MESHLESS METHODS FOR COMPUTATIONAL FLUID DYNAMICS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF
AERONAUTICS AND ASTRONAUTICS
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Aaron Jon Katz

January 2009

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Antony Jameson) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Robert W. MacCormack)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Sanjiva K. Lele)

Approved for the University Committee on Graduate Studies.

Abstract

While the generation of meshes has always posed challenges for computational scientists, the problem has become more acute in recent years. Increased computational power has enabled scientists to tackle problems of increasing size and complexity. While algorithms have seen great advances, mesh generation has lagged behind, creating a computational bottleneck. For industry and government looking to impact current and future products with simulation technology, mesh generation imposes great challenges. Many generation procedures often lack automation, requiring many man-hours, which are becoming far more expensive than computer hardware. More automated methods are less reliable for complex geometry with sharp corners, concavity, or otherwise complex features. Most mesh generation methods to date require a great deal of user expertise to obtain accurate simulation results. Since the application of computational methods to real world problems appears to be paced by mesh generation, alleviating this bottleneck potentially impacts an enormous field of problems.

Meshless methods applied to computational fluid dynamics is a relatively new area of research designed to help alleviate the burden of mesh generation. Despite their recent inception, there exists no shortage of formulations and algorithms for meshless schemes in the literature. A brief survey of the field reveals varied approaches arising from diverse mathematical backgrounds applied to a wide variety of applications. All meshless schemes attempt to bypass the use of a conventional mesh entirely or in part by discretizing governing partial differential equations on scattered clouds of points.

A goal of the present thesis is to develop a meshless scheme for computational fluid dynamics and evaluate its performance compared with conventional methods.

The meshless schemes developed in this work compare favorably with conventional finite volume methods in terms of accuracy and efficiency for the Euler and Navier-Stokes equations. The success of these schemes may be largely attributeed their sound mathematical foundation based on a local extremum diminishing property, which has been generalized to handle local clouds of points instead of mesh-based topologies.

In addition, powerful algorithms are developed to accelerate convergence for meshless schemes, which also apply to mesh based schemes in a mesh transparent manner. The convergence acceleration technique, termed "multicloud," produces schemes with convergence rates rivaling structured multigrid. However, the advantage of multicloud is that it makes no assumptions regarding mesh topology or discretization used on the finest level. Thus, multicloud is extrememly general and widely applicable.

Finally, a unique application of meshless methods is demonstrated for overset grids in which a meshless method is used to seamlessly connect different types of grids. It is shown that meshless methods provide significant advantages over conventional interpolation procedures for overset grids. This application serves to highlight the practical utility of meshless schemes for computational fluid dynamics.

Acknowledgements

While it is impossible to acknowledge all who have contributed to my thesis, I would like to mention a few of the most influential individuals. First, my advisor, Professor Antony Jameson, has served as my first and most dominate lens through which I have discovered the world of computational fluid dynamics. His passion for simple, yet powerful mathematics has provided me with a solid foundation upon which to build. Professor Jameson has always taken great pride in his own work while never closing his mind to new ideas. He has always been respectful of my needs and those of my family, for which I am very grateful. I am proud to be able to add my name to the list of Professor Jameson's advisees as yet another beneficiary of his dedication to CFD.

Next, I would like to acknowledge the members of the HPC Institute for Advanced Rotorcraft Modeling and Simulation (HIARMS) located at NASA Ames Research Center, of which I have been an intermittent member for nearly four years. First on the list is Dr. Robert Meakin, who first recruited me to work with him even before the inception of HIARMS. His invitation, which later evolved into close collaboration, provided me with enormous resources from which to formulate ideas. The subsequent team leader of HIARMS, Dr. Roger Strawn, has likewise provided me with an inviting work environment. Two members of the HIARMS team in particular, Dr. Andrew Wissink and Dr. Venkateswaran Sankaran, have been key collaborators on certain aspects of this work. In particular, the work with the overset grids was inspired by their development efforts in HIARMS.

Even more than the above mentioned, no one has been more devoted than my family. My wife has always been a believer in me. Without her, this work would

never have been brought to pass. My two kids have helped me have fun along the sometimes rocky and clouded road to my degree. Thanks to my kids, I have been able to stay excited about airplanes and helicopters even on those days of staring at a computer for hours. My wife has done a wonderful job of managing our family so that I could get my work done. This work is dedicated to them.

Finally, my fellow students and labmates deserve my sincere gratitude. They have helped to provide a great environment in which to learn and grow. They have been good friends. Thank you to all who have helped.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Defining Meshless Methods

Meshless methods applied to computational fluid dynamics (CFD) is a relatively new area of research. Despite their recent inception, there exists no shortage of formulations and algorithms for meshless schemes in the literature. A brief survey of the field reveals varied approaches arising from diverse mathematical backgrounds applied to a wide variety of applications. Sorting and classifying the many meshless methods is no simple task. To add to the confusion, meshless schemes fall under many other names including *meshfree*, *gridfree*, *gridless*, *generalized finite difference*, and *smooth particle hydrodynamics*.

Despite their varied names, all meshless schemes bypass the use of a conventional mesh to some degree. All the major fields of computational mechanics, including finite element methods (FEM), finite difference methods (FDM), and finite volume methods (FVM), have traditionally relied on the use of elements, interlaced grids, or finite volumes as the underlying structures upon which to discretize governing partial differential equations (PDE). On the other hand, meshless schemes only require clouds of points, from which PDEs may be discretized. Local clouds for each point in a domain are proximity-based subsets of the global set of points. Local clouds of points replace the more traditional forms of connectivity found in FEM, FDM, and FVM. This loose definition of connectivity forms the basis for a wide variety of numerical methods for PDE's, many of the most notable of which are discussed here.

1.2 Motivation for Developing Meshless methods

Before surveying the field of meshless methods, it is helpful to motivate their use. After all, why go to the trouble to develop meshless methods while FEM, FDM, and FVM are widely used and validated with extensive formulations and analyses? The answer lies with a difficulty common to all traditional discretizations–the difficulty of obtaining a suitable mesh. While the generation of meshes has always posed challenges for computational scientists, the problem has become more acute in recent years. Increased computational power has enabled scientists to tackle problems of increasing size and complexity. While algorithms have seen great advances, mesh generation has lagged behind, creating a computational bottleneck. For industry and government looking to impact current and future products with simulation technology, mesh generation imposes great challenges. Many generation procedures lack automation, requiring many man-hours, which are becoming far more expensive than computer hardware. More automated methods are less reliable for complex geometry with sharp corners, concavity, or otherwise complex features. Most mesh generation methods to date require a great deal of use expertise to achieve proper stretching, resolution, and structure.

The motivation behind meshless methods lies in relieveing the burden of mesh generation. Since the application of computational methods to real world problems appears to be paced by mesh generation, alleviating this bottleneck potentially impacts an enormous field of problems. It is not clear at this point how effective meshless methods will be at alleviating meshing problems. While a rigid mesh is not required, sufficiently dense point distributions are still required. Moreover, points must be grouped locally to form clouds. Obtaining optimal clouds for different methods is also a non-trivial problem. However, recent progress in the area of point distribution and cloud generation by Löhner and others [1, 2] has shown great promise is this area.

1.3 Meshless Methods in the Literature

Several of the most notable meshless methods are discussed below. These include Smooth Particle Hydrodynamics (SPH), the Meshless Local Petrov-Galerkin (MLPG) method, methods based on Radial Basis Functions (RBF), and Finite Point Methods (FPM). These methods are also summarized in excellent works by Liu [3] and Liu and Gu [4].

1.3.1 Smooth Particle Hydrodynamics

The method of SPH, introduced by Monaghan [5, 3], makes use of an integral representation of a function at a point given a set of surrounding points, called a kernel approximation. It uses no mesh, and points are free to move past one another consistent with a Lagrangian approach. While SPH was first developed to handle astrophysical phenomena in open space, the method was later applied to strucures, fracture simulation, fluid flow, and other fields. Monaghan [5] showed that the SPH method with artificial viscosity could accurately capture shock waves in one-dimensional shock tube problems. Methods based on an SPH formulation are well-suited for problems of infinite domain in which the problem size is not know a priori.

While SPH has become popular for intensely dynamic problems in which a static or even dynamic mesh may not properly resolve relavent physics, certain implementation difficulties are inherent in the method. These difficulties include the selection of a proper domain of influence with weighting functions, efficient nearest neighbor particle searching, and the determination of a smoothing length for force computations at each particle.

The Reproducing Kernel Particle Method (RKPM), introduced by Liu, Jun and Zhang [6], is very similar to the SPH method in that it uses a finite integral representation to discretize the governing PDEs. However, RKPM adds a correction function to the base kernal approximation, improving the accuracy especially near boundaries [7]. The RKPM method has been applied to fluids, structures, and acoustics [3]. Lesoine and Kaila [8] used RKPM to compute aeroelastic effects of aircraft with large control surface deflections. Zhang, Wagner, and Liu [9] showed that RKPM was well suited for domain decomposition for large-scale parallel computing.

1.3.2 Meshless Local Petrov-Galerkin

The MLPG method has arisen from the finite element community and is based on the weak form of a given PDE. While the use of the weak form of PDEs relaxes consistency requirements of field variable approximation, many algorithms in CFD bypass the rigorous use of weak forms. Weak forms require the use of numerical integration since they satisfy global integral forms of the governing equations. Numerical integration, along with other rigorous aspects of weak forms makes them computationally inefficient compared with simple FDM or FVM approaches. For example, in an overview by Belytschko [10], the element-free Galerkin method, a precursor to MLPG, acheived computational costs which were 4 to 10 times higher than low order FEM. While Jameson [11] showed the equivalency of one FVM scheme with a Galerkin method, most development in CFD has been based on strong forms of the governing equations, which lead to simple and efficient conservative schemes. Nonetheless, an immense mathematical foundation has been developed based on weak forms used for a variety of FEM applications.

Developed by Atluri and others [12, 13], the MLPG method is based on a Petrov-Galerkin formulation in which weight and trial functions used in the weak form of the equations need not be the same. This gives the method a "local" nature in which the integral in the weak form is satisfied over a local domain. The MLPG method thus requires a local "background grid" to perform the integral as demanded by the weak form. However, the integral is performed locally, relieving the need for a global background integration as is used in related methods [14, 15]. The local background grid may be simple shapes, such as circles or squares. By all practical measures, MLPG is essentially meshless.

Approximation of the field variables for the MLPG method is constructed using a moving least squares approach. Least squares representations of a function do not pass through the discrete sampling points of the function. Instead, they construct a smooth representation which minimizes the error of approximation. This fact has posed some difficulties in obtaining accurate and stable boundary conditions for the MLPG approach.

The MLPG scheme is very general and has been applied to various problems. Specific to fluid mechanics, Lin and Atluri [16] have used MLPG to solve the incompressible Navier-Stokes equations. They used an upwinding scheme for stabilization of the convection operator in the streamwise direction.

1.3.3 Meshless Methods Based on Radial Basis Functions

Radial basis functions are functions which have no preferred direction, but only depend on norms in space. Most often, the Euclidean distance is used as the norm. Common RBFs include Gaussians, thin plate splines, and multiquadrics. In general, RBFs are smooth and continuously differentiable. When used for interpolation purposes, RBF approximations are constructed such that they pass through data points exactly. It is difficult to prove any order of accuracy of such approximations since RBFs are not based on Taylor series or polynomial expansions.

While RBFs have been widely used in scattered data interpolation [17], their application to the solution of PDEs is relatively new. Kansa [18, 19] was the first to solve fluid dynamics problems with an RBF method. Kansa introduced an unsymmetric form of an RBF method in an attempt to improve accuracy and conditioning. The symmetric and unsymmetric forms were compared independently by Hon and Schaback [20] and Power and Barraco [21]. Li [22] compared an RBF method to the finite element method in terms of accuracy and efficiency, showing improved accuracy of the RBF method over FEM. Sharan [23] has used the popular multiquadric RBFs to solve elliptic PDEs. In a similar work, Sarler [24] formulated a solution method for diffusion problems based on RBFs. In a more general work, Wendlend [25] integrated the theory of Galerkin methods with radial basis functions.

More recently, Divo and Kassab [26, 27] have used RBFs to model convective viscous flows and heat transfer problems. Chinchapatnam [28] has used a localized RBF method to compute incompressible viscous flows. Radial basis methods for compressible flows are much less common, however Shu [29] has recently proposed such a method based on an upwind approach.

1.3.4 Finite Point Methods

By far, the most prevalent meshless schemes for CFD have been the so-called finite point methods. Finite point methods are usually based on the strong form of the governing PDEs and have given rise to several variants. In general, FPMs are based on least squares fitting of functions to discrete points. These approximate functions form the basis of discretization methods for PDEs. Least squares techniques have been widely used in traditional CFD methods as a means of reconstructing high order solutions, as discussed by Mavriplis [30]. However, the use of least squares as the primary mechanism for PDE discretization in the meshless sense is relatively new. Finite point methods were originally derived as generalizations of FDM for irregular point distributions by Chung [31] and Liszka [32].

Finite point methods may be categorized into two main classes: methods derived from Taylor series, and methods based on polynomial basis functions. Actually the Taylor series approach is a specific case of a polynomial method in which the approximated function is constrained to pass through the local cloud center. The Taylor approach is intuitive and has formed the basis for many schemes, including the Least Squares Kinetic Upwind Method (LSKUM) of Deshpande [33, 34]. Ghosh and Deshpande [35] and Ramesh and Deshpande [36] used LSKUM to compute compressible inviscid flows. Anandhanarayanan and Nagarathinam [37] parallelized the LSKUM method. Harish and Pavanakumar [38] simulated store separation dynamics using an approach based on LSKUM. Srinarayana [39] and Kumar [40] used kinetic approaches within optimization algorithms. Praveen [41, 42, 43], has formulated a similar kinetic approach called the Kinetic Meshless Method, which has given accurate results for two dimensional inviscid flows. Other approaches based on Taylor series expansions include that of Sridar and Balakrishnan [44, 45], in which they examined the order of accuracy of the Taylor method for an upwind scheme. A stabilized central scheme was developed by Jaisankar and others [46], in which they used a convective pressure splitting of the flux. Morinishi [47] evaluated the order of accuracy and conservation of a Taylor-based meshless solver. Balasubramanyam and Rao [48, 49] have developed a similar scheme.

The methods based on polynomial basis functions are equally numerous as the Taylor based methods. Batina [50] was one of the first to use a polynomial basis in conjunction with least squares to compute derivatives for the Euler and Navier-Stokes equations. He used an unweighted least squares approach. A similar method was proposed a few years later by Liu and Su [51]. Oñate and others [52, 53, 54] developed a more rigorous method based on polynomial basis functions. Their method incorporated different least squares weighting methods to improve the accuracy of derivatives and formulations for higher order methods. They applied their method to subsonic compressible inviscid and viscous flows. Löhner and others [55] extended the method of Oñate to compressible aerodynamic applications with shocks in three dimensions. They implemented their scheme with the van Leer approximate Riemann solver [56], gradient reconstruction for high resolution, and limiters to capture shocks.

1.3.5 Meshless Boundary Schemes

Many of the methods discussed above have been used to enforce boundary conditions for embedded boundary systems. Embedded boundaries arise with the use of non-body-conforming grids, such as Cartesian grids. Meshless methods have been used in place of cut cells and other related methods. Kirshman and Liu [57, 58, 59] have used a polynomial least squares method to compute inviscid slip boundary conditions using embedded Cartesian meshes. Koh, Tsai and Liu have implemented a similar method [60]. Luo, Baum and Lohner [61] implemented an embedded boundary scheme with a Taylor series weighted dual least squares approach. They presented encouraging results for two and three dimensional inviscid test cases. Kamatsuchi [62] has implemented meshless embedded boundary conditions for high Reynolds number viscous flows using the concept of a subgrid to resolve boundary layers. The subgrid adds additional resolution near the surface, providing points on which to perform meshless computations. Kamatsuchi applied the method to an entire formula 1 race car as a demonstration around complex geometry. All these methods appear to provide attractive alternatives to Cartesian cut cells or other methods of embedded boundary conditions

1.4 Scope of the Present Thesis

The above review of meshless methods in the literature indicates a wide variety of meshless schemes which have been applied to solve numerous problems in computational mechanics. The goal of the present thesis is to combine the best of existing meshless technology along with new formulations to develop a superior meshless algorithm for the solution of the Euler and Navier-Stokes equations. The meshless algorithm should be superior in that it satisfies the following three criteria as much as possible:

1. Give results which are at least as accurate and efficient as mesh-based solvers, such that it can be applied to practical problems of interest,

2. Minimize the sensitivity of the results to point and local cloud configuration, and

3. Be simple in implementation.

To date, a truly meshless method which satisfies these criteria simultaneously has proven elusive. In attempting to satisfy these criteria, comparisons will be made with existing methods and data when possible. The first requirement is essentially a validation effort. Throughout the thesis, all results obtained with any meshless method are compared to mesh based schemes, both qualitatively and quantitatively when possible. To obtain the most accurate results possible in a meshless framework, methods which have been developed over decades of structured and unstructured CFD research have been modified and tailored to meshless algorithms. The approach taken here is to use existing algorithms where possible to achieve superior accuracy, and to develop new technology to meet the needs specific to meshless schemes. The result is a highly accurate meshless scheme which corresponds well to finite volume results for difficult test cases, such as transonic and viscous flows.

Even more than accuracy, efficiency for meshless algorithms appears to be lacking. Even without the increase in computer speed, the last three decades of CFD algorithms and implementation research have reduced solution times by several orders of magnitude. Many of the algorithms developed for mesh-based schemes may be applied to meshless schemes with appropriate modifications. The most notable example of this in the present thesis is the concept of "multicloud", discussed in Chapter 4, which generalizes the concepts of multigrid to meshless schemes. In the process of formulating multicloud, it became apparent that multicloud was a powerful spin-off in and of itself, applicable to both meshless and mesh-based schemes in a mesh transparent fashion. Results showing dramatic convergence of both meshless and mesh based schemes using multicloud are given.

Requirement two alludes to the motivation for developing meshless schemes: alleviating the mesh generation bottleneck. To date, most of the meshless schemes have been tested on point distributions obtained from meshes. This is a valid form of testing a new method. For example, when unstructured CFD algorithms first appeared, they were implemented on triangular meshes derived from structured meshes [63, 64]. Later, more robust techniqes were developed to generate unstructured meshes of more general form. Much of the work in the present thesis is performed using point distributions and connectivity derived from meshes. This provides a convenient way to focus on the algorithm itself and validate results on similar point distributions using established methods. In addition, aspects of this work break new ground in terms of defining suitable point distributions. Specifically, the work on multicloud in Chapter 4 and the work dealing with meshless grid interfaces for overset grids in Chapter 5 require unique methods of defining local clouds of points. For highly random point distributions, such as those used in multicloud, RBF methods appear to be substantially more robust than least squares methods.

The final requirement of simplicity is addressed by using the strong form of the governing PDEs in a collocation approach instead of the FEM derived weak forms of other methods. While the use of the strong form is sometimes blamed for low accuracy and instability [4, 10], this work provides encouraging results to the contrary. Good convergence and a high degree of accuracy are obtained for a wide variety of test cases. The use of the strong form of the equations leads to a class of stable schemes based on a local extremum diminishing (LED) property. In Chapter 2, a variety of methods for obtaining derivatives on scattered data are shown to fit well into the LED framework. The result is a stable class of meshless schemes, which are as simple to implement as many unstructured finite volume schemes.

The simplicity of the meshless implementation developed here is shown to fit well within the framework of overset methods. A meshless method is used to update the flow at intergrid boundary points, coupling the patchwork grid system characteristic of overset methods. The meshless method offers many advantages over conventional interpolation for domain connectivity in overset approaches. Not only is the procedure to identify local clouds quite simple, all nodes in the domain are updated using the PDEs instead of a purely numerical procedure such as interpolation. The result is a seamless transition between grid components.

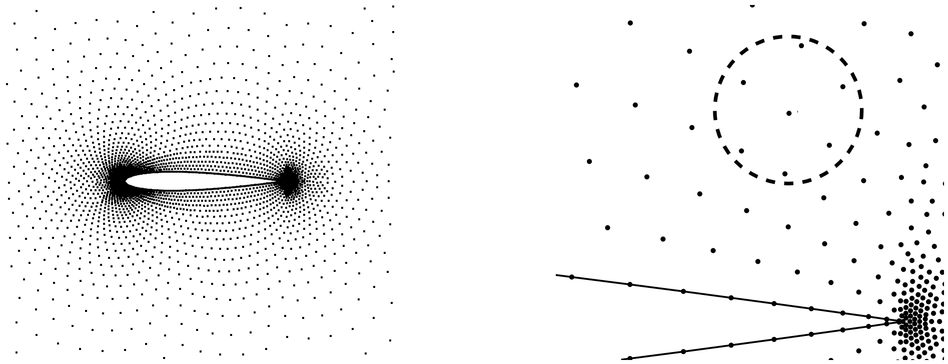
The outline of the thesis is as follows. Chapter 2 addresses the problem of obtaining partial derivatives on scattered data. Three existing methods are presented based on polynomials, Taylor series, and radial basis functions. These methods are shown to be suitable for LED based algorithms. Also, a new method for discretization, termed the “meshless volume” scheme, is derived. Chapter 3 applies the scattered derivative methods to the Euler and Navier-Stokes equations, focusing on the meshless volume scheme. An LED based framework is established for the meshless formulation. High resolution schemes with carefully constructed diffusion operators are formulated. An approach to the discretization of viscous terms appearing in the Navier-Stokes equations is presented. Issues dealing with boundary conditions and convergence acceleration are addressed. Chapter 4 involves the formulation of the multicloud algorithm, the meshless generalization of the well-known multigrid principles. Automatic coarsening procedures are developed to produce coarse level point distributions suitable for a meshless algorithm. Transfer operators are defined to transfer solution data between coarse and fine levels. Chapter 5 deals with an important application of the meshless scheme for overset grids. It is shown that a meshless scheme is ideally suited to perform the task of grid communication for domains consisting of various grid types. Finally, conclusions and future work are discussed in Chapter 6.

Chapter 2

Partial Derivatives on Scattered Data

While the inherent structure of meshes facilitates the discretization of PDEs, meshless methods demand a different approach. Meshless discretizations are based on clouds of points. A distinction should be make between global and local clouds of points. A global cloud of points contains all points in a given domain, while a local cloud of points is a small subset of the global cloud. It is on the local cloud of points that PDEs are discretized. The use of a local cloud enables compact support with small bandwidth of the resulting linear system instead of large non-sparse systems. Global and local clouds in two dimensions are illustrated in Figure 2.1.

Before describing the specifics of meshless derivative methods, it should be emphasized that obtaining both a global cloud of points and corresponding local clouds which are optimal for meshless methods is an ongoing area of research. As a minimum requirement, local clouds must be defined such that the matrices arising from local basis functions are well-conditioned and non-singular. Methods to determine local clouds are usually based on proximity in space search methods including quadtree/octree or bin type searches. However, other methods are possible, such as in the multicloud algorithm of Chapter 4, or the meshless interface method of Chapter 5. The methods discussed below attempt to improve the conditioning of the local matrices to improve derivative accuracy.



(a) Global point cloud.

(b) Local point cloud.

Figure 2.1: Meshless discretization framework.

While research into meshless methods is relatively new, scattered data interpolation has been addressed in the research for decades. In 1982, Franke [17] published an extensive review of nearly all scattered data interpolation techniques available at the time. His conclusion was that methods based on radial basis functions provided superior accuracy and convergence for use on a global set of scattered data. This conclusion has prompted interest in radial basis functions as a means to locally discretize PDEs. In addition, the finite element community has often advocated the use of polynomials using least squares as a basis for approximation. Often, some combination of both polynomials and radial basis functions is used.

This chapter establishes a common framework for both polynomial and radial basis schemes, as well as a special case of a polynomial scheme based on Taylor series expansion. Furthermore, a new method is presented, termed the "meshless volume" scheme, in which finite volume-like metrics are obtained from local clouds of points. The meshless volume scheme results in a much simpler and more efficient discretization over more traditional meshless solvers. All methods discussed here satisfy a key requirement of basis function approximation for PDEs: the methods are *reproducible*. That is, the methods described here can approximate all the functions contained in their basis exactly. The reproducibility of the approximations has important consequences in later chapters to formulate stable schemes for convection dominated problems.

The polynomial, Taylor, radial basis, and meshless volume schemes will be discussed, in that order. In will be shown that all the methods described here have a common formulation for partial derivatives of a function ϕ at node 0 of the form

$$\frac{\partial \phi}{\partial x} \approx \sum_i a_i(\phi_i - \phi_0), \quad \frac{\partial \phi}{\partial y} \approx \sum_i b_i(\phi_i - \phi_0), \tag{2.1}$$

where a_i and b_i are metric coefficients independent of ϕ. Here, the summation is over the nodes in local cloud of node 0. The metric weights depend purely on geometric considerations, and may be computed in a preprocess step for use throughout the computation for static problems. The chapter concludes by presenting a grid convergence study to characterize the discretization error of the meshless volume scheme.

2.1 Least Squares with a Polynomial Basis

Consider a local cloud of n scattered points surrounding a central point designated point 0, as shown in Figure 2.1. The goal of the polynomial least squares method is to obtain the gradient of relavent independent variables. Given a function $\phi(\boldsymbol{x})$, with $\boldsymbol{x} = (x, y)$ in two dimensions, known only at discrete values $\phi_i = \phi(\boldsymbol{x}_i)$ in the local cloud, we attempt to construct an approximation, $\hat{\phi}(\boldsymbol{x})$, using polynomials. The approximation takes the form

$$\hat{\phi}(\boldsymbol{x}) = \boldsymbol{p}(\boldsymbol{x})^T \boldsymbol{\alpha}, \tag{2.2}$$

where

$$\boldsymbol{p}^T = \begin{bmatrix} 1 & x & y & \cdots & p_m(\boldsymbol{x}) \end{bmatrix}$$

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_0 & \alpha_1 & \cdots & \alpha_m \end{bmatrix}$$

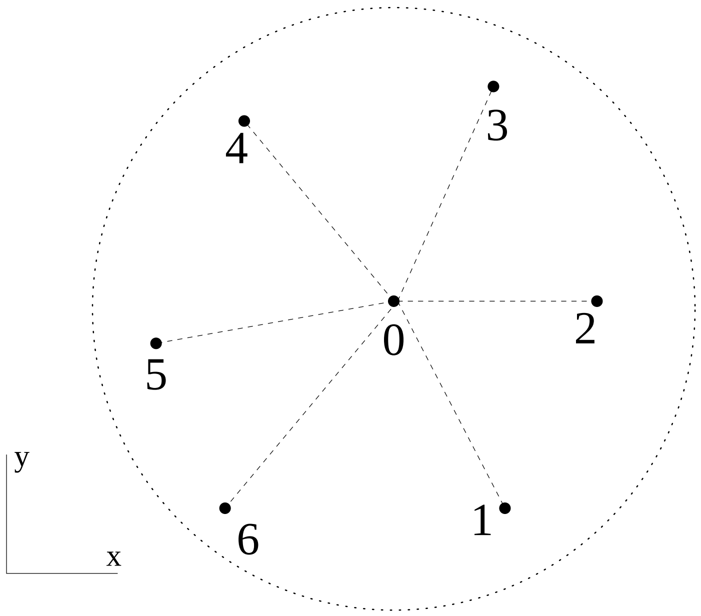


Figure 2.2: Local cloud of points surrounding point 0.

Here, $m \leq n+1$ is the number of polynomial basis functions used. The case of $n+1 = m$ exclusively has been used by some researchers including Liu [3] and Chung [31]. A far more common approach is to choose a basis of m monomials, then ensure $n > m-1$. As will be discussed, this leads to a non-square system of equations, leading to the use of least squares procedures. The polynomial basis should be complete up to a desired degree. Greater accuracy can be expected as more monomial terms are retained. However, more monomials means n must also increase, increasing the size of the local fitting procedure. While high order approximations are possible, the most common approach is to use a linear basis.

With the local cloud and basis defined, the unknown weight vector, $\boldsymbol{\alpha}$, may be determined from the following n equations:

$$\hat{\phi}(\boldsymbol{x}_i) = \boldsymbol{p}(\boldsymbol{x}_i)^T \boldsymbol{\alpha} = \phi_i, \quad i = 0, \ldots, n. \tag{2.3}$$

Because in general, Equation 2.3 is non-square, the method of weighted least squares

[65] is invoked in which we seek to minimize a function, $f(\boldsymbol{\alpha})$, where

$$f(\boldsymbol{\alpha}) = \sum_{i=0}^{n} w_i (\boldsymbol{p}(\boldsymbol{x}_i)^T \boldsymbol{\alpha} - \phi_i)^2 \tag{2.4}$$

with respect to the unknown coefficients, $\boldsymbol{\alpha}$. The weighting function, w_i is included to emphasize points which are closer to the cloud center at point 0. Generally, the weight function is chosen such that is attains a value of 1 at point 0, decreases monotonically with distance, and smoothly reaches a value of 0 outside the local cloud. A function which satisfies these criteria is the normalized Gaussian,

$$w_i = \frac{e^{-\left(\frac{r_i}{c}\right)^2} - e^{-\left(\frac{r_d}{c}\right)^2}}{1 - e^{-\left(\frac{r_d}{c}\right)^2}}, \tag{2.5}$$

where $r_i^2 = (x_i - x_1)^2 + (y_i - y_1)^2$ is the Euclidean distance from the cloud center, $r_d = (1 + \epsilon) r_{max}$ is a multiple of the maximum nodal distance in the cloud, and $c = \kappa r_d$. In practice, $\epsilon = 1$ and $\kappa = \frac{1}{2}$ have given the most accurate results. The normalized Gaussian function in two dimensions is shown in Figure 2.1.

The minimization of f in Equation 2.4 results in a square system of equations known as the normal equations. The normal equations may be expressed as

$$P^T W P \boldsymbol{\alpha} = P^T W \boldsymbol{\phi}, \tag{2.6}$$

where $\boldsymbol{\phi}$ is the vector of nodal values

$$\boldsymbol{\phi} = \begin{bmatrix} \phi_0 & \phi_1 & \cdots & \phi_n \end{bmatrix},$$

and P is the polynomial matrix and W is the diagaonal weight matrix. Both are defined as

$$P^T = \begin{bmatrix} \boldsymbol{p}(\boldsymbol{x}_1) & \boldsymbol{p}(\boldsymbol{x}_2) & \cdots & \boldsymbol{p}(\boldsymbol{x}_n) \end{bmatrix}, \quad \text{diag}(W) = \begin{bmatrix} w_0 & w_1 & \cdots & w_n \end{bmatrix}.$$

Provided the system in Equation 2.6 is non-singular, the coefficient vector $\boldsymbol{\alpha}$, may

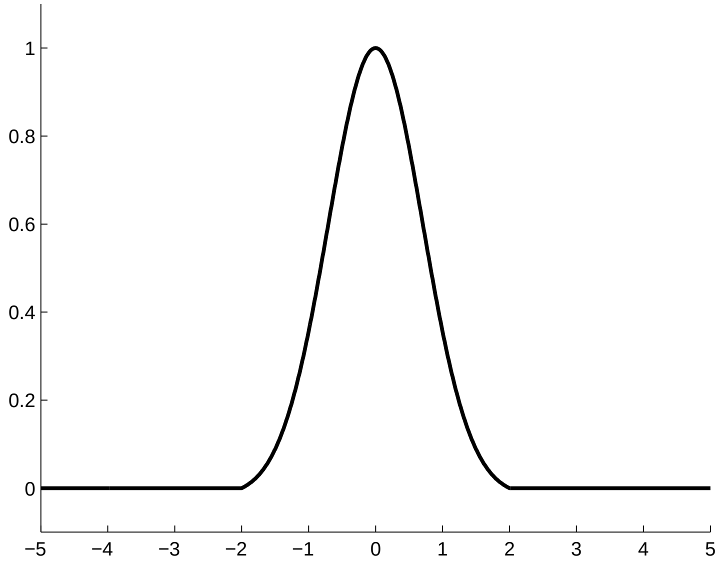


Figure 2.3: Normalized Gaussian function.

be obtained from

$$\boldsymbol{\alpha} = (P^T W P)^{-1} P^T W \boldsymbol{\phi} = C\boldsymbol{\phi}, \tag{2.7}$$

where C is the m by $n+1$ constant matrix

$$\mathcal{C} = \begin{bmatrix} c_{10} & c_{11} & \cdots & c_{1n} \\ c_{20} & c_{21} & \cdots & c_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{m0} & c_{m1} & \cdots & c_{mn} \end{bmatrix}. \tag{2.8}$$

It should be emphasized that the approximating function does not possess the so-called delta property which most finite element basis functions exhibit. That is, $\hat{\phi}(\boldsymbol{x}_i) \neq \phi(\boldsymbol{x}_i)$. Rather, we have merely fit a polynomial of desired degree through discrete data by minimizing the approximization error. From the one dimensional illustration in Figure 2.1, we see that in general, the approximating polynomial does not pass through any of the local cloud points, including point 0.

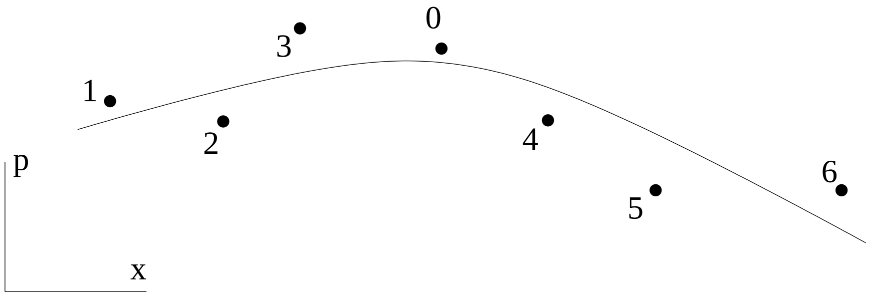


Figure 2.4: Local polynomial fit.

For the discretization of PDEs, we are most interested in partial derivatives of certain quantities. Partial derivatives at any point within the non-zero region covered by the weight function may be obtained directly from the approximation:

$$\frac{\partial \phi(\boldsymbol{x})}{\partial \boldsymbol{x}} \approx \frac{\partial \hat{\phi}(\boldsymbol{x})}{\partial \boldsymbol{x}} = \frac{\partial \boldsymbol{p}(\boldsymbol{x}_i)^T}{\partial \boldsymbol{x}} \boldsymbol{\alpha}. \tag{2.9}$$

For example, for a basis in two dimensions including linear terms, the partial first derivatives are constant:

$$\frac{\partial \phi(\boldsymbol{x})}{\partial x} \approx \frac{\partial \hat{\phi}(\boldsymbol{x})}{\partial x} = \alpha_2 = \sum_{i=0}^{n} c_{2i}\phi_i, \quad \frac{\partial \phi(\boldsymbol{x})}{\partial y} \approx \frac{\partial \hat{\phi}(\boldsymbol{x})}{\partial y} = \alpha_3 = \sum_{i=0}^{n} c_{3i}\phi_i. \tag{2.10}$$

Oñate [52] showed that the approximating function $\hat{\phi}$ can reproduce exactly any of the basis functions in $\boldsymbol{p}$. It follows that for the linear approximation of Equation 2.10,

$$\sum_{i=0}^{n} c_{2i} = 0, \quad \sum_{i=0}^{n} c_{3i} = 0. \tag{2.11}$$

Thus, without loss of generality, we may state

$$\frac{\partial \hat{\phi}(\boldsymbol{x})}{\partial x} = \sum_{i=1}^{n} c_{2i}(\phi_i - \phi_0), \quad \frac{\partial \hat{\phi}(\boldsymbol{x})}{\partial y} = \sum_{i=1}^{n} c_{3i}(\phi_i - \phi_0). \tag{2.12}$$

By setting $c_{2i} = a_i$ and $c_{3i} = b_i$, we obtain the common formulation described in Equation 2.1 since C is independent of ϕ. The forms in Equation 2.12 are the key to obtaining stable schemes for convection dominated partial differential equations as we will see in Chapter 3.

