Numerical solution of inverse problems in nondestructive evaluation using the boundary element method and multivariate adaptive regression splines

Sinniah Balakrishnan

Iowa State University
INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI
A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600
Numerical solution of inverse problems in nondestructive evaluation
  using the boundary element method and multivariate adaptive
  regression splines

by

Sinniah Balakrishnan

A Dissertation Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Department: Aerospace Engineering and Engineering Mechanics
Major: Engineering Mechanics

Approved:
  Signature was redacted for privacy.

Signature was redacted for privacy.

In Charge of Major Work
  Signature was redacted for privacy.

For the Major Department
  Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa
1996

Copyright © Sinniah Balakrishnan, 1996. All rights reserved.
# TABLE OF CONTENTS

**ACKNOWLEDGMENTS** ..................................................... xiv

**CHAPTER 1. INTRODUCTION** ........................................... 1

**CHAPTER 2. THEORY AND APPLICATION OF THE BOUNDARY ELEMENT METHOD IN FLAW IDENTIFICATION** ........................................... 8

- The Problem .............................................................. 8
- Formulation of the Boundary Integral Equation ..................... 9
- The Minimization of the Functional .................................. 13
  - Minimization process ................................................. 14
  - Spline shape of the flaw ............................................. 15
- Applications .............................................................. 16
- Discussion ............................................................... 23

**CHAPTER 3. MULTIVARIATE ADAPTIVE REGRESSION SPLINES AND THEIR USE FOR FLAW IDENTIFICATION** ........................................... 25

- MARS Algorithm ......................................................... 25
  - Forward Stepwise Algorithm ........................................ 26
  - Backward Stepwise Procedure ....................................... 29
- Selection of the Training Set ......................................... 30
- Selection of the Testing Set .......................................... 33
LIST OF TABLES

Table 3.1: Performance of MARS in sizing the elliptical flaw’s semi-major axis (a) 35
Table 3.2: Performance of MARS in sizing the elliptical flaw’s semi-major axis (b) 36
Table 3.3: Performance of MARS in sizing the elliptical flaw’s center (xc) 37
Table 3.4: Performance of MARS in sizing the elliptical flaw’s center (yc) 38
Table 3.5: Identification of a flaw using MARS; after a global search with a 700 element training set for example 1 41
Table 3.6: Identification of a flaw using MARS; after a local search with a 45 element training set extracted from the 700 element set for example 1 41
Table 3.7: Identification of a flaw using MARS; after a local search with a newly trained 540 element training set for example 1 42
Table 3.8: Identification of a flaw using MARS; after a global search with a 700 element training set for example 2 43
Table 3.9: Identification of a flaw using MARS; after a global search with a newly trained 540 element training set for example 2 43
Table 3.10: Identification of a flaw using MARS, after a global search with a 700 elements training set for example 3.

Table 3.11: Identification of a flaw using MARS; after a global search with a newly trained 540 element training set for example 3.

Table 5.1: Comparison of the number of iterations (N) and the functional value (F) in the minimization process with and without the knowledge of the approximate location of single flaws.
LIST OF FIGURES

Figure 2.1: A domain with Neumann and Dirichlet boundaries .......... 9
Figure 2.2: A domain with a guessed flaw ............................ 11
Figure 2.3: An elliptical real flaw (solid line) and a circular guessed flaw (dot-dot line) in a 2X2 square domain for example 1 ........ 18
Figure 2.4: Predicted flaws (dot-dot lines) at different stages of iteration for example 1 .............................................. 19
Figure 2.5: Variation of the value of the functional with the number of iterations for example 1 .............................................. 19
Figure 2.6: The real flaw (solid line) and a circular guessed flaw (dot-dot line) for example 2 .............................................. 20
Figure 2.7: Predicted elliptical shaped flaw (dot-dot line) and the real flaw (solid line) for example 2 .............................................. 20
Figure 2.8: Predicted spline shaped flaw (dot-dot line) and the real flaw (solid line) for example 2 .............................................. 21
Figure 2.9: The real flaw (solid line) and a circular guessed flaw (dot-dot line) for example 3 .............................................. 22
Figure 2.10: Predicted elliptical shaped flaw (dot-dot line) and the real flaw (solid line) for example 3 .............................................. 22
Figure 2.11: Predicted spline shaped flaw (dot-dot line) and the real flaw (solid line) for example 3 23

Figure 3.1: The forward stepwise MARS algorithm 27

Figure 3.2: The backward stepwise MARS algorithm 31

Figure 3.3: MARS performance in sizing the semi-major axis, $a$ of elliptical flaws as a function of the maximum number of basis functions 35

Figure 3.4: MARS performance in sizing the semi-minor axis, $b$ of elliptical flaws as a function of the maximum number of basis functions 36

Figure 3.5: MARS performance in sizing $x$-coordinate, $x_c$ of elliptical flaws as a function of the maximum number of basis functions 37

Figure 3.6: MARS performance in sizing $y$-coordinate, $y_c$ of elliptical flaws as a function of the maximum number of basis functions 38

Figure 3.7: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test 1 for example 1 40

Figure 3.8: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test 2 for example 1 42

Figure 3.9: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test 1 for example 2 44

Figure 3.10: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test 2 for example 2 44

Figure 3.11: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test 1 for example 3 46
Figure 3.12: Identification of a flaw using MARS: The real flaw (solid line) and the predicted flaw (dot-dot line) after test 2 for example 3

Figure 3.13: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 1

Figure 3.14: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 2

Figure 3.15: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 3

Figure 3.16: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 4

Figure 3.17: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 5

Figure 3.18: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 6

Figure 4.1: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 1

Figure 4.2: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 1 when the added experimental noise is zero

Figure 4.3: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 1 when the added experimental noise is 0.01

Figure 4.4: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 1 when the added experimental noise is 0.05

Figure 4.5: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 2
Figure 4.6: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 2 when the experimental noise is zero ....... 59

Figure 4.7: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 2 when the experimental noise is 0.01 ....... 59

Figure 4.8: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 2 when the experimental noise is 0.05 ....... 60

Figure 4.9: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 1 ........................................ 61

Figure 4.10: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 1 when the added noise is 0.01 ....... 61

Figure 4.11: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 1 when the added noise is 0.05 ....... 62

Figure 4.12: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 2 ........................................ 63

Figure 4.13: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 2 when the added noise is 0.01 ....... 63

Figure 4.14: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 2 when the added noise is 0.05 ....... 64

Figure 4.15: Two real flaws (solid lines) and a single guessed flaw (dot-dot line) in example 1 ........................................ 66

Figure 4.16: Two real flaws (solid lines) and the updated guessed flaw (dot-dot line) in example 1 ........................................ 66

Figure 4.17: Three real flaws (solid lines) and a single guessed flaw (dot-dot line) in example 2 ........................................ 67
Figure 4.18: Three real flaws (solid lines) and the updated guessed flaw (dot-dot line) in example 2 ........................................... 67
Figure 4.19: Two real flaws (solid lines) and two guessed flaws (dot-dot line) in example 1 ......................................................... 69
Figure 4.20: Two real flaws (solid lines) and the predicted flaws (dot-dot line) in example 1 ............................................................. 69
Figure 4.21: Three real flaws (solid lines) and three guessed flaws (dot-dot line) in example 2 ....................................................... 70
Figure 4.22: Three real flaws (solid lines) and the predicted flaws (dot-dot line) in example 2 ....................................................... 70
Figure 5.1: Flow around a closed body ........................................... 73
Figure 5.2: Flow around an infinitely long cylinder of radius, a .......... 74
Figure 5.3: Geometry of a doublet at O in the uniformed stream, U' .... 74
Figure 5.4: Geometry of a circular flaw in a square domain ............... 76
Figure 5.5: A comparison of temperature perturbations on (\(y = -1\)) ... 76
Figure 5.6: A comparison of temperature perturbations on (\(y = 1\)) ... 77
Figure 5.7: Superposition of two single flaws ................................. 78
Figure 5.8: Geometry of the two flaws in the domain for example 1 ... 79
Figure 5.9: Temperature perturbation due to flaw 1 in example 1 ... 79
Figure 5.10: Temperature perturbation due to flaw 2 in example 1 ... 80
Figure 5.11: A comparison of calculated and simulated temperature perturbations for example 1 ................................................. 80
Figure 5.12: Geometry of the two flaws in the domain for example 2 ... 81
Figure 5.13: Temperature perturbation due to flaw 1 in example 2 ... 82
Figure 5.14: Temperature perturbation due to flaw 2 in example 2 82
Figure 5.15: A comparison of calculated and simulated temperature perturbations for example 2 83
Figure 5.16: Geometry of the two flaws in the domain for example 3 84
Figure 5.17: Temperature perturbation due to flaw 1 in example 3 84
Figure 5.18: Temperature perturbation due to flaw 2 in example 3 85
Figure 5.19: Temperature perturbation due to flaw 3 in example 3 85
Figure 5.20: A comparison of calculated and simulated temperature perturbations for example 3 86
Figure 5.21: Geometry of the two flaws in the domain for example 4 87
Figure 5.22: Temperature perturbation due to flaw 1 in example 4 87
Figure 5.23: Temperature perturbation due to flaw 2 in example 4 88
Figure 5.24: Temperature perturbation due to flaw 3 in example 4 88
Figure 5.25: A comparison of calculated and simulated temperature perturbations for example 4 89
Figure 5.26: Geometry of the two flaws in the domain for example 5 90
Figure 5.27: Temperature perturbation due to flaw 1 in example 5 90
Figure 5.28: Temperature perturbation due to flaw 2 in example 5 91
Figure 5.29: Temperature perturbation due to flaw 3 in example 5 91
Figure 5.30: A comparison of calculated and simulated temperature perturbations for example 5 92
Figure 5.31: The real flaw (solid lines) and the predicted flaw (dot-dot line) for example 1 95
Figure 5.32: The real flaws (solid lines) and the predicted flaws (dot-dot line) for example 2 ........................................ 96
Figure 5.33: The real flaws (solid lines) and the predicted flaws (dot-dot line) for example 3 ........................................ 96
Figure 6.1: The real flaws in the 2X2 square domain for example 1 ...... 100
Figure 6.2: Predicted flaws using an exhaustive search method for exam­ple 1 ................................................................. 101
Figure 6.3: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 1 .................... 101
Figure 6.4: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 1 ................. 102
Figure 6.5: The real flaws in the 2X2 square domain for example 2 ...... 103
Figure 6.6: Predicted flaws using the exhaustive search method for exam­ple 2 ................................................................. 103
Figure 6.7: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 2 .................... 104
Figure 6.8: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 2 .................... 104
Figure 6.9: The real flaws in the 2X2 square domain for example 3 ...... 105
Figure 6.10: Predicted flaws using the exhaustive search method for exam­ple 3 ............................................................... 106
Figure 6.11: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 3 .................... 106
Figure 6.12: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 3  
Figure 6.13: The real flaws in the 2X2 square domain for example 4  
Figure 6.14: Predicted flaws using the exhaustive search method for example 4  
Figure 6.15: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 4  
Figure 6.16: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 4  
Figure 6.17: The real flaws in the 2X2 square domain for example 5  
Figure 6.18: Predicted flaws using the exhaustive search method for example 5  
Figure 6.19: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 5  
Figure 6.20: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 5  
Figure 6.21: The real flaws in the 2X2 square domain for Example 6  
Figure 6.22: Predicted flaws using the exhaustive search method for Example 6  
Figure 6.23: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 6  
Figure 6.24: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 6
ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my major professors, Dr. Ambar K. Mitra, and Dr. Suresh Kothari for their guidance, support, and patience which have made this research a valuable experience. Dr. Mitra's kindly attitude towards students allowed me to think freely and try new techniques in this research. I would like to thank Dr. Kothari for providing the MARS program and his helpful questions and comments during our discussions of this research.

I also extend my thanks to professor Thomas J. Rudolphi for his encouragement and understanding which made my graduate studies in Engineering Mechanics more memorable and meaningful. I am also deeply indebted to the rest of my graduate committee: Dr. Paul Sacks, Dr. William Lord, and Dr. Vinay Dayal.

Appreciation is also extended to my friends in Ames, especially my former office mates Jason Ting, Elias Teodore, and Paul Holst for their enjoyable discussions, picnics, and best of all, their friendship.

Finally, I want to express my deepest appreciation to my wife, Maithili, and daughter, Archana, for their patience and cooperation during the course of this work. All I have accomplished wouldn't have been possible without the blessing and love from my parents.
CHAPTER 1. INTRODUCTION

In engineering applications, the structural integrity of materials is quite often evaluated using fracture mechanics. This evaluation requires information about the flaw geometry (location, size, and shape of the flaws). In this dissertation, we describe a method for characterization of multiple flaws in a material. The material is in a steady thermal field and temperature at certain location on the boundary of the material is known. These known temperatures are used to characterize the flaws.

Identification of multiple flaws is a difficult problem due to the interaction between the flaws. The existing methods are known to work only for the identification of a single flaw in a material. At the present stage, our algorithm characterizes the multiple flaws if the number of flaws is known.

The identification of flaws in the material leads to inverse problems that involve partial differential equations. Inverse problems are concerned with the determination of properties of some inaccessible regions from observations on the boundaries of that region.

In 1923, J. Hadamard [12] introduced the notion of a well-posed problem. Hadamard required that three conditions must be satisfied for a well-posed problem, which are (a) existence of a solution, (b) uniqueness of the solution, and (c) continuity of the input data. The problems that fail to meet these requirements were labeled as ill-posed
by mathematicians. Later, it became apparent that a large number of extremely im-
portant problems failed to satisfy one of Hadamard’s conditions (continuity of the
input data) and these problems were named as inverse problems.

The terms "direct problems" and "inverse problems" are used to describe physi-
cal systems. In a direct problem, all the information necessary for the solution is
given as model parameters. In an inverse problem, sufficient information is not pro-
vided and the estimation of model parameters from experimentally measured data is
required. In a matrix system, calculation of $Af$ with given $A$ and $f$ is called a direct
problem. It is possible to solve using the direct forward methods of mathematics.
The determination of $f$ or parameters of $A$ from the relation $Af = g$ with given $g$, is
an inverse problem. This can be summarized as follows:

Direct problem:

model parameters $\rightarrow$ model $\rightarrow$ prediction of data

Inverse problem:

data $\rightarrow$ model $\rightarrow$ estimation of model parameters.

