{i\,/\eta} x_i & \sum_1^4 N_{i\,/\eta} y_i \end{array} \right] \tag{6.21}$$

We have then

$$\mathbf{\Gamma} = \mathbf{J}^{-1} \tag{6.22}$$

so that the derivative of ϕ on the quadrilateral can be expressed as

$$\left\{ \begin{array}{c} u_{/x} \\ u_{/y} \end{array} \right\} = \mathbf{\Gamma} \left\{ \begin{array}{c} u_{/\xi} \\ u_{/\eta} \end{array} \right\} \tag{6.23}$$

Three dimensions with least squares The Jacobian for a three dimensional quantity $u = u\left(x, y, z\right) = u\left(\xi, \eta\right)$ can be expressed as:

$$\mathbf{J} = \left[\begin{array}{ccc} x_{/\xi} & y_{/\xi} & z_{/\xi} \\ x_{/\eta} & y_{/\eta} & z_{/\eta} \end{array} \right] \tag{6.24}$$

$$\mathbf{J} = \frac{1}{4} \left[\begin{array}{cccc} (1+\eta) & -(1+\eta) & -(1-\eta) & (1-\eta) \\ (1+\xi) & (1-\xi) & -(1-\xi) & -(1+\xi) \end{array} \right] \left[\begin{array}{ccc} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \\ x_4 & y_4 & z_4 \end{array} \right] \tag{6.25}$$

We have then

$$\left\{ \begin{array}{c} u_{/\xi} \\ u_{/\eta} \end{array} \right\} = \mathbf{J} \left\{ \begin{array}{c} u_{/x} \\ u_{/y} \\ u_{/z} \end{array} \right\} \tag{6.26}$$

Since $\mathbf{J}$ is not a square matrix anymore, the inversion of this relation requires a least square pseudo-inverse:

$$\left\{ \begin{array}{c} u_{/x} \\ u_{/y} \\ u_{/z} \end{array} \right\} = \left[\left(\mathbf{J}^T \mathbf{J}\right)^{-1} \mathbf{J}^T \right] \left\{ \begin{array}{c} u_{/\xi} \\ u_{/\eta} \end{array} \right\} = \tilde{\mathbf{\Gamma}} \left\{ \begin{array}{c} u_{/\xi} \\ u_{/\eta} \end{array} \right\} \tag{6.27}$$

6.4.4 Directional derivative on a quadrilateral patch

For the use of Meador Smart reference enthalpy, a definition of the tangential derivative of the velocity is required. We will use isoparametric formulation to obtain the tangential derivative projecting our velocity field on the parametric element. Notice that the isoparametric element is now chosen as the panel identified by the *IM1, IP1, JM1, JP1* centroids surrounding the middle *I,J* centroid. In this way a derivative for every centroid $\mathbf{x}_{c,IJ}$ can be defined. The vertices of the panel on which we define the velocity derivative are:

$$\mathbf{x}_{c,IM1,JM1},\ \mathbf{x}_{c,IM1,JP1}, \mathbf{x}_{c,IP1,JP1}, \mathbf{x}_{c,IP1,JM1}$$

which, paying attention to not confuse them with the vertices definition of the Runge Kutta inverse problem presented in the previous subsection, are re-indexed as

$$\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3, \mathbf{x}_4$$

for the sake of simplicity.

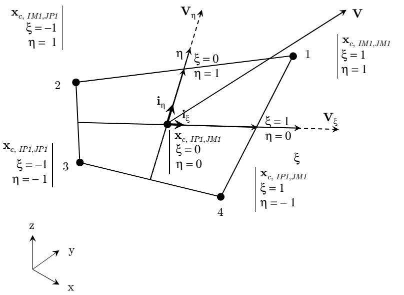

$\mathbf{x}_{c,\ IM1,JP1}$ $\xi=-1$ $\eta=1$
$\mathbf{V}_\eta$
$\mathbf{V}$
η
$\xi=0$ $\eta=1$
1
$\mathbf{x}_{c,\ IM1,JM1}$ $\xi=1$ $\eta=1$
2
$\mathbf{i}_\eta$
$\mathbf{i}_\xi$
$\xi=1$ $\eta=0$
$\mathbf{V}_\xi$
$\mathbf{x}_{c,\ IP1,JM1}$ $\xi=0$ $\eta=0$
$\mathbf{x}_{c,\ IP1,JP1}$ $\xi=-1$ $\eta=-1$
3
ξ
$\mathbf{x}_{c,\ IP1,JM1}$ $\xi=1$ $\eta=-1$
4
z
y
x

Figure 6.2: Quadrilateral patch with base and velocities definitions

define the tangent vector on the centroid of the quadrilateral as

$$\mathbf{t} = \frac{1}{V}\left\{ \begin{array}{ccc} V_x & V_y & V_z \end{array} \right\}^T \tag{6.28}$$

The directional derivative of a function $f = f(x, y, z) = f(\mathbf{x})$along $\mathbf{t}$ can be then expressed as:

$$\frac{\partial f}{\partial \mathbf{t}} = \frac{\partial f}{\partial \mathbf{x}} \cdot \mathbf{t} \tag{6.29}$$

Over a quadrilateral, it is easier to work in parametric coordinates:

$$\mathbf{t} = \frac{1}{V}\left\{ \begin{array}{cc} V_\xi & V_\eta \end{array} \right\}^T \tag{6.30}$$

so that

$$\frac{\partial f}{\partial \mathbf{t}} = \frac{\partial f}{\partial \xi} \cdot \mathbf{t} \tag{6.31}$$

where $\xi = \left\{ \begin{array}{cc} \xi & \eta \end{array} \right\}^T$

If we consider the velocity vector as the function to be derived, we have then

$$\frac{\partial V}{\partial t} = \frac{\partial V}{\partial \xi} = \frac{1}{V}\left\{ \begin{array}{cc} \frac{\partial V}{\partial \xi} & \frac{\partial V}{\partial \eta} \end{array} \right\}\left\{ \begin{array}{c} V_\xi \\ V_\eta \end{array} \right\} \tag{6.32}$$

We define then V_ξ and V_η as the projections of $\mathbf{V} = \left\{ \begin{array}{ccc} V_x & V_y & V_z \end{array} \right\}^T$ over the quadrilateral patch. If we define a versor base of the quadrilateral

$$\hat{\mathbf{i}}_\xi = \frac{\mathbf{x}_{(\xi=1,\eta=0)} - \mathbf{x}_{(\xi=-1,\eta=0)}}{\left\|\mathbf{x}_{(\xi=1,\eta=0)} - \mathbf{x}_{(\xi=-1,\eta=0)}\right\|} \tag{6.33}$$

$$\hat{\mathbf{i}}_\eta = \frac{\mathbf{x}_{(\xi=0,\eta=1)} - \mathbf{x}_{(\xi=0,\eta=-1)}}{\left\|\mathbf{x}_{(\xi=0,\eta=1)} - \mathbf{x}_{(\xi=0,\eta=-1)}\right\|} \tag{6.34}$$

we can define

$$V_\xi = \mathbf{V} \cdot \hat{\mathbf{i}}_\xi \tag{6.35}$$

$$V_\eta = \mathbf{V} \cdot \hat{\mathbf{i}}_\eta \tag{6.36}$$

and hence $\mathbf{t} = \frac{1}{V} \left\{ \begin{array}{cc} V_\xi & V_\eta \end{array} \right\}^T$.

Velocity derivatives in the panel centroid $\mathbf{x}_{c,IJ}$, i.e. $\left\{ \begin{array}{cc} \frac{\partial V}{\partial \xi} & \frac{\partial V}{\partial \eta} \end{array} \right\}_{IJ}$,can be found by applying the finite difference method directly along the ξ, η directions:

$$\left. \frac{\partial V}{\partial \xi} \right|_{IJ} = \frac{V_{(\xi=1,\eta=0)} - V_{(\xi=-1,\eta=0)}}{h_\xi} \tag{6.37}$$

and

$$\left. \frac{\partial V}{\partial \eta} \right|_{IJ} = \frac{V_{(\xi=0,\eta=1)} - V_{(\xi=0,\eta=-1)}}{h_\eta} \tag{6.38}$$

if we define the derivation steps in the ξ, η directions, which correspond to the isoparametric elements dimensions along these directions, as:

$$h_\xi = \left\| \mathbf{x}_{(\xi=1,\eta=0)} - \mathbf{x}_{(\xi=-1,\eta=0)} \right\| \tag{6.39}$$

$$h_\eta = \left\| \mathbf{x}_{(\xi=0,\eta=1)} - \mathbf{x}_{(\xi=0,\eta=-1)} \right\| \tag{6.40}$$

The velocities moduli $V_{(\xi=1,\eta=0)}$, $V_{(\xi=-1,\eta=0)}$, $V_{(\xi=0,\eta=1)}$ and $V_{(\xi=0,\eta=-1)}$ can be defined, knowing velocities in the panel vertices, using the three dimensional variable interpolation defined in eqn. (6.8), substituing the appropriate ξ, η values.

We have now defined all the terms which are required to define the tangential derivative in every centroid $\mathbf{x}_{c,IJ}$, which is, recalling eqn.(6.32):

$$\left. \frac{\partial V}{\partial t} \right|_{IJ} = \frac{1}{V} \left\{ \begin{array}{cc} \frac{\partial V}{\partial \xi} & \frac{\partial V}{\partial \eta} \end{array} \right\}_{IJ} \left\{ \begin{array}{c} V_\xi \\ V_\eta \end{array} \right\} \tag{6.41}$$

6.4.5 Topological study on local vector field

We report here the determination of the eigenvalues from the parametric velocity representation, to be used in the alignment matrix al_{ij} determination for the analysis of the stagnation point, as defined in 5.5. The main concept is to extract the eigenvectors of the local surface jacobian $\nabla\mathbf{V}$ of the surface velocity. Given the eigenvectors $\mathbf{e}_i$, where $i = 1 : 2$ are two independent directions of derivation along the panel surface, arranged in order that $\mathbf{e_1}$ is the largest modulus eigenvector, recalling eqn.s (5.26) and (5.27), we have attachment when

$$\mathbf{e_2} \times \mathbf{V} = 0 \tag{6.42}$$

and separation when

$$\mathbf{e_1} \times \mathbf{V} = 0 \tag{6.43}$$

the local Jacobian, in parametric coordinates, can be expressed as:

$$\nabla\mathbf{V} = \mathbf{J_v} = \begin{bmatrix} V_{\xi/\xi} & V_{\xi/\eta} \\ V_{\eta/\xi} & V_{\eta/\eta} \end{bmatrix} \tag{6.44}$$

To obtain the jacobian elements, we must derive V_ξ and V_η along the ξ, η directions. We will adopt then finite differences over the isoparametric element, due to the noticeable simplicity of this formulation.