2.2 Least Squares Based on Taylor Series

Another way to obtain partial derivatives directly from scattered data is by way of least squares based on Taylor series. The Taylor method is closely related to the polynomial method. Without loss of generality assume a local origin, $(0,0)$, at point 0 in Figure 2.1. Consider the Taylor series of a function, ϕ, expanded around point 0, truncated after the linear terms:

$$\hat{\phi}(\boldsymbol{x}) = \phi_0 + x\frac{\partial\hat{\phi}(\boldsymbol{x_0})}{\partial x} + y\frac{\partial\hat{\phi}(\boldsymbol{x_0})}{\partial y}. \qquad (2.13)$$

In light of Equation 2.10, this is precisely the form of the polynomial approximation of Equation 2.2, with $\alpha_1 = \phi_0$, retaining only the linear terms. In other words, the Taylor series method is similar to the polynomial method, except by construction, the approximating polynomial passes through the cloud center, as shown in Figure 2.5 in one dimension.

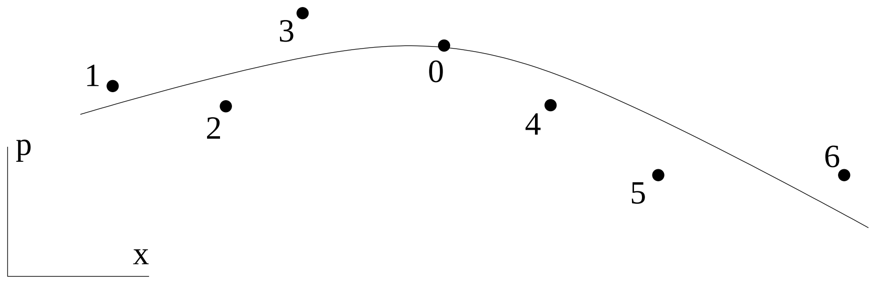


Figure 2.5: Illustration of Taylor series least squares approximation.

In general, we may expand the function $\hat{\phi}(\boldsymbol{x})$ around the local origin at point 0 with complete derivatives up to the m^{th} partial derivative as

$$\hat{\phi}(\boldsymbol{x}) = \phi_0 + \boldsymbol{d}(\boldsymbol{x})^T\boldsymbol{\phi}', \qquad (2.14)$$

with

$$\boldsymbol{d}(\boldsymbol{x})^T = \begin{bmatrix} x & y & \cdots & \frac{y^m}{m!} \end{bmatrix},$$

$$\boldsymbol{\phi}'^T = \begin{bmatrix} \frac{\partial\hat{\phi}(\boldsymbol{x_0})}{\partial x} & \frac{\partial\hat{\phi}(\boldsymbol{x_0})}{\partial y} & \cdots & \frac{\partial^m\hat{\phi}(\boldsymbol{x_0})}{\partial y^m} \end{bmatrix}.$$

Expanding Equation 2.14 to each of the n points in the local cloud of Figure 2.1, excluding the cloud center at point 0, we arrive at the following set of n equations:

$$\boldsymbol{d}(\boldsymbol{x_i})^T\boldsymbol{\phi'} = \phi_i - \phi_0 = \Delta\phi_{0i}, \quad i = 1, \cdots, n. \tag{2.15}$$

For a complete set of partial derivatives, the number of derivative terms in $\boldsymbol{\phi}'$ is $l = \frac{m(m+3)}{2}$. If $n = l$, Equation 2.15 may be solved uniquely without least squares. Otherwise, if $n \geq l$, an overdetermined system results, leading to least squares minimization procedures. In general, we seek to minimize a function, $f(\boldsymbol{\phi}')$, where

$$f(\boldsymbol{\phi}') = \sum_{i=1}^{n} w_i(\boldsymbol{d}(\boldsymbol{x}_i)^T\boldsymbol{\phi}' - \Delta\phi_{0i})^2, \tag{2.16}$$

with respect to $\boldsymbol{\phi}'$, where w_i is a weighting function, possibly the same as that in Equation 2.5.

A similar procedure may be invoked here as was done for the polynomial least squares method. Expressing Equation 2.16 in terms of the normal equations results in

$$D^TWD\boldsymbol{\phi'} = A\boldsymbol{\phi'} = D^TW\Delta\boldsymbol{\phi}, \tag{2.17}$$

where $\Delta\boldsymbol{\phi}$ is the vector of undivided differences,

$$\Delta\boldsymbol{\phi} = \begin{bmatrix} \Delta\phi_{01} & \Delta\phi_{02} & \cdots & \Delta\phi_{0n} \end{bmatrix},$$

and D is the metric matrix, W is the diagaonal weight matrix, and $A = D^TWD$ is the least squares matrix. Here, D and W are defined as

$$D^T = \begin{bmatrix} \boldsymbol{d}(\boldsymbol{x}_1) & \boldsymbol{d}(\boldsymbol{x}_2) & \cdots & \boldsymbol{d}(\boldsymbol{x}_n) \end{bmatrix}, \quad diag(W) = \begin{bmatrix} w_1 & w_2 & \cdots & w_n \end{bmatrix}.$$

Provided the system in Equation 2.17 is non-singular, the derivaties in $\boldsymbol{\phi}'$, may be obtained directly from

$$\boldsymbol{\phi}' = A^{-1}B^T W \Delta\boldsymbol{\phi} = G\Delta\boldsymbol{\phi}, \tag{2.18}$$

where G is a constant matrix

$$G = \begin{bmatrix} g_{11} & g_{12} & \cdots & g_{1n} \\ g_{21} & g_{22} & \cdots & g_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ g_{l1} & g_{l2} & \cdots & g_{ln} \end{bmatrix}. \tag{2.19}$$

For example, the first derivatives of $\hat{\phi}(\boldsymbol{x})$, at point 0 are

$$\frac{\partial\hat{\phi}(\boldsymbol{x_1})}{\partial x} = \sum_{i=2}^{n} g_{1i}\Delta\phi_{1i}, \quad \frac{\partial\hat{\phi}(\boldsymbol{x_1})}{\partial y} = \sum_{i=2}^{n} g_{2i}\Delta\phi_{1i}. \tag{2.20}$$

By setting $g_{1i} = a_i$ and $g_{2i} = b_i$, we obtain the common formulation described in Equation 2.1 since G is independent of ϕ. Note that this is of the same form as Equation 2.12, indicating the method is able to reproduce a constant field exactly. This is not obvious since we are not constructing an approximation to $\phi(\boldsymbol{x})$ directly. With the Taylor method, we solve only for the derivatives. However, the entire function approximation may be obtained by substituting the derivatives back into Equation 2.14.

Though both the polynomial and Taylor methods are quite similar, the Taylor method presents two slight advantages. First, the approximating function is constrained to pass through the cloud center at point 0. Second, the matrix size is one less in the Taylor approach than for the polynomial case. This is because the Taylor approach assumes $\alpha_0 = \phi_0$ in Equation 2.2. This may seem trivial, but for one important case. For a linear approximation, the Taylor method requires the inversion of a 2 by 2 matrix, leading to simple explicit formulas for the metric weights in Equation

2.20. The two dimensional linear coefficients are

$$g_{1i} = \frac{w_i x_i \sum wy^2 - w_i y_i \sum wxy}{\sum wx^2 \sum wy^2 - \left(\sum wxy\right)^2}, \quad g_{2i} = \frac{w_i y_i \sum wx^2 - w_i x_i \sum wxy}{\sum wx^2 \sum wy^2 - \left(\sum wxy\right)^2}, \tag{2.21}$$

where the summations are $i = 1, \ldots, n$. The ability to explicity manipulate the metric terms for the linear Taylor based approach leads to the meshless volume scheme discussed in a subsequent section.

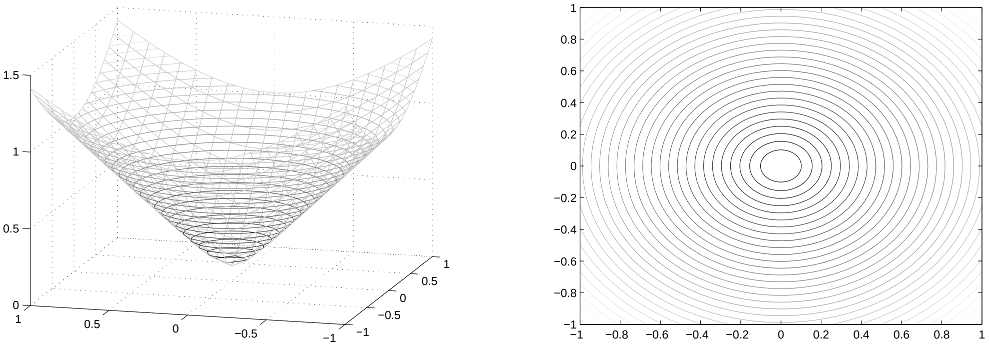
2.3 Local Radial Basis Method

Unlike the Taylor and polynomial methods of the previous sections, the local radial basis method does not rely on least squares to obtain partial derivatives from local scattered data. Instead, the method seeks a function which passes through all the data points in a local cloud. This leads to the so-called delta property in which the approximating function is equal to the actual discrete values at the nodes. The Taylor method only possesses the delta property at the node center, while the polynomial method possesses no delta property at all in general. If constructed carefully, radial basis function approximations can reproduce constant and linear fields, as will be discussed.

Many types of radial basis functions have been developed over the years. Any function which satisfies

$$\theta(\boldsymbol{x}) = \theta(||\boldsymbol{x}||)$$

may be considered a radial basis function. Usually the norm used is simply the Euclidean distance. Commonly used radial basis functions include Gaussians, thin plate splines, logarithmics, and multiquadrics. Franke [17] concluded that multiquadrics converged most rapidly and produced the least error of nearly all radial basis functions when used for global scattered data interpolation. This conclusion has led to the widespread use of radial basis functions for the discretization of PDEs [18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29]. The use of the multiquadric radial basis



(a) Surface plot.

(b) Contour plot.

Figure 2.6: Multiquadric radial basis function.

function is also used here and may be expressed as

$$\theta_i(\boldsymbol{x}) = \sqrt{(x - x_i)^2 + (y - y_i)^2 + c^2}, \tag{2.22}$$

where the basis function is centered at $\boldsymbol{x}_i$, and c is a free parameter. Unfortunately, there exists no analysis to guide in the selection of c. Usually it is empirically determined based on accuracy and convergence considerations. Surface and contour plots of the multiquadric function of Equation 2.22 are shown in Figure 2.6.

Construction of a radial basis approximating function, $\hat{\phi}(\boldsymbol{x})$, which approximates the function, $\phi(\boldsymbol{x})$, known only at the discrete values, $\phi_i = \phi(\boldsymbol{x}_i)$, shown in Figure 2.1, is now described. The approximating function takes the form

$$\hat{\phi}(\boldsymbol{x}) = \boldsymbol{t}^T(\boldsymbol{x})\boldsymbol{\lambda} + \boldsymbol{p}^T(\boldsymbol{x})\boldsymbol{\alpha}, \tag{2.23}$$

where $\boldsymbol{t}$ and $\boldsymbol{p}$ are the vectors of radial basis functions and optionally appended polynomials respectively:

$$\boldsymbol{t}^T(\boldsymbol{x}) = \begin{bmatrix} \theta_0(\boldsymbol{x}) & \theta_1(\boldsymbol{x}) & \cdots & \theta_n(\boldsymbol{x}) \end{bmatrix},$$

$$\boldsymbol{p}^T(\boldsymbol{x}) = \begin{bmatrix} p_1(\boldsymbol{x}) & p_2(\boldsymbol{x}) & \cdots & p_m(\boldsymbol{x}) \end{bmatrix}.$$

The appended polynomials are necessary in some cases to ensure a non-singular approximation, and improve consistency. For the multiquadric function, Schaback and Wendlend [66] showed at least a linear polynomial should be added to the multiquadric approximation to ensure an invertible system. In addition, the effect of adding a linear polynomial to acheive consistency may be seen by expanding the one-dimensional multiquadric function in a Taylor series:

$$\sqrt{c^2+x^2} = c\sqrt{1+\tilde{x}^2} = c\left[1+\frac{\tilde{x}^2}{2}+\frac{\tilde{x}^4}{8}+\frac{\tilde{x}^6}{16}+\cdots\right]. \tag{2.24}$$

It is clear that the multiquadric basis function omits all odd powers of the independent variable, including the linear terms. Therefore, adding a linear polynomial to the approximation of Equation 2.23 enables linear consistency and the reproduction of a linear function exactly.

The unknown weights, $\boldsymbol{\lambda}$ and $\boldsymbol{\alpha}$, may be obtained by invoking the following $n+1$ equations and m constraints:

$$\hat{\phi}(\boldsymbol{x}_i) = \phi_i, \quad i = 0, \ldots, n, \qquad \sum_{i=1}^{n} \lambda_i p_j(\boldsymbol{x}_i) = 0, \quad j = 1, \ldots, m. \tag{2.25}$$

The equations in 2.25 may be recast as

$$\begin{bmatrix} T & P^T \\ P & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{\lambda} \\ \boldsymbol{\alpha} \end{Bmatrix} = \begin{Bmatrix} \boldsymbol{\phi} \\ \mathbf{0} \end{Bmatrix}, \tag{2.26}$$

where the submatrices, T and P, are defined as

$$T^T = \begin{bmatrix} \boldsymbol{t}(\boldsymbol{x}_0) & \boldsymbol{t}(\boldsymbol{x}_1) & \cdots & \boldsymbol{t}(\boldsymbol{x}_n) \end{bmatrix},$$

$$P = \begin{bmatrix} \boldsymbol{p}(\boldsymbol{x}_0) & \boldsymbol{p}(\boldsymbol{x}_1) & \cdots & \boldsymbol{p}(\boldsymbol{x}_n) \end{bmatrix},$$

and the vector of nodal values, $\boldsymbol{\phi}$, is defined as

$$\boldsymbol{\phi}^T = \begin{bmatrix} \phi_0 & \phi_1 & \cdots & \phi_n \end{bmatrix}.$$

Writing the inverse of the symmetric matrix in Equation 2.26 as

$$\begin{bmatrix} T & P^T \\ P & 0 \end{bmatrix}^{-1} = \begin{bmatrix} A & B^T \\ B & C \end{bmatrix}, \tag{2.27}$$

the approximation of Equation 2.23 becomes

$$\hat{\phi}(\boldsymbol{x}) = \left(\boldsymbol{t}^T(\boldsymbol{x})A + \boldsymbol{p}^T(\boldsymbol{x})B\right)\boldsymbol{\phi}. \tag{2.28}$$

Once again, since we are interested in derivatives for PDE discretization, we seek the derivatives of $\hat{\phi}$, which may be obtained from

$$\frac{\partial \hat{\phi}(\boldsymbol{x}_0)}{\partial x} = \left(\frac{\partial \boldsymbol{t}^T(\boldsymbol{x}_0)}{\partial x}A + \frac{\partial \boldsymbol{p}^T(\boldsymbol{x}_0)}{\partial x}B\right)\boldsymbol{\phi} = \boldsymbol{b}_x^T\boldsymbol{\phi},$$

$$\frac{\partial \hat{\phi}(\boldsymbol{x}_0)}{\partial y} = \left(\frac{\partial \boldsymbol{t}^T(\boldsymbol{x}_0)}{\partial y}A + \frac{\partial \boldsymbol{p}^T(\boldsymbol{x}_0)}{\partial y}B\right)\boldsymbol{\phi} = \boldsymbol{b}_y^T\boldsymbol{\phi},$$

where $\boldsymbol{b}_x$ and $\boldsymbol{b}_y$ are the vectors of the derivative metric terms. Using the linear reproducibility property, the derivatives may be expressed as

$$\frac{\partial \hat{\phi}(\boldsymbol{x}_0)}{\partial x} = \sum_{i=0}^{n} b_{xi}\phi_i = \sum_{i=1}^{n} b_{xi}\Delta\phi_{0i}, \quad \frac{\partial \hat{\phi}(\boldsymbol{x}_0)}{\partial y} = \sum_{i=0}^{n} b_{yi}\phi_i = \sum_{i=1}^{n} b_{yi}\Delta\phi_{0i}, \tag{2.29}$$

where $\Delta\phi_{0i} = \phi_i - \phi_0$ as before. By setting $b_{xi} = a_i$ and $b_{yi} = b_i$, we obtain the common formulation described in Equation 2.1 since $\boldsymbol{b}_x$ and $\boldsymbol{b}_y$ are independent of ϕ.

Thus, we arrive at the same form for the derivatives as the polynomial and Taylor least squares methods. The use of radial basis functions to approximate field variables on scattered data generally requires the inversion of larger matrices than the least squares methods. The size of the matrix in Equation 2.27 is $(n+m+1)$ by $(n+m+1)$, where n is the number of points in the local cloud at point 0, and m is the number of added monomials. The increased computational cost comes with the benefit of increased stability and robustness. Radial basis methods are not as sensitive to local node movement and position. In addition, the matrix of Equation 2.27 is always invertible provided the right polynomial basis terms are added. However, accuracy

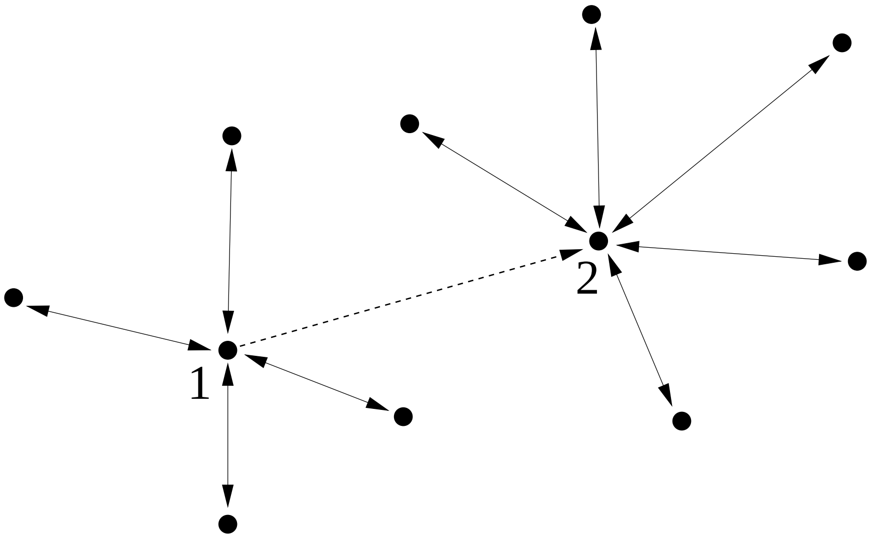


Figure 2.7: Directed edge which breaks reciprocity.

appears to suffer in the presence of discontinuities, as will be discussed in Chapter 3.

2.4 A Meshless Volume Scheme

All three of the discretization schemes above are entirely local, only considering points belonging to a local cloud. While the local nature of these procedures is necessary for the discretization of partial differential equations, local meshless discretizations present drawbacks. The most obvious is that reciprocity of nodes is not guaranteed, as shown in Figure 2.4. While node 2 may belong in the local cloud of node 1, node 1 does not necessarily belong in the local cloud of node 2. Thus, a sort of directional influence may be established in which flux distribution is one-way. This corresponds to the presence of directed edges in graph theory terms [67]. This may be circumvented in part by describing local clouds implicitly by means of an edge vertex list. By considering edges, meshless algorithms may be formulated much like edge-based finite volume algorithms [68, 69].

While the use of an edge-vertex list guarantees reciprocity of nodes, fundamental metric computation is still local. The result is that the metrics, such as those of Equation 2.1, are not equal and opposite for nodes connected by an edge. For example, suppose that the directional edge in Figure 2.4 was actually non-directional and reciprocity was attained. Points 1 and 2 each have a local cloud defined. Performing any of the aforementioned scattered data techniques would result in metric terms (a_{12}, b_{12}) and (a_{21}, b_{21}), where the first number in the subscript indicates the cloud center, and the second number indicates a node in the local cloud. Thus for each edge, there would be four unique metric terms, since unlike conservative finite volume schemes, $a_{12} \neq a_{21}$ and $b_{12} \neq b_{21}$. This results in double metric storage and double flux computation compared with finite volume schemes, which compute a single directed flux to be distributed to the endpoints of an edge with a single edge based metric. By construction, such finite volume schemes are conservative in that the sum of all flux contributions reduces to the flux through the domain boundaries, since interior fluxes cancel. No meshless scheme to date can be shown to be conservative in this way. This presents a theoretical drawback for meshless schemes, since it would be desirable for a scheme which approximates a conservation law to also be conservative. Formal conservation for a discrete scheme is often considered a prerequisite to correctly capturing shock waves and other flow features.

In this section, a method which addresses many of the aforementioned issues is developed. While still not formally conservative, the method described below allows for a construction similar to finite volume schemes, increasing efficiency and streamlining memory usage. The main goal of the present analysis is to arrive at edge based metrics which are equal and opposite for nodes sharing an edge. Because of its similarity to finite volume schemes, the new meshless discretization will be termed the “meshless volume” scheme.

2.4.1 Similarities between the Taylor Least Squares Method and Finite Volume Methods

The meshless volume scheme may be constructed by noting similarities between finite volume formulations and the strong form meshless schemes, which may be considered generalized finite difference schemes. Consider the semi-discretization of a scalar

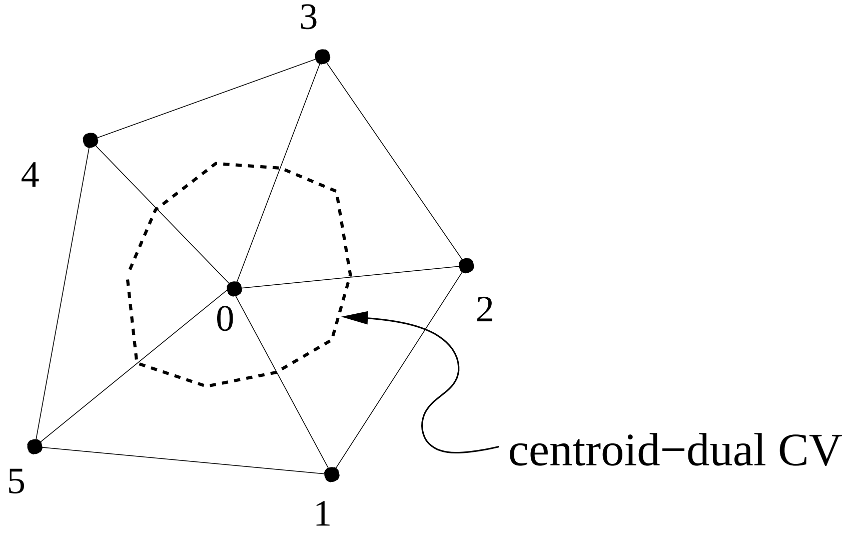


Figure 2.8: Centroid dual control volume on a triangular mesh.

conservation law for independent variable, v, based on an integral form over the finite volume at node 0,

$$V_0 \frac{dv_0}{dt} + R_0(\boldsymbol{v}, \boldsymbol{a}) = 0, \tag{2.30}$$

where V_0 is the volume, v_0 is the volume averaged conserved variable in the volume 0, and R_0 is the residual. Such a configuration is illustrated in Figure 2.4.1 for the well-konwn centroid-dual control volume [70]. Here, the residual at 0 is a function of the surrounding conserved variables as well as the directed face area metrics, $\boldsymbol{a} = \begin{bmatrix} \boldsymbol{a}_1 & \boldsymbol{a}_2 & \dots & \boldsymbol{a}_n \end{bmatrix}$, forming V_0. As shown in Figure 2.4.1, n nodes surround node 0 in the triangulation.

The finite volume form of Equation 2.30 is slightly different than that of the meshless schemes described in previous sections. Since the meshless schemes compute partial derivatives directly, the semi-discretization of the meshless schemes may be cast into a generalized finite difference form,

$$\frac{dv_0}{dt} + \tilde{R}_0(\boldsymbol{v}, \tilde{\boldsymbol{a}}) = 0, \tag{2.31}$$

where v_0 is interpreted as the nodal conserved variable at node 0, and the residual, $\tilde{R}_0$, is a function of metric terms, $\tilde{\boldsymbol{a}} = \begin{bmatrix} \tilde{\boldsymbol{a}}_1 & \tilde{\boldsymbol{a}}_2 & \dots & \tilde{\boldsymbol{a}}_n \end{bmatrix}$. The metrics, $\tilde{\boldsymbol{a}}$, have dimension $(1/h)$, while the face areas, $\boldsymbol{a}$, have dimension of (V/h), where h is the mesh spacing.

By restricting the analysis to the linear Taylor least squares method described above, it is possible to cast the meshless discretization of Equation 2.31 into the finite volume form of Equation 2.30 by simply multiplying the equation by a meshless volume, V_0. Just what the meshless volume, V_0, represents in the context of a meshless method will be discussed and quantified later. For now, it is only necessary that the volume, V_0, be a circle (in two dimensions) of some radius, r_0, centered at (x_0, y_0), as shown in Figure 2.4.1. It follows, then, that the nodal value, v_0, may be interpreted as a volume average over the circular volume, V_0. This is easily verified for the Taylor approach with linear terms:

$$\begin{aligned} \int_{V_0} v(x,y)dV &= \iint (v_0 + xv_x + yv_y)\, dxdy \\ &= \int_0^{r_0} \int_0^{2\pi} (v_0 + r\cos(\theta)v_x + r\sin(\theta)v_y)\, r d\theta dr \\ &= \int_0^{r_0} 2\pi r v_0 dr \\ &= \pi r_0^2 v_0 = V_0 v_0. \end{aligned}$$

Thus, for static mesh problems, a simple multiplication of the meshless discretization by a circular volume, V_0, yields a finite volume-like form,

$$\begin{aligned} V_0\left(\frac{dv_0}{dt} + \tilde{R}_0(\boldsymbol{v}, \tilde{\boldsymbol{a}})\right) &= V_0\frac{dv_0}{dt} + V_0\tilde{R}_0(\boldsymbol{v}, \tilde{\boldsymbol{a}}) \\ &= V_0\frac{dv_0}{dt} + R_0(\boldsymbol{v}, V_0\tilde{\boldsymbol{a}}) \\ &= V_0\frac{dv_0}{dt} + R_0(\boldsymbol{v}, \boldsymbol{a}), \end{aligned}$$

since R_0 is a linear function of the face areas, $a = V_0\tilde{a}$. In other words, multiplying the gneralized finite difference form by a meshless volume, yields a finite volume-like formulation for the linear Taylor based approach.

Technically, the “face areas,” $\boldsymbol{a}$, formed by multiplying the least squares metrics,

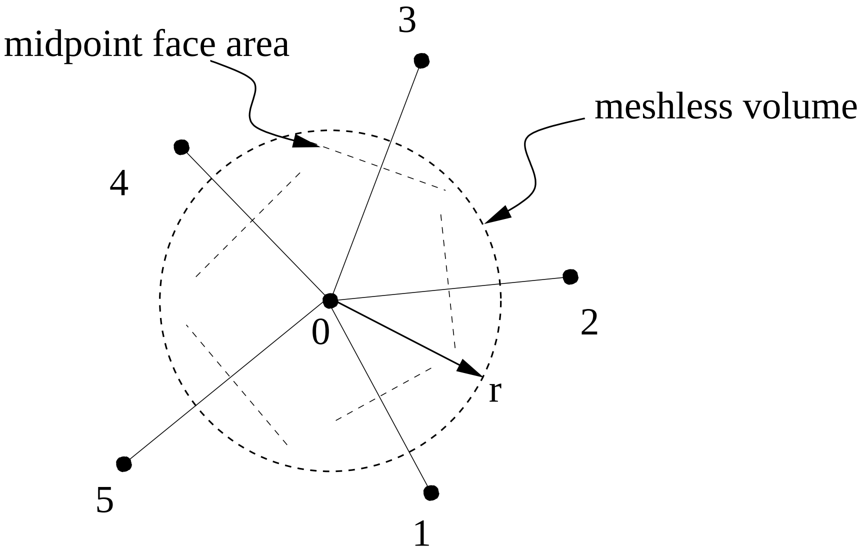


Figure 2.9: Meshless volume showing midpoint face areas.

$\tilde{a}$, by the meshless volume, are not faces of a control volume. In general, their sum is not zero as in the case of a closed, faceted volume. Also, the face areas corresponding to the two nodes shared by an edge are still not equal. This means that adjacent meshless volumes do not share a common face area, and little progress has been made. However, the concept of a meshless volume scheme is important in formulating certain constraints on weights, which enable equal face areas for adjacent meshless volumes.

2.4.2 Weight Constraints

In order to obtain equal face areas for adjacent nodes, similar to a finite volume scheme, it is important to understand the nature of the metric coefficients in Equation 2.21 arising from the Taylor-based least squares procedure. As pointed out by Praveen [42], the least squares matrix, A, in Equation 2.17 has interesting properties. Most importantly in this context, if the local constraints,

$$\sum w_i x_i^2 = \sum w_i y_i^2, \quad \sum w_i x_i y_i = 0 \tag{2.32}$$

are satisfied, then the two eigenvalues of the least squares matrix are equal with a value of $\lambda_0 = \sum_i wx^2 = \sum_i wy^2$. The summations appearing in Equation 2.32 are over the nodes in the local cloud for node 0. Furthermore, in such a case, the least squares metric coefficents of Equation 2.21 take on a particularly simple and useful form:

$$g_{1i} = \frac{w_i x_i}{\lambda_0}, \quad g_{2i} = \frac{w_i y_i}{\lambda_0}, \tag{2.33}$$

for every node, i, in the cloud of node 0. Note that the coefficients in Equation 2.33 define a vector which aligns with the edge itself. Thus, two nodes sharing an edge will have metrics which are parallel, but not necessarily equal in magnitude. This is an important step in establishing common face areas in the meshless volume scheme.

In general, the constraints of Equation 2.32 are not satisfied except in simple configurations, such as uniform Cartesian spacing. However, the constraints can be satisfied on arbitrary point distributions as part of a local weight optimization procedure. In the optimization problem, a minimum of $f(\boldsymbol{w})$ is sought, subject to the constraints $h_1(\boldsymbol{w})$ and $h_2(\boldsymbol{w})$, where

$$f(\boldsymbol{w}) = \sum_i (w_i - \tilde{w}_i)^2, \quad h_1(\boldsymbol{w}) = \sum_i w_i(x_i^2 - y_i^2), \quad h_2(\boldsymbol{w}) = \sum_i w_i x_i y_i, \tag{2.34}$$

where $\boldsymbol{w}^T = \begin{bmatrix} w_1 & w_2 & \cdots & w_n \end{bmatrix}$ is the vector of weights for which we seek corresponding to each node in the local cloud, and $\tilde{\boldsymbol{w}}^T = \begin{bmatrix} \tilde{w}_1 & \tilde{w}_2 & \cdots & \tilde{w}_n \end{bmatrix}$ are the corresponding target weights obtained from any conventional weighting procedure, such as that of Equation 2.5. The method of Lagrange multipliers may be invoked here, yielding the following system of equations at each node:

$$\begin{bmatrix} I & U^T \\ U & \mathbf{0} \end{bmatrix} \begin{Bmatrix} \boldsymbol{w} \\ \boldsymbol{l} \end{Bmatrix} = \begin{Bmatrix} \tilde{\boldsymbol{w}} \\ \mathbf{0} \end{Bmatrix}, \tag{2.35}$$

with

$$U = \begin{bmatrix} x_1^2 - y_1^2 & x_2^2 - y_2^2 & \cdots & x_n^2 - y_n^2 \\ x_1 y_1 & x_2 y_2 & \cdots & x_n y_n \end{bmatrix},$$

and $\boldsymbol{l} = (l_1, l_2)$ is the vector of Lagrange multipliers. Since this system of equations is solved on a local node by node basis, matrix sizes are kept small and efficiency is maintained.

In practice, it has been observed that the optimized weight vector, $\boldsymbol{w}$, does not differ much from the optimal weights, $\tilde{\boldsymbol{w}}$. In other words, for reasonably even point distributions, the least squares metric coefficients are nearly parallel with their edges anyway, so invoking the optimization procedure amounts to merely tweaking the weights in order to satisfy the constraints.

2.4.3 Volume Definition and Reciprocal Face Areas

With the least squares coefficients aligned with their edges, it is now possible to define shared face area metrics for adjacent meshless volumes. As pointed out by Praveen [42], the eigenvalues of the least squares matrix actually represent the semi-major and semi-minor axes of an ellipse which bounds the local cloud for a given node. Since the constraints of Equation 2.32 equalize the eigenvalues, the resulting eigenvalue may be used as the basis for the circular meshless volume definition. If the radius of the volume at node 0 is $r_0 = \sqrt{\lambda_0/\pi}$, then the meshless volume at each node becomes $V_0 = \lambda_0$.

Now consider performing the least squares Taylor expansion procedure to the midpoint of each edge in a local cloud as shown in Figure 2.4.1. For an edge, k, with endpoints i and j, the least squares metric coefficients for node j in the cloud of node i become

$$\tilde{a}_{ij} = 2\frac{w_{ij}\Delta x_{ij}}{\sum_j w_{ij}\Delta x_{ij}^2}, \quad \tilde{b}_{ij} = 2\frac{w_{ij}\Delta y_{ij}}{\sum_j w_{ij}\Delta y_{ij}^2}, \tag{2.36}$$

assuming the weights, w_{ij}, have been computed using the constrained optimization procedure. Here, the factor of two is needed since the expansion is only to the midpoint of the edge. The notation, w_{ij}, denotes the weight that node j is assigned in the cloud of node i, and $\Delta x_{ij} = x_j - x_i$.

While the midpoint was chosen as the Taylor expansion point, potentially any point between nodes i and j on edge k could have been used. Therefore, a parameter, α_k, may be defined for the edge such that $0 < \alpha_k < 1$, which allows the expansion point to move freely, as shown in Figure 2.10. Now the expansion may be taken to a

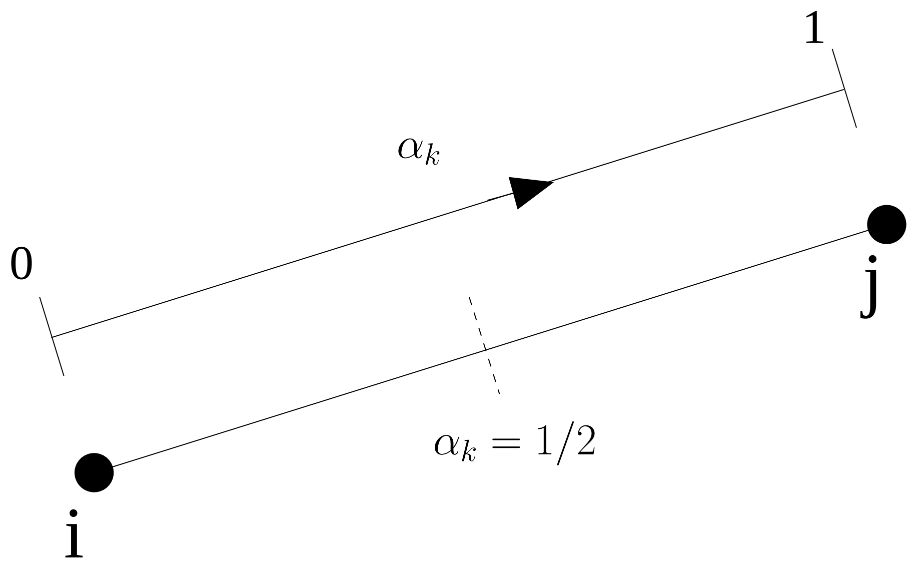


Figure 2.10: Illustration of the sliding parameter, α_k.

point $\boldsymbol{x}_k = \boldsymbol{x}_i + \alpha_k \Delta \boldsymbol{x}_{ij} = \boldsymbol{x}_j - (1-\alpha_k)\Delta \boldsymbol{x}_{ij}$. It can be verified easily that a value of $\alpha_k = 1/2$ reverts the expansion to the midpoint. It is now reasonable to ask whether a value of α_k exists such that a common face area may be computed for adjacent nodes i and j. Mathematically, this translates to enforcing the constraint

$$V_i \tilde{a}_{ik} = -V_j \tilde{a}_{jk}, \tag{2.37}$$

where

$$\tilde{a}_{ik} = \frac{w_{ik}(x_k - x_i)}{\sum_k w_{ik}(x_k - x_i)^2}, \quad \tilde{a}_{jk} = \frac{w_{jk}(x_k - x_j)}{\sum_k w_{jk}(x_k - x_j)^2}$$

are valid least squares formulas if w_{ik} and w_{jk} are chosen to satisfy the constrained optimization problem of Equation 2.34. A similar constraint on the y-coefficients is not needed since if the coefficients are parallel to the edge, the y-constraint, $V_i \tilde{b}_{ik} = -V_j \tilde{b}_{jk}$, is automatically satisfied.