This study employed the boundary element method (BEM) as a computational
tool. The primary advantage of the BEM is that for linear problems only the bound-
ary of the domain needs to be discretized. The BEM was initially applied in 1967 as
an integral equation approach by Rizzo [25] to boundary value problems of classical
elastostatics. In recent years, the boundary integral technique has attracted a sig-
nificant interest as a computational tool in the solution of computational mechanics
problems. However, not much work appeared in the literature on the solution of
inverse problems using the boundary integral approach. There are two specific ad-
vantages in applying the BEM to the present problem. Firstly, the characteristics of
the flaws appear directly in the integral equation through nodal coordinates on the flaws. Secondly, BEM does not deal with and yield any domain information that is redundant in the present situation.

Kassab and Pollard [13] used a technique called automated algorithm to identify subsurface cavities in a 2D material using steady state heat conduction. Their procedure begins with an initial guess for the inner boundary of a cavity and uses the boundary element method to determine the boundary temperatures and fluxes at each iterative step by solving a forward heat conduction problem. The Newton Raphson method was used to solve the nonlinear set of equations.

Das and Mitra [7] applied the boundary element method to solve the inverse Laplace problems in steady state heat conduction. By minimizing an error functional, Das and Mitra developed an iterative scheme to identify a single flaw in a 2D material. In this algorithm, the flaw is assumed to be an ellipse which uniquely defined by five quantities. In this present work, we extended Das and Mitra's algorithm to multiple flaws and employed another new method called multivariate adaptive regression splines (MARS) developed by Friedman [8].

The characterization of flaws is performed in two stages. First, the specimen probe data is compared with the training set to obtain the approximate locations and sizes of the assumed circular flaws. These approximate locations and sizes serve as the guessed flaws. This guessed flaw information is then passed on to an iteration algorithm, developed by Das and Mitra [7] or the MARS algorithm.

In extending Das and Mitra's method for multiple flaws, insertion of probe data in the functional minimization algorithm yielded unsatisfactory solution. Thus, it was decided that before minimizing the functional, one must determine the approximate
location of the flaws.

This determination of approximate location involves a matching of the probe data with a training set. Conceptually, this pre-determined training set should contain the probe data for 2,3,4, etc., flaws of various shape and size located at various possible locations over the domain. However, the size of such a training set will be enormous and the creation of such a training set will be prohibitively expensive.

In order to reduce the size of the training set, we include information of circular flaws only. After the approximate location of circular flaws is determined by comparing the specimen probe data with the training set, the shape of the flaws can be updated by employing Das and Mitra's scheme. Furthermore, the size of the training set is reduced by considering only one flaw instead of multiple flaws. The probe data for multiple flaws can be generated from the single flaw data through simple superposition. The length scale of the flaws is expected to be much smaller compared to the length scale of separation between the flaws. Thus, the disturbance caused by each flaw, in a multiple flaw situation, does not influence the disturbances caused by other flaws, but all the disturbances simply get added together. This argument is invalid when the multiple flaws are crowded together, but in that case they need not be identified individually and a single flaw calculation yields a single aggregate flaw. Next, we focus on the issue of the size of a flaw. In order to resolve this issue, we used an idea from potential flows in fluid mechanics. When a cylinder of radius $a$ is placed in a parallel flow with parallel equipotential lines, the disturbance caused by the cylinder is proportional to $a^2$. Thus, the probe data for a circular flaw of radius $ka$ can be obtained by multiplying the probe data for a circular flaw of radius $a$ by $k^2$. The distinguishing feature of this technique is that only a small training
set is stored in the memory. The enlarged training set is not stored, but is internally created as and when necessary.

We also employed another reliable method called MARS to the identification of the multiple flaws. MARS, developed by Jerome Friedman of Stanford University, is a method for adaptive regression modeling to handle multi-dimensional data. The model takes the form of an expansion of product spline basis functions. The number of basis functions as well as the parameters associated with each one are automatically determined by the algorithm. This method produces continuous functions with continuous derivatives to given data. The multivariate adaptive method is common in many disciplines, and frequently it is used in applied mathematics (non parametric function application), statistics (non parametric multiple regression) and in computer science and engineering (neural networks).

The objective of MARS is to model the dependence of a response variable \( y \) on one or more predictor variables \( x_1, x_2, ..., x_n \), given realization, \( \{y_i, x_1, x_2, ..., x_n\}_1 \). The system that generated the data is presumed to be described by

\[
y = f(x_1, x_2, ..., x_n) + \epsilon
\]

over some domain \( D \subset \mathbb{R}^n \) containing the data. The single valued deterministic function \( f \), captures the joint predictive relationship of \( y \) on \( x_1, x_2, ..., x_n \). The additive stochastic components \( \epsilon \), whose expected value is defined to be zero, usually reflects the dependence of \( y \) on quantities other than \( x_1, x_2, ..., x_n \). The aim of regression analysis is to use the data to create a function \( \hat{f}(x_1, x_2, ..., x_n) \) over the domain \( D \) of interest. More information on MARS can be found in Friedman [26, 27] and an application to quantitative non-destructive evaluations (QNDE) dealing
with sizing and classifying flaws in materials is given in [52].

In Chapter 2, the mathematical formulation of the boundary element method is given for the potential problems. We also present a technique to identify the unknown boundary of a flaw in the domain using some measurement on the outer boundary of the domain. In this method, the temperature difference on the boundary due to an actual flaw and a guessed flaw in the domain is used to identify the actual flaw's location and shape. The algorithm described is based on the boundary element method coupled with a non-linear optimization technique. Finally, three examples are also given to test the above iteration method.

In Chapter 3, the MARS is described. The formulation of an approximator \( \hat{f} \) (a linear expansion) and the automatic formulation of the basis functions are also given. The forward and backward stepwise MARS algorithms are given explicitly. Many features of MARS are included for computational reasons. The performance of MARS is analyzed for several test cases. Three examples are solved completely by the two-step MARS process and the error analysis is also given. Finally, the MARS method is applied to some irregular shaped flaws and six examples are given.

In Chapter 4, the tolerance of the methods to experimental noise is analyzed. By adding uniform noise (error) in the measurements, data with experimental noises are generated. The location and size for the data are calculated with experimental noises and compared with the results having no experimental noises. Finally, the difficulties encountered in the identification of multiple flaws in a domain are identified.

In Chapter 5, a method is developed to find the approximate locations and shapes after studying the patterns of temperature perturbation at the insulated boundary of the domain. The relationship between the flaw size and shape to the temperature
perturbation can be discovered for any flaw location in the domain. The concept of linearity of the partial differential equation is utilized to construct the temperature changes using a small sample of data set of single flaw cases. An exhaustive search is performed to identify the approximate location and size of multiple flaws. Three examples are given to justify the method. Finally, an improvement is noticed in the number of iterations in the single flaw identification using the knowledge of the approximate location and size of the flaw.

In Chapter 6, six examples are given for multiple flaws identification. Different cases of two flaws and three flaws in the domain are considered. The predictions of the methods are given and compared against the actual flaws. The limitations of the methods are discussed.

In Chapter 7, the conclusion of the dissertation is given and some recommendation for future work are also made.
CHAPTER 2. THEORY AND APPLICATION OF THE BOUNDARY ELEMENT METHOD IN FLAW IDENTIFICATION

The Problem

Consider a problem of steady state heat conduction in a two-dimensional domain $D$ as shown in Fig. 2.1. Some parts ($S1N$) of the boundary are insulated, and the rest ($S1D$) is kept at a constant temperature distribution.

The temperature in this steady state heat conduction problem satisfies the Laplace equation. Temperature at $S1N$ is obtained experimentally at several selected points. If the domain has any flaw(s) at $S2$, as shown in Fig. 2.2, the measured temperature will be different from that of a domain without any flaw; that is, the perturbation on the temperature at the insulated boundary indicates the existence of the flaw(s). Specified temperature and flux conditions were used on $S1N + S1D$, and additional experimental information is used to determine the location and shape of the flaw(s).

Once it is known that flaws exist in the domain of the problem, changes are made from a simply connected to a multiply connected domain. The inner boundary condition on $S2$ is known to be the homogeneous Neumann condition. This corresponds to the physical situation of a flaw with very small conductivity. Our aim is to determine $S2$ from the measured temperature data on $S1N$. 
Formulation of the Boundary Integral Equation

Consider the Laplace problem for the domain shown in Fig. 2.1, as

\[ \nabla^2 T = 0 \quad \text{in} \quad D, \]  
\[ (2.1) \]

with the boundary conditions

\[ \frac{\partial T(q)}{\partial n} = 0 \quad \text{on} \quad S1N \]
\[ T(q) = T_0(q) \quad \text{on} \quad S1D \]  
\[ (2.2) \]

where \( S1N \) and \( S1D \) are Neumann and Dirichlet boundary, respectively, and \( n \) is the unit outward normal.

Figure 2.1: A domain with Neumann and Dirichlet boundaries
In addition, the condition

$$\frac{\partial T(q)}{\partial n} = 0 \text{ on } S_2, \quad (2.3)$$

is specified on the boundary of the flaw.

The differential equation (2.1) can be converted to an integral equation. After inserting the boundary conditions this integral equation becomes

$$C(p)T(p) = \int_{S_1D} (G'T_o(q) - GT'(q))ds + \int_{S_1N} G'T(q)ds + \int_{S_2} G'T(q)ds \quad (2.4)$$

where:

$$C(p) = \begin{cases} 2\pi & p \in D \\ \alpha & p \in S \\ 0 & p \text{ not in } D \text{ or } S, \end{cases}$$

where $\alpha$ is the included angle between two adjacent tangent at $p$. The function $G$ is the fundamental solution of the Laplace equation and the prime denotes normal derivative. The details of the procedure for converting the partial differential equation into the integral equation can be found in any standard text [10] on the BEM.

Now, by substituting $T = T_2, \ T' = T_2'$ and rearranging the above equation, one gets

$$C(p)T_2(p) - \int_{S_1N} (G'T_2(q)ds = \int_{S_1D} (G'T_o(q) - GT_2'(q))ds + \int_{S_2} G'T_2(q)ds.$$  

(2.5)
Now, consider a flaw boundary $S_3$ in $D$ as shown in Fig. 2.2, instead of the actual flaw boundary $S_2$, with the following boundary conditions:

\[
\begin{align*}
T(q) &= T_0(q), \quad q \in S_{1D} \\
\frac{\partial}{\partial n} T(q) &= 0, \quad q \in S_{1N} \\
\frac{\partial}{\partial n} T(q) &= 0, \quad q \in S_3.
\end{align*}
\tag{2.6}
\]

For the guessed flaw situation, the integral equation becomes

\[
C(p)T_3(p) - \int_{S_{1N}} (G'T_3(q)) ds = \int_{S_{1D}} (G'T_0(q) - GT_3'(q)) ds + \int_{S_3} G'T_3(q) ds
\tag{2.7}
\]

If one substitutes $S_2 = S_3$, $T_2(q) = T_3(q)$, and $T_2'(q) = T_3'(q)$ in all the terms on the right hand side of Eq. (2.5), the equation will not be satisfied. Denoting the residual by $\delta(p)$ after such substitution we rewrite Eq. (2.5) as
\[ C(p)T_2(p) - \int_{S_1N} (G'T_2(q))ds = \int_{S_1D} (G'T_0(q)-G'T_3'(q))ds + \int_{S_2} G'T_3(q)ds + \delta(p) \]

(2.8)

Subtracting Eq. (2.7) from Eq. (2.8), one gets

\[ C(p_n)(T_2(p_n) - T_3(p_n)) + \int_{S_1N} G'T_3(q)ds - \int_{S_1N} G'T_2(q)ds = \delta(p_n). \]

(2.9)

where \( n = 1, 2, ..., N \). The position \( p_n \) denotes the location of \( N \) number of probes.

The temperature \( T_2(p_n) \) is assumed to be known from experiments for the real flaw at number of selected points \( p_1, p_2, ..., p_n \). The each term in the left hand side of Eq. (2.9) can be determined as follows:

**Term1:** \( T_2(p_n) \) is obtained from experiments; and \( T_3(p_n) \) is obtained from the numerical solution for the known guessed flaw.

**Term2:** \( T_3(q) \) at \( S1N \) can be obtained numerically for the guessed flaw boundary \( S_3 \).

**Term3:** \( T_2(q) \) is obtained from the experimentally measured data \( T_2(p_n) \), on \( S1N \) by interpolation.

After evaluating \( N \) quantities \( [\delta_n, n = 1, 2, ..., N] \) for the \( N \) number of selected points of \( p_n \), one can define a functional,

\[ F = \sum_{n=1}^{N} \delta_n^2. \]

(2.10)

When the functional \( F \) is minimized, the residual \( \delta_n \) approaches zero in a least square sense. One can see from Eq. (2.9) that as \( \delta(p_n) \) approaches zero, \( S_3 \) ap-
proaches $S_2$ of in the other words the guessed flaw approaches the real flaw. During
the minimization process, the boundary $S_3$ and the $T_3(q)$ are modified, and at the
end $S_3$ converges to $S_2$. The details of the minimization process is given in the next
section.

**The Minimization of the Functional**

The real unknown flaw, $S_2$, could be of any general shape. A very irregular
shaped flaw is constructed by using only straight elements, which will require very
fine discretisation. The coordinates of the ends points of such elements will then be
the ultimate unknown in the problem. For each additional element, the size of the
problem is increased by two unknowns which are the coordinates of the element’s end
point of the added element. This procedure requires large computational time and
many nodal coordinates.