Hence, the finite differences in isoparametric coordinates are easily obtained as:

$$V_{\xi/\xi} = \frac{V_{\xi(\xi=1,\eta=0)} - V_{\xi(\xi=-1,\eta=0)}}{h_\xi} \tag{6.45}$$

$$V_{\xi/\eta} = \frac{V_{\xi(\xi=0,\eta=1)} - V_{\xi(\xi=0,\eta=-1)}}{h_\eta} \tag{6.46}$$

$$V_{\eta/\xi} = \frac{V_{\eta(\xi=1,\eta=0)} - V_{\eta(\xi=-1,\eta=0)}}{h_\xi} \tag{6.47}$$

$$V_{\eta/\eta} = \frac{V_{\eta(\xi=0,\eta=1)} - V_{\eta(\xi=0,\eta=-1)}}{h_\eta} \tag{6.48}$$

The isoparametric velocity components which are necessary to the finite derivation can be found by projecting the velocity vector in the appropriate positions of the isoparametric quadrilateral. For example $V_{\xi(\xi=1,\eta=0)}$ can be obtained from:

$$V_{\xi(\xi=1,\eta=0)} = \mathbf{V}_{(\xi=1,\eta=0)} \cdot \hat{\mathbf{i}}_\xi \tag{6.49}$$

where $\mathbf{V}_{(\xi=1,\eta=0)}$ can be obtained by the three-dimensional variable interpolation formula (6.8), with $\xi = 1$ and $\eta = 0$.

Following the same approach, we can find as well:

$$V_{\xi(\xi=-1,\eta=0)} = \mathbf{V}_{(\xi=-1,\eta=0)} \cdot \hat{\mathbf{i}}_\xi \tag{6.50}$$

$$V_{\xi(\xi=0,\eta=1)} = \mathbf{V}_{(\xi=0,\eta=1)} \cdot \hat{\mathbf{i}}_\xi \tag{6.51}$$

$$V_{\xi(\xi=0,\eta=-1)} = \mathbf{V}_{(\xi=0,\eta=-1)} \cdot \hat{\mathbf{i}}_{\xi} \tag{6.52}$$

for the velocity component along ξ, and

$$V_{\eta(\xi=1,\eta=0)} = \mathbf{V}_{(\xi=1,\eta=0)} \cdot \hat{\mathbf{i}}_{\eta} \tag{6.53}$$

$$V_{\eta(\xi=-1,\eta=0)} = \mathbf{V}_{(\xi=-1,\eta=0)} \cdot \hat{\mathbf{i}}_{\eta} \tag{6.54}$$

$$V_{\eta(\xi=0,\eta=1)} = \mathbf{V}_{(\xi=0,\eta=1)} \cdot \hat{\mathbf{i}}_{\eta} \tag{6.55}$$

$$V_{\eta(\xi=0,\eta=-1)} = \mathbf{V}_{(\xi=0,\eta=-1)} \cdot \hat{\mathbf{i}}_{\eta} \tag{6.56}$$

We can then calculate the finite difference derivatives (6.45), (6.46), (6.47) and (6.48), and find the local velocity field jacobian $\mathbf{J_v}$. The jacobian eigenvalues we need for the topological study can then be obtained by solving the eigenvalues problem

$$det\left(\mathbf{J_v} - \lambda\mathbf{I}\right) = 0 \tag{6.57}$$

obtaining the two eigenvalues λ_1 and λ_2.
The eigenvectors $\mathbf{e_1}$ and $\mathbf{e_2}$ are found from the solution of the system

$$\mathbf{J_v}\mathbf{e_i} = \lambda_i\mathbf{e_i} \tag{6.58}$$

paying attention to arrange $\mathbf{e_i}$ so that $\mathbf{e_1}$ is the largest modulus eigenvector. Our attachment and detachment alignment factors can be then calculated:

$$\mathbf{e_2} \times \mathbf{V} = e_{2,1}V_{\eta} - e_{2,2}V_{\xi} \tag{6.59}$$

$$\mathbf{e_1} \times \mathbf{V} = e_{1,1}V_{\eta} - e_{1,2}V_{\xi} \tag{6.60}$$

and the raw alignment map can be defined as, recalling eqn. (5.28):

$$al_{ij} = \max\left(\left|\mathbf{e_1} \times \mathbf{V}_{ij,\text{body}}\right|, \left|\mathbf{e_2} \times \mathbf{V}_{ij,\text{body}}\right|\right) \tag{6.61}$$

Then, as been explained in sec. 5.5, filtering operations are performed and stagnation nearest centroid $\mathbf{x}_{c,\text{stag}}$ is found.

6.4.6 Effects on the Runge-Kutta integrator

We present here the confront of the streamline field determination accuracy improvement by the use of quadrilateral interpolation against a non interpolated RK streamline field. A simplified model of RK integrator is used, by defining a mean velocity component for every streamline or intermediate step point on the panel, which consists simply of the mean of the corner velocites components:

$$V_{i,\text{panel}} = \frac{V_{i1} + V_{i2} + V_{i3} + V_{i4}}{4} \qquad i = x, y, z \tag{6.62}$$

Results are presented over the Dart reference geometry at incidence, and cleary shows an improvement of streamlines resolution over the bottom surface. In fact, the asymptotic streamline on the windward centerline is not matched well by the non interpolated streamlines field, while the RK integration with quadrilateral interpolation feature catches better this particular flow pattern, as can be clearly seen in the following figure:

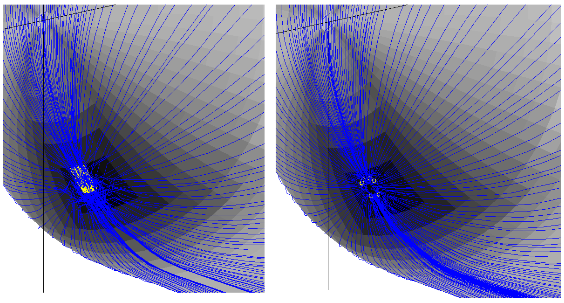


Figure 6.3: RK streamlines calculation - without interpolation (top) and with quadrilateral interpolation (bottom)

Streamline reconstruction with quadrilateral interpolation produces hence an accuracy improvement over the RAM v2.0 algorithm. However, as we can see, streamlines patterns far away of stagnation points and asymptotic lines do not differ sensibly, justyfing only in part the adoption of such grade of interpolation, which require the solving of a non linear system, with the pseudo - inversion of the coefficients matrix, as in system (6.14). This procedure is to be repeated for every iteration step or intermediate streamline point, hence twice the times the streamline points used in the whole algorithm. Such computational effort is not always justifiable, hence it will be left as an user option in the RAM v2.0 input file, default setting will be the non interpolated constant mean velocity value on the panel, as identified by eqn.s (6.62).

Part III

Numerical Results and Outlook

Chapter 7

Results and comparisons

7.1 Comparison on IXV geometry

7.1.1 General

We will present here the comparison between our algorithm and a CFD simulation over the IXV geometry at incidence, see fig. 1.2 to have an idea of the spacecraft layout. The IXV vehicle is an unmanned testbed, developed by ESA to provide a platform for the experimental verification of re-entry materials and technologies, and to provide an autonomous european aerodynamic controlled re-entry capability. The IXV first flight is expected in 2013, using the new Vega launch system, which has been succesfully tested in February 2012. We will confront here the patterns of several surface quantities, e.g. friction factor, shear stress, heat flow and temperature distribution, against the calculations which were committed by the European Space Agency to verify the aerodynamic characteristics of the IXV. The simulations were performed by a numerical Navier Stokes hypersonic and thermochemically active code, with laminar and turbulent regime description capability, developed by Thorsten Scholz for the Von Karman Institute of Fluid Dynamics, in collaboration with professor Patrick Rambaud, and it is described in [24]. Out from the several simulations, we confronted the results of our algorithm with three cases, obtained in condition of symmetrical flight, high incidence AOA (45°) and at three different Mach numbers. Turbulent modeling is activated to esteem its validity with VKI CFD simulations.

With respect to the algorithm development case - i.e. the DART reference case presented in sec. 1.7 - we have here a complex geometry, hence to obtain the best correlation possible between our algorithm and the actual physical phenomena, we described the nose geometry with a variable curvature radius, in order to obtain a more accurate representation of the stagnation region characteristic length which will be used in Simeonides shear stress formula

(3.29). A center for the curvature radius is proposed, and then a radius for every centroid in the nose region is calculated.

Another geometric complexity of the IXV is the presence of two control flaps in the rear of the vehicle. At the current status of the algorithm implementation, non-fusiform parts are treated as wings, hence the running length are calculated from the stagnation point. In the IXV case, this representation is not anymore correct, at least for a comparsion with the CFD model, since in this case the two flaps are part of the fuselage body, hence the boundary layer growth length should be calculated from the vehicle nose, and not from the flap part stagnation point, as our streamline solver does, since it is optimized for wing analysis. In the future developments we will expect to correct this feature, but for now we accept poor results on the flap region, since our main aim is to verify the soundness of the model. Another difference from our model is that in VKI CFD simulation, they are deflected downward with an angle $\delta = 10°$, while our *.wgs file they are not still deflected, although it could be done by using the rotating part definition defined in LaWGS format, as defined in sec. 1.6.

7.1.2 M6 CFD simulation comparison

Freestream values We will provide here the freestream condition for this simulation reference:

Table 7.1: Freestream and after shock values for VKI CFD $M_\infty = 6$ simulation

freestream	*value*	*shocked*	*value*
α	45°	-	-
β	0°	-	-
V_∞	1824.56 $\left[\frac{m}{s}\right]$	V_s	307.9993 $\left[\frac{m}{s}\right]$
M_∞	6.019	M_s	0.3912
Pr_∞	0.75	Pr_s	0.7547
P_∞	838.234 $[Pa]$	P_s	36173 $[Pa]$
ρ_∞	0.012758804 $\left[\frac{kg}{m^3}\right]$	ρ_s	0.0756 $\left[\frac{kg}{m^3}\right]$
T_∞	228.9943 $[K]$	T_s	1667.1 $[K]$

Surface values comparison We report here the comparison of the surface distribution of c_p, c_f, q_w and τ_w between the von Karman CFD high temperature effects code simulation and our algorithm solution. The pressure coefficient will esteem the correctness of the RAM v1.0 inviscid model, the other quantities the quality of the viscous solver of RAM v2.0.