To assist in the selection of α_k such that Equation 2.37 is satisfied, both the target weights of Equation 2.34 and the computed optimal weights may be assumed to vary inversely with the square of the distance from a given node. This is a reasonable weighting scheme, since it emphasizes nodes which are close in proximity to a given node. With this weight selection it follows that

$$w_{ik} = \frac{w_{ij}}{\alpha_k^2}, \quad w_{jk} = \frac{w_{ji}}{(1-\alpha_k)^2}. \tag{2.38}$$

Somewhat surprisingly, the eigenvalues of the least squares matrix, as well as the constraints in Equation 2.34 remain unchanged as α_k varies along each edge under the inverse square weighting scheme:

$$\begin{aligned} \lambda_i &= \sum_k w_{ij}\Delta x_{ij}^2 \\ &= \sum_k \frac{w_{ij}}{\alpha_k^2}(\alpha_k \Delta x_{ij})^2 \\ &= \sum_k w_{ik}(x_k - x_i)^2 \end{aligned}$$

In other words, λ_i, and thus the volume, V_i, remain unchanged as α_k changes. Since λ_i and λ_j are independent of α_k, we may select any $\boldsymbol{x}_k$ to which to perform the expansion without changing the eigenvalues, λ_i and λ_j. Additionally, since the constraints of Equation 2.34 remain satisfied, the simplified least squares formulas of Equation 2.36 remain valid for any $0 < \alpha_k < 1$.

It follows that Equation 2.37 will be satisfied with

$$\alpha_k = \frac{w_{ij}}{w_{ij} + w_{ji}}. \tag{2.39}$$

when $V_i = \lambda_i$ and $V_j = \lambda_j$. However, this choice of α_k is equivalent to defining a single average edge weight, $w_k = w_{ij} + w_{ji}$, and performing the expansion to the edge midpoint. This form is desirable since approximate Riemann solvers often solve for midpoint fluxes.

In summary, the least squares coefficients corresponding to node j in the cloud of node i may be expressed as

$$\tilde{a}_{ij} = \frac{w_k \Delta x_{ij}}{\lambda_i}, \quad \tilde{b}_{ij} = \frac{w_k \Delta y_{ij}}{\lambda_i}. \tag{2.40}$$

Similarly,

$$\tilde{a}_{ji} = \frac{w_k \Delta x_{ji}}{\lambda_j}, \quad \tilde{b}_{ji} = \frac{w_k \Delta y_{ji}}{\lambda_j}. \tag{2.41}$$

When multiplied by their respective meshless volumes, these coefficients represent a common face area shared by nodes i and j:

$$a_{ij} = w_k \Delta x_{ij}, \quad b_{ij} = w_k \Delta y_{ij}, \quad a_{ji} = w_k \Delta x_{ji}, \quad b_{ji} = w_k \Delta y_{ji}. \tag{2.42}$$

Thus the face area for an edge is completely defined by the average weight, w_k, which is shared by both nodes which share the edge.

2.4.4 Charcterization of Discretization Error

In order to characterize the discretization error of the meshless volume derivative method, a grid convergence study was performed. The spatial accuracy of the method was examined by comparing the results with an exact solution on a series of increasingly dense point distributions. In addition, the meshless volume scheme was compared with mesh based methods of computing derivatives. The methods and results of this study are presented in this section.

In order to verify the accuracy of the meshless volume scheme, four types of point distributions were generated, as shown in Figure 2.11. Figure 2.11(a) shows a simple Cartesian distribution of points, which served as the first distribution. Second, a perturbed distribution was generated by randomly perturbing the Cartesian system a small distance from the regular distribution, as shown in Figure 2.11(b). Third, a regular triangular mesh was generated, as shown in Figure 2.11(c). Finally, a similar perturbation procedure was applied to the regular triangular mesh to form the irregular distribution of Figure 2.11(d). When the meshless volume scheme was tested, the mesh lines of each mesh were used to define local clouds of points.

A simple trigonometric function with simple partial derivatives was used as a basis for comparison of the discrete schemes. The function and its derivates were

$$
\begin{aligned}
f(x,y) &= \sin x \sin y, \\
\frac{\partial f}{\partial x} &= \cos x \sin y, \\
\frac{\partial f}{\partial y} &= \sin x \cos y.
\end{aligned}
$$

The meshless volume scheme was used to compute discrete approximations to these derivatives on all four point distributions described. In addition, three other mesh based methods were implemented for comparison. On the structured grid data of Figures 2.11(a)-2.11(b), two methods were implemented in addition to the meshless volume scheme. These methods were based on a finite difference approxmation and a finite volume approximation based on the divergence theorem. The finite difference method was performed using a discrete mapping to (ξ, η) coordinates as

$$
\begin{aligned}
\frac{\partial f}{\partial x} &= \frac{\partial f}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial f}{\partial \eta}\frac{\partial \eta}{\partial x}, \\
\frac{\partial f}{\partial y} &= \frac{\partial f}{\partial \xi}\frac{\partial \xi}{\partial y} + \frac{\partial f}{\partial \eta}\frac{\partial \eta}{\partial y}.
\end{aligned}
$$

Central differences were performed both to compute the derivatives of f and the mapped coordinates. The finite volume method was implemented by invoking the divergence theorem consistent with a linear approximation:

$$
\begin{aligned}
\frac{\partial f}{\partial x} &= \frac{1}{V}\int_{\partial V} f dy, \\
\frac{\partial f}{\partial y} &= -\frac{1}{V}\int_{\partial V} f dx.
\end{aligned}
$$

Finite volumes were taken to be the volumes formed by the primary mesh, and the surface integral was evaluated by averaging the function values on either side of the volume faces and multiplying by the proper face projection.

To compare the meshless volume scheme to a mesh based scheme on triangles, a nodal finite volume method was used, again based on the divergence theorem, but with a trapezoidal integration rule. Nodal finite volumes were defined to be the union of all triangles sharing a common node, similar to the unstructured scheme of Jameson [71]. The discrete scheme may be described as

$$\begin{aligned}\frac{\partial f}{\partial x} &\approx \frac{1}{2V}\sum_k (f_k + f_{k-1})(y_k - y_{k-1}),\\ \frac{\partial f}{\partial y} &\approx -\frac{1}{2V}\sum_k (f_k + f_{k-1})(x_k - x_{k-1}),\end{aligned}$$

where the nodes are ordered counterclockwise.

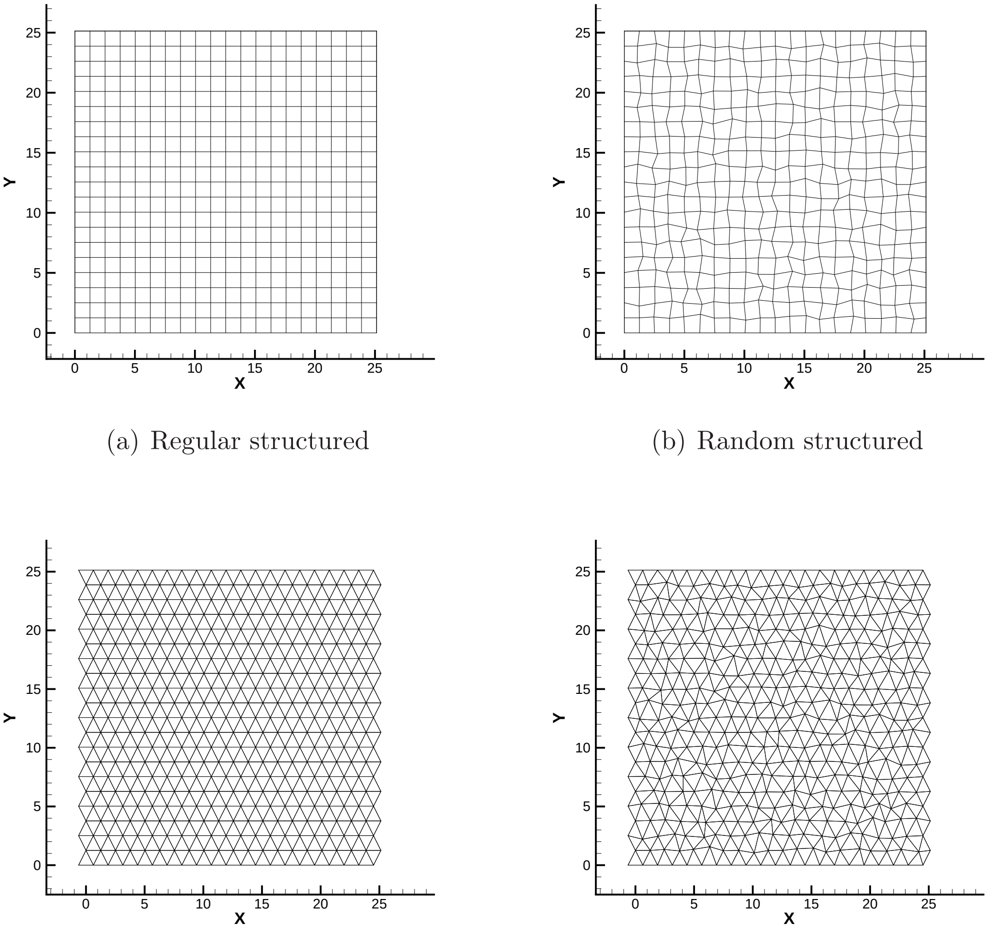
For both structured and triangular distributions, the meshless volume scheme exhibited the behavior shown in Figure 2.12. The method showed second order accuracy for the regular distributions, but reverted to first order when the points were perturbed. This was precisely the behavior of the mapped finite difference method, as shown in Figure 2.13. While second order behavior was observed for the structured finite volume method on a regular grid, the perturbed structured grid introduced 0^{th} order errors, as shown in Figure 2.14. A similar result was obtained using the unstructured finite volume method, as shown in Figure 2.15. The 0^{th} order errors were likely due to the discrepancy of a volume based approach, in which partial derivatives are assumed to be constant in a given finite volume. It is therefore difficult to compare the result with an exact solution at an exact location. Nonetheless, the cases are presented since finite volume methods are widely used in practice, often resulting in subtle contradictions. It should be emphasized that no volume/nodal location contradictions arise with the meshless volume scheme, which is a generalized finite difference method.

In conclusion, the meshless volume scheme possesses similar discretization errors as a conventional second order finite difference method. Both methods revert to first order in the face of nodal irregularities. However, the meshless volume scheme may be used on much more general mesh topologies, such as triangles or nodal distributions not arising from a mesh at all. Unfortunately, the study also shows, that point distribution strongly affects the accuracy of meshless methods, just as it does for mesh based methods.

2.4.5 Advantages of the Meshless Volume Approach

The meshless volume method of computing least squares weights in the context of the Taylor series approach has several distict advantages. First, the use of an edge-vertex list to establish cloud connectivity ensures reciprocity of nodes. Second, memory demands are reduced. A close inspection of the procedure reveals that only a single weight, w_k, need be stored at each edge k, along with a volume at each node. This is a consequence of the fact that the least squares metrics align with edges, such that the nodal coordinates themselves define face areas along with a proper weighting. For cloud distributions similar to unstructured meshes in two dimensions, there are approximately $3N$ edges, where N is the number of nodes. This means that only $4N$ real variables (a single weight for each edge plus the nodal volumes) need to be stored to completely contain the metric information. This is a factor of three reduction over the other partial derivative schemes on scattered data in which $12N$ real variables (four coefficients per edge) need to be stored. The meshless scheme is lighter in memory than even many unstructured schemes, in which $7N$ real variables (a two dimensional face vector at each edge plus a nodal volume) need to be stored.

Finally, the meshless volume scheme is unique in that it allows for the use of approximate Riemann solvers for the computation of a single midpoint flux. This avoids double solving for fluxes based on different least squares coefficients for adjacent nodes. Unfortunately, the meshless volume scheme is still not formally conservative, since the sum of the least squares coefficients around a given node is not zero. Without closed meshless volumes, it will be difficult to prove conservation. To close the meshless volumes would imply a mesh, and so the use of a meshless method would be unnecessary.



(a) Regular structured

(b) Random structured

(c) Regular triangular

(d) Random triangular

Figure 2.11: Point distrubutions used for grid convergence.

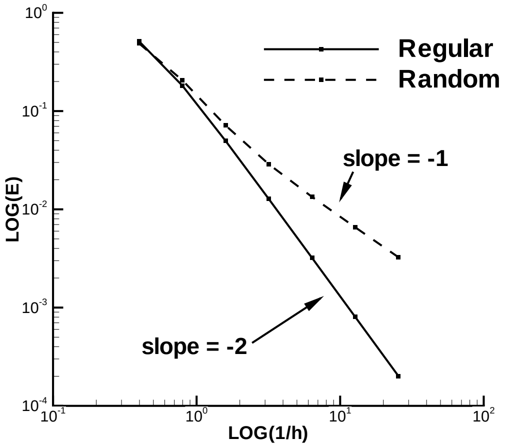


Figure 2.12: Meshless volume grid convergence.

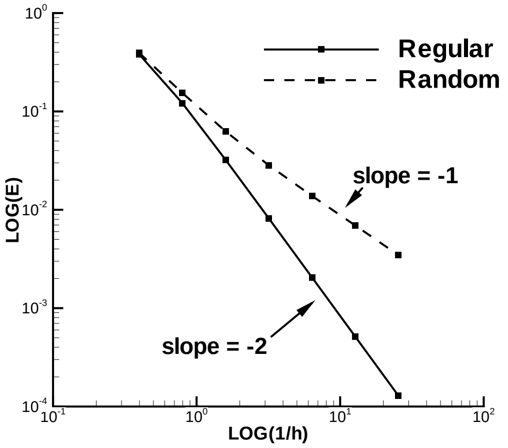


Figure 2.13: Finite difference grid convergence.

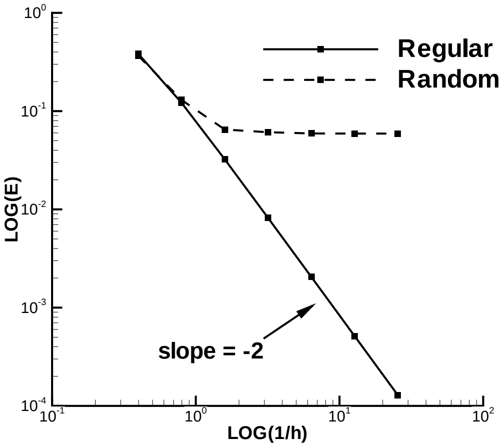


Figure 2.14: Structured finite volume (quadrilaterals) grid convergence.

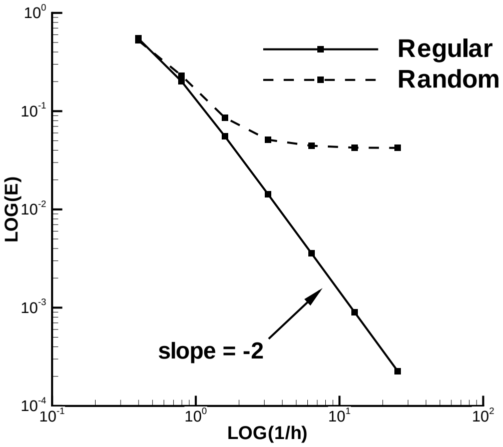


Figure 2.15: Unstructured finite volume (triangles) grid convergence.

Chapter 3

Meshless Euler and Navier-Stokes Schemes

This chapter highlights the fact that the Local Extremum Diminishing (LED) principle is flexible and general enough to extend to meshless schemes. In particular, the meshless volume scheme based on the constrained weighting procedure of Chapter 2 is used as a basis for discretization. The meshless volume discretization is quite reminiscent of conventional finite volume schemes. This enables the meshless volume scheme to leverage the vast amount of knowledge and experience derived from finite volume methods over the past few years. However, significant differences arise in the implementation of boundary conditions and other procedures, such as the convergence acceleration techniques described in Chapter 4.

This chapter begins with the formulation for the Euler and Navier-Stokes equations. Next, stabilization via artificial diffusion based on the LED principle is formulated in the general sense and for the meshless volume scheme. Reconstruction procedures are highlighted, which enable high order diffusion schemes. A method of discretization of the viscous terms of the Navier-Stokes equations is then discussed, followed by a discussion of boundary conditions. Methods of steady state integration are highlighted next. Finally, some inviscid results are given, highlighting the accuracy of the discretization scheme compared with conventional finite volume methods.

3.1 The Euler and Navier-Stokes Equations

Most modern CFD codes are based on the Euler or Navier-Stokes equations. The Navier-Stokes equations are a system of PDEs which mathematically describe the conservation of mass, momentum, and energy for fluids under a continuum assumption. The full Navier-Stokes equations have been used to effectively model unsteady, compressible, viscous flows with complex features over complex geometry [72]. The Navier-Stokes equations may be expressed in strong conservation law form in two dimensions as

$$\frac{\partial \boldsymbol{w}}{\partial t} + \frac{\partial \boldsymbol{f}_e}{\partial x} + \frac{\partial \boldsymbol{g}_e}{\partial y} = \frac{M_\infty}{Re_\infty}\left(\frac{\partial \boldsymbol{f}_v}{\partial x} + \frac{\partial \boldsymbol{g}_v}{\partial y}\right), \tag{3.1}$$

where the vector of conserved variables and Euler fluxes are

$$\boldsymbol{w} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix}, \quad \boldsymbol{f}_e = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho u v \\ \rho u H \end{pmatrix}, \quad \boldsymbol{g}_e = \begin{pmatrix} \rho v \\ \rho v u \\ \rho v^2 + P \\ \rho v H \end{pmatrix},$$

and the viscous fluxes are defined as

$$\boldsymbol{f}_v = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} - q_x \end{pmatrix}, \quad \boldsymbol{g}_v = \begin{pmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ u\tau_{yx} + v\tau_{yy} - q_y \end{pmatrix}.$$

In the above notation, ρ, u, v, P, E, and $H = E + \frac{P}{\rho}$ are the density, Cartesian velocity components, pressure, total energy, and total enthalpy. Also, M_∞ and Re_∞ are the free stream Mach and Reynolds numbers, respectively. The viscous fluxes contain shear stress terms, τ, and heat flux terms, q. The Navier-Stokes equations are completed by the ideal gas law,

$$E = \frac{P}{(\gamma - 1)\rho} + \frac{1}{2}(u^2 + v^2),$$

and Sutherland's law, which relates viscosity to temperature.

The Euler equations of gas dynamics are a set of hyperbolic partial differential equations, which are a subset of the Navier-Stokes equations. The Euler equations may be obtained by setting the right hand side of Equation 3.1 to zero. The major assumption underlying the Euler equations is that viscous effects are negligible. Since, for high Reynolds number flows, viscous effects are largely confined to thin boundary layers, the Euler assumption works well for a wide range of high speed flows encountered in aerodynamics. A more detailed discussion of the Euler and Navier-Stokes equations is given in Appendix A.

3.2 An LED-based approach

A convenient basis for the construction of non-oscillatory schemes for gas dynamics is the Local Extremum Diminishing (LED) prinicple [71] for discrete approximations to conservation laws. LED-based schemes have also been shown to work well for the Navier-Stokes equations. The simplicity of the LED principle, along with its wide-ranging applicability, has led to its widespread use for CFD. The LED principle is reiterated here, and extended to the meshless schemes developed in the previous chapter.

Given a two-dimensional scalar conservation law of the form

$$\frac{\partial v}{\partial t} + \frac{\partial f(v)}{\partial x} + \frac{\partial g(v)}{\partial y} = 0, \tag{3.2}$$

the LED criterion assumes that a discrete approximation to the spatial derivatives at i may be expressed as

$$\frac{\partial v_i}{\partial t} = \sum_j c_{ij} v_j, \tag{3.3}$$

where the summation is over the nearest neighbors, j. A consistent approximation which meets the minimal requirement to approximate a constant function exactly should have the property that

$$\sum_j c_{ij} = 0. \tag{3.4}$$

All partial derivative schemes derived in Chapter 2 have been shown to reproduce linear funtions exactly, thus satisfying Equation 3.4. It follows that with no loss of generality, Equation 3.3 may be written as

$$\frac{\partial v_i}{\partial t} = \sum_{j \neq i} c_{ij}(v_j - v_i). \tag{3.5}$$

By constructing discrete schemes which ensure the coefficients, c_{ij}, are positive, a local minimum cannot decrease and a local maximum cannot increase in time. This is the essential principle of LED schemes. The LED principle assures positivity and leads to the class of TVD schemes in one dimension proposed by Harten [73]. The requirement to produce positive coefficients has led to a variety of schemes with additive artificial diffusion terms. Jameson [71] showed that a variety of popular diffusion schemes including JST [74], scalar, and matrix schemes satisfy the LED criterion. The majority of LED schemes to date have been constructed for traditional mesh-based discretizations including finite volume, finite element, and finite difference. However, the goal of this chapter is to extend the LED principles to the meshless volume scheme. Actually, in light, of Equation 2.1, the LED principle is easily extended to all meshless schemes discussed in Chapter 2.

3.3 Stabilizing the Meshless Operator

Following the meshless volume scheme of Chapter 2, a naive discretization of Equation 3.1, neglecting the viscous terms is

$$V_i \frac{d\boldsymbol{w}_i}{dt} + \sum_j (\boldsymbol{F}_{ij} - \boldsymbol{F}_i) = 0, \tag{3.6}$$

where $\boldsymbol{F}_{ij} = \frac{1}{2}(\boldsymbol{F}_i + \boldsymbol{F}_j)$ is a midpoint flux. Here, the uppercase flux, $\boldsymbol{F} = a_{ij}\boldsymbol{f} + b_{ij}\boldsymbol{g}$ is a directed flux in the direction of the least squares coefficients, which, for the meshless volume scheme, are also aligned with the edge connecting nodes i and j. The midpoint flux alignment with the edge is shown in Figure 3.1. The least squares coefficients, a_{ij} and b_{ij}, are those of Equation 2.42.

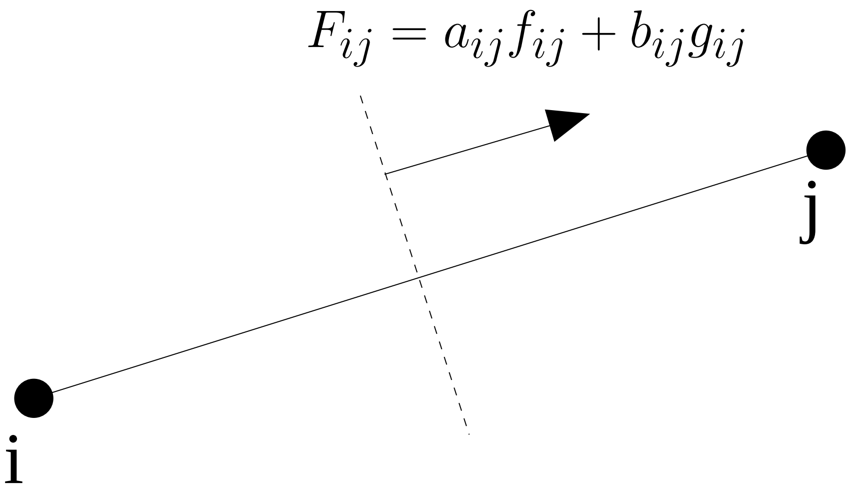


Figure 3.1: Directed flux in the direction of the edge connecting nodes i and j.

The discretization of Equation 3.6 is naive because it does not take into account characteristic directions of wave propagation of the hyperbolic system. Mathematically, Equation 3.6 does not not satisfy the LED criteria and may result in odd-even decoupling modes, overshoots and undershoots at shocks, or worse, blow-up due to instability. This may be seen by defining a mean-valued Jacobian, A_{ij}, as in the analysis of Roe [75], such that

$$\boldsymbol{F}_j - \boldsymbol{F}_i = A_{ij}(\boldsymbol{w}_j - \boldsymbol{w}_i). \tag{3.7}$$

The mean-valued Jacobian, A_{ij}, is just the typical flux Jacobian, evaluated with the Roe-averaged square root variables as shown in Appendix A. The discretization becomes

$$\begin{aligned} V_i \frac{d\boldsymbol{w}_i}{dt} &= -\sum_j (\boldsymbol{F}_{ij} - \boldsymbol{F}_i) \\ &= -\frac{1}{2} \sum_j (\boldsymbol{F}_j - \boldsymbol{F}_i) \\ &= -\frac{1}{2} \sum_j A_{ij}(\boldsymbol{w}_j - \boldsymbol{w}_i), \end{aligned}$$

which generally does not satisfy the LED criteria since the eigenvalues of the flux Jacobian, A_{ij}, vary in sign. The LED criteria may be satisfied for the meshless volume scheme by adding an artificial diffusion term, $\boldsymbol{d}_{ij}$:

$$V_i \frac{d\boldsymbol{w}_i}{dt} + \frac{1}{2}\sum_j (\boldsymbol{F}_j - \boldsymbol{F}_i - \boldsymbol{d}_{ij}) = 0. \tag{3.8}$$

This is equivalent to the midpoint discretization of Equation 3.6, but with the midpoint flux defined as

$$\boldsymbol{F}_{ij} = \frac{1}{2}(\boldsymbol{F}_i + \boldsymbol{F}_j) - \frac{1}{2}\boldsymbol{d}_{ij}.$$

The design of $\boldsymbol{d}_{ij}$ is critical to the accuracy and convergence characteristics of the meshless volume scheme. Three alternatives are presented below.

3.3.1 Scalar Diffusion

The simplest diffusion scheme is scalar diffusion. Since the Euler equations represent a superposition of waves, one may scale the diffusion to the absolute value of the maximum wave speed. Since the wave speeds are described by the eigenvalues of the Jacobian, q, $q + c$, and $q - c$, scalar diffusion takes the form

$$\boldsymbol{d}_{ij} = \lambda_{max}(\boldsymbol{w}_j - \boldsymbol{w}_i), \tag{3.9}$$

where

$$\lambda_{max} = \max(|q|, |q + c|, |q - c|),$$

and $q = a_{ij}u + b_{ij}v$ is the directed velocity.

The diffusion form of Equation 3.9 is robust and inexpensive to construct. However, scalar diffusion fails to capture shocks with a single interior point [76]. Instead, shocks and other features may be smeared as more diffusion is present than is actually needed.

3.3.2 Matrix Diffusion

In constrast with scalar diffusion, matrix diffusion is highly accurate, but at a greater computational expense. With matrix diffusion, each wave in the characteristic superposition is treated separately, reducing the amount of added diffusion. Matrix diffusion takes the form

$$\boldsymbol{d}_{ij} = |A_{ij}|(\boldsymbol{w}_j - \boldsymbol{w}_i). \tag{3.10}$$

Here, $|A| = T|\Lambda|T^{-1}$, where $|\Lambda|$ is a diagonal matrix containing the absolute values of the eigenvalues of A, and the columns of T contain the eigenvectors of A. Such a construction requires tedious evaluation and multiplication of the eigenvectors described in Appendix A. However, the matrix scheme above admits ideal shocks with a single interior point in one dimension [76].

3.3.3 Convective Upwind Split Pressure (CUSP) Scheme

A good compromise between the inexpensive scalar diffusion and the highly accurate matrix diffusion is the convective upwind split pressure (CUSP) scheme of Jameson [71]. The CUSP scheme produces shocks with a single interior point in one dimension, while minimizing computational cost. Therefore, CUSP is the preferred scheme in this work.

The CUSP scheme may be formulated to admit solutions with constant stagnation enthalpy, a defining characteristing of steady inviscid flows [77]. The diffusive flux may be defined as

$$\boldsymbol{d}_{ij} = \alpha^* c(\boldsymbol{w}_{hj} - \boldsymbol{w}_{hi}) + \beta(\boldsymbol{F}_j - \boldsymbol{F}_i), \tag{3.11}$$

with speed of sound, c, and enthalpy variables, $\boldsymbol{w}_h = (\rho, \rho u, \rho v, \rho H)^T$. Here, α^* and β are dimensionless parameters controlling the amount of diffusion to be added. Following the analysis of Jameson [76], if the total convective contribution to the diffusion is αc, we may define $\alpha^* c = \alpha c - \beta q$. A good choice for the dimensionless parameters is

$$\alpha = |M|$$

$$\beta = \begin{cases} \max\left(0, \frac{q+\lambda^-}{q-\lambda^-}\right), & 0 \leq M \leq 1 \\ -\max\left(0, \frac{q+\lambda^+}{q-\lambda^+}\right), & -1 \leq M \leq 0 \\ \operatorname{sign}(M), & M \geq 1. \end{cases}$$

Here, M is the local Mach number, and

$$\lambda^{\pm} = \frac{\gamma+1}{2\gamma}q \pm \sqrt{\left(\frac{\gamma+1}{2\gamma}q\right)^2 + \frac{c^2-q^2}{\gamma}}$$

are the eigenvalues of the Jacobian of the flux with respect to the enthalpy variables. Thus, the CUSP scheme only requires the evaluation of slightly more complex eigenvalues, but bypasses the tedious eigenvector evaluation and multiplication of matrix schemes.

3.4 Limited Reconstruction for Meshless Schemes

Since the states at the endpoints i and j are used, all the diffusion schemes described above are far too diffusive, being only first order accurate. These schemes may be made more accurate by the addition of anti-diffusive terms, which increase the order of accuracy to third order. Anti-diffusion may be conveniently added via the reconstruction of left and right states at the midpoint of the line connecting points i and j. This practice effectively reduces the difference in the state variables, in turn reducing the diffusion. A proven reconstruction method is the symmetric limited positive (SLIP) scheme of Jameson [71], which leads to high order accuracy in smooth regions and reverts to first order accuracy in the vicinity of discontinuities to avoid oscillations. While the SLIP scheme was originally designed for structured grids, it has also proven effective for unstructured meshes. Here the SLIP scheme is extended to meshless methods.

The use of left and right reconstructed states replaces $\boldsymbol{d}_{ij}$ with $\boldsymbol{d}_{LR} = \alpha^* c(\boldsymbol{w}_{hR} - \boldsymbol{w}_{hL}) + \beta(\boldsymbol{F}_R - \boldsymbol{F}_L)$, where

$$\boldsymbol{w}_L = \boldsymbol{w}_i + \frac{1}{4}s(\Delta\boldsymbol{w}^+, \Delta\boldsymbol{w}^-)\Delta\boldsymbol{w}^+, \quad \boldsymbol{w}_R = \boldsymbol{w}_j - \frac{1}{4}s(\Delta\boldsymbol{w}^+, \Delta\boldsymbol{w}^-)\Delta\boldsymbol{w}^-. \tag{3.12}$$

Here, $\Delta \boldsymbol{w}^+$ and $\Delta \boldsymbol{w}^-$ are estimates of the change in the solution on either side of the edge connecting point i and j, as shown in Figure 3.2. First, points I and J, the points in the local clouds of i and j which are most closely aligned with the edge ij are identified. Estimates with $\Delta \boldsymbol{w}^+ = \boldsymbol{w}_J - \boldsymbol{w}_j$ and $\Delta \boldsymbol{w}^- = \boldsymbol{w}_i - \boldsymbol{w}_I$ may be used. However an increase in accuracy is observed by making corrections of the form

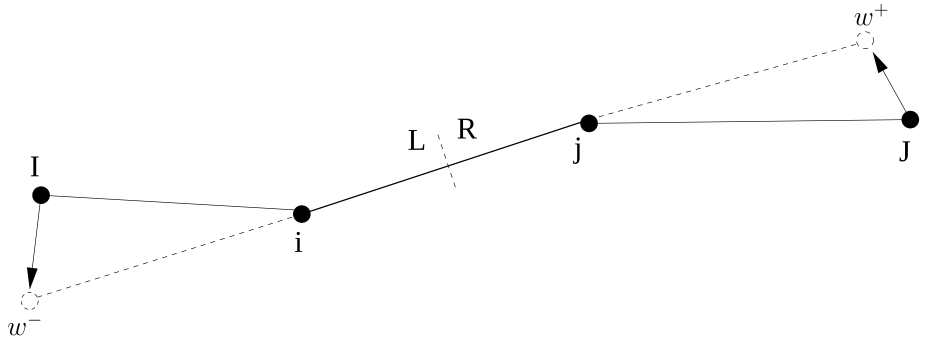


Figure 3.2: Edge reconstruction procedure to obtain left and right states.

$$\Delta \boldsymbol{w}^+ = \boldsymbol{w}_J - \boldsymbol{w}_j + l_j \cdot \nabla \boldsymbol{w}_J, \quad \Delta \boldsymbol{w}^- = \boldsymbol{w}_i - \boldsymbol{w}_I - l_i \cdot \nabla \boldsymbol{w}_I, \tag{3.13}$$

where the gradients of $\boldsymbol{w}$ are computed using the same least squares coefficients as the base meshless volume scheme, and l_i and l_j are the position vectors from I and J to the psuedo-points past the edge.

The $\Delta \boldsymbol{w}$ estimates obtained in this way lead to sharp capturing of discontinuities, turning off the limiting function,

$$s(u,v) = 1 - \left| \frac{u - v}{|u| + |v| + \epsilon \Delta x^r} \right|^q, \tag{3.14}$$

in the presence of shocks. Here q is a positive integer, usually taken to be $q = 3$, and $\epsilon \Delta x^r$ is a threshold proportional to the cloud spacing to improve accuracy and convergence. The exponent $r = 3/2$ ensures no degradation in accuracy. With the limiter turned on, the left and right states revert back to the i and j states, maintaining the LED quality at a local extremum. The key is the accurate detection of local extrema and shocks, which the above procedure performs quite well in practice.

It should be emphasized that the same metric information derived from the least squares procedure may be used in the gradient reconstruction procedure. This means that a single set of metrics completely satisfies all demands of the algorithm. This stands in contrast to many unstructured codes, which used separate metrics for reconstruction as for flux discretization. Significant savings in memory have been acheived over many types of unstructured algorithms in this manner.

3.5 Boundary Conditions

The LED principle may be enforced at domain boundaries by using the same interior scheme for boundary points. At domain boundaries, interior points are reflected across tangent boundary planes to form ghost points, as shown in Figure 3.5. The ghost points serve two purposes. First, they balance the local clouds of the boundary points, improving the condition of meshless discretization procedures. Second, solution values at ghost points may be set consistent with any number of physical boundary conditions to obtain a solution. With the injection of proper conditions at the boundary, no modification of the interior scheme is necessary.

