SANDIA REPORT

SAND96–0199 • UC–706
Unlimited Release
Printed January 1996

A Gridless Technique for Fluid/Structural Dynamic Coupling on Flexible Membranes

Walter P. Wolfe, James M. Nelsen, Roy S. Baty, Glenn A. Laguna,
Frank J. Mello, Christine E. Hailey, N. Todd Snyder

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550
for the United States Department of Energy
under Contract DE-AC04-94AL85000

Approved for public release; distribution is unlimited.

SF2900Q(8-81)

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
Office of Scientific and Technical Information
PO Box 62
Oak Ridge, TN 37831

Prices available from (615) 576-8401, FTS 626-8401

Available to the public from
National Technical Information Service
US Department of Commerce
5285 Port Royal Rd
Springfield, VA 22161

NTIS price codes
Printed copy: A05
Microfiche copy: A01

SAND 96-0199
Unlimited Release
Printed January 1996

Distribution
Category UC-706

A Gridless Technique for Fluid/Structural Dynamic Coupling on Flexible Membranes

Walter P. Wolfe and James M. Nelsen
Unsteady and Reactive Fluid Mechanics Department

Roy S. Baty
Aerosciences and Compressible Fluid Mechanics Department

Glenn A. Laguna
Intelligent System Principles Department

Frank J. Mello
Material and Structural Mechanics Department

Sandia National Laboratories
Albuquerque, New Mexico 87185-0836

Christine E. Hailey and N. Todd Snyder
Department of Mechanical and Aerospace Engineering
Utah State University
Logan, Utah 84322-4130

Abstract

A gridless method has been developed for the simulation of coupled fluid/structural interactions over arbitrary bodies. This method uses Eulerian-based points arbitrarily distributed over the computational domain with no formal connectivity as typically required for a traditional grid. Comparisons are made with known exact solutions for simple two-dimensional model problems. Methods of improving the accuracy of the current implementation by using higher order approximations have been implemented. Accuracy improvement by using point adaption has been investigated. Plane strain and axisymmetric shells have been added to the code structural code PRONTO2D for future fluid/structural calculations. To date, coupled fluid/structure calculations have not been made.

Blank Page

Table of Contents

1 Introduction **11**

2 Overview of Gridless Methods **13**
- 2.1 Introduction and General Gridless Philosophy 13
- 2.2 Previous Gridless Investigations 13
- 2.3 The Gridless Numerical Method 14
- 2.4 Least Squares Solutions for Coupled Linear Systems 17
- 2.5 Least Squares Solutions for the Gridless Method 21

3 Current Gridless Scheme **25**
- 3.1 Overview 25
- 3.2 Method Development 25
  - 3.2.1 Cloud Formation 25
  - 3.2.2 Cloud Equations 26
  - 3.2.3 Boundary Conditions 27
  - 3.2.4 Solution Procedure 28
- 3.3 Numerical Examples 29
  - 3.3.1 Model Potential Flow Problems 29
  - 3.3.2 Numerical Results 31

4 Higher Order Taylor Series **43**
- 4.1 Overview of Tasks 43
- 4.2 Two-Dimensional Taylor Series 43
- 4.3 Higher-Order Solutions of Laplace's Equation 49
- 4.4 Summary and Recommendations for Future Work 52

5 Point Adaption Methodology **61**
- 5.1 Introduction 61
- 5.2 Goals for Adaption 61
- 5.3 The Adaption Algorithm 62
- 5.4 Adaption Algorithm Performance 64
- 5.5 1-D Adaption Summary 67
- 5.6 The Adaption Methodology in Multi-Dimensions 68
- 5.7 The Adaption Methodology for Time-Accurate Problems 69

6 Shell Elements **71**
- 6.1 Introduction 71
- 6.2 Governing Equations 71
  - 6.2.1 Equations of Motion 71
- 6.3 Shell Kinematics 72
  - 6.3.1 Constitutive Assumptions 73
  - 6.3.2 Element Formulation 73
  - 6.3.3 Coordinate Systems 73
  - 6.3.4 Element Equations 74

6.3.5 Computation of Internal Force and Moment Resultants 75
6.3.6 Hourglass Control 75
6.3.7 Calculation of Stable Time Increment 75
6.4 Constitutive Models 76

7 Summary and Recommendations 77

8 References 79

Appendix A:LDRD Reporting Requirements 81

Distribution 82

List of Figures

Figure 2.1 An arbitrary computational cell in the interval (0,1) for the gridless numerical method 16

Figure 3.1 Field points for calculation of the potential flow around a circular cylinder - 3280 points 34

Figure 3.2 Close-up view of the field point distribution in Figure 3.1 for the circular cylinder 34

Figure 3.3 Behavior of global residual and error during calculation of potential flow over a circular cylinder using 3280 points 35

Figure 3.4 Contour plot of the local error of the velocity potential computed with the gridless scheme - 3280 pts 35

Figure 3.5 Comparison of the exact and computed pressure coefficients for the circular cylinder - basic scheme 36

Figure 3.6 Global root-sum-squared error of the gridless numerical method as a function of the number of field points - basic scheme, circular cylinder 36

Figure 3.7 Global root-sum-squared residual of the gridless numerical method as a function of the number of field points - basic scheme, circular cylinder 37

Figure 3.8 Contour plot of the local error of the velocity potential computed with the gridless scheme - 13180 points 37

Figure 3.9 Comparison of the global root-sum-square residual for the basic and coupled gridless numerical schemes - circular cylinder 38

Figure 3.10 Comparison of the global root-sum-square error for the basic and coupled gridless numerical schemes - circular cylinder 38

Figure 3.11 Field point distribution for potential flow around a 5:1 aspect ratio elliptical cylinder - 3280 points 39

Figure 3.12 Close-up view of the field point distribution in Figure 3.11 for the elliptical cylinder 39

Figure 3.13 Behavior of global residual and error during calculation of potential flow over a 5:1 aspect ratio elliptic cylinder using 3280 points 40

Figure 3.14 Contour plot of the absolute error of the velocity potential computed with the gridless scheme 40

Figure 3.15 Comparison of the exact and computed pressure coefficients for the elliptic cylinder - basic scheme 41

Figure 3.16 Comparison of the global root-sum-square residual for the basic and coupled gridless numerical schemes - elliptic cylinder 41

Figure 3.17 Comparison of the global root-sum-squared error for the basic and coupled gridless numerical schemes - elliptic cylinder 42

Figure 4.1 Surface Plot of $f(x, y) = x^4 y^4$ 45

Figure 4.2 Example of Circular Cloud Geometry 46

Figure 4.3 Example of Rectangular Cloud Geometry 47

Figure 4.4 Plot of Cloud Geometry for Irregular Geometry 48

Figure 4.5 Normalized Error As a Function of Order and Cloud Geometry for Evaluation Point (0.5, 0.5) 48

Figure 4.6 Normalized Error As a Function of Order and Cloud Geometry for Evaluation Point (0.9, 0.8) 49

Figure 4.7 Several Higher-Order Calculations of Laplace's Equation Compared with the Second-Order Solution. 51

Figure 4.8 Results of Predictions with Highest Order on Wall, Lowest Order in Interior . . 52

Figure 5.1 Solution to the linear ODE $\frac{d^2 f}{dx^2} = (5)\sin(\pi x)$ 65

Figure 5.2 Point Movement. 66

Figure 5.3 Solution to the linear ODE $\frac{d^2 f}{dx^2} = (5\pi)^2 \sin(5\pi x)$ 67

Figure 5.4 Solution to the nonlinear ODE $\frac{\partial \phi}{\partial t} + \phi \frac{\partial \phi}{\partial x} - \frac{1}{Re}\frac{\partial^2 \phi}{\partial x^2} = 0$ 68

List of Tables

Table 4.1 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Circular Geometry, Evaluated at x = 0.5, y = 0.5 54

Table 4.2 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Rectangular Geometry, Evaluated at x = 0.5, y = 0.5 56

Table 4.3 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Irregular Geometry, Evaluated at x = 0.5, y = 0.5 57

Table 4.4 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Evaluated at x = 0.9, y = 0.8 58

blank page

1 Introduction

There are numerous important fluid and structural problems which require tracking a moving membrane responding to the forces induced by the motion of a fluid. Examples include the deployment and inflation process of a parachute and air bag; the aeroelastic phenomenon known as aileron buzz; and in the biomedical field, the opening and closing of pressure-driven heart valves and angioplasty balloon devices. Safety issues are also a concern in those areas where a rapid decompression may occur, such as a tire blowout at highway speeds. New techniques in mold processing using inflatable bladders may improve quality control by reducing anisotropic strain rates during manufacturing.

Fluid/structure problems, such as the examples listed above, are difficult to solve using existing numerical techniques. The difficulty in simulating coupled problems persists because the solution algorithm, in addition to solving for the fluid dynamic field variables (e.g., density, pressure, velocity, etc.), must also provide for the motion of both the membrane surface and the surrounding mesh. Transient remeshing of unstructured grids is computationally expensive, and convergence and solution accuracy degrade appreciably when the motion of the grid results in a highly distorted mesh. Typically, such grid related errors occur where large-scale structural deformation and/or large flow gradients occur, and it is precisely in those regions where the greatest accuracy is required.

In order to avoid the problems associated with grid generation and grid distortion, we proposed to develop a new technique which provides for the solution of both the field variables and the structural motion utilizing a gridless technology based on a random point distribution. The strength of the methodology lies in its ability to model any geometry in two and three dimensions using "clouds" of discrete points with no requirement for the points to be connected to form a grid as in conventional computational fluid dynamics (CFD) and structural mechanics algorithms. The governing partial differential equations (PDE's) which determine the equations of motion are solved locally at every point in the domain.

The proposed method is not a finite-difference, finite-volume, or finite-element approach since differences, metrics, areas, or volumes, are not computed. The methodology is extremely versatile in that the governing equations of motion for the fluid dynamics are not limited to one class of problems. That is, the method permits one to solve the relevant equations based on the complexity of the problem at hand. Hence, the difficulty of simulating either a potential flow, Euler flow, or full Navier-Stokes flow, and the ability to model compressible or incompressible flows is dictated only by the set of PDE's solved and is easily amenable to a modular code structure. This modularity appears as a simple consequence of the fact that the equations of motion are independent of the point distribution and the solution algorithm is completely grid free.

In this report we summarize research conducted during FY94 and FY95 on a gridless technique for fluid/structural dynamic coupling on flexible membranes. This project was funded by Sandia National Laboratories' Laboratory Directed Research and Development (LDRD) program. There were a number of principal investigators for this project. These were divided into smaller teams, each responsible for one part of the project. In this report, each chapter

describes one aspect of the project and was authored by those team members responsible for that aspect. If a chapter does not list specific authors, e.g., this chapter, then all members of the research team contributed to its content.

Chapter 2 presents an overview of gridless methods along with a brief survey of recent work by others in this area. We also give an example of the gridless method applied to a simple one-dimensional problem and present the concept of coupled linear operators for solving systems of differential equations. In Chapter 3 we give a description of the specific methods currently implemented in our gridless code and present some numerical results for our two-dimensional model problem: fluid flow described by Laplace's equation. Chapters 4 and 5 describe the results of efforts to improve the solution accuracy. Chapter 4 gives results ~~of~~ research into higher-order representation of the variables over the computational domain, and Chapter 5 presents the results of our investigations into point adaption. Chapter 6 contains results of research into shell elements, which are needed for coupled fluid/structural calculations. Chapter 7 contains a summary of the work and recommendations for the direction of continued research in gridless methods. Because of this modular report structure, each chapter is a semiautonomous unit with its own equations and nomenclature. Symbols are defined in each chapter as they are used. We have, therefore, decided not to include a nomenclature section for the entire report.

From its inception to completion, this project had several personnel turnovers. One of the individuals responsible for the original concept of this project and the LDRD proposal was Anthony Thornton, currently Director of Diversity Leadership and Education Outreach. Due to his promotion in early FY94, he was unable to participate in the actual research. Christine Hailey left Sandia at the beginning of FY95 to take a position at Utah State University. She was able to continue her involvement during the summer of 1995 through a university research contract. Walter Wolfe joined the project in January of 1995.

Because of unexpected difficulties with some of the tasks, LDRD funding cutbacks in FY95, and personnel turnovers, we were not able to complete all of the tasks originally planned. Extension of the method to the Navier-Stokes equations and the actual coupling of the fluid and structural calculations were not completed.

2 Overview of Gridless Methods

Roy S. Baty & Walter P. Wolfe

2.1 Introduction and General Gridless Philosophy

In this chapter we present a summary of gridless methods, a brief review of recent work by others in this area, and an outline of our current gridless method applied to a simple, one-dimensional problem. We also present an optimization of our calculations using the concept of coupled linear operators. The extension of the current gridless method to two dimensions is presented in Section 2.5 and Chapter 3.

The gridless method we present here uses Eulerian-based points arbitrarily distributed over the computational domain with no formal connectivity as typically required for a traditional grid. Each point in the domain has associated with it a cloud of neighboring points. These neighbors are chosen by a scheme that considers their position relative to the point of interest. Derivatives in the governing PDE's are numerically approximated by assuming a functional dependence of each variable within the local cloud of points, where the number of points in the cloud is greater than or equal to the number of unknown derivatives. The number of points is usually greater than the number of unknown derivatives in order to improve the accuracy of the derivative estimates. We have assumed here that the form of the functional dependence is a Taylor series. The governing partial differential equations are then solved locally at each point in the computational domain. This gives a solution to the governing equations without any a priori assumptions concerning grid or element topology, and allows for addition, deletion, or movement of points with no change in solution algorithms.

2.2 Previous Gridless Investigations

In recent years, a few individuals have begun to investigate gridless techniques. Liska and Orkisz[1] and Batina[2] have reported results of initial investigations into the use of gridless methods for two-dimensional applied mechanics and fluid flow applications, respectively. Liska and Orkisz calculated derivatives by assuming a second-order Taylor series expansion about the central point in a nine point cloud. This formulation gives eight equations for the calculation of five derivatives. To solve this overdetermined set of equations, they used a weighted least squares formulation. They chose a weighting factor of $1/\Delta_j^3$, where Δ_j is the distance between the cloud's central point and the jth neighboring point. The justification for the use of this weighting factor was not given. Boundary conditions were imposed through the use of ghost points.

One of the significant points to come from the work of Liska and Orkisz is the importance of point geometry when forming a local cloud. The most straightforward means of selecting points for a cloud is to pick the N points that are closest to the point of interest. However, this criterion often gives an ill-conditioned or singular set of equations due to an irregular distribution of points. They found that an additional criterion is often needed to insure that the points have a more uniform angular distribution about the point of interest. We confirmed this finding in our research.

Batina investigated the use of gridless methods for calculation of flow over 2-D airfoils, using both Euler and Navier-Stokes equations. He assumed a first-order Taylor series expansion for the cloud of points and used a least squares method to determine the derivatives. Details of the least squares method are not given. He also imposed boundary conditions by using ghost points. In the examples presented in reference 2, the points in the computational domain were determined by using an unstructured-grid generator and picking the centers of the resulting triangles as the points. The cloud of neighboring points was determined by choosing the centers of the triangles that shared sides with triangle of interest. This gives a four-point cloud where the points are reasonably evenly distributed around the point of interest. Any problems arising from irregular point distribution were, therefore, not addressed.

2.3 The Gridless Numerical Method

The gridless numerical method developed here consists of two parts: (1) a scheme that approximates a function and its derivatives and, (2) a scheme that combines the function/derivative approximation with iterative techniques to satisfy the differential equation and the boundary conditions. The solution proceeds in a interactive manner between these two parts, or steps, until a specified convergence criteria is met. To start the iterative process, an initial distribution for the function is assumed. The gridless scheme separates the approximation of the function and its derivatives from the solution of the differential equation so that arbitrary distributions of field points may be applied to compute the solution. In this section, a gridless method is derived for a 1-D model boundary value problem, illustrating the basic approach for linear differential equations.

To develop the scheme for a linear boundary value problem, consider the simple 1-D model problem defined on the closed interval [0,1]:

$$a_2\phi''(x) + a_1\phi'(x) + a_0\phi(x) = 0 \tag{2.1}$$

and

$$\phi(0) = \alpha \text{ and } \phi(1) = \beta \tag{2.2}$$

where α, β, and a_i for $i = 0, 1, 2$ are constants.

Now, let $\{x_n\}$ be any finite partition of [0,1] such that $x_0 = 0$, $x_N = 1$, and $x_j < x_k$ for all $j < k$. Note that $\{x_n\}$ is not assumed to be uniformly spaced. Next, recall that the Taylor series for the function ϕ expanded about any partition point x_i in [0,1] is given by:

$$\phi(x_i + h_j) = \phi(x_i) + \phi'(x_i)h_j + \frac{1}{2!}\phi''(x_i)h_j^{\,2} + \frac{1}{3!}\phi'''(x_i)h_j^{\,3} + \ldots \tag{2.3}$$

where $h_j = x_i - x_j$. Then truncating the Taylor series after the second derivative and rearranging yields:

$$\phi(x_i + h_j) - \phi(x_i) \cong \phi'(x_i)h_j + \frac{1}{2!}\phi''(x_i)h_j^{\,2} \tag{2.4}$$

Applying equation (2.4) at l ($l \geq 2$) points neighboring x_i produces the linear system:

$$D\xi = \eta \tag{2.5}$$

where

$$D = \begin{bmatrix} h_1 & \frac{1}{2!}h_1^2 \\ h_2 & \frac{1}{2!}h_2^2 \\ \cdots & \cdots \\ h_l & \frac{1}{2!}h_l^2 \end{bmatrix} \tag{2.6}$$

$$\xi = [\phi' \ \phi'']^T \tag{2.7}$$

and

$$\eta = [\phi_1 - \phi \ \ \phi_2 - \phi \ \ \ldots \ \ \phi_l - \phi]^T \tag{2.8}$$

Here the superscript T indicates the transpose of the vector. The boundary conditions given by equation (2.2) are satisfied by fixing the value of the solution ϕ at the points 0 and 1. For more general boundary value problems, such as potential flow, the derivative matrix D is reformulated to satisfy the boundary conditions on the boundary of the computational domain. The details of this reformulation are contained in subsection 3.2.3 on page 27.

As an example of how equations (2.6), (2.7) and (2.8) are evaluated, a typical calculation might use four grid points to evaluate the matrix equation (2.5): the point where the differential equation is solved and three neighboring points, generating a 3×2 matrix for D. Only determined and overdetermined linear systems are used to approximate the function and its derivatives. The determined case (D is a nonsingular square matrix) produces a unique solution for the derivatives, while the overdetermined case (D is a rectangular matrix with nullity of zero) yields a unique least squares solution. The underdetermined case where there are more unknowns than points has an infinite number of solutions and was not consider in this study.

The differential equation (2.1) defining the boundary value problem can also be written in matrix form as:

$$L\xi = v \tag{2.9}$$

where

$$L = \begin{bmatrix} a_1 & a_2 \\ 0 & 0 \\ \cdots & \cdots \\ 0 & 0 \end{bmatrix} \tag{2.10}$$

$$\xi = [\phi' \ \phi'']^T \tag{2.11}$$

and

$$v = \begin{bmatrix} -a_0\phi & \dots & 0 \end{bmatrix}^T \tag{2.12}$$

Equation (2.10) representing the differential operator is written as a matrix with the same row and column dimensions as the derivative matrix D by adding zeros as needed. This representation for L is used to simplify the analysis for the coupled linear operators presented in Sections 2.4 and 2.5.

To see how the basic gridless scheme works for the model problem, the local approximation defined by equations (2.5) and (2.9) must be applied at each grid point in (0,1). This requires defining the computational cells used to approximate the solution of the boundary value problem. Figure 2.1 shows the computational interval (0,1). The solid dots represent an arbitrary computational cell in this interval. For each grid point x_i in (0,1), choose at least two distinct points x_j neighboring x_i. Each x_i and its neighbors will constitute the computational cell, or cloud, for x_i, yielding the explicit equations for the local approximation. The boundary conditions (2.2) are satisfied by fixing the value of the solution ϕ at $x_0 = 0$ and $x_N = 1$.

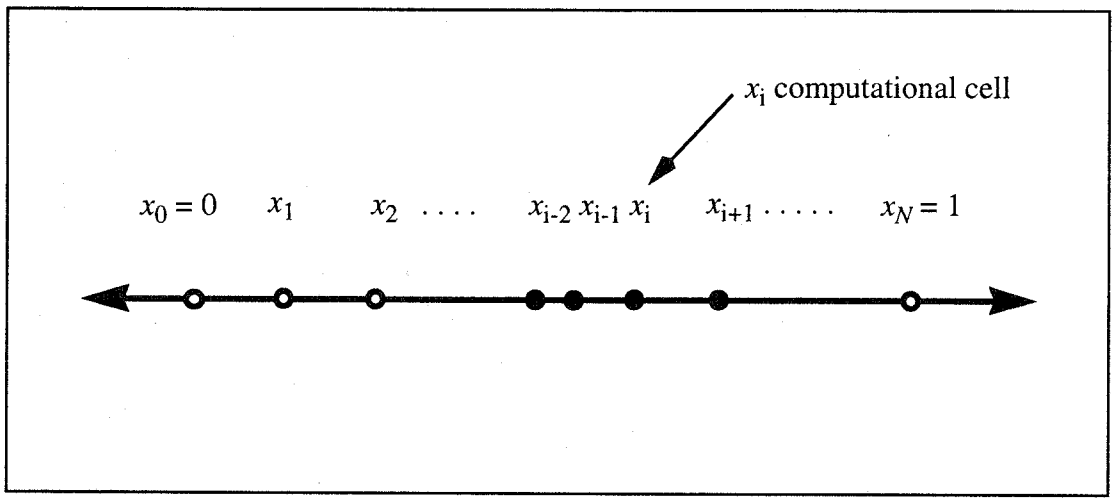

Figure 2.1 An arbitrary computational cell in the interval (0,1) for the gridless numerical method

Once the local approximation of the boundary value problem has been determined, the gridless scheme computes the solution using an iterative process. The following five steps are performed on the computational domain:

1. Guess a value for the solution $\phi(x_i)$ at each grid point in (0,1). Fix ϕ at $x_0 = 0$ and $x_N = 1$ so that the boundary conditions are satisfied.

2. On each computational cell, compute the approximate derivatives $\xi(\phi)$ from equation (2.5). Since, in general, equation (2.5) represents an overdetermined system of equations, a least squares scheme must be used to compute ξ.
3. Substitute ξ and ϕ into matrix equation (2.9) and define a local residual as a function of ϕ:

$$R_L(\phi) = L\xi - v \tag{2.13}$$

4. Compute ϕ such that $-\varepsilon < R_L(\phi) < \varepsilon$ for some small value $\varepsilon > 0$. Any convergent root-finding scheme can be used. Steps 2 through 4 are repeated at each point in the computational domain.
5. After steps 2 through 4 are completed for each cell in the domain, a global residual is computed and checked for convergence. The global residual, R_G, is defined as

$$R_G = \sqrt{\sum_1^N R_L^2} \tag{2.14}$$

where N is the number of field points in the computational domain. If the global convergence criteria has not been met, the solution procedure repeats starting with step 2.