We report the freestream values table for this simulation:

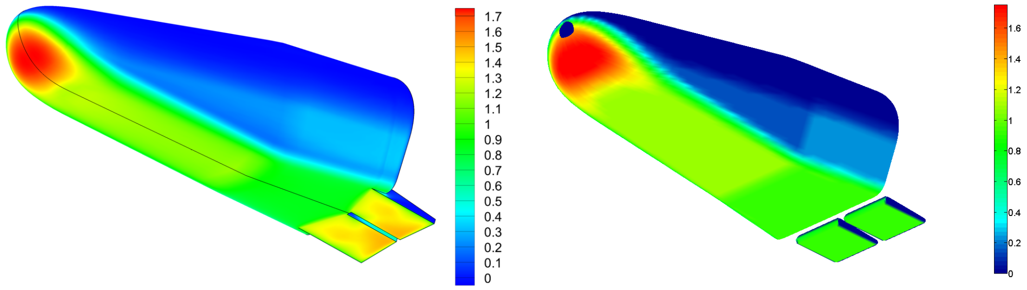


Figure 7.1: c_p distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 6.019$, $\alpha = 45°$, $\beta = 0°$

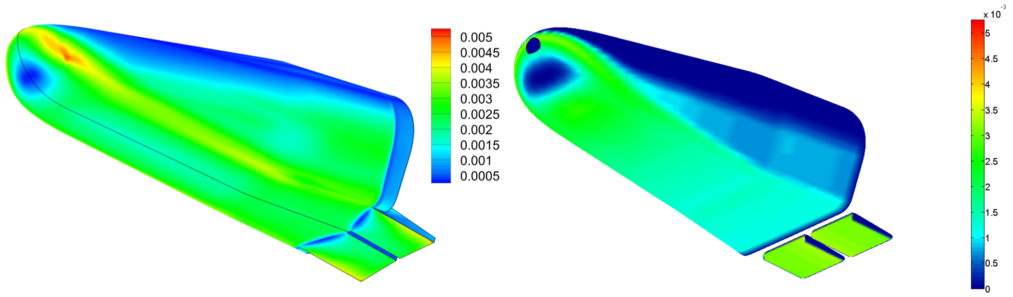


Figure 7.2: c_f distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 6.019$, $\alpha = 45°$, $\beta = 0°$

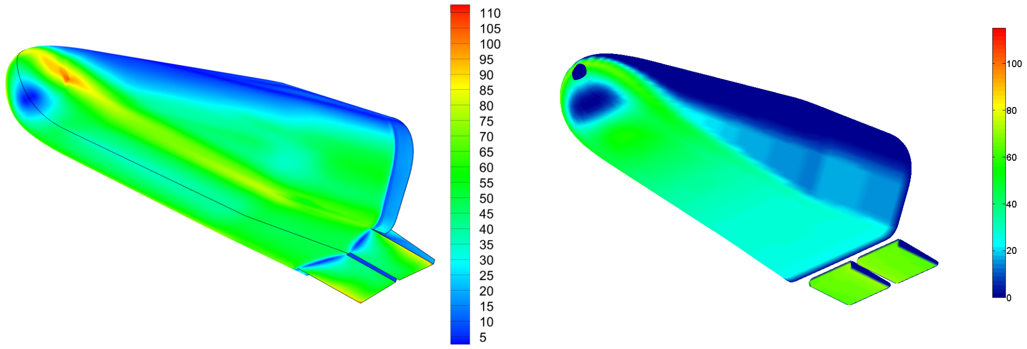


Figure 7.3: τ_w $[Pa]$ distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 6.019$, $\alpha = 45°$, $\beta = 0°$

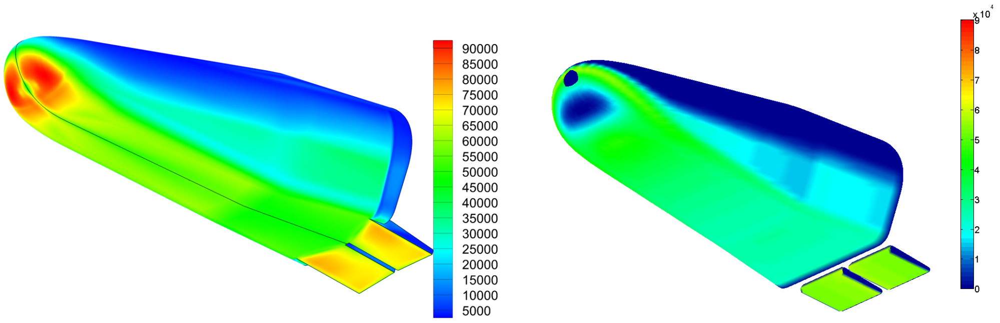


Figure 7.4: q_w $[\frac{W}{m^2}]$ distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 6.019$, $\alpha = 45°$, $\beta = 0°$

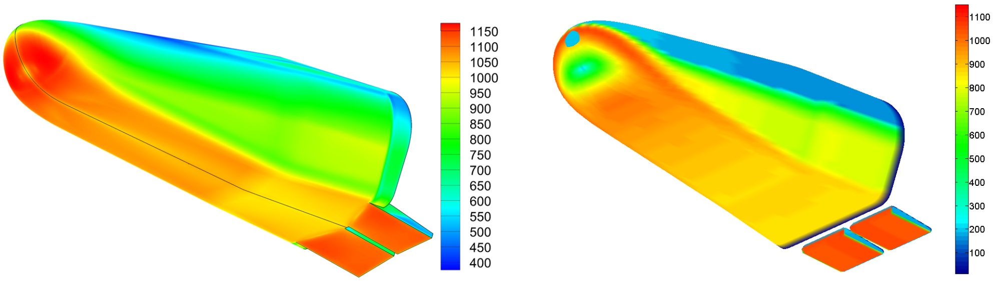


Figure 7.5: T_w $[K]$ distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 6.019$, $\alpha = 45°$, $\beta = 0°$

As one can see, very good agreement is obtained for all the patterns, except for the heat flow and wall temperature in the nose region, where we have an underprediction of actual values. The reason for this behaviour lies in the RAM v2.0 determination of heat flow through Reynolds analogy. As we have seen while studying Reynolds analogy on the reference case, this method of obtaining heat flow provides very convincing values away from the stagnation point, but it limits the heat flow q_w to follow the friction coefficient c_f curve trend, as it was obtained from the comparison with DART reference case, see fig. 3.3. We remember that flap values can be different from the actual calculated values, since our flap are not deflected and the boundary layer running length should be increased by the fuselage aft running length values.

7.1.3 M11 CFD simulation comparison

Table 7.2: Freestream values for VKI CFD $M_\infty = 11$ simulation

freestream	*value*	*shocked*	*value*
α	45°	-	-
β	0°	-	-
V_∞	3624.26 $\left[\frac{m}{s}\right]$	V_s	353.6559 $\left[\frac{m}{s}\right]$
M_∞	11.035	M_s	0.3023
Pr_∞	0.75	Pr_s	0.6802
P_∞	214.658 $[Pa]$	P_s	33045 $[Pa]$
ρ_∞	0.002768343 $\left[\frac{kg}{m^3}\right]$	ρ_s	0.0284 $\left[\frac{kg}{m^3}\right]$
T_∞	270.2692 $[K]$	T_s	3583.0 $[K]$

Freestream values

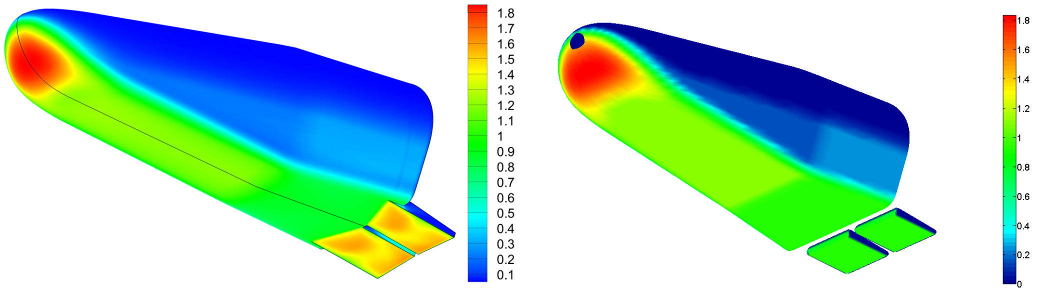


Figure 7.6: c_p distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 11.035$, $\alpha = 45°$, $\beta = 0°$

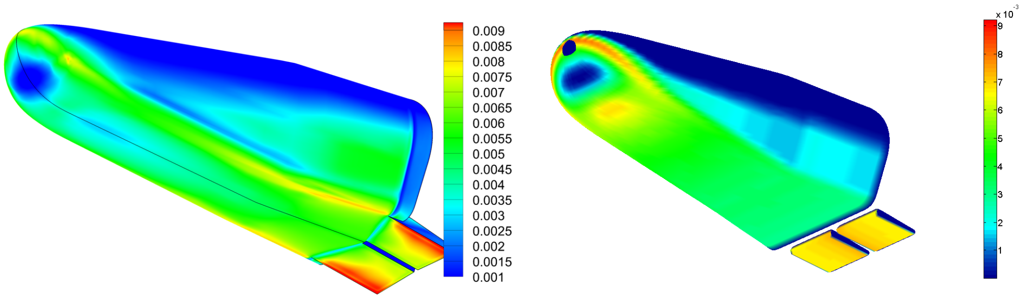


Figure 7.7: c_f distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 11.035$, $\alpha = 45°$, $\beta = 0°$

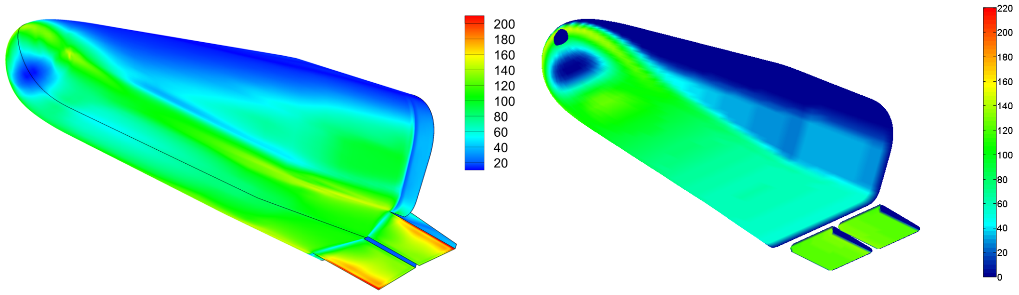


Figure 7.8: τ_w $[Pa]$ distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 11.035$, $\alpha = 45°$, $\beta = 0°$

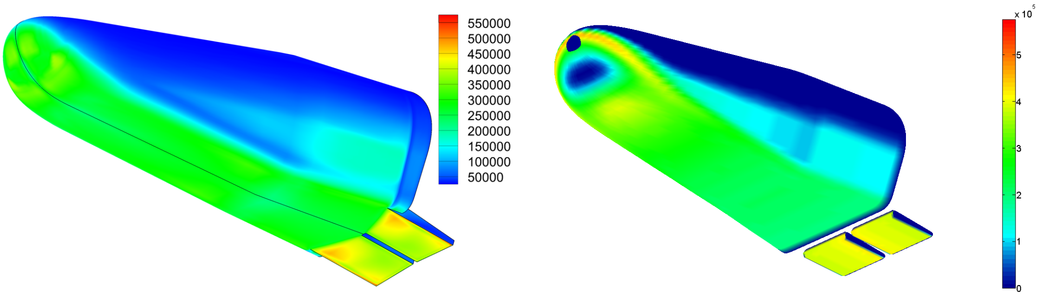


Figure 7.9: q_w $[\frac{W}{m^2}]$ distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 11.035$, $\alpha = 45°$, $\beta = 0°$