3.5.1 Solid Walls

At solid walls in inviscid flow, slip boundary conditions are enforced. This may be accomplished by defining an outward unit normal direction, (n_x, n_y), at each node on a solid wall. The Cartesian velocity components of the interior point, (u_i, v_i), are then rotated into the local normal/tangential components, (u_t, u_n):

$$\begin{aligned} u_t &= n_y u_i - n_x v_i \\ u_n &= n_x u_i + n_y v_i. \end{aligned}$$

To compute the Cartesian velocity components of the ghost nodes, (u_g, v_g), the

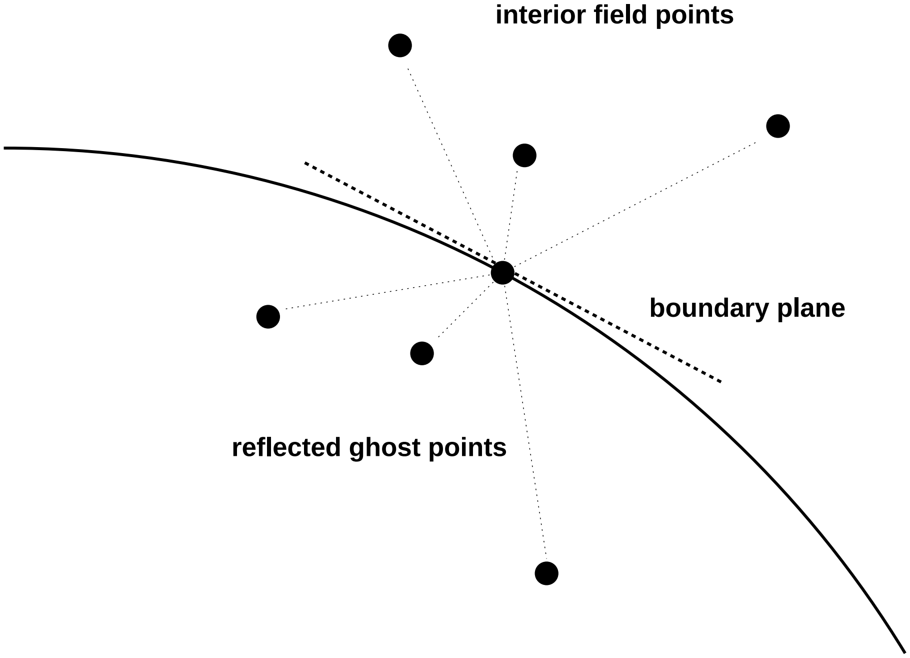


Figure 3.3: Reflection of interior points to produce ghost nodes.

tangential velocity is untouched, while the normal velocity is negated:

$$\begin{aligned} u_g &= n_y u_t + n_x(-u_n) \\ v_g &= -n_x u_t + n_y(-u_n). \end{aligned}$$

To obtain values for density, ρ_g, and energy, $(\rho E)_g$, at the ghost nodes, an extrapolation procedure has proven accurate. Values at the interior node and surface node are used to extrapolate to the ghost node:

$$\begin{aligned} \rho_g &= 2\rho_s - \rho_i \\ (\rho E)_g &= 2(\rho E)_s - (\rho E)_i. \end{aligned}$$

Here, the subscripts s and i denote the surface node and interior nodes respectively. To be consistent with the interior scheme, which admits constant stagnation enthalpy, the pressure at the ghost node may be computed as $P_g = \rho_g(H_o - E_g)$, where H_o is the total enthalpy at freestream.

While the above procedure is designed for slip wall conditions, a wide variety of boundary conditions may be enforced by injecting the relavent values into the ghost nodes. For example, no slip boundary conditions for viscous flow may be enforced by setting the ghost velocities to

$$\begin{aligned} u_g &= -u_i \\ v_g &= -v_i. \end{aligned}$$

Similar procedures would be enforced for temperature and pressure in viscous flow.

3.5.2 Far Field

At the far field, one-dimensional Riemann invariants are used, while enforcing constant stagnation enthalpy and a compressible vortex correction [78] for lifting flows. The vortex correction involves the use of corrected velocities, (u_f, v_f), instead of free stream values, (u_o, v_o), when values external to the domain are needed:

$$u_f = u_o + \frac{\Gamma \sin\theta}{R\,(1 - M_o^2(\sin\theta\cos\alpha - \cos\theta\sin\alpha)^2)}$$

$$v_f = v_o - \frac{\Gamma \cos\theta}{R\,(1 - M_o^2(\sin\theta\cos\alpha - \cos\theta\sin\alpha)^2)}$$

Here, Γ is the computed circulation, R and θ are the polar coordinates of the boundary point measured from the quarter-chord of the airfoil, M_o is the freestream Mach number, and α is the angle of attack.

Following the work of Jameson [79], the one dimensional incoming and outgoing Riemann invariants normal to the boundary may then be defined as

$$\begin{aligned} R_o &= u_{no} - \frac{2c_o}{\gamma - 1} \\ R_e &= u_{ne} + \frac{2c_e}{\gamma - 1}, \end{aligned}$$

where the subscripts o and e denote freestream and interior extrapolated quantities, respectively, and c is the speed of sound. Extrapolation from the interior is facilitated by the boundary node reflection procedure illustrated in Figure 3.5. The normal velocity at the ghost node, u_{ng} may then be obtained from $u_{ng} = \frac{1}{2}(R_o + R_e)$. Additionally, the tangential velocity at the ghost node, u_{tg}, is obtained from the corrected freestream values for inflow or extrapolated from the interior for outflow. Entropy is obtained in the same way. Finally, by using the freestream stagnation enthalpy, H_o, the state at the ghost nodes is completely defined, with the ability to admit constant stagnation enthalpy globally.

3.6 Discretization of Viscous Terms

While it has been shown that many flux-limited schemes developed for the Euler equations work well for viscous flows [80], certain complications arise with the addition of the viscous fluxes. The difficulties with the viscous fluxes are essentially two-fold. First, the viscous fluxes contain first derivatives of flow quantities. This is in contrast to the inviscid fluxes, which are only functions of the flow variables themselves, not their derivatives. Second, As the Reynolds number increases and becomes large for many practical applications, anisotropic point distributions become necessary to efficiently capture boundary layers, wakes, and other features unique to viscous flows.

In the context of the strong conservation law form of Equation 3.1, three approaches appear most often in the literature to discretize the viscous fluxes for meshless schemes. The first, advocated by Batina [50], is to simply use the same least squares coefficients as are used to discretize the partial derivatives of the Euler fluxes to compute the gradients in the viscous fluxes. The second approach, used by Oñate et. al.[52, 54], involves the selection of a higher order polynomial basis, such that second derivatives may be computed directly from the approximating function. The third approach involves the use of radial basis functions [26, 28, 23, 19, 24] to obtain higher derivatives since radial basis functions are continuously differentiable.

The last two approaches would require modification of the conservation law form to obtain second derivative terms. Moreover, the polynomial basis and radial basis methods described in Chapter 2 fall outside the framework of the meshless volume scheme, which is based on the Taylor series approximation. Thus, in order to use the same metrics available for the meshless volume scheme and the strong conservation law form directly, a variant of the approach of Batina appears most viable.

The use of anisotropic point clouds with least squares procedures often leads to poor accuracy, as pointed out by Mavriplis [30]. Problems arise in regions of high curvature, in which nodes exert influence on gradient computation in unexpected ways. Similar problems have been observed by Liu and Jameson for kinked meshes using the Gauss formula for gradient estimation [81]. For this reason, a hybrid scheme has been developed to capture anisotropic features with a mesh, while using a meshless method elsewhere in the domain. This hybrid scheme is the main focus of Chapter 5. Here, the viscous discretization is formulated for use in the hybrid scheme.

A naive discretization of the right hand side of Equation 3.1 would be

$$\frac{M_\infty}{Re_\infty}\left(\frac{\partial \boldsymbol{f}_v}{\partial x}+\frac{\partial \boldsymbol{g}_v}{\partial y}\right)=\frac{M_\infty}{Re_\infty}\sum_{j=1}^{n}(\boldsymbol{F}_{vij}-\boldsymbol{F}_{vi}), \tag{3.15}$$

where again $\boldsymbol{F}=a\boldsymbol{f}+b\boldsymbol{g}$ is a directed flux along the least squares coefficient vector, and $\boldsymbol{F}_{vi}=\boldsymbol{F}_v(\boldsymbol{w}_i)$. Here, $\boldsymbol{F}_{vij}=\frac{1}{2}(\boldsymbol{F}_{vi}+\boldsymbol{F}_{vj})$ is a midpoint flux. However, such a discretization would involve a recycling of first derivatives at each node, potentially leading to odd/even node decoupling.

A stable and accurate discretization is obtained by carefully defining the midpoint viscous flux, $\boldsymbol{F}_{vij}$. The midpoint viscous flux is computed with a mix of arithmetic and modified averages. When non-derivative quantities appear in the flux definition, arithmetic averages of the values at i and j are used. When derivative quantities are needed, a modified average is used, following the scheme of May and Jameson [82]. The modified gradient is thus defined as

$$\nabla\phi_{ij}=\overline{\nabla\phi_{ij}}-\left(\overline{\nabla\phi_{ij}}\cdot\vec{s}_{ij}-\frac{\phi_j-\phi_i}{|\vec{r}_j-\vec{r}_i|}\right)\vec{s}_{ij}, \tag{3.16}$$

where $\overline{\nabla\phi_{ij}}=\frac{1}{2}(\nabla\phi_i+\nabla\phi_j)$ is the average of the gradients at i and j using the base least squares method, and $\vec{s}_{ij}=\frac{\vec{r}_j-\vec{r}_i}{|\vec{r}_j-\vec{r}_i|}$ is the unit vector in the direction of the edge.

Here, ϕ represents velocity and temperature. For example, the shear stress term, τ_{xy}, would be computed as

$$\tau_{xy,ij} = \frac{1}{2}(\mu_i + \mu_j)(u_{y,ij} + v_{x,ij}),$$

where the derivatives of velocity are computed as shown in Equation 3.16. This discretization scheme has proven robust and accurate when applied to the hybrid scheme to be discussed in Chapter 5.

3.7 Integration to Steady State

Integration to steady state may be accomplished with the modified Runge-Kutta approach of Jameson [74]. Since steady state is the primary objective, the time marching scheme may be optimized for stability and large time steps with no regard to consistency. In this approach, a distinction is made between diffusive and convective contributions to the total residual at each node.

The semi-discretized form of the Navier-Stokes equations may be expressed as

$$\frac{\partial \boldsymbol{w}_i}{\partial t} + \boldsymbol{R}_i = 0, \tag{3.17}$$

where the total residual is defined as

$$\boldsymbol{R}_i = \frac{1}{V_i}(\boldsymbol{Q}_i - \boldsymbol{D}_i), \tag{3.18}$$

with meshless volume, V_i. Here the convective portion of the residual,

$$\boldsymbol{Q}_i = \sum_{j=1}^{n} \frac{1}{2}(\boldsymbol{F}_{ej} - \boldsymbol{F}_{ei}), \tag{3.19}$$

is just the collection of Euler flux differences across edges. The diffusive portion of the residual,

$$\boldsymbol{D}_i = \frac{1}{2}\sum_{j=1}^{n} \boldsymbol{d}_{ij} + \frac{M_\infty}{Re_\infty}\sum_{j=1}^{n}(\boldsymbol{F}_{vij} - \boldsymbol{F}_{vi}), \tag{3.20}$$

is the collection of artificial and viscous fluxes.

An m-stage scheme may be expressed as

$$\begin{aligned} \boldsymbol{w}^{(n+1,0)} &= \boldsymbol{w}^n \\ &\vdots \\ \boldsymbol{w}^{(n+1,k)} &= \boldsymbol{w}^n - \alpha_k \Delta t \left(\boldsymbol{Q}^{(k-1)} + \boldsymbol{D}^{(k-1)} \right) \\ &\vdots \\ \boldsymbol{w}^{(n+1)} &= \boldsymbol{w}^{(n+1,m)}, \end{aligned}$$

where

$$\begin{aligned} \boldsymbol{Q}^{(0)} &= \boldsymbol{Q}(\boldsymbol{w}^n), \quad \boldsymbol{D}^{(0)} = \boldsymbol{D}(\boldsymbol{w}^n) \\ &\vdots \\ \boldsymbol{Q}^{(k)} &= \boldsymbol{Q}(\boldsymbol{w}^{(n+1,k)}), \quad \boldsymbol{D}^{(k)} = \beta_k \boldsymbol{D}(\boldsymbol{w}^{(n+1,k)}) + (1-\beta_k)\boldsymbol{D}^{(k-1)}. \end{aligned}$$

The following five-stage scheme has been used with much success [63]:

$$\begin{aligned} \alpha_1 &= \frac{1}{4} \qquad \beta_1 = 1 \\ \alpha_2 &= \frac{1}{6} \qquad \beta_2 = 0 \\ \alpha_3 &= \frac{3}{8} \qquad \beta_3 = 0.56 \\ \alpha_4 &= \frac{1}{2} \qquad \beta_4 = 0 \\ \alpha_5 &= 1 \qquad \beta_5 = 0.44 \end{aligned}$$

3.7.1 Local Time Stepping

Since steady state is the primary objective, local time stepping may be employed to accelerate convergence. A local time step estimate can be derived based on the spectral radius of the inviscid and viscous flux Jacobians. The spectral radius of the inviscid flux Jacobian for the meshless volume scheme at a node i, λ_I, may be approximated as

$$\lambda_I \approx \sum_{j=1}^{n} \left(|a_{ij}u_i + b_{ij}v_i| + \sqrt{a_{ij}^2 + b_{ij}^2}c_i \right), \tag{3.21}$$

where u,v, and c are the local velocities and speed of sound. Adapting the derivation of Mavriplis for unstructured meshes [64] to the meshless scheme, the spectral radius of the viscous flux jacobian, λ_V, may be approximated as

$$\lambda_V \approx \frac{\gamma^{\frac{3}{2}} M_\infty}{Re_\infty Pr} \sum_{j=1}^{n} \frac{\mu_{ij}}{\rho_{ij}} \left(a_{ij}^2 + b_{ij}^2 \right), \tag{3.22}$$

where Pr is the Prandtl number and μ_{ij} and ρ_{ij} are the average viscosity and density across the edge connecting nodes i and j. The local time step estimate for the meshless scheme is then

$$dt_i = \text{CFL} \frac{V_i}{\lambda_I + \lambda_V} \tag{3.23}$$

where V_i is the meshless volume. Using the above time step estimates, CFL = 3 was obtained in practice.

3.7.2 Implicit Residual Smoothing and Enthalpy damping

Along with local time stepping, implicit residual smoothing [83] may be used to accelerate convergence. Implicit residual smoothing was implemented in a manner similar to that of edge-based unstructured grid algorithms:

$$\bar{R}_i = R_i + \epsilon \nabla^2 \bar{R}_i \tag{3.24}$$

The solution of Equation 3.24 is costly to obtain in general, but two Jacobi iterations were sufficient to increase the CFL by a factor of two. The use of implicit residual smoothing is a good example of the advantage of using an edge based data structure. The implementation of the smoothing procedure is greatly facilitated by using edges. Finally, enthalpy damping [74] may be used to further accelerate convergence. The enthalpy damping procedure tends to drive the solution to the constant enthalpy solution. It should be noted that enthalpy damping should only be used with schemes which admit constant stagnation enthalpy, such as the enthalpy variation of the CUSP scheme for inviscid flow. The procedure should be turned off for viscous simulations.

3.8 Inviscid Results

Along with the meshless volume scheme described above, denoted "MV" in the following figures, three other meshless schemes were implemented for comparison. The first of these schemes, denoted "TLS", uses the Taylor least squares procedure, but with unconstrained weights. The second scheme for comparison, denoted "PLS" makes use of the polynomial least squares procedure with unconstrained weights, as described in Chapter 2. The third scheme for comparison, denoted "RBF" is based on the multiquadric radial basis method, also described in Chapter 2. The unified framework for meshless derivative procedures as indicated by Equation 2.1 allows for a simple comparison of these schemes.

The four meshless schemes–meshless volume, Taylor least squares, polynomial least squares, and radial basis funtion–were compared with an inviscid structured finite volume code based on the CUSP algorithm of Jameson [71, 76], denoted "FV" in the following tables. The finite volume code uses a conservative approach of numerical fluxes, taking on the form,

$$V_{ij}\frac{d\boldsymbol{w}_{ij}}{dt} + \boldsymbol{h}_{i+\frac{1}{2},j} - \boldsymbol{h}_{i-\frac{1}{2},j} + \boldsymbol{h}_{i,j+\frac{1}{2}} - \boldsymbol{h}_{i,j-\frac{1}{2}} = 0, \tag{3.25}$$

where

$$\boldsymbol{h}_{i+\frac{1}{2},j} = \frac{1}{2}(\boldsymbol{f}_{i,j} + \boldsymbol{f}_{i+1,j}) - \frac{1}{2}(\boldsymbol{d}_{i+\frac{1}{2},j}) \tag{3.26}$$

is the numerical flux. The numerical flux, $\boldsymbol{h}$, is based on the average of the directed Euler fluxes, $\boldsymbol{f}$, augmented with a diffusive flux, $\boldsymbol{d}$, which is based on the CUSP formulation. Similar procedures were used for the finite volume algorithm to reach steady state, including explicit Runge-Kutta local time stepping, implicit residual smoothing, enthalpy damping, and multigrid. Similar point densities for both the finite volume and meshless schemes were used, with 160 points on the surface of airfoils, and roughly 6400 nodes in the interior of the domain.

Table 3.1: Drag convergence for inviscid test cases

	NACA 0012 $M = 0.5$, $\alpha = 3^o$		KORN Airfoil $M = 0.75$, $\alpha = 0^o$		RAE 2822 $M = 0.5$, $\alpha = 3^o$	
Surface points	Total points	C_d	Total points	C_d	Total points	C_d
40	526	0.0082	474	0.0036	457	0.0104
80	1591	0.0009	1919	0.0013	1580	0.0018
160	5443	0.0001	7810	0.0000	5733	0.0003

In order to assess the order of accuracy of the meshless volume scheme, a drag convergence study was performed on three cases, as shown in Table 3.1. In all three cases, the correct Euler solution should give zero drag. As the point distributions are refined, the order of accuracy appears to be between second and third order. This is higher than expected, but can be explained by the fact that the diffusion is third order. It is likely the diffusion is creating most of the error in the drag. In the case of the KORN airfoil, the drag convergence is slightly worse. This may be explained by the fact that the coarse point distributions may inaccurately place a weak shock on the upper surface. Since at a shock the diffusion is only first order, the drag convergence will degrade.

Figures 3.4-3.5 show meshless pressure curves (dots) overplotted with the finite volume results (lines) for subsonic flow with no shocks. All methods perform essentially equally well, computing lift and drag closely to the finite volume results, as shown in Tables 3.3-3.4. Figure 3.4 is a NACA 0012 no lift and no drag case, which is computed quite accurately by all methods. Figure 3.5 is a lifting subsonic case, but should give zero drag. Again, all methods perform quite well. Additionally, Figure 3.9 is the KORN airfoil, which is designed to be shock free at $M = 0.75$, and $\alpha = 0^o$. The least squares methods give a shock free upper surface, while the radial basis method produces a very weak shock. Still all meshless methods performed well for shock free solutions.

While shock free solutions are computed very well by all methods, discrepancies arise in the presence of shock waves, as shown in Figures 3.6-3.8 and Tables 3.5-3.7.

The meshless volume and other least squares methods appear to give the correct shock locations and jumps. However, significant errors in both shock location and magnitude are present with the radial basis method. Instead of errors of less than 3% in lift and drag coefficients as with the least squares methods, the RBF method produces errors in the 10-15% range. A close examination of the pressure curves reveals these errors. A variety of shape parameters in Equation 2.22 were tested to try and improve the accuracy of the shock capturing but with little success.

It is possible that the incorrect shock capturing may be due to the non-conservative nature of the radial basis scheme. To investigate this, a study was performed to assess the ability of the scheme to satisfy the Rankine-Hugoniot shock jump conditions:

$$\begin{aligned}(\rho u)_L &= (\rho u)_R, \\ (P+\rho u^2)_L &= (P+\rho u^2)_R, \\ H_{tL} &= H_{tR}\end{aligned}$$

Here the left (L) and right (R) states must satisfy these conditions on either side of the shock. The error in each of these conditions for the structured finite volume method of Equation 3.25, the meshless volume scheme, and the radial basis method was examined. The results for a transonic case are shown in Table 3.2

Table 3.2: Error in Rankine-Hugoniot condtions, NACA 0012, $M = 0.8$, $\alpha = 1.25^o$.

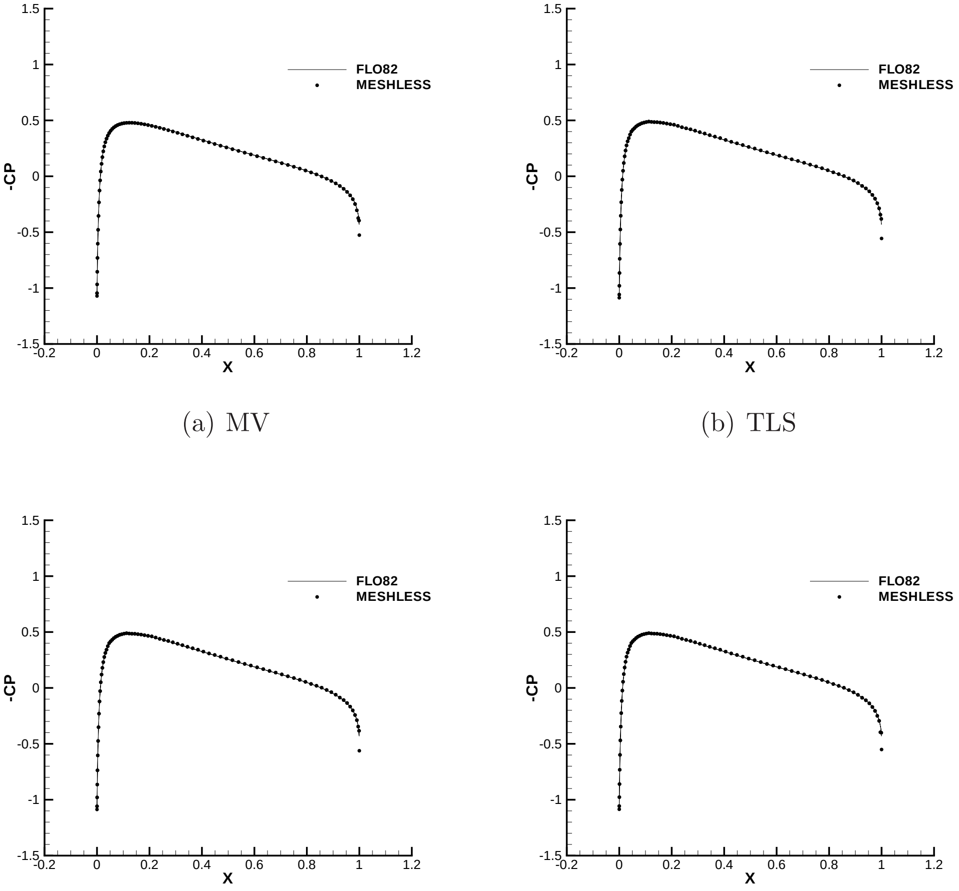
	$\lvert(\rho u)_L - (\rho u)_R\rvert$	$\lvert(P+\rho u^2)_L - (P+\rho u^2)_R\rvert$	$\lvert H_{tL} - H_{tR}\rvert$
FV	0.0023	0.0035	0.0000
MV	0.0026	0.0036	0.0000
RBF	0.0015	0.0047	0.0000

As the table shows, there is no clear evidence that the radial basis method is actually miscapturing shocks. The error in the Rankine-Hugoniot conditions is on the same order for all methods. Tests with other transonic cases showed similar results. Therefore, it is more likely that some other source other than non-conservation is the cause of the misplaced shocks. After all, the meshless volume scheme is also non-conservative, but consistently captures shocks correctly. Possible explanations may include a failure to adequately satisfy the Kutta condition or boundary conditions. Further tests are needed to determine the cause of the poor accuracy for transonic flow using the radial basis method.

An important observation gleaned from Figures 3.4-3.9 is that the meshless volume scheme obtains results which are just as accurate as the polynomial scheme. The weight constraining procedure appears to have retained the accuracy while enabling a much more efficient algorithm.

As the meshless volume scheme is of primary interest here, a more detailed flow solution is presented for each of the cases tested. Figures 3.10-3.15 show the surface pressure, pressure contours, mach contours, and the meshless point distribution for each of these cases. In the case of the Korn airfoil, the upper surface is virtually shock free, indicating a high level of accuracy. In the transonic cases, the shock location and magnitude coincide well with the finite volume results. The point distributions were obtained from an unstructured mesh in order to validate the results and compare them with existing solution methods. While the use of a mesh to obtain local clouds of points seems to defeat the purpose of using a meshless method, it has proven quite useful as a validation procedure. While the task of point generation remains an open problem, the focus here was on obtaining the most efficient and accurate meshless scheme possible. In this context, Chapter 3 may be considered a validation work for subsequent studies. In later chapters, the meshless methods described here will be applied in scenarios where a mesh is not used as the primary means of distributing points., such as in the cases of multicloud and the meshless interface for overset grids.

In conclusion, the meshless volume method here produces results comparable to the most accurate and efficient grid-based solvers for inviscid flow in two-dimensions. The accuracy is obtained with high order reconstruction of diffusion terms in an edge-based framework. The efficiency of the scheme is obtained with convergence acceleration schemes borrowed from mesh-based schemes, such as local time stepping, implicit residual smoothing, and enthalpy damping. The results presented here encourage increased use of the meshless volume scheme for practical flow computation.



(a) MV

(b) TLS

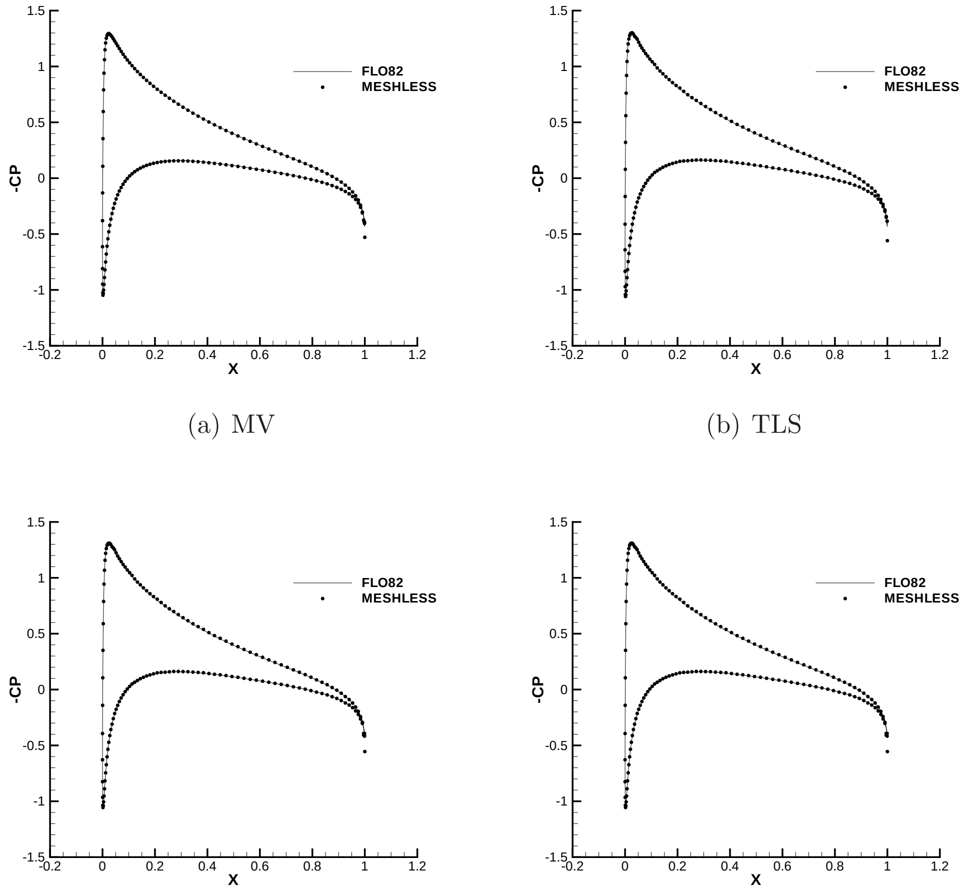
(c) PLS

(d) RBF

Figure 3.4: Surface pressure coefficient NACA 0012, $M = 0.5$, $\alpha = 0.0^o$.

Table 3.3: Lift and drag coefficients, NACA 0012, $M = 0.5$, $\alpha = 0.0^o$.

	c_l	% difference	c_d	% difference
FV	0.0000	-	0.0000	-
MV	0.0001	-	0.0001	-
TLS	0.0001	-	0.0001	-
PLS	0.0000	-	0.0001	-
RBF	0.0001	-	0.0004	-



(a) MV

(b) TLS

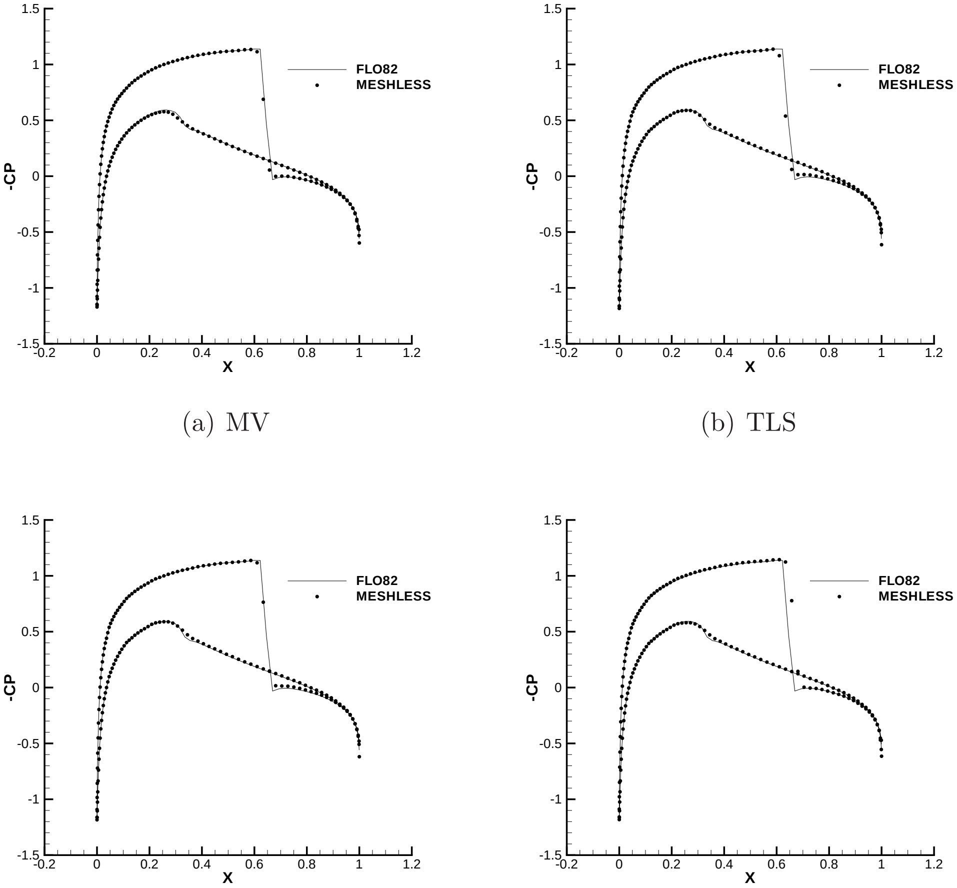
(c) PLS

(d) RBF

Figure 3.5: Surface pressure coefficient NACA 0012, $M = 0.5$, $\alpha = 3.0^o$.

Table 3.4: Lift and drag coefficients, NACA 0012, $M = 0.5$, $\alpha = 3.0^o$.

	c_l	% difference	c_d	% difference
FV	0.4313	-	0.0000	-
MV	0.4321	0.2	0.0001	-
TLS	0.4312	0.0	0.0007	-
PLS	0.4326	0.3	0.0005	-
RBF	0.4340	0.6	0.0002	-



(a) MV

(b) TLS

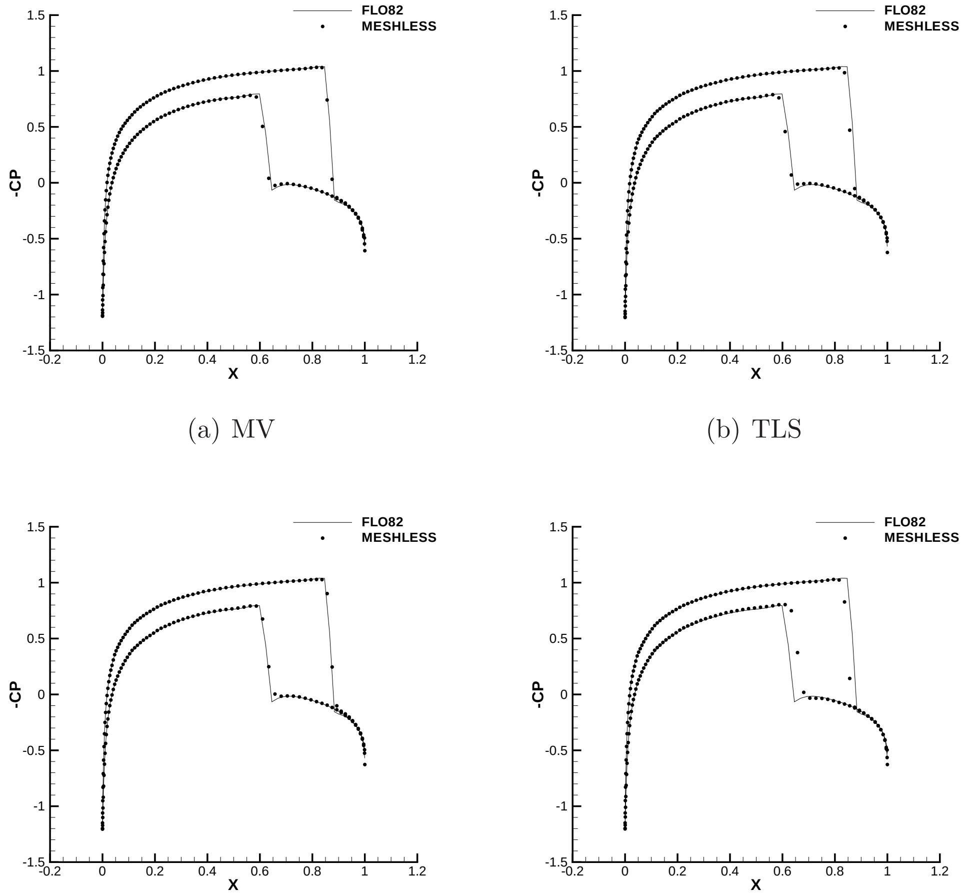
(c) PLS

(d) RBF

Figure 3.6: Surface pressure coefficient NACA 0012, $M = 0.8$, $\alpha = 1.25^o$.