In some cases, the boundary $S_3$ wraps around itself and finally leads to an
unrealistic solution. In order to avoid such an unrealistic solution, all flaws are
assumed to be elliptic. The assumption that all flaws are elliptic reduces the degrees
of freedom, and thereby reduces the computational time. The possibility of unrealistic
solutions is also avoided by restraining the movement of nodal points. Further, the
elliptic shape covers a wide ranges of shapes from circular holes to straight cracks.
The assumption regarding elliptic flaws is found to work well for a flaw which is
very much different from an elliptic. The algorithm yields excellent estimates for the
location, and the linear dimensions of the flaw.

The semi-major axis $(a)$, the semi-minor axis $(b)$, location of the center $(x_c,y_c)$,
and the angle of orientation($\phi$) of the major axis with a fixed global direction are
considered as the ultimate unknowns of the problem. The degrees of freedom is 5. Actually, one must obtain \( T_2(p_n) \) from experiments; however, for the need of any experimental data, a Laplace solver, based on boundary integral equation method was used to compute \( T_2(p_n) \).

Minimization process

At the beginning of the iteration process an initial guess of \( S_3 \) is chosen, and for this guessed flaw, the \( T_3(p) \) is calculated by using the Laplace solver. A subroutine named LSFCN1 calculates for the functional by using the description of \( S_3 \) from Eqns. (2.9) and (2.10). The NAG library subroutine E04FDF was used for minimization.

The subroutine E04FDF is an algorithm one can use easily for finding an unconstrained minimum of the sum of squares of \( m \) nonlinear functions in \( n \) variables \((m \leq n)\) where derivatives are not required. It is intended for functions which are continuous and also have continuous first and second derivatives, although it will usually work even if the derivatives have occasional discontinuities.

This subroutine uses the functional calculated by LSFCN1, and updates the 5 degrees of freedom that define \( S_3 \). The NAG subroutine calculates the derivatives of the functional internally, and the user does not need to furnish expressions for these. The updated \( S_3 \) is then used by the Laplace solver to update \( T_3(q) \). The functional is then calculated again by LSFCN1, and the process continues until convergence.

The number of iterations required depends on the number of variables, the number of residuals and their behavior, and the difference between the starting point and the solution. The number of multiplications performed per iteration of E04FDF varies, but for \( m \gg n \) it is approximately \( nm^3 + O(n^3) \). In addition, each iteration
makes at least \( n + 1 \) call of LSFCN1. Unless the residuals can be evaluated very quickly, the run time will be dominated by the time spent in LSFCN1. Although the minimization process outlined above involves repeated solution of discretized integral equations, the computational cost is reasonable since the convergence is fast.

**Spline shape of the flaw**

In order to predict the contour of irregular shaped flaws which are very much different than an ellipse Mitra and Das [41] developed a six-noded periodic B-Spline shape, after minimized by an elliptical shape. The assumption, that all the flaws are elliptic, accelerates the convergence of the minimization process and does not allow for any unrealistic solution. After an approximate elliptical solution obtained through above minimization process, one can remove the elliptical constraints and continue with a second stage of minimization with a six noded periodic B-Spline, see Foley [25].

The six-noded B-Spline curve is completely defined by the coordinates of six points \( P_0, P_1, P_2, \ldots, P_5 \). The B-Spline is closed by repeating the first three nodes at the end of the sequence. Thus, the complete sequence is between \( P_0, P_1, P_2, \ldots, P_5, P_0, P_1, P_2 \). Six sequents of cubic curves \( Q_3, Q_4, \ldots, Q_8 \) are now drawn from this sequence of nine points, and each segment \( Q_i \) controlled by four nodes. For examples, \( Q_3 \) is a cubic curve controlled by points \( P_0, P_1, P_2, \) and \( P_3 \), where \( Q_8 \) is controlled by points \( P_5, P_0, P_1, \) and \( P_2 \).

The equation of the \( i^{th} \) segment of the cubic spline is

\[
Q_i = T_i \cdot M_{BS} \cdot G_{BS_i},
\] (2.11)
where,

\[ T_i = \left[ (t - t_i)^3 (t - t_i)^2 (t - t_i) 1 \right], t_i \leq t \leq t_{i+1}. \]

\[ M_{BS} = \frac{1}{6} \begin{pmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{pmatrix}, \]

\[ G_{BS_i} = \begin{bmatrix} P_{i-3} \\ P_{i-2} \\ P_{i-1} \\ P_i \end{bmatrix}, \]

and \( t \) is the length measured along the perimeter of the closed curve.

After the elliptic approximation of a predicted flaw is obtained through the first stage of minimization, the six-nodes for the periodic B-Spline are placed on this ellipse. Then the second stage of minimization starts with 12 degrees of freedom. Some results are shown in the next section as examples 2 and 3.

**Applications**

In order to test the minimization technique and the iteration process, an elliptical flaw and two irregularly shaped flaws are given. First, each flaw is minimized to an elliptical flaw from a circular guessed flaw at the center. Secondly, the minimization process is continued with a six-noded periodic B-Spline shape, after removing the elliptical constraints.

The exterior boundary of the domain is a 2 X 2 square in all three example. The origin of the Cartesian axis system is located at the center of the square with the
x-axis horizontal, and the y-axis vertical. The vertical sides of the square are held at temperatures 400° and 50°. The other two sides are insulated. The computation in all the cases begin with a guessed flaw in the shape of a circle of radius 0.1, which is located at the center of the domain. Thirty two linear elements on the exterior boundary of the domain and 12 elements on the boundary of the guessed flaw were used, for the formulation of the boundary element discretization.

The number of probes for the experimental measurement was fixed at N=12. One-half of the probes were placed uniformly on the top and the other half at the bottom of the square domain. In this approximation, N should be more than the degrees of freedom. In a single flaw case, the degrees of freedom in the elliptical approximation are 5. The number of probes kept at 30, for the spline approximation. As before, one-half of the probes were placed uniformly at the top and other half at the bottom. In this case, N should be more than 12, since a six-noded spline approximation has 12 degrees of freedom. The calculations were performed using a DECstation 5000/25.

Example 1: In this example, the actual flaw is an ellipse given by (a = 0.1, b = 0.025, xc = 0.5, yc = 0.4, φ = $\frac{3\pi}{4}$). The real and guessed flaw in the square domain are shown in Fig. 2.3. The predicted flaw at different stages of iteration are shown in Fig. 2.4. After 110 iterations, the guessed flaw coincided with the actual flaw. At this stage the functional value is F = 6.59E-9. The variation among the values of the functional with the number of iterations is shown in Fig. 2.5. Although, after 110 iterations, there was no significant improvement in the shape of the guessed flaw, the functional value became smaller and smaller until the program stopped. The spline approximation did not add further improvement to the shape of the flaw and
came to a stop after 5 iterations, since the actual flaw is an ellipse and the elliptical approximation identify the exact shape without any problem.

Example 2: In this example, the actual flaw is a crack bent in an arc-like shape. The approximate ratio of the length, average width, and area of the flaw and the domain are 5.0E-2, 5.0E-3, and 4.5E-4, respectively. The actual flaw and the initial guessed flaw are shown in Fig. 2.6. The elliptical approximation minimized the functional value to 7.03E-7 after 100 iterations. The predicted elliptical shape of the flaw and the actual flaw are shown in Fig. 2.7. By taking six nodes from the predicted elliptical shape, the minimization continued with the spline approximation, which reduced the functional values (F) to 9.21E-4 after 60 iterations. The predicted spline approximation is shown in Fig. 2.8.

Figure 2.3: An elliptical real flaw (solid line) and a circular guessed flaw (dot-dot line) in a 2X2 square domain for example 1
Figure 2.4: Predicted flaws (dot-dot lines) at different stages of iteration for example 1.

Figure 2.5: Variation of the value of the functional with the number of iterations for example 1.
Figure 2.6: The real flaw (solid line) and a circular guessed flaw (dot-dot line) for example 2

Figure 2.7: Predicted elliptical shaped flaw (dot-dot line) and the real flaw (solid line) for example 2
Example 3: In this example, the actual flaw is a bug-shaped figure located in the second quadrant of the square domain. The actual flaw and the guessed flaw are shown in Fig. 2.9. The approximate ratios of length, average width, and area of the flaw and the domain are 5.0E-2, 3.5E-2, and 1.114E-3, respectively. The elliptical approximation minimized the functional values to 1.31E-9 after 60 iteration. The predicted elliptical shape and the actual flaw are shown in Fig. 2.10. After this stage, the spline approximation minimized the functional value to 6.92E-6 in 60 iterations. The actual flaw and the predicted spline shape are shown in Fig. 2.11.
Figure 2.9: The real flaw (solid line) and a circular guessed flaw (dot-dot line) for example 3

Figure 2.10: Predicted elliptical shaped flaw (dot-dot line) and the real flaw (solid line) for example 3
Extension of the single flaw algorithm to multiple flaw situation is straightforward. Each of the multiple flaws is assumed to be elliptic. Thus, five degrees of freedom are associated with each flaw. Spline approximation technique should also work in the multiple flaw situation, but is not attempted here. However, this simple extension of the algorithm to multiple flaws encounters certain computational difficulties and their remedy will be postponed until Chapter 4 and 5.

Discussion

After testing many examples, it was determined that the number of iterations needed to approximate a flaw depends on the difference between flaw location from the guessed flaw, the shape of the flaw, and the number of probes used and their locations. The tolerance due to experimental noise is analyzed in Chapter 4. It was
also noticed that the number of elements on the boundary of the domain is a factor in the number of iterations. When the number of outer boundary elements were changed from 8 to 32 and the guessed flaw boundary from 8 to 12, the number of iterations were reduced from 81 to 62 for an elliptical flaw given by \( a = 0.05, b = 0.025, xc = 0.0, yc = 0.0, \phi = 0.0 \).

Finally, the performance of a simple algorithm for flaw identification in a two dimensional materials was demonstrated by employing a temperature field. In the beginning stages of minimization, it was necessary to restrict the distortion of the guessed flaw by assuming that all flaws are elliptically shaped. After an approximate solution was obtained, the assumption of elliptic flaw could be relaxed and a periodic B-Spline could be applied. The periodic B-Spline approximation has the power to interpret the complicated odd shapes of the actual flaw.
CHAPTER 3. MULTIVARIATE ADAPTIVE REGRESSION
SPLINES AND THEIR USE FOR FLAW IDENTIFICATION

Development of a new method to predict the flaw locations and sizes in a material is given in this chapter. Nonlinear functional approximation has been studied in applied mathematics and statistics as regression analysis for many years. A new method called multivariate adaptive splines (MARS) has been developed by Jerome Friedman [26, 27] to predict the responses from the multi-dimensional input data. This method builds a functional expansion in product spline basis function. The MARS algorithm accepts a set of input and output data as training set and the maximum number of the basis function to construct a model, then accepts a set of input data as a testing set and predicts the output data. The features of the MARS algorithm, selection of the training set, selection of the testing set, choosing the maximum number of basis functions and application of MARS in flaw identification are given in the following sections.

MARS Algorithm

MARS is an adaptive method for modeling an approximation of a function that fits the given training data. The MARS approximator \( \hat{f} \) is a linear expansion of the form.
\[ \hat{f} = \sum_{m=1}^{M} a_mB_m(\tilde{x}), \]  

(3.1)

where each \( B_m \) is called a basis function and it will be described later. The \( a_m(m = 1, 2, ..., M) \) are coefficients of the linear combination. The MARS attempts to give the best fit to the training set data by dynamically changing the set for the basis functions and the coefficients. The number of the basis function \( M \), and the exact form of each basis function \( B_m \) are determined by the method. The number of input dimension and the number of elements in the training set depends on the problem.

The MARS algorithm contains a forward stepwise algorithm and a backward stepwise algorithm. The forward stepwise algorithm builds the model by formulating the basis functions according to the training data. The backward stepwise algorithm eliminates basis functions that do not help the overall fit.

**Forward Stepwise Algorithm**

The forward stepwise MARS algorithm is described in Fig. 3.1. Line 1 of the algorithm initializes the model to respond with a value of 1 for all inputs. Line 2 assigns 2 for the number of basis functions. In line 3, the WHILE-loop iterates \( M \), the number of basis functions used by the model from 2 up to the maximum number of basis functions \( M_{max} \) of which an input of the algorithm. Each iteration through this loop adds two more basis functions \( B_m \) and \( B_{m+1} \) by splitting an existing basis function \( B_m* \) on dimension \( v* \) at value \( t* \). The notion of "splitting" a basis function is accomplished in line 19 and 20 by multiplying the existing basis function \( B_m \) by itself times each side of the first order truncated power representation of a spline function. The two-sided truncated power basis functions for representing \( q \)th order
1. \( B_1(\bar{x}) \leftarrow 1 \)
2. \( M \leftarrow 2 \)
3. WHILE \( M < M_{\text{max}} \) DO
4. \( \text{lof}^* \leftarrow \infty \)
5. FOR \( m=1 \) TO \( M-1 \) DO
6. FOR \( v \in \{v(k,m) \mid 1 \leq k \leq K_m\} \) DO
7. FOR \( t \in \{x_v \mid B_m(\bar{x}_j) > 0\} \) DO
8. \( g \leftarrow \sum_{i=1}^{M-1} a_i B_i(\bar{x}) + a_M B_m(\bar{x}) [x_v - t]_+ + a_{M+1} B_{M+1}(\bar{x}) [t - x_v]_+ \)
9. \( \text{lof} \leftarrow \text{min}_1 a_M \ldots a_{M+1} \text{LOF}(g) \)
10. IF \( \text{lof} < \text{lof}^* \) THEN
11. \( \text{lof}^* \leftarrow \text{lof} \)
12. \( m^* \leftarrow m \)
13. \( v^* \leftarrow v \)
14. \( t^* \leftarrow t \)
15. ENDIF
16. END FOR \( t \)
17. END FOR \( v \)
18. END FOR \( m \)
19. \( B_M(\bar{x}) \leftarrow B_{m^*}(\bar{x}) [x_{v^*} - t^*]_+ \)
20. \( B_{M+1}(\bar{x}) \leftarrow B_{m^*}(\bar{x}) [t^* - x_{v^*}]_+ \)
21. \( M \leftarrow M + 2 \)
22. END WHILE

Figure 3.1: The forward stepwise MARS algorithm
splines are:

\[ b(x - t) = [x - t]^q_+ \]  \hspace{1cm} (3.2)

\[ b(t - x) = -[x - t]^q_+ , \]  \hspace{1cm} (3.3)

where \( t \) is a knot location "splitting point"., and \( q \) is the spline order of the spline and the subscript indicates the positive part of the argument. For \( q > 0 \), the spline approximation is continuous and has \( q - 1 \) continuous derivatives. The choice of continuous spline basis functions produces continuous models.

