2015-05-01

An Automated Method for Hot-to-Cold Geometry Mapping

Brandon Levi Doolin
Brigham Young University - Provo

An Automated Method for Hot-to-Cold Geometry Mapping

Brandon Levi Doolin

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

C. Greg Jensen, Chair
Steven E. Gorrell
W. Edward Red

Department of Mechanical Engineering

Brigham Young University

May 2015

ABSTRACT

An Automated Method for Hot-to-Cold Geometry Mapping

Brandon Levi Doolin
Department of Mechanical Engineering, BYU
Master of Science

Shape optimization is critical in turbomachinery design for creating the most efficient and durable designs. Running or un-running (also known as cold-to-hot or hot-to-cold transformations) is frequently performed on rotor blades when optimizing the shape of these blades as well as preparing the design for manufacture. Accuracy in these processes is crucial to preserve the optimum shape and, consequently, its flow properties. The needed accuracy can be achieved and quantified by an iterative finite element method, but a geometry mapping must occur afterward to recreate the geometry from the discretized elements. Current methods, however, do not provide a method of quantifying the error that is introduced in the mapping step.

This research develops a method to perform this geometry mapping while also quantifying the mapping error. Kriging is implemented as a method to interpolate displacements in blade running and un-running transformations. Kriging is compared to radial basis function interpolation as a mapping scheme and Sculptor™'s Arbitrary Shape Deformation. Kriging is found comparable to these methods in speed, accuracy, and mapping smoothness. The Kriging standard deviation is investigated as a method to approximate mapping error. The standard deviation is found to be useful but not conclusive in approximating the mapping error.

Keywords: hot-to-cold, cold-to-hot, running, un-running, rotor blade, geometry mapping, kriging, radial basis function, Sculptor, arbitrary shape deformation

ACKNOWLEDGMENTS

I would like to thank my wife, Liz, for her patience and support in helping me to accomplish this research. Her love and constant encouragement was the largest part in finishing this research. I would also like to thank my son, Will, for his smile every night I come home. I love you both!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

NOMENCLATURE

w_i	Weight used in interpolation corresponding to the i^{th} known point
p	A point to interpolate at
p'_i	The i^{th} known point to use in interpolating
$\varphi(p - p'_i)$	A radial basis function evaluated using the distance between point p and p'_i
φ_{ij}	Equivalent to $\varphi(p'_i - p'_j)$
u	The displacement of a point
B_i^n	The i^{th} Bernstein basis function of order n
P_i	The coordinates of the i^{th} control point
$\binom{n}{i}$	Represents the binomial coefficient
C_n	The continuity of a piecewise curve where C_0 implies connectivity, C_1 implies tangency, C_2 implies curvature continuity, and so forth.
$Z(x, \omega)$	A random function
E	The expected value of a function
Var	Denotes the variance of a function
m	The mean of a function
σ_{ij}	The covariance function evaluated using the distance between point p and p'_i
	Represents a Lagrange multiplier
γ_{ij}	The semivariogram function evaluated using the distance between point p and p'_i
σ	Represents the standard deviation

CHAPTER 1. INTRODUCTION

1.1 Problem Statement

Today's turbomachinery rotor design generally follows one of two design methodologies. One of these two methods is to create a rotor blade geometry that would represent the as-manufactured shape. This geometry is referred to as the "cold" shape. Aerodynamic, centripetal, and other forces are then applied to the cold geometry using engineering analysis to determine the shape of the running or "hot" blade [1]. This method is commonly referred to as "cold-to-hot". The downside to this method is that the blade must be optimized for performance in the hot configuration, otherwise losses on the order of 3% in important parameters such as thrust, TSFC, etc. can be experienced due to blade deformation and untwist [2, 3]. Usually this means that for every design iteration, a structural analysis must be performed before fluid or any other analysis can be performed, though FEA and CFD coupling can be done to execute both simultaneously. Iteration must be performed as well to determine the correct aerodynamic loads as seen in Figure 1.1 [4].