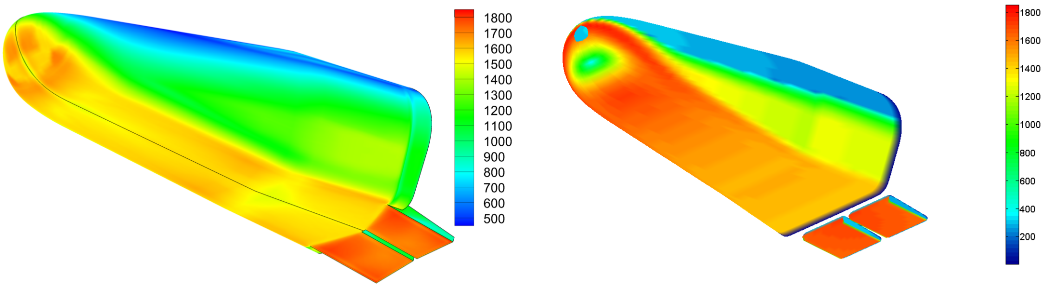


Figure 7.10: T_w $[K]$ distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution (right) - IXV geometry - $M_\infty = 11.035$, $\alpha = 45°$, $\beta = 0°$

Surface values comparison As we can see, the flowfield reconstruction is even better than the $M_\infty = 6.019$ case. The considerations which were exposed for the previous simulation are still applicable here, except for the discussion about the turbulent flow, which in these conditions is seen by our algorithm as absent, since the transition condition defined by eqn. (3.40) is not reached for this flow condition. This difference is visible in the aft region of the windward surface, i.e. the bottom rear zone and the lateral rear zone, where CFD simulation show clearly an increment of friction factor and shear stress, which was observed also in our algorithm for the $M = 6.019$ simulation, and is now missing.

7.2 Comparison on Expert geometry

As a further verification of the validity of the aerodynamic coefficients calculation, we will confront the data generated from RAM v2.0 with a database of coefficients which has been used for the design of the Expert mission. Expert is another aerothermodynamics testbed capsule developed by ESA, principally intended as an in-flight verification tool for ESA's design software and CFD validation. It is composed of a pyramidal conic shape with four fixed flaps, its geometry and mass being suitable to be launched from a Russian submarine Volna ballistic missile. The Expert shape is also conceived to test the idea of hot-structure, non ablative thermal shielding. Figure 7.11 shows the Expert geometry along with the streamlines pattern as calculated by RAM v2.0 algorithm:

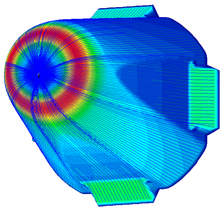


Figure 7.11: Expert shape with streamlines, $\alpha = 5°$,$\beta = 0°$, and the relative shear stress pattern on the surface

The values of the RAM v2.0 will be presented against the data extracted from a $M - \alpha$ aerodynamic coefficients matrix database, generated for the Expert mission design. The data are generated from a Thales Alenia in-house flight mechanics Simulink tool, called FMST. The aerodynamic tools used inside this design software are substantially the same used by RAM v1.0, i.e. LSI methods such as the modified Newtonian theory. There is then no clue about the fact that the data provided from this Expert database are resembling the actual values more than our RAM v2.0 viscous or inviscid results. However, we will present here the comparison anyway, in order to show that the coefficients obtained in sec. 5.7 are plausible:

Table 7.3: Expert lift coefficient comparison - FMST values (top) against RAM v1.0 inviscid values (center) and RAM v2.0 viscous values (bottom)

FMST C_l	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.0000	0.0000	0.0000
$\alpha = 5°$	0.0648	0.0530	0.0611
$\alpha = 10°$	0.1243	0.1009	0.1168
$\alpha = 20°$	0.2219	0.1781	0.2078
RAM v1.0 C_l	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.0000	0.0000	0.0000
$\alpha = 5°$	0.0948	0.0955	0.0958
$\alpha = 10°$	0.1818	0.1832	0.1837
$\alpha = 20°$	0.3245	0.3269	0.3277
RAM v2.0 C_l	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.0000	0.0000	0.0000
$\alpha = 5°$	0.0946	0.0950	0.0946
$\alpha = 10°$	0.1814	0.1822	0.1814
$\alpha = 20°$	0.3238	0.3251	0.3236

Table 7.4: Expert drag coefficient comparison - FMST values (top) against RAM v1.0 inviscid values (center) and RAM v2.0 viscous values (bottom)

FMST C_d	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.3659	0.3474	0.3446
$\alpha = 5°$	0.3758	0.3561	0.3539
$\alpha = 10°$	0.4065	0.3834	0.3834
$\alpha = 20°$	0.5151	0.4799	0.4896
RAM v1.0 C_d	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.3905	0.3934	0.3944
$\alpha = 5°$	0.4030	0.4060	0.4071
$\alpha = 10°$	0.4395	0.4428	0.4439
$\alpha = 20°$	0.5738	0.5781	0.5796
RAM v2.0 C_d	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.3936	0.4008	0.4105
$\alpha = 5°$	0.4061	0.4134	0.4232
$\alpha = 10°$	0.4425	0.4502	0.4604
$\alpha = 20°$	0.5769	0.5858	0.5968

Table 7.5: Expert pitching moment coefficient comparison - FMST values (top) against RAM v1.0 inviscid values (center) and RAM v2.0 viscous values (bottom)

FMST $C_{\mathcal{M}}$	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.0000	0.0000	0.0000
$\alpha = 5°$	-0.0063	-0.0039	-0.0067
$\alpha = 10°$	-0.0125	-0.0078	-0.0134
$\alpha = 20°$	-0.0261	-0.0162	-0.0278

RAM v1.0 $C_{\mathcal{M}}$	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.0000	0.0000	0.0000
$\alpha = 5°$	-0.0084	-0.0085	-0.0085
$\alpha = 10°$	-0.0166	-0.0167	-0.0167
$\alpha = 20°$	-0.0329	-0.0331	-0.0332

RAM v2.0 $C_{\mathcal{M}}$	$M = 6$	$M = 10$	$M = 16$
$\alpha = 0°$	0.0000	0.0000	0.0000
$\alpha = 5°$	-0.0085	-0.0088	-0.0093
$\alpha = 10°$	-0.0168	-0.0173	-0.0181
$\alpha = 20°$	-0.0332	-0.0340	-0.0351

Since our configuration has no sideslip, the side force and the moment coefficients around the yaw and roll angle results zero under a $10e^{-10}$ tolerance.

Notice that RAM produces higher drag coefficients with respect with the FMST algorithm, expecially when using the RAM v2.0 viscous calculation, coherently with the account for viscous effects. RAM produces also higher lift coefficients than FMST.

As for the pitching moment, FMST shows a particular behaviour, with the coefficient showing a great non-monotonic dependence over Mach number, increasing and then decreasing with M_∞. RAM algorithm shows instead a constant behaviour of the pitching moment coefficient as the Mach number rises.

These results are obtained for a zero sideslip angle. We will confront now some results from the Expert database, obtained from the same database, but with an aerodynamic roll angle $\phi = 45°$, in order to obtain some values on the lateral force and on the yaw and roll angles. We report the conversion between the angles on which the database is parametrized, i.e. the total angle of attack α_T and the roll angle ϕ, and the classical α and $\bar{\beta}$, where :

$$\cos\alpha_T = \cos\alpha\cos\bar{\beta} \tag{7.1}$$

$$\tan\phi = \frac{\tan\beta}{\sin\alpha} \tag{7.2}$$

$$\tan\alpha = \tan\alpha_T \cos\phi \tag{7.3}$$

$$\sin\bar{\beta} = \sin\phi \sin\alpha_T \tag{7.4}$$

We remember that our sideslip angle is revered with respect to the classical definition, hence:

$$\beta = -\bar{\beta} \tag{7.5}$$

We report the comparison of forces and moments coefficients for the above flight conditions for $\phi = 45°$ and $\alpha_T = 5°$, which correspond in our reference system to the angles

$$\alpha = 3.535° \tag{7.6}$$

and

$$\beta = 3.535° \tag{7.7}$$

Freestream condition are the same defined in the preceeding cases for $M = 6,\ 10,\ 16$

Table 7.6: Aerodynamic forces and moments coefficient comparison between FMST and RAM v2.0, $\alpha = 3.535°$, $\beta = -3.535°$

	FMST			*RAM v2.0*		
	$M = 6$	$M = 10$	$M = 16$	$M = 6$	$M = 10$	$M = 16$
C_l	0.0648	0.0530	0.0611	0.0675	0.0672	0.0669
C_d	0.3578	0.3561	0.3539	0.3980	0.4020	0.4118
C_s	-0.0487	-0.0419	-0.0459	-0.1172	-0.1175	-0.1184
$C_{\mathcal{L}}$	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
$C_{\mathcal{M}}$	-0.0044	-0.0028	-0.0047	-0.0061	-0.0062	-0.0065
$C_{\mathcal{N}}$	-0.0044	-0.0028	-0.0047	-0.0061	-0.0062	-0.0066

As we can see, even in this case our algorithm provides higher estimates of the lift and drag coefficients, and shows no moment coefficent dependency over the Mach number. Notice the fact that the yaw and pitching moments coefficients are identical, due to the Expert axial symmetry and to the equal angle deflection in the α and β directions.

Chapter 8

Conclusions and future developments

8.1 Considerations on final results

The results provided in the preceeding chapter show that an accurate reconstruction of the shear stress and the heat flow over the surface can be obtained from approximate streamline reconstruction and reference enthalpy methods, as can be seen in the comparisons with IXV VKI simulations. The limitations included in the determination of the heat flow and temperature distribution through Reynolds analogy produce however an underestimation of these quantities in the stagnation region.

Comparisons with Thales approximated coefficients on the Expert shape show that a good reconstruction of the aerodynamic coefficients is achieved. The confront with RAM v1.0 inviscid calculations shows however that, except for the drag coefficient, which sees a noticeable rise, as can be expected from the fact that we have introduced the viscous drag, the other coefficients are not sensibly modified by the shear stress distribution over the vehicle surface. This must not surprise us, since, as can be seen by the IXV datas obtained from our algorithm or from VKI simulations, the magnitude of the shear stresses over the surface is of the order of some percents compared to the pressure inviscid field. This means that the contribution of the shear stresses integration over the surface produces sensible results only in the C_D calculation. Since the calculation times for RAM v2.0 are at the present moment of the order of minutes - by now 305.76 seconds are taken for the viscous calculation on the IXV shape $M_\infty = 11$ simulation presented in the previous chapter using a Dell laptop PC - a viscous approximate solution for the only purpose to obtain aerodynamic coefficients could be still too expensive. This problem can be solved by an appropriate future implementation in C++, using the classes defined in WgsReader, presented in appendix B. We will expand the optimization of the C++ implementation in the section dedicated to future developments.