Table 3.5: Lift and drag coefficients, NACA 0012, $M = 0.8$, $\alpha = 1.25^o$.

	c_l	% difference	c_d	% difference
FV	0.3737	-	0.0237	-
MV	0.3713	0.6	0.0229	3.4
TLS	0.3616	3.2	0.0231	2.5
PLS	0.3684	1.4	0.0242	2.1
RBF	0.3986	6.7	0.0269	13.5



(a) MV

(b) TLS

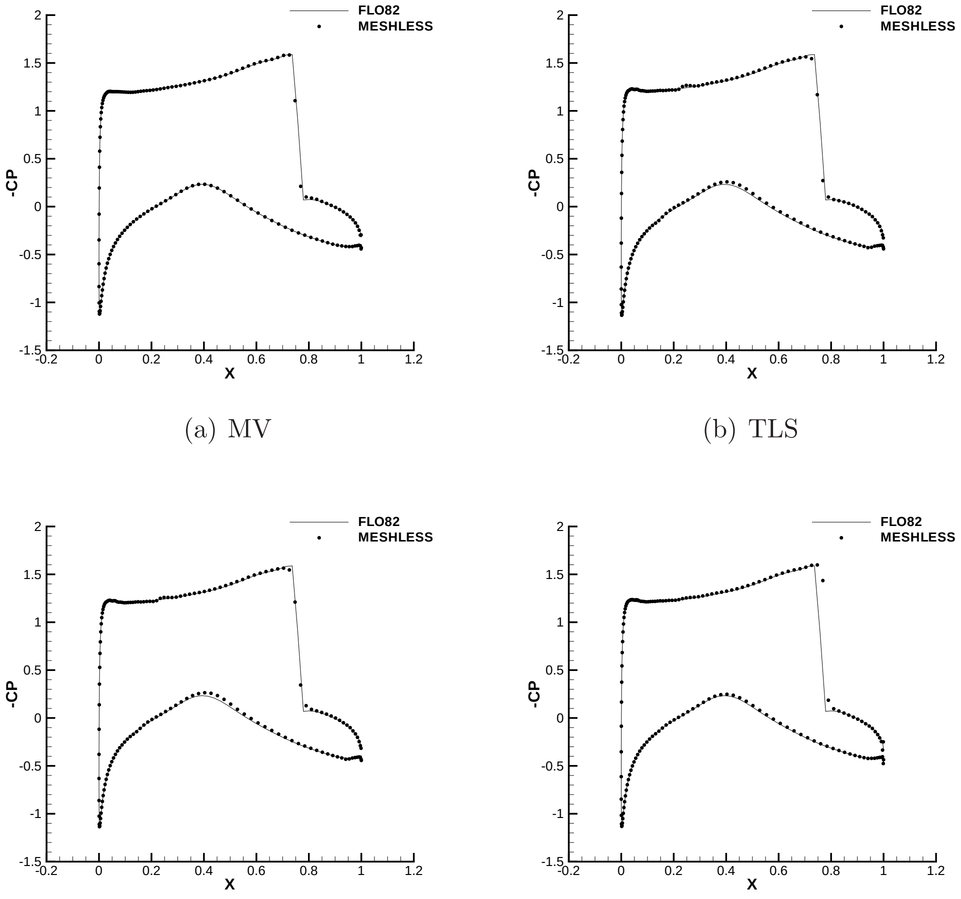
(c) PLS

(d) RBF

Figure 3.7: Surface pressure coefficient NACA 0012, $M = 0.85$, $\alpha = 1.0^o$.

Table 3.6: Lift and drag coefficients, NACA 0012, $M = 0.85$, $\alpha = 1.0^o$.

	c_l	% difference	c_d	% difference
FV	0.3891	-	0.0582	-
MV	0.3923	0.8	0.0572	1.7
TLS	0.3830	1.6	0.0565	2.9
PLS	0.3883	0.2	0.0593	1.9
RBF	0.3343	14.1	0.0570	2.1



(a) MV

(b) TLS

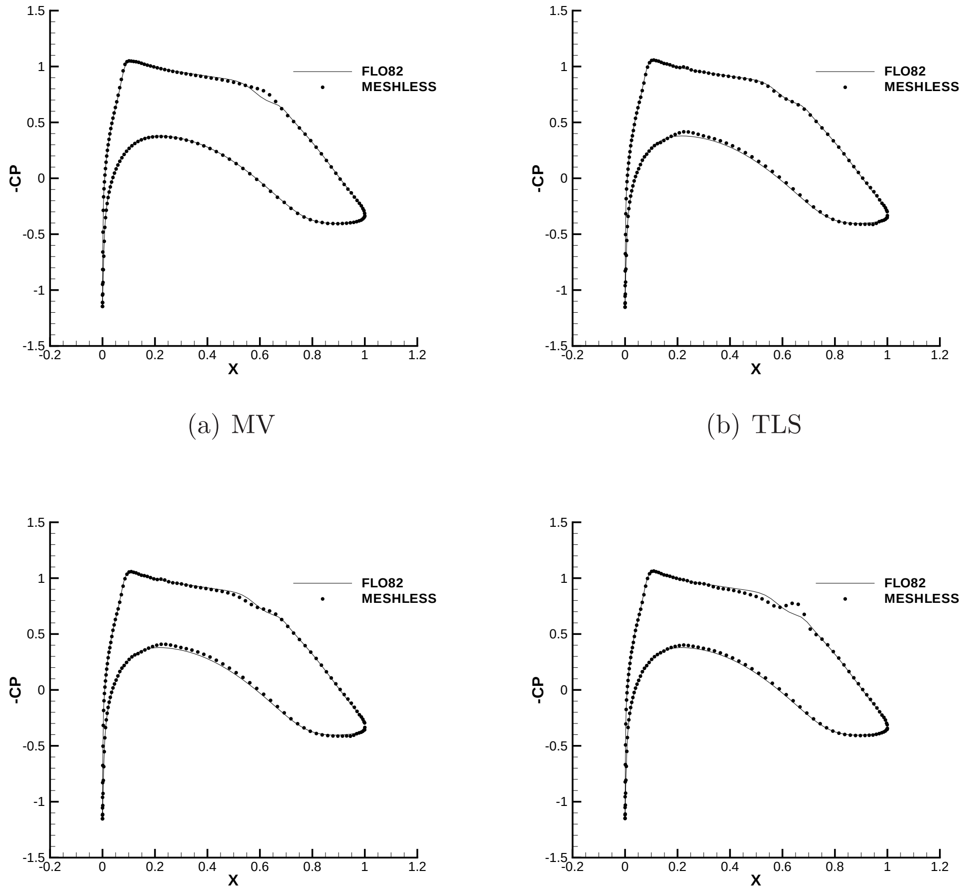
(c) PLS

(d) RBF

Figure 3.8: Surface pressure coefficient RAE 2822, $M = 0.75$, $\alpha = 3.0^o$.

Table 3.7: Lift and drag coefficients, RAE 2822, $M = 0.75$, $\alpha = 3.0^o$.

	c_l	% difference	c_d	% difference
FV	1.1481	-	0.0486	-
MV	1.1338	1.2	0.0474	2.5
TLS	1.1319	1.4	0.0481	1.0
PLS	1.1347	1.2	0.0491	1.0
RBF	1.1744	2.3	0.0545	12.1



(a) MV

(b) TLS

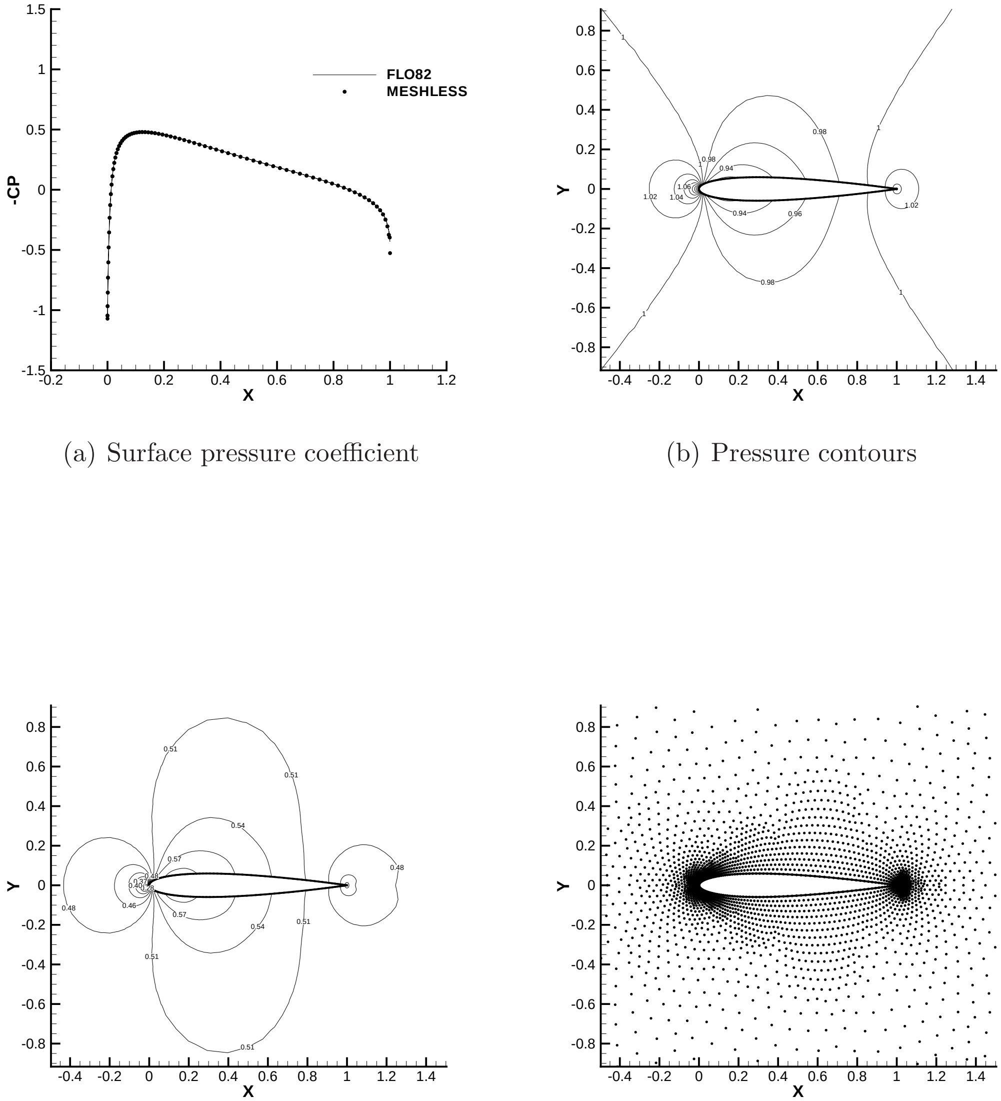
(c) PLS

(d) RBF

Figure 3.9: Surface pressure coefficient KORN airfoil, $M = 0.75$, $\alpha = 0.0^o$.

Table 3.8: Lift and drag coefficients, KORN airfoil, $M = 0.75$, $\alpha = 0.0^o$.

	c_l	% difference	c_d	% difference
FV	0.6308	-	0.0000	-
MV	0.6353	0.9	0.0000	-
TLS	0.6178	2.1	0.0009	-
PLS	0.6164	2.3	0.0011	-
RBF	0.6200	1.7	0.0010	-



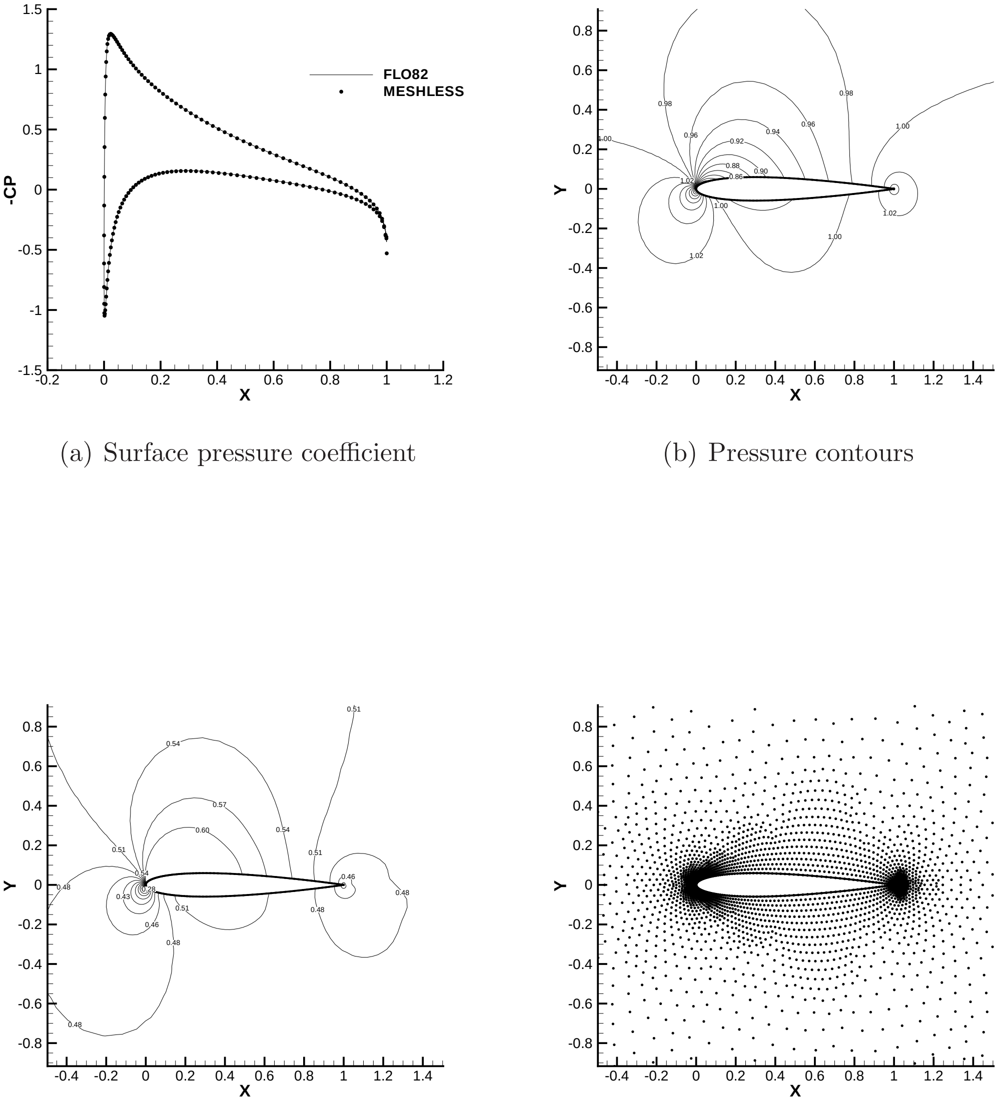
(a) Surface pressure coefficient

(b) Pressure contours

(c) Mach contours

(d) Point distribution

Figure 3.10: Flow over NACA 0012, $M = 0.50$, $\alpha = 0.0^o$, Meshless volume scheme.



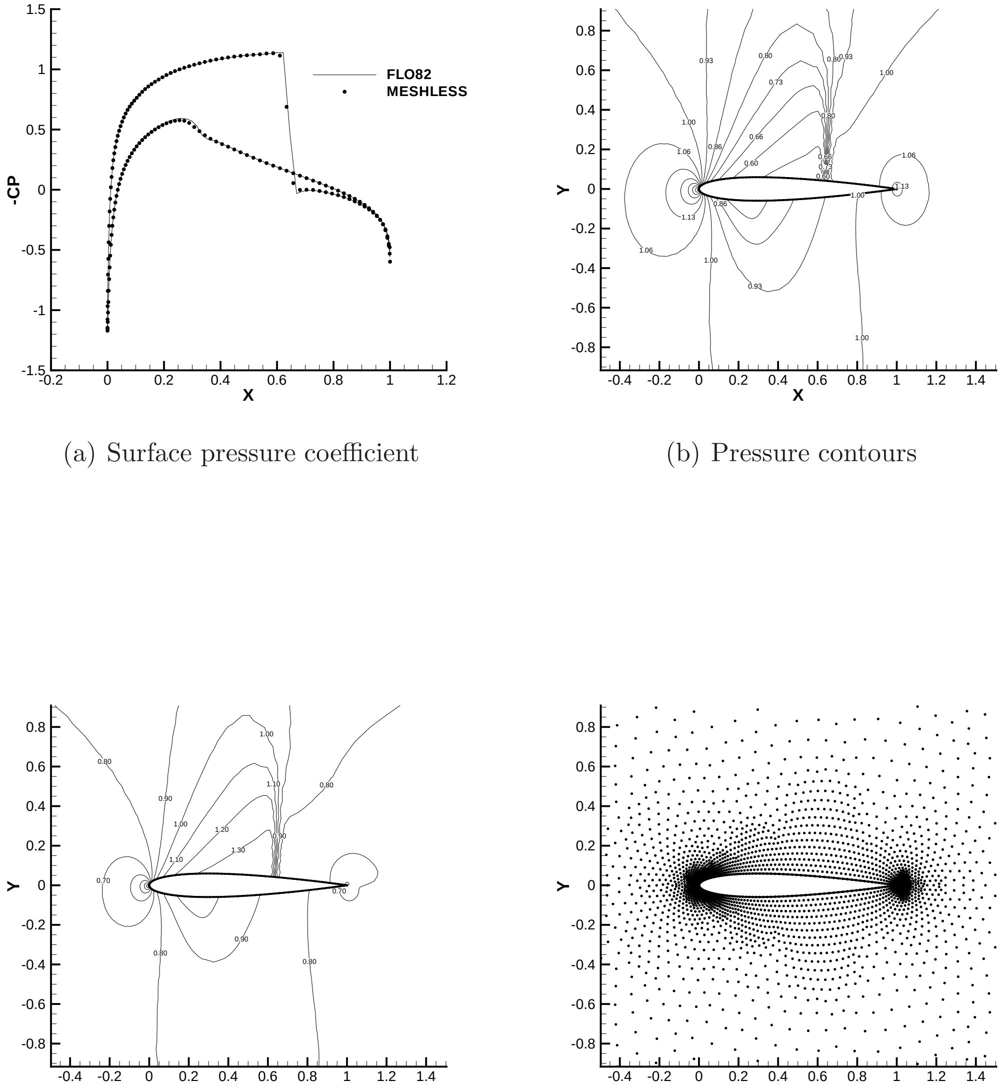
(a) Surface pressure coefficient

(b) Pressure contours

(c) Mach contours

(d) Point distribution

Figure 3.11: Flow over NACA 0012, $M = 0.50$, $\alpha = 3.0^o$, Meshless volume scheme.



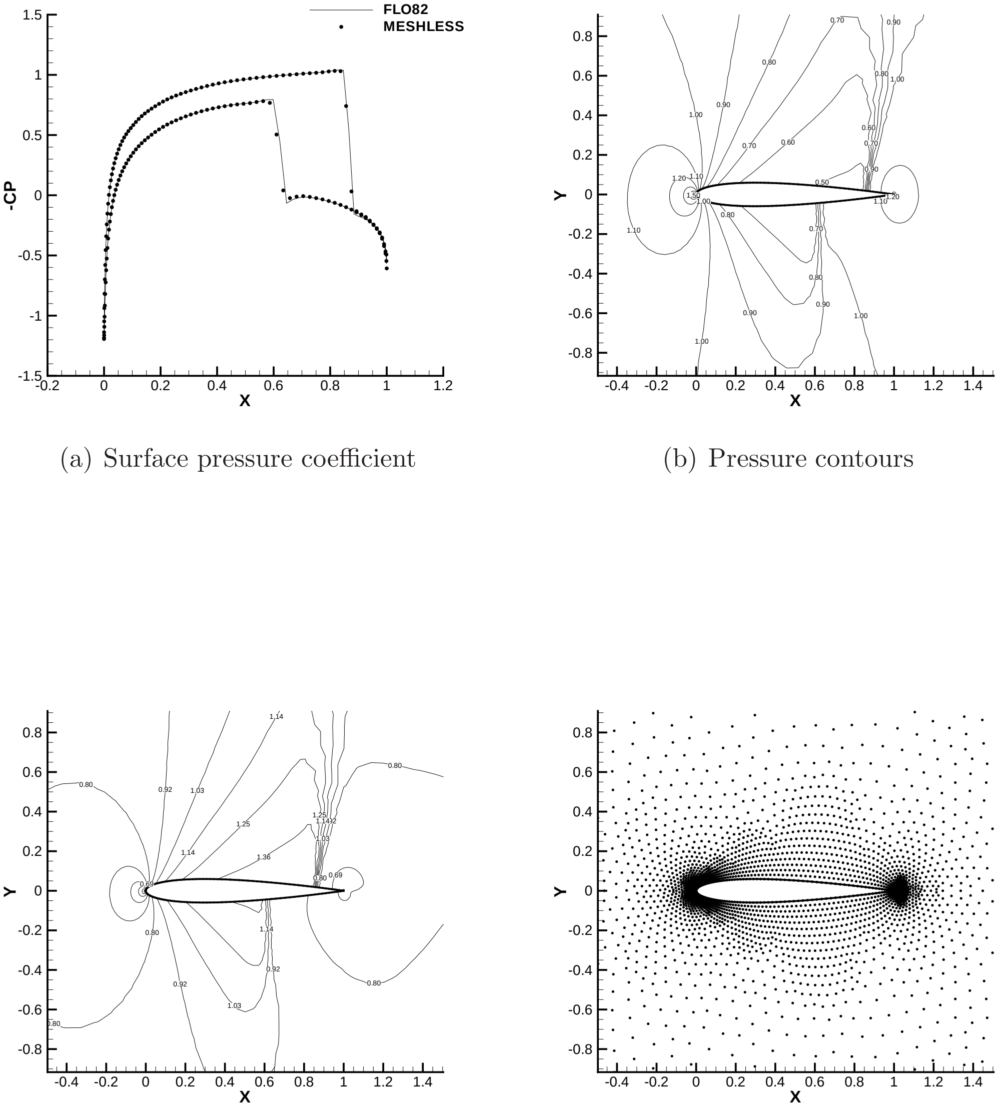
(a) Surface pressure coefficient

(b) Pressure contours

(c) Mach contours

(d) Point distribution

Figure 3.12: Flow over NACA 0012, $M = 0.80$, $\alpha = 1.25^o$, Meshless volume scheme.



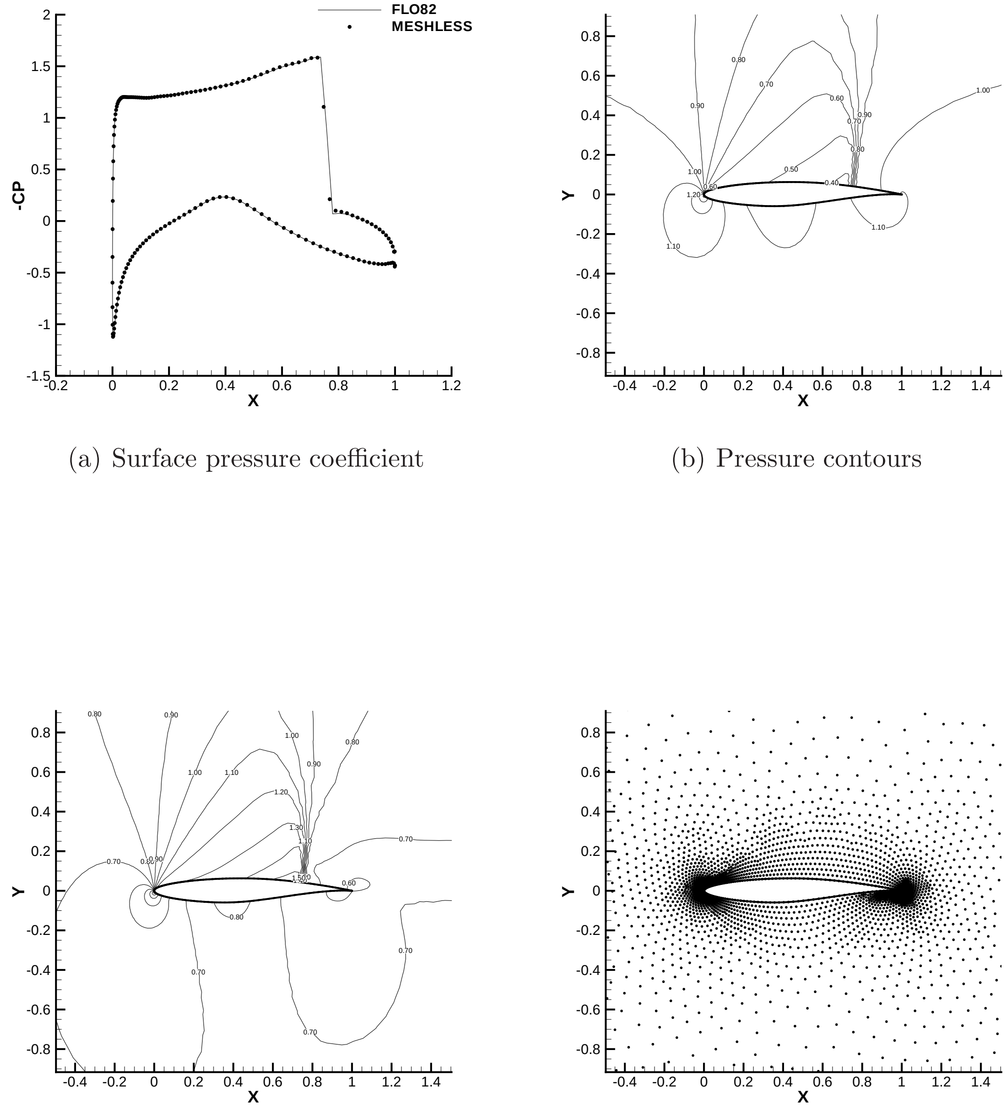
(a) Surface pressure coefficient

(b) Pressure contours

(c) Mach contours

(d) Point distribution

Figure 3.13: Flow over NACA 0012, $M = 0.85$, $\alpha = 1.0^o$, Meshless volume scheme.



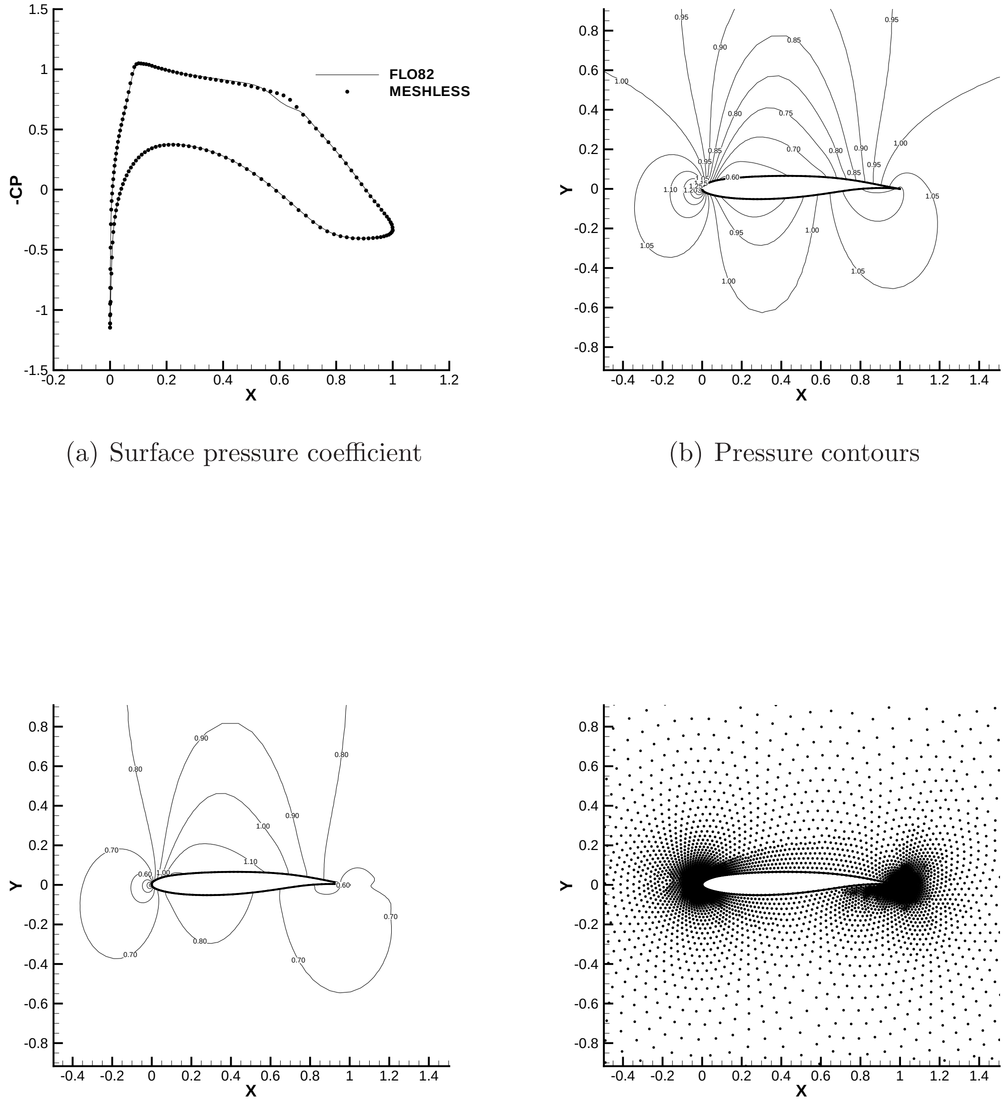
(a) Surface pressure coefficient

(b) Pressure contours

(c) Mach contours

(d) Point distribution

Figure 3.14: Flow over RAE 2822, $M = 0.75$, $\alpha = 3.0^o$, Meshless volume scheme.



(a) Surface pressure coefficient

(b) Pressure contours

(c) Mach contours

(d) Point distribution

Figure 3.15: Flow over Korn airfoil, $M = 0.75$, $\alpha = 0.0^o$, Meshless volume scheme.

Chapter 4

Multicloud

The increasing use of computational fluid dynamics for engineering design and anaylsis demands highly efficient solution methods. To address the need for efficiency, multigrid algorithms have been developed over the past few decades to improve the convergence rate of iterative solvers. The amount of research into multigrid algorithms has increased dramatically in recent years, as reported by Wesseling [84]. Fedorenko [85] formulated first true “multigrid” algorithm for the standard five point discretization of the Poisson equation on a square. He showed that multigrid is truly an $O(N)$ work algorithm. Brandt [86] formulated multigrid algorithms for linear and non-linear PDEs of elliptic and mixed type. His approach was practical, with applicability to aerodynamics problems. Ni [87] first applied multigrid to the Euler equations, including transonic flow applications. Furthering the work of Ni, Jameson [79, 88] combined multigrid with a multistage explicit scheme, which was designed to efficiently damp high frequency modes.

While the aforementioned work was applied to structured grids, unstructured multigrid algorithms were also formulated in pioneering works by Perez [89], Löhner and Morgan [90], Mavriplis and Jameson [63], and Lallemand, Steve, and Dervieux [91]. While not a problem with the structured algorithms, a major difficulty for unstructured multigrid is efficiently obtaining coarse grids. As Venkatakrishnan and Mavriplis [92] point out, there have been three approaches to generating coarse grids:

(1) begin with a coarse grid, and uniformly refine the mesh [89, 93], (2) use non-nested coarse grids which are independently generated [63], (3) or use agglomeration techniques [92]. The first option allows little control over the fine mesh and may not adequately resolve the geometry since the fine mesh relies entirely on a "starter" coarse mesh. The second option places a burden on the user to generate coarse meshes. This is not feasible when the user is handed a fine mesh only without access to the underlying geometry. Because of its automation, agglomeration is the most practical and widely used method of generating coarse grids. However, agglomeration still has drawbacks. For example, the fine grid algorithm is often generalized to handle arbitrary polyhedra, resulting in a loss of accuracy and efficiency. Also, algorithms used to generate coarse volumes can become quite complex. Additionally, edge fusing is needed to eliminate the multi-faceted nature of the coarse volumes. The procedure is often ad-hoc, lacking robustness.

The multicloud algorithm described in this chapter provides a simple and efficient means of obtaining coarse levels and operators for unstructured and mixed meshes. Multicloud possess inherent advantages over agglomeration because it relies upon a robust meshless discretization, which uses point clouds in place of a mesh. This allows for greater flexibility in the definition of coarse levels. Like agglomeration, the coarsening procedure developed in this work is fully automatic. However, multicloud coarsening is much simpler to implement than agglomeration, relying on simple rules to obtain coarse levels. Moreover, multicloud appears to work well for a variety of fine level discretization schemes. This means that multicloud may be utilized with no modifications to fine level schemes, allowing researchers freedom to develop accurate numerical schemes completely independent of the convergence acceleration technique. This modular aspect of multicloud represents a powerful shift in the way iterative schemes may be developed.

The power of multicloud is derived from the robust radial basis method of Chapter 2. While a number of excellent meshless schemes have been proposed in the literature [50, 52, 54, 47, 94, 53, 95, 55, 44, 51, 7, 6, 14, 96], their popularity has remained relatively low. This is most likely for two reasons: (1) meshless schemes have not been shown to relieve mesh generation difficulties since point cloud generation seems to be nearly as difficult, and (2) no meshless scheme has been shown to be discretely conservative. As will be discussed, multicloud circumvents these two difficulties. First, point clouds for the meshless scheme are used only on coarse levels, which are derived entirely from the fine level connectivity. Therefore, no special point generation procedures are needed. Second, formal conservation on the coarse levels is irrelevant, since the meshless operator is only used to accelerate convergence. The fine level scheme drives the coarse level scheme, making the fine level solution entirely independent of the coarse level algorithm.

The chapter begins by describing an automatic coarsening procedure to obtain point clouds. Following the coarsening procedure, methods of prolongation and restriction on clouds of points are developed. Finally, two dimensional results of the Euler equations are presented and major conclusions regarding multicloud are drawn. While the results of this work are limited to two dimensions, the power of the algorithm will be fully realized in three dimensions. However, the two dimensional work represented here has served as an invaluable proof of concept.

4.1 Multigrid Principles

One of the most popular multigrid algorithms for non-linear problems is the Full Approximation Storage (FAS) algorithm of Brandt [86]. In the FAS algorithm, the solution variables are first restricted from a fine cloud level, $k-1$, to a coarse cloud level, k, with a solution coarsening operator, $T_{k,k-1}$:

$$\boldsymbol{w}_k^{(0)} = T_{k,k-1}\boldsymbol{w}_{k-1}. \tag{4.1}$$

Likewise, the residuals are restricted to a coarse cloud level with a residual coarsening operator $Q_{k,k-1}$. A forcing function, $\boldsymbol{P}_k$, is computed such that

$$\boldsymbol{P}_k = Q_{k,k-1}\boldsymbol{R}_{k-1}(\boldsymbol{w}_{k-1}) - \boldsymbol{R}_k(\boldsymbol{w}_k^{(0)}). \tag{4.2}$$

The forcing function, $\boldsymbol{P}_k$, represents the difference between the aggregated fine cloud residuals, and the residuals computed with the coarse cloud solution. Subsequently,

$\boldsymbol{R}_k(\boldsymbol{w}_k)$ is replaced by $\boldsymbol{R}_k(\boldsymbol{w}_k)+\boldsymbol{P}_k$ at each stage of the iterative time stepping scheme. In this manner, the coarse level iterations are driven by the fine level residuals. At convergence on the fine mesh, the coarse levels do nothing to alter the converged solution.

An iteration on a coarse level results in a corrected solution, $\boldsymbol{w}_k^+$. Coarse level corrections, based on the difference between the corrected solution and the original solution transfered from the fine level, are then interpolated back to the fine grid with a prolongation operator, $I_{k-1,k}$,

$$\boldsymbol{w}_{k-1}^+ = \boldsymbol{w}_{k-1} + I_{k-1,k}(\boldsymbol{w}_k^+ - w_k^{(0)}). \tag{4.3}$$

The above procedure is often invoked recursively on a series of increasingly coarser meshes with the effect of quickly damping high frequency errors on the fine mesh.