The choice of \( m^*, v^*, \) and \( t^* \) is done at Fig. 3.1, by iterating over all currently existing basis functions (of which there are \( M - 1 \)), the dimensions and the data values \( t \) that satisfies the following criteria: (1) \( t \) be equal to a value of the \( v^{th} \) dimension of some vector \( j \) from the training set. The dimension \( v \) is set by the surrounding FOR-loop (line 6) and \( j \) ranges from 1 to \( N \), where \( N \) is the size of the training set; and (2) The current basis function under careful inspection for possible splitting \( (B_m \) with \( m \) set by line 5) must return a positive output when applied to the \( j^{th} \) input vector found in the first criterion.

The split points are chosen to satisfy the above criteria. After the candidate values for \( m^*, v^*, \) and \( t^* \), are chosen, the heart of the MARS algorithm is described in line 8 through 15 of Fig. 3.1. A new model \( g \) by adding two more basis functions to the current model by splitting candidate basis function \( B_m \) on dimension \( v \) at data point \( t \) in line 8. Model \( g \) is then evaluated in line 9 using the criterion \( \text{LOF} \) that gives a measure of the lack-of-fit of \( g \) to the training set data. The \( \text{LOF} \) function is a modified version of the generalized cross validation criterion given in Crawen.
and Wahba [16]. More details of the MARS LOF functions are given in Friedman [27]. In general, the MARS LOF($g$) criterion is the average squared error of the fit of the model $g$ to the training set, multiplied by a penalty function. A parameter $d$ is associated with the penalty function. The value $d$, which can be regulated by the user, according to the size of the training set. The algorithm finishes with a model consisting of $M_{max}$ basis functions, where each function has the form:

$$B_m(\tilde{x}) = \prod_{k=1}^{K_m} \left( S_{K_m} \cdot \left[x_{v(k,m)} - t_{km}\right]_+ \right). \quad (3.4)$$

The MARS produces basis functions that are products of $K_m$ first order spline functions as shown in Eq. (3.4). Since each spline function resulted from a "split", the quantity $K_m$ can also be viewed as the number of splits that were required to produce basis function $B_m$. Each split is parameterized by the arguments of the spline function associated with the split. The sign of the argument is given by $S_{km}$ (either positive or negative), and $v(k,m)$ specifies the input dimension on which split $K$ occurred for basis function $B_m$. The split value used (from the training set) for the split $K$ of basis function $B_m$ is given by $t_{km}$.

**Backward Stepwise Procedure**

The backward stepwise procedure is often required to eliminate basis functions that do not contribute to the overall fit. The backward stepwise MARS algorithm gives in Fig. 3.2 takes the resultant model from the forward stepwise procedure and eliminates one basis function at a time. The effect is to search for the best model using the LOF criterion as a judge of model quality.

The set of basis functions that should be included in the final model is tracked...
in variables $J^*$. Thus, line 1 initializes the final model as the entire basis function set that was obtained from the forward MARS procedure. The outer FOR-loop of line 4 repeatedly builds the best model with $M$ basis functions, where $M$ ranges from $M_{\text{max}}$ to 2. The inner FOR-loop builds multiple models by removing one basis function from the current set of basis functions given in $L$. Each model is compared with all the others and the best model of size $M$ is saved in $K^*$ for use by the next iteration of the outer FOR-loop. Variable $J^*$ is updated such that the best model found any size less than or equal to $M_{\text{max}}$ is saved. This process is shown in lines 8 through 17 of Fig. 3.2.

The MARS has several parameters which may be set by the user during the execution. The experiments reported in this dissertation, the maximum number of the basis function $M_{\text{max}}$, and the parameter $d$, are set by the user each application according to the size of the training set. All the other parameters were left at their default locations. The range of $d$ is between 2 and 3, a larger size training set needs a higher value of $d$ than a smaller size training set. The run time of the MARS is linear in the number of the input dimensions and the size of the training set, and are approximately cubic in $M_{\text{max}}$. Obviously, the $M_{\text{max}}$ turns out to be the important parameter. The lower the setting of $M_{\text{max}}$, the faster the algorithm will execute; however, if the $M_{\text{max}}$ is very small, there will be less accuracy in the result. The selection of the number of maximum basis functions will be discussed later.

**Selection of the Training Set**

The user have to select a training set which cover the entire range of the prediction values. The size of the training set and the number of input variables are
1. $J^* = (1, 2, \ldots, M_{\text{max}})$
2. $K^* \leftarrow J^*$
3. $\text{lof}^* \leftarrow \min_{j \in J^*} \text{LOF} \left( \sum_{j \in J^*} a_j B_j(\tilde{x}) \right)$
4. FOR $M = M_{\text{max}}$ TO 2 DO
5. \hspace{1cm} $b \leftarrow \infty$
6. \hspace{1cm} $L \leftarrow K^*$
7. \hspace{1cm} FOR $m = 2$ to $M$ DO
8. \hspace{2cm} $K \leftarrow L - m$
9. \hspace{2cm} $\text{lof} \leftarrow \min_{k \in K} \text{LOF} \left( \sum_{k \in K} a_k B_k(\tilde{x}) \right)$
10. \hspace{2cm} IF $\text{lof} < b$ THEN
11. \hspace{3cm} $b \leftarrow \text{lof}$
12. \hspace{3cm} $K^* \leftarrow K$
13. \hspace{2cm} ENDIF
14. \hspace{2cm} IF $\text{lof} < \text{lof}^*$ THEN
15. \hspace{3cm} $\text{lof}^* \leftarrow \text{lof}$
16. \hspace{3cm} $J^* \leftarrow K$
17. \hspace{2cm} ENDIF
18. \hspace{1cm} END FOR $m$
19. \hspace{1cm} END FOR $M$

**Figure 3.2:** The backward stepwise MAkS algorithm
depend on the problem of interest. Identification of flaws in a material by employing a thermal field is a difficult nonlinear inverse problem. The location, shape, and size of the flaw(s) have to be identified based on the temperature perturbation at the boundary.

In order to select a training set, in a 2 X 2 square domain, 100 locations is chosen uniformly as flaw locations. In this selection the difference between the flaws is 0.2. Then three different sizes of circular flaws were taken with radii of 0.03, 0.06, and 0.09, and four different sizes of ellipses were taken. The ellipses are given by:

\[(a = 0.08, b = 0.04, \text{ and } \alpha = \pi/4),\]
\[(a = 0.08, b = 0.04, \text{ and } \alpha = 3\pi/4),\]
\[(a = 0.05, b = 0.025, \text{ and } \alpha = \pi/4), \text{ and}\]
\[(a = 0.05, b = 0.025, \text{ and } \alpha = 3\pi/4),\]

where \(a\)-semi major axis, \(b\)-semi minor axis and \(\alpha\) is the angle of orientation of the semi-major axis with x-axis. The above selection of circles and ellipses has a variation in sizes and shapes to predict any flaw, which length is less than 0.2 and with any orientation.

As described above, a total of 700 different flaws were chosen, with 100 different locations and 7 different sizes and shapes. The temperature perturbation was calculated at 20 points at the insulated boundary of the domain by placing one flaw at a time in the domain. The probes data and the flaw descriptions are stored in a data file for every flaw to create a 700 element training set. A 700 element training set is a moderate size for an input dimension of 20 variables for a \(\alpha\)-DECstation computer which is used in this work. The user set the maximum number of basis functions, \(M_{max}\). In order to predict the value of \(M_{max}\), a testing set is selected as follows.
Selection of the Testing Set

In the selection of a testing set, 361 locations are selected evenly in the $2 \times 2$ domain. Then, a circle with radius of 0.5, and six different sizes of ellipses are chosen to have many different kind of flaws. The semi-major, semi-minor, and the orientation of the ellipses are given by:

- $(a = 0.05, b = 0.025, \text{and } \alpha = 0.0)$.
- $(a = 0.05, b = 0.025, \text{and } \alpha = \pi/4)$.
- $(a = 0.05, b = 0.025, \text{and } \alpha = \pi/2)$.
- $(a = 0.075, b = 0.05, \text{and } \alpha = 0.0)$.
- $(a = 0.075, b = 0.05, \text{and } \alpha = \pi/4)$, and
- $(a = 0.075, b = 0.05, \text{and } \alpha = \pi/2)$.

By placing this seven different sizes and shapes of flaws at the 361 locations one at a time, one can create a 2527 element testing set, using the calculated temperature perturbation at the boundary. The MARS model reads the temperature perturbations at boundary and predicts the flaw descriptions, $(a, b, xc, yc, \alpha)$. The performance of MARS depend on the choice of $M_{max}$ for the selected training set.

Choosing the Maximum Number of the Basis Function

In the process of testing, the mean absolute error and standard deviation were calculated in the evaluation of $a, b, xc, yc$ as a function of maximum number of the basis functions to choose the $M_{max}$ for the 2527 element testing set. The absolute mean error and standard deviation as a function of the maximum number of basis function are shown in Figs. 3.3, 3.4, 3.5, and 3.6 for the evaluation of $a, b, xc, yc$. The
prediction of $\alpha$ is not considered, because a circular flaw of orientation zero degree and ninety degrees gives the same perturbation. That leads to more error in the absolute mean in the predictions of $\alpha$ by MARS.

There was no improvement in the accuracy in the approximation when the $M_{\text{max}}$ got larger than 30 in the prediction of the elliptical parameters. The average error in $a$ or $b$ is less than 0.01, and in $x_c$ or $y_c$ it was less than 0.1 for $M_{\text{max}} \geq 30$. One can select 30 as an optimal value of $M_{\text{max}}$ for the training set used in this test. The maximum number of basis functions required seems to be related to the size of the training set. A small training set required only a smaller number of basis functions.

The run time for MARS and the average error in the prediction of a flaw are relatively less compared to existing neural networks methods found in the literature. Tables 3.1, 3.2, 3.3, and 3.4 show the time taken to execute MARS and the percentage of flaws identified for the given tolerance in this test. The results show that by choosing the optimal value for the $M_{\text{max}}$ one can save the computational time and get better accuracy in the predictions.

**Application of MARS in Flaw Identification**

In this section, regular shaped flaws, such as circles and ellipses and irregular shaped flaws were identified using MARS. In both cases the outer boundary of the domain is a 2 X 2 square. The origin of the Cartesian axis system is located at the center of the square, with the x-axis horizontal, and the y-axis vertical. The vertical sides of the square are held at temperature of 400$^\circ$ and 50$^\circ$ respectively. The other sides are insulated. The temperature probes are placed at the insulated boundary.
Figure 3.3: MARS performance in sizing the semi-major axis, $a$, of elliptical flaws as a function of the maximum number of basis functions.

Table 3.1: Performance of MARS in sizing the elliptical flaw's semi-major axis ($a$)

<table>
<thead>
<tr>
<th>Number of basis functions</th>
<th>CPU time (seconds)</th>
<th>Identified with error &lt; 0.02</th>
<th>Identified with error &lt; 0.03</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>9.1</td>
<td>86%</td>
<td>94%</td>
</tr>
<tr>
<td>15</td>
<td>14.3</td>
<td>86%</td>
<td>95%</td>
</tr>
<tr>
<td>20</td>
<td>21.4</td>
<td>88%</td>
<td>97%</td>
</tr>
<tr>
<td>25</td>
<td>31.2</td>
<td>87%</td>
<td>98%</td>
</tr>
<tr>
<td>30</td>
<td>45.7</td>
<td>91%</td>
<td>98%</td>
</tr>
<tr>
<td>35</td>
<td>52.6</td>
<td>90%</td>
<td>99%</td>
</tr>
<tr>
<td>40</td>
<td>74.0</td>
<td>92%</td>
<td>99%</td>
</tr>
<tr>
<td>45</td>
<td>94.4</td>
<td>91%</td>
<td>98%</td>
</tr>
<tr>
<td>50</td>
<td>113.3</td>
<td>92%</td>
<td>98%</td>
</tr>
</tbody>
</table>
Figure 3.4: MARS performance in sizing the semi-minor axis, $b$, of elliptical flaws as a function of the maximum number of basis functions

Table 3.2: Performance of MARS in sizing the elliptical flaw's semi-major axis ($b$)

<table>
<thead>
<tr>
<th>Number of basis functions</th>
<th>CPU time (seconds)</th>
<th>Identified with error &lt; 0.02</th>
<th>Identified with error &lt; 0.03</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>9.3</td>
<td>92%</td>
<td>99%</td>
</tr>
<tr>
<td>15</td>
<td>14.1</td>
<td>93%</td>
<td>99%</td>
</tr>
<tr>
<td>20</td>
<td>21.7</td>
<td>95%</td>
<td>99.3%</td>
</tr>
<tr>
<td>25</td>
<td>30.3</td>
<td>95%</td>
<td>99.4%</td>
</tr>
<tr>
<td>30</td>
<td>46.4</td>
<td>96%</td>
<td>99.5%</td>
</tr>
<tr>
<td>35</td>
<td>57.7</td>
<td>96%</td>
<td>99.5%</td>
</tr>
<tr>
<td>40</td>
<td>76.7</td>
<td>96%</td>
<td>99.4%</td>
</tr>
<tr>
<td>45</td>
<td>89.5</td>
<td>97%</td>
<td>99.5%</td>
</tr>
<tr>
<td>50</td>
<td>104.3</td>
<td>97%</td>
<td>99.5%</td>
</tr>
</tbody>
</table>
Figure 3.5: MARS performance in sizing x-coordinate, $x_c$ of elliptical flaws as a function of the maximum number of basis functions

Table 3.3: Performance of MARS in sizing the elliptical flaw’s center ($x_c$)