However, the present work shows the feasibility of a viscous approximate solver and validates the concept that results of an accuracy suitable to preliminary design and optimization of space vehicles can be obtained, using calculation times which are signifcantly reduced if compared with an hypersonic CFD simulation with high temperature effects, which still take several hours even if using dedicated machines. Several tools for the automatic treatment of velocity fields are also provided in this work, looking for example at the stagnation point determination algorithm or the quadrilateral interpolation and derivation tools, which are powerful developing tools for an approximated solver of any nature.

To improve the heat flow results in the stagnation region, in order to get a more plausible distribution of q_w over the surface, an option to correct the heat flow over the nose is included in RAM v2.0. As already discussed in sec. 3.6.2, a better representation of the heat flow in the stagnation region can be reached by imposing the heat flux peak value in the stagnation region. The following figure presents this concept over the DART test case, showing that by this mean the local and integral error over the heat flow distribution are greatly reduced, with respect to the pure application of Reynolds analogy.

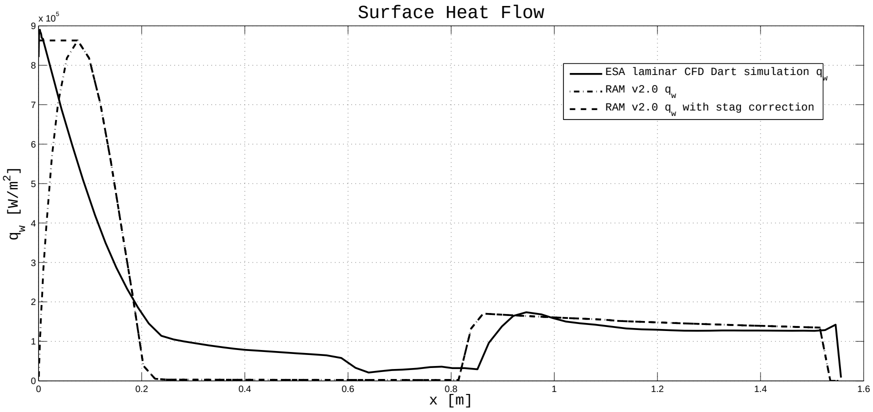


Figure 8.1: DART heat flow distribution, displaying actual ESA results together with the pure Reynolds analogy implementation and the heat peak corrected flow in the stagnation region.

This solution may seem rough, but is the only way to obtain more accurate matches without a CFD solving of the external flow, coupled with a thermal analysis of the entire flowfield. This is surely is not a task which can be performed by an approximate solver, since the only approximated correlation which exists between shear stress and heat flow is the Reynolds analysis, which constrains the heat flow trend to adapt to the c_f distribution, hence misrepresenting the stagnation region. We show here the results for the heat flow and temperature distribution over the IXV geometry, for the $M_\infty = 11$ simulation presented in the preceding chapter, where can be seen that a better agreement with simulation datas can be achieved.

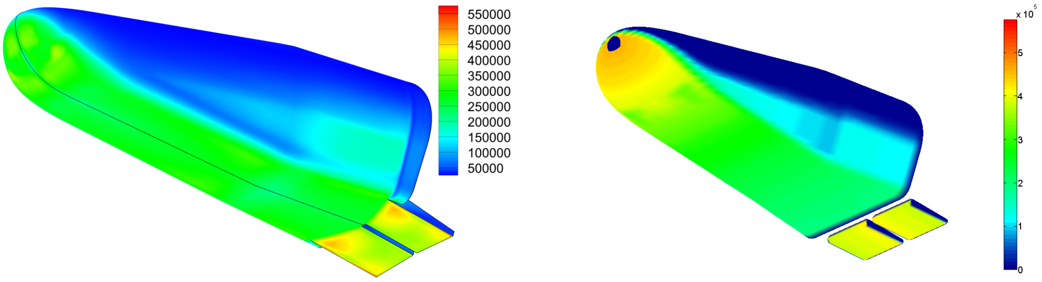


Figure 8.2: q_w $[\frac{W}{m^2}]$ distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution with thermal distribution correction (right) - IXV geometry - $M_\infty = 11.035$, $\alpha = 45°$, $\beta = 0°$

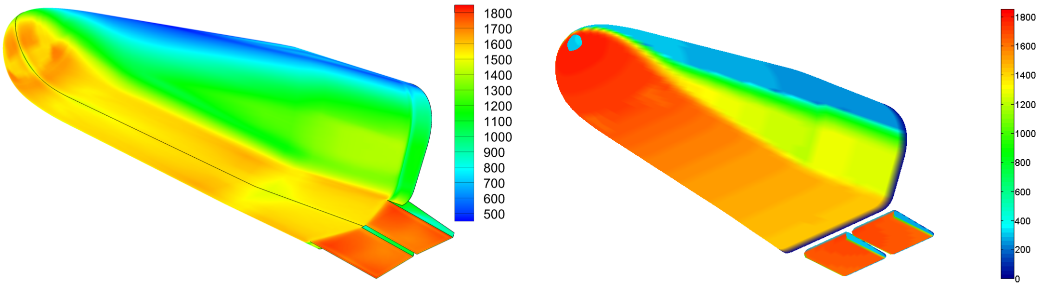


Figure 8.3: T_w $[\frac{W}{m^2}]$ distribution comparison between VKI CFD simulations (left) and RAM v2.0 solution with thermal distribution correction (right) - IXV geometry - $M_\infty = 11.035$, $\alpha = 45°$, $\beta = 0°$

Using these results will lead to a distribution which is suitable to perform vehicle geometry shape optimization in a preliminar design phase, granting us reliable confronts between different set-ups.

8.2 Future developments

The RAM v2.0, although has proven to reconstruct satisfying shear stress and heat flow distribution, could take advantage of several improvements, both on the front of the modeling as of the calculation optimization.

As for the flow modelization, an improved implementation of Simeonides algorithm can be afforded, in particular by linking the automatic part characterization elaborated by Dirkx, and presented in sec. 5.1, which permits to identify a region of the vehicle as blunt, non blunt, three- or two- dimensional, with the appropriate Simeonides coefficients, as presented in tables 3.2 and 3.3. This solution will lead to a more accurate reconstruction of the viscous and thermal solutions over geometries of different nature.

The other main improvement that will be done during the translation of the algorithm from Matlab to C++ is the search optimization through search trees implementation. Trees are informatic data structures used to optimize queries on big amount of data. Data is organized in a hierarchical way from a first data node called root. From the root there departs two or more nodes, called branches or childs. Every step downwards is called a level, and the number of childs is increasing depending of the number of childs for every parent. The branches which are not connected to new childs are called leaves. If the number of childs for every parent is restricted to two, we are using binary trees. These structures can be utilized for optimizing the data in order to perform faster searches inside the database. In the present work, they will be used to enhance the search of the centroid mesh point nearest to a given streamline point, since, as we have seen in sec. 5.6.2, streamlines are defined over different points with respect to the LaWGS mesh, so the tree data structure will be optimized for nearest neighbour distance searches. The concept behind nearest neighbour search is very simple. We have a point that has to be confronted with a set of points - e.g., in our algorithm, the streamline point with respect to the centroid mesh , so what the algorithm basically does is to split every time the set of points in two. Imagine you have to search which is the nearest point in an 1D set of points, e.g. a line, to a new point you want to evaluate. We split the domain in two, by the point which is the nearest to the center of the line. If the point searched is contained in the right child set of points, then the left child set is discarded and the first is kept. The child set which contains our point becomes the new parent. Then the set of points is split again in two childs, right and left, and we search if our point is placed in the first or second child, and the correct child becomes the new parent. By doing so the number of points between which you have to search decreases at every time, and you get a faster search. The points of our mesh will be put inside the tree following a positioning criteria, so that if a point lies in a certain section of the domain it will be put in a certain branch of a node, according to the principle of nearest neighbour search, in order to reduce search time. This concept will be extended in the next subsections. The idea behind trees is that you take some calculation effort to build a tree, but after this you will have very fast researches in the tree, due to the optimized indexing. For more information, one could want to look at [25], where there is an exhaustive chapter on trees description and implementation, and at [5], where the implementation of search trees to approximate hypersonic solutors is performed.

To conclude the outline of the future developments, there are some fixes that have to be done to RAM v2.0, the main of them could be an automated deflection of moving parts, such as flaps, using the appropriate part rotation definition contained in the LaWgs format, in order to obtain easily solutions with deflected surfaces, to perform better correlations of the results over maneuvering simulations such as the IXV VKI CFD models.

Appendix A

Reference case Tutorial

In this chapter we report a tutorial to show how to correctly calculate data output for a given *.wgs file, showing how the data presented in this work can be calculated. Files and folders are referred to the attached disk folder structure, but it is not mandatory that WgsReader and Matlab algorithm are in the same place. The tutorial and screenshots presented here are referred to the IXV $M_\infty = 6$ calculation presented in sec. 7.1.2, for all the others cases the procedure is identical.

A.1 WgsReader Preprocessing

A.1.1 Defining the simulation parameters

First, one should open WgsReader, which is in the Ram V2.0 main folder. The programs consists of three data interfaces, the first one reading the currently loaded *.wgs file contents, the second shows the point mesh geometry of the processed file, the third is a multipurpose window in which we will load the Matlab preprocessing script that will be used in the second part of the algorithm.

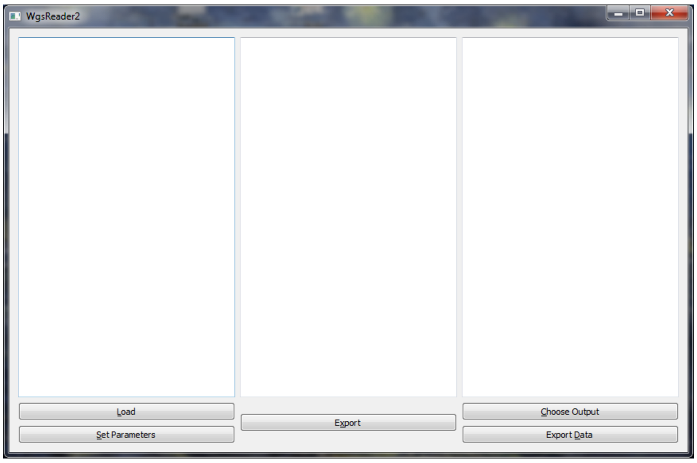
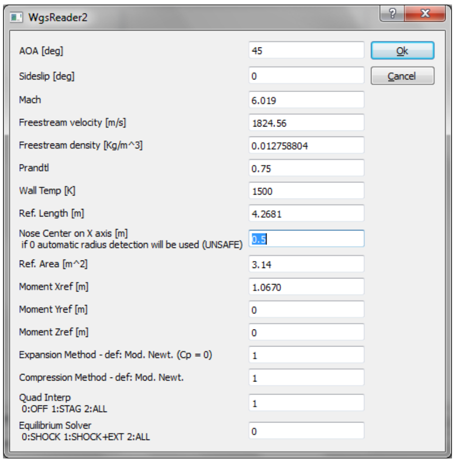


Figure A.1: WgsReader Main Window

We show here the preprocessing procedure which has to be done with WgsReader in order to get a proper Matlab input pre-processing file.