To measure the convergence of this scheme, a global convergence criteria is defined as:

$$\frac{R_G^n}{R_G^0} < C_G \tag{2.15}$$

where R_G^n is the nth iteration global residual, R_G^0 is the initial starting solution global residual, and C_G is the global convergence criteria. The basic gridless scheme consisting of steps 1 through 5 is computed iteratively until equation (2.15) is satisfied. Numerical examples of the convergence and accuracy of this scheme are given in Chapter 3 for potential flow.

2.4 Least Squares Solutions for Coupled Linear Systems

In the preceding section, a general numerical scheme was derived to approximate solutions of differential equations on irregular meshes. For arbitrary linear differential equations, this numerical technique yields coupled systems of linear equations on each local computational cell. This section outlines an algebraic method to find the solution of an arbitrary coupled linear system on an arbitrary computational cell. The solution obtained for the coupled system on each computational cell at each solution iteration will be a least squares solution which will minimize the local residual computed in step 3 above for a fixed value of v. Recall that least squares solutions minimize the error in terms of Euclidean length.

Consider the following system of linear matrix equations:

$$D\xi = \eta \tag{2.16}$$

$$L\xi = v \tag{2.17}$$

where D and L are $M \times N$ matrices, ξ is a N-component vector, and η, v are M-component vectors. On an arbitrary computational cell for some solution iterate, $D\xi = \eta$ approximates the derivatives, ξ, of the unknown function, η, and $L\xi = v$ approximates the differential equation. Equations (2.16) and (2.17) define a coupled linear system.

In general, coupled linear systems do not have solutions. As an example of a coupled linear system with no solutions, suppose that equations (2.16) and (2.17) represent the simple scalar equations

$$a\xi = \eta \tag{2.18}$$

$$b\xi = v \tag{2.19}$$

where a, b, η and v are any nonzero scalar values such that $\eta/a \neq v/b$. Notice that the coupled equations (2.18) and (2.19) are solvable only in the event

$$\eta/a = v/b \tag{2.20}$$

Hence, a solution to equations (2.18) and (2.19) does not exist. However, a least squares solution may be found. The least squares solution is the best approximation to equations (2.18) and (2.19) in the sense that it minimizes the error defined in terms of the Euclidean norm.

The problem defined by the coupled linear system (2.16) and (2.17) has been solved in a general algebraic setting for abstract linear operators by Bryan, Kinyon and Tucker.[3] Their work is applicable to any coupled linear system. Following the development of reference 3, a new matrix G is introduced:

$$G = \begin{bmatrix} D \\ L \end{bmatrix} \tag{2.21}$$

So, the coupled linear system then becomes

$$G\xi = z \tag{2.22}$$

where $z = [\eta \;\; v]^T$. Now, a least squares solution of (2.22) is a least squares solution of the original problem defined by equations (2.16) and (2.17). And applying the results of Bryan, Kinyon and Tucker, the least squares solution to equation (2.22) is given by

$$G^+\begin{bmatrix} \eta \\ v \end{bmatrix} = \frac{1}{2}\,[L^+v + (I - L^+L)D^+\eta] + \frac{1}{2}\,[D^+\eta + (I - D^+D)L^+v] \tag{2.23}$$

where G^+, D^+, and L^+ are the Moore-Penrose inverses[4,5] of G, D, and L respectively.

Therefore, to obtain a least squares approximation to the computational problem defined by equations (2.16) and (2.17), equation (2.23) must be computed on each interior computational cell at each solution iteration. Now, to see how equation (2.23) might be applied in practice, an example is given. In the coupled linear matrix problem, let D and L be the rectangular matrices

$$D = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 2 & 2 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} \quad \text{and} \quad L = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{2.24}$$

and let the vectors η and v in equations (2.16) and (2.17) be given by:

$$\eta = [2\ 0\ 2\ 2]^T. \tag{2.25}$$

and

$$v = [0\ 0\ 0\ 0]^T. \tag{2.26}$$

The differential operator L is written as a matrix with the same row and column dimensions as the derivative matrix D by adding zeros as needed. This representation for L is used to simplify the analysis for the coupled linear operators. In this example, the matrix D is just a simple matrix with integer entries which has no immediate physical or numerical interpretation. However, the matrix L can be associated with a linear differential operator and a matrix similar to L will be applied to approximate Laplace's equation in Section 2.5.

To compute the least squares solution of (2.21), first notice that the Moore-Penrose inverses of D and L are given by:

$$D^+ = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 3 & 3 & -1 & -1 \\ 2 & 4 & 1/2 & 1/2 \\ 2 & 1 & 1 & 0 \\ 2 & 1 & 0 & 1 \end{bmatrix}^{-1} \tag{2.27}$$

and

$$L^+ = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \tag{2.28}$$

Analytical methods to evaluate the Moore-Penrose inverses of these rectangular matrices are developed in Campbell and Meyer;[4] however, in numerical practice, the singular value decomposition (SVD) method can be used to compute the Moore-Penrose inverses.[5]

Substituting equation (2.26) into (2.23) yields

$$G^+ \begin{bmatrix} \eta \\ v \end{bmatrix} = \frac{1}{2}\,(I - L^+L)D^+\eta + \frac{1}{2}\,D^+\eta \tag{2.29}$$

which reduces to

$$G^{+}\begin{bmatrix}\eta\\ v\end{bmatrix} = (I - \frac{1}{2}L^{+}L)D^{+}\eta \tag{2.30}$$

To evaluate equation (2.30) numerically, equation (2.27) and the first term of equation (2.30) are simplified, yielding

$$D^{+} = \begin{bmatrix} 0.14286 & -0.23810 & 0.26190 & 0.26190 \\ 0.00000 & 0.33333 & -0.16667 & -0.16667 \\ 0.14286 & 0.09523 & 0.09523 & 0.09523 \end{bmatrix} \tag{2.31}$$

and

$$(I - \frac{1}{2}L^{+}L) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 3/4 & -1/4 \\ 0 & -1/4 & 3/4 \end{bmatrix} \tag{2.32}$$

Combining equations (2.31) and (2.32) then gives

$$(I - \frac{1}{2}L^{+}L)D^{+} = \begin{bmatrix} 0.14286 & -0.23810 & 0.26190 & 0.26190 \\ -0.03572 & 0.22619 & -0.14881 & -0.14881 \\ 0.10715 & -0.01191 & 0.11309 & 0.11309 \end{bmatrix} \tag{2.33}$$

Finally, applying $y = [2\ 0\ 2\ 2]^{T}$ to this matrix produces the least squares solution

$$x = (I - \frac{1}{2}B^{+}B)D^{+}\eta = \begin{bmatrix} 4/3 \\ -2/3 \\ 2/3 \end{bmatrix} \tag{2.34}$$

Now, to see that this solution satisfies the coupled linear system, substituting the right-hand-side of (2.34) and equations (2.24) into equations (2.16) and (2.17) yields

$$D\xi = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 2 & 2 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}\begin{bmatrix} 4/3 \\ -2/3 \\ 2/3 \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 2 \\ 2 \end{bmatrix} \tag{2.35}$$

and

$$L\xi = \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 4/3 \\ -2/3 \\ 2/3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.36}$$

as required. Notice that the solution ξ computed here is not unique; although it is an optimal solution in the least squares sense.

2.5 Least Squares Solutions for the Gridless Method

This section applies the least squares solutions for coupled linear systems to an example of the gridless method. The specific gridless scheme developed here is used to simulate steady, incompressible, potential flow in two-dimensions. Potential flow is described by Laplace's equation:

$$\nabla^2\phi = 0 \tag{2.37}$$

subject to the boundary conditions:

$$\nabla\phi \cdot \hat{n} = 0 \text{ on stationary solid bodies} \tag{2.38}$$

and

$$\nabla\phi \text{ approaches the freestream velocity as } r \to \infty. \tag{2.39}$$

Here, in equation (2.38), $\hat{n}$ is a normal vector for the solid bodies. In rectangular coordinates equations (2.37) and (2.38) may be written as:

$$\phi_{xx} + \phi_{yy} = 0 \tag{2.40}$$

and

$$\phi_x n_1 + \phi_y n_2 = 0 \tag{2.41}$$

where $\hat{n} = [n_x \;\; n_y]^T$ is the normal vector.

The linear partial differential equation (2.40) requires the numerical approximation of the second derivatives of ϕ. Therefore, the Taylor series used to approximate the derivatives of D will be truncated after the second order terms:

$$\phi_j \cong \phi_i + h_j\phi_x + k_j\phi_y + \frac{1}{2}h_j^{\,2}\phi_{xx} + \frac{1}{2}k_j^{\,2}\phi_{yy} + h_jk_j\phi_{xy} \tag{2.42}$$

Equation (2.42) is expanded about the field point (x_i, y_i) where $h_j = x_i - x_j$ and $k_j = y_i - y_j$. For the examples computed in this report, each computational cell in the flow field will assume nine grid points: a single field point where ϕ is approximated and eight surrounding field points. This yields a 8×5 derivative matrix D, a 5-component derivative vector ξ and an 8-component potential-value vector η:

$$D = \begin{bmatrix} h_1 & k_1 & \frac{1}{2}h_1^2 & \frac{1}{2}k_1^2 & h_1k_1 \\ h_2 & k_2 & \frac{1}{2}h_2^2 & \frac{1}{2}k_2^2 & h_2k_2 \\ \dots & \dots & \dots & \dots & \dots \\ h_8 & k_8 & \frac{1}{2}h_8^2 & \frac{1}{2}k_8^2 & h_8k_8 \end{bmatrix} \tag{2.43}$$

$$\xi = [\phi_x \ \phi_y \ \phi_{xx} \ \phi_{yy} \ \phi_{xy}]^T \tag{2.44}$$

and

$$\eta = [\phi_1 - \phi \ \ \phi_2 - \phi \ \ \dots \ \ \phi_8 - \phi]^T \tag{2.45}$$

On solid bodies, the boundary condition given by equation (2.38) must be satisfied. This is accomplished by substituting $\nabla\phi \cdot \hat{n} = 0$ into the Taylor series so that the derivative matrix D is modified. For the numerical examples presented in Chapter 3, six field and boundary points are used to approximate the potential flow at each grid point of the body. Furthermore, the equation (2.41) is combined with the Taylor series so that singularities do not occur in the event that $n_x = 0$ or $n_y = 0$ (see Subsection 3.2.3 for details).

The next step in developing the gridless method is to obtain a matrix representation for Laplace's equation (2.40). Since this differential equation is a linear combination of two components of the derivative vector, a 8×5 matrix L representing the Laplacian is given by:

$$L = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{2.46}$$

where the image vector is zero:

$$v = [0 \ \ 0 \ \dots \ 0]^T \tag{2.47}$$

Notice that the dimensions of L are chosen to agree with the dimensions of the derivative matrix. This representation for L will simplify the numerical evaluation of the gridless numerical scheme on a computational cell.

Now to compute the least squares solution of the coupled linear system:

$$D\xi = \eta \tag{2.48}$$

$$L\xi = v \tag{2.49}$$

defined by equations (2.43), (2.45), (2.46) and (2.47), the Moore-Penrose inverses must be computed for D and L. The Moore-Penrose inverse for D is computed numerically using

singular-value-decomposition, while the Moore-Penrose inverse for L is computed analytically. Recall that L is very similar to the example derived in Section 2.4. This analysis implies that L^+ is given by:

$$L^+ = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \tag{2.50}$$

Simplifying L^+ yields the 5×8 matrix:

$$L^+ = \begin{bmatrix} 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \\ 1/2 & 0 & \dots & 0 \\ 1/2 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 \end{bmatrix} \tag{2.51}$$

Substituting equations (2.50) and (2.51) into the first term of (2.30) produces:

$$(I - \frac{1}{2} L^+ L) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 3/4 & -1/4 & 0 \\ 0 & 0 & -1/4 & 3/4 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{2.52}$$

Finally, ξ is evaluated by computing $D^+\eta$ using the singular-value-decomposition and then applying the result as shown in equation (2.52).

Blank Page

3 Current Gridless Scheme

Walter P. Wolfe, Roy S. Baty, & Glenn A. Laguna

3.1 Overview

In this chapter we present a description of the procedures used in our current two dimensional gridless code and then present numerical results for our 2-D model problem. For our model problem, we chose fluid flow modeled by Laplace's equation because exact solutions are readily obtainable and this allows us to quantify the accuracy of the method. Only the basic code is described in this chapter. Enhancements, such as higher-order distributions within a cloud and point adaption, are presented in later chapters. The code is written in ANSI standard C.

The current code does not have a built-in point generator. The initial distribution of points must be generated externally and are read from a grid file. A boundary condition flag is associated with each point to indicate whether that point is an interior point or a boundary point, and, if a boundary point, what type of boundary. Some of the test versions of the code also read in the exact solution in order to compute the final solution error.

3.2 Method Development

3.2.1 Cloud Formation

The only requirement for the distribution of points in a gridless method is that it allows an accurate solution of the governing equations over the entire computational domain. Each point has associated with it a cloud of neighboring points that are used to determine the derivatives needed to solve the governing equations. In the following discussion, we have chosen the word *focus* to designate this central point and the word *neighbors* to designate the other members of a cloud.

We have assumed that the points are read into the code in a completely random order, i.e., there is no underlying structure to the point array. Therefore, the entire array of points must be searched for each focus to find the points that form its cloud of nearest neighbors. Although this is a time consuming process, it need only be done once for each point during the initial problem setup and, therefore, is not a large fraction of the total run time. Once found, the neighboring points' indices are stored in each focus' data structure. A point's index number is its position in the original point array. We made no attempt to develop efficient functions for this search.

The search procedure proceeds as follows for each point in the interior of the domain:

1. For each point or focus, the distances to all other points are computed.
2. This array of distances is sorted from smallest to largest using a Quicksort function from reference 10.
3. The closest point's index number is stored in the focus' cloud array.
4. The next closest point is then checked to see if it is suitable for inclusion within the cloud. After the first point, the angular position of each point relative to the focus is checked against the angular position of each of the previously selected neighbors. If

the point lies within a wedge of $\pm 180/N°$, centered on the focus, from any of the previously selected cloud members, then that point is rejected. N is the number of neighboring points which form the cloud.

5. The routine repeats step 4 until the set of N neighboring points has been selected.

We found that step 4 was necessary for a stable and accurate solution procedure. Without this step, the points of the cloud could be biased toward one side of the focus, resulting in an ill-conditioned matrix.

It is possible for the above search procedure to pick points that are on the other side of the body from the cloud's focus. In practice, this never happened with the interior points, but it did occasionally occur when selecting points for clouds on the body's boundary. For boundary points on the body's surface, the following steps were performed before the above procedure was initiated:

1. The two closest boundary points, one on either side of the focus, were first added to the cloud.
2. The plane tangent to the body at the focus was computed.
3. Only those interior (non-boundary) points that lay "above" this tangent plane, where "above" means in the direction of the outward normal, were included in the above search.

3.2.2 Cloud Equations

The equations presented in this subsection are a repeat of the equations presented in Section 2.5. They are repeated here to facilitate the following discussion. Within each cloud of points, we modeled the functional variation of the unknowns in the governing equations with a 2nd-order Taylor series expansion about the cloud focus, i.e.,

$$f(x, y) = f(x_0, y_0) + \{hf_x(x_0, y_0) + kf_y(x_0, y_0)\} + \frac{1}{2!}\{h^2 f_{xx}(x_0, y_0) + 2hkf_{xy}(x_0, y_0) + k^2 f_{yy}(x_0, y_0)\} \tag{3.1}$$

where (x_0, y_0) are the coordinates of the focus, h and k are given by $h = x - x_0$ and $k = y - y_0$, and the subscripts x and y denote partial derivatives in the usual notation.

Writing equation (3.1) for each of the nodes in the cloud results in the set of linear equations of the form

$$[A][Df] = [f] \tag{3.2}$$

where

$$[A] = \begin{bmatrix} h_1 & k_1 & \frac{h_1^2}{2} & h_1 k_1 & \frac{k_1^2}{2} \\ h_2 & k_2 & \frac{h_2^2}{2} & h_2 k_2 & \frac{k_2^2}{2} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ h_m & k_m & \frac{h_m^2}{2} & h_m k_m & \frac{k_m^2}{2} \end{bmatrix}$$

$$[f]^T = [f_1 - f_0 \quad f_2 - f_0 \quad \cdots \quad f_m - f_0]$$

$$[Df]^T = \left[\left.\frac{\partial f}{\partial x}\right|_0 \quad \left.\frac{\partial f}{\partial y}\right|_0 \quad \left.\frac{\partial^2 f}{\partial x^2}\right|_0 \quad \left.\frac{\partial^2 f}{\partial x \partial y}\right|_0 \quad \left.\frac{\partial^2 f}{\partial y^2}\right|_0 \right]$$

and $m \geq 5$.

The number of nodes used to determine the $[Df]$ vector may be increased ($m > 5$) in order to improve the accuracy in approximating the derivatives, resulting in an overdetermined set of linear equations. Equation (3.2) is solved using the singular value decomposition (SVD) routines from reference 10. The use of SVD in the current gridless code offers several advantages: (1) SVD gives the least squares solution for an overdetermined set of equations, and (2) if the $[A]$ matrix is ill-conditioned or singular, SVD can detect the problem and, in some cases, can provide a solution. Within a cloud, the $[A]$ matrix is constant unless the cloud's points move or change. Therefore, this matrix is formed and its decomposition calculated and stored at the beginning of the solution procedure. Derivatives are then easily recomputed whenever the $[f]$ matrix changes. In the example problems described in Section 3.3, nine points were used for the interior clouds and six points for the boundary clouds.

3.2.3 Boundary Conditions

Rather than use ghost points to impose boundary conditions, we decided to incorporate them directly into the cloud equations. We have examined only two types of boundaries, far field and solid surface. Far field boundary conditions were imposed by fixing the values of the unknown field variables. At wall boundaries, we examined two different methods for imposing boundary conditions.

The normal velocity condition at a solid surface is expressed as

$$\hat{n} \bullet \vec{V} = \vec{V_s} \tag{3.3}$$

where $\hat{n}$ is the unit normal to the surface, $\vec{V}$ is the fluid velocity vector, and $\vec{V_s}$ is the surface velocity. For the remainder of this discussion, we assume that $\vec{V_s} = 0$. For a two dimensional problem, this reduces to

$$n_x V_x + n_y V_y = 0 \tag{3.4}$$

This equation can be rearranged to give

$$V_x = -\frac{n_y}{n_x} \cdot V_y \quad \text{for} \quad n_x \neq 0 \tag{3.5}$$

or

$$V_y = -\frac{n_x}{n_y} \cdot V_x \quad \text{for} \quad n_y \neq 0 \tag{3.6}$$

If the function f in equation (3.1) is the velocity potential, ϕ, then superimposing the above boundary condition on equation (3.1) gives

$$\phi = \phi_0 + \left(k - \frac{n_y}{n_x}h\right)\phi_y\Big|_0 + \frac{h^2}{2}\phi_{xx}\Big|_0 + \frac{k^2}{2}\phi_{yy}\Big|_0 + hk\phi_{xy}\Big|_0 \quad \text{for} \quad n_x \neq 0 \tag{3.7}$$

or

$$\phi = \phi_0 + \left(h - \frac{n_x}{n_y}k\right)\phi_x\Big|_0 + \frac{h^2}{2}\phi_{xx}\Big|_0 + \frac{k^2}{2}\phi_{yy}\Big|_0 + hk\phi_{xy}\Big|_0 \quad \text{for} \quad n_y \neq 0 \tag{3.8}$$

The first two columns in the $[A]$ matrix in equation (3.2) are replaced by the appropriate terms from equation (3.7) or (3.8), depending on the relative magnitudes of n_x and n_y. The component with the largest absolute value is always placed in the denominator. After solution of equation (3.2), equation (3.5) or (3.6) is used to determine the final derivative.

The second way of imposing the wall boundary conditions was to add equation (3.4) as the $m + 1$ equation in the matrices of equation (3.2). Neither method seems to offer an advantage in either speed or accuracy over the other. The second method has the advantage of being cleaner and easier to program than the first.

3.2.4 Solution Procedure

The solution procedure used for the example calculations given below follows the procedure described in Section 2.3 on page 14. The solution procedure consists of two nested iteration loops. The outer, or global, loop spans the computational domain and uses a Gauss-Seidel scheme to reduce the global residual of the governing equation. We found that a Gauss-Seidel scheme converges two to three times faster than a scheme using Jacobi iteration. At the beginning of each global loop, the values of the velocity potential at each point in the domain are updated, the set of derivatives computed, and the local residuals from the governing equation sorted from largest to smallest. The inner, or local, loop is then begun, starting at the point with the largest local residual and proceeding in sorted order to the smallest. The inner loop calculations are performed for each cloud of points to find the value of the velocity potential at the cloud focus that temporarily brings the local residual of the governing equation to $0 \pm \varepsilon$. For the test cases discussed below, $\varepsilon = 1\times10^{-6}$. Smaller values of ε were tried, but did not improve the overall solution accuracy. These local calculations were performed using Brent's method.

This is a modified secant search routine that is guaranteed not to diverge. If the routine detects any divergence in the search, it switches to a bisection routine. Timings showed that this method converged two to three times faster than pure bisection. The actual C function for Brent's method was taken from reference 10. Attempts to use a Newton-Raphson scheme were not successful. After the calculations for the last cloud are completed, the local and global residuals are recomputed. The calculations are terminated when the global residual is reduced by five orders of magnitude from the global residual of the starting solution. If the solution is not converged, another global iteration is performed.

3.3 Numerical Examples

In this section, the gridless numerical method is applied to simulate steady, incompressible, potential flow around simple two-dimensional cylinders. The inviscid flow field is computed for two geometries: a circular cylinder and an elliptic cylinder of aspect ratio 5:1. We first describe these model problems and their exact analytical solutions. We then give the numerical results of the gridless method for the model problems and discuss the types and magnitudes of numerical errors discovered. The numerical results presented here were generated with and without application of the coupled linear operator described in Sections 2.4 and 2.5. Comparison of the gridless method with and without the coupled linear solutions shows that the basic gridless scheme closely approximates the optimal coupled solution for a given spatial resolution. Overall, the numerical results of the gridless method are in very good agreement with the known exact solutions for the circular and elliptic cylinders.

3.3.1 Model Potential Flow Problems

This section describes the exact analytical solutions to two simple model flow problems. To define the model problems, consider steady, incompressible, potential flow around an arbitrary two-dimensional cylinder with no circulation. The first model problem is to compute this ideal flow field for the circular cylinder of radius one, while the second model problem is to compute this flow for an elliptic cylinder of aspect ratio 5:1. Recall that these idealized flows are described by Laplace's equation subject to simple boundary conditions on the surface of the cylinders and at infinity, i.e.,

$$\begin{gathered} \nabla^2\phi = 0 \\ \nabla\phi \cdot \hat{n} = 0 \text{ on stationary solid bodies} \\ \nabla\phi \text{ approaches the freestream velocity as } r \to \infty \end{gathered} \tag{3.9}$$

For both model problems, the gridless method is used to compute the velocity potential ϕ around the cylinders as specified by these equations. The results were used to compute the pressure coefficient on the surface of the circular and elliptic cylinders. The computed velocity potentials and pressure coefficients are then compared with the exact analytical solutions of the model problems.