The other common methodology is to simply begin with the hot shape and perform the design optimization. Once the hot shape has been determined, an iterative process is then used to determine the cold shape. This process is referred to as "un-running" or "hot-to-cold" [1]. The un-running begins by meshing the geometry with finite element software and applying aerodynamic, centripetal, and other loads to the hot mesh. The result of this analysis is referred to as the "double deflected" mesh. From the double deflected mesh, a set of node displacements is calculated and the negative of these are input along with the original hot mesh into an iteration loop. At the beginning of the loop the displacements are applied to the hot mesh to create an approximation to the cold mesh. Next, the aerodynamic and centripetal forces are applied to the cold approximation to create a hot mesh approximation. If the difference between the hot mesh and the approximate hot mesh is within tolerance, then the cold mesh approximation is accepted as the actual cold mesh. Otherwise, the displacements from the cold approximation to the hot approximation are calculated, and the

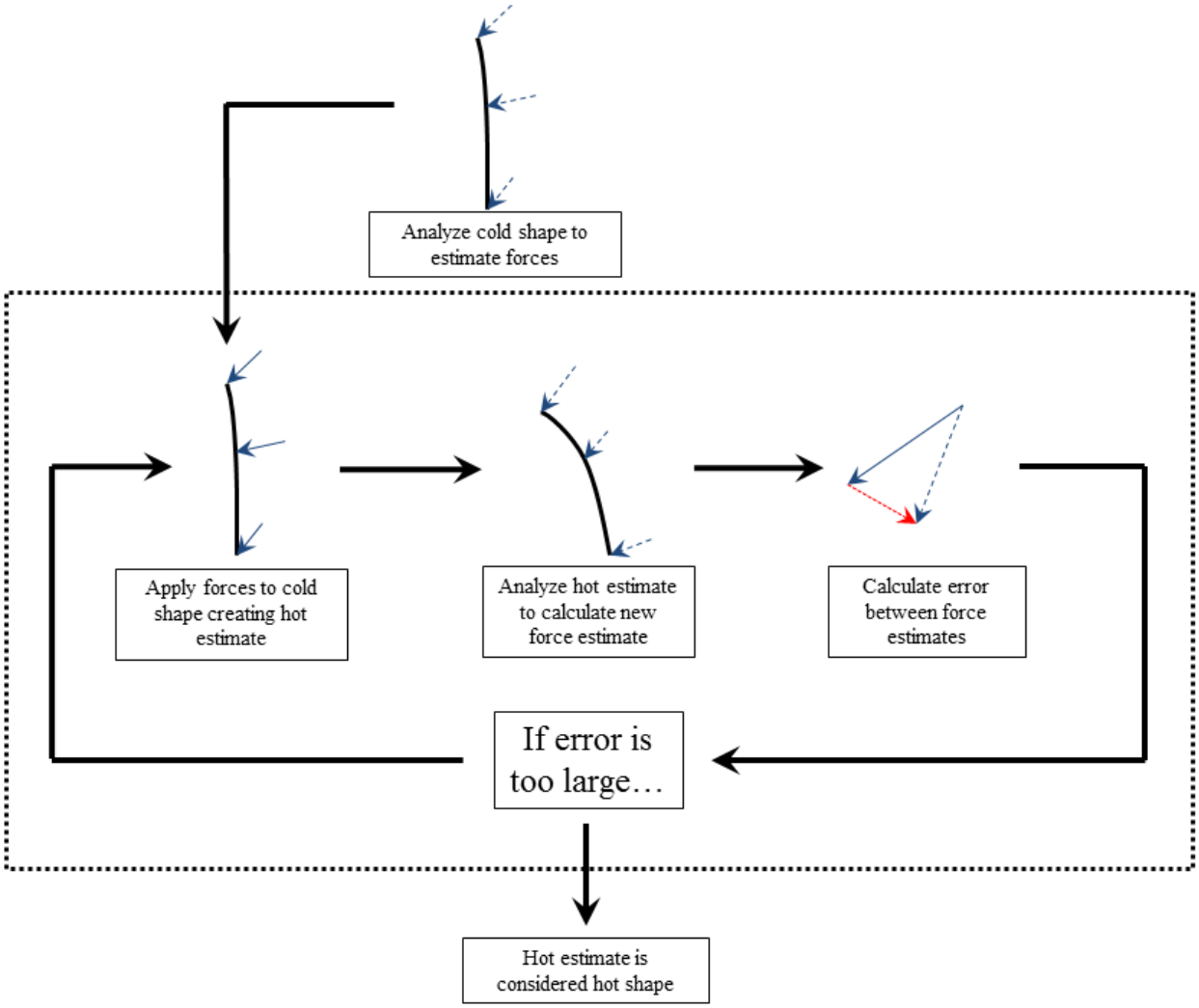

Figure 1.1: A flow chart illustrating the iterative loop for the cold-to-hot transformation

negative of these is input to the beginning of the loop. This loop is repeated until the correct cold mesh is found [5]. A diagram of this process can be seen in Figure 1.2.

Regardless of method choice, the designer will be left with three sets of data. Both methods will have a hot and cold mesh and one geometry representation—cold geometry in the case of cold-to-hot or hot geometry in the case of hot-to-cold. The other geometry file is unknown. This geometry file differs from the mesh file in it's placement of points or nodes. The mesh files have nodes placed at locations and in densities chosen by the meshing software. The geometry files have points chosen in specific locations and densities for use in the CAD system they come from. Consequently, these files almost always differ from each other. Since a geometry file is needed to