<table>
<thead>
<tr>
<th>Number of basis functions</th>
<th>CPU time (seconds)</th>
<th>Identified with error &lt; 0.2</th>
<th>Identified with error &lt; 0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>9.6</td>
<td>71%</td>
<td>84%</td>
</tr>
<tr>
<td>15</td>
<td>14.7</td>
<td>81%</td>
<td>92%</td>
</tr>
<tr>
<td>20</td>
<td>22.5</td>
<td>88%</td>
<td>97%</td>
</tr>
<tr>
<td>25</td>
<td>30.7</td>
<td>92%</td>
<td>98.5%</td>
</tr>
<tr>
<td>30</td>
<td>43.6</td>
<td>94%</td>
<td>99.0%</td>
</tr>
<tr>
<td>35</td>
<td>57.3</td>
<td>94%</td>
<td>99.5%</td>
</tr>
<tr>
<td>40</td>
<td>79.6</td>
<td>95%</td>
<td>99.5%</td>
</tr>
<tr>
<td>45</td>
<td>102.0</td>
<td>96%</td>
<td>99.5%</td>
</tr>
<tr>
<td>50</td>
<td>125.0</td>
<td>96%</td>
<td>99.5%</td>
</tr>
</tbody>
</table>
Figure 3.6: MARS performance in sizing y-coordinate, $y_c$ of elliptical flaws as a function of the maximum number of basis functions

Table 3.4: Performance of MARS in sizing the elliptical flaw's center ($y_c$)

<table>
<thead>
<tr>
<th>Number of basis functions</th>
<th>CPU time (seconds)</th>
<th>Identified with error &lt; 0.2</th>
<th>Identified with error &lt; 0.3</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>8.9</td>
<td>59%</td>
<td>73%</td>
</tr>
<tr>
<td>15</td>
<td>13.2</td>
<td>68%</td>
<td>81%</td>
</tr>
<tr>
<td>20</td>
<td>19.1</td>
<td>72%</td>
<td>85%</td>
</tr>
<tr>
<td>25</td>
<td>30.1</td>
<td>77%</td>
<td>91%</td>
</tr>
<tr>
<td>30</td>
<td>36.8</td>
<td>81%</td>
<td>94%</td>
</tr>
<tr>
<td>35</td>
<td>55.4</td>
<td>83%</td>
<td>95%</td>
</tr>
<tr>
<td>40</td>
<td>69.0</td>
<td>84%</td>
<td>95%</td>
</tr>
<tr>
<td>45</td>
<td>90.7</td>
<td>84%</td>
<td>95.5%</td>
</tr>
<tr>
<td>50</td>
<td>149.1</td>
<td>85.5%</td>
<td>95.7%</td>
</tr>
</tbody>
</table>
The probes data was calculated using the boundary element method for each example to create the testing set. The 700 element training set as described in the previous section was used as a training set. In each example, five elliptical parameters are evaluated using MARS five times, because the output dimension of MARS is one.

A two step process is used to improve the MARS predictions. First, using a global training set predicts the approximate locations and sizes of the flaws. The 700 element training set is used as a global training set. Second, a fine and new training set was created locally from the subdomain around the approximate location of the flaw. The prediction based on this fine training set gives a better prediction of the flaw information. The area of the subdomain could be as small as 1% of the whole area of the domain. Instead of creating a fine training set, one could extract a subset from the global training set around the neighborhood of the approximate location of the flaw.

Identification of Regular Shaped Flaws

In this section, three examples with elliptical flaws are randomly selected at different locations, sizes, and orientation in the domain. In the example 1, the flaw is located in the first quadrant of the square domain. In the example 2, the flaw is located in the second quadrant and the orientation angle is not zero. In the example 3, the flaw is located in the fourth quadrant and the value of \( b \) is almost two times of \( a \). In each example, the predicted values by the global search (test 1) and the local search (test 2) are given. The error analysis is also given.

Example 1: In this example, an ellipse was used, whose semi-major axis, \( a = 0.055 \), semi-minor axis, \( b = 0.05 \), center of the ellipse at \((0.64, 0.52)\), and the angle
of orientation is zero, as a real flaw. Fig. 3.7, and Table 3.5 show the results and the error analysis after test 1 from a global search. A subset was extracted from the training set used earlier in the neighborhood of the approximate location and the MARS was applied to find a better solution. The predicted values and the errors are shown in Table 3.6. More improved results could be achieved, by creating a new training set in the subdomain of the neighborhood of the approximate location. The results and the error analysis are given in Fig. 3.8, and Table 3.7.

Figure 3.7: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test1 for example 1
Table 3.5: Identification of a flaw using MARS; after a global search with a 700 element training set for example 1

<table>
<thead>
<tr>
<th>Variables</th>
<th>Real value</th>
<th>Test value</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0.055</td>
<td>0.04603</td>
<td>0.00897</td>
</tr>
<tr>
<td>b</td>
<td>0.040</td>
<td>0.03718</td>
<td>0.00282</td>
</tr>
<tr>
<td>xc</td>
<td>0.640</td>
<td>0.65917</td>
<td>0.01917</td>
</tr>
<tr>
<td>yc</td>
<td>0.520</td>
<td>0.65168</td>
<td>0.13168</td>
</tr>
<tr>
<td>α</td>
<td>0.000</td>
<td>0.41700</td>
<td>0.41700</td>
</tr>
</tbody>
</table>

Table 3.6: Identification of a flaw using MARS; after a local search with a 45 element training set extracted from the 700 element set for example 1

<table>
<thead>
<tr>
<th>Variables</th>
<th>Real value</th>
<th>Test value</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0.055</td>
<td>0.04612</td>
<td>0.00888</td>
</tr>
<tr>
<td>b</td>
<td>0.040</td>
<td>0.03427</td>
<td>0.00573</td>
</tr>
<tr>
<td>xc</td>
<td>0.640</td>
<td>0.65930</td>
<td>0.01930</td>
</tr>
<tr>
<td>yc</td>
<td>0.520</td>
<td>0.52187</td>
<td>0.00187</td>
</tr>
<tr>
<td>α</td>
<td>0.000</td>
<td>0.62832</td>
<td>0.62833</td>
</tr>
</tbody>
</table>
Figure 3.8: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test2 for example 1

Table 3.7: Identification of a flaw using MARS; after a local search with a newly trained 540 element training set for example 1

<table>
<thead>
<tr>
<th>Variables</th>
<th>Real value</th>
<th>Test value</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a$</td>
<td>0.055</td>
<td>0.05579</td>
<td>0.00079</td>
</tr>
<tr>
<td>$b$</td>
<td>0.040</td>
<td>0.03995</td>
<td>0.00050</td>
</tr>
<tr>
<td>$xc$</td>
<td>0.640</td>
<td>0.64107</td>
<td>0.00107</td>
</tr>
<tr>
<td>$yc$</td>
<td>0.520</td>
<td>0.51225</td>
<td>0.00775</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>0.000</td>
<td>0.09931</td>
<td>0.09931</td>
</tr>
</tbody>
</table>
Example 2: In this example, an ellipse is given by \((a = 0.052, b = 0.035, xc = -0.54, yc = 0.28, \alpha = 0.7)\) as a real flaw. After a global search with a 700 element training set, the result and error analysis are shown in Fig. 3.9 and Table 3.8. The results and error analysis are given in the Fig. 3.10 and Table 3.9, after the local search with a newly trained 540 element training set.

Table 3.8: Identification of a flaw using MARS; after a global search with a 700 element training set for example 2

<table>
<thead>
<tr>
<th>Variables</th>
<th>Real value</th>
<th>Test value</th>
<th>Absolute error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0.052</td>
<td>0.0406</td>
<td>0.0114</td>
</tr>
<tr>
<td>(b)</td>
<td>0.035</td>
<td>0.0339</td>
<td>0.0011</td>
</tr>
<tr>
<td>(xc)</td>
<td>-0.540</td>
<td>-0.6251</td>
<td>0.0851</td>
</tr>
<tr>
<td>(yc)</td>
<td>0.280</td>
<td>0.1534</td>
<td>0.1266</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>0.700</td>
<td>0.5156</td>
<td>0.1844</td>
</tr>
</tbody>
</table>

Table 3.9: Identification of a flaw using MARS; after a global search with a newly trained 540 element training set for example 2

<table>
<thead>
<tr>
<th>Variables</th>
<th>Real value</th>
<th>Test value</th>
<th>Absolute error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0.052</td>
<td>0.0494</td>
<td>0.0026</td>
</tr>
<tr>
<td>(b)</td>
<td>0.035</td>
<td>0.0316</td>
<td>0.0034</td>
</tr>
<tr>
<td>(xc)</td>
<td>-0.540</td>
<td>-0.5456</td>
<td>0.0055</td>
</tr>
<tr>
<td>(yc)</td>
<td>0.280</td>
<td>0.2883</td>
<td>0.0083</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>0.700</td>
<td>1.3807</td>
<td>0.6807</td>
</tr>
</tbody>
</table>
Figure 3.9: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test1 for example 2

Figure 3.10: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test2 for example 2
Example 3: In this example, an ellipse is given by \(a = 0.024, \ b = 0.046, \ xc = 0.32,\ yc = -0.62, \) and \(\alpha = 0.0\) as a real flaw. After a global search, the results and error analysis are given in the Fig. 3.11 and the Table 3.10. The results and error analysis are given in the Fig. 3.12 and Table 3.11, after the local search with a newly trained 540 element training set.

In the all three examples given above, one can see that MARS method successfully identifies the flaws in a two-step process. An advantage of the MARS method over any other method is that one can test many cases with one model for a domain.

Table 3.10: Identification of a flaw using MARS, after a global search with a 700 elements training set for example 3

<table>
<thead>
<tr>
<th>Variables</th>
<th>Real value</th>
<th>Test value</th>
<th>Absolute error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0.024</td>
<td>0.050</td>
<td>0.026</td>
</tr>
<tr>
<td>(b)</td>
<td>0.046</td>
<td>0.025</td>
<td>0.021</td>
</tr>
<tr>
<td>(xc)</td>
<td>0.320</td>
<td>0.30</td>
<td>0.02</td>
</tr>
<tr>
<td>(yc)</td>
<td>-0.620</td>
<td>-0.60</td>
<td>0.02</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>0.000</td>
<td>1.57</td>
<td>1.57</td>
</tr>
</tbody>
</table>

Table 3.11: Identification of a flaw using MARS; after a global search with a newly trained 540 element training set for example 3

<table>
<thead>
<tr>
<th>Variables</th>
<th>Real value</th>
<th>Test value</th>
<th>Absolute error</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>0.024</td>
<td>0.0442</td>
<td>0.0202</td>
</tr>
<tr>
<td>(b)</td>
<td>0.046</td>
<td>0.0378</td>
<td>0.0082</td>
</tr>
<tr>
<td>(xc)</td>
<td>0.320</td>
<td>0.3153</td>
<td>0.0047</td>
</tr>
<tr>
<td>(yc)</td>
<td>-0.620</td>
<td>-0.6201</td>
<td>0.0001</td>
</tr>
<tr>
<td>(\alpha)</td>
<td>0.000</td>
<td>0.2045</td>
<td>0.2045</td>
</tr>
</tbody>
</table>
Figure 3.11: Identification of a flaw using MARS; the real flaw (solid line) and the predicted flaw (dot-dot line) after test1 for example 3

Figure 3.12: Identification of a flaw using MARS: The real flaw (solid line) and the predicted flaw (dot-dot line) after test2 for example 3
Identification of Irregular Shaped Flaws

In this section, MARS was applied to irregular shaped flaws. Irregular shaped flaws were approximated in a best-fit elliptic shapes. Six examples are presented in this section to show that the MARS method has the capacity to identify the irregular shaped flaw. The final result of the two-step process is given for all six examples. In the first test, a 700 element training set was used. In the second test, a training set of 80 elements was used in a 0.3 X 0.3 subdomain around the prediction of test 1 for the local search.

Example 1: In this example, a diamond-shaped flaw was chosen in the 1st quadrant. After test 1, the approximate elliptical shape is given by \((a = 0.074, b = 0.0397, \quad x_c = 0.721, \quad y_c = 0.82, \quad \phi = 0.838)\). After test 2, the predicted elliptical shape is given by \((a = 0.0678, b = 0.0278, \quad x_c = 0.773, \quad y_c = 0.718, \quad \phi = 1.063)\). The predicted elliptical shape flaw and the actual flaw are shown in Fig. 3.13.

![Figure 3.13: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 1](image-url)
Example 2: In this example, a semi-circular flaw was chosen in the upper half plane. After test 1, the approximate elliptical shape is given by \((a = 0.0675, b = 0.0492, x_c = -0.098, y_c = 0.83, \text{ and } \phi = 0.25)\). After test 2, the predicted elliptical shape is given by \((a = 0.0844, b = 0.066, x_c = -0.32, y_c = 0.698, \text{ and } \phi = 0.214)\). The predicted elliptical shape flaw and the actual flaw are shown in Fig. 3.14.

Figure 3.14: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 2
Example 3: In this example, an octagonal-shaped flaw was chosen in the 2nd quadrant. After test 1, the approximate elliptical shape is given by \( (a = 0.0918, b = 0.07, xc = -0.556, yc = 0.139, \text{ and } \phi = 1.2) \). After test 2, the predicted elliptical shape is given by \( (a = 0.0842, b = 0.0623, xc = -0.46, yc = 0.194, \text{ and } \phi = 1.79) \). The predicted elliptical shape flaw and the actual flaw are shown in Fig. 3.15.

![Figure 3.15: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 3](image-url)
Example 4: In this example, a quarter of circular shaped flaw was chosen in the 3nd quadrant. After test 1, the approximate elliptical shape is given by ($a = 0.0674$, $b = 0.0404$, $xc = -0.458$, $yc = -0.25$, and $\phi = 0.46$). After test 2, the predicted elliptical shape is given by ($a = 0.0747$, $b = 0.048$, $xc = -0.427$, $yc = -0.187$, and $\phi = 0.298$). The predicted elliptical shape flaw and the actual flaw are shown in Fig. 3.16.