As is well known from classical aerodynamics, the complex velocity potential around a circular cylinder with no circulation is given by

$$F(\zeta) = U\zeta + \frac{A}{\zeta} \tag{3.10}$$

where $\zeta = x + iy$, U is the freestream velocity, and $A = a^2 U$. Here a is the radius of the circular cylinder. All the numerical results in this report assume $a = 1$ and $U = 1$. Recall, that the complex velocity potential is defined by

$$F(x + iy) = \phi(x, y) + i\psi(x, y) \tag{3.11}$$

where ϕ is the velocity potential, and ψ is the stream function. Differentiating the complex velocity potential with respect to ζ then yields the complex velocity. Therefore, equation (3.10) implies that the exact solution for the complex velocity around a circular cylinder with no circulation is given by

$$W(\zeta) = U\left(1 - \frac{a^2}{\zeta^2}\right) \tag{3.12}$$

The exact analytical solution for potential flow around an elliptic cylinder can be obtained in a similar manner by transforming the complex potential obtained for the circular cylinder via conformal mapping. Let $z(\zeta)$ denote a conformal mapping which carries the unit circular cylinder onto an elliptic cylinder. It can be shown that the Joukowski map defined by

$$z(\zeta) = \zeta + \frac{\eta}{\zeta} \tag{3.13}$$

with $\eta = 2/3$ will map the unit cylinder onto an elliptic cylinder of aspect ratio 5:1. Now, recall that z maps the velocity potential of the circular cylinder to the velocity potential of the elliptic cylinder. So, given the velocity potential for the circular cylinder, $F(\zeta)$, the velocity potential for the elliptic cylinder becomes

$$F(z(\zeta)) \tag{3.14}$$

where $\zeta = z^{-1}(\zeta)$ is the inverse of equation (3.13). Then differentiating equation (3.14) with respect to ζ and applying equation (3.13) yields the complex velocity for the elliptic cylinder

$$\hat{W}(z(\zeta)) = W(\zeta)\left(\frac{1}{\frac{dz}{d\zeta}}\right) \tag{3.15}$$

where $W(\zeta)$ is the complex velocity around the circular cylinder and

$$\frac{dz}{d\zeta} = 1 - \frac{\eta}{\zeta^2} \tag{3.16}$$

Once the complex velocity has been determined, the pressure coefficient C_p on the surface of a given cylinder may be computed from

$$C_p = 1 - W^2 \tag{3.17}$$

where W is given by equation (3.12) for the circular cylinder or equation (3.15) for the elliptic cylinder. Further details of the model problems outlined here and their exact analytical solutions may be found in Karamcheti.[6]

3.3.2 Numerical Results

To test the gridless numerical method, several computational experiments were conducted. This section presents numerical results generated by these experiments for three versions of the gridless scheme for both model flow problems. The first gridless scheme considered is referred to as the basic scheme and applies the algorithm defined by steps 1 through 5 on page 16 of Section 2.3 without modification. The second and third gridless schemes are referred to as coupled-schemes one and two and modify how the algorithm approximates the derivatives and the differential equation on each local computational cell. Coupled-scheme one computes the least squares solution for the coupled linear system defined by equations (2.48) and (2.49) (the approximation of the derivatives and the differential equation) on each interior cell in the computational domain. Coupled-scheme two generalizes scheme one by computing a least squares solution for the coupled system given by (2.48) and (2.49) on the boundary as well as the interior cells of the computational domain. On the interior of the computational domain, the local coupled system is evaluated using equations (2.30) and (2.52) developed in Sections 2.4 and 2.5.

For all of these calculations, the problem was initialized by assuming that the velocity potential was equal to the freestream velocity potential at each field point. The potentials at the outer boundary points were set equal to those of the exact solution potential to satisfy the boundary condition at infinity. This was done so that any errors in the solution would be due to the gridless numerical method and not attributable to boundary conditions. The calculations proceeded until the global residual was reduced by more than five orders of magnitude from the global residual of the starting solution, i.e.,

$$\frac{R_G^n}{R_G^0} < 10^{-5} \tag{3.18}$$

The global residual is defined as the root-sum-square of the local residuals at each field point. The local residual is the deviation of the governing equation from zero, i.e.,

$$R_L = \nabla^2\phi \tag{3.19}$$

Since we know the exact solution for these test cases, we have also defined a local error, E_L, as the difference between ϕ exact and ϕ calculated

$$E_L = \phi_{\text{exact}} - \phi_{\text{calculated}} \tag{3.20}$$

The global error, E_G, is defined as the root-sum-square of the local errors

$$E_G = \sqrt{\sum_{1}^{N} E_L^2} \tag{3.21}$$

where N is the number of points in the computational domain.

Circular Cylinder Calculations

Figures 3.1 and 3.2 show the point distribution used in the circular cylinder calculations. There are 80 points in the circumferential direction and 41 points in the radial direction, for a total of 3280 points. The inner layer of points lie on the body's surface. The point locations where chosen coincident with polar coordinate lines so that the point distribution is uniform in the angular direction. In the radial direction, the spacing increases with a geometric ratio of 1.05 as one moves away from the body. This calculation took 1423 cpu seconds on a 120 MHz Sun SPARC 20 workstation.

Figure 3.3 shows the behavior of the global residual and error during this calculation. While the global residual continues to decrease as the solution progresses, the global error reaches a minimum just past an iteration count of 800 and then asymptotically approaches a steady-state value of approximately 9×10^{-3}, i.e., two orders of magnitude reduction from the starting solution error. Figure 3.4 shows the distribution of the local error around the cylinder. The largest local error is less that one percent of the exact solution and occurs at the flow stagnation points. To obtain a second quantitative view of this error, Figure 3.5 shows the pressure coefficient on the surface of the cylinder. Here, the zero circular cylinder angle is aligned with the X-axis. Since the gridless scheme yields a symmetric solution about the X-axis, the angle can be measured either clockwise or counter-clockwise. Figure 3.5 shows that the gridless scheme generated a pressure coefficient in very good agreement with the exact analytical result.

The overshoot in the global velocity potential error before approaching a steady-state value (Figure 3.3) is believed to be a numerical oddity with no real significance. During the code development, other solution techniques were examined. For some of them, the solution error approached the same asymptotic value from above, i.e., with no overshoot. We believe that given this point distribution and second-order derivative approximation, this is the best solution that the current method can achieve. In order to improve the solution accuracy, point adaption and/or a higher-order derivative approximation are needed, at least in the vicinity of the stagnation points.

A second test calculation was made for flow over the cylinder with the number of field points doubled in both directions, i.e., 160 by 82, for a total of 13,120 points. The convergence characteristics are shown in Figures 3.6 and 3.7. The distribution of the local error around the cylinder is shown in Figure 3.8. The increased spatial resolution reduced the global error by a factor of approximately 4.5. However, the number of global iterations required to achieve the same level of residual convergence increased by almost a factor of approximately 3.75. The computer time required to achieve this solution increased by a factor of approximately 16 over the previous solution, indicating that the current implementation gives an N^2 method. We have made no effort to optimize the efficiency of the current code.

We also compare the basic gridless scheme with the two coupled schemes for potential flow around the circular cylinder. Figure 3.9 compares the global residual of the basic scheme with the coupled gridless schemes one and two. Both coupled schemes yield the same global residual as a function of iteration number. Moreover, the coupled schemes produce a residual which was slightly smaller than the basic scheme. For a given approximation of the derivatives and the differential equation, the least squares solution generates the approximation with the smallest residual. This suggests that the basic gridless scheme closely approximates the optimal least squares solution for a given spatial resolution. Figure 3.10 shows the global errors computed with all three schemes. They all produced the same global error as a function of iteration number, since all three schemes performed a local iteration on each computational cell by calculating the root of an equation. Recall that the local iteration process is defined in step 4 of the basic gridless method. The differences between the coupled and uncoupled gridless schemes were so small that the local iteration yielded the same global error. One main conclusion of Figures 3.9 and 3.10 is that the basic scheme produces numerical approximations which are essentially identical to the optimal coupled solutions.

Elliptic Cylinder Calculations

Calculations were also performed for potential flow around an elliptic cylinder of aspect ratio 5:1. Figures 3.11 and 3.12 show the field point distributions used in these calculations. For this example, the field point locations were obtained through the conformal mapping of the points for the circular cylinder (Figure 3.1). As for the first model problem, these calculations used 80 points circumferentially and 41 points radially for a total of 3280 field points. Figure 3.13 shows the convergence properties for this calculation using the basic gridless scheme. As with the circular cylinder, the global error approaches an asymptotic value while the global residual continues to decrease. Figure 3.14 shows the spatial distribution of the local error. The largest error observed is less then six-tenths of a percent of the exact solution and occurs at the flow stagnation points. Figure 3.15 shows the pressure coefficient on the surface of the cylinder. The zero elliptic cylinder angle is aligned with the X-axis. As in the circular case, the gridless scheme yields a solution symmetric about the X-axis, hence, the angle of Figure 3.15 can be measured either clockwise or counter-clockwise. The plot of the pressure coefficient demonstrates that the basic gridless scheme generated numerical results in excellent agreement with the exact analytical solution.

Figures 3.16 and 3.17 compare the global residual and error of the basic gridless scheme with both coupled gridless schemes. The numerical results shown in these figures are qualitatively similar to the results found for the circular cylinder. However, the global residuals computed with the coupled gridless schemes are almost identical to the residuals computed with the basic scheme. For the elliptic cylinder, the basic scheme approximates the coupled solution very well.

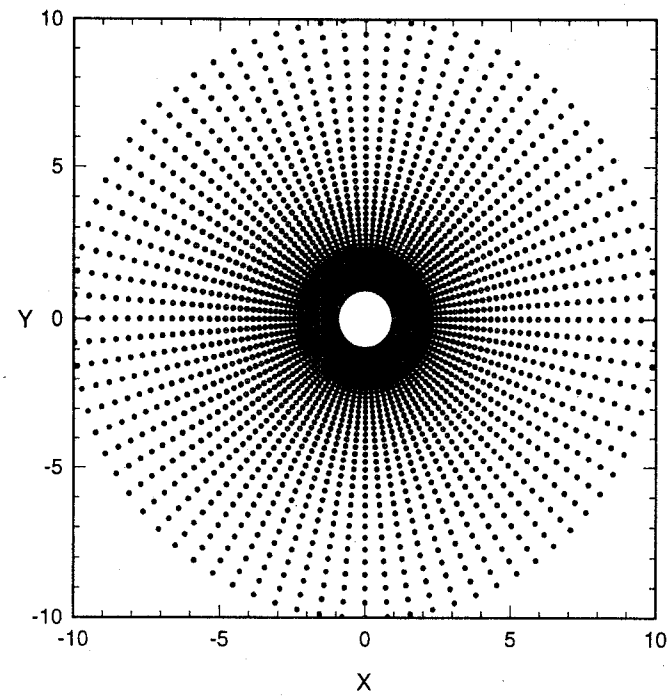

Figure 3.1 Field points for calculation of the potential flow around a circular cylinder - 3280 points

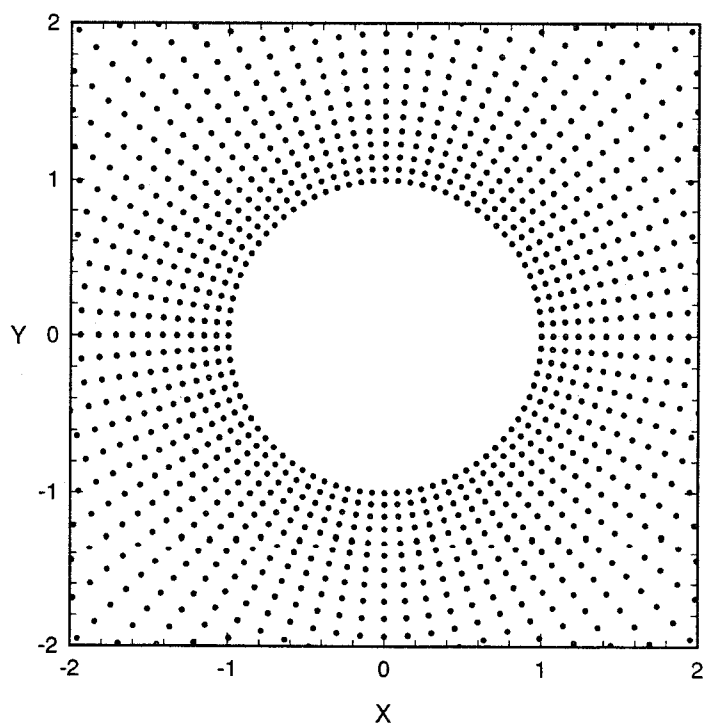

Figure 3.2 Close-up view of the field point distribution in Figure 3.1 for the circular cylinder

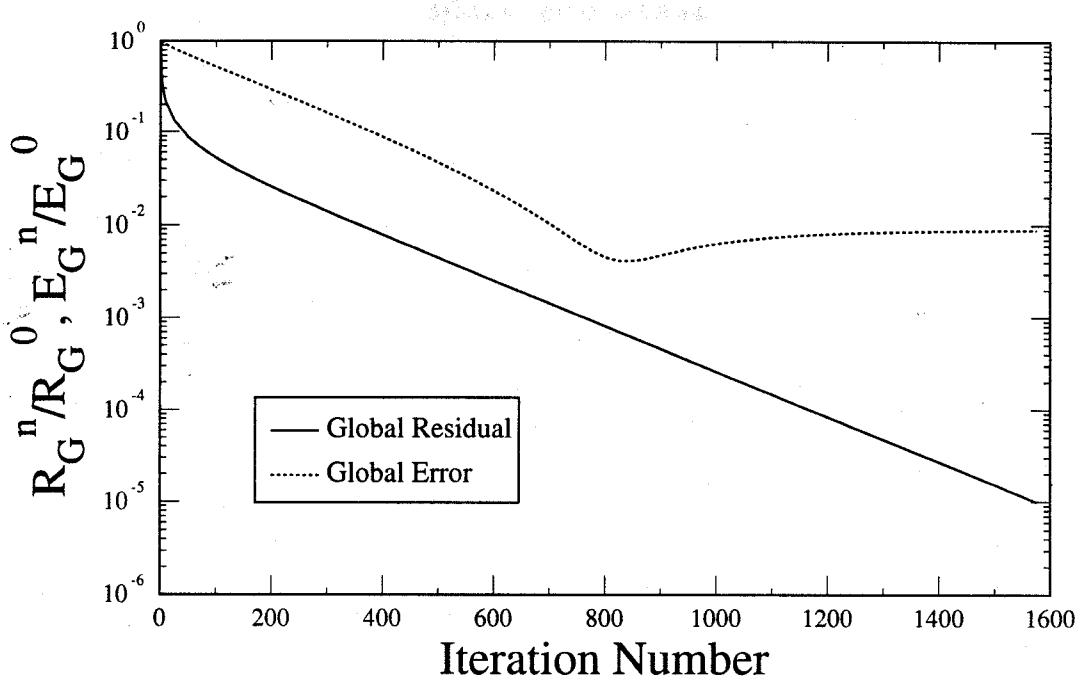

Figure 3.3 Behavior of global residual and error during calculation of potential flow over a circular cylinder using 3280 points

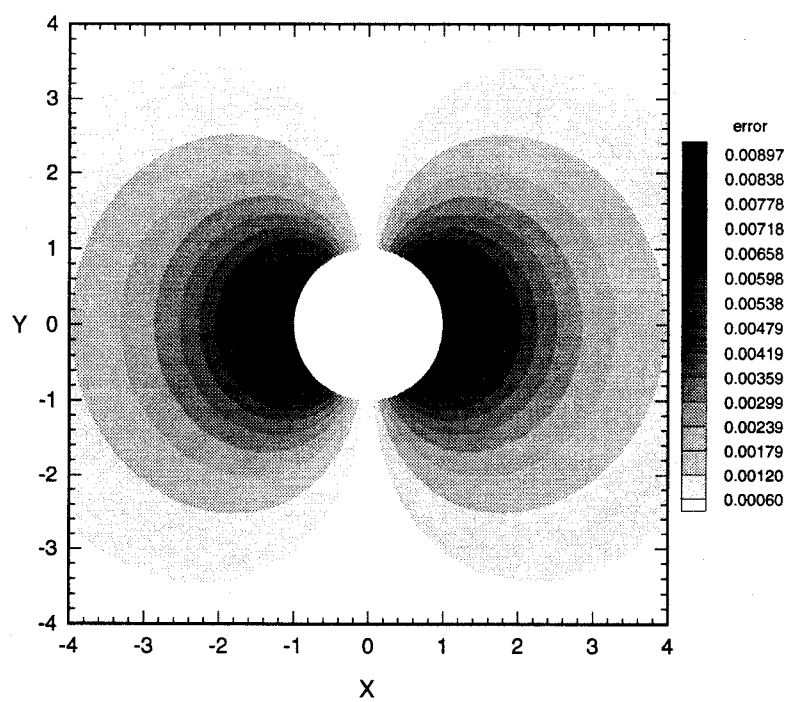

Figure 3.4 Contour plot of the local error of the velocity potential computed with the gridless scheme - 3280 pts

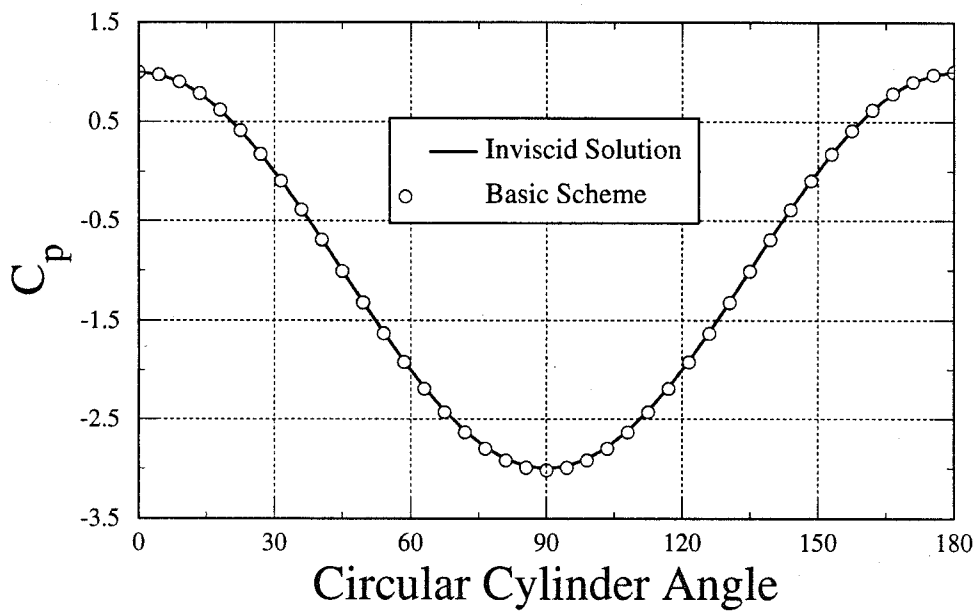

Figure 3.5 Comparison of the exact and computed pressure coefficients for the circular cylinder - basic scheme

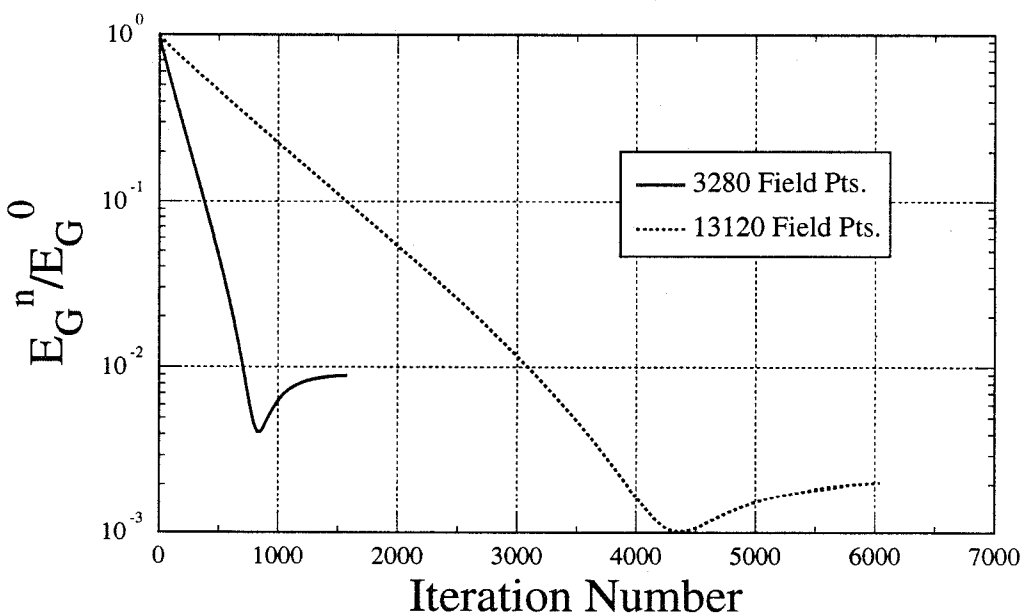

Figure 3.6 Global root-sum-squared error of the gridless numerical method as a function of the number of field points - basic scheme, circular cylinder

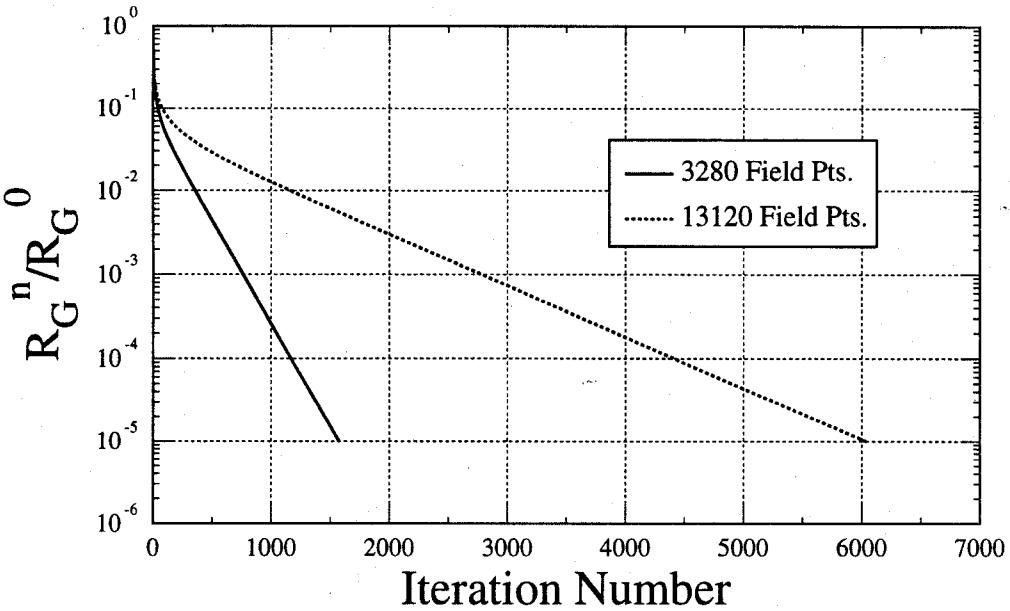

Figure 3.7 Global root-sum-squared residual of the gridless numerical method as a function of the number of field points - basic scheme, circular cylinder