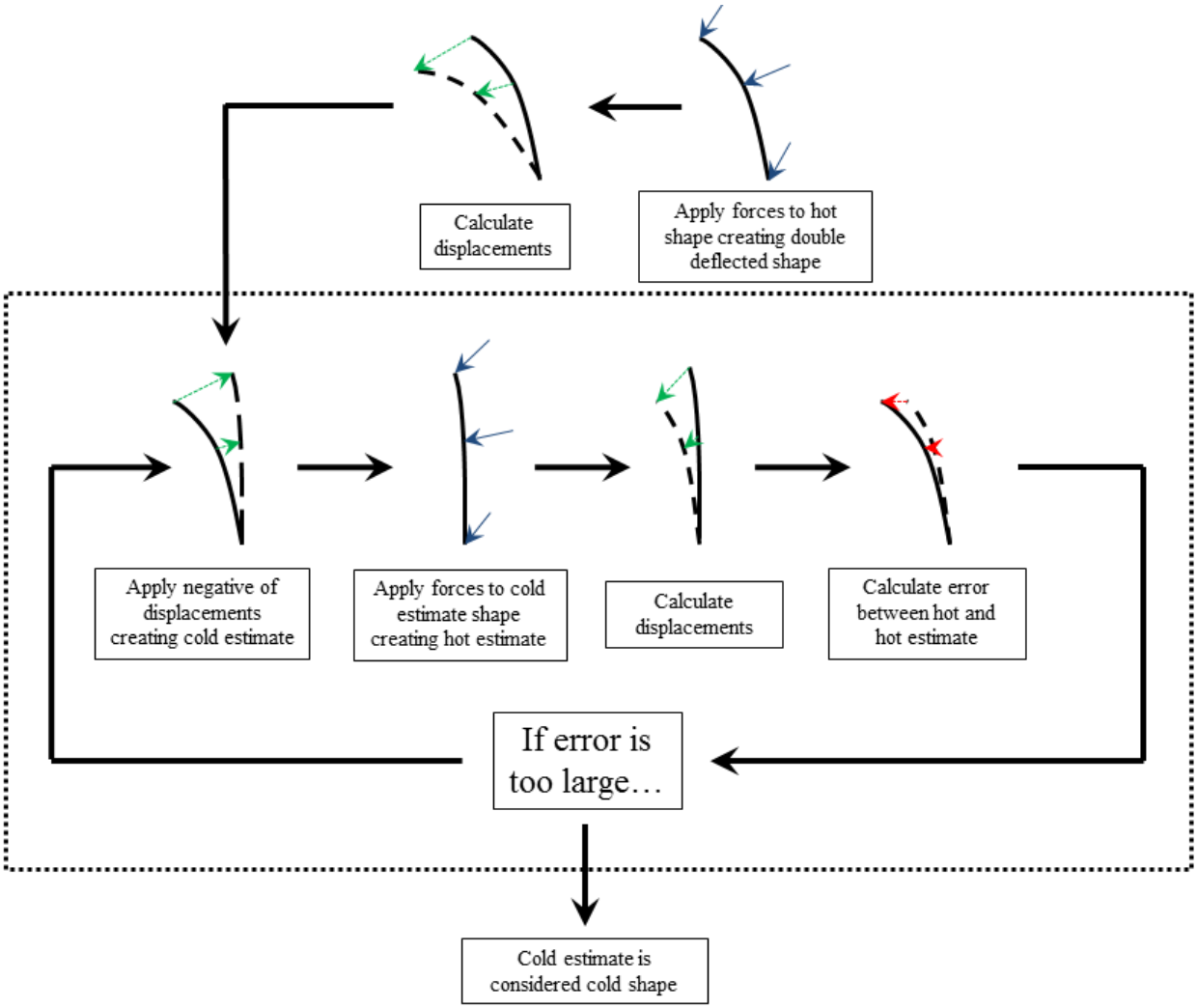

Figure 1.2: A flow chart illustrating the iterative loop for the hot-to-cold transformation

analyze the hot shape in CAx systems, create manufacturing drawings on the cold shape, or analyze at off-design conditions [1], engineers must map their existing geometry file from it's current hot state to a cold state (or conversely from a cold state to a hot state).

The techniques discussed in this work provide methods for determining what this unknown geometry file is by mapping the known geometry file. To avoid repetition, this work will only refer to the hot-to-cold case from here on, but the cold-to-hot case is implied as well whenever the former is mentioned.

Some software packages have methods to accomplish this internally such as T/BEST and STAT, but do not publish their exact process. These packages may use information from other

internal subsystems other than the three known data files mentioned [3, 4]. These methods, while useful, are not very portable to future systems and software packages.

Other methods are stand-alone processes that work off of only generic inputs, namely these mesh and geometry files, and output the necessary mapped file to the user. These methods treat the process for obtaining the hot and cold mesh as a "black box" and are only concerned with one input being a deformed mesh and the other input being undeformed. As this research focuses on these type of methods, from this point on, the hot mesh will be referred to as the deformed mesh, while the cold mesh will be referred to as the undeformed mesh.

Few methods of this type exist—only two were found by the author—warranting further investigation into these types of methods. Also, these two methods lack an estimation of error when performing the geometry mapping which could lead to undetected errors when performing the mapping with these methods either in local areas (e.g. areas of high curvature, discontinuities, etc.) or globally (e.g. the program failed to find a good mapping). Lastly, the methods that this research has identified have not been rigorously evaluated for criteria of accuracy, speed, and smoothness.

1.2 Research Objectives and Scope

This research seeks to fill the gaps in existing methods and research by developing a new stand-alone method for performing hot-to-cold geometry mapping and comparing it to the existing methods. The methods will be evaluated and compared by determining if they can:

1. Perform the geometry mapping using only generic inputs, that is using the x, y, and z coordinates of the points from the deformed mesh, the undeformed mesh, and the deformed geometry file.

2. Provide a mapping that is accurate within engineering tolerance (error on the order of 0.001–0.0001 in. in magnitude).

3. Produce a smooth shape from the mapping, free of wrinkles and divots.

4. Perform the mapping in under 30 minutes using modern computing resources (a quad-core, 2.67 GHz processor will be used for testing).