Figure 3.16: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 4
Example 5: In this example, a cross-shaped flaw was chosen as a real flaw. After test 1, the approximate elliptical shape is given by \((a = 0.0944, b = 0.0825, x_c = 0.275, y_c = -0.377, \text{and } \phi = 1.08)\). After test 2, the predicted elliptical shape is given by \((a = 0.103, b = 0.091, x_c = 0.102, y_c = -0.195, \text{and } \phi = 1.607)\). The predicted elliptical shape flaw and the actual flaw are shown in Fig. 3.17.

Figure 3.17: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 5
Example 6: In this example, an equilateral triangular shaped flaw was chosen as a real flaw. After test 1, the approximate elliptical shape is given by \((a = 0.0642, b = 0.0375, xc = 0.721, yc = 0.278, \text{ and } \phi = 0.408)\). After test 2, the predicted elliptical shape is given by \((a = 0.0631, b = 0.0365, xc = 0.707, yc = 0.194, \text{ and } \phi = 0.354)\). The predicted elliptical shape flaw and the actual flaw are shown in Fig. 3.18.

Figure 3.18: The real flaw (solid line) and the predicted flaw (dot-dot line) of MARS for example 6
In the previous six examples, the MARS method was the only one used to identify the flaws. Thus, one could say that the MARS method can be applicable to identify irregular shaped flaws and it is not restricted to only regular shaped flaws (circles or ellipses). After knowing the best fit elliptical shape, one could use B-Spine approximation to get the shape of the actual flaw as shown in Chapter 2 to get the spline shape of the flaw. In general, the best fit elliptical shape is adequate and the precise shape of the flaw is not required in nondestructive evaluations.
CHAPTER 4. TOLERANCE OF THE ALGORITHMS TO EXPERIMENTAL NOISE

In this chapter, two examples are considered to test the tolerance of the algorithms due to experimental errors. In addition, the difficulties encountered in the multiple flaws identification are discussed. In the laboratory experiments, there is always a possibility of errors in the measurements. The average maximum error (noise) is approximately 0.01 units in our simulated laboratory experiments. In order to simulate experimental noises in the experimental measurements, uniform noises were added to the actual flaw responses. Then, they were used as an input to the algorithm. The predictions were compared with the real flaw’s location and size. When there were more than one flaw in the domain, some additional information was needed to identify the flaws. The difficulties involved are discussed in detail.

Tolerance of the Iteration Method Due to Experimental Noises

In this section, two examples are considered. Uniform noises 0.01 and 0.05 were added to the measurements and the predictions are compared with the zero experimental error results. In each example, a circular flaw was considered at the location (-0.5, 0.5) in the domain. For the first example the radius of the circular flaw was 0.1, and in the second example the radius is 0.05. In all the cases, 12 probes
were used with 6 probes at the top and 6 probes at the bottom of the domain that are uniformly placed. One of the insulated sides was kept at 400° and the other at 50°.

**Example 1:** The real flaw at (-0.5,0.5) and a guessed flaw at the center of the domain are shown in Fig. 4.1. When there were no experimental noise, the predicted flaw coincided with the real flaw as shown in Fig. 4.2. The minimization process took 756 iterations and the functional value was reduced to 1.25E-24. When an error (noise) of 0.01 was added uniformly in the measurements, the minimization stopped after 363 iterations and the functional value was reduced to 3.59E-4 with a message “unable to minimize further”. The predicted flaw and the real flaw are shown in Fig. 4.3. Although there existed a little difference in the real flaw and the predicted flaw, the result was quite satisfactory. When an error (noise) of 0.05 was added uniformly in the measurement, the minimization stopped after 259 iterations, and the functional value was reduced to 8.97E-3, and the same error message appeared. The predicted flaw and the real flaw are shown in the Fig. 4.4. As expected, when the added errors grew larger, the predicted flaws differed from the real flaw.

**Example 2:** In this example, a circular flaw was considered with a radius that was half of example 1, in the same location and experimental conditions. The real flaw and the guessed flaw are shown in Fig. 4.5. The predicted flaw with no experimental errors and the real flaw are shown in Fig. 4.6. The minimization process converged in 172 iterations and the functional value was reduced to 8.199E-17. When an error (noise) of 0.01 was added uniformly in the measurements, the minimization stopped after 325 iterations and the functional value was reduced to 3.5E-4 and the message “unable to minimize further” appeared. The predicted flaw and the real flaw are
Figure 4.1: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 1

shown in Fig. 4.7. Finally, an error (noise) of 0.05 was added uniformly in the measurement, and the minimization stopped after 250 iterations with the functional value was reduced 9.02E-3, and with the same error message. The predicted flaw and the real flaw are shown in Fig. 4.8. As the added errors grew larger the predicted flaw differed from the real flaw.

In the above two examples, it has been shown that the program has the capability to handle experimental errors to some degree. When the actual flaw is small, then the added error makes a considerable difference in the prediction of the flaw. By testing the number of cases with noises, one can produce a calibration standard for practical applications.
Figure 4.2: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 1 when the added experimental noise is zero

Figure 4.3: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 1 when the added experimental noise is 0.01
Figure 4.4: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 1 when the added experimental noise is 0.05

Figure 4.5: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 2
Figure 4.6: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 2 when the experimental noise is zero.

Figure 4.7: The real flaw (solid line) and the guessed flaw (dot-dot line) in example 2 when the experimental noise is 0.01.
Tolerance of the MARS Method Due to Experimental Noises

In this section, the MARS method is applied to the same examples considered in the previous section. A two-step process was used in the approximations, first a global training set of 700 element was used in the whole domain to get the approximate location and size of the flaw. Second, a fine 80 element training set is created in the neighborhood of the approximate location of the flaw in a 0.2 X 0.2 subdomain.

Example 1: In this example, a circular flaw of radius 0.1 at (-0.5,0.5) is considered. The predicted flaw with no noise in the measurement and the real flaw are shown in Fig. 4.9. Fig. 4.10 shows the results with an error (noise) of 0.01 for the predicted flaw and the real flaw, while Fig. 4.11 shows the results of an error (noise) of 0.05 for the predicted flaw and the real flaw.
Figure 4.9: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 1 when the added noise is 0.01.

Figure 4.10: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 1 when the added noise is 0.01.
Example 2: In this example, a circular flaw centered at (-0.5, 0.5) with a radius of 0.05 was considered. Fig. 4.12 shows the results the real flaw and the predicted flaw with zero experimental error. Fig. 4.13 shows the results of an error (noise) of 0.01 for the predicted flaw and the real flaw, while Fig. 4.14 shows the results of an error (noise) of 0.05 for the predicted flaw and the real flaw.
Figure 4.12: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 2

Figure 4.13: The real flaw (solid line) and the predicted flaw (dot-dot line) in example 2 when the added noise is 0.01
The predictions of the MARS method have similarities to the predictions of the iteration method. When the error (noise) is larger the difference in the predicted flaw has a larger variation. If the flaw size is small, then the temperature perturbation on the boundary will be small; hence the error (noise) in the measurement makes a greater difference in the prediction of the flaw. Finally, both methods have enough tolerance to approximate the location and size for an approximate experimental error (noise) of 0.01 in the temperature measurements.

**Difficulties in the Identification of Multiple Flaws in a Domain**

Identification of multiple flaws has a challenging problem for researchers in recent years because of the complexity involved due to the physical nature of the problem.
This problem was attempted with a single guessed flaw and multiple guessed flaws along with the iteration method. Two examples was considered, one of them having two arbitrary flaws in the domain and the other with three arbitrary flaws.

**Single guessed flaw:** When there existed more than one flaw in the domain, a single guessed flaw was tried with a radius of 0.1 at the center. It was hoped that the guessed flaw would reach the biggest or the nearest flaw, and all the flaws could be identified one by one. Twenty probes were used which were uniformly placed on the insulated boundary.

For example 1. two arbitrary flaws were considered in the first and second quadrant of the domain. The real flaws and the guessed flaw are shown in Fig. 4.15. The minimization process stopped with an error message “an absolute minimum can not be reached” after 127 iterations and functional value was reduced to 0.52. One could see there were no improvements in the minimization of the functional value. Fig. 4.16 shows the new location of the guessed flaw and the real flaws, at this stage.

Example 2. also show the attempt to identify a case of three flaws in the domain with a single guessed flaw. Fig. 4.17 shows the real flaws and the guessed flaw at the initial stage. The minimization process came to a stop after 182 iterations with the same error message as before. At this stage, the functional value was 0.8747, with no improvements in the minimization. Fig. 4.18 shows the new location and size of the guessed flaw and the real flaws.
Figure 4.15: Two real flaws (solid lines) and a single guessed flaw (dot-dot line) in example 1

Figure 4.16: Two real flaws (solid lines) and the updated guessed flaw (dot-dot line) in example 1
Figure 4.17: Three real flaws (solid lines) and a single guessed flaw (dot-dot line) in example 2

Figure 4.18: Three real flaws (solid lines) and the updated guessed flaw (dot-dot line) in example 2
Multiple guessed flaws: Finally, an attempt was made to approximate the multiple flaws with same number of guessed flaws. Now, the question is, “Where does one guess the multiple flaws?” After the trial and error method, it was found that when the locations were guessed close to the real flaws, the flaws could be identified without any trouble.

Fig. 4.19 shows Example 1, with two real flaws and two guessed flaws as circles with radius of 0.05 centered at (-0.25,0.3) and (0.6,0.6). The minimization method predicted flaw locations and sizes after 640 iterations. The functional value became 2.5E-7. Fig. 4.20 shows the elliptical shape of the predicted flaws and the real flaws.

Fig. 4.21 shows example 2, with three real flaws and the three guessed flaws as circles of radii of 0.05, and centered at (-0.25,0.3), (0.6,0.6), and (0.6,-0.2). The minimization process approximated the real flaws, after 902 iterations. The functional value became to 3.18E-10. Fig. 4.22 shows the final stage for the three elliptical shaped flaws and the three real flaws.

Hence, one could identify multiple flaws starting with multiple guessed flaws, if one knows the approximate locations of the flaws. If the approximate locations of the flaws are unknown, it is impossible to apply the iteration method for multiple flaws in a domain. In the next chapter, a method is developed to find the approximate locations of multiple flaws.
Figure 4.19: Two real flaws (solid lines) and two guessed flaws (dot-dot line) in example 1

Figure 4.20: Two real flaws (solid lines) and the predicted flaws (dot-dot line) in example 1
Figure 4.21: Three real flaws (solid lines) and three guessed flaws (dot-dot line) in example 2

Figure 4.22: Three real flaws (solid lines) and the predicted flaws (dot-dot line) in example 2
CHAPTER 5. DEVELOPMENT OF A METHOD TO FIND THE APPROXIMATE LOCATIONS AND SIZES OF MULTIPLE FLAWS

A method to find the approximate locations and sizes is considered in this chapter. Flaws can be anywhere in the domain and can have a variety of complicated geometrical shapes. From an extensive literature search, no artificial neural networks method were found which can be used for this purpose.

The first step in developing a method was to study the temperature perturbation on the boundary due to flaw(s) in the boundary for a large number of cases. Second, a connection was made between the relationship of temperature change to the flaw size and shape. Third, the superposition principle was used to simulate the multiple flaw responses from single flaw responses. Finally, an exhaustive search was conducted to find a match a test case to a combination of single flaws as approximate locations and sizes of the flaws.

Three examples are given with the real flaw(s) and the predicted flaw(s) in their approximate locations. For the single flaw cases an improvement can be noticed in the reduction in the iterations using the minimization method with the knowledge of approximate location and size, when the guessed flaw is chosen at the approximate location rather than at the center of the domain.
The Patterns of the Temperature Perturbation

Presently, the only way to identify the approximate locations and sizes of the multiple flaws is by analyzing the temperature measurements on the boundary. First, the temperature perturbations were analyzed due to the many different sizes and shapes of flaws at a particular location. Second, locations were moved to different places and the changes of the temperature perturbations were noted. After testing many cases, it was noted that changes in the patterns of the temperature perturbations on the insulated boundary were due to different flaws in the domain.

In the beginning, a single flaw, with same size at 100 locations distributed uniformly in the domain was considered. Then, calculations were made of the corresponding temperature perturbations at 20 points on the insulated boundary and kept in a data file. Use of this data enabled the identification of the approximate location of a same size flaw at anypoint in the domain. Considering many different sizes and locations of the flaws a large data set is needed. After comparing many test cases, it was discovered that a relationship existed between some of the data sets which enabled the reduction of the number of cases in the data file.

In the multiple flaws case, many combinations of the flaws are possible, therefore, it is impossible to store all the possible patterns in a data file. Thus, the superposition principle was used to construct multiple flaw patterns from single flaw patterns. Several examples are given in the following sections to demonstrate the applicability of the method.
The Relationship between Temperature Perturbation and Flaw Geometry

The heat flow in a two-dimensional domain is similar to a two-dimensional inviscid fluid flow. The temperature perturbation on the boundary due to a flaw in the domain can be considered to be similar to the potential change due to a solid body in the inviscid fluid flow. Consider a uniform stream flowing past a cylinder, as shown in Fig. 5.1. The source and sink are at F1 and F2. If the source and sink approach each other, it will form a doublet, as shown in Figs. 5.2 and 5.3.

The potential and stream functions are given by

\[ \phi = U r \cos \theta - \frac{\mu}{r} \cos \theta, \]
\[ \psi = U r \sin \theta + \frac{\mu}{r} \sin \theta. \]  

(5.1)

Figure 5.1: Flow around a closed body
Figure 5.2: Flow around an infinitely long cylinder of radius, $a$

Figure 5.3: Geometry of a doublet at O in the uniformed stream, $U$
In general, the values of \( \mu, \phi \) are clearly variables, but if one chooses the strength of the doublet to be \( \mu = Ua^2 \), then \( \phi = 0 \) on \( r = a \).