4.2 Coarse Level Algorithm

An important property of the FAS algorithm as described above is that the solution on the fine level is independent of the coarse level discretization scheme. Coarse levels are only used to accelerate convergence and have no influence on the spatial discretization of the fine level. This fact may be used to generate a coarse level scheme based on a meshless method. Here, the radial basis method of Chapter 2 was used on the coarse level since robustness was the primary consideration, not accuracy.

A stable meshless scheme for the coarse levels may be obtained by invoking the Local Extremum Diminishing (LED) principle of [71] as has already been shown in Chapter 3. The coarse level algorithm follows closely the meshless volume algorithm of Chapter 3, but with the radial basis metrics and first order scalar diffusion. First order scalar diffusion is sufficient since accuracy is of little concern on coarse levels. A diffusive term, d_{ij}, may be added to each node of a local cloud of the form

$$d_{ij} = |\lambda_{max}|(\boldsymbol{w}_j - \boldsymbol{w}_i), \tag{4.4}$$

with $\lambda_{max} = |a_{ij}u + b_{ij}v| + c\sqrt{a_{ij}^2 + b_{ij}^2}$, where u,v, and c are the Cartesian velocity components and speed of sound, respectively.

Boundary conditions may be enforced by reflecting interior nodes across normal lines of boundaries and injecting the resulting ghost nodes with solutions consistent with the physics of the problem [50]. For the Euler equations, slip conditions are enforced at solid surfaces, while one dimensional Riemann invariants are used to solve for the conditions at the far field.

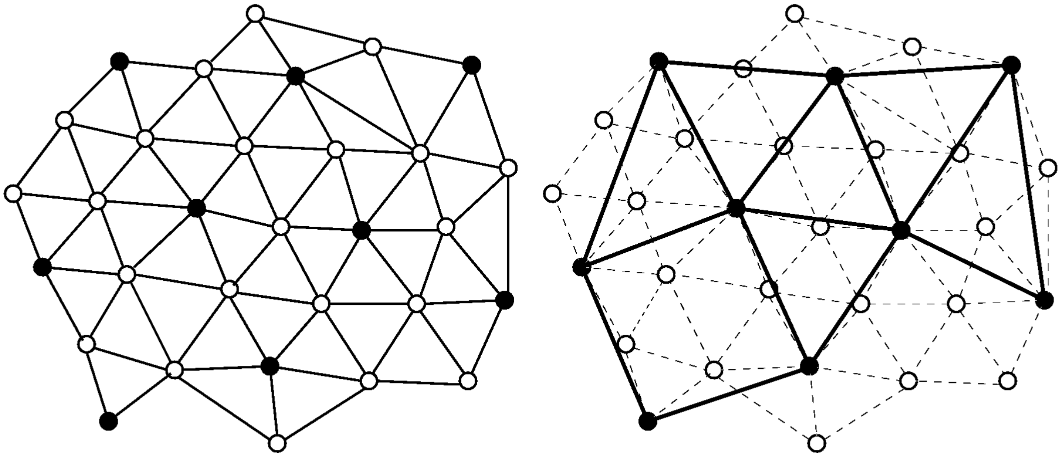
The result of the aforementioned spatial discretization is a coupled system of ordinary differential equations of the form

$$\frac{\partial \boldsymbol{w}_i}{\partial t} + R_i = 0, \quad R_i = Q_i - D_i. \tag{4.5}$$

Here, the convective (Q) and diffusive (D) portions of the residual (R) may be integrated separately using the modified Runge-Kutta scheme of [11] to effectively damp high frequency error. In addition, local time stepping, enthalpy damping, and implicit residual smoothing may be used to further accelerate convergence.

4.3 Multicloud Coarsening Procedure

The multigrid algorithm described in the previous section is designed to damp high frequency errors present in a discretized system of equations on a mesh. In this work, no assumption is made on the mesh topology of or the specific spatial discretization of the fine level. Instead, a general procedure for obtaining coarse levels of point clouds automatically from any mesh is developed. Generality is achieved by noting that any mesh in any number of spatial dimensions may be expressed as an edge-vertex list. Actually, meshless point clouds may also be expressed implicity using edges if the two verticies forming an edge belong to the local clouds of one another [68]. This fact enables a two-step recursive coarsening procedure which is simple and robust.



(a) Subset of fine level points. (b) Local cloud definition for coarse level.

Figure 4.1: Point coarsening procedure.

4.3.1 Obtaining a Coarse Subset of Points

The first step of the coarsening procedure is illustrated in Figure 4.1(a). Each fine level point is first initialized as "valid." The list of fine level points is then traversed. If a fine level point is not blanked, the nearest neighbors in its local cloud are blanked. The next point is then tested, until all the points have been traversed and labeled "valid" or "blanked." The set of valid points is not independent of the order in which the points are traversed. In order to preserve the domain boundaries as much as possible, the boundary nodes are traversed first, followed by the interior nodes. For two dimensional boundaries, this results in the selection of every other fine level node for the coarse level boundary definition. The result of this simple procedure is a subset of the fine level nodes, illustrated as the filled nodes in Figure 4.1(a), whose nearest neighbors are all blanked. This subset of nodes forms the next coarse level.

4.3.2 Forming Local Clouds on Coarse Levels

The second step in the coarsening procedure, illustrated in Figure 4.1(b), is to group the coarse level points into local clouds upon which to apply the radial basis method. While most local cloud definition procedures have focused on nearest neighbor search methods, such methods can incur significant overhead for complex problems. A simple and much cheaper approach based on edges is taken here. An edge in the meshless sense is simply a connection between two nodes which contain eachother in their local clouds. A global list of edges completely defines local cloud definitions. For each blanked point on the fine level, the valid nearest neighbors in its local cloud are connected to form an edge for the coarse level. It is possible that the blanking procedure performed in the first step could result in a single valid point in the local cloud of a blanked point on the fine level. In this case, the status of the blanked point is changed to "valid," and an edge between this new valid point and the single valid nearest neighbor is added to the coarse level. The above procedure is much simpler and faster than agglomeration techniques and is entirely mesh transparent. For cases tested so far the procedure has shown to be robust, producing good quality point distributions up to five levels deep.

4.4 Multicloud Transfer Operators

Of course, suitable coarse levels upon which to apply the meshless method are only part of the multicloud framework. Additionally, prolongation and restriction operators between successive levels must be defined. In the context of multicloud, the finest level, which is actually a mesh, is considered a collection of point clouds and treated exactly the same as the coarse levels. This establishes a common method of transfer for all levels. The only difference present on the fine mesh is in the case of a cell-centered scheme, in which nodal values of the solution and residuals should first be obtained via conservative volume weighting procedures.The three operators, needed for the FAS algorithm, T, Q, and I, will now be described. The operations are illustrated in Figure 4.2 in detail.

4.4.1 Restriction in Multicloud

First, the solution coarsening operator, T, is simply an injection of the fine level solution to coarse level points since all coarse level points are coincident with fine

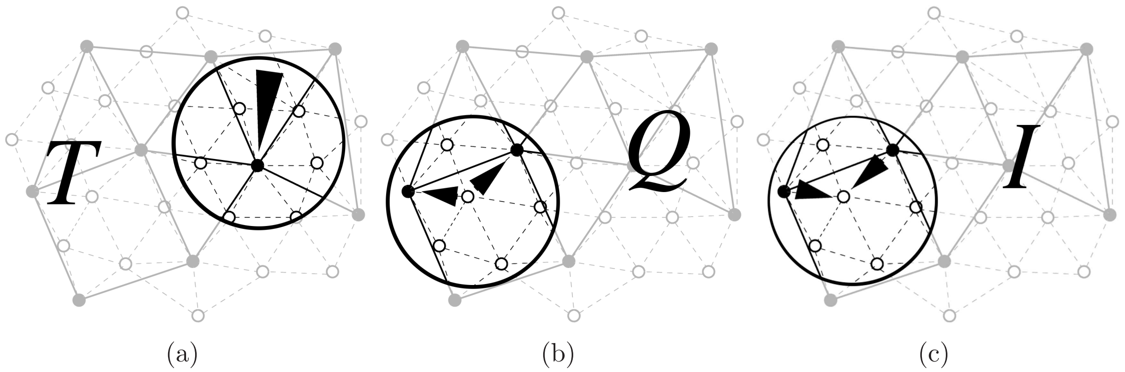


Figure 4.2: Illustration of multicloud transfer operators. Circles highlight the transfer operation described.

level points. This is a consequence of the fact that coarse level points are subsets of fine level points. The injection process is shown in Figure 4.2(a).

The residual restriction operator involves coincident and non-coincident points. It can be seen from Figure 4.1 that all fine level points are either coincident with a coarse level point, or have at least two coarse level points as nearest neighbors in their local clouds. For fine level points, i, coincident with or neighboring a coarse level point, j, inverse distance weight coefficients may be defined:

$$c_i = \begin{cases} |\boldsymbol{r}_i - \boldsymbol{r}_j|^{-1} & \text{for} \quad i \neq j \quad \text{(not coincident)} \\ 1 & \text{for} \quad i = j \quad \text{(coincident).} \end{cases} \tag{4.6}$$

Both the residual transfer operator and the correction prolongation operator rely on the distance weights of Equation 4.6. The residual transfer operator involves distributing the residual at each fine level point to the valid coarse level points in its local cloud with a weighting, as shown in Figure 4.2(b). The aggregated coarse level residual at node j, denoted $(Q\boldsymbol{R})_j$, may be obtained from the coincident and surrounding fine level residuals, $\boldsymbol{R}_j$ and $\boldsymbol{R}_i$, from

$$(Q\boldsymbol{R})_j = \alpha_j \boldsymbol{R}_j + \sum_i \beta_i \boldsymbol{R}_i, \tag{4.7}$$

where $\alpha_j = (ds_{fj}/ds_{cj})^2$ is the ratio of the average edge lengths of fine and coarse clouds at the coincident node, j, and

$$\beta_i = \left(\frac{1 - \alpha_j}{\sum_i c_i} \right) c_i \tag{4.8}$$

is a normalized inverse distance weight. The residual at the coincident node is scaled with the ratio of the average edge lengths to obtain the correct amount of residual on the coarse level. This is similar to nodal restriction for a Cartesian mesh, shown in Figure 4.4.1. The weight on the coincident node is 1/4, which would be the value of α_j for a Cartesian mesh. Also, from Figure 4.4.1, it can be seen that the corner nodes, which are further away from the side nodes, carry less weight, contributing 1/16 instead of 1/8 of their residuals. The inverse distance weights of Eq.(4.6) attempt to mimic this behavior. Additionally all the weights influencing a coarse level node in the figure sum to one. This same property can be verified by noting that $\alpha_j + \sum_i \beta_i = 1$ for the meshless transfer coefficients.

4.4.2 Prolongation in Multicloud

The prolongation operator is similar to the residual restriction operator, with the exception that the direction of transfer is coarse to fine instead of fine to coarse, as shown in Figure 4.2(c). A corrected fine cloud solution at fine level node i is computed with a weighted sum over the coarse level points, j, in its local cloud by

$$\boldsymbol{w}_i^+ = \boldsymbol{w}_i + \frac{\sum_j c_j (\boldsymbol{w}_j^+ - \boldsymbol{w}_j^{(0)})}{\sum_j c_j}. \tag{4.9}$$

If the fine level point, i, is coincident with a coarse level point, the corrections are a simple injection since $c = 1$ for coincident points. Otherwise, corrections are weighted based on distance.

It can bee see that all coefficients are based on weights which may be computed in a preprocess step and stored in a linked list for use in transfer subroutines. It should be noted that no search algorithms are used in either the coarsening procedure or any

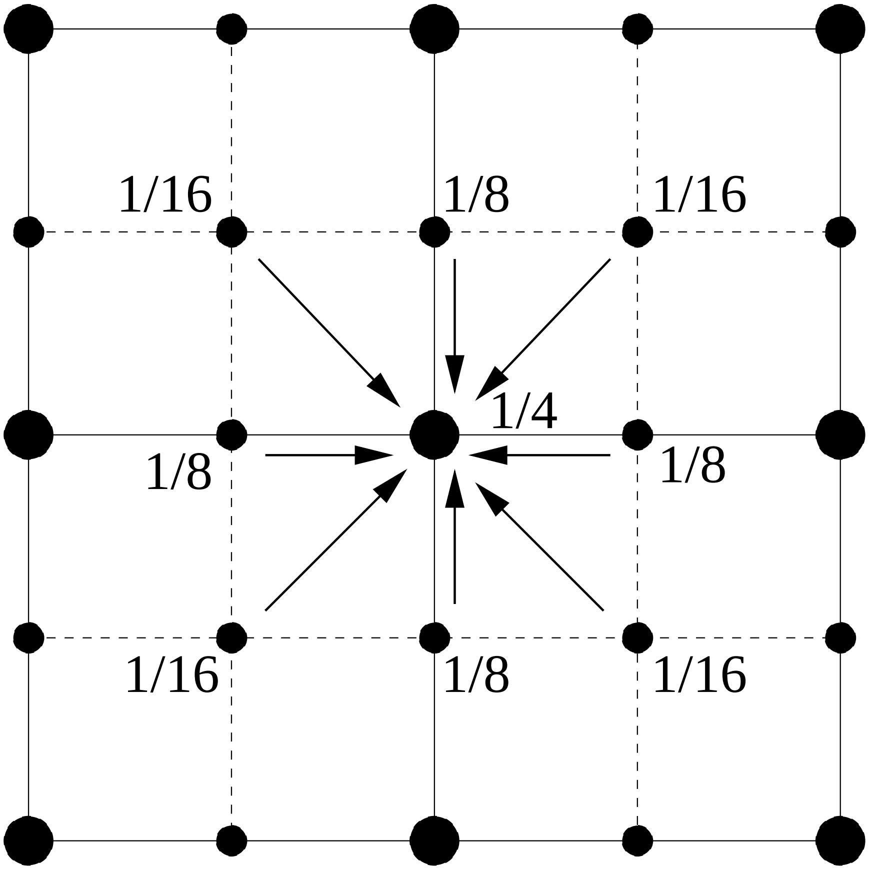


Figure 4.3: Node centered residual restriction weights for a Cartesian mesh.

of the multicloud transfer operators. This makes the multicloud procedure extremely fast to set up for an arbitrary mesh.

4.5 Results with Multicloud

To highlight the capabilities of multicloud, two dimensional Euler computations for airfoils are given. First, the results of the automatic coarsening procedure for a NACA 0012 airfoil containing 5903 nodes is shown in Figure 4.4. The fine level is an unstructured mesh obtained from the *delaundo* package [97]. The coarsened global point distributions are shown in the right column, while the edge connectivity, which defines local clouds is shown in the left column. Levels 2-4 are not meshes, but collections of edges. A close inspection of the coarse level edges, shows edges crossing and other features unacceptable for a mesh. This, however, presents no problem for the meshless method, which easily accomodates this more general definition of connectivity. It should be noted that the number of nodes coarsens by roughly a factor of three between successive levels for this case. This is significantly lower than the ideal coarsening factor of four obtained from structured coarsening. While not ideal, the efficient damping of the meshless operator seems to compensate for the non-ideal coarsening, resulting in excellent convergence rates, as will be shown. However, it would be desirable to more closely obtain the ideal coarsening to save on memory. Future research will focus on obtaining more ideal coarsening rates.

Multicloud was used to acclerate three fine level discretization schemes, illustrated in Figures 4.5(a)-4.5(c). While the spatial discretization varied among these three schemes, the method of achieving steady state was held constant to test the multicloud convergence properties. In all cases, the modified Runge-Kutta scheme of Jameson [11] was used in conjunction with multicloud. The first scheme tested was the meshless volume (MV) scheme of Chapter 2, as shown in Figure 4.5(a).

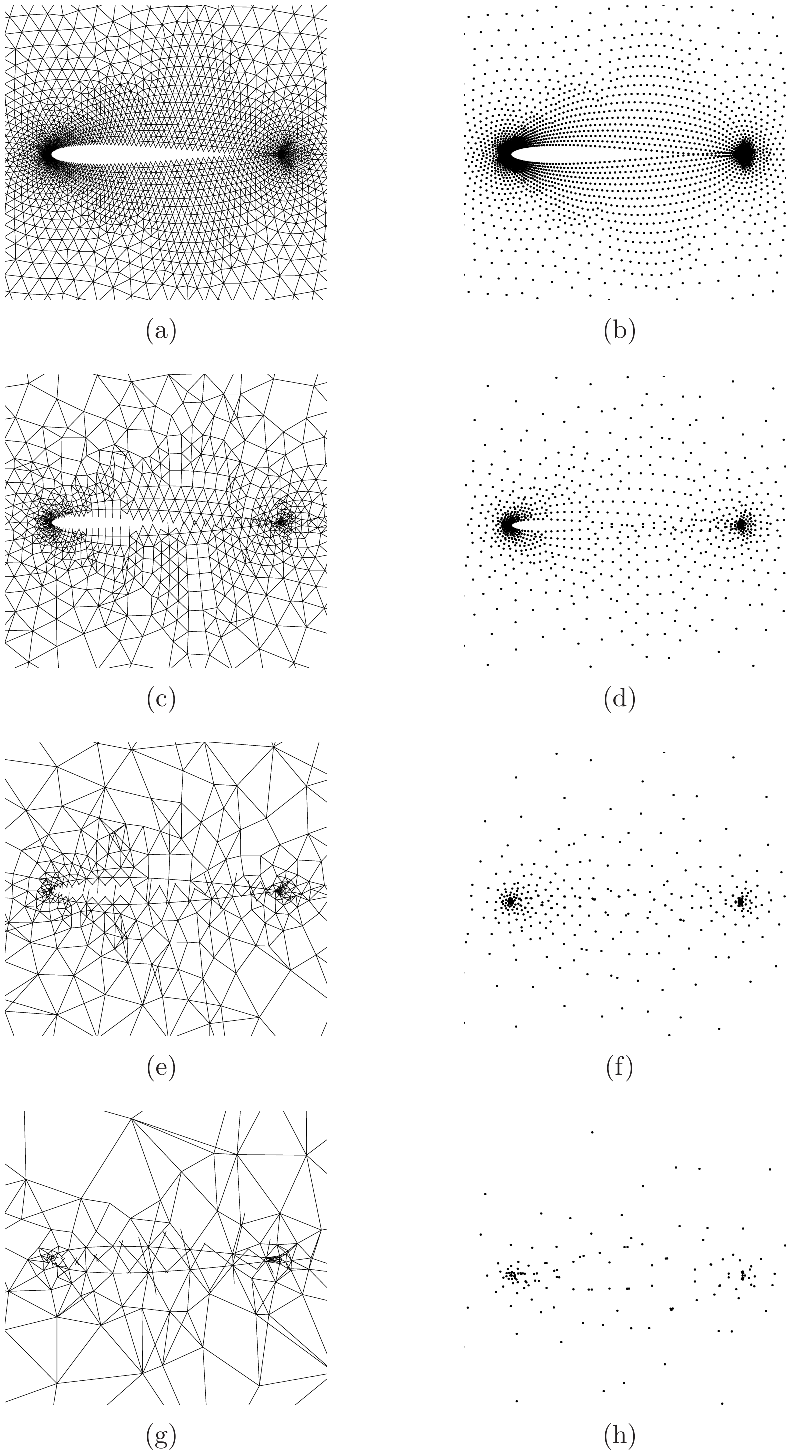
The second fine level scheme which used multicloud was a nodal finite volume (NFV) scheme, illustrated in Figure 4.5(b). The NFV scheme was implemented similar to the unstructured scheme of Jameson [71], in which nodal volumes are taken to be the union of triangles sharing a common node. For a node 0 surrounded by neiboring nodes k, as defined by the triangulation, the semi-discrete form may be expressed as

$$V_0 \frac{d\boldsymbol{w}_0}{dt} + \sum_k (\boldsymbol{F}_{k,k-1} - \boldsymbol{d}_{k0}) = 0, \tag{4.10}$$

where the directed flux, $\boldsymbol{F}_{k,k-1}$, is defined as

$$\boldsymbol{F}_{k,k-1} = \frac{1}{2}(\boldsymbol{g}_k + \boldsymbol{g}_{k-1})(x_k - x_{k-1}) - \frac{1}{2}(\boldsymbol{f}_k + \boldsymbol{f}_{k-1})(y_k - y_{k-1}).$$

The diffusive flux, $\boldsymbol{d}_{k0}$, is also based on the CUSP scheme and is proportional to the metrics $\frac{1}{2}(x_{k+1} - x_{k-1})$ and $\frac{1}{2}(y_{k+1} - y_{k-1})$, as shown in Figure 4.5(b).



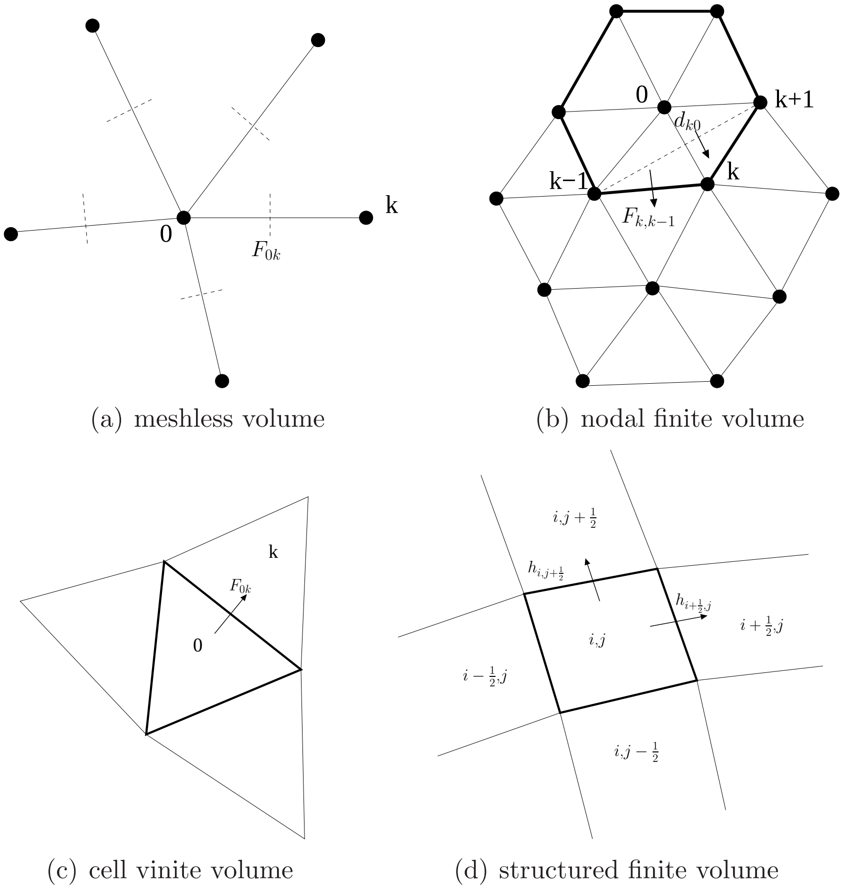
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 4.4: Results of coarsening procedure for NACA 0012, showing three coarsened cloud levels and corresponding edge connectivity. Number of nodes: level 1–5903, level 2–1941, level 3–728, level 4–282.



(a) meshless volume

(b) nodal finite volume

(c) cell vinite volume

(d) structured finite volume

Figure 4.5: Illustration of multicloud test schemes.

The third fine level scheme tested with multicloud was a cell finite volume (CFV) scheme, as shown in Figure 4.5(c). In the CFV scheme, the flow variables are stored at the cell centers, and the semi-discrete form of the Euler equations at cell 0 surrounded by neighboring cells k, may be expressed simply as

$$V_0 \frac{d\boldsymbol{w}_0}{dt} + \sum_k \boldsymbol{F}_{0k} = 0, \tag{4.11}$$

where the directed flux,

$$\boldsymbol{F}_{0k} = \frac{1}{2}(\boldsymbol{F}_0 + \boldsymbol{F}_k) - \frac{1}{2}\boldsymbol{d}_{0k},$$

Table 4.1: Comparison of convergence rate per unit work for the unstructured NFV scheme with multicloud versus the FV scheme with nested multigrid.

	Multicloud (NFV)		Multigrid (FV)	
Case tested	single grid	4-levels	single grid	4-levels
NACA 0012, $M = 0.5$, $\alpha = 3.0^o$	0.981	0.872	0.982	0.880
NACA 0012, $M = 0.85$, $\alpha = 1.0^o$	0.989	0.916	0.990	0.926
KORN, $M = 0.75$, $\alpha = 0.0^o$	0.985	0.945	0.987	0.940

is the average of the fluxes sharing a common edge augmented with an artificial diffusion term, which is again based on the CUSP scheme. Reconstruction of left and right states is performed by first transfering the solution to the nodes with a conservative weighting procedure. Subsequently, estimates of the change in the solution on either side of a given edge may be obtained from the nodal values opposite the edge.

A structured finite volume (FV) scheme using structured multigrid, shown in Figure 4.5(d), was used as a baseline for comparison. The FV scheme was implemented using the conservative numerical flux approach of [71, 76], as described in Equation 3.25.

The dramatic effect of multicloud accleration using four levels for the NFV scheme may be seen in Figures 4.6-4.8. Each figure shows surface pressure, residual history, and convergence of both lift and drag. For all cases tested, multicloud reduced the amount of time to attain steady state lift and drag within 1% by a factor of around 20. The performance of multicloud for these three cases was compared with the FV structured multigrid algorithm. The results of this comparison are shown in Table 4.1. As the table indicates, convergence rates per unit work with four levels of multicloud are comparable to those obtained with the FV scheme. This is particularly promising, since the FV scheme is structured and often gives “textbook” multigrid acceleration. It should be emphasized that since convergence rates are expressed per unit work, the effect of the non-ideal coarsening of Figure 4.4 has been factored into the comparison, showing no adverse effects.

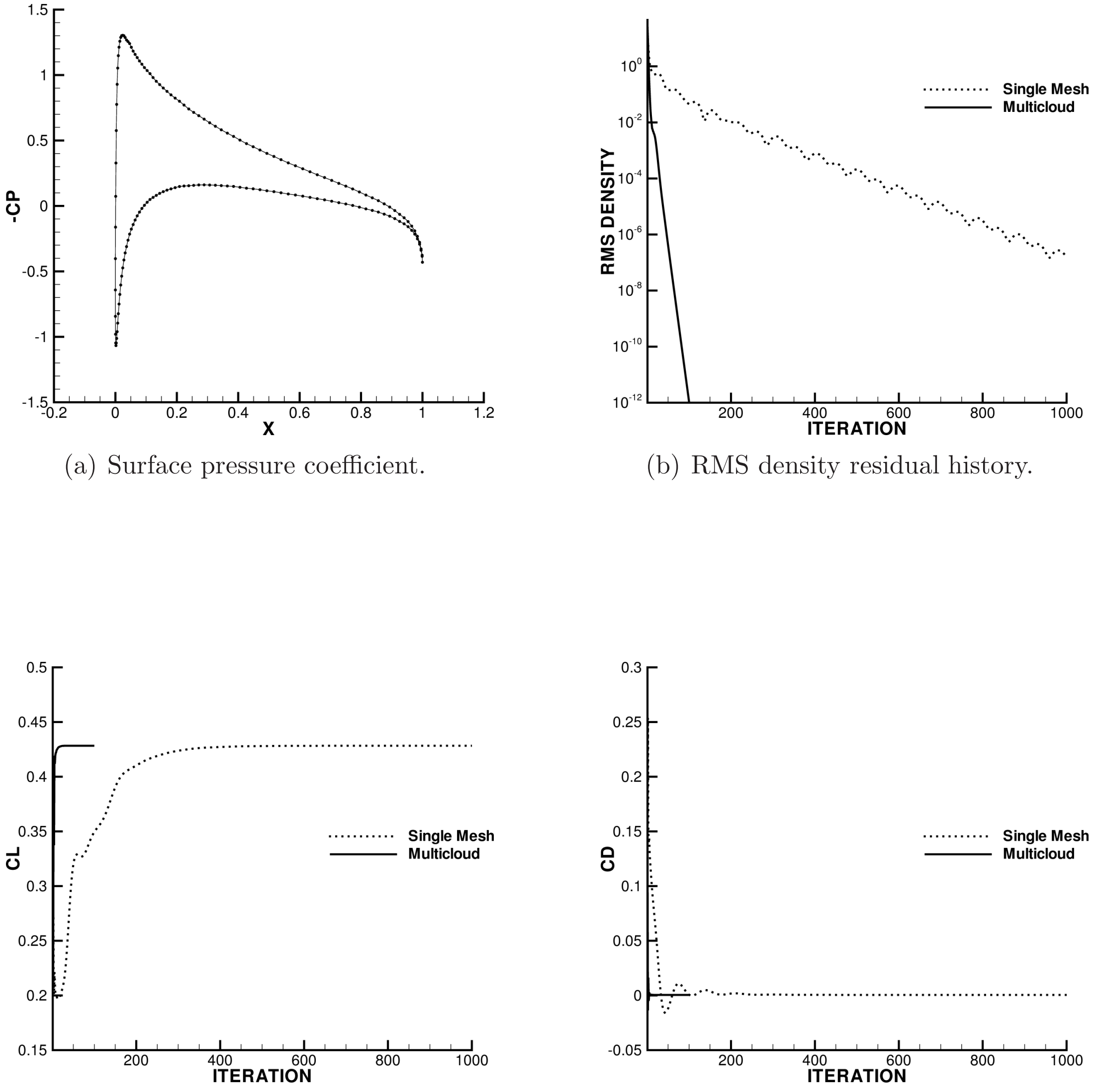
Finally, Table 4.2 highlights the fact that multicloud works well for a variety of fine level schemes. A variety of schemes were tested using both multicloud and

Table 4.2: Comparison of convergence rate per unit work for various fine level schemes for the NACA 0012, $M = 0.8$, $\alpha = 1.25^o$.

fine level scheme	convergence accelerator	single grid	4-levels	overhead
MV	multicloud	0.983	0.921	79%
NFV	multicloud	0.986	0.922	71%
CFV	multicloud	0.990	0.955	63%
FV	nested multigrid	0.988	0.932	50%
CFV	non-nested multigrid	0.992	0.959	38%

multigrid for the NACA 0012 at $M = 0.8$ and $\alpha = 1.25^o$. The three schemes used with multicloud included the MV scheme, the NFV scheme, and the CFV scheme. Multicloud was effective at accelerating convergence with all schemes. Wile the cell-centered scheme was significantly slower than the meshless or nodal schemes, this is more likely a property of the scheme itself rather than multicloud, since the CFV scheme with non-nested multigrid showed similar behavior. Once again, convergence acceleration was as dramatic for multicloud as for the “textbook” nested multigrid observed with the FV scheme. The overhead of the convergence accelerator was also measured. The overhead was defined as the CPU time to perform all coarse level iterations and transfers, expressed as a percent of the fine level CPU time per iteration. The overhead of using four levels of multigrid or multicloud associated for each scheme is also shown in Table 4.2. The higher overhead associated with multicloud is largely a reflection of the non-ideal coarsening, not the inefficiency of the meshless method. However, convergence rates per unit work are still excellent for this case in spite of the non-ideal coarsening. The most notable feature of multicloud is its transparent ability to accelerate convergence for a variety of fine level schemes.

In conclusion, multicloud represents a powerful convergence accelerator for iterative schemes, which appears to work equally well for a variety of fine level algorithms. The scheme transparency of multicloud will allow researchers to develop accurate schemes independent of any convergence accelerator, since multicloud is a versatile, modular algorithm. Since no assumptions are made regarding the topology of the fine level mesh, multicloud can be implemented on arbitrary mesh types in a mesh transparent manner. While coarsening procedures used in multicloud result in non-ideal coarsening ratios, this does not appear to degrade convergence when compared per unit work with established methods. The non-ideal coarsening appears to be a valid trade-off for the simplicity and automation offered from multicloud compared with agglomeration. The two dimensional results presented here indicate the vast potential of multicloud to accelerate complex three-dimensional cases in the future.



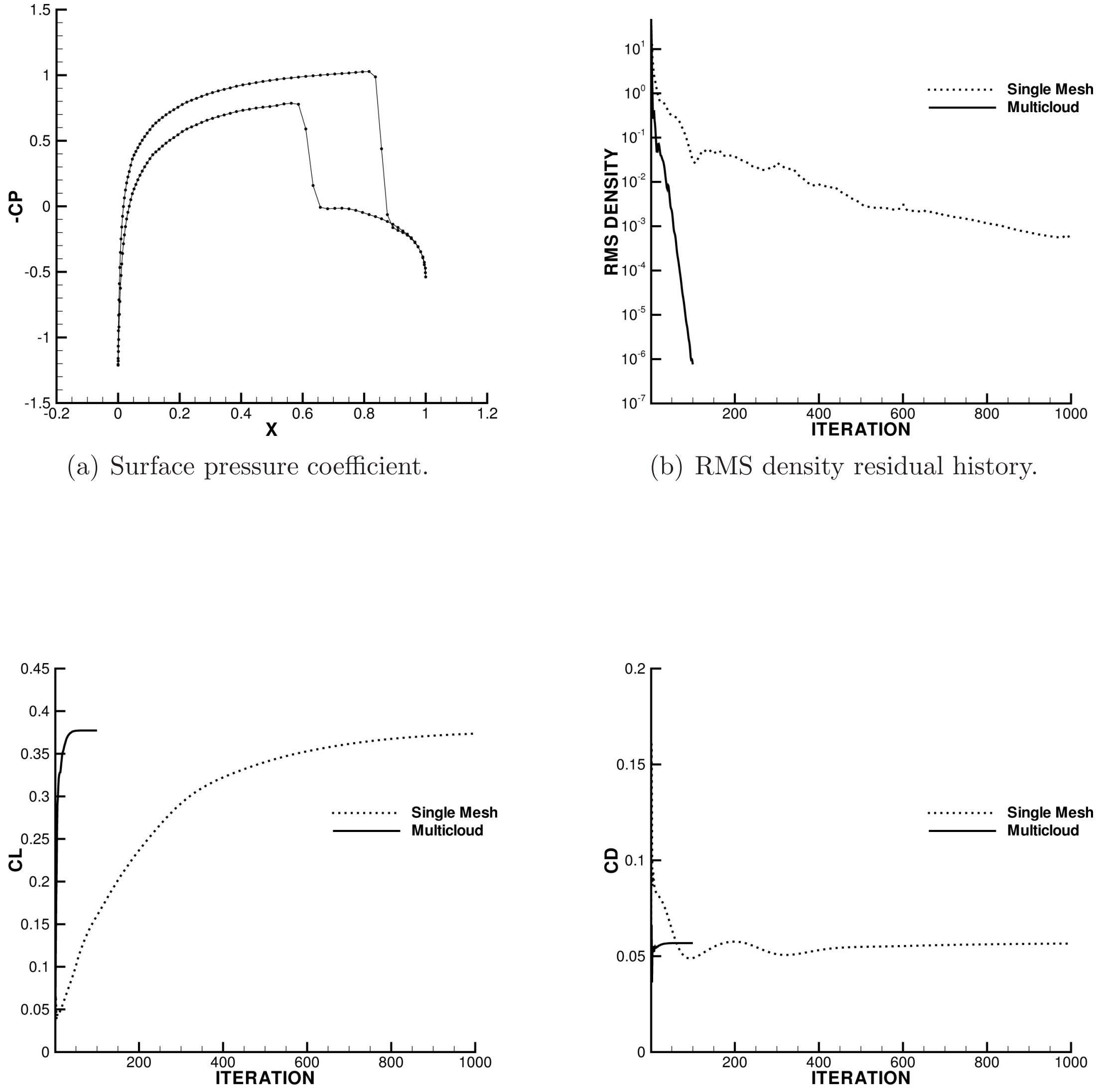
(a) Surface pressure coefficient.