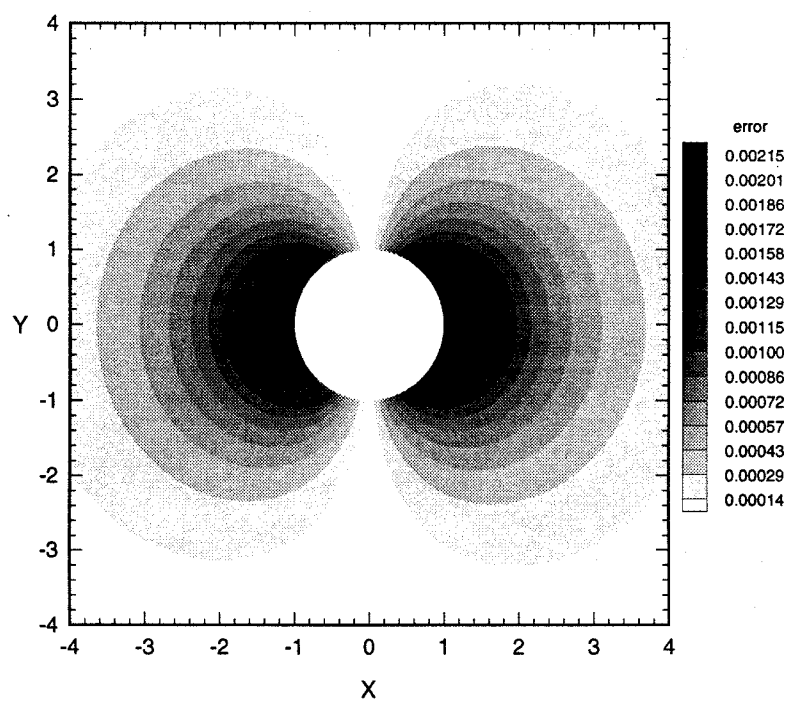

Figure 3.8 Contour plot of the local error of the velocity potential computed with the gridless scheme - 13180 points

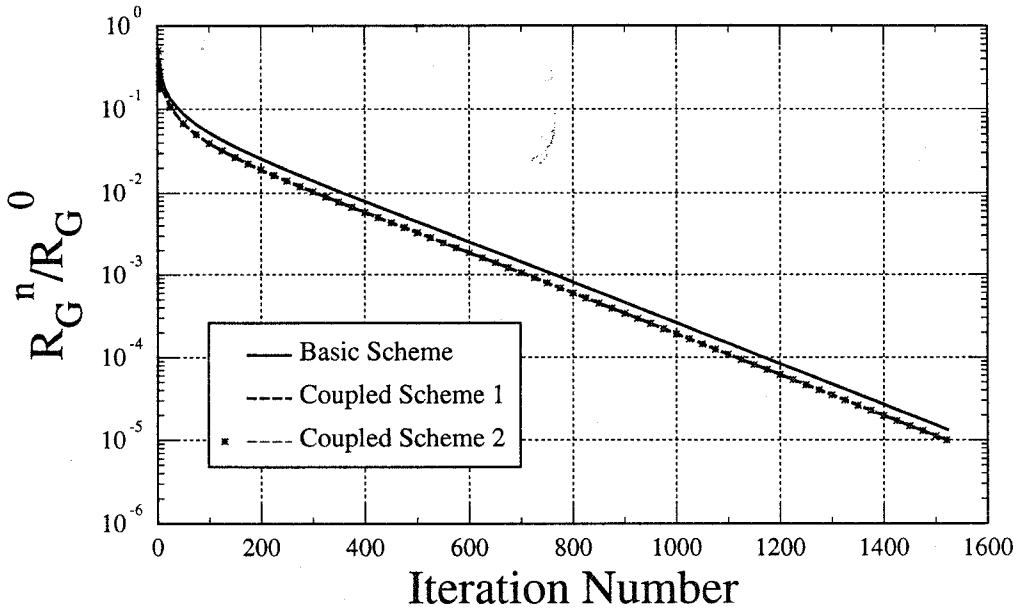

Figure 3.9 Comparison of the global root-sum-square residual for the basic and coupled gridless numerical schemes - circular cylinder

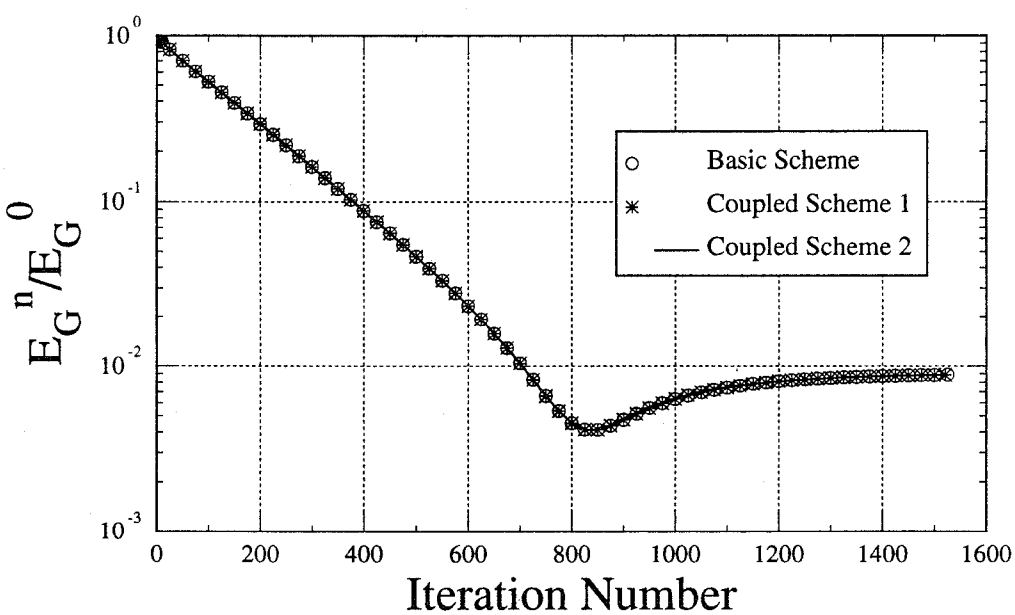

Figure 3.10 Comparison of the global root-sum-square error for the basic and coupled gridless numerical schemes - circular cylinder

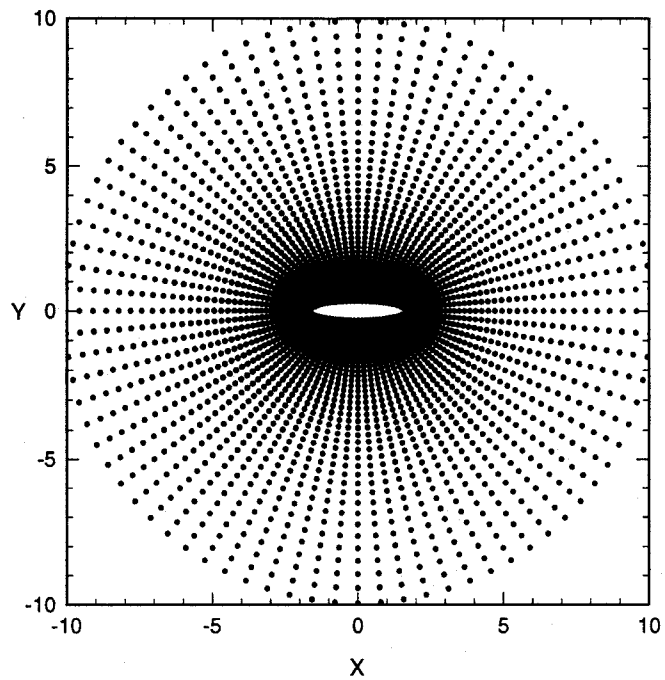

Figure 3.11 Field point distribution for potential flow around a 5:1 aspect ratio elliptical cylinder - 3280 points

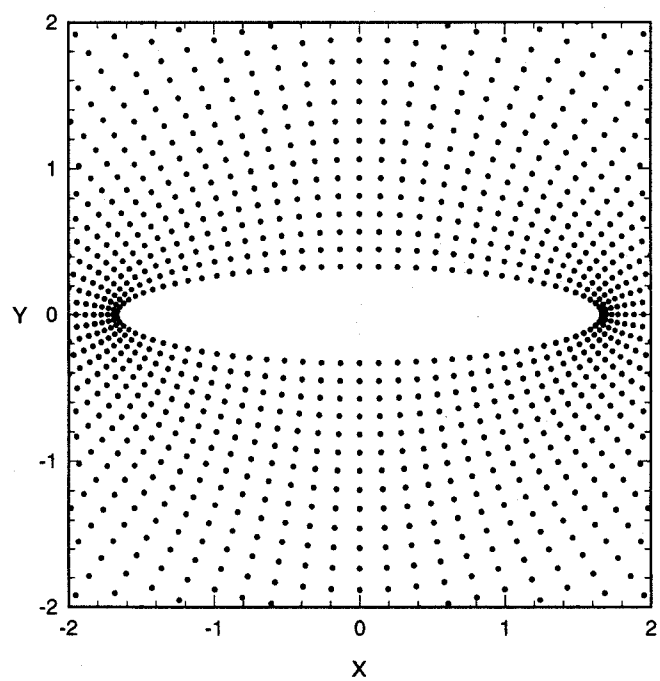

Figure 3.12 Close-up view of the field point distribution in Figure 3.11 for the elliptical cylinder

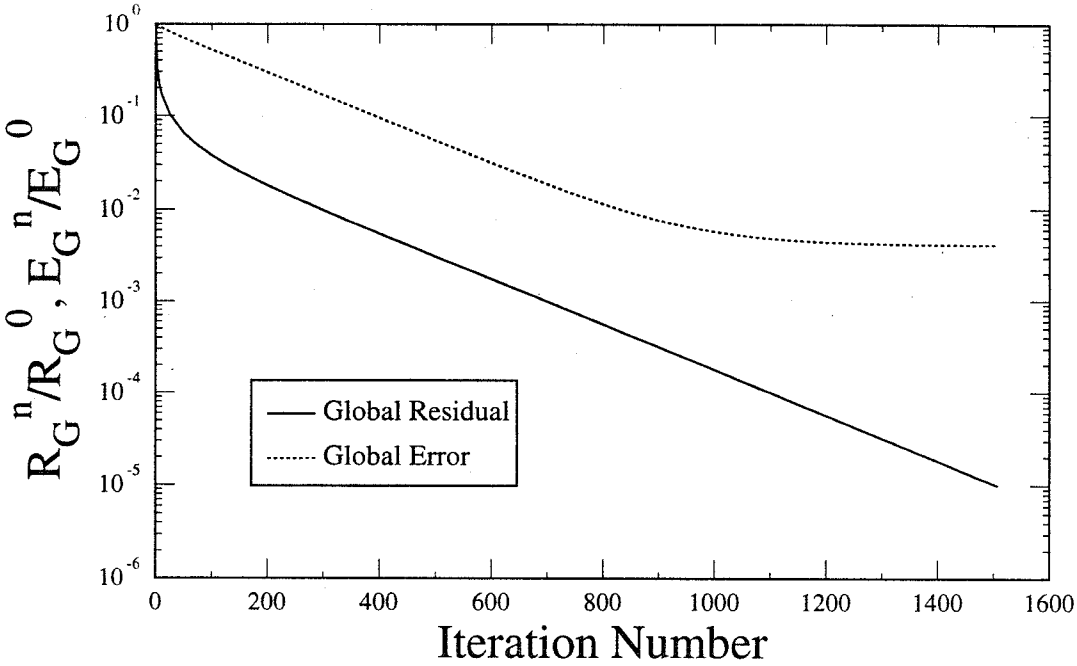

Figure 3.13 Behavior of global residual and error during calculation of potential flow over a 5:1 aspect ratio elliptic cylinder using 3280 points

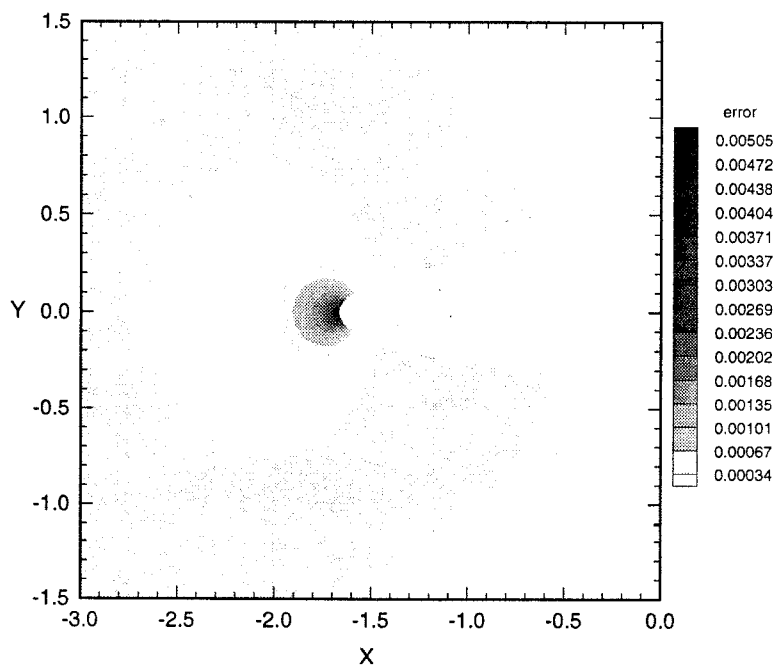

Figure 3.14 Contour plot of the absolute error of the velocity potential computed with the gridless scheme

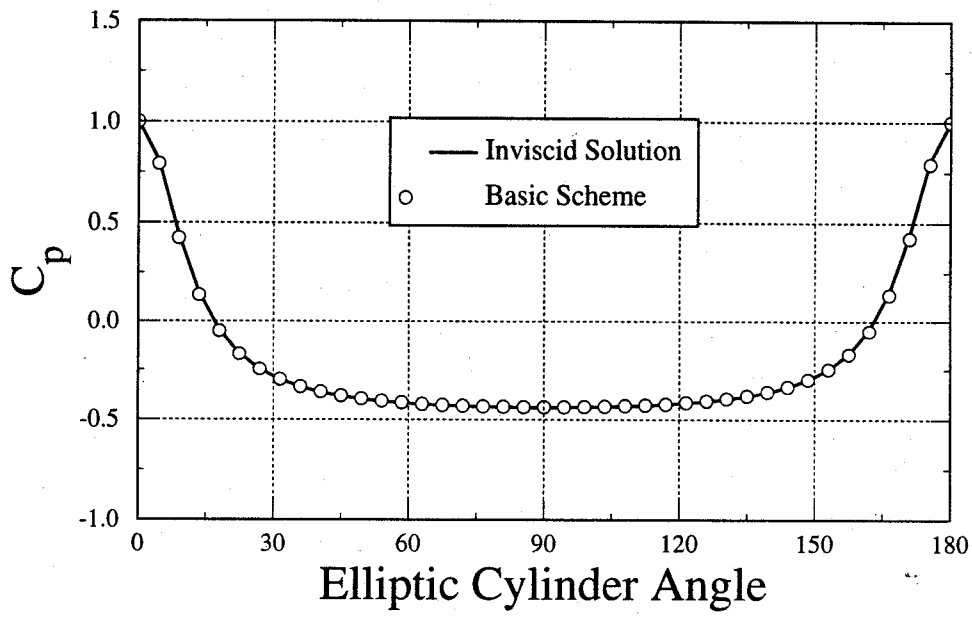

Figure 3.15 Comparison of the exact and computed pressure coefficients for the elliptic cylinder - basic scheme

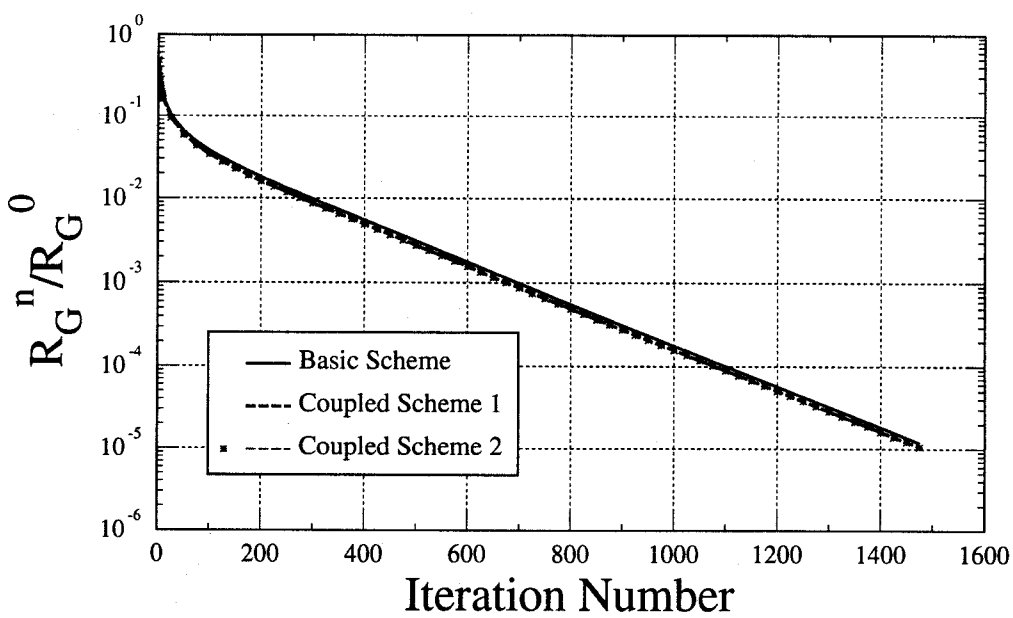

Figure 3.16 Comparison of the global root-sum-square residual for the basic and coupled gridless numerical schemes - elliptic cylinder

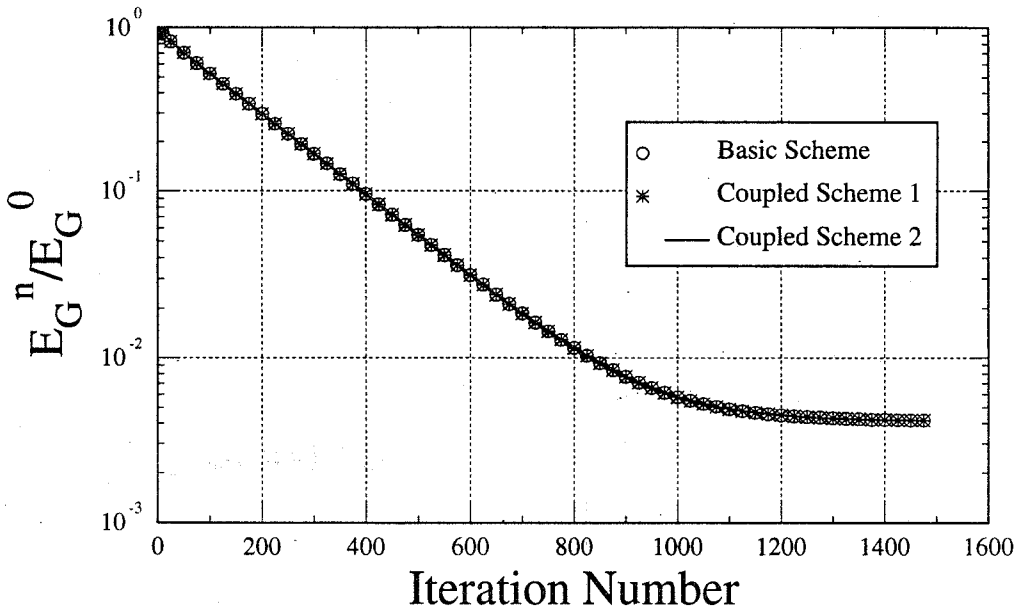

Figure 3.17 Comparison of the global root-sum-squared error for the basic and coupled gridless numerical schemes - elliptic cylinder

4 Higher Order Taylor Series

Christine E. Hailey & N. Todd Snyder

4.1 Overview of Tasks

In order to assist with the development of gridless CFD techniques, we were asked to analyze several aspects of the problem. The ideas for our tasks were based on work previous finite-element researchers performed using h and p techniques for improving the solution.[7,8] In general, h versions of the finite-element method fix the degree of the element and convergence is achieved by mesh refinement. The p version fixes the mesh and convergence is obtained by increasing the degree of the element. The gridless technique also appears to lend itself to a form of h-p adaption. The tasks summarized in this chapter focus on investigating the effect that the order of the Taylor series has upon the solution convergence and accuracy, analogous to the p finite-element techniques. Subsequent tasks, to be reported in phase two of our project, focus on h adaption techniques.

The first task required the development of a generalized recursion relationship for an nth-order, two-dimensional Taylor series. This was then incorporated into a C-language program. The routine requires the user to define the number of terms in the Taylor series. The second task was to assess the impact of additional terms in the local Taylor series approximation on convergence and accuracy of the solution of a two-dimensional equation, such as Laplace's.