Hence,

\[
\phi = Ua (1 - \frac{a}{r}) \cos \theta. \tag{5.2}
\]

This is the potential function which is a superposition of the uniform flaw and a doublet, thus,

\[
\phi_{\text{doublet}} = -\frac{Ua^2}{r} \cos \theta. \tag{5.3}
\]

Next by investigating, the above equation, one can conclude that the potential change due the doublet is directly proportional to the square of the radius \( a \). Once, the relationship known from Eq. (5.3) it can be shown that the temperature perturbation at \( A \) due to the flaw (circle) at \( P \) is directly proportional to the square of the radius. Then, the temperature perturbation due to different sizes of circular flaws can be constructed from the single case for a particular location as shown in Fig. 5.4.

A circular flaw radius of 0.02 was placed at \((0.5,0.5)\), and the temperature perturbations were calculated using the Laplace solver on the insulated boundaries. Then, were multiplied by the appropriate ratios to simulate the temperature perturbations of circular flaws of radii 0.04, 0.06, and 0.08, and compared with the calculated perturbations by the Laplace solver. A comparison of temperature perturbations on the lower \((y = -1)\) and lower \((y = 1)\) boundaries are given in Figs. 5.5 and 5.6.
Figure 5.4: Geometry of a circular flaw in a square domain

Figure 5.5: A comparison of temperature perturbations on \( y = -1 \)
Simulation of Multiple Flaw Responses

One could use the superposition principle to simulate the temperature perturbation on the boundary due to multiple flaws in the domain with the temperature perturbations of single flaws in the domain, since the Laplace equation and the boundary conditions are linear

\[ L(\alpha U_1 + \beta U_2) = \alpha L(U_1) + \beta L(U_2), \]

where \( L \) is a linear operator. Therefore, one could add two single flaw cases to create a multiple flaw case as shown in Fig. 5.7. The following five examples are presented to justify the concepts that use the superposition principle to simulate the temperature perturbations on the boundary due to multiple flaws in the domain from single flaws temperature perturbation.
Figure 5.7: Superposition of two single flaws

Example 1: In this example, two flaws were considered in a 2 X 2 square domain. Flaw 1 is an ellipse given by $(a = 0.055, b = 0.04, xc = 0.64, yc = 0.52, \phi = 0.0)$. and flaw 2 is a circle given by radius $a = 0.044$ at $(-0.51, -0.43)$. The two flaws and the domain are shown in Fig. 5.8. The temperature perturbation on the insulated boundaries due to the flaw 1 in the domain is shown in the Fig. 5.9, and in the Fig. 5.10 for flaw 2. The comparison of the calculated temperature perturbations by Laplace solver for the two flaws and simulated by the superposition principle from the two single flaws in the domain is shown in Fig. 5.11.
Figure 5.8: Geometry of the two flaws in the domain for example 1

Figure 5.9: Temperature perturbation due to flaw 1 in example 1
Figure 5.10: Temperature perturbation due to flaw 2 in example 1

Figure 5.11: A comparison of calculated and simulated temperature perturbations for example 1
Example 2: In this example, two flaws were considered, which are close to each other in a 2 X 2 square domain, as shown in Fig. 5.12, Flaw 1 is an ellipse given by $(a = 0.034, b = 0.021, xc = 0.81, yc = 0.25,$ and $\phi = 0.5)$, and flaw 2 is also an ellipse given by $(a = 0.025, b = 0.056, xc = 0.34, yc = 0.24,$ and $\phi = 0.0)$. Both ellipses are in the first quadrant. The temperature perturbation on the insulated boundaries due to the flaw 1 in the domain is shown in Fig. 5.13, and flaw 2 in the domain is shown in Fig. 5.14. The comparison of the calculated temperature perturbations by the Laplace solver for the two flaws and simulated by the superposition principle from the two single flaws in the domain is shown in Fig. 5.15.

Figure 5.12: Geometry of the two flaws in the domain for example 2
Figure 5.13: Temperature perturbation due to flaw 1 in example 2

Figure 5.14: Temperature perturbation due to flaw 2 in example 2
Example 3: In this example, three flaws were considered in a 2 X 2 square domain as shown in Fig. 5.16. Flaw 1 is an ellipse given by \((a = 0.055, b = 0.04, x_c = 0.64, y_c = 0.52, \phi = 0.0)\), flaw 2 is also an ellipse given by \((a = 0.043, b = 0.063, x_c = -0.62, y_c = 0.72, \phi = 0.0)\), and flaw 3 is a circle given by radius \(a = 0.054\) center at \((-0.34, -0.43)\). The temperature perturbation on the insulated boundaries due to flaw 1 in the domain is shown in the Fig. 5.17, flaw 2 is shown in the Fig. 5.18, and flaw 3 in Fig. 5.19. The comparison of the calculated temperature perturbations by the Laplace solver for the three flaws and simulated by the superposition principle from the single flaws in the domain is shown in Fig. 5.20.
Figure 5.16: Geometry of the two flaws in the domain for example 3

Figure 5.17: Temperature perturbation due to flaw 1 in example 3
Figure 5.18: Temperature perturbation due to flaw 2 in example 3

Figure 5.19: Temperature perturbation due to flaw 3 in example 3
Example 4: In this example, three elliptical flaws were considered such that two of them are closer to each other in a 2 X 2 square domain as shown in Fig. 5.21. The flaw 1 is given by \((a = 0.055, b = 0.04, xc = 0.64, yc = 0.52, \text{ and } \phi = 0.0)\). flaw 2 is given by \((a = 0.034, b = 0.021, xc = 0.81, yc = 0.25, \text{ and } \phi = 0.5)\). and flaw 3 is given by \((a = 0.037, b = 0.074, xc = -0.41, yc = 0.47, \text{ and } \phi = 0.0)\). The temperature perturbation on the insulated boundaries due to flaw 1 in the domain is shown in the Fig. 5.22, the flaw 2 in the Fig. 5.23, and the flaw 3 in the Fig. 5.24. The comparison of the calculated temperature perturbations by the Laplace solver for the three flaws and simulated by superposition principle from the single flaws in the domain is shown in Fig. 5.25.
Figure 5.21: Geometry of the two flaws in the domain for example 4

Figure 5.22: Temperature perturbation due to flaw 1 in example 4
Figure 5.23: Temperature perturbation due to flaw 2 in example 4

Figure 5.24: Temperature perturbation due to flaw 3 in example 4
Example 5: In this example, three elliptical flaws were considered which are close to each other in a 2 X 2 square domain as shown in Fig. 5.26. Flaw 1 is given by \((a = 0.055, \ b = 0.04, \ xc = 0.64, \ yc = 0.52, \text{ and } \phi = 0.0)\), flaw 2 is given by \((a = 0.034, \ b = 0.021, \ xc = 0.81, \ yc = 0.25, \text{ and } \phi = 0.5)\), and flaw 3 is given by \((a = 0.025, \ b = 0.056, \ xc = 0.34, \ yc = 0.24, \text{ and } \phi = 0.0)\). The temperature perturbation on the insulated boundaries due to the flaw 1 in the domain is shown in Fig. 5.27, flaw 2 in Fig. 5.28, and flaw 3 in Fig. 5.29. The comparison of the calculated temperature perturbations by the Laplace solver for the three flaws and simulated by the superposition principle from the single flaws in the domain is shown in Fig. 5.30.
Figure 5.26: Geometry of the two flaws in the domain for example 5

Figure 5.27: Temperature perturbation due to flaw 1 in example 5
Figure 5.28: Temperature perturbation due to flaw 2 in example 5

Figure 5.29: Temperature perturbation due to flaw 3 in example 5
From the preceding five examples, it has been successfully demonstrated that one can simulate the temperature perturbations of multiple flaws in a domain using the temperature perturbations of the single flaws in the domain. In the following sections, the selection of data set are discussed and three examples are given to identify the approximate locations of the flaw(s) using simulation.

**Selection of the Data Set for Flaw Identification**

In the previous sections, the relationship between temperature perturbation and flaw geometry was discussed. Using this relationship, one could simulate the temperature perturbation of many circles and ellipses from the temperature perturbation of a circle at a location. Therefore, one only needs the perturbation of one size flaw in a few locations in the domain.
Single flaw: In the case of identifying a single flaw of length $\leq 0.2$ at any location in a 2X2 domain, a (19X19)=361 element data set was used. First, the domain was divided as a 20 X 20 grid and the temperature perturbations were calculated using the Laplace solver by placing a circular flaw with a radius of 0.02 at all the 361 internal points. The temperature perturbations were calculated at 20 points on the boundary for each of the flaw locations.

Next, at each location of the flaw, temperature perturbations of 64 different sizes of ellipses were produced by taking 8 different semi-major and semi-minor axes which varied from 0.02 to 0.09. Using these 361 locations and 64 different sizes, one can simulate total of (361 X 64) = 23,104 cases of elliptical flaws. Therefore, using a sample of 361 element data set, one can identify the approximate location and size of any flaw having a length $\leq 0.2$ at any location in a 2 X 2 domain. A few examples are given in the next section. The selection of a data set can be any good sample in the domain. Ten points per side in the domain were taken and the temperature perturbation was evaluated and used as the input for the exhaustive search method and the MARS method.

Multiple flaws: To identify the approximate locations of multiple flaws, only a data set of single flaws is needed, since use of superposition principle allows the simulation of temperature perturbations due to multiple flaws. When a multiple flaw is tested and compared with, the temperature perturbation is all possible combinations of the single flaws. In order to reduce the search time, one must take a smaller data set than is used in the single flaw identification case. By considering a 10 X 10 grid, a 100 element data set of circles having radii of 0.02 was used. The closest distance between any two flaws was 0.2.
In this case, 8 different sizes of circles were taken whose radii varied from 0.02 to 0.09 at each location, since only the approximate locations and sizes of the flaws were needed. Using this 100 element data set, 800 flaws were simulated at 100 locations. By considering all the combination of circles which are not in the same locations, one can successfully identify the approximate locations and sizes of the multiple flaws by the exhaustive search. In this research, the number of flaws in the domain were restricted to two or three. To get a better result in the exhaustive search, in the multiple flaws identification, the temperature perturbations were obtained for all four sides of the domain by repeating the experiment.

Applications for Finding the Approximate Location and Size of Flaws

In this section, three examples were considered to find the approximate locations and sizes using the method described in this chapter. Next, the approximate location was identified and a tolerance was given then, the program selected all the possible flaws which had an error less than the tolerance. If the selected set was larger, a smaller set could be selected from the previously selected set. After reducing the number of selected flaws to less than five, the average locations and the sizes, or the one having the smallest error as the approximate locations and sizes, could be considered.

Example 1: In this example, an ellipse was considered, given by \((a = 0.043, b = 0.063, x_c = -0.62, y_c = 0.72, \phi = 0.0)\) as a real flaw in a 2 X 2 domain. The 361 element data set was used and tested with 23,104 possible cases of single cases. The real flaw and the predicted approximate flaw using the exhaustive search method are shown in Fig. 5.31.
Example 2: In this example, two elliptical flaws were considered, given by \( (a = 0.055, b = 0.04, x_c = 0.64, y_c = 0.52, \phi = 0.0) \), and \( (a = 0.043, b = 0.063, x_c = -0.62, y_c = 0.72, \phi = 0.0) \) as real flaws in the 2 X 2 domain. A 100 element data set was used and 259,200 possible combination of flaws were tested. Fig. 5.32 shows the real flaws and the predicted approximate flaws using the exhaustive search method.

Example 3: In this example three ellipses are considered as real flaws, which are given by \( (a = 0.055, b = 0.04, x_c = 0.64, y_c = 0.52, \phi = 0.0) \), \( (a = 0.052, b = 0.035, x_c = -0.544, y_c = 0.28, \phi = 0.7) \), and \( (a = 0.045, b = 0.069, x_c = 0.62, y_c = -0.32, \phi = 1.72) \). The 100 element data set was used and 44,236,800 possible cases of different combinations of three flaws were tested which were at different locations. The real flaws and the predicted approximate flaws are shown in Fig. 5.33.
Figure 5.32: The real flaws (solid lines) and the predicted flaws (dot-dot line) for example 2

Figure 5.33: The real flaws (solid lines) and the predicted flaws (dot-dot line) for example 3
Reduction in the Number of Iterations

The number of iterations required in a minimization algorithm depends on the number of variables involved, the number of residuals, and the difference between the guessed flaw and the solution. If one knows the approximate location and size of a flaw, then the minimization process takes a fewer number of steps than when starting at the center of the domain. Consider 9 examples as given below. All the examples are ellipses and given by:

Example 1: \((a = 0.055, b = 0.040, xc = 0.64, yc = 0.52, \text{ and } \alpha = 0.0)\);
Example 2: \((a = 0.034, b = 0.021, xc = 0.81, yc = 0.25, \text{ and } \alpha = 0.5)\);
Example 3: \((a = 0.025, b = 0.056, xc = 0.34, yc = 0.24, \text{ and } \alpha = 0.0)\);
Example 4: \((a = 0.043, b = 0.063, xc = -0.62, yc = 0.72, \text{ and } \alpha = 0.0)\);
Example 5: \((a = 0.052, b = 0.035, xc = -0.54, yc = 0.28, \text{ and } \alpha = 0.7)\);
Example 6: \((a = 0.037, b = 0.074, xc = -0.41, yc = 0.47, \text{ and } \alpha = 0.0)\);
Example 7: \((a = 0.054, b = 0.054, xc = -0.34, yc = -0.43, \text{ and } \alpha = 0.0)\);
Example 8: \((a = 0.044, b = 0.044, xc = -0.51, yc = -0.43, \text{ and } \alpha = 0.0)\); and
Example 9: \((a = 0.024, b = 0.046, xc = 0.32, yc = -0.62, \text{ and } \alpha = 0.0)\).