Once the program is running, the first thing to do is to set the freestream and simulation parameters, which can be reached using the Set Parameters button. The button opens a dialog window in which all the properties needed for a simulation have to be set:



AOA [deg]	45
Sideslip [deg]	0
Mach	6.019
Freestream velocity [m/s]	1824.56
Freestream density [Kg/m^3]	0.012758804
Prandtl	0.75
Wall Temp [K]	1500
Ref. Length [m]	4.2681
Nose Center on X axis [m] if 0 automatic radius detection will be used (UNSAFE)	0.5
Ref. Area [m^2]	3.14
Moment Xref [m]	1.0670
Moment Yref [m]	0
Moment Zref [m]	0
Expansion Method - def: Mod. Newt. (Cp = 0)	1
Compression Method - def: Mod. Newt.	1
Quad Interp 0:OFF 1:STAG 2:ALL	1
Equilibrium Solver 0:SHOCK 1:SHOCK+EXT 2:ALL	0

Figure A.2: Set Properties Window - the properties set here are referred to the IXV $M_\infty = 6$ simulation

All the parameters which are set here can affect dramatically the results, so they have to be introduced carefully. We give here a brief description of every entry:

- AOA: insert here the angle of attack α - in *degrees*

- Sideslip: insert here the sideslip angle β - in *degrees*
- Mach: the freestream Mach number M_∞ is to be inserted.
- Freestream velocity V_∞ must be put here, in $\left[\frac{m}{s}\right]$
- Freestream density ρ_∞ must be inserted here, in $\left[\frac{kg}{m^3}\right]$.
  - The other freestream quantities are calculated from the following perfect gas relations, since the freestream gas is considered to be not dissociated before the shock, with $\gamma_\infty = 1.4$ and $R_\infty = 286.9 \left[\frac{J}{kg\,K}\right]$ for standard air:

$$c_\infty = \frac{V_\infty}{M_\infty} \tag{A.1}$$

$$T_\infty = \frac{c_\infty}{\gamma_\infty R_\infty} \tag{A.2}$$

$$P_\infty = \rho_\infty R_\infty T_\infty \tag{A.3}$$

- Prandtl: freestream Pr_∞ number goes here
- Wall Temp: this cell contains an initial guess for the wall temperature T_w. It is not used anymore and the default value should be left.
- Ref. length: here the reference length that will be used for the aerodynamic coefficients calculations, in $[m]$.
- Nose center on X axis: the position of the nose center x_{nose} $[m]$, from which the nose radius of every point in the stagnation region will be calculated. This is a key value, since the friction factor and heat flow relations show a great dependence from nose radius. We report here the nose center values used in the production of the results contained in chapter 7 and in the validation of the software. Notice that the nose of some LaWgs geometries could not be at $x = 0$, hence the actual nose center position as defined in the original LaWgs main coordinate system has to be inserted.

Table A.1: Nose center coordinates for the LaWgs geometries analyzed in this work

vehicle	nose center x coordinate
DART	$x_{nose} = 0.225$
IXV	$x_{nose} = 0.5$
Expert	$x_{nose} = -0.75$

The procedure is applied only on fusiform parts, the nose on non-fusiform parts being determinated from the actual wing geometry. Unfortunately, at the present moment, this means that only one center can be imposed, so only one main fusiform part will show correct nose radius calculation. To obtain good correlations over different fusiform parts, one should repeat the preprocessing phase for every new fusiform part, using the results of the algorithm only on that part and discarding the others. This feature will be corrected in future release, but is not a major problem, since non exotic re-entering space vehicles usually have only a single fusiform body with winged extensions. If the nose radius is not reported, an automatic calculator of the nose radius will be used, but a single radius for every point of the nose will be used, hence loosing a great part of the algorithm accuracy in the nose region.

- Ref Area: this is the reference area used for aerodynamic coefficient calculations, in $[m^2]$
- Moment Xref, Yref, Zref: these are the moment coefficients reference point coordinates, expressed in $[m]$ from the center of the LaWGS mesh main coordinate system.
- Expansion method: this parameter select the expansion method, from the models described in chapter 2.
- Compression method: this parameter identifies the compression model, taken from chapter 2 as well. Expansion and compression method input selection are resumed in table A.2:

Table A.2: Compression and Expansion Method Selection Table

input	*Compression*	*Expansion*
0	Newtonian	Vacuum
1 (default)	Modified Newtonian	Modified Newtonian
2	Mod. Newtonian - Prandtl Meyer	Mod. Newtonian - Prandtl Meyer
3	Tangent Wedge	PM Freestream
4	Tangent Wedge Empirical	High Mach Base Pressure
5	Tangent Cone Empirical	*Van Dyke Unified*
6	*Modified Dahlem Buck*	ACM Empirical - see sec. 5.1

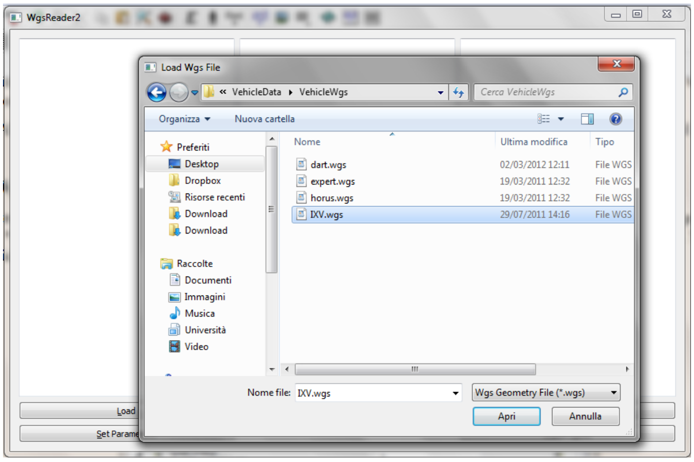
The methods indicated in *Italic* are not exposed in this work, since they are RAM v1.0 add-ons taken from other solvers procedures, some of them being used e.g. in [3]. However, they haven't been tested and validated yet, so their verification is left to the user.

- Quad Interp: selects where the quadrilateral interpolation will be used:
  - 0: quadrilateral interpolation is not used - faster calculation but less accurate results
  - 1: quadrilateral interpolation is used in the stagnation region - default choice
  - 2: quadrilateral interpolation is used everywhere - slower calculation
- Equilibrium solver: selects where the thermochemical equilibrium gas model will be used:
  - 0: Equilibrium is used through the shock, the mixture being considered frozen afterward - default choiche
  - 1: Equilibrium is imposed through the shock and for the determination of the external flow properties - slower cand without significant increment of accuracy calculation.
  - 2: Equilibrium is imposed everywhere, even in the boundary layer reference values determination - dramatically slow and without accuracy increment calculation, should be avoided.

As the parameters for the RAM calculation are set, clicking the ok button saves the values into WgsReader, preparing us to load and pre-process the geometry file.

A.1.2 Loading the LaWgs file

Once the simulation parameters have been set, we can load the *.wgs file which contains the complete vehicle geometry in LaWgs format. In our distribution, one can find the *.wgs input files for all the configurations showed in this work in the subfolder \VehicleData\VehicleWgs. The Load button opens a dialog window in which we can search the correct *.wgs file and load it into the WgsReader.



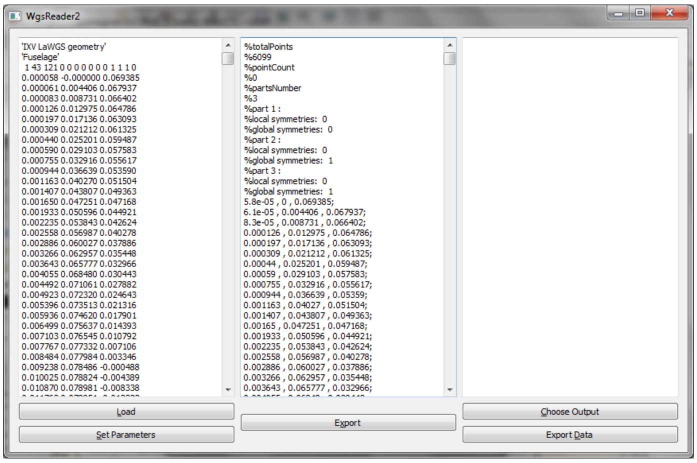
Nome	Ultima modifica	Tipo
dart.wgs	02/03/2012 12:11	File WGS
expert.wgs	19/03/2011 12:32	File WGS
horus.wgs	19/03/2011 12:32	File WGS
IXV.wgs	29/07/2011 14:16	File WGS

Nome file: IXV.wgs

Wgs Geometry File (*.wgs)

Figure A.3: Wgs File loading

After the file is loaded, all the LaWgs transformations and RAM v1.0 inviscid calculation are performed, and the other simulation parameters and Matlab arrays containing vehicle centroid geometry, normals, velocities, and areas for every vehicle part are generated. The file is correctly loaded when in the left window the content of the original *.wgs file is shown, while the center windows contains the coordinates of the vehicle points after the LaWgs transformations.



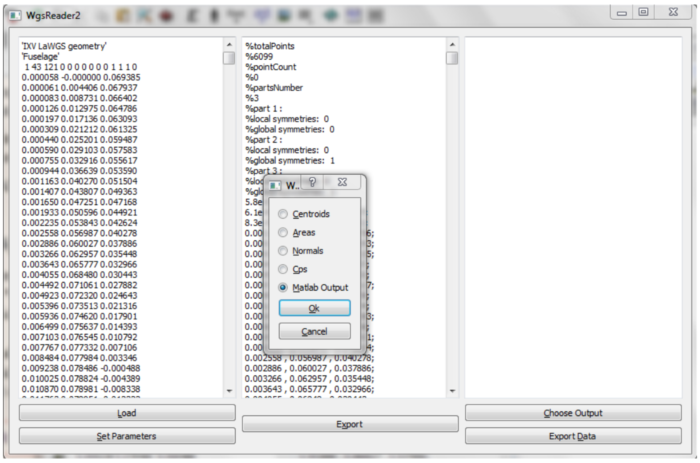
```
'IXV LaWGS geometry'
'Fuselage'
 1 43 121 0 0 0 0 0 0 0 1 1 1 0
0.000058 -0.000000 0.069385
0.000061 0.004406 0.067937
0.000083 0.008731 0.066402
0.000126 0.012975 0.064786
0.000197 0.017136 0.063093
0.000309 0.021212 0.061325
0.000440 0.025201 0.059487
0.000590 0.029103 0.057583
0.000755 0.032916 0.055617
0.000944 0.036639 0.053590
0.001163 0.040270 0.051504
0.001407 0.043807 0.049363
0.001650 0.047251 0.047168
0.001933 0.050596 0.044921
0.002235 0.053843 0.042624
0.002558 0.056987 0.040278
0.002886 0.060027 0.037886
0.003266 0.062957 0.035448
0.003643 0.065777 0.032966
0.004055 0.068480 0.030443
0.004492 0.071061 0.027882
0.004923 0.072320 0.024643
0.005396 0.073513 0.021316
0.005936 0.074620 0.017901
0.006499 0.075637 0.014393
0.007103 0.076545 0.010792
0.007767 0.077332 0.007106
0.008484 0.077984 0.003346
0.009238 0.078486 -0.000488
0.010025 0.078824 -0.004389
0.010870 0.078981 -0.008338
```

```
%totalPoints
%6099
%pointCount
%0
%partsNumber
%3
%part 1 :
%local symmetries:  0
%global symmetries:  0
%part 2 :
%local symmetries:  0
%global symmetries:  1
%part 3 :
%local symmetries:  0
%global symmetries:  1
5.8e-05 , 0 , 0.069385;
6.1e-05 , 0.004406 , 0.067937;
8.3e-05 , 0.008731 , 0.066402;
0.000126 , 0.012975 , 0.064786;
0.000197 , 0.017136 , 0.063093;
0.000309 , 0.021212 , 0.061325;
0.00044 , 0.025201 , 0.059487;
0.00059 , 0.029103 , 0.057583;
0.000755 , 0.032916 , 0.055617;
0.000944 , 0.036639 , 0.05359;
0.001163 , 0.04027 , 0.051504;
0.001407 , 0.043807 , 0.049363;
0.00165 , 0.047251 , 0.047168;
0.001933 , 0.050596 , 0.044921;
0.002235 , 0.053843 , 0.042624;
0.002558 , 0.056987 , 0.040278;
0.002886 , 0.060027 , 0.037886;
0.003266 , 0.062957 , 0.035448;
0.003643 , 0.065777 , 0.032966;
```