4.2 Two-Dimensional Taylor Series

An expression for the Taylor expansion about a point (x_0, y_0) is given by[9]

$$f(x, y) = \sum_{n=0}^{N} \frac{1}{n!}\left(h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right)^n f(x_0, y_0) + R_{N+1} \tag{4.1}$$

where N is the order of the Taylor series and the remainder term, R_{N+1}, is given by

$$R_{N+1} = \frac{1}{(N+1)!}\left(h\frac{\partial}{\partial x} + k\frac{\partial}{\partial y}\right)^{N+1} f(\zeta, \eta) \tag{4.2}$$

The point (ζ, η) is somewhere on the line segment joining the points (x_0, y_0) and (x, y). In this expression h and k are given by $h = x - x_0$ and $k = y - y_0$. For example, a third-order Taylor series expansion is written as:

$$
\begin{aligned}
f(x, y) = {} & f(x_0, y_0) + \{hf_x(x_0, y_0) + kf_y(x_0, y_0)\} \\
& + \frac{1}{2!}\{h^2 f_{xx}(x_0, y_0) + 2hkf_{xy}(x_0, y_0) + k^2 f_{yy}(x_0, y_0)\} \\
& + \frac{1}{3!}\{h^3 f_{xxx}(x_0, y_0) + 3h^2 kf_{xxy}(x_0, y_0) + 3hk^2 f_{xyy}(x_0, y_0) + k^3 f_{yyy}(x_0, y_0)\} \\
& + R_4
\end{aligned}
\tag{4.3}
$$

where the subscripts x and y denote partial derivatives in the usual notation. In order to write a recursive relation for the two-variable Taylor series which could readily be implemented in a computer program, we let $a = h\frac{\partial}{\partial x}$ and $b = k\frac{\partial}{\partial y}$. Then the Taylor series can be written in terms of the binomial theorem

$$
f(x, y) = \sum_{n=0}^{N} \frac{1}{n!}(a+b)^n f(x_0, y_0) + R_{N+1} \tag{4.4}
$$

where $(a+b)^n$ is given by

$$
(a+b)^n = a^n + \binom{n}{1} a^{n-1} b + \binom{n}{2} a^{n-2} b^2 + \binom{n}{3} a^{n-3} b^3 + \ldots + b^n \tag{4.5}
$$

The binomial coefficient $\binom{n}{m}$ is given by

$$
\binom{n}{m} = \frac{n!}{(n-m)!m!} \tag{4.6}
$$

Both the binomial coefficient and the factorial term can be easily calculated using canned computational routines, such as those found in *Numerical Recipes in C*.[10]

A C-language program was written in order to check the higher-order Taylor series algorithm. A function whose derivatives could be determined analytically, given by

$$
f(x, y) = x^4 y^4 \tag{4.7}
$$

was used to test the C-code. A surface plot of the function is presented in Figure 4.1. The function is well-behaved in the domain of interest, [0,1]×[0,1]. The derivatives for these two functions were evaluated at two points: (0.5, 0.5) and (0.9, 0.8). The first point is located in a region with a fairly flat gradient and curvature, and the second point is located in a region with a steep gradient.

The numerical evaluation of the derivatives depended on two different parameters. The first parameter of interest was the number of points necessary to form a "cloud" surrounding the evaluation point. Clearly, in order to obtain higher-order derivatives, more points were needed.

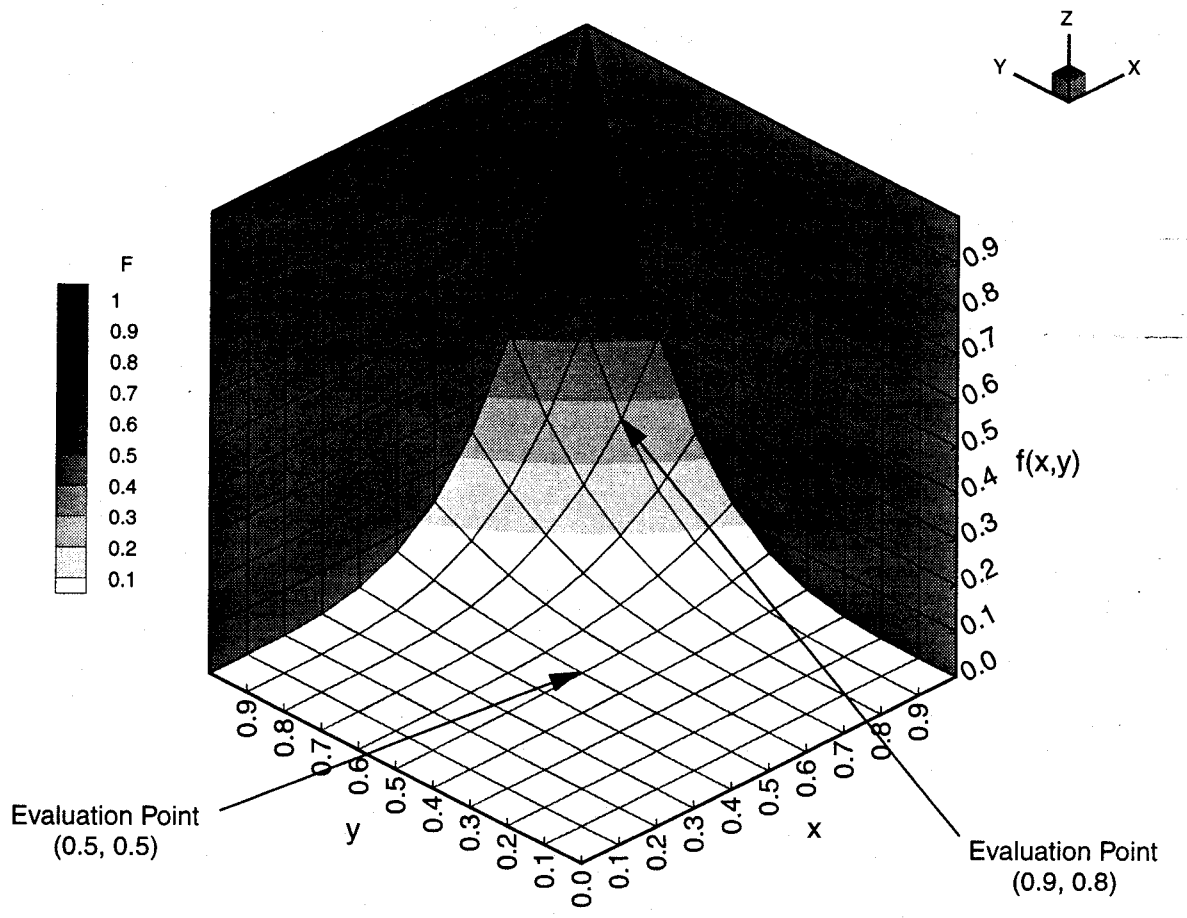

Figure 4.1 Surface Plot of $f(x, y) = x^4 y^4$

However, we were also interested in the effect the number of points in the cloud might have on the solution. For example, a third-order solution requires a minimum of 9 points, however it is possible to form a cloud with more than 9 points. The second parameter of interest was the cloud geometry. Circular, rectangular, and irregular clouds were formed.

In the first investigation, all points were equally spaced on a circle or circles, centered at (0.5, 0.5) or (0.9, 0.8). For example, for a cloud of 20 points, 10 points were equally spaced on a circle of radius 0.01 and another 10 points were equally spaced on a circle of radius 0.02 (see Figure 4.2 for an illustration).

The results of the first investigation for the function are shown in Table 4.1 for the evaluation point (0.5, 0.5) and in Table 4.4 for the evaluation point (0.9, 0.8). Second, third, fourth, and fifth-order terms in the Taylor series were examined. Clouds contained either 5, 9, 14, or 20 points. The cloud geometry consisted of a single circle centered about the evaluation point with radius of 0.01 (denoted R = 0.01 in the tables) or families of circles centered about the evaluation point with increasing radii, as indicated in the tables.

We offer several comments about the results. We observe that the selection of cloud geometry greatly influences the solution, especially for the evaluation point (0.9, 0.8) which is

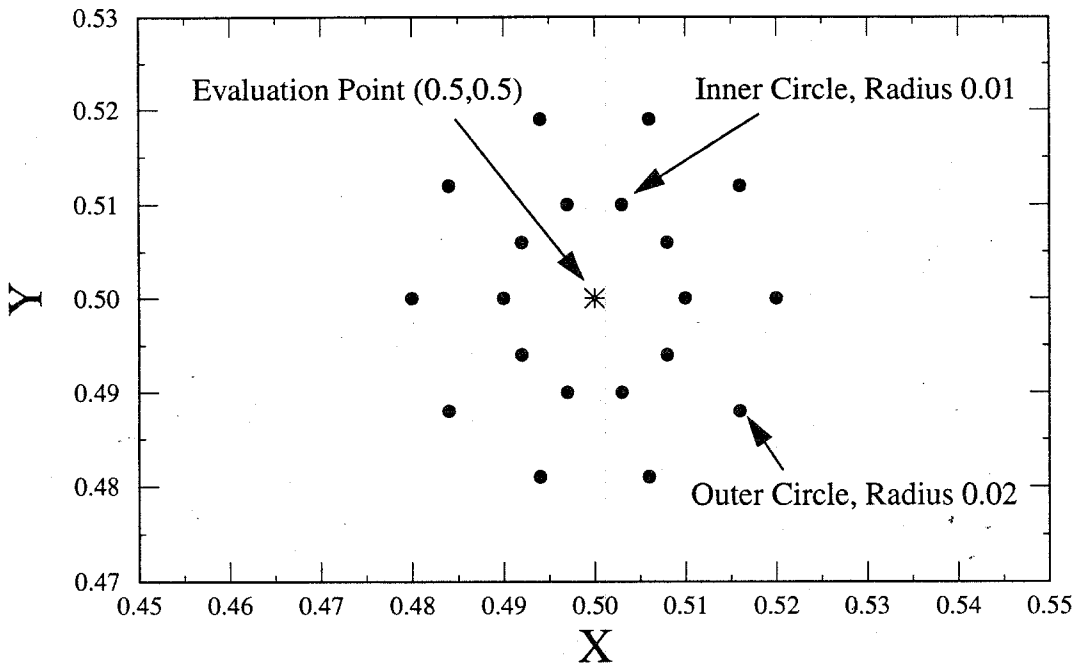

Figure 4.2 Example of Circular Cloud Geometry

located in a region of very steep gradient. If all points are selected to lie on the circumference of a single circle, the results for the higher-order derivatives are poor. The fifth-order solution gave best results for points selected to lie on either two or three circles. Spreading the points out over too many circles did not give as good of results for the fifth-order solution. Clearly, there is an optimum cloud geometry which is dependent on the function and evaluation point. In general, the fifth-order results are poor for a fifth-order Taylor series, the fourth-order results are poor for a fourth-order Taylor series, and so on. We noted that one must select a Taylor series order that is at least one order higher, preferably two orders higher, than the order of the derivative of interest. Since most governing equations of interest contain only first and second order derivative, it appears one could select a fourth-order Taylor series and obtain good estimates for the first and second derivatives.

The next investigation was to form several "clouds" which better resemble those that might be formed using a rectangular-based mesh. Again, the evaluation points were taken as (0.5, 0.5) and (0.9, 0.8). A rectangle is formed which surrounds the evaluation points. The midpoints of the sides of the rectangles are located a distance of 0.01 from the evaluation point. The vertices were located $0.01\sqrt{2}$ units from the evaluation point. Thus, eight points are used to evaluate the Taylor series which allows a second-order solution. A second rectangle can be formed which surrounds the first one. Each side of this larger rectangle contains five points spaced at a distance of 0.02. With the inner and outer rectangle surrounding the evaluation point, 24 points are available to evaluate the Taylor series which allows up to a fifth order solution. An example of the rectangular cloud geometry for the evaluation point (0.5, 0.5) is shown in Figure 4.3. The results of these calculations are shown in Table 4.2 and in Table 4.4.

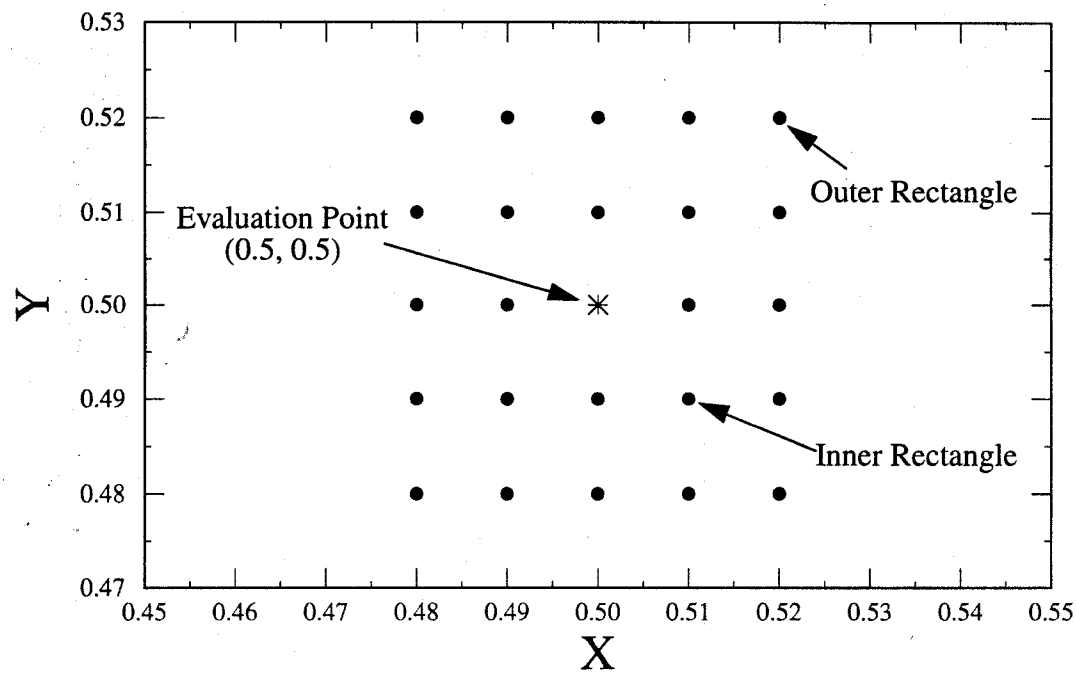

Figure 4.3 Example of Rectangular Cloud Geometry

By selecting a rectangular clustering of points, we ensured that not all points lie on a circle with a fixed radius. Hence, all results give fairly good approximation to the lower-order terms in the Taylor series. In fact, it is interesting that the 8-point rectangle which provides only a 2nd-order approximation to the Taylor series, gives reasonable results. Clearly, if one is interested in only first and second derivatives, a fourth-order solution using a double rectangle should give very good results.

Finally, we considered a more pathological case of cloud point distribution. In particular, suppose grid adaption or mesh refinement/coarsening has occurred and the points have clustered to one side of the evaluation point. Again, two evaluation points were used: (0.5, 0.5) and (0.9, 0.8). In this case, seventeen neighboring points were selected to evaluate the derivatives and are shown in Figure 4.4. The results of these calculations are shown in Tables 4.3 and 4.4.

In order to summarize the results presented in Tables 4.1 through 4.4, we calculated the absolute percent error in the f_{xx} term. We selected this term because it is second-order, typical of a term found in Laplace's equation. The results are shown in Figures 4.5 and 4.6. On the y-axis, we have plotted the normalized error, which is give by

$$\text{normalized error} = \left| \frac{\text{Calculated Value} - \text{Exact Value}}{\text{Exact Value}} \right| \tag{4.8}$$

We offer some observations based on the above results. First, a fourth-order Taylor series approximation for f_{xx} gives better results than the third-order Taylor series. And the third-order

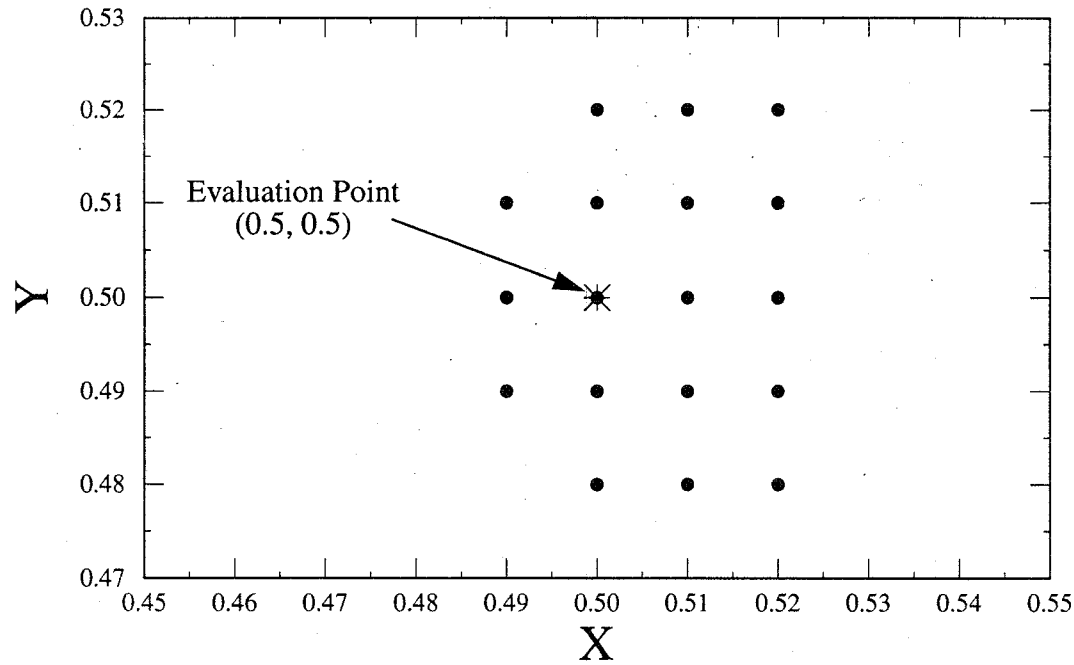

Figure 4.4 Plot of Cloud Geometry for Irregular Geometry

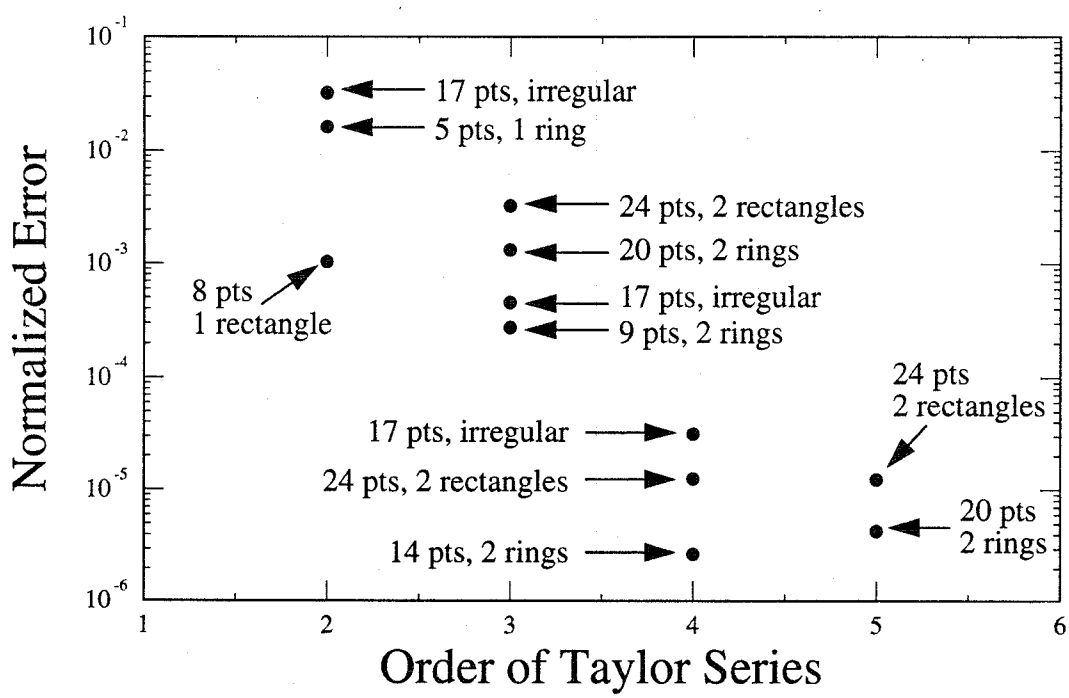

Figure 4.5 Normalized Error As a Function of Order and Cloud Geometry for Evaluation Point (0.5, 0.5)

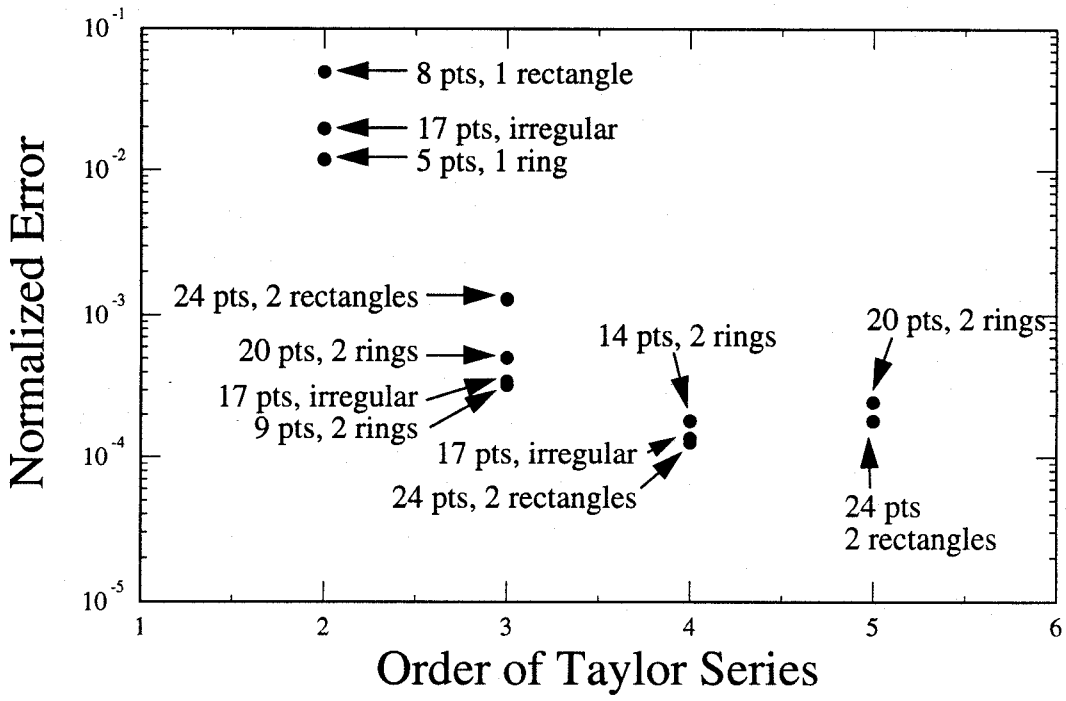

Figure 4.6 Normalized Error As a Function of Order and Cloud Geometry for Evaluation Point (0.9, 0.8)

Taylor series gives better results than the second-order Taylor series. These results are very consistent with the results found by finite-element researchers for p-adaption. Higher-order terms can be used to improve solution accuracy. However, these results suggest there is an optimum higher order. Clearly the fifth-order results are no better than the fourth-order results. The fifth-order Taylor series approximation may have provided too many degrees of freedom for estimation of the function. We also note that we would be hard pressed to answer the question: "What cloud geometry gives the best results?" The cloud geometry and number of points in the cloud affect the solution, but these results do not indicate which combination is best or worst. The irregular cloud geometry gave reasonable results, which is an important result as we proceed in looking at h-adaption where the points will be moved/added/deleted and irregular clouds may be formed. Finally, the function had a less steep gradient at the (0.5, 0.5) evaluation point and the numerical simulation did a better job of predicting the value of f_{xx}. As the function steepened, the scheme did not predict the f_{xx} term as well.

4.3 Higher-Order Solutions of Laplace's Equation

The 2-D Laplace's code described in Chapters 2 and 3 had to be modified in two places in order to permit higher-order terms: in the formation of cloud geometry and in the formation of the [A] matrix used to evaluate derivatives (see equation (3.2) on page 26). First we noted that in order to include third-order Taylor series terms, at least 9 points must be used to form a cloud, similarly for fourth-order Taylor series at least 14 points must be used, and for fifth-order Taylor series at least 20 points must be used to form the cloud. Since we were interested in comparing

higher-order results with the second-order results described in Section 3.3, we utilized the same point distribution, which results in cloud geometries similar to the sketch shown in Figure 4.4.