The results are presented in Table 5.1 to show the reduction in the number of iterations by choosing the guessed flaw at the approximate location over choosing a circular flaw with a radius 0.1 at the center of the domain. The approximate locations and sizes of the predicted ellipses by the exhaustive search method for the above examples, are given by:

Example 1: \((a = 0.09, b = 0.03, xc = 0.7, yc = 0.5, \text{ and } \alpha = 0.0)\);
Example 2: \((a = 0.05, b = 0.02, xc = 0.8, yc = 0.3, \text{ and } \alpha = 0.0)\);
Example 3: \((a = 0.03, b = 0.06, xc = 0.3, yc = 0.2, \text{ and } \alpha = 0.0)\);
Example 4: \((a = 0.09, b = 0.05, xc = -0.6, yc = 0.7, \text{ and } \alpha = 0.0)\);

Example 5: \((a = 0.05, b = 0.04, xc = -0.5, yc = 0.2, \text{ and } \alpha = 0.0)\);

Example 6: \((a = 0.04, b = 0.07, xc = -0.4, yc = 0.5, \text{ and } \alpha = 0.0)\);

Example 7: \((a = 0.03, b = 0.07, xc = -0.3, yc = -0.4, \text{ and } \alpha = 0.0)\);

Example 8: \((a = 0.06, b = 0.04, xc = -0.5, yc = -0.4, \text{ and } \alpha = 0.0)\); and

Example 9: \((a = 0.02, b = 0.05, xc = 0.3, yc = -0.6, \text{ and } \alpha = 0.0)\).

Table 5.1: Comparison of the number of iterations \((N)\) and the functional value \((F)\) in the minimization process with and without the knowledge of the approximate location of single flaws

<table>
<thead>
<tr>
<th>Flaws</th>
<th>Guessed flaw at the center</th>
<th>Guessed flaw at the approx. location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Example 1</td>
<td>(N=407, F=1.67E-20)</td>
<td>(N=224, F=6.51E-19)</td>
</tr>
<tr>
<td>Example 2</td>
<td>(N=1001, F=1.75E-07)</td>
<td>(N=241, F=1.22E-19)</td>
</tr>
<tr>
<td>Example 3</td>
<td>(N=264, F=9.22E-15)</td>
<td>(N=562, F=4.98E-20)</td>
</tr>
<tr>
<td>Example 4</td>
<td>(N=405, F=1.23E-19)</td>
<td>(N=104, F=1.31E-20)</td>
</tr>
<tr>
<td>Example 5</td>
<td>(N=126, F=3.05E-17)</td>
<td>(N=196, F=8.19E-21)</td>
</tr>
<tr>
<td>Example 6</td>
<td>(N=418, F=4.05E-14)</td>
<td>(N=143, F=5.94E-23)</td>
</tr>
<tr>
<td>Example 7</td>
<td>(N=157, F=4.84E-24)</td>
<td>(N=52, F=1.58E-21)</td>
</tr>
<tr>
<td>Example 8</td>
<td>(N=260, F=1.47E-17)</td>
<td>(N=047, F=5.42E-24)</td>
</tr>
<tr>
<td>Example 9</td>
<td>(N=233, F=1.59E-22)</td>
<td>(N=104, F=1.40E-21)</td>
</tr>
</tbody>
</table>

Twenty probes uniformly placed on the insulated boundary were used to measure the temperature, while 32 linear elements were taken on the outer boundary of the square domain and 12 linear elements were taken around the boundary of the flaws in the boundary element equation. All the above examples were selected randomly. By comparing the results, one can see there is a considerable reduction in the number of iterations, except for examples 3 and 5. Therefore, in general, knowledge of the approximate location and size has an advantage in single flaw identification.
CHAPTER 6. EXAMPLES OF THE IDENTIFICATION OF MULTIPLE FLAWS

Six examples with multiple flaws are considered in this chapter. The first two examples have two real flaws in the 2 X 2 domain the last four examples have three real flaws. For each example, the real flaws, approximate predicted flaws by the exhaustive search method, flaws predicted by the MARS method, and predicted by iteration method are given. After knowing the approximate locations and the sizes of the flaws, the MARS method and iteration method were applied independently.

In the MARS method a 0.2 X 0.2 subdomain was taken around the approximate location of the flaws and a few combination of flaws in the regions with their temperature perturbation are used for training set. Training sets were formed of \((25 \times 25) = 625\) elements for the two-flaw cases, and \((4 \times 4 \times 4) = 64\) elements for the three-flaw cases.

For the iteration method, 64 linear elements were taken on the outer boundary of the square for the two-flaw cases, and 32 linear elements on the outer boundary of the square for three-flaw cases. Double nodes were placed at the corners of each square to handle the discontinuous boundary conditions as proposed by Mitra [42]. Twelve linear elements on each guessed flaws were also taken.
Example 1: In this example, an ellipse was defined by \( a = 0.055, b = 0.04, xc = 0.64, yc = 0.52, \phi = 0.0 \) and a circle of radius \( a = 0.044 \) at \((-0.51, -0.43)\) represented two real flaws as shown in Fig. 6.1. The approximate locations of the predicted flaws by the exhaustive search method are shown in Fig. 6.2. The predicted flaws by the MARS method and the real flaws are shown in Fig. 6.3. The iteration method used with 30 probes on the boundary, converges after 391 iterations, and the predicted flaws and the real flaws are shown in Fig. 6.4.

Figure 6.1: The real flaws in the 2X2 square domain for example 1
Figure 6.2: Predicted flaws using an exhaustive search method for example 1

Figure 6.3: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 1
Example 2: In this example, two ellipses were considered, given by $(a = 0.034, b = 0.021, xc = 0.81, yc = 0.25, \phi = 0.5)$, and $(a = 0.025, b = 0.056, xc = 0.34, yc = 0.24, \phi = 0.0)$ as real flaws as shown in Fig. 6.5. The approximate location of the predicted flaws by the exhaustive search method are shown in Fig. 6.6. The predicted flaws by MARS method and the real flaws are shown in Fig. 6.7. The iteration method converges 770 iterations with 26 probes and the functional value is $F = 4.298\times10^{-10}$. The predicted flaws and the real flaws are shown in Fig. 6.8.
Figure 6.5: The real flaws in the 2X2 square domain for example 2

Figure 6.6: Predicted flaws using the exhaustive search method for example 2
Figure 6.7: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 2

Figure 6.8: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 2
Example 3: In this example, three flaws were considered, two of them were ellipses and the remaining one a circle in the domain. The ellipses are given by \((a = 0.055, b = 0.04, xc = 0.64, yc = 0.52, \phi = 0.0)\), and \((a = 0.043, b = 0.063, xc = -0.62, yc = 0.72, \phi = 0.0)\), and the circle is given by radius \(a = 0.054\) at \((-0.34, -0.43)\) as shown in Fig. 6.9. The predicted flaws by the exhaustive search method are shown in Fig. 6.10. The predicted flaws by the MARS and the real flaws shown in Fig. 6.11. The iteration method converges in 489 iterations with 20 probes on the outer boundary. The functional value was 1.06E-24. The predicted flaws and the real flaws are shown in Fig. 6.12.

Figure 6.9: The real flaws in the 2X2 square domain for example 3
Figure 6.10: Predicted flaws using the exhaustive search method for example 3

Figure 6.11: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 3
Figure 6.12: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 3

Example 4: In this example, three flaws were considered, two of them were ellipses given by $(a = 0.055, \; b = .04, \; xc = 0.64, \; yc = 0.52, \; \phi = 0.0)$, and $(a = 0.024, \; b = .046, \; xc = 0.32, \; yc = -0.62, \; \phi = 0.0)$, and the other a circle with radius of $a = 0.044$ at $(-0.51, -0.43)$ in the domain as shown in Fig. 6.13. The predicted flaws by the exhaustive search method are shown in Fig. 6.14. The predicted Flaws by MARS method and the real flaws are shown in Fig. 6.15. The iterative method converges in 259 iterations with 16 probes on the outer boundary. The functional value, $F$, at this stage was $8.85\text{E-19}$. The predicted flaws and the real flaws are shown in Fig. 6.16.
Figure 6.13: The real flaws in the 2X2 square domain for example 4

Figure 6.14: Predicted flaws using the exhaustive search method for example 4
Figure 6.15: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 4

Figure 6.16: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 4
Example 5: In this example, three elliptical flaws were considered, given by \((a = 0.055, b = 0.04, xc = 0.64, yc = 0.52, \phi = 0.0)\), \((a = 0.034, b = 0.021, xc = 0.81, yc = 0.25, \phi = 0.5)\), and \((a = 0.037, b = 0.074, xc = -0.41, yc = 0.47, \phi = 0.0)\) in the domain. Two of them were in the first quadrant as shown in Fig. 6.17. The predicted flaws by the exhaustive search method are shown in Fig. 6.18. The predicted flaws by the MARS method and the real flaws are shown in the Fig. 6.19. The iteration method took 2000 iterations with 24 probes to reduce the functional value to \(3.79E^{-10}\). The predicted flaws and the real flaws are shown in Fig. 6.20.

Figure 6.17: The real flaws in the 2X2 square domain for example 5
Figure 6.18: Predicted flaws using the exhaustive search method for example 5

Figure 6.19: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 5
Example 6: In this example, three elliptical flaws were considered, given by \((a = 0.055, b = 0.04, x_c = 0.64, y_c = 0.52, \phi = 0.0), (a = 0.034, b = 0.021, x_c = 0.81, y_c = 0.25, \phi = 0.5),\) and \((a = 0.025, b = 0.056, x_c = 0.34, y_c = 0.24, \phi = 0.0)\) in the domain. All three of the flaws were in the first quadrant as shown in the Fig. 6.21. The predicted flaws by the exhaustive search method are shown in Fig. 6.22. The predicted flaws by the MARS method and the real flaws are shown in the Fig. 6.23. The iteration method minimized the functional to 1.619E-4 after 3000 iterations with 16 probes on the boundary. The predicted flaws using iteration method and real flaws are shown in Fig. 6.24.
Figure 6.21: The real flaws in the 2X2 square domain for Example 6

Figure 6.22: Predicted flaws using the exhaustive search method for Example 6
Figure 6.23: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the MARS method for example 6

Figure 6.24: The real flaws (solid lines) and predicted flaws (dot-dot lines) using the iteration method for example 6
It has been demonstrated that multiple flaws in a 2D domain can be identified by the MARS method and the iteration method by initially finding the approximate locations and sizes of the flaws. In chapter 5, a method was given to identify the approximate locations and sizes of multiple flaws by the exhaustive search method. If the flaws were closer to each other, then both methods had a little difficulty in identifying them, since the interaction between the flaws are greater. In general, the iteration method gives a more accurate result than the MARS method.

Although, the iteration method is the traditional method for solving inverse problems, the multiple flaws problem is too highly complicated to apply iteration method without any knowledge of the approximate locations of the flaws. In this research, this nonlinear problem was made solvable by coupling with the exhaustive search method as described in chapter 5. Sixty-four linear elements were used on the outer boundary of the domain in the identification of the problems with two flaws in the boundary, and 32 linear elements were used on the outer boundary for the three-flaw problems in order to reduce the number of iterations and the computational time. The minimization process takes different numbers of iterations when different numbers of probes on the boundary, since this inverse problem is highly nonlinear. The key to identify multiple flaws is, to initially find the approximate locations and sizes of the flaws, then apply a multivariate regression method or the iteration method.
CHAPTER 7. CONCLUSION

In this thesis, a new method is developed to identify the multiple flaws by their locations, shapes, and sizes in a material. The method requires knowledge of the number of flaws in the material, and the temperature perturbations on a part of the outer boundary of the domain.

Identification of multiple flaws in a material is an inverse problem which naturally occurs in nondestructive testing in engineering, physical science, and medicine. This is a difficult problem and it's solution requires sophisticated methods of mathematics. The problem in its generality is an unsolved problem of pure mathematics (see Anger [1]). In practice, the disturbance in the physical field (in this case the temperature field) is caused by the combination of flaws and effect of each individual flaw is not known. It makes more difficult characterizing the multiple flaws. Traditionally, iteration method used to solve inverse problems. The iteration method works well for the single flaw case, but it fails in the multiple flaws case, unless the approximate locations of the multiple flaws are known.

The multiple flaws are identified in a two-step process in this thesis. In the first step, the approximate locations of flaws are determined using a search method. In the second step, the iteration method (the traditional method) or the MARS method (the regression method) is applied independently to find the final predictions. Finding of
the approximate locations of the flaws leads to the application of existing methods.

The approximate locations of the flaws are determined using a truncated training set which consists of a fixed size single circular flaw at different locations. Each element in the training set has the information of a flaw and its temperature perturbation on the outer boundary of the domain. The truncated training set is expanded to different sizes of flaws using the relationship between the flaw size and the temperature perturbation on the boundary. The superposition principle is used to simulate multiple flaws temperature perturbations from single flaw temperature perturbations. The linearity of the governing equations allows the use of superposition principle, since the expanded training set can be very large is not stored, but created as and when necessary.

Application of MARS an alternative method to iteration method is a new development in the identification of material flaws. Some features of the MARS algorithm are presented in this thesis. User have to choose a maximum number of the basis functions and design a training set for the particular Problem. The maximum number of basis functions can be find for a training set by testing some known cases. Finding the an optimum number for the maximum basis functions is important, since the accuracy does not improve when the maximum number of the basis functions get larger than the optimum number, but the cpu time gets increase exponentially. The training set for the MARS method in the identification of the multiple flaws is obtained by considering a few combinations of approximate locations and sizes of the multiple flaw cases.

The examples show that the MARS method has the potential in the identification of the regular and irregular-shaped flaws. The iteration method gives a more accurate
solution than MARS, however, in the case of testing a large number of cases for a particular domain and boundary condition, MARS is more appropriate.

The possible experimental noise and the tolerance of the MARS method and iteration method are analyzed using simulated noises in the data. The results show that the predictions of the methods are successful, unless a real flaw is very small compared to the domain or the simulated noise is much higher than average real noise.

Finally, the multiple flaws problem is solved by applying the new technique in combination with the existing methods. The emphasis throughout the research has been on the development of a fast and accurate algorithm in computational mechanics. The identification methods for multiple flaws based on the application of superposition principle which requires linearity in the governing equations. The preceding techniques are transferable to inverse problems in other fields such as in ultrasonic wave scattering, stress analysis, locating earthquake, electric conduction, etc., if the forward problems have linear or linearied governing equations. One could also extend this method to three dimensional problems. A possible future work is the application of these techniques to problems in wave scattering and stress analysis.
BIBLIOGRAPHY