Figure A.4: WgsReader after loading a *.wgs file

A.1.3 Saving the Matlab pre-processing input

At this point, we have to export the Matlab pre-processing file, using the third window on the right. First we load the Matlab data into the window, by pushing the Choose Output button, and selecting the Matlab Output in the little dialog which displays:



WgsReader2

```
'IXV LaWGS geometry'
'Fuselage'
 1 43 121 0 0 0 0 0 0 0 1 1 1 0
0.000058 -0.000000 0.069385
0.000061 0.004406 0.067937
0.000083 0.008731 0.066402
0.000126 0.012975 0.064786
0.000197 0.017136 0.063093
0.000309 0.021212 0.061325
0.000440 0.025201 0.059487
0.000590 0.029103 0.057583
0.000755 0.032916 0.055617
0.000944 0.036639 0.053590
0.001163 0.040270 0.051504
0.001407 0.043807 0.049363
0.001650 0.047251 0.047168
0.001933 0.050596 0.044921
0.002235 0.053843 0.042624
0.002558 0.056987 0.040278
0.002886 0.060027 0.037886
0.003266 0.062957 0.035448
0.003643 0.065777 0.032966
0.004055 0.068480 0.030443
0.004492 0.071061 0.027882
0.004923 0.072320 0.024643
0.005396 0.073513 0.021316
0.005936 0.074620 0.017901
0.006499 0.075637 0.014393
0.007103 0.076545 0.010792
0.007767 0.077332 0.007106
0.008484 0.077984 0.003346
0.009238 0.078486 -0.000488
0.010025 0.078824 -0.004389
0.010870 0.078981 -0.008338
```

```
%totalPoints
%6099
%pointCount
%0
%partsNumber
%3
%part 1 :
%local symmetries:  0
%global symmetries:  0
%part 2 :
%local symmetries:  0
%global symmetries:  1
%part 3 :
%loc
%glo
5.8e
6.1e
8.3e
0.00                6;
0.00                3;
0.00                5;
0.00                ;
0.00                ;
0.00                7;
0.00                ;
0.00                ;
0.00                3;
0.00                ;
0.00                1;
0.00                4;
0.002558 , 0.056987 , 0.040278;
0.002886 , 0.060027 , 0.037886;
0.003266 , 0.062957 , 0.035448;
0.003643 , 0.065777 , 0.032966;
```

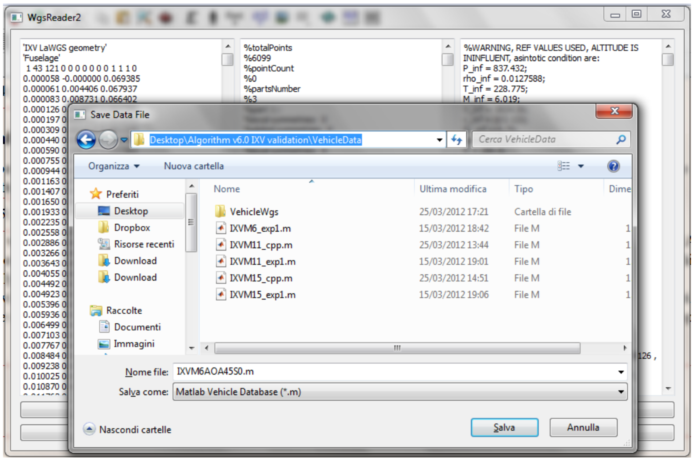
W..

○ Centroids
○ Areas
○ Normals
○ Cps
● Matlab Output

Figure A.5: Choosing Matlab output

Matlab input file content will show up in the third window. Finally, we can export the Matlab input file to be processed by the RAM v2.0 Matlab script. First we select the export data button, then we save the *.m file into the \VehicleData\ folder of the RAM Algorithm main folder. This step is crucial for RAM v2.0 to work properly.

The filename which we will give to this file will be the one to be invoked in Matlab, and it has to be representative of the configuration choosen, e.g. *IXVM6AOA45S0.m* can be the name of a file for an IXV simulation at $M = 6$, $\alpha = 45°$, $\beta = 0°$. Any filename not containing spaces or special characters can be choosen anyway.



WgsReader2

```
'IXV LaWGS geometry'
'Fuselage'
 1 43 121 0 0 0 0 0 0 0 1 1 1 0
0.000058 -0.000000 0.069385
0.000061 0.004406 0.067937
0.000083 0.008731 0.066402
0.000126 0
0.000197 0
0.000309 0
0.000440 0
0.000590 0
0.000755 0
0.000944 0
0.001163 0
0.001407 0
0.001650 0
0.001933 0
0.002235 0
0.002558 0
0.002886 0
0.003266 0
0.003643 0
0.004055 0
0.004492 0
0.004923 0
0.005396 0
0.005936 0
0.006499 0
0.007103 0
0.007767 0
0.008484 0
0.009238 0
0.010025 0
0.010870 0
```

```
%totalPoints
%6099
%pointCount
%0
%partsNumber
%3
```

```
%WARNING, REF VALUES USED, ALTITUDE IS
ININFLUENT, asintotic condition are:
P_inf = 837.432;
rho_inf = 0.0127588;
T_inf = 228.775;
M_inf = 6.019;
126 ,
```

Save Data File

Desktop\Algorithm v6.0 IXV validation\VehicleData

Nome	Ultima modifica	Tipo	Dime
VehicleWgs	25/03/2012 17:21	Cartella di file	
IXVM6_exp1.m	15/03/2012 18:42	File M	1
IXVM11_cpp.m	25/03/2012 13:44	File M	1
IXVM11_exp1.m	15/03/2012 19:01	File M	1
IXVM15_cpp.m	25/03/2012 14:51	File M	1
IXVM15_exp1.m	15/03/2012 19:06	File M	1

Nome file: IXVM6AOA45S0.m

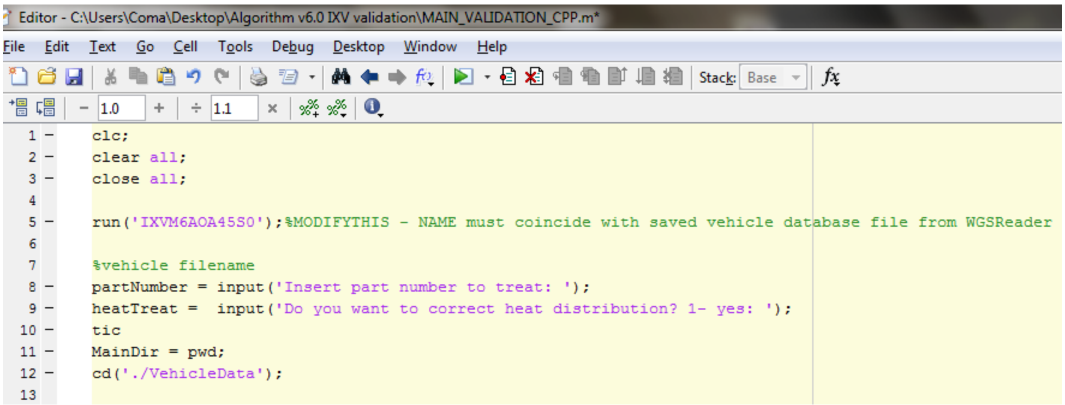
Salva come: Matlab Vehicle Database (*.m)

Figure A.6: Saving the *.m file inside VehicleData folder

If the pre-processing file is saved in the proper directory, we can close WgsReader and head for Matlab to run the RAM v2.0 calculation. If however we want to save another pre-processing file, we can define new properties in the Set Parameters window, and then we have to load again the *.wgs file and select again the Matlab Output in the Choose Output dialog, in order to reload the Matlab data into WgsReader. At this point we can export the *.m file, but not before. This is a known issue and will be fixed soon.

A.2 Matlab processing

Once the file is correctly saved in the \VehicleData\ folder, the script MAIN.m can be executed, taking care of specifying the correct vehicle database filename in the first line of the active code.



Editor - C:\Users\Coma\Desktop\Algorithm v6.0 IXV validation\MAIN_VALIDATION_CPP.m*
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
fx
1.0
1.1

```
clc;
clear all;
close all;

run('IXVM6AOA45S0');%MODIFYTHIS - NAME must coincide with saved vehicle database file from WGSReader

%vehicle filename
partNumber = input('Insert part number to treat: ');
heatTreat =  input('Do you want to correct heat distribution? 1- yes: ');
tic
MainDir = pwd;
cd('./VehicleData');
```

Figure A.7: Specifying the correct database name in the beginning of the MAIN.m script, the example name *IXVM6AOA45S0* being used

The user is then asked for the part to be treated, and if he wants to perform the heat flow correction as presented in chapter 8, then viscous calculations are performed and outputs are produced and saved in the folder \Figures\. Aerodynamic coefficients are printed on the screen, however one can access the variables c_aer for force aerodynamic coefficients and c_aer_body and c_mom for moments and forces in body axes.

A script called MERGE.m is present in the \Figures\ folder to merge the output figures of different parts in one single figure.

A.3 Current Implementation

After explaining the algorithm use, we will give in this section an in depth glance at the current RAM v2.0 implementation status, explaining how RAM v1.0 is actually coded and how it interacts with the RAM v2.0 algorithm. The current RAM v1.0 algorithm has been extrapolated from RAM v1.0 STA Cambrian release classes, and included in a stand-alone software program, called WgsReader, which performs the inviscid calculation and the preliminar velocity field versor calculation using the thin shock layer hypothesis.