(b) RMS density residual history.

(c) Convergence of lift.

(d) Convergence of drag.

Figure 4.6: Convergence of NFV scheme with multicloud for NACA 0012, $M = 0.5$, $\alpha = 3.0^o$.



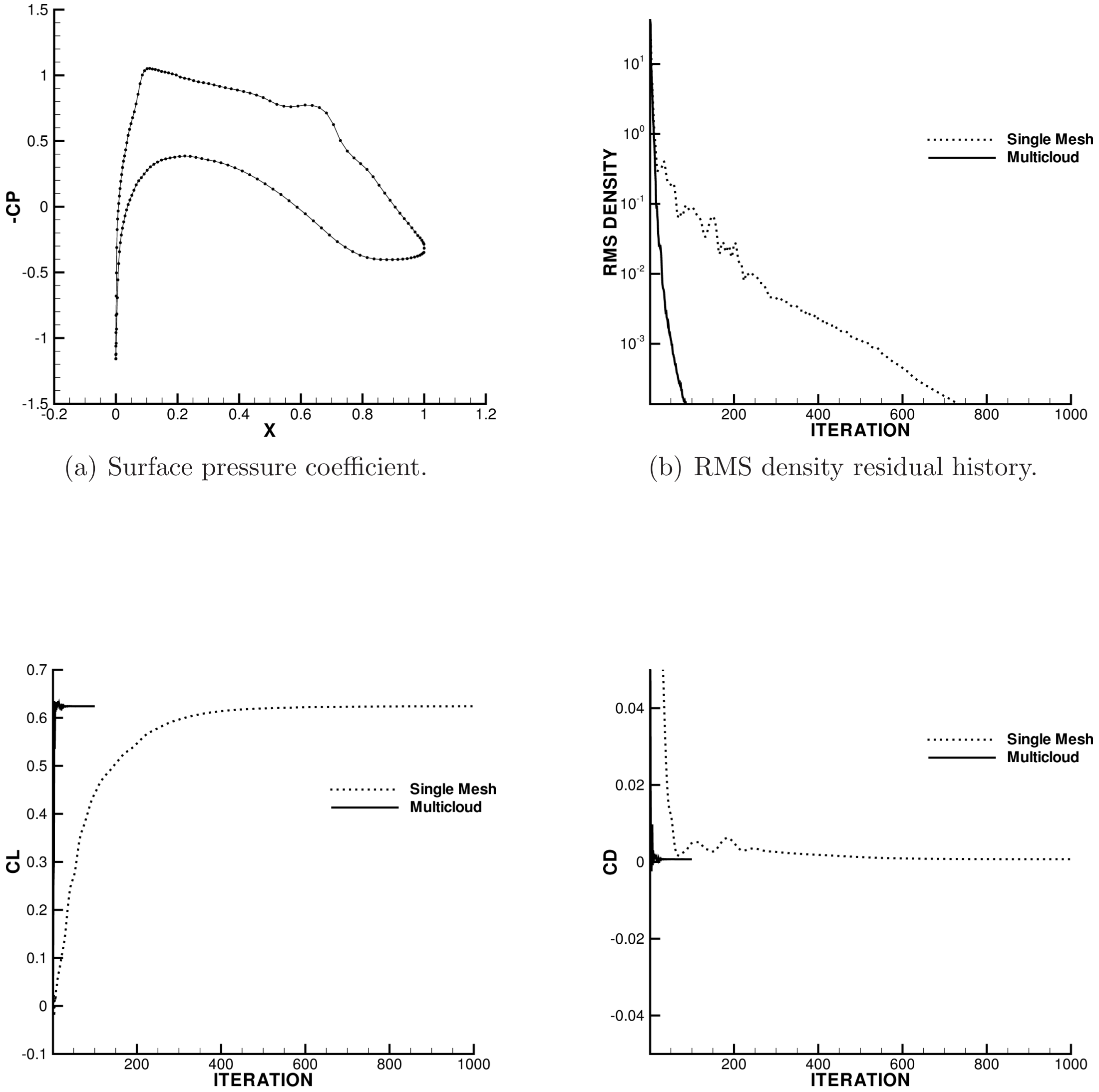
(a) Surface pressure coefficient.

(b) RMS density residual history.

(c) Convergence of lift.

(d) Convergence of drag.

Figure 4.7: Convergence of NFV scheme with multicloud for NACA 0012, $M = 0.85$, $\alpha = 1.0^o$.



(a) Surface pressure coefficient.

(b) RMS density residual history.

(c) Convergence of lift.

(d) Convergence of drag.

Figure 4.8: Convergence of NFV scheme with multicloud for KORN airfoil, $M = 0.75$, $\alpha = 0.0^o$.

Chapter 5

A Meshless Interface for Overset Grids

Besides multicloud convergence acceleration, another practical application of meshless methods involves interfacing grids in overset schemes. Overset schemes have been widely used to simulate flow around complex geometry with complex physical models. Overset schemes consist of simple overlapping grids which cover the domain in a patched fashion. The most daunting challenge with the overset approach is how to efficiently and accurately establish inter-grid communication between the various grid components. According to Meakin et. al [98], the state of the art as of 2007, is domain connectivity software that costs roughly 10-20% of a flow solver iteration for simulations accomodating relative motion. Moreover, domain connectivity software is quite complex, making it error prone and suffering robustness issues.

This chapter elaborates on a simple and efficient framework for using a meshless method along intergrid boundaries to accomplish domain connectivity for overset approaches. The flexibility of the meshless method allows for solution updates in the nebulous region between conventional grids, integrating the problem physics into grid interfaces. The search routines used to form local clouds of points in the interface region are simpler and cheaper than conventional donor/recipient identification. Moreover, the meshless interface approach is completely modular requiring no modification of neighboring grid schemes. The chapter begins by outlining the overset grid approach. Next, the meshless interface method is set forth, followed by a brief discussion of off-body and near-body solution schemes used to obtain the results in this chapter. Next, a method of using the computer language, *Python*, to implement the multi-solver code is discussed. Finally, the advantages of the meshless approach are highlighted, followed by two dimensional viscous results.

5.1 The Overset Grid Approach

The overset approach was first developed by Benek, Steger, and Dougherty [99, 100] as a method of accomodating complex geometry consisting of multiple bodies. They referred to their scheme as "Chimera" after the mythological Greek character composed of many different animal features. The motivation behind the Chimera schemes was that relatively simple grid topologies could be used around individual geometric features and then combined to create a global simulation. The result is a system of patched grids, which together cover the problem domain, as illustrated in Figure 5.1. The method was extended to three dimensions for the solution of the unsteady Navier-Stokes equations by many researchers, including Meakin [101]. Relative motion between various bodies was handled smoothly since grid components were allowed to slide past one another. Overset schemes have seen widespread use with popular codes such as *Overflow*, which has seen continual enhancements over the years [102].

The major difficulty in assembling a practical overset scheme is the efficient and accurate implementation of domain connectivity. Domain connectivity provides each individual grid in the domain with information about the surrounding flow, coupling the grid system. Conventionally, domain connectivity has been accomplished by linear interpolation along inter-grid boundaries, as originially proposed by Benek, Steger, and Dougherty [99, 100]. Efficient algorithms have been developed to identify donor and recipient nodes for use in interpolation [103]. Many researchers have pointed out that formal conservation is lost in the global sense when interpolation is used. However, Meakin [104] showed that with proper grid resolution, spatial and temporal accuracy may be maintained for unsteady problems. Alternatives to interpolation include triangulation of the interface region to maintain conservation, and hybrid grid

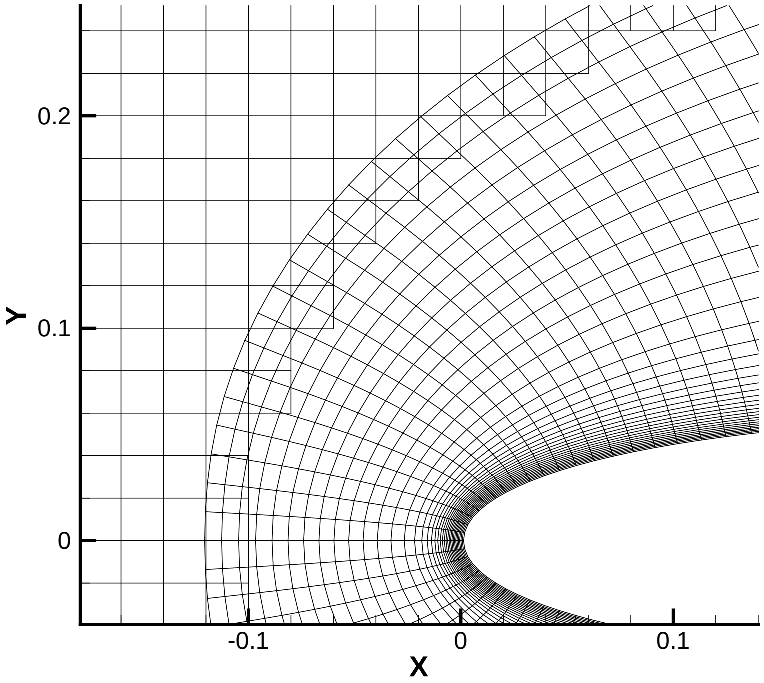


Figure 5.1: Overlapping grids in the overset approach.

systems which blend regions of different grid topologies [82, 105, 106] with arbitrary polyhedra.

5.2 Meshless Grid Interfaces

While interpolation has seen widespread use, the ideal domain connectivity algorithm would be more efficient and more accurate. In this section, a novel approach to domain connectivity is proposed using the meshless method described in Chapter 3. The meshless approach presents potential advantages in accuracy and efficiency over interpolation [69]. First, since the meshless method uses the fluid equations integrated in time just as other grid components, potential time lagging errors for unsteady problems with implicit integration [104] using large time steps could be alleviated to some degree. Accuracy may also be improved in the presence of solution discontinuities and high gradients, since the meshless method, unlike interpolation, obeys the proper characteristic propagation of information. In addition to accuracy, the task of defining a meshless cloud, as discussed in this section, appears to be less computationally demanding than donor cell identification, leading to gains in algorithm efficiency. Though many of these claims require further investigation in a variety of scenarios, preliminary results are encouraging.

5.2.1 Determining Meshless Clouds in the Interface Region

An efficient method for determining which points receive a meshless discretization and the selection of point clouds will now be described. In short, all points in all grid components of the overset system that do not have a complete stencil for their respective solution procedures are flagged as meshless. This means these points will receive a meshless spatial discretization, while all other points receive a discretization from their respective grid method. While overset grid simulations may contain multiple aerodynamic bodies each containing multiple near-body grids, as well as an arbitrary number of off body Cartesian blocks, the following procedure will be limited to a single body possessing a single near-body grid embedded in a single off body Cartesian block. Future work will be generalized to handle more complex test cases. The steps for determining the meshless points and clouds may be summarized as follows:

1. Determine near-body meshless points
2. Determine Cartesian meshless points
3. Determine near-body support points
4. Determine Cartesian support points

The above steps will now be described in detail and are illustrated in Fig.(5.2).

Determine near-body meshless points

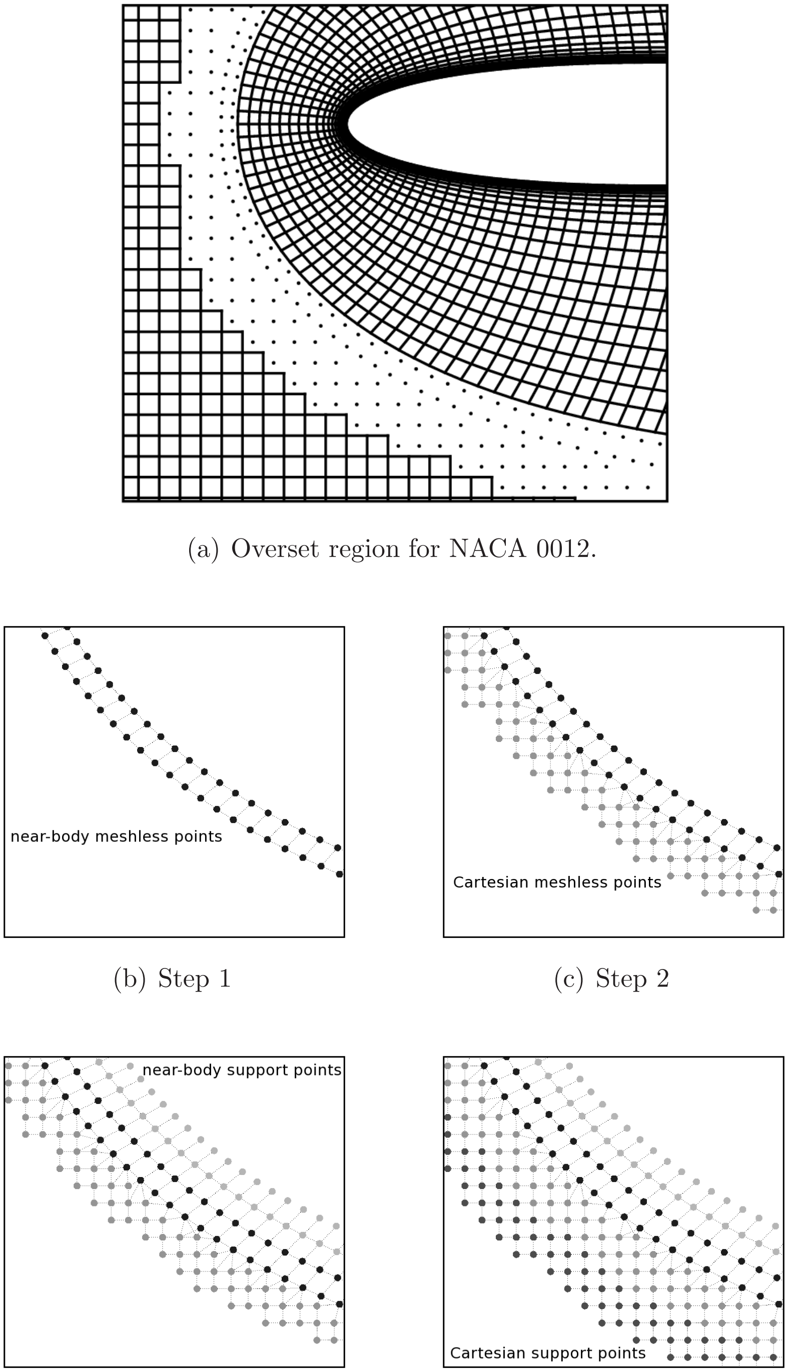
First, near-body meshless points are identified. Near-body meshless points are points within the near-body which do not have a complete finite volume stencil arising from domain boundaries. For conventional second-order methods, the near-body meshless points are the two outermost layers of the near-body grid system. Near-body meshless points inherit the connectivity that is available from the near-body, as shown by the edges in Figure 5.2(b). Once the near-body meshless points are identified, any number of Cartesian hole-cutting algorithms may be employed to blank cartesian cells which lie within the near-body domain. In this work, a ray-tracing based hole-cutting algorithm has been used [103].

Determine Cartesian meshless points

Second, Cartesian meshless points are identified, and their connectivity to the near-body meshless points is established. For each outermost near-body meshless point, the four Cartesian points which contain it are identified. The (I, J) coordinates of the lower left Cartesian point may be found from

$$I_1 = FLOOR\left(\frac{(x_n - x_L)}{dx}\right) \tag{5.1}$$

$$J_1 = FLOOR\left(\frac{(y_n - y_L)}{dy}\right). \tag{5.2}$$



(a) Overset region for NACA 0012.

(b) Step 1

(c) Step 2

(d) Step 3

(e) Step 4

Figure 5.2: Procedure for determining the meshless interface

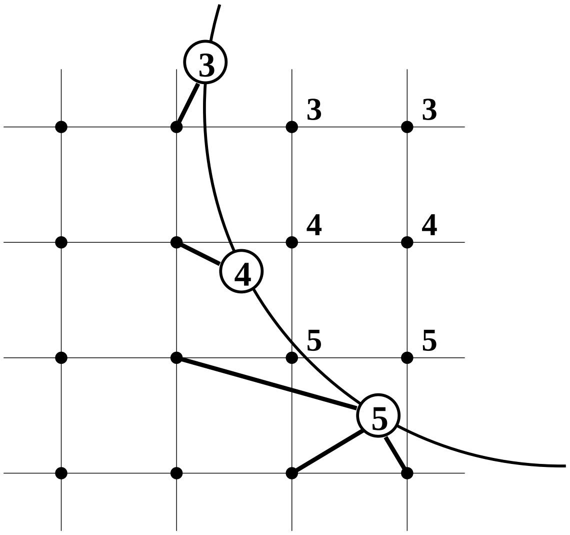


Figure 5.3: Connecting Cartesian points to near-body points.

Of the four Cartesian points containing the near-body point, those points that are not blanked from the hole-cutting procedure are flagged as Cartesian meshless points, and their connectivity to the near-body point which found them is established.

While the result of this procedure results in well-conditioned meshless clouds for all near-body points, in general there will remain Cartesian points along the hole-cut boundary which do not connect to any near-body point. These Cartesian points will remain poorly conditioned unless they are connected to nearby near-body meshless points. The procedure to establish these connections is detailed in Figure 5.3. First, the blanked points within the vicinity of the hole-cut surface are flagged with their nearest near-body meshless point. Next, connections are formed from the Cartesian points with incomplete stencils to the flag of the blanked Cartesian point. This procedure is extremely fast, since proximity searches may be done from the perspective of the near-body searching for Cartesian points in Cartesian index space, similar to Equation 5.1.

Determine near-body support points

The previous two steps involved the determination of near-body and Cartesian meshless points. These points receive a meshless discretization in space. The next two steps involve the determination of support points, which do not receive a meshless discretization, but merely provide boundary condition layers for the meshless points. The support points are owned by other grid components and receive the spatial discretization typical of their respective grids.

Near-body support points are determined by the requirement that each near-body meshless node must have a well-defined least squares stencil. Additionally, each of these nearest neighbors must have a well-defined least squares stencil. These requirements would suggest picking points in the near-body grid corresponding to the nodes in Figure 5.2(d). For structured near-body grids, determination of near-body support points is trivial. Again, near-body support nodes inherit the connectivity of the near-body grid.

Determine Cartesian support points

Similar to near-body support points, Cartesian support points do not receive a meshless discretization, but provide an adequate boundary region for the Cartesian meshless points. The selection of the Cartesian support points is trivial and is performed by marching away from the Cartesian meshless nodes along coordinate directions, until their meshless stencil requirements are met. The procedure results in adding the nodes shown in Figure 5.2(e). Once again the connectivity for these nodes is borrowed from the Cartesian mesh itself.

Once all meshless points, support points, and connecting edges have been identified, the condition number, κ, of the least squares matrix based on the Taylor approach in Equation 2.17 at each node is computed to assess local cloud quality. In practice, for $\kappa < 4$, stability and good convergence have been observed. The quality of local clouds as reflected in the condition number appears to be strongly correlated to the discrepancy in resolution of near-body and off-body grid components at the interface region. As the off-body spacing becomes more than a factor of about five larger or smaller than the near-body grid, the condition of local clouds is greatly compromised. Clearly, matching grid resolution in the interface is needed for stability and accuracy for the meshless approach just as for interpolation. This is consistent with the findings of Meakin [104] who found that accuracy for time accurate overset problems is most strongly influenced by mesh resolution of grid components in the overlap region.

From the above procedures it is clear that all algorithms required to obtain suitable meshless clouds for use in the overset scheme are quite simple and fast. These simple algorithms contrast with the complex and relatively expensive bin or octree search techniques to find near-body donor cells for off-body Cartesian grids. Moreover, when the meshless volume scheme of Chapter 3 is used, minimal memory is required, and computational efficiency is attained.

5.3 Off-body and Near-body Solution Schemes

One advantage of the meshless interface over hybrid or triangulated interface regions is modularity. Near-body and off-body solvers may be selected independently of eachother and independently of the meshless scheme. The results in this work were obtained using conventional finite volume algorithms in the near-body and off-body, with the meshless scheme of Chapter 3 in the interface region. The finite volume algorithms of the near-body and off-body followed the conservative numerical flux procedure of Equation 3.25. However, viscous terms were added to the numerical flux:

$$\boldsymbol{h}_{i+\frac{1}{2},j} = \frac{1}{2}(\boldsymbol{f}_{i,j} + \boldsymbol{f}_{i+1,j}) - \boldsymbol{f}_{vi+\frac{1}{2},j} - \frac{1}{2}(\boldsymbol{d}_{i+\frac{1}{2},j}).$$

The construction of the viscous flux, $\boldsymbol{f}_{vi+\frac{1}{2},j}$, was performed similar to the algorithm described in the thesis of Nadarajah [107]. Small changes were made to the off-body to enhance efficiency for Cartesian topology. Multigrid was also used in both the near-body and the off-body schemes, similar to Martinelli, Jameson, and Grasso [108].

5.4 Multi-solver Management with *Python*

It should be emphasized that for steady-state problems, the entire meshless connectivity routines may be performed in a preprocess step and accessed throughout the computation. The problem of grid communication is then reduced to the proper injection of Dirichlet boundary conditions for each grid component in the domain at each time step. Boundary conditions for each domain component are injected, the solution is advanced in pseudo-time, and the computation proceeds until steady-state is acheived. In this work, the iterative process is driven at the top level by a *Python* script. The *Python* script calls the necessary solution procedures (near-body, off-body, and meshless), which have been compiled as shared object files to give *Python* access to data at run time. This approach requires minimal modification to existing codes to be placed in the context of a globally hybrid scheme, and has recently been implemented successfully by Sitaraman et. al. [109].

A simplified version of the top level *Python* script used in this work is shown in Figure 5.4. Note that each solution procedure is imported as a shared object and called as a *Python* routine. Each solver is literally a module which can be called from the high level infrastructure offered by *Python*. Note that the near-body solver may be run stand-alone or in the hybrid context in this code infrastructure by simply switching the flag, `bc`, to various values. In summary, the *Python* infrastructure is an extremely uninstrusive method of coupling existing codes to form a hybrid scheme. Only minimal modifications to stand-alone source codes need to be made, and only at the highest levels which are actually called by the *Python* script.

5.5 Advantages of the Meshless Interface

The meshless interface provides a unique advantage over other grid communication schemes. First, the Chimera principle is maintained. Simple grids may be generated completely independent of one another for mulitple bodies. The meshless interface procedure is general enough to handle any type of grid topology as long as interfacing grids have similar resolution. The meshless interface may be considered another

```
# import necessary modules
import nearBody
import offBody
import meshless

# Allocation and initialization steps
nearBody.dimension()
nearBody.initialize()
if bc==-1:
    offBody.dimensionc()
    offBody.initializec()
    meshless.dimensionm()
    meshless.initializem()

# Iterate until steady state is reached
time      = 0
converged = False
while not converged:
    time  = time+1
    converged = nearBody.takestep()
    if bc==-1:
        convergedc = offBody.takestepc()
        convergedm = meshless.takestepm()
    elif bc>=0:
        nearBody.farfield()
```

Figure 5.4: Top level *Python* driver for hybrid meshless scheme

part of the Chimera creature, maintaining modularity while enhancing utility. This stands in contrast to triangulated interfaces or hybrid mesh schemes with arbitrary polyhedra. These schemes require grid generation tools that attempt to forge a single global grid. Such an approach is failure prone as the complexity level of geometry increases. Furthermore, such approaches are not as suitable for bodies in relative motion since grid components are not allowed to slide past one another. The meshless interface could potentially allow for relative motion by redefining clouds after every mesh movement. Since the local cloud definition procedure is relatively simple, the cost would be minimal. Of course, the penalty of using the meshless interface is the loss of formal conservation. However, this is no worse than conventional interpolation procedures, which are also non-conservative.

Another advantage of the meshless interface is the tight integration of the problem physics into the interface region. With interpolation, flow values for inter-grid boundary points are obtained by a purely numerical procedure. Interpolation tends to smear sharp features, especially for poorly resolved grids. With the meshless interface, the physics of the Euler or Navier-Stokes equations is preserved. Shock capturing is maintained while artificial diffusion is kept low. The result is a seamless transition between grids. Full convergence at every node in the domain is obtained, which means all nodes satisfy their parent discretization of the governing equations.

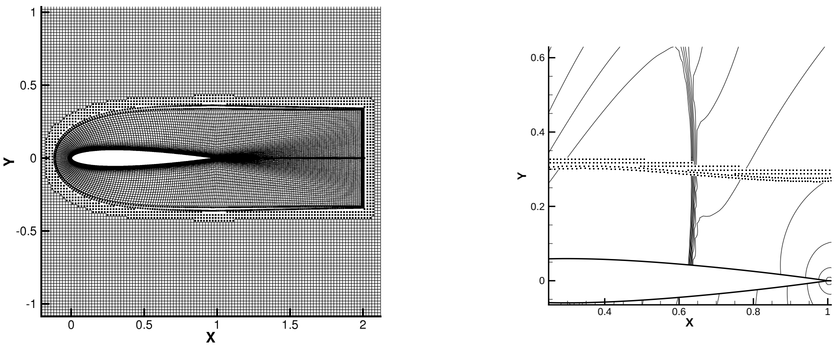
5.6 Viscous Results

The results in this section were obtained using the hybrid mesh shown in Figure 5.5(a) around the NACA 0012 airfoil. The near-body mesh was generated using the mesh generation package of FLO103, with certain modifications to smooth the wake spacing behind the boundary layer. The objective of smoothing the wake region was to provide even spacing from which to interface with the meshless scheme. All the results in this section are for relatively low Reynolds number, so high reslution of the wake is only critical a short distance behind the airfoil. For higher Reynolds numbers, the smoothing region should be extended several chord lengths downstream to properly capture the wake.

Shown in Figure 5.5(b) is a close-up view of the interface region showing a shock in transonic flow. Note the clean capturing of the discontinuity and the seamless transition across the meshless interface. This case highlights one strength of the meshless scheme within the hybrid approach, that of capturing regions of high solution gradients. Clearly, the meshless scheme is extremely well-suited to propagate fluid information across meshes since it makes use of the underlying physical model (Euler or Navier-Stokes).

The hybrid scheme also performs extremely well at predicting separation compared with the scheme of Mavriplis [64]. The predicted separation point at 82.7% for the NACA 0012 at $M = 0.5$, $\alpha = 0^o$ and $Re = 5000$ is in the middle of the range predicted by Mavriplis as shown in Figure 5.6(a) and displayed in Table 5.1. Contours of Mach number for this case are shown in Figure 5.6(b).

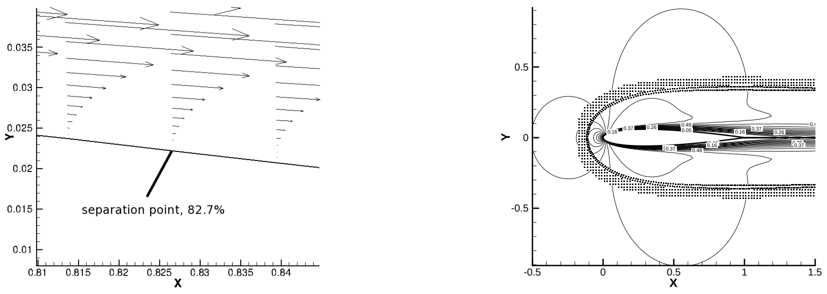
The remaining figures, Figsure 5.7-5.10, containing solution contours and surface pressure plots, highlight the accuracy of the hybrid meshless scheme for laminar viscous flows in two-dimensions. The surface pressure curves for the hybrid scheme are overplotted with the stand-alone results, showing excellent agreement. The stand-alone results use the near-body finite volume scheme in the entire domain Accompanying each set of plots is a table of lift and drag coefficients due to pressure comparing the hybrid scheme with FLO103, and when available, the viscous scheme of Mavirplis [64]. The lift and drag coefficients compare quite well for all cases tested. The conclusion to be drawn from these results is that the meshless interface provides an accurate means of computing viscous flows for complex configurations within an overset approach. Solution features are captured cleanly and seamlessly, as indicated by the solution contours. Global accuracy is attained, as indicated by the lift and drag coefficients. The results appear very promising, and encourage further investigation for more complex problems, such as high Reynolds number flows and flows in three dimensions.



(a) Hybrid mesh with meshless interface

(b) Shock capturing ability for inviscid transonic flow

Figure 5.5: Flow over NACA 0012, $M = 0.8$, $\alpha = 1.25^o$, showing mesh



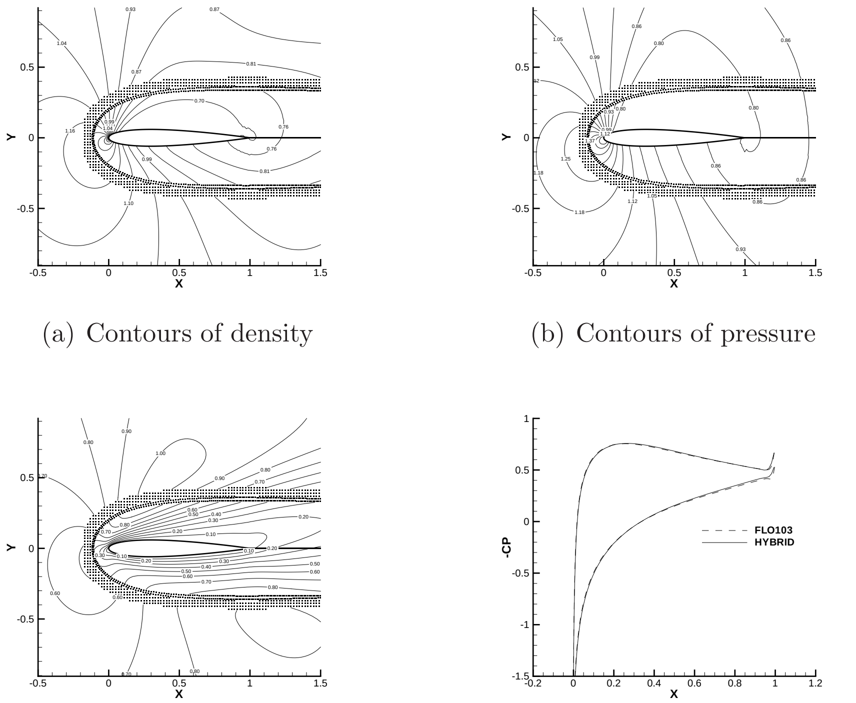
(a) Separation point

(b) Contours of Mach number

Figure 5.6: Flow over NACA 0012, $M = 0.5$, $\alpha = 0^o$, $Re = 5000$

Table 5.1: Comparison of separation point location for conventional and hybrid schemes.

	separation location
Mavriplis [64]	80.9%-83.4%
Hybrid	82.7%



(a) Contours of density

(b) Contours of pressure

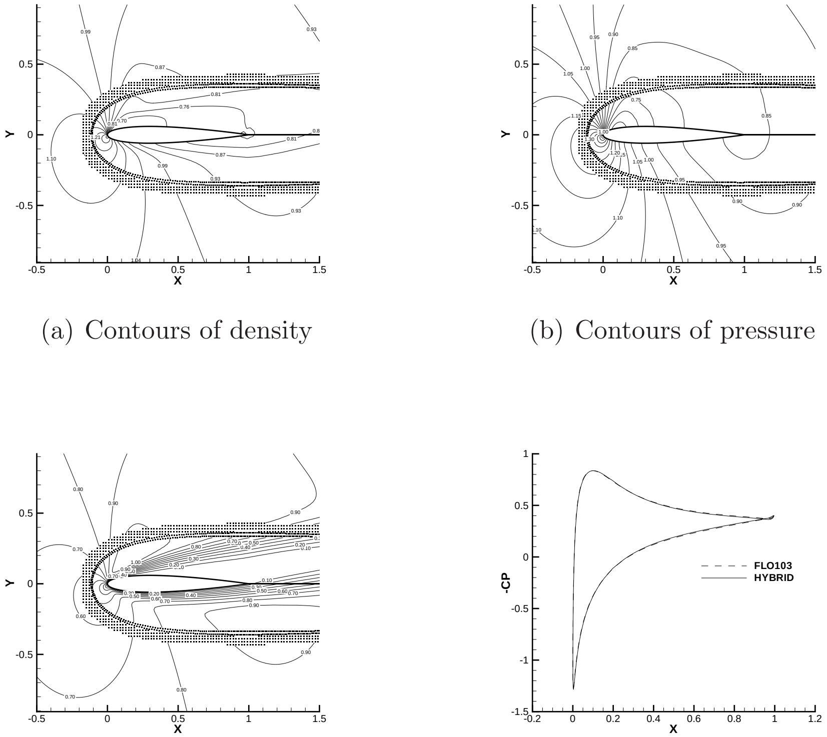
(c) Contours of Mach number

(d) Surface pressure coefficient

Figure 5.7: Flow over NACA 0012, $M = 0.8$, $\alpha = 10^o$, $Re = 73$

Table 5.2: Lift and drag coefficients due to pressure, NACA 0012, $M = 0.8$, $\alpha = 10^o$, $Re = 73$

	c_l	c_d
FLO 103	0.5480	0.2083
Mavriplis [64]	0.5886	0.2191
Hybrid	0.5446	0.2114



(a) Contours of density

(b) Contours of pressure

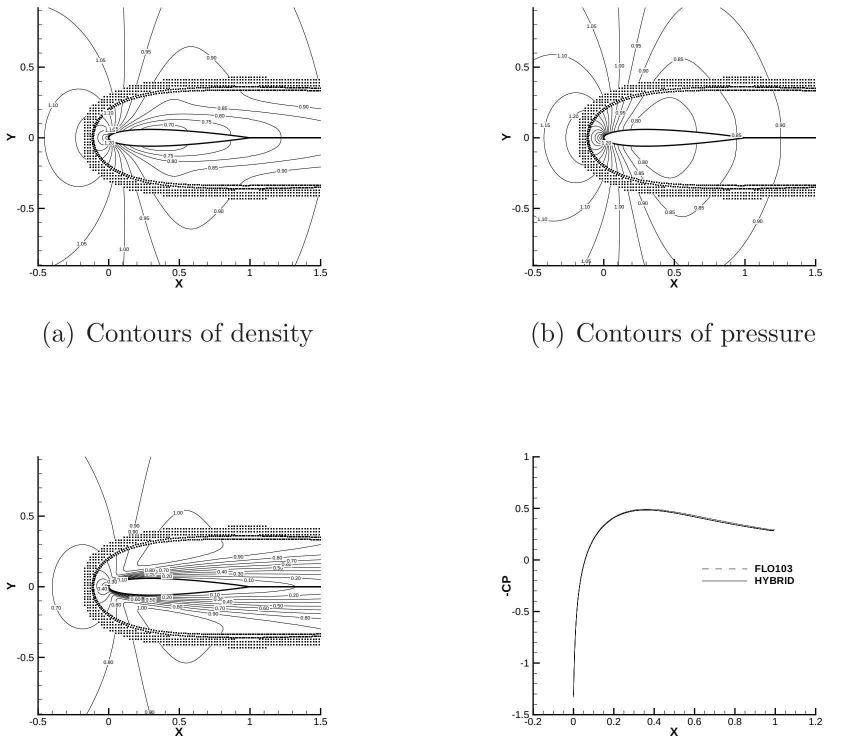
(c) Contours of Mach number

(d) Surface pressure coefficient

Figure 5.8: Flow over NACA 0012, $M = 0.8$, $\alpha = 10^o$, $Re = 500$

Table 5.3: Lift and drag coefficients due to pressure, NACA 0012, $M = 0.8$, $\alpha = 10^o$, $Re = 500$

	c_l	c_d
FLO 103	0.4311	0.1421
Mavriplis [64]	0.4469	0.1474
Hybrid	0.4234	0.1420



(a) Contours of density

(b) Contours of pressure

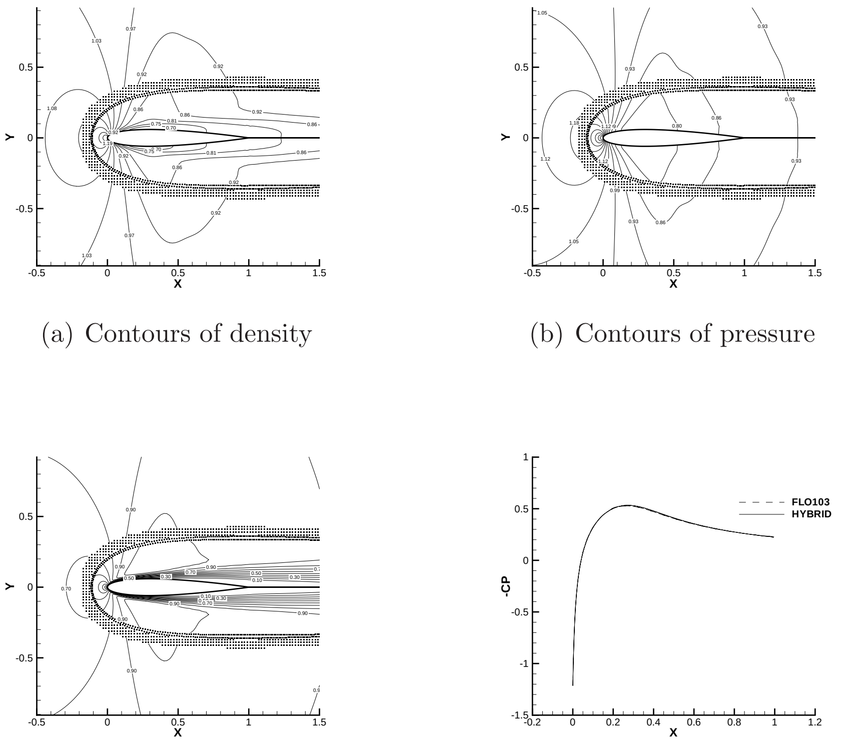
(c) Contours of Mach number

(d) Surface pressure coefficient

Figure 5.9: Flow over NACA 0012, $M = 0.85$, $\alpha = 0^o$, $Re = 500$

Table 5.4: Lift and drag coefficients due to pressure, NACA 0012, $M = 0.85$, $\alpha = 0^o$, $Re = 500$

	c_l	c_d
FLO 103	0.0000	0.0832
Hybrid	0.0000	0.0827



(a) Contours of density

(b) Contours of pressure

(c) Contours of Mach number

(d) Surface pressure coefficient

Figure 5.10: Flow over NACA 0012, $M = 0.85$, $\alpha = 0^o$, $Re = 2000$

Table 5.5: Lift and drag coefficients due to pressure, NACA 0012, $M = 0.85$, $\alpha = 0^o$, $Re = 2000$

	c_l	c_d
FLO 103	0.0000	0.0608
Hybrid	0.0000	0.0616

Chapter 6

Conclusions and Future Work

The work in this thesis attempted to develop the most efficient and accurate meshless schemes available for two dimensional subsonic, transonic, and viscous flows. This goal was largely accomplished through the use of specific weighting procedures which enabled finite volume-like algorithms to be implemented naturally. The development of the meshless volume scheme represents a streamlined and efficient scheme which rivals the accuracy and efficiency of most finite volume solution methods. When used in conjunction with the multicloud algorithm, the meshless volume scheme achieved convergence rates comparable to structured multigrid. To compute viscous flow, a hybrid algorithm was developed, which leverages various grid types to enhance accuracy. In this overset approach, a meshless method was used to seamlessly couple grid components together to obtain a globally accurate scheme. The meshless interface method proved to be both robust and simple in implementation. The test cases presented here showed excellent agreement with established finite volume methods.