5. Provide an error estimation for the mapping.

Error magnitude was chosen to evaluate error instead of relative percent error. While relative percent error helps to normalize the error relative to its displacement, when manufacturing or analyzing a blade, only the magnitude of the error is important. Also, relative percent error skews the level of error of mapping near the fixed ends of blades where the displacement is nearly zero as well as at the free end where the displacement is large.

This research attempts to meet these criteria by implementing Kriging, an n-dimensional interpolation method, and mapping the displacements of points in each of the euclidean directions. Centripetal and constant pressure load cases were used. These were applied to a generic rotor blade and similarly sized cantilever beam using linear analysis in ANSYS 15.0™. The process was compared to two existing stand-alone methods for hot-to-cold geometry methods, namely radial basis function interpolation (RBF interpolation for short) as implemented by Persson and Runsten [6] and Sculptor™'s shape matching technology [7].

This research is limited to only linear analysis. Any non-linear effects from large displacements were not considered. Force cases from point-loads, varying pressures, temperature, and other sources were not considered to simplify the analysis. This research only uses these load cases to determine if the mapping methods are dependent on the forces applied, or if the force cases are unnecessary information to the process (important for portability). Ordinary Kriging was the only method used. Other Kriging methods such as Universal Kriging and Co-Kriging were not considered. The variogram was assumed to be isotropic. Implementing an anisotropic variogram is beyond the scope of this research as will be discussed later in section 3.1.4. Test models were limited to the two blade shapes mentioned above. Internal channels, such as those in turbine blades, were not included in the models.

1.3 Thesis Outline

This thesis begins with background information on existing methods in hot-to-cold geometry mapping in Chapter 2. RBF interpolation is found to be a very simple and accurate method.

Free-form deformation is discussed including Sculptor™'s improvements to the method. Sculptor™'s shape matching is found to be a robust and accurate method for hot-to-cold mapping. Chapter 3 includes information on Kriging's current uses and capabilities. An efficient methodology for organizing the data files is created. Methodology for preliminary analysis required for Kriging is covered including a genetic algorithm for variogram fitting. Kriging is implemented. Chapter 4 includes accuracy, speed, and complexity comparisons for all three methods. Each method's unique benefits are quantified. Chapter 5 interprets the results. It also suggests future improvements to the Kriging method for further research.

CHAPTER 2. BACKGROUND

2.1 Overview

This chapter covers background information on existing, published methods for performing geometry mapping in the hot-to-cold context. These methods cover the general, stand-alone case where only coordinates of points from the deformed mesh, undeformed mesh, and deformed geometry are known.

2.2 Radial Basis Function Mapping

Persson and Runsten [6] created a method for performing a hot-to-cold geometry mapping (or in their case, cold-to-hot) by reducing the mapping problem into an interpolation problem. For the hot-to-cold case, each point in the hot mesh is associated with an x, y, and z displacement. These displacements are the values needed to morph the hot mesh into its corresponding cold mesh. From this point, they mapped the geometry using radial basis function interpolation.

2.2.1 Radial Basis Functions

Radial basis functions (RBFs) are functions that output a value given a distance from the function's origin. This is defined by the relation $f(x) = f(\|x\|)$. The norm is generally taken to be the euclidean norm (as it is in this case). These functions are defined for all positive and negative values of the input variable r, though values of $r < 0$ are not generally used. RBFs come in many varieties, some of them can be seen in table 2.1. The Gaussian RBF is the familiar Gaussian distribution. The inverse quadratic and inverse multiquadratic have similar shapes to the Gaussian RBF.

Many RBFs are also positive definite, that is for every $x \in \mathbb{R}$, $f(x) > 0$. Many are also symmetric about a center point, usually chosen to be the origin, that is the value $r = 0$. These properties

Table 2.1: Some examples of common radial basis functions

Type	Equation
Gaussian	e^{-r^2/b^2}
Inverse Quadratic	$\frac{1}{b^2+r^2}$
Inverse Multiquadratic	$\frac{1}{\sqrt{b^2+r^2}}$

are useful in solving systems of linear equations from RBFs since stable, efficient solutions exist to symmetric, positive definite matrices. This will be seen later when the interpolation scheme is defined.