A.3.1 RAM v1.0 classes

The current RAM v1.0 module is coded inside the STA project, written in C++ and using the open source Nokia Qt libraries. These libraries contain several tools used to develop graphical user interfaces, or GUI, which have been used to shape STA environment as well as the Re-entry Aerodynamic Module properties window, which defines the aerodynamic properties of the vehicles introduced in the space scenario of the current simulation. RAM v1.0 consist of six files, which are in part dedicated to the aerodynamic properties GUI, and in part to the actual calculation of the inviscid flow. Obviously, the classes contained in this file are linked to various levels of the STA software: for example the aerodynamic properties GUI is linked to the main STA graphic environment, and the inviscid calculations performed by RAM v1.0 can be used for the re-entry trajectory propagation in a STA simulation, as well as to define aerodynamic coefficients for a given flight condition.

We will describe here the RAM v1.0 implementation files, focusing on the source code related to the inviscid calculation classes, which have been exported to WgsReader, since the GUI of this little stand-alone program was developed independently by us and is not strictly related with this work. RAM v1.0 consist of six C++ files, developed in the Qt IDE by Dominic Dirkx, with their related headers. Files related to GUI implementations are accompanied also by *.ui files which contain graphic design of the interface in the Qt libraries format. The structure of the module is the following:

- ► GUI definition files, these files are used only in STA project, and are reported here for sake of completness:
  - ▷ *aerodynamicmethods.cpp*, *aerodynamicmethods.h, aerodynamicmethods.ui:* these files are related to the main RAM v1.0 user interface window, in which all the general vehicle aerodynamic properties are described, such as reference lengths and surfaces and other features. Loading of *.wgs files containing the LaWGS geometry is also permitted. Through this interface we can select two other windows, the Advanced Methods Selection and the Parametrization windows, which are presented herebelow.

  - ▷ *advancedselectionGUI.cpp, advancedselectionGUI.h, advancedselectionGUI.ui:* these files define the Advanced Methods Selection window, in which the method selection used for the inviscid solver, which has been defined in table 5.1, can be modified by the user. An output file containing the aerodynamic properties, such as aerodynamic coefficients, can be generated.
  - ▷ *parametrization.cpp, parametrization.h, parametrizedgeometry.ui:* these sources contains a routine which permits to generate new *.wgs parametrized capsule geometries, as described by Dirkx in [1].

- ▶ Inviscid calculation files, these files are used to read *.wgs files and to perform inviscid calculations. They have been exported to WgsReader, and modification to the original STA files are underlined here:

  - ▷ *vehiclegeometry.cpp, vehiclegeometry.h:* these files include the definition of VehicleGeometry class, which is the main RAM v1.0 class. This class loads the *.wgs file, reads all the data in the LaWGS format and creates the single parts which compose the vehicle. This class has been renamed ReadGeometry in WgsReader program, hence the respective files are now called *readgeometry.cpp, readgeometry.h*
  - ▷ *partgeometry.cpp, partgeometry.h:* these files defines the PartGeometry class, which is used from VehicleGeometry class to define single parts objects. This class includes all the transformation functions which perform the symmetric mirroring, the rotation, the translation and the scaling, as specified in the LaWGS header. A panel properties calculation function, which calculates the centroids $\mathbf{x}_{c,ij}$ from the original *.wgs points shape and defines all the properties such as local normals $\hat{\mathbf{n}}_{ij}$ and centroid areas A_{ij} is present. Centroid's inviscid pressure coefficients $c_{p,ij}$ are also initialized. Other classes contained in these files are the PartAnalysis class and its childs PlanarPartAnalysis and FusiformPartAnalysis, which perform automated controls such as automatic blunt or slender nose determination, and all the automated geometry identification operations presented in sec. 5.1.2 and the relative part splitting.
  - ▷ *aeroanalysis.cpp, aeroanalysis.h:* these files contain the class AeroAnalysis, which is the place where all the RAM v1.0 inviscid algorithm calculations are performed. This class takes a PartGeometry object as an input and calculates the inviscid flow as it has been defined in chapter 2. The AeroAnalysis class contains functions which

take the input parts, determine the local inclination of every centroid, hence defining the leeward and windward portions, as defined by relation 2.12, and performs the appropriate compression or expansion calculations, choosing the LSI method which matches the user selection for that kind of region. A function for every method presented in chapter 2 is defined in this class, so that we have every tool needed to solve the inviscid flowfield. Functions for writing output are also defined in this class

A.3.2 WgsReader classes

Inside the WgsReader programs, which contains the inviscid calculations files from STA RAM v1.0 module, i.e. *readgeometry.cpp, readgeometry.h, partgeometry.cpp, partgeometry.h, aeroanalysis.cpp, aeroanalysis.h* files and related classes, we have new classes definition, partially to have an independent GUI which let us perform RAM v1.0 calculations from *.wgs files independently from STA environment, and partially to implement new features related to the viscous analysis. These new classes are still in an embrional phase, but they contain the initialization of RAM v2.0 features, such as streamlines, shock calculation and running length determination. *.ui files are not used here since we have used Qt libraries without external design file support. We will look at the main new classes of WgsReader, which are added to the ones we have exported from RAM v1.0:

- *wgsreader.h, wgsreader.cpp*: here the main program core is contained in the WgsReader class, along with the main window GUI definition, the *.wgs file loader and the Matlab database output routine, which creates the input database which will be used by the Matlab script in RAM v2.0, as well as other output options. In this file, the ReadGeometry vehicle is created from *.wgs file, and all the part definition, transformation and inviscid calculations performed by the PartGeometry and PartAnalysis classes are performed

- *choosemethod.cpp, choosemethod.h* contain the window which select the appropriate output option, e.g. Matlab database or text file containing different data which were used in the RAM v2.0 development and are now no longer in use.

- *setangle.h, setangle.cpp:* another GUI window, which lets the user to select the α and β angles, the freestream conditions such as M_∞, Pr_∞, V_∞, p_∞, ρ_∞, the reference length and area for aerodynamic coefficients calculations, the moments reference points, and the selection of the compression and expansion LSI methods which have to be used in the windward and leeward regions.

- *thermovehicle.h, thermovehicle.cpp*: in these files, a ThermoVehicle class, child of the ReadGeometry (ex VehicleGeometry) class is defined. This class is the viscous extension of the STA VehicleGeometry class, and takes the VehicleGeometry object which contains the parts and the inviscid flowfield results, extending it with external velocity calculation function, after shock properties determination function, running lentgh determination function, reference enthalpy calculations and isoentropic expansion functions. These function are all in an embrional phase, and are not yet used, the corrispective viscous calculations being performed by the Matlab script. The only working function which is actually used in RAM v2.0 is the determineVe function, which in WgsReader defines correctly the velocity field direction versors $\mathbf{t}_{ij}$ in every centroid, accordingly with the thin shock layer hypothesis. These versors are used in the Matlab script to determine external velocity and other features. This class will contain the main viscous routine, which is actually implemented in Matlab, in the future full translation of the RAM v2.0 viscous routine in C++ language, for STA integration.

- *streamline.h, streamline.cpp* is an empty class which will contain the streamline solutor in an apprpriate StreamLine class.

These classes are the backbone of WgsReader program, the user manual and tutorial being presented in appendix A.

A.3.3 Matlab script

The Matlab script is the exact implementation of the viscous algorithm presented in the previous chapter. It takes as input the WgsReader matlab vehicle database output file.

Bibliography

[1] Dirkx, D., *STA Re-Entry Aerodynamic Module - Technical Mathematical Note,* STA project documentation, ESTEC, 2010

[2] Anderson, John, D., *Hypersonic and High-Temperature Gas Dynamics Second Edition*, AIAA series, 2006

[3] Arvel E. Gentry et al. ,*The Mark IV Supersonic-Hypersonic Arbitrary Body Program- volume II*, Douglas Aircraft Company, 1973

[4] Fisher, Carren M. E., *Experiences using the Mark IV Supersonic Hypersonic Arbitrary Body program,* Aerodynamics of Hypersonic Lifting Vehicles, *AGARD Conference Proceedings 428*, 1987, pp. 31-1 to 31-18

[5] Kinney, David J, *Aero-Thermodynamics for Conceptual Design*, AIAA Paper 13382, January 2004, Reno Nevada

[6] C. Craidon, *A Description of the Langley Wireframe Geometry Standard (LaWGS) Format*, NASA Technical Report TM 85767, 1985

[7] Sims, Joseph Z., *Tables for Supersonic Flow Around Right Circular Cones at Zero Angle of Attack*, NASA SP-3004, 1964

[8] Eckert, E. R. G., *Engineering Relations for Heat Transfer and Friction in High-Velocity Laminar and Turbulent Boundary-Layer Flow over Surfaces with Constant Pressure and Temperature*, Transaction of the American Society of Mechanical Engineers, Vol. 78, No. 6, Aug 1956, p. 1273

[9] Meador, W. E., Smart, M. K., *Reference Enthalpy Method Developed from Solutions of the Boundary-Layer Equations*, AIAA Journal, Vol. 43, No. 1, January 2005, pp. 135-139

[10] Simeonides, G., *Generalized Reference Enthalpy Formulations and Simulation of Viscous Effects in Hypersonic Flow*, Shock Waves, Vol. 8 - 1998, pp. 161-172

[11] Simeonides, G., *Simple Formulation for Convective Heat Transfer Prediction over Generic Aerodynamic Configurations and Scaling of Radiation-equilibrium Wall Temperature*, ESA/ESTEC EWP-1860, 1995

[12] Reshotko, Eli, *Boundary-Layer Stability and Transition*, Annual Review of Fluid Mechanics, Vol.8, 1976

[13] Van Driest, E. R., *The Problem of Aerodynamic Heating*, Aeronautical Engineering Review , October 1956, pp. 26 – 39

[14] Gordon, S., McBride, B. J., *Thermodynamic Data for Fifty Reference Element"*, NASA TP 3287, 2001

[15] http://www.grc.nasa.gov/WWW/CEAWeb/

[16] Gordon, S., McBride, B. J., *Computer Program for Calculation of Complex Chemical Equilibrium Composition and Applications, vol I. Analysis,* NASA RP 1311, 1994

[17] Kenwright, D. N., Henze, C., Levit, C., *Feature Extraction of Separation and Attachment Lines*, IEEE Transactions on Visualization and Computer Graphic, Vol. 5 No.2, April-June 1999

[22] Cook, R. D., David, S. M., and Plesha, M. E., *Concepts and Applications of Finite Element Analysis*, John Wiley & Sons, 2001

[23] Chongyu Hua, *An inverse transformation for quadrilateral isoparametric elements - Analysis and application*, Elsevier, 1990

[24] Scholz, T., *"Numerical Prediction of Aerodynamic Coefficients and Trajectory of IXV Re-entry Vehicle"*, VKI PR 2011-24, June 2011

[25] Berger, R., *Intègration et validation expèrimentale de la mèthode VOF dans les calculs aèrodynamiques automobiles: application au cas de l'entrainment d'eau dans les conduits de climatisasions* , Universitè Libre de Bruxelles Ph.D. thesis, October 2010