Despite the excellent agreement with established methods, the meshless algorithms described in this thesis remain non-conservative. For many researchers, this is an unacceptable consequence of using meshless methods. Possibly, this has prevented the widespread use of meshless methods compared with grid-based methods. Hopefully, the work presented here will encourage further investigation into meshless schemes for practical use in computational fluid dynamics.

In the end, the promise of meshless schemes is to relieve, at least to some degree, the need for a rigid mesh. In order to fulfill this potential, some sort of "point generation" procedure would need to be developed. The point generator would need to show clear advantages over traditional mesh generation in terms of efficiency and robustness in order to be of practical value. Efforts to develop such capabilities are few and far between. However, encouraging results in this area have been published by Löhner and Oñate [1] and Löhner, Sacco, and Oñate [2]. The work in this thesis makes no such attempts to develop a point generator. The lack of point generation development in the literature would seem to suggest that there is no overwhelming advantage to such a capability over traditional meshing technology. In fact, generating points and defining local clouds appears to be a very similar task to generating an unstructured mesh, which would indicate a similar degree of difficulty. In this sense, the work of this thesis seems to offer no relief for mesh generation difficulties.

On the contrary, the work performed in this thesis has centered around applications for which meshless methods are ideally suited, such as multicloud and meshless interfaces for overset methods. These important developments leverage the best that meshless schemes have to offer: flexibility which uniquely compliments grid-based computation. It is unlikely that meshless methods will replace grid-based methods. This can be inferred by examining an example recent in the history of CFD. When unstructured algorithms became popular, they never replaced structured algorithms. Rather, intelligent researchers found niches in which they could thrive, namely, complex three dimensional simulations. Structured algorithms are still widely used in Cartesian and curvilinear methods. Meshless methods will likely find niches as well, which will only compliment grid-based approaches. Multicloud and meshless interface for overset approaches are two such niches.

To continue the work of this thesis, development will center around three dimensional applications. The work performed thus far has proven invaluable and necessary in terms of algorithmic development and validation. The next step is to increase the level of practical utility by extending the methods described here to three dimensions. While the implementation difficulties associated with three dimensions are real and should not be underestimated, there appear to be no theoretical barriers to achieving full three dimensional versions of the algorithms contained here. The realization the three dimensional meshless volume schemes, multicloud, and meshless interface methods have the potential to impact a wide variety of problems in computation mechanics.

Appendix A

The Euler and Navier-Stokes Equations

For many classes of fluid flow problems, the general formulation of the Navier-Stokes equations provides an accurate description of relevant physics [72]. The differential form of the Navier-Stokes equations in two dimensions and in strong conservation law form may be expressed as

$$\frac{\partial \boldsymbol{w}}{\partial t} + \frac{\partial \boldsymbol{f}_e}{\partial x} + \frac{\partial \boldsymbol{g}_e}{\partial y} = \frac{M_\infty}{Re_\infty} \left(\frac{\partial \boldsymbol{f}_v}{\partial x} + \frac{\partial \boldsymbol{g}_v}{\partial y} \right), \tag{A.1}$$

where $\boldsymbol{w}$ is the vector of conserved variables, $\boldsymbol{f}_e$ and $\boldsymbol{g}_e$ are the inviscid flux vectors, $\boldsymbol{f}_v$ and $\boldsymbol{g}_v$ are the viscous flux vectors, and M_∞ and Re_∞ are the free stream Mach and Reynolds numbers, respectively.

The Euler equations are obtained by setting the right-hand side of Equation A.1 to zero. The Euler equations provide an accurate description for many flow regimes since viscous effects are often negligible outside thin boundary layers. The Euler equations become an increasingly poor approximation for low Reynolds number flows with thick boundary layers and flows with separation. In these cases, the full Navier-Stokes equations provide a more accurate mathematical model of fluid flow.

The solution vector, $\boldsymbol{w}$, consists of the vector of non-dimensional conserved variables:

$$\boldsymbol{w} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{pmatrix},$$

where ρ, u, v, and E are density, Cartesian velocity components, and total energy respectively.

The inviscid fluxes, $\boldsymbol{f}_e$ and $\boldsymbol{g}_e$, are defined as:

$$\boldsymbol{f}_e = \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ \rho uH \end{pmatrix}, \quad \boldsymbol{g}_e = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + P \\ \rho vH \end{pmatrix},$$

where P is the pressure and $H = E + \frac{P}{\rho}$, is the total enthalpy. The additional variable, P, requires an equation of state to complete the set of equations. The ideal gas equation of state is an excellent description for a wide variety of flow regimes:

$$E = \frac{P}{(\gamma - 1)\rho} + \frac{1}{2}(u^2 + v^2),$$

where γ is the ratio of specific heats.

Viscous effects are introduced by the viscous flux terms, $\boldsymbol{f}_v$ and $\boldsymbol{g}_v$, which are defined as

$$\boldsymbol{f}_v = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ u\tau_{xx} + v\tau_{xy} - q_x \end{pmatrix}, \quad \boldsymbol{g}_v = \begin{pmatrix} 0 \\ \tau_{yx} \\ \tau_{yy} \\ u\tau_{yx} + v\tau_{yy} - q_y \end{pmatrix}.$$

The shear stress and heat flux terms are expanded as

$$\begin{aligned}
\tau_{xx} &= 2\mu u_x + \lambda(u_x + v_y) \\
\tau_{yy} &= 2\mu v_y + \lambda(u_x + v_y) \\
\tau_{xy} &= \tau_{xy} = \mu(u_y + v_x) \\
q_x &= -\kappa \frac{\partial T}{\partial x} \\
q_y &= -\kappa \frac{\partial T}{\partial y},
\end{aligned}$$

where μ and λ are the first and second coefficients of viscosity, and following Stoke's hypothesis, $\lambda = -\frac{2}{3}\mu$. The coefficient of thermal conductivity, κ, and temperature, T, are defined as

$$\kappa = \frac{C_p \mu}{Pr}, \quad T = \frac{P}{R\rho},$$

where C_p is the constant pressure specific heat, Pr is the Prandtl number, and R is the ideal gas constant. The Navier-Stokes equations are completed by Sutherland's law, which may be used to compute the viscosity:

$$\frac{\mu}{\mu_{ref}} = \left(\frac{T}{T_{ref}}\right)^{\frac{3}{2}} \frac{T_{ref} + T_o}{T + T_o},$$

where μ_{ref} is a reference viscosity at a reference temperature T_{ref}, and T_o is a constant.

The non-dimensionalization of the dependent and independent variables in Equation A.1 is

$$(x, y) = \left(\frac{x^*}{c}, \frac{y^*}{c}\right), \quad t = t^* \frac{\sqrt{\frac{P_\infty}{\rho_\infty}}}{c}, \quad \rho = \frac{\rho^*}{\rho_\infty}, \quad u = \frac{u^*}{\sqrt{\frac{P_\infty}{\rho_\infty}}}, \quad v = \frac{v^*}{\sqrt{\frac{P_\infty}{\rho_\infty}}},$$

$$E = \frac{E^*}{\frac{P_\infty}{\rho_\infty}}, \quad P = \frac{P^*}{P_\infty}, \quad \mu = \frac{\mu^*}{\mu_{ref}} \frac{\sqrt{\gamma} M_\infty}{Re},$$

where the $*$ variables are dimensional, c is a characteristic length scale, and the ∞

variables represent free stream values.

In constructing artificial diffusion vectors, it is helpful to define a directed inviscid flux, $\boldsymbol{F} = n_x \boldsymbol{f}_e + n_y \boldsymbol{g}_e$, in a direction, (n_x, n_y). Roe [75] established the following relation:

$$\boldsymbol{F}_R - \boldsymbol{F}_L = A(\boldsymbol{w}_R, \boldsymbol{w}_L)(\boldsymbol{w}_R - \boldsymbol{w}_L), \tag{A.2}$$

where A is the directed flux Jacobian. The mean value Jacobian, $A(w_L, w_R)$, is simply the standard Jacobian evaluated using Roe-averaged variables. The Roe-averaging of left and right states proceeds as follows:

$$u = \frac{\sqrt{\rho_R}u_R + \sqrt{\rho_L}u_L}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \quad v = \frac{\sqrt{\rho_R}v_R + \sqrt{\rho_L}v_L}{\sqrt{\rho_L} + \sqrt{\rho_R}}, \quad H = \frac{\sqrt{\rho_R}H_R + \sqrt{\rho_L}H_L}{\sqrt{\rho_L} + \sqrt{\rho_R}}.$$

The two dimensional Jacobian matrix, A, expressed with the Roe-averaged variables is then

$$A = \begin{bmatrix} 0 & n_x & n_y & 0 \\ n_x(\gamma-1)\frac{q^2}{2} - uu_n & u_n - (\gamma-2)n_x u & n_y u - (\gamma-1)n_x v & n_x(\gamma-1) \\ n_y(\gamma-1)\frac{q^2}{2} - vu_n & n_x v - (\gamma-1)n_y u & u_n - (\gamma-2)n_y v & n_y(\gamma-1) \\ u_n\left((\gamma-1)\frac{q^2}{2} - H\right) & n_x H - (\gamma-1)uu_n & n_y H - (\gamma-1)vu_n & \gamma u_n \end{bmatrix},$$

where $u_n = un_x + vn_y$, $q^2 = u^2 + v^2$, and $H = \frac{c^2}{\gamma-1} + \frac{q^2}{2}$.

Now define $|A| = T|\Lambda|T^{-1}$, where $|\Lambda|$ is a diagonal matrix containing the absolute values of the eigenvalues of A, and the columns of T contain the eigenvectors of A. It follows that A may be diagonalized by $\Lambda = T^{-1}AT$, with

$$\Lambda = \begin{bmatrix} u_n & 0 & 0 & 0 \\ 0 & u_n & 0 & 0 \\ 0 & 0 & u_n + c & 0 \\ 0 & 0 & 0 & u_n - c \end{bmatrix},$$

$$T = \begin{bmatrix} 1 & 0 & 1 & 1 \\ u & cn_y & u + cn_x & u - cn_x \\ v & -cn_x & v + cn_y & v - cn_y \\ \frac{q^2}{2} & c(n_y u - n_x v) & H + cu_n & H - cu_n \end{bmatrix},$$

$$T^{-1} = \begin{bmatrix} 1 - \frac{\gamma-1}{c^2}\frac{q^2}{2} & \frac{\gamma-1}{c^2}u & \frac{\gamma-1}{c^2}v & -\frac{\gamma-1}{c^2} \\ -\frac{un_y - vn_x}{c} & \frac{n_y}{c} & -\frac{n_x}{c} & 0 \\ \frac{1}{2c^2}\left((\gamma-1)\frac{q^2}{2} - cu_n\right) & \frac{1}{2c^2}\left(-(\gamma-1)u + cn_x\right) & \frac{1}{2c^2}\left(-(\gamma-1)v + cn_y\right) & \frac{\gamma-1}{2c^2} \\ \frac{1}{2c^2}\left((\gamma-1)\frac{q^2}{2} + cu_n\right) & \frac{1}{2c^2}\left(-(\gamma-1)u - cn_x\right) & \frac{1}{2c^2}\left(-(\gamma-1)v - cn_y\right) & \frac{\gamma-1}{2c^2} \end{bmatrix}.$$

Bibliography

[1] R. Löhner and E. Oñate. An advancing front point generation technique. *Communications in Numerical Methods in Engineering*, 14:1097–1108, 1998.

[2] R. Löhner, C. Sacco, and E. Oñate. A general advancing front technique for filling space with arbitrary objects. *Int. J. Numer. Meth. Engng.*, 61:1977–1991, 2004.

[3] G. R. Liu. *Mesh Free Methods: Moving Beyond the Finite Element Method*. CRC Press, 2003.

[4] G. R. Liu and Y. T. Gu. *An Introduction to Meshfree Methods and Their Programming*. Springer, 2005.

[5] J. J. Monaghan and R. A. Gingold. Shock simulation by the particle method sph. *Journal of Computational Physics*, 52:374–389, 1983.

[6] W. K. Liu, S. Jun, and Y. F. Zhang. Reproducing kernel particle methods. *International Journal for Numerical Methods in Fluids*, 20:1081–1106, 1995.

[7] F. C. Gunther and W. K. Liu. Implementation of boundary conditions for meshless methods. *Comput, Methods Appl. Mech. Engrg.*, 163:205–230, 1998.

[8] M. Lesoinne and V. Kaila. Meshless aeroelastic simulations of aircraft with large control surface deflections. *AIAA paper* 2005-1089, AIAA 43rd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2005.

[9] L. T. Zhang, G. J. Wagner, and W. K. Liu. A parallelized meshfree method with boundary enrichment for large-scale cfd. *Journal of Computational Physics*, 176:483–506, 2002.

[10] T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl. Meshless methods: An overview and recent developments. *Comput, Methods Appl. Mech. Engrg.*, 139:3–47, 1996.

[11] A. Jameson, T. J. Baker, and N. P. Weatherill. Calculation of inviscid transonic flow over a complete aircraft. *AIAA paper* 1986-0103, AIAA 24th Aerospace Sciences Meeting, Reno, NV, January 1986.

[12] S. N. Atluri and T. Zhu. A new meshless local petrov-galerkin (mlpg) approach in computational mechanics. *Computational Mechanics*, 22:117–127, 1998.

[13] S. N. Atluri, H. G. Kim, and J. Y. Cho. A critical assessment of the truly meshless local petrov-galerkin (mlpg) and local boundary integral equation (lbie) methods. *Computational Mechanics*, 24:348–372, 1999.

[14] T. Belytschko, Y. Y. Lu, and L. Gu. Element-free galerkin methods. *International Journal for Numerical Methods in Engineering*, 37:229–256, 1994.

[15] B. Nayroles, G. Touzot, and P. Villon. Generalizing the finite element method: Diffuse approximation and diffuse elements. *Computational Mechanics*, 10:307–318, 1992.

[16] H. Lin and S. N. Atluri. The meshless local petrov-galerkin (mlpg) method for solving incompressible navier-stokes equations. *CMES*, 2(2):117–142, 2001.

[17] R. Franke. Scattered data interpolation: Tests of some methods. *Mathematics of Computation*, 38:181–200, 1982.

[18] E. J. Kansa. Multiquadrics–a scattered data approximation scheme with applications to computational fluid-dynamics–i. *Computers Math. Applic.*, 19:127–145, 1990.

[19] E. J. Kansa. Multiquadrics–a scattered data approximation scheme with applications to computational fluid-dynamics–ii. *Computers Math. Applic.*, 19:147–161, 1990.

[20] Y. C. Hon and R. Schaback. On unsymmetric collocation by radial basis functions. *Applied Mathematics and Computation*, 119:177–186, 2001.

[21] H. Power and V. Barraco. Comparison analysis between unsymmetric and symmetric radial basis function collocation methods for the numerical solution of partial differential equations. *Computers Math. Applic.*, 43:551–583, 2002.

[22] J. Li, A. H. D. Cheng, and C. Chen. A comparison of efficiency and error convergence of multiquadric collocation method and finite element method. *Engineering Analysis with Boundary Elements*, 27:251–257, 2003.

[23] M. Sharan, E. J. Kansa, and S. Gupta. Application of the multiquadric method for numerical solution of elliptic partial differential equations. *Applied Mathematics and Computation*, 84:275–302, 1997.

[24] B. Sarler and R. Vertnik. Meshfree explicit local radial basis function collocation method for diffusion problems. *Computers Math. Applic.*, 51:1269–1282, 2006.

[25] H. Wendlend. Meshless galerkin methods using radial basis functions. *Mathematics of Computation*, 68:1521–1531, 1999.

[26] E. Divo and A. J. Kassab. Efficient localized meshless modeling of natural convective viscous flows. *AIAA paper* 2006-3089, 9th AIAA/ASME Joint Thermophysics and Heat Transfer Conference, San Francisco, CA, June 2006.

[27] E. Divo and A. J. Kassab. A meshless method for conjugate heat tranfer problems. *Engineering Analysis with Boundary Elements*, 29:136–149, 2005.

[28] P. P. Chinchapatnam, K. Djidjeli, and P. B. Nair. Meshless rbf collocation for steady incompressible viscous flows. *AIAA paper* 2006-3525, AIAA 36th Fluid Dynamics Conference and Exhibit, San Francisco, CA, June 2006.

[29] C. Shu, H. Ding, H. Q. Chen, and T. G. Wang. An upwind local rbf-dq method for simulation of inviscid compressible flows. *Comput, Methods Appl. Mech. Engrg.*, 194:2001–2017, 2005.

[30] D. J. Mavriplis. Revisiting the least-squares procedure for gradient reconstruction on unstructured meshes. *AIAA paper* 2003-3986, AIAA 16th Computational Fluid Dynamics Conference, Orlando, FL, June 2003.

[31] K. C. Chung. A generalized finite-difference method for heat transfer problems of irregular geometries. *Numerical Heat Transfer*, 4:345–357, 1981.

[32] T. Liszka and J. Orkisz. The finite difference method at arbitrary irregular grids and its application in applied mechanics. *Computers and Structures*, 11:83–95, 1980.

[33] S. M. Deshpande, N. Anil, K. Arora, K. Malagi, and M. Varma. Some fascinating new develpments in kinetic schemes. In A. Avudainayagam, P. Misra, and S. Sundar, editors, *Proceedings Workshop on Modeling and Simulation in Life Sciences, Materials and Technology*, pages 43–64. 2004.

[34] S. M. Deshpande, K. Anandhanarayanan, C. Praveen, and V. Ramesh. Theory and application of 3-d lskum based on entropy variables. *International Journal for Numerical Methods in Fluids*, 40:47–62, 2002.

[35] A. K. Ghosh and S. M. Deshpande. Least squares kinetic upwind method for inviscid compressible flows. *AIAA paper* 1995-1735, AIAA 12th Computational Fluid Dynamics Conference, San Diego, CA, June 1995.

[36] V. Ramesh and S. M. Deshpande. Euler computations on arbitrary grids using lskum. In N. Satofuka, editor, *Computational Fluid Dynamics 2000: Proceedings of the First International Conference on Computational Fluid Dynamics*, pages 783–784. Springer-Verlag, 2000.

[37] K. Anandhanarayanan and M. Nagarathinam. Parallelisation of a gridfree kinetic upwind solver. *AIAA paper* 2005-4628, AIAA 17th Computational Fluid Dynamics Conference, Toronto, June 2005.

[38] G. Harish and M. Pavanakumar. Store separation dynamics using grid-free euler solver. *AIAA paper* 2006-3650, AIAA 24th Applied Aerodynamics Conference, San Francisco, CA, June 2006.

[39] N. Srinarayana, L. F. Gonzalez, E. J. Whitney, and K. Srinivas. Aerodynamic optimisation using a robust evolutionary algorithm and grid-free flowsolver. *AIAA paper* 2006-52, AIAA 44th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2006.

[40] G. N. Sashi Kumar, A. K. Mahendra, and S. V. Raghurama Rao. Shape optimization using hybrid ga-aco method and grid-free cfd solver. *AIAA paper* 2007-3830, AIAA 18th Computational Fluid Dynamics Conference, Miami, FL, June 2007.

[41] C. Praveen. Development and application of kinetic meshless methods for euler equations. Phd thesis, IISc, 2004.

[42] C. Praveen. Some results on the least squares formula. *FM Report* 2003-FM-10, IISc, 2001.

[43] C. Praveen and S. M. Deshpande. A new grid-free method for conservation laws. In S. Armfield, P. Morgan, and K. Srinivas, editors, *Computational Fluid Dynamics 2002: Proceedings of the Second International Conference on Computational Fluid Dynamics*, pages 128–133. 2002.

[44] D. Sridar and N. Balakrishnan. An upwind finite difference scheme for meshless solvers. *Journal of Computational Physics*, 189:1–29, 2003.

[45] D. Sridar and N. Balakrishnan. Convergence acceleration of an upwind least squares finite difference based meshless solver. *AIAA Journal*, 44:2189–2196, 2006.

[46] S. Jaisankar, K. Shivashankar, and S. V. Raghurama Rao. A grid-free central scheme for inviscid compressible flows. *AIAA paper* 2007-3946, AIAA 18th Computational Fluid Dynamics Conference, Miami, FL, June 2007.

[47] K. Morinishi. Effective accuracy and conservation consistency of gridless type solver. In N. Satofuka, editor, *Computational Fluid Dynamics 2000: Proceedings of the First International Conference on Computational Fluid Dynamics*, pages 325–330. Springer-Verlag, 2000.

[48] S. Balasubramanyam and S. V. Raghurama Rao. A new grid-free relaxation scheme for conservation laws. In S. Armfield, P. Morgan, and K. Srinivas, editors, *Computational Fluid Dynamics 2002: Proceedings of the Second International Conference on Computational Fluid Dynamics*, pages 259–264. 2002.

[49] S. Balasubramanyam and S. V. Raghurama Rao. A grid-free upwind relaxation scheme for inviscid compressible flows. *International Journal for Numerical Methods in Fluids*, 51:159–196, 2006.

[50] J. T. Batina. A gridless euler/navier-stokes solution algorithm for complex aircraft applications. *AIAA paper* 1993-0333, AIAA 31st Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1993.

[51] J. Liu and S. Su. A potentially gridless solution method for the compressible euler/navier-stokes equations. *AIAA paper* 1996-0526, AIAA 34th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1996.

[52] E. Oñate, S. Idelsohn, O. C. Zienkiewicz, R. L. Taylor, and C. Sacco. A stabilized finite point method for analysis of fluid mechanics problems. *Comput. Methods Appl. Mech. Engrg.*, 139:315–346, 1996.

[53] E. Oñate, S. Idelsohn, O. C. Zienkiewicz, and R. L. Taylor. A finite point method in computational mechanics. applications to convective transport and fluid flow. *International Journal for Numerical Methods in Engineering*, 39:3839–3866, 1996.

[54] E. Oñate and S. Idelsohn. A mesh-free finite point method for advective-diffusive transport and fluid flow problems. *Computational Mechanics*, 21:283–292, 1998.

[55] R. Löhner, C. Sacco, E. Oñate, and S. Idelsohn. A finite point method for compressible flow. *Int. J. Numer. Meth. Engng.*, 53:1765–1779, 2002.

[56] B. Van Leer. Towards the ultimate conservative difference scheme. v a second order sequel to gudonov's method. *Journal of Computational Physics*, 32:101–136, 1979.

[57] D. J. Kirshman and F. Liu. Gridless boundary condition treatment for a non-body-conforming mesh. *AIAA paper* 2002-3285, AIAA 32nd Fluid Dynamics Conference, St. Louis, MO, June 2002.

[58] D. J. Kirshman and F. Liu. Cartesian grid solution of the euler equations using a gridless boundary treatment. *AIAA paper* 2003-3974, AIAA 16th Computational Fluid Dynamics Conference, Orlando, FL, June 2003.

[59] D. J. Kirshman and F. Liu. A gridless boundary condition method for the solution of the euler equations on embedded cartesian meshes with multigrid. *Journal of Computational Physics*, 201:119–147, 2004.

[60] E. P. C. Koh and H. M. Tsai. Euler solution using cartesian grid with least squares technique. *AIAA paper* 2003-1120, AIAA 41st Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2003.

[61] H. Luo, J. D. Baum, and R. Löhner. A hybrid cartesian grid and gridless method for compressible flows. *AIAA paper* 2005-0492, AIAA 43rd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2005.

[62] T. Kamatsuchi. Turbulent flow simulation around complex geometries with cartesian grid method. *AIAA paper* 2007-1459, AIAA 45th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2007.

[63] D. Mavriplis and A. Jameson. Multigrid solution of the two-dimensional euler equations on unstructured triangular meshes. *AIAA paper* 1987-0353, AIAA 25th Aerospace Sciences Meeting, Reno, NV, January 1987.

[64] D. J. Mavriplis, A. Jameson, and L. Martinelli. Multigrid solution of the navier-stokes equations on triangular meshes. *AIAA paper* 1989-0120, AIAA 27th Aerospace Sciences Meeting, Reno, NV, January 1989.

[65] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1983.

[66] R. Schaback and H. Wendlend. Characterization and construction of radial basis functions. In N. Dyn, D. Leviatan, D. Levin, and A. Pinkus, editors, *Multivariate Approximations and Applications*, pages 1–24. Cambridge University Press, 2001.

[67] D. West. *Introduction to Graph Theory*. Prentice Hall, 2001.

[68] A. Katz and A. Jameson. Edge-based meshless methods for compressible flow simulations. *AIAA paper* 2008-669, AIAA 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2008.

[69] A. Katz and A. Jameson. Edge-based meshless methods for compressible viscous flow with applications to overset grids. *AIAA paper* 2008-3989, AIAA 38th Fluid Dynamics Conference, Seattle, WA, June 2008.

[70] D. J. Mavriplis. Unstructured mesh discretizations and solvers for computational aerodynamics. *AIAA paper* 2007-3955, AIAA 18th Computational Fluid Dynamics Conference, Miami, FL, June 2007.

[71] A. Jameson. Analysis and design of numerical schemes for gas dynamics 1 artificial diffusion, upwind biasing, limiters and their effect on accuracy and multigrid convergence. *International Journal of Computational Fluid Dynamics*, 4:171–218, 1995.

[72] F. White. *Viscous Fluid Flow*. McGraw Hill, 2006.

[73] A. Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 49:357–393, 1983.

[74] A. Jameson, W. Schmidt, and Eli Turkel. Numerical solutions of the euler equations by finite volume methods using runge-kutta time-stepping schemes. *AIAA paper* 1981-1259, AIAA 14th Fluid and Plasma Dynamic Conference, Palo Alto, CA, June 1981.

[75] P. Roe. Approximate riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43:357–372, 1981.

[76] A. Jameson. Analysis and design of numerical schemes for gas dynamics 2 artificial diffusion and discrete shock structure. *International Journal of Computational Fluid Dynamics*, 5:1–38, 1995.

[77] H. W. Liepmann and A. Roshko. *Elements of Gasdynamics*. Dover Publications, Inc., 1957.

[78] T. Pulliam. Efficient solution methods for the navier-stokes equations. *Lecture Notes for the Von Karman Institute for Fluid Dynamics Lecture Series*, 1986.

[79] A. Jameson. Solution of the euler equations for two dimensional transonic flow by a multigrid method. *Applied Mathematics and Computation*, 13:327–355, 1983.

[80] S. Tatsumi, L. Martinelli, and A. Jameson. Flux-limited schemes for the compressible navier-stokes equations. *AIAA Journal*, 33:252–261, 1995.

[81] F. Liu and A. Jameson. Multigrid navier-stokes calculations for three-dimensional cascades. *AIAA paper* 1992-0190, AIAA 30th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1992.

[82] G. May and A. Jameson. Unstructured algorithms for inviscid and viscous flows embedded and a unified solver architecture: Flo3xx. *AIAA paper* 2005-0318, AIAA 43rd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2005.

[83] A. Jameson and D. Mavriplis. Finite volume solution of the two-dimensional euler equations on a regular triangular mesh. *AIAA Journal*, 24:611–618, 1986.

[84] P. Wesseling. *An Introduction to Multigrid Methods.* John Wiley and Sons, 1992.

[85] R. Fedorenko. The speed of convergence of one iterative process. *USSR Computational Mathematics and Mathematical Physics*, 4:227–235, 1964.

[86] A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Mathematics of Computation*, 31:333–390, 1977.

[87] R. Ni. A multiple grid scheme for solving the euler equations. *AIAA paper* 1981-1025, AIAA 5th Computational Fluid Dynamics Conference, Palo Alto, CA, June 1981.

[88] A. Jameson. Multigrid algorithms for compressible flow calculations. In W. Hackbusch and U. Trottenberg, editors, *Lecture Notes in Mathematics*, pages 166–201. Springer-Verlag, 1986.

[89] E. Perez. Finite element and multigrid solution of the two-dimensional euler equations on a non-structured mesh. *Report* 442, INRIA, September 1985.

[90] R. Löhner and K. Morgan. Unstructured multigrid methods. Paper presentation, 2nd European Conference on Multigrid Methods, October 1985.

[91] M. Lallemand, H. Steve, and A. Dervieux. Unstructured multigridding by volume agglomeration: Current status. *Computers and Fluids*, 21:397–433, 1992.

[92] V. Venkatakrishnan and D. J. Mavriplis. Agglomeration multigrid for the three dimensional euler equations. *AIAA paper* 1994-0069, AIAA 32nd Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1994.

[93] S. D. Connell and D. G. Holmes. A 3d unstructured adaptive multigrid scheme for the euler equations. *AIAA paper* 1993-3339, 1993.

[94] N. Balakrishnan and C. Praveen. A new upwind least squares finite difference scheme (lsfd-u) for euler equations of gas dynamics. In F. Benkhaldoun, R. Vilsmeier, and D Hänel, editors, *Finite Volumes for Complex Applications, vol. II*, page 331. Hermes Science Publications, 1999.

[95] C. Praveen, , and N. Balakrishnan. New least squares based finite difference scheme for compressible flows. Technical report, Proceedings of the 8th Asian Congress of Fluid Mechanics, Shenzen, China, December 1999.

[96] C. Duarte and J. Oden. Hp clouds - a meshless method to solve boundary-value problems. Technical Report 95-05, Texas Institute for Computational and Applied Mathematics, Austin, TX, 1995.

[97] J. Müller. On triangles and flow. Phd thesis, The University of Michigan, 1996.

[98] R. L. Meakin, A. M. Wissink, W. M. Chan, S. A. Pandya, and J. Sitaraman. On strand grids for complex flows. *AIAA paper* 2007-3834, AIAA 18th Computational Fluid Dynamics Conference, Miami, FL, June 2007.

[99] J. A. Benek, J. L. Steger, and F. C. Dougherty. A flexible grid embedding technique with application to the euler equations. *AIAA paper* 1983-1944, AIAA 6th Computational Fluid Dynamics Conference, Danvers, MA, July 1983.

[100] J. Steger, F. Dougherty, and J. Benek. A chimera grid scheme. Technical report, ASME Mini-Symposium on Advances in Grid Generation, Houston, TX, June 1983.

[101] R. Meakin. The chimera method of simulation for unsteady three-dimensional viscous flow. In M. Hafez and K. Oshima, editors, *Computational Fluid Dynamics Review 1995*, pages 70–86. John Wiley and Sons Ltd, 1995.

[102] D. Jespersen, T. H. Pulliam, and P. Buning. Recent enhancements to overflow (navier-stokes code). *AIAA paper* 1997-0644, AIAA 35th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 1997.

[103] R. Meakin. Composite overset structured grids. In *Handbook of Grid Generation*. CRC Press, 1999.

[104] R. L. Meakin. On the spatial and temporal accuracy of overset grid methods for moving body problems. *AIAA paper* 1994-1925, AIAA 12th Applied Aerodynamics Conference, Colorado Springs, CO, June 1994.

[105] A. Haselbacher and J. Blazek. Accurate and efficient discretization of navier-stokes equations on mixed grids. *AIAA Journal*, 38:2094–2102, 2000.

[106] S. Kano and K. Nakahashi. Flow computations around delta wings using unstructured hybrid grids. *Journal of Aircraft*, 36:374–379, 1999.

[107] S. K. Nadarajah. The discrete adjoint approach to aerodynamic shape optimization. Phd thesis, Stanford University, 2003.

[108] L. Martinelli, A. Jameson, and F. Grasso. A multigrid method for the navier-stokes equations. *AIAA paper* 1986-0208, AIAA 24th Aerospace Sciences Meeting, Reno, NV, January 1986.

[109] J. Sitaraman, A. Katz, B. Jayaraman, A. M. Wissink, and V. Sankaran. Evaluation of a multi-solver paradigm for cfd using overset unstructured and structured adaptive cartesian grids. *AIAA paper* 2008-660, AIAA 46th Aerospace Sciences Meeting and Exhibit, Reno, NV, January 2008.