The parameter b in these functions is a shape factor that is used to alter the shape or distribution of the RBF. Larger values of the shape factor will create a wider shape of the function. Conversely, smaller values of b produce a thinner function shape. This can be seen in Figure 2.1. Shape factors are commonly used in improving the condition of matrices formed from RBFs [6], as will be seen later.

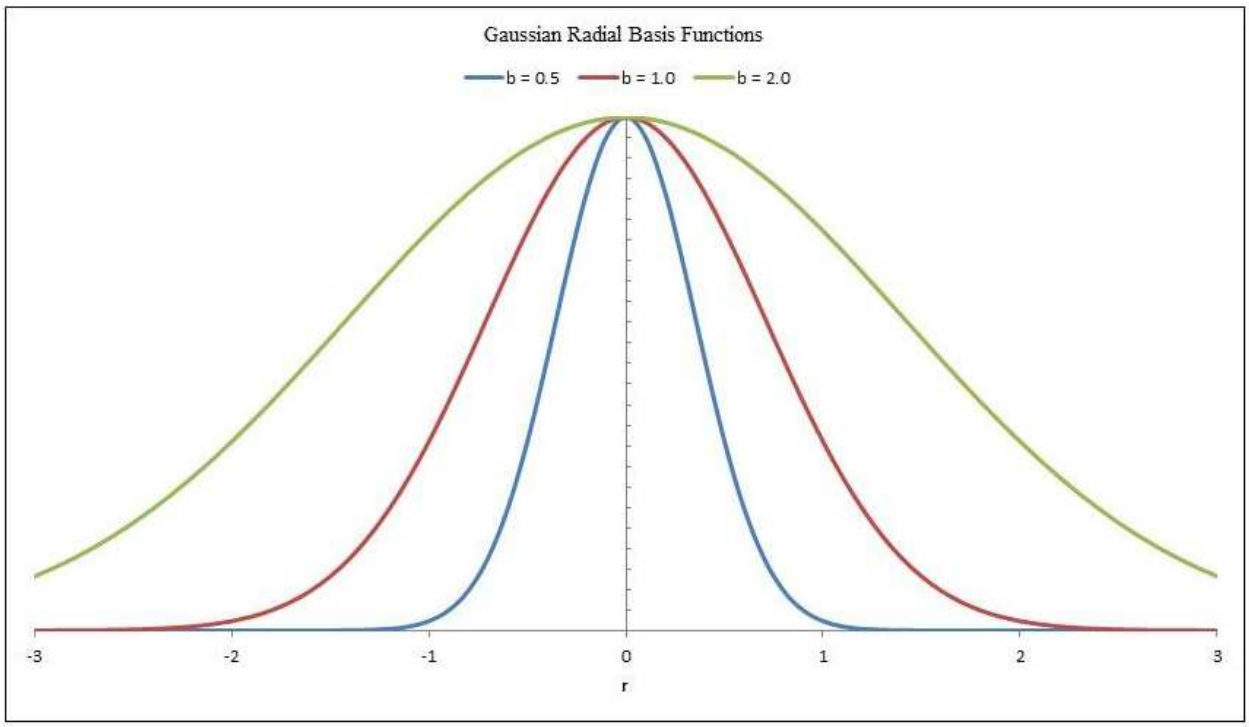

Figure 2.1: Gaussian RBFs with different shape factors

2.2.2 RBF Interpolation

RBF interpolation is a method of using RBFs to form a surrogate model for an unknown function that is known on a subset of the function's range [8]. That is, given an unknown function $f\colon X \to Y$, create a function $g\colon X \to Y \mid g(x) \approx f(x)$. The function g is created using subsets $X', Y' \mid f(X') = Y'$.