In order to develop a robust code which would allow a user-defined number of points for a cloud, we began with the existing cloud formation algorithm. The best way to summarize the this algorithm is to imagine the evaluation point as the origin of a local x-y coordinate system. The existing algorithm selects eight points which are fairly uniformly distributed in all four quadrants of this x-y coordinate system. This is accomplished by calculating the distance all points are from the evaluation point, sorting the distances and then selecting the point which is closest to the evaluation point. The closest point is the first member of the cloud. The point which is second closest to the evaluation point will be allowed in the cloud formation provided it is not located within a pie-shaped wedge of 45° (±22.5°) which emanates from the origin and contains the first-closest point. Similarly, the third point in the cloud formation will be third closest to the evaluation point provided it does not lie in the pie-shaped wedges which surround the previous two points. Should the third point lie within a pie-shaped wedge, it is rejected and the next closest point is selected as a candidate. This algorithm effectively finds eight cloud members which are fairly uniformly distributed in all four quadrants which surround the evaluation point.

Since we were interested in higher-order terms, we always needed to form clouds with more than eight points. We used the existing algorithm to select the first eight points. We then return to the list of sorted distances and select the next closest to the evaluation point which has not already been used in the cloud geometry. We form a 22.5° (±11.25°) pie-shaped wedge centered around this point, return to the sorted list of distances and select the next closest value. If this value is not within the 22.5° pie region, it becomes a member of the cloud geometry, we form another 22.5° pie region surrounding this point and then we select the next closest point which does not lie in the previous two points' pie-shaped wedges. This process is repeated until the required number of points have been selected. Again, we are trying to ensure the cloud contains points which are more uniformly distributed in all four quadrants surrounding the evaluation point.

The second modification to the 2-D Laplacian code was to calculate an appropriate [A] matrix which would contain the higher-order terms. We were able to use the C-function used to generate the results in Section 4.2. The user has to specify the order of the Taylor series, the number of points in an interior point cloud, the number points in a wall cloud, and the number of derivatives.

Results of applying higher-order terms to the solution of Laplace's equation are described below. We utilized the grid shown in Figure 3.11, which contained 3280 points, and solved Laplace's equation for flow over a 5:1 ellipsoid. The boundary conditions were the usual for potential flow, no flow normal to the boundary. First, we ran the existing 2-D code which was second-order using 8 points in a cloud. Then we ran a number of cases for higher order solutions.

The results are disappointing. They indicate that by using a higher-order Taylor series everywhere in the solution domain, the solution accuracy is diminished. Shown in Figure 4.7 are several of the third- and fourth-order solutions we calculated compared with the original second-order solution which utilized an eight point cloud. We have plotted the results in terms

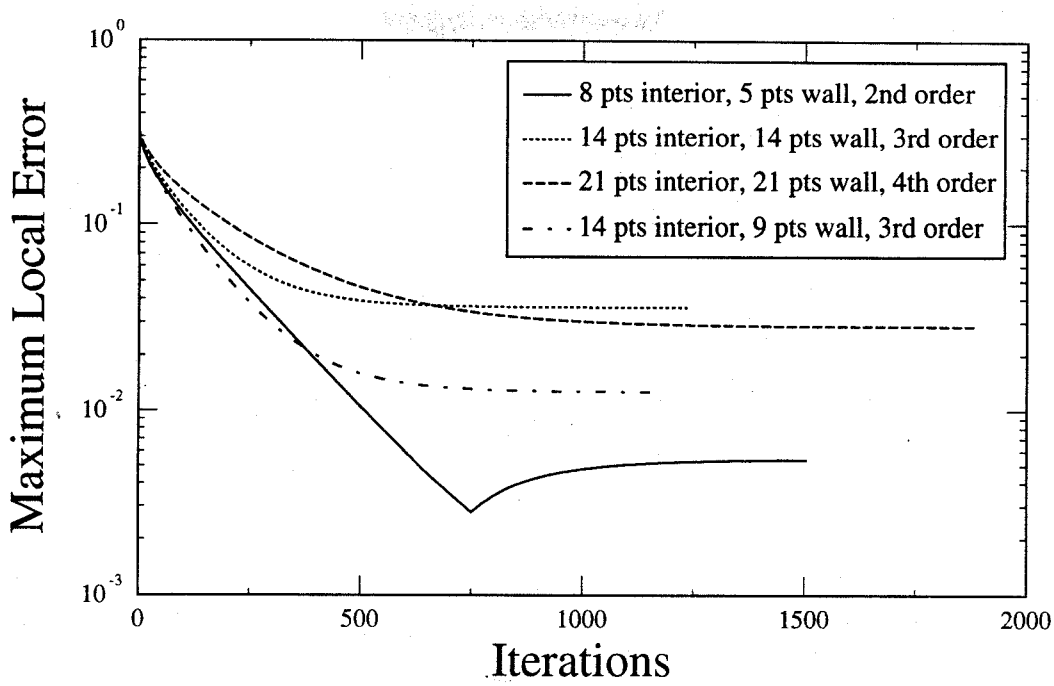

Figure 4.7 Several Higher-Order Calculations of Laplace's Equation Compared with the Second-Order Solution

of number of iteration until the global residual reached a value smaller than 10^{-5}. After each iteration, the local error at each point is calculated by subtracting the calculated solution from the exact solution. Shown in Figure 4.7 is the largest local error value in the domain. The second-order Taylor series calculation using eight nodes gives the best results, although the error reaches a minimum and then begins to grow before the convergence criterion is reached.

We observed that the largest local error in all cases occurred on the wall. So we tried one additional case. Without too much effort, we were able to modify the 2-D Laplacian code to allow one order of solution on the wall and another order of solution in the interior. This gave better results, as shown in Figure 4.8. We were able to continue to reduce the error as the residual was reduced when we selected third-order on the wall and second-order in the interior. Finally, it is important to note that a number of cases would not converge. For example, the case of 19 points on the interior, 16 points on the wall converged, however, for cases with the same order of Taylor series and less than 19 points on the interior, the solution did not converge.

These results suggest that we would probably have reduced error for higher-order terms if we would select a region "close" to the wall and incorporate a higher-order solution. This is not surprising. The flow field is changing rapidly near the wall and requires some form of adaption. In the far field, the flow is essentially freestream which has no higher order behavior. By trying to insist on higher-order functions for freestream behavior we are giving the solution too many degrees of freedom.

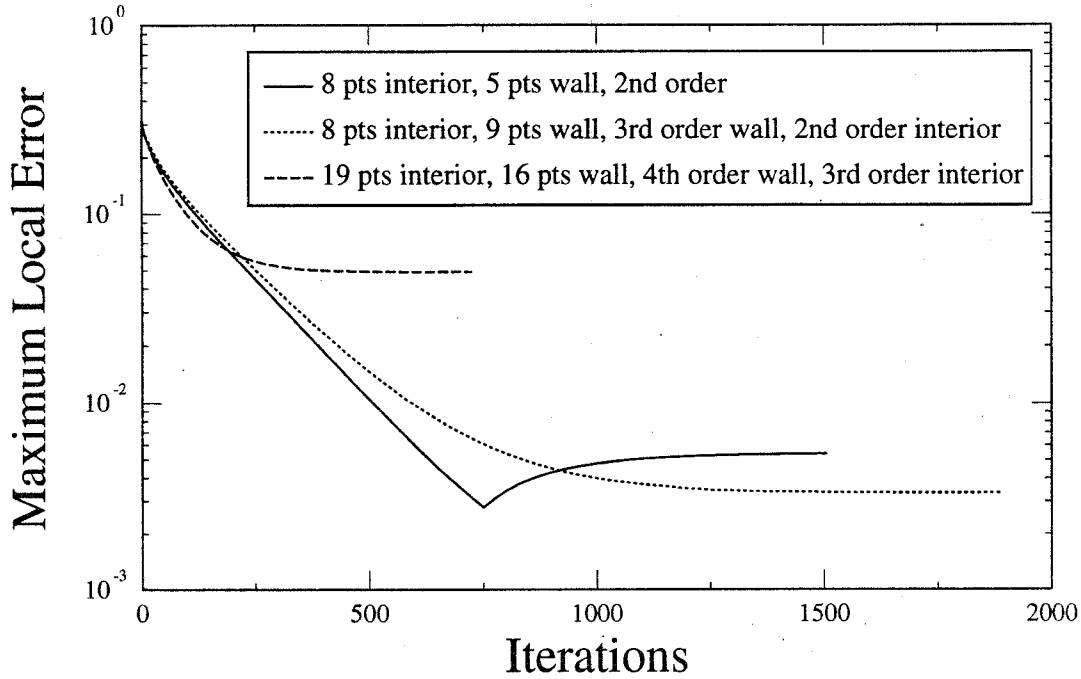

Figure 4.8 Results of Predictions with Highest Order on Wall, Lowest Order in Interior

4.4 Summary and Recommendations for Future Work

We developed a generalized recursive relationship for an nth-order, two-dimensional Taylor series which was incorporated into a C-program where the user is allowed to define the number of terms in the Taylor series. We used this code to evaluate behavior of higher-order terms on solution accuracy. We varied both the number of points in a particular cloud and also the cloud geometry. In general, we found the most accurate results were obtained if the user selected terms in the Taylor series which are two orders higher than the highest derivative to be evaluated. However, the accuracy is a strong function of cloud geometry and number of points in a cloud.

We then incorporated the higher-order Taylor series code into a Sandia developed code which solved the two-dimensional Laplace's equation. We evaluated the effects of the addition of terms in the Taylor series approximation on the convergence and accuracy of solutions. We found that simply fixing a higher-order Taylor series everywhere in the flow field did not give better accuracy because in the far field the higher-order terms provide too many degrees of freedom for a flow field that is essentially freestream. Since the largest error occurred near the body, we allowed the solution to be higher order on the wall and lower order in the interior. These results provided close to the same accuracy as the simple second-order Taylor series results and better convergence properties. The error continued to diminish as the residual diminished.

Future efforts will be focused on incorporating both h and p adaption into the two-dimensional Laplace's equation solver. The results found to date suggest that if we could select a region of points sufficiently close to the wall and apply a higher-order Taylor series, we would probably obtain better accuracy than the second order results. This will require a major rewrite of the two-dimensional code. We also plan to investigate the effects of h adaption on the solution accuracy.

Table 4.1 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Circular Geometry, Evaluated at $x = 0.5$, $y = 0.5$

Order of Taylor Series	Exact Solution	5th order	5th order	5th order	4th order	4th order	2nd order
Points in Cloud		20 points	20 points	20 points	20 points	14 points	5 points
Cloud Configuration		R = 0.01 10 pts R = 0.02 10 pts	R = 0.01 7 pts R = 0.02 7 pts R = 0.03 6 pts	R = 0.01 5 pts R = 0.02 5 pts R = 0.03 5 pts R = 0.04 5 pts	R = 0.01 7 pts R = 0.02 7 pts R = 0.03 6 pts	R = 0.01 7 pts R = 0.02 7 pts	R = 0.01 5 pts
f_x	0.03125	-3.00E-08	-6.000E-08	2.000E-08	1.000E-08	-2.000E-08	2.78E-05
f_y	0.03125	-1.00E-08	-1.900E-07	8.000E-08	1.50E-07	-5.000E-08	2.81-E-05
f_{xx}	0.1875	-8.000E-07	4.800E-06	-2.400E-06	1.56E-05	-5.000E-07	3.061E-03
f_{xy}	0.25	1.200E-06	-7.000E-06	-1.410E-05	-2.56E-05	-6.800E-06	3.030E-03
f_{yy}	0.1875	-2.000E-07	-4.500E-06	1.290E-05	-1.380E-05	3.7000E-06	-3.187E-03
f_{xxx}	0.75	5.114E-04	4.123E-04	9.1000E-05	-3.899E-03	1.945E-04	
f_{xxy}	1.5	1.081E-03	3.268E-03	-5.808E-03	-1.338E-02	2.7500E-03	
f_{xyy}	1.5	1.019E-03	2.109E-03	-4.435E-03	1.491E-03	1.251E-03	
f_{yyy}	0.75	3.768E-04	2.699E-03	1.515E-04	-3.312E-03	-3.411E-04	
f_{xxxx}	1.5	3.600E-02	-5.002E-02	4.385E-03	2.355E-01	3.605E-03	
f_{xxxy}	6.0	-2.797E-02	3.466E-01	-1.106E+00	-2.877E+00	4.189E-01	
f_{xxyy}	9.0	-4.356E-03	-1.638E-01	-7.891E-01	-5.404E-01	-5.99eE-02	
f_{xyyy}	6.0	-6.330E-04	-4.303E-02	5.149E-01	1.6717E+00	-1.204E-01	
f_{yyyy}	1.5	-1.186E-02	3.815E-01	1.094E+00	5.594E-01	1.325E-03	
f_{xxxxx}	0.0	1.493E-05	7.055E-03	-4.03E-02			
f_{xxxxy}	12.0	-1.200E+01	-1.201E+01	-1.208E+01			
f_{xxxyy}	36.0	-3.600E+01	-3.602E+01	-3.61E+01			
f_{xxyyy}	36.0	-3.600E+01	-3.599E+01	-3.60E+01			
f_{xyyyy}	12.0	-1.200E+01	-1.202E+01	-1.200E+01			
f_{yyyyy}	0.0	1.489E-05	-9.794E-04	4.563E-02			

Table 4.1 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Circular Geometry, Evaluated at $x = 0.5$, $y = 0.5$

		Data Shown Above are (Calculated - Exact) Values					
Order of Taylor Series	Exact Solution	3rd order	3rd order	3rd order	3rd order	3rd order	3rd order
Points in Cloud		20 points	20 points	20 points	9 points	20 points	9 points
Cloud Configuration		R = 0.01 10 pts R = 0.02 10 pts	R = 0.01 7 pts R = 0.02 7 pts R = 0.03 6 pts	R = 0.01 5 pts R = 0.02 5 pts R = 0.03 5 pts R = 0.04 5 pts	R = 0.01 5 pts R = 0.02 4 pts	R = 0.01 20 pts	R = 0.01 9 pts
f_x	0.03125	-3.00E-08	-6.00E-08	2.91E-06	-3.60E-07	1.870E-05	1.870E-05
f_y	0.03125	-1.00E-08	-1.90E-07	-3.40E-07	2.45E-05	1.870E-05	1.87-E-05
f_{xx}	0.1875	2.511E-04	1.402E-04	9.274E-04	5.14E-05	6.520E-05	6.280E-05
f_{xy}	0.25	3.827E-04	7.996E-04	-9356E-04	-3.571E-03	1.010E-04	1.000E-04
f_{yy}	0.1875	2.502E-04	9.081E-04	8.548E-04	5.14E-05	6.520E-05	6.640E-05
f_{xxx}	0.75	5.114E-04	-1.111E-03	-7.517E-02	5.179E-03	-1.125E+00	-1.125E+00
f_{xxy}	1.5	1.081E-03	2.303E-03	-1.852E-01	-1.5889E+00	-3.742E-01	-3.736E-01
f_{xyy}	1.5	1.109E-03	3.712E-03	-1.791E-02	1.286E-03	-3.742E-01	-3.752E-01
f_{yyy}	0.75	3.767E-04	3.667E-03	1.923E-01	-3.675E-01	-1.125E+00	-1.126E+00
		Data Shown Above are (Calculated - Exact) Values					

Table 4.2 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Rectangular Geometry, Evaluated at $x = 0.5$, $y = 0.5$

Order of Taylor Series	Exact Solution	5th order	4th order	3rd order	2nd order	2nd order
Points in Cloud		24 points	24 points	24 points	24 points	8 points
Cloud Configuration		Double Rectangles, 8-point and 16-point	Double Rectangles, 8-point and 16-point	Double Rectangles, 8-point and 16-point	Double Rectangles, 8-point and 16-point	Single, 8-point Rectangle
f_x	0.03125	3.0E-07	2.0E-07	2.0E-07	-1.927E-04	-6.25E-05
f_y	0.03125	3.0E-07	2.0E-07	2.0E-07	-1.927E-04	-6.25E-05
f_{xx}	0.1875	2.3E-06	-2.3E-06	-6.13E-04	-6.131E-04	1.945E-04
f_{xy}	0.25	3.0E-06	2.7E-06	6.8E-04	6.8E-04	-1.997E-04
f_{yy}	0.1875	2.3E-06	-2.3E-06	-6.13E-04	-6.131E-04	1.945E-04
f_{xxx}	0.75	-3.15E-03	3.51E-03	-3.517E-03		
f_{xxy}	1.5	-2.47E-03	2.475E-03	2.475E-03		
f_{xyy}	1.5	-2.47E-03	2.475E-03	2.475E-03		
f_{yyy}	0.75	-3.15E-03	3.51E-03	-3.517E-03		
f_{xxxx}	1.5	6.09E-02	6.1E-02			
f_{xxxy}	6.0	2.59E-02	2.58E-02			
f_{xxyy}	9.0	2.01E-03	2.01E-03			
f_{xyyy}	6.0	2.59E-02	2.58E-02			
f_{yyyy}	1.5	6.09E-02	6.1E-02			
f_{xxxxx}	0.0	1.88E-05				
f_{xxxxy}	12.0	12.00				
f_{xxxyy}	36.0	36.000				
f_{xxyyy}	36.0	36.00				
f_{xyyyy}	12.0	12.00				
f_{yyyyy}	0.0	1.88E-05				
		Data Shown Above are (Calculated - Exact) Values				

Table 4.3 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Irregular Geometry, Evaluated at $x = 0.5$, $y = 0.5$

Order of Taylor Series	Exact Solution	4th order	3rd order	2nd order
Points in Cloud		17 points	17 points	17 points
Cloud Configuration		Concentrated on the Right-Hand Side	Concentrated on the Right-Hand Side	Concentrated on the Right-Hand Side
f_x	0.03125	-2E-07	5.5E-06	9.45E-05
f_y	0.03125	3E-08	3.9E-06	5.76E-05
f_{xx}	0.1875	5.9E-06	8.5E-05	6.1E-03
f_{xy}	0.25	1.47E-05	1.89E-04	1.267E-02
f_{yy}	0.1875	-2.8E-06	1.51E-05	9.23E-03
f_{xxx}	0.75	8.46E-03	9.08E-02	
f_{xxy}	1.5	-3.41E-04	5.807E-02	
f_{xyy}	1.5	-1.48E-04	8.468E-02	
f_{yyy}	0.75	-7.72E-04	6.469E-02	
f_{xxxx}	1.5	-1.496E00		
f_{xxxy}	6.0	3.174E-01		
f_{xxyy}	9.0	2.963E-01		
f_{xyyy}	6.0	3.877E-01		
f_{yyyy}	1.5	7.23E-02		
		Data Shown Above are (Calculated - Exact) Values		

Table 4.4 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Evaluated at $x = 0.9$, $y = 0.8$

Order of Taylor Series	Exact Solution	5th order	5th order	4th order	4th order	4th order	3rd order
Points in Cloud		24 points	20 points	24 points	17 points	14 points	24 points
Cloud Configuration		Two Rectangles	Two Rings R = 0.01 10 pts R = 0.02 10 pts	Two Rectangles	Irregular	Two Rings R = 0.01 7 pts R = 0.02 7 pts	Two Rectangles
f_x	1.194396	-2.0E-06	4.0E-06	-2.0E-06	1.0E-06	-5.0E-06	-2.0E-06
f_y	1.3436928	2.0E-07	2.2E-06	2.0E-07	2.0E-07	3.36E-04	2.0E-07
f_{xx}	3.981312	7.22E-04	9.81E-04	7.22E-04	5.43E-04	5.05E-04	5.12E-03
f_{xy}	5.971968	-2.49E-04	7.1E-05	-2.49E-04	-3.45E-04	-4.92E-02	5.676E-03
f_{yy}	5.038848	5.47E-04	4.02E-04	5.47E-04	4.69E-04	6.2E-04	5.289E-03
f_{xxx}	8.84736	1.277E-02	2.743E-02	1.277E-02	-7.262E-02	-4.31E-02	1.277E-02
f_{xxy}	19.90656	2.407E-02	-9.2E-03	2.407E-02	3.106E-02	-21.6	2.407E-02
f_{xyy}	22.39488	9.98E-03	1.059E-02	9.98E-03	-3.064E-02	-1.3E-01	9.98E-03
f_{yyy}	12.59712	-1.297E-02	-3.995E-02	-1.297E-02	-2.284E-02	-5.0558	-1.298E-02
f_{xxxx}	9.8304	-19.782	-26.38	-19.782	-9.786	-47.76	
f_{xxxy}	44.2368	2.060	-1.84	2.060	14.354	75.77	
f_{xxyy}	74.6496	0.4599	-8.274	0.4599	4.878	3.785	
f_{xyyy}	55.9872	2.2604	-0.736	2.2604	3.4469	-41.93	
f_{yyyy}	15.7464	-16.295	2.795	-16.295	-13.801	-44.26	
f_{xxxxx}	0.0	2.213E-04	1.71E-04				
f_{xxxxy}	49.152	-49	-49				
f_{xxxyy}	165.888	-165	-165				
f_{xxyyy}	186.624	-186	-186				
f_{xyyyy}	69.984	-69	-69				
f_{yyyyy}	0.0	3.14E-04	2.5E-04				
		Data Shown Above are (Calculated - Exact) Values					

Table 4.4 Higher-Order Taylor Series Calculations for $f(x, y) = x^4 y^4$, Evaluated at $x = 0.9$, $y = 0.8$

Order of Taylor Series	Exact Solution	3rd order	3rd order	3rd order	2nd order	2nd order	2nd order
Points in Cloud		20 points	17 points	9 points	17 points	8 points	5 points
Cloud Configuration		Two Rings R = 0.01 10 pts R = 0.02 10 pts	Irregular	Two Rings R = 0.01 5 pts R = 0.02 4 pts	Irregular	One Rectangle	One Ring R = 0.01 5 pts
f_x	1.194396	4.0E-06	-4.4E-05	-3.0E-06	1.347E-03	1.13E-03	3.85E-04
f_y	1.3436928	2.2E-06	-3.28E-0.5	-2.8E-06	-8.952E-04	1.343E-03	4.092E-04
f_{xx}	3.981312	2.0E-03	1.375E-03	1.288E-03	7.723E-02	-1.958E-01	-4.681E-02
f_{xy}	5.971968	3.182E-03	1.419E-03	-1.2E-03	1.663E-01	-1.212E-02	-3.848E-02
f_{yy}	5.038848	2.57E-03	2.57E-04	1.22E-03	1.361E-01	2.281E-01	4.982E-02
f_{xxx}	8.84736	2.743E-02	6.131E--01	7.51E-01			
f_{xxy}	19.90656	-9.2E-03	4.952E-01	2.41E-01			
f_{xyy}	22.39488	1.059E-02	6.912E-01	-6.73E-02			
f_{yyy}	12.59712	-3.995E-02	5.69E-01	-2.26E-01			
		Data Shown Above are (Calculated - Exact) Values					

Blank Page

5 Point Adaption Methodology

James M. Nelsen & N. Todd Snyder

5.1 Introduction

Adaption is a procedure that complements and enhances the capability of numerical methods. The goal of adaption is twofold: provide a more accurate solution and minimize the computational effort. A more accurate solution will globally approach the true solution while capturing the important local features. Computational effort is minimized when calculations that provide negligible contributions to the accuracy or resolution of the solution are eliminated.

During the initial process of applying most numerical methods, a preprocessing procedure of "seeding" the problem domain, whether by meshes or points, is required. Generally, this initial "seeding" is not an optimum one regarding the criteria of accuracy and computational effort. Hence the introduction and utility of adaption.

Adaption techniques can be categorized into three distinct types: movement, refinement/coarsening and higher-order approximations. All three types have been investigated and utilized in traditional grid-based solvers. Adaption by movement is the most mature of the three, with the spring analogy perhaps the most widely used implementation. In this technique, the mesh structure is moved according to a local spring constant, which is itself a predefined function of the solution field. Adaption by mesh refinement/coarsening, sometimes referred to as h-adaption, will locally refine or coarsen the mesh according to a programmed algorithm. Similarly, adaption by higher-order approximations, sometimes referred to as p-adaption, will locally modify the order of the approximating function to an appropriate level according to a programmed algorithm.

Extension of these standard adaption techniques for gridless methods is straightforward. In fact, adaption for gridless techniques should be easier since points are geometrically simpler structures than cells. The primary consideration for adaption, regardless of the type, is the composition and structure of the algorithm that drives the adaption procedure. This logic path must consider relevant issues such as: when to adapt; the proper forcing function for adaption; the order and/or concurrency of adaption types; and perhaps most importantly, any imposed limitations on the adaption procedure. This is the primary focus of the research work in adaption for this LDRD.

5.2 Goals for Adaption

The goals for the adaption algorithm under development are stated as follows:

1. Achieve a level of semi-autonomy, allowing the user to intervene only to prescribe a general purpose accuracy parameter;
2. Incorporate generality, permitting the solution of a wide class of problem sets;
3. Achieve a state of robustness, in which the user-defined level of accuracy can be accomplished, regardless of the initial point seeding;
4. Allow generality of the adaption algorithm technology to accommodate multi-dimensional problems.

A general methodology was formulated and developed to achieve these goals. The basic rule set for the adaption algorithm is outlined in this chapter.

5.3 The Adaption Algorithm

Determination of the appropriate parameters on which to adapt is the primary consideration in constructing the adaption algorithm. A sampling of these parameters include: the local error from the "true" solution (generally not available), the approximate local error associated with truncation of the approximating function (which usually is dependent on the independent spatial variable), the local residual (generally a function of the rate of convergence which may not be directly associated with the error from the true solution), gradients of dependent variables and their derivatives, the condition number of the local neighboring point approximation matrix (which may affect accuracy through numerical round-off) and the interaction of the spatial resolution and grouping of points related to the approximating function. In some form, all of these parameters must be considered.

Because of their generality and their availability with this gridless technique, the current adaption algorithm utilizes the dependent variable, the gradient of the dependent variable and the derivative of the gradient. This implementation is a slight departure from adaption in commercial solvers, which primarily use the gradients of the dependent variables. By allowing differences in both the variable magnitudes and their first and second derivatives (changes in the slope and curvature of the solution) to influence the adaption of points, the methodology is both generalized and moderated.

The mathematical construction of the adaption parameter is shown in equations (5.1) and (5.2). The adaption parameter comprises two spring coefficients that are essentially linear sums of the uniformly weighted, normalized, absolute values of the function and its first and second derivatives. These linear spring coefficients are utilized in multiple roles in the hierarchical process of the adaption algorithm that is described below.

$$K_{i-1} = \left(\left| \frac{f(x)_i - f(x)_{i-1}}{\Delta f(x)_{max}} \right| + \left| \frac{f'(x)_i - f'(x)_{i-1}}{\Delta f'(x)_{max}} \right| + \left| \frac{f''(x)_i - f''(x)_{i-1}}{\Delta f''(x)_{max}} \right| \right) \tag{5.1}$$

$$K_{i+1} = \left(\left| \frac{f(x)_i - f(x)_{i+1}}{\Delta f(x)_{max}} \right| + \left| \frac{f'(x)_i - f'(x)_{i+1}}{\Delta f'(x)_{max}} \right| + \left| \frac{f''(x)_i - f''(x)_{i+1}}{\Delta f''(x)_{max}} \right| \right) \tag{5.2}$$

The first step in the adaption process is to identify the criteria under which adaption would be advantageous to the solution. Certainly, prior to engaging any form of adaption, the solution should have achieved some level of convergence such that the dependent variable's values and derivatives are reasonably accurate. This condition ensures that adaption will not contribute to solution divergence. Ideally, the local and global solution residuals should have converged to a specified level (e.g., two orders of magnitude on a normalized scale) and demonstrated a decrease in the rate of residual convergence.

Assuming the solution has converged to an appropriate level, the first adaption procedure implemented is point movement. Adaption by point movement is governed by spring forces, where a generalized spring force is given by equation (5.3) (the term $F_{i-1 \rightarrow i}$ refers to the force acting on point "i" by point "i-1"). The actual point movement is governed by equation (5.4),

which incorporates the force vectors and is normalized by the sum of the spring coefficients. Point movement continues until the discrete spring force imbalance declines below a user specified tolerance, as provided by equation (5.5).

$$F_{i-1 \to i} = K_{i-1} \cdot (x_{i-1} - x_i) \tag{5.3}$$

$$x_{chg} = (F_{i-1 \to i} + F_{i+1 \to i})/(K_{i-1} + K_{i+1}) \tag{5.4}$$

$$\left| |F_{i-1 \to i}| - |F_{i+1 \to i}| \right| < \text{tolerance} \tag{5.5}$$

The process of adaption by point movement is an iterative procedure. That is, multiple passes of each individual point may occur during a single adaption event, without the solution being updated during this process. This iterative procedure is necessary due to the discrete spring approach adopted for the system point movement algorithm. However, because of this iterative point movement approach, several constraints must be imposed on the process, as described below.

The first constraint imposed on this adaption technique is to limit the maximum distance that an individual point may move during a single iteration. This constraint is necessary for two reasons: it would be undesirable to have points leapfrog neighboring points and the algorithm that updates the dependent variables' value and derivatives (during the adaption procedure) is based on linear interpolation, which is relevant for short distance moves only. The second constraint is similar to the first, in that the cumulative movement of individual points during the entire iterative procedure of a single event of adaption by point movement is limited. Typically, an individual point is limited to a single iteration movement distance of 5% of the distance to the nearest neighboring point, while the cumulative constraint is restricted to twice that of the single iteration distance.

A third constraint imposed during the point movement algorithm is necessary to counteract the potential oscillatory nature of a system of individual, but coupled springs. That is, point movement is actually governed by equation (5.6), which is similar to equation (5.4), but incorporates a numerical viscosity dampening factor, ω. The dampening factor ranges from 0 to 1, with a value of 0.9 to 0.95 typical. It is only invoked when an individual point movement reverses direction during a single adaption event, thereby dampening the oscillatory movement.

$$x_{chg} = [(F_{i-1 \to i} + F_{i+1 \to i})/(K_{i-1 \to i} + K_{i+1 \to i})] \cdot \omega \tag{5.6}$$

Upon completion of adaption by point movement, the next procedure implemented in the adaption process is a refinement/coarsening scheme. Essentially, if the magnitude of any one of the individual terms that are summed in the spring coefficients, given by equations (5.1) and (5.2), exceed a user-defined upper bound accuracy parameter (typically 0.05), then a point is added in this interval. Conversely, if the magnitude of all of the individual terms that are summed in the aforementioned spring coefficients decline below a user-defined lower bound accuracy parameter (typically 1/3 of the upper bound value), then the point is deleted from the system.

The addition of a point using this adaption algorithm is a three-step process. First, the distance between the nearest neighbors on either side of the central point is calculated. Second, the central point is moved to the 1/3 position of this calculated distance. Third, a new point is

positioned at the 2/3 distance location. This procedure attempts to uniformly space the new addition, allowing the final positioning to be accomplished by the next iteration cycle of the point movement adaption process. Values of the dependent variables magnitude and derivatives for the repositioned center point and the new point are derived using linear interpolation from the nearest neighboring points.

A single constraint is imposed in conjunction with this refinement/coarsening adaption procedure. That is, points that are nearest neighbors to a point that has been refined or adapted cannot themselves be refined or adapted during this particular adaption cycle. This condition ensures that adaption by point movement has an opportunity to reposition the points appropriately, without the potential for extraneous local refinement or coarsening.

The aforementioned procedures and constraints reflect the 1-D point adaption scheme as currently implemented. This algorithm addresses the first three goals desired for the 1-D adaption implementation. It achieved the semi-autonomy desired in goal one, while providing a broad based from which to verify the generality of problem scope and robustness desired in goals two and three.

In summary, the current adaption scheme incorporates both point movement and point refinement and coarsening procedures. While the gridless code possesses the capability to globally utilize higher-order approximation functions, the extension to local p-type adaption has not been implemented, due to unresolved issues concerning the appropriate combination of p- versus h-type adaption and the coexistence of neighboring points with an incongruent order of the approximation function.

5.4 Adaption Algorithm Performance

The performance of the adaption algorithm is now assessed through application of several test problems. These test problems are posed in the form of linear and nonlinear ODE's. Each differential equation considered has a known closed-form solution, which allows an accurate assessment of the numerical gridless solution technique and the assistance provided by implementation of the current adaption algorithm.

The first problem investigated is the simple linear, 2nd order ODE, given by eq, (5.7):

$$\frac{\partial^2 f}{\partial x^2} = (5)\sin(\pi x) \tag{5.7}$$

Using the two Dirichlet boundary conditions $f(0) = 1$ and $f(1) = 0$ results in the closed form solution given by equation (5.8).

$$f = 1 - ((5/\pi^2)\sin(\pi x) + x) \tag{5.8}$$

Figure 5.1 shows the comparison between this closed form solution (the lines represent the function and its first and second derivatives) and the solution generated by the gridless technique, utilizing 21 uniformly spaced points across the x-domain [0,1]. As observed, the comparison is excellent for this simple linear ODE.

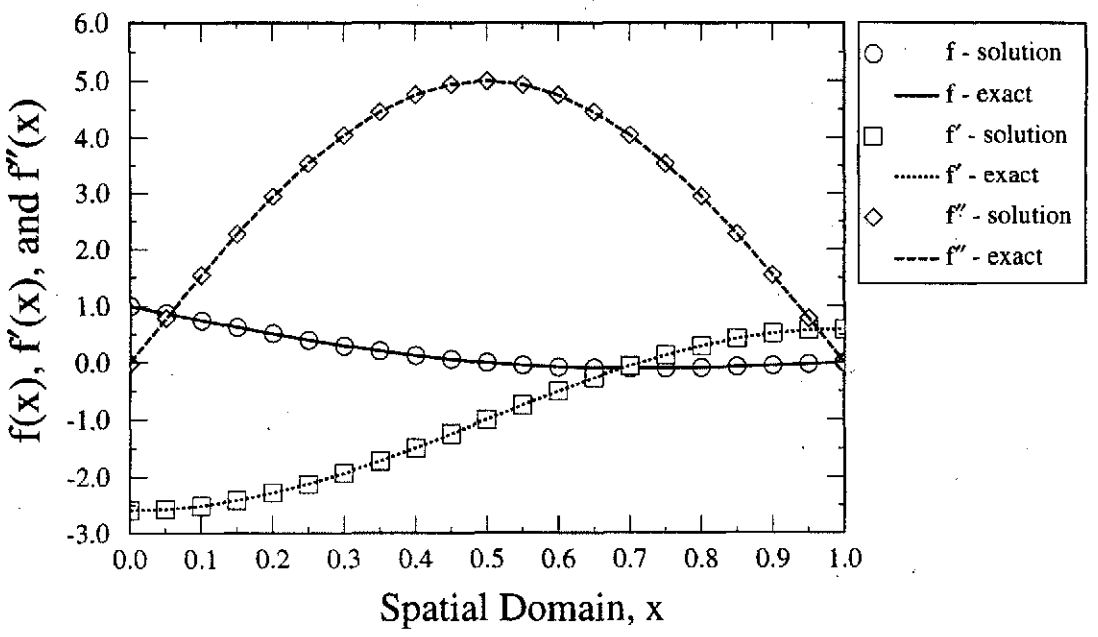

Figure 5.1 Solution to the linear ODE $\frac{d^2 f}{dx^2} = (5)\sin(\pi x)$

Since this particular ODE is well behaved, it does not pose a severe challenge to the adaption algorithm in capturing large gradients in the solution. However, two other aspects of the adaption routine can be investigated with this simple ODE. The first considers the movement of points that have been intentionally clustered. If the adaption by point movement is configured correctly, the clustered points should be repositioned to a more uniform spacing. For this test case, the adaption by refinement/coarsening has been disabled, even though the latter type of adaption would be more computationally effective through judicious implementation of point deletion and insertion.

The result of the adaption by point movement test case is displayed in Figure 5.2. The top line shows the initial clustering of the points. The middle line demonstrates the final point locations subsequent to repositioning by movement adaption. As observed and desired, adaption by movement has repositioned the initially clustered points to a more uniform spacing. The bottom line provides an "ideal" point positioning for this method of adaption. This ideal distribution represents a point seeding in which the analytically determined $k\Delta x$ value is identical for each point. As observed, the adaption method has distributed the points adequately, when compared with the ideal distribution.

The second test case considers the robustness of adaption by refinement/coarsening. Specifically, in solving the aforementioned linear ODE, two sets of initial point distributions were utilized. Within the tolerance and accuracy of the user-defined upper and lower bounds parameter, the adaption algorithm should be sufficiently robust to determine the approximate optimum number of points for this solution, regardless of the initial point seeding. In the first

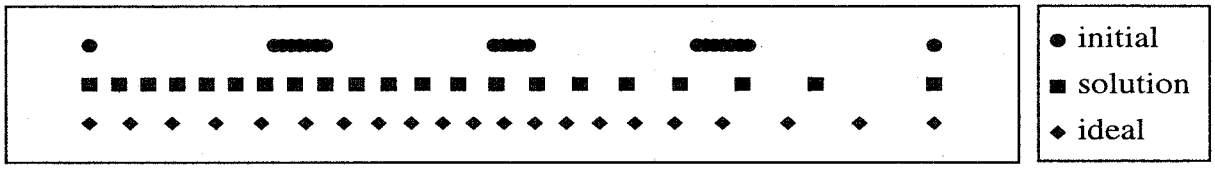

Figure 5.2 Point Movement

case, the gridless calculations were initiated with 11 points, while the second case utilized 101 points. For the first case, the refinement algorithm added points resulting in a final value of 30 points. In the second case, the coarsening algorithm deleted points, resulting in a final value of 35 points. As indicated previously, the number of final points is highly dependent on the ratio of the user-defined upper and lower bounds parameter, which results in the approximate, but not identical, final point values.

To further demonstrate the robustness and overall capability of the gridless technique, two additional differential equations were solved. For both of these solutions, adaption by point movement and refinement/coarsening were utilized.

The next example is a slight modification to the previous linear 2nd-order ODE, equation (5.7), given by

$$\frac{d^2 f}{dx^2} = (5\pi)^2 \sin(5\pi x) \tag{5.9}$$

Utilizing the boundary conditions $f(0) = 1$ and $f(1) = 0$ results in the solution shown in Figure 5.3. The higher frequency oscillations of this ODE solution require additional fidelity from the solution technique. However, as observed, the gridless solution has captured the true solution. Further, the point adaption appears to have clustered at the peaks and troughs of the solution where the first and second derivatives exhibit their greatest change.

The final example considered is the nonlinear Burger's equation, given by equation (5.10).

$$\frac{\partial \phi}{\partial t} + \phi \frac{\partial \phi}{\partial x} - \frac{1}{Re} \frac{\partial^2 \phi}{\partial x^2} = 0 \tag{5.10}$$

The closed form solution for this equation was obtained using a traveling wave solution technique, which results in equation (5.11):

$$\phi(x,t) = \frac{f_2 + f_1 e^{\left(\frac{Re}{2}(f_2 - f_1)(x - ct)\right)}}{1 + e^{\left(\frac{Re}{2}(f_2 - f_1)(x - ct)\right)}} \tag{5.11}$$

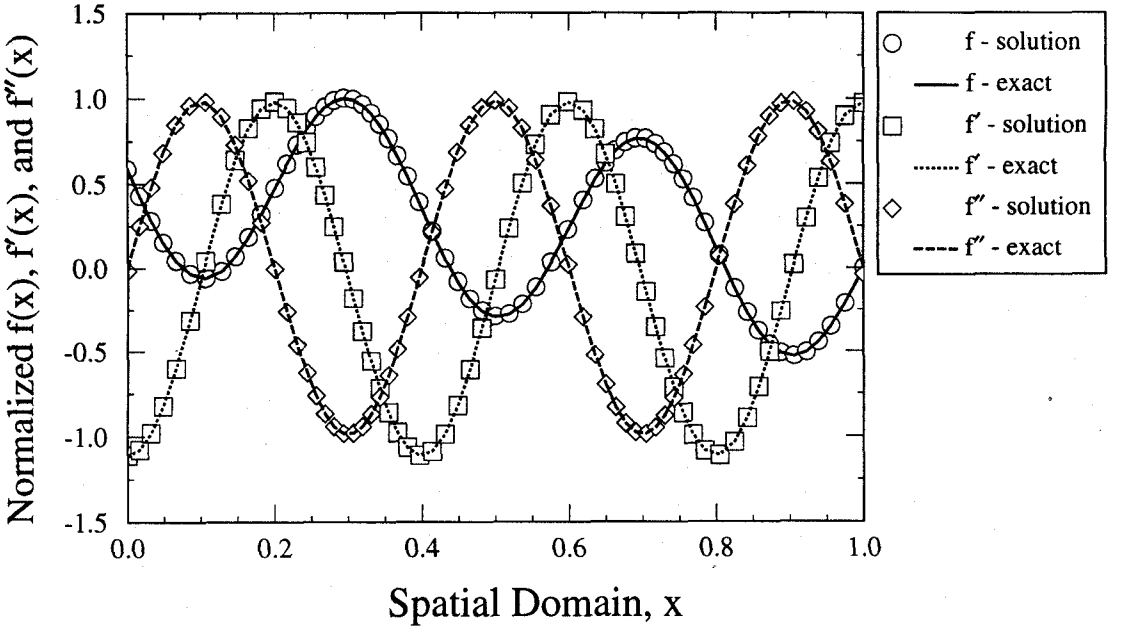

Figure 5.3 Solution to the linear ODE $\frac{d^2 f}{dx^2} = (5\pi)^2 \sin(5\pi x)$

where the wave speed is given by $c = (f_2 + f_1)/2$. The variable ϕ asymptotically approaches the limits f_1 to the right and f_2 to the left. By selecting the parameters $f_2 = 1$ and $f_1 = -1$, the wave speed is set to zero, which simplifies the solution and eliminates the time dependency.

As a result of the choice of f_1 and f_2, the boundary conditions are established as $\phi(-1) = 1$ and $\phi(1) = -1$. The Reynolds number for this solution is arbitrarily established at 50. The comparison between the numerical gridless solution and the analytic closed form solution is presented in Figure 5.4. As observed, the gridless technique has captured the true solution for this nonlinear ODE. However, it should be noted that the tolerance applied to the point movement algorithm was relaxed slightly, which stabilized the solution in the region near the discontinuity and reduced convergence times by a factor of three to four.

5.5 1-D Adaption Summary

Several example problems, including both linear and nonlinear ODE's, were solved using the gridless technique developed in this LDRD project. Incorporated within this gridless technique is the capability to adaptively move points, which can enhance solution accuracy and minimize computational expense. Point adaption was demonstrated using both movement and coarsening/refinement techniques. The adaption algorithm was demonstrated to be autonomous and robust, which achieved the goals established for this task in the LDRD. Further, the unique formulation of the adaption parameter that utilizes local differences in the variable's value and

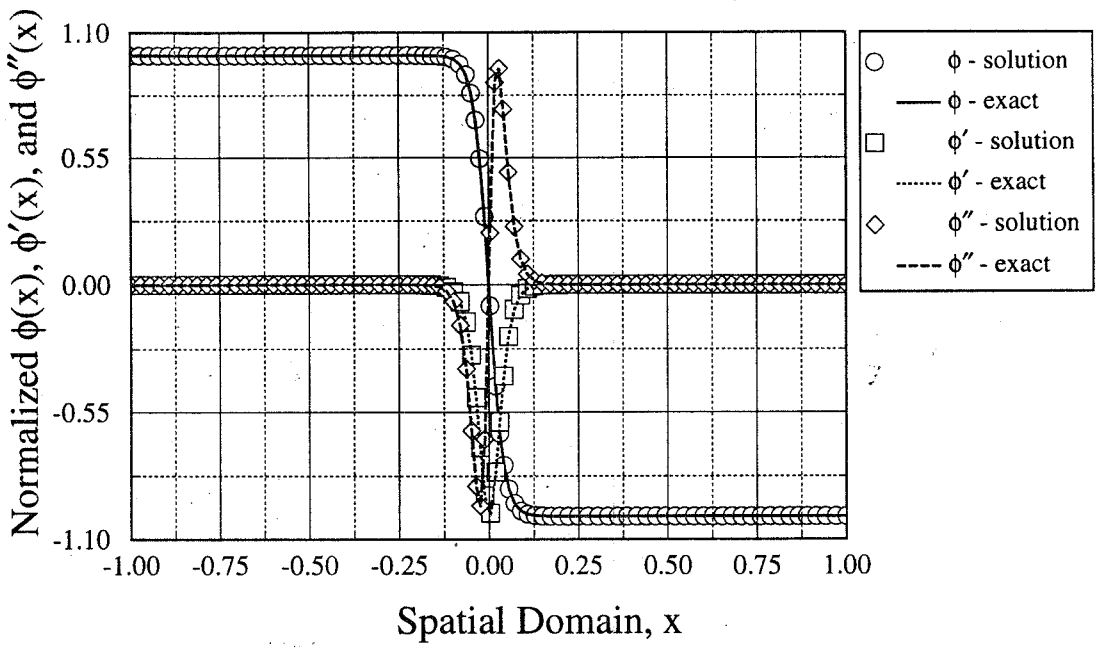

Figure 5.4 Solution to the nonlinear ODE $\frac{\partial\phi}{\partial t} + \phi\frac{\partial\phi}{\partial x} - \frac{1}{Re}\frac{\partial^2\phi}{\partial x^2} = 0$

its first and second derivatives performs extremely well in capturing the proper characteristics of the ODE's solution.

5.6 The Adaption Methodology in Multi-Dimensions

The extension of the 1-D adaption methodology to 2-D (or higher dimensions) is not difficult. However, it encompasses additional degrees of freedom for implementation that generally manifest as more complicated limitations on the algorithm. The fundamental logic for point adaption is independent of the spatial dimension. That is, point movement precedes point refinement/coarsening and its complementary technology of higher order approximations. The user defined accuracy bounds defined for 1-D are still applicable, as are the limitation imposed on the adaption procedures outlined for 1-D. The departures from the 1-D adaption methodology are outlined in the paragraphs below.

Regarding point movement, the consequence of the multi-dimensional spring system is not 1-D translation, but a linear translation that incorporates the contributions of all the surrounding points of the cloud group. This requires the simultaneous solution of the cloud group spring forces to achieve the point adaption equilibrium during a single iteration of the movement adaption cycle. By necessity, the point distribution of the cloud group surrounding the focal point will be important to minimize directional dependence of the point adaption through movement.

Adaption by point deletion and insertion is less constrained than the analogous 1-D case, however the basic rule set is still applicable. That is, adaption by point movement continues until a prescribed level of translation equilibrium is achieved, at which time the adaption by

point deletion and insertion may be considered. As in the 1-D case, if the adaption parameter (specifically the linear sum of the difference in point values, first, and second derivatives detailed previously) exceeds a user-defined accuracy value for all the neighbors in a cloud group (or a majority of the neighboring points, depending on the degree of equilibrium achieved), then a point insertion process may be implemented. Conversely, if the adaption parameter declines below a user-defined lower bound for all the neighbors in a cloud group, then the point may be deleted. For the point insertion algorithm, the following procedure is proposed. Insert a point between the focal point and its neighbor with the highest adaption parameter magnitude. Continue this procedure with all of the other neighboring points. It may be preferable, however, to limit the number of points added by skipping every other neighbor point during a circumferential search. Other limitations, such as distance parameters for validity of the Taylor series approximation may be important and should be investigated either mathematically or empirically as a future research topic.

5.7 The Adaption Methodology for Time-Accurate Problems

The implementation of adaption for time-accurate problems is very complex. Whereas an optimum point location can be determined gradually during the iterative process associated with the solution of a steady-state problem, a time-accurate problem requires reasonable point placement at each successive time step. This necessitates an adaption algorithm that can rapidly react to the transient, even though it will inherently lag the solution. It would be more desirable to produce an algorithm that is not reactive, but predictive in nature. This might be accomplished utilizing a predictor-corrector scheme that is intrinsically coupled to the time-accurate solver. By tracking trends in the point adaption, points could be more optimally positioned for future time steps. This task represents another future research topic, but certainly not one that represents a barrier to the generality of the adaption methodology under development.

Blank Page

6 Shell Elements

Frank Mello

6.1 Introduction

Recent advancements in computing technology have broadened the horizons of simulation capabilities by increasing the size and complexity of the models that can be considered. Frequently these simulations span different time scales as well as regions with different governing equations. Such is the case in coupled fluid structure modeling. A new technique for solving fluid dynamics equations on a random unstructured collection of points is being evaluated. In support of this gridless fluid modeling LDRD, plane strain and axisymmetric shells have been added to PRONTO2D. The 3D shell element developed by Belytschko, et al.[11, 12] has been specialized for 2D. The reduction in dimension dramatically simplifies the implementation. Rotation reduces to a vector of fixed orientation and the need for tracking nodal and element coordinates is eliminated. Further, there are no hourglass modes to control in 2D shells. The same commands used to describe shell element attributes in PRONTO3D[13, 14] are available in 2D, including shell thickness scaling and layered shell definitions. Much of the text in this chapter is borrowed from documentation of the 3D features.[13, 14]

The coupling between fluid calculations and structural calculations will occur by passing boundary condition data between the codes which will run simultaneously. This strategy has been effective in linking PRONTO3D and CTH.

6.2 Governing Equations

6.2.1 Equations of Motion

Motion of a deformable body is governed by the balance of linear momentum

$$\nabla \cdot \boldsymbol{T} - \rho \ddot{\boldsymbol{u}} + \rho \boldsymbol{b} = 0 \tag{6.1}$$

where $\boldsymbol{T}$ is the stress tensor, ρ is the material density, $\ddot{\boldsymbol{u}}$ is the acceleration vector and $\boldsymbol{b}$ is the body force vector. When a body is decomposed into a finite element mesh, the momentum balance equation is enforced in a weak sense over each element volume. The assembled weak form, after an integration by parts is then;

$$\sum_e \left[\int_{S_e} T_{ij} n_j \delta \dot{u}_i dS_e - \int_{V_e} T_{ij} \delta \dot{u}_{i,j} dV_e + \int_{V_e} \rho b_i \delta \dot{u}_i dV_e - \int_{V_e} \rho \ddot{u}_i \delta \dot{u}_i dV_e \right] = 0 \tag{6.2}$$

where $\delta \dot{u}_i$ is an arbitrary virtual velocity field in the shell body. The first term in this expression is the rate of work of externally applied tractions and/or displacements. The following terms correspond to the rates of strain energy, work of the body forces, and work of the inertial forces, respectively. Since PRONTO uses a lumped mass approach, the above relation reduces to a set of uncoupled equations describing the motion of the nodal points. The acceleration of the Jth nodal mass may then be expressed as

$$\ddot{u}_{iJ} = \frac{(f_{iJ}^{EXT} - f_{iJ}^{INT})}{M_J} \tag{6.3}$$

In shell elements the nodal points carry rotational degrees of freedom in addition to translations. The kinematical assumptions discussed in the following section require the tracking of curvature rates at the nodal points and give rise to rotational acceleration relations in the principal inertia frame of the form

$$\begin{aligned}
\alpha_{1J} &= \frac{[(m_{1J}^{EXT} - m_{1J}^{INT}) - (I_{3J} - I_{2J})\omega_{3J}\omega_{2J}]}{I_{1J}} \\
\alpha_{2J} &= \frac{[(m_{2J}^{EXT} - m_{2J}^{INT}) - (I_{1J} - I_{3J})\omega_{1J}\omega_{3J}]}{I_{2J}} \\
\alpha_{3J} &= \frac{[(m_{3J}^{EXT} - m_{3J}^{INT}) - (I_{2J} - I_{1J})\omega_{2J}\omega_{1J}]}{I_{3J}}
\end{aligned} \tag{6.4}$$

In 2D this reduces to a single equation, since rotations can occur only about the normal to the plane of the problem.

$$\alpha_J = \frac{m_J^{EXT} - m_J^{INT}}{I_J} \tag{6.5}$$

The simplicity of this single equation for rotation leads to the primary reduction in complexity for 2D shells. Since the only principal direction which admits rotation always has a fixed orientation the need to track nodal rotation coordinates is eliminated.

6.3 Shell Kinematics

Mindlin shell theory[15] assumes that sections through the shell thickness remain straight, although they may rotate. Rotation of the section allows the element to model transverse shear strains. Since displacements are assumed to vary linearly through the thickness, the velocity at any point may be expressed in terms of midplane values as

$$\dot{\boldsymbol{u}} = \dot{\mathbf{u}} + \hat{z}\boldsymbol{e} \times \omega\boldsymbol{k} \tag{6.6}$$

in which $\dot{\boldsymbol{u}}$ is the velocity of a point in the shell body, $\dot{\mathbf{u}}$ is the velocity of the point on the midsurface which has the same normal, and $\omega\boldsymbol{k}$ is the rotational velocity of the section. The coordinate of the point through the shell thickness is $\hat{z}$ and the unit vector along the normal is $\boldsymbol{e}$. Since this is a 2D shell formulation, the unit vector $\boldsymbol{k}$, indicating the axis of rotational velocity, is always normal to the plane of the problem.

The components of velocity strains in the corotational coordinate system are

$$\hat{d}_{ij} = \frac{1}{2}\left(\frac{\partial \dot{u}_i}{\partial \hat{x}_j} + \frac{\partial \dot{u}_j}{\partial \hat{x}_i}\right) \tag{6.7}$$

Substitution of the above relation for $\dot{\boldsymbol{u}}$ into the velocity strain definitions yields the strain displacement relations for the shell.

$$\begin{aligned}
\hat{d}_{11} &= \frac{\partial \dot{u}_1}{\partial \hat{x}_1} - \hat{z}\frac{\partial \omega}{\partial \hat{x}_1} \\
\hat{d}_{33} &= 0 \text{ for plane strain; } \hat{d}_{33} = \frac{\dot{r}}{r} \text{ for axisymmetric} \\
\hat{d}_{22} &= -\frac{\nu}{2(1-\nu)}(\hat{d}_{11} + \hat{d}_{33}) \text{ from plane stress} \\
\hat{d}_{12} &= \frac{1}{2}\frac{\partial \dot{u}_2}{\partial \hat{x}_1} - \frac{1}{2}\omega
\end{aligned} \tag{6.8}$$

where the subscript 1 refers to the direction defined by progressing from the first node of the element to the second. The subscript 2 refers to the thickness direction and subscript 3 refers to the direction into the plane of the problem.

6.3.1 Constitutive Assumptions

If the velocity strain components in the corotational system are arranged in a column matrix as;

$$\hat{d} = [\hat{d}_{11}, \hat{d}_{22}, \hat{d}_{33}, 2\hat{d}_{12}]^T \tag{6.9}$$

then the conjugate Cauchy stress components can be written as

$$\hat{\sigma} = [\hat{\sigma}_{11}, \hat{\sigma}_{22}, \hat{\sigma}_{33}, \hat{\sigma}_{12}]^T \tag{6.10}$$

Furthermore, the shell is assumed to be in a state of plane stress, so

$$\hat{\sigma}_{22} = 0 \tag{6.11}$$

These representations are conjugate in the sense that the rate of internal strain energy density, $\dot{W}$, can be expressed as

$$\dot{W} = \hat{d}^T \hat{\sigma} \tag{6.12}$$

6.3.2 Element Formulation

The shell element in PRONTO2D uses a two noded line segment to model the middle surface of a thin structure (or some other reference surface if an offset is given).

6.3.3 Coordinate Systems

In the three dimensional formulation three coordinate systems are needed. Translational motion is described in the global system. Element calculations are performed in an element-based system and rotational motion is described in a nodal system aligned with the principle directions of the nodal inertia tensor. Approximating the nodes as spheres (i.e. homogenizing

their inertias) is a reasonable approximation for well shaped elements and eliminates the need for a nodal coordinate system.

In the two dimensional framework we use a global coordinate system and an element system. No nodal system is required since the inertia of each node is a scalar (motion is confined to the plane of the problem). Transformations between global and element systems is particularly simple since both have the same third axis.

6.3.4 Element Equations

The shell element is based on a two noded line segment with linear interpolation of reference surface coordinates and of both translational and rotational velocities. Coordinates in the reference surface of the shell are approximated as

$$x_i = x_{iI}\phi_I(\xi) \tag{6.13}$$

where ϕ_I are the shape functions. Repeated upper case indices indicate summation over the two nodes of the element. Similarly, the velocity of the reference surface and the angular velocity of the shell normal are interpolated as

$$\begin{aligned} \dot{u}_i &= \dot{u}_{iI}\phi_I(\xi) \\ \omega &= \omega_I\phi_I(\xi) \end{aligned} \tag{6.14}$$

using the same shape functions.

For the case of two dimensional shells, these shape functions may be expanded in terms of an orthogonal set of base vectors as

$$\phi_I = \frac{1}{2}\Sigma_I + \xi\Lambda_{1I} \tag{6.15}$$

where ξ is the coordinate in the unit line into which the element is mapped. The base vectors represent deformation of the unit line.

Node	ξ	Σ_I	Λ_{1I}
1	-.5	1	-1
2	.5	1	1

Rigid body motion is represented by the first vector, Σ_I. The volumetric base vector, Λ_{1I}, represents the stretching of the element and completely describes the element area. For this element, there are no neglected modes and therefore no hourglass control is required. Since we are only interested in quantities at the element center, it is easily seen that the velocities are the average of the nodal values and the velocity gradients are the differences between node 2 and node 1 values divided by the current element length.

6.3.5 Computation of Internal Force and Moment Resultants

As mentioned above the gradient operator for this element reduces to a difference between nodal quantities divided by the element length. So the velocity strains at the reference surface may be expressed as

$$
\begin{gathered}
d_{11} = \frac{v_{21} - v_{11}}{l_{12}} \\
d_{22} \text{ determined from plane stress assumptions} \\
d_{33} = 0 \text{ plane strain; } d_{33} = \frac{\dot{r}}{r} \text{ axisymmetry} \\
d_{12} = \frac{1}{2}\frac{(v_{22} - v_{12})}{l_{12}} - \frac{1}{4}(\omega_2 + \omega_1) \\
\dot{\kappa} = \frac{\omega_2 - \omega_1}{l_{12}}
\end{gathered}
\tag{6.16}
$$

where $\dot{\kappa}$ is the curvature rate. The constitutive relations operate on these velocity strains to produce stress increments. The total stress is then integrated to element forces and moments. These forces are transformed to global components and assembled. Moments are computed directly from the integration point forces and the offsets.

6.3.6 Hourglass Control

There is no hourglass control for two dimensional shells.

6.3.7 Calculation of Stable Time Increment

The central-difference operator is conditionally stable with the stability limit for a given system with no damping being

$$\Delta t \leq \frac{2}{\omega_{max}} \tag{6.17}$$

where ω_{max} is the maximum frequency of the system. Here the maximum frequencies of a two dimensional shell system are conservatively estimated based on the most heavily constrained collection of elements which still admits deformation. Frequencies associated with translations and rotation have been considered. The membrane frequency accounts for the hoop stiffness in axisymmetric problems. Further, the rotational frequency will always bound the transverse deformation case. So the critical time step is based on the largest of the membrane and rotational frequencies.

$$\omega_{m,\,max} = 2\sqrt{\frac{EA}{lM}}$$

$$\omega_{\Theta,\,max} = \sqrt{\frac{24EI}{l^3 M}}$$

$$E = \text{Youngs Modulus} \tag{6.18}$$

$$A = t \text{ in-plane; } A = lt \text{ hoop}$$

$$I = \frac{t^3}{12} = \text{bending inertia}$$

$$M = \text{element mass}$$

$$l \text{ element length} = 2\pi r \text{ for axisymmetry}$$

6.4 Constitutive Models

Currently two material models have been implemented, elastic and combined hardening elastic-plastic. More constitutive models will be included as needed.

7 Summary and Recommendations

A gridless method has been developed for the simulation of coupled fluid/structural interactions over arbitrary bodies. This method uses Eulerian-based points arbitrarily distributed over the computational domain with no formal connectivity as typically required for a traditional grid. Each point in the domain has associated with it a cloud of neighboring points. Derivatives in the governing PDE's are numerically approximated by assuming a functional dependence of each variable within the local cloud of points, where the number of points in the cloud is greater than or equal to the number of unknown derivatives. The governing partial differential equations are then solved locally at each point in the computational domain. This gives a solution to the governing equations without any a priori assumptions concerning grid or element topology, and allows for addition, deletion, or movement of points with no change in solution algorithms.

The gridless method is neither a finite difference, finite volume, nor a finite element approach since differences, metrics, areas, or volumes, are not computed. The methodology is extremely versatile in that the method permits one to solve the relevant equations based on the complexity of the problem at hand. Hence, the difficulty of simulating either a potential flow, Euler flow, or full Navier-Stokes flow, and the ability to model compressible or incompressible flows is dictated only by the set of PDE's solved and is easily amenable to a modular code structure. This modularity appears as a simple consequence of the fact that the equations of motion are independent of the point distribution and the solution algorithm is completely grid free.

The gridless method had been implemented in a two-dimensional code written in ANSI standard C. Example calculations using Laplace's equation as the model test problem have shown that the current implementation of the gridless method gives reasonable accuracy when a second-order approximation is used to calculate local derivatives. Comparisons with optimized coupled linear methods for the solution of the equations have shown that the current basic implementation is very near optimum, given the point distribution and order of approximations. Accuracy can be improved by using a third-order approximation in regions with larger gradients, such as on the body surface. Point adaption has also shown promise for improved accuracy. However, point adaption has not yet been extended to two dimensions. The optimum balance between higher-order derivative approximations and point adaption has not been determined.

For the structural simulation side of the coupled fluid/structure problem, plane strain and axisymmetric shells have been added to the code PRONTO2D. The three-dimensional shell element developed by Belytschko, et al, has been specialized for two dimensions. The reduction in dimension dramatically simplifies the implementation. Rotation reduces to a vector of fixed orientation and the need for tracking nodal and element coordinates is eliminated. Further, there are no hourglass modes to control in two dimensional shells. The same commands used to describe shell element attributes in PRONTO3D are available in two dimensions, including shell thickness scaling and layered shell definitions. The coupling between fluid calculations and structural calculations occurs by passing boundary condition data between the codes which

run simultaneously. This strategy has proven effective in linking the codes PRONTO3D and CTH.

More work needs to be done to bring the current gridless code to a production status. One of the more pressing needs is to extend the point adaption scheme to two dimensions. Once this is done, trade-off studies can be run to determine the balance between point adaption and higher order approximations for improved solution accuracy and speed. Work also is needed to improve the code efficiency so that run times do not increase as the square of the number of field points. Improvements need to be incorporated to extend the code beyond the current Laplacian model problem to more complicated fluid equations, such as the Euler and Navier-Stokes equations. Extensive coupled fluid/structural calculations are needed to ensure that all of the problems of coupling the two codes are resolved. Finally, after the above improvements are made, detailed comparisons are needed between the gridless method code and existing fluid mechanics codes with similar capabilities. Even after the gridless code has been shown to work for all cases, it will not have been proven that this method is more efficient than, say, a fluid code that uses an unstructured grid and regrids as needed to handle a moving structural boundary.

8 References

1. T. Liska, and J. Orkisz, "The Finite Difference Method at Arbitrary Irregular Grids and Its Application in Applied Mechanics," *Computer and Structures, Vol. II.*, 1980, pp. 83-95.

2. J. T. Batina, "A Gridless Euler/Navier-Stokes Solution Algorithm for Complex-Aircraft Applications, *AIAA 93-0333*, presented at the 31st Aerospace Sciences Meeting and Exhibit, January 11-14, 1993, Reno, NV.

3. R. N. Bryan, M. K. Kinyon, and D. H. Tucker, "Coupled Linear Operators," *Results in Mathematics*, Vol. 15, 1989, pp. 1-14.

4. S. L. Campbell, and C. D. Meyer, *Generalized Inverses of Linear Transformations*, Dover Publications, 1991.

5. G. W. Stewart, *Introduction to Matrix Computations*, Academic Press, 1973.

6. K. Karamcheti, *Principles of Ideal-Fluid Aerodynamics*, Robert E. Krieger Publishing, 1980.

7. W. Gui, and I. Babuska, "The h, p and h-p Versions Of The Finite Element In One Dimension, Parts 1, 2, and 3", *Numerische Mathematik*, 49, Springer-Verlage, 1986, pp. 577-683.

8. L. Demkowicz, J. T. Oden, W. Rachowicz, and O. Hardy, "Toward a Universal h-p Adaptive Finite Element Strategy, Parts I, II and III," *Computer Methods in Applied Mechanics and Engineering*, 77, Elsevier Science Publishers (North-Holland), 1989, pp 79-212.

9. F. B. Hildebrand, *Advanced Calculus for Applications*, 2nd Ed., Prentice Hall, pp. 354-355.

10. W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd Ed., Cambridge University Press

11. T. Belytschko, J. I. Lin, and C. S. Tsay, "Explicit Algorithms for the Nonlinear Dynamics of Shells," *Computer Methods in Applied Mechanics and Engineering*, Vol. 42, pp. 225-251, 1984.

12. T. Belytschko, J. S.-J. Ong, W. K. Liu, and J. M. Kennedy, "Hourglass Control in Linear and Nonlinear Problems," *Computer Methods in Applied Mechanics and Engineering*, Vol. 43, pp. 251--276, 1984.

13. L. M. Taylor, and D. P. Flanagan, "PRONTO 3D--A Three Dimensional Transient Solid Dynamics Program," SAND87--1912, Sandia National Laboratories, Albuquerque, NM, March, 1989.

14. V. L. Bergmann, "Transient Dynamic Analysis of Plates and Shells with PRONTO 3D," SAND91--1181, Sandia National Laboratories, Albuquerque, NM, September, 1991.

15. R. D. Mindlin, "Influence of Rotatory Inertia and Shear on Flexible Motions of Isotropic, Elastic Plates," *Journal of Applied Mechanics*, Vol. 18, pp. 31-38, 1951.

Appendix A: LDRD Reporting Requirements

Awards: none

Publications and presentations: R. S. Baty and W. P. Wolfe, "A Numerical Method for Arbitrary Irregular Grids," to be submitted to the *International Journal of Numerical Methods in Engineering*, Feb. 1996.

Invention disclosures: none

Patents: none

Copyrights: none

Employee recruitment: none

Student involvement: N. Todd Snyder
Department of Mechanical and Aerospace Engineering
Utah State University
Logan, Utah 84322-4130
Master's thesis

Follow-on work: none

Distribution

1	MS 1088	A. L. Thornton, 3600
1	0159	V. L. Dugan, 4500
1	0811	S. P. Goudy, 4918
1	0458	J. R. Asay, 5132
1	0573	D. A. Yocky, 5913
1	0841	P. J. Hommert, 9100
1	0828	R. D. Skocypec, 9102
1	0833	J. H. Biffle, 9103
1	0826	S. N. Kempka, 9111
1	0825	W. H. Rutledge, 9115 please route to 9104, 9111, 9114
1	0825	D. P. Aeschliman, 9115
5	0825	R. S. Baty, 9115
1	0825	F. G. Blottner, 9115
1	0825	J. K. Cole, 9115
1	0836	C. W. Peterson, 9116 please route to 9112, 9113
5	0836	J. M. Nelsen, 9116
1	0836	J. H. Strickland, 9116
5	0836	W. P. Wolfe, 9116
1	0443	A. E. Hodapp, 9117
1	0437	E. P. Chen, 9118 please route to 9117
5	0437	F. J. Mello, 9118
1	0819	T. G. Trucano, 9231
1	0951	D. R. Strip, 9621
1	0951	R. A. LaFarge, 9621
5	0951	G. A. Laguna, 9621
1	1436	LDRD Office
1	9018	Central Technical Files, 8523-2
5	0899	Technical Library, 4414
1	0619	Print Media, 12615
2	0100	Document Processing, 7613-2 For DOE/OSTI

External

5 Dr. Christine E. Hailey
Department of Mechanical and Aerospace Engineering
Utah State University
Logan, Utah 84322-4130

5 N. Todd Snyder
Department of Mechanical and Aerospace Engineering
Utah State University
Logan, Utah 84322-4130

1 Charlie Vono
MS HAFB / 1218
TRW
1104 Country Hills Drive
Ogden, UT 84403

1 Glenn Gebert
Sverdrup Technologies, Inc.
TEAS Group
PO Box 1935
Eglin AFB, FL 32542