The function g in this case is given using a RBF, here denoted by φ. The value of the RBF is evaluated at the distance between the point of interpolation, p, and known points, p'. These values are multiplied by a set of weights, w, to come up with an approximated value of the function to estimate. This can be seen in equation (2.1).

$$g(p) = \sum_{i=1}^{n} w_i \varphi(p - p_i') \tag{2.1}$$

By enforcing the interpolating condition, namely $g(p') = u$ where u is the sampled function value at the known point p', the weights can be determined by creating a system of equations and solving. Equation (2.1) is solved for each known p' and its corresponding u as seen in (2.2).

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1n} \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \varphi_{n1} & \varphi_{n2} & \cdots & \varphi_{nn} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{bmatrix} \tag{2.2}$$

If the RBF is positive definite, then this system can be shown to be non-singular, that is there is a single solution for all the weights. However, the system does become ill-conditioned, i.e. it becomes close to or nearly singular, if the values of φ are too close together. This can be solved by increasing the value of the shape factor b until the system is better conditioned. This increase in shape factor generally has a corresponding decrease in accuracy since larger weights are given to known points further from the point of interest [6]. The magnitude of the decrease in accuracy will vary from problem to problem.

Further trouble arises if the RBF isn't positive definite. In this case a solution to the the system in (2.2) is not guaranteed. However, by including a linear polynomial with the approxima-

tion function, the system can once again be guaranteed to have a solution [8]. This new form of the function g can be seen in (2.3).

$$g(p) = \sum_{i=1}^{n} w_i \varphi(p - p_i') + c_0 + c_1 x + c_2 y + c_3 z \tag{2.3}$$

Where x, y, and z are the coordinates of the point p.

This equation has 4 more unknowns. In order to solve for the weights and the additional polynomial coefficients, several constraints are added to the weights to generate 4 more functions. These can be seen in equation (2.4)

$$\sum_{i=1}^{n} w_i = 0, \sum_{i=1}^{n} w_i x_i = 0, \sum_{i=1}^{n} w_i y_i = 0, \sum_{i=1}^{n} w_i z_i = 0 \tag{2.4}$$

This leads to a new system of linear equations as seen in equation (2.5)

$$\begin{bmatrix} \varphi_{11} & \varphi_{12} & \cdots & \varphi_{1n-1} & \varphi_{1n} & 1 & x_1 & y_1 & z_1 \\ \varphi_{21} & \varphi_{22} & \cdots & \varphi_{2n-1} & \varphi_{2n} & 1 & x_2 & y_2 & z_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \varphi_{n1} & \varphi_{n2} & \cdots & \varphi_{nn-1} & \varphi_{nn} & 1 & x_n & y_n & z_n \\ 1 & 1 & \cdots & 1 & 1 & 0 & 0 & 0 & 0 \\ x_1 & x_2 & \cdots & x_{n-1} & x_n & 0 & 0 & 0 & 0 \\ y_1 & y_2 & \cdots & y_{n-1} & y_n & 0 & 0 & 0 & 0 \\ z_1 & z_2 & \cdots & z_{n-1} & z_n & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \\ c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.5}$$

Once the polynomial coefficients and weights are known, the displacement of the point of interest, u_p, can be solved for using (2.3). The weights and coefficients need not be recalculated for further interpolations but can be reused for all points of interest in the domain. This means that while the first interpolation comes at a cost of $O(n^3)$, further interpolations are much cheaper at a cost of $O(n)$.

2.2.3 RBF Mapping Procedure

To perform a hot-to-cold mapping using RBF interpolation, Persson and Runsten [6] first associated each point in the mesh with a displacement value in each of the euclidean directions.

Then, for each geometry point that they needed to map, a small number of the mesh points, n, was chosen that surrounded the geometry point. A RBF interpolation was then performed for each displacement direction. If during the RBF interpolation the RBF matrix in equation (2.5) was ill-conditioned, either a smaller number of the mesh points was used or the shape parameter was increased to stabilize the matrix. These parameters, n and b, were altered as little as possible in order to maintain accuracy. This procedure can be seen in Figure 2.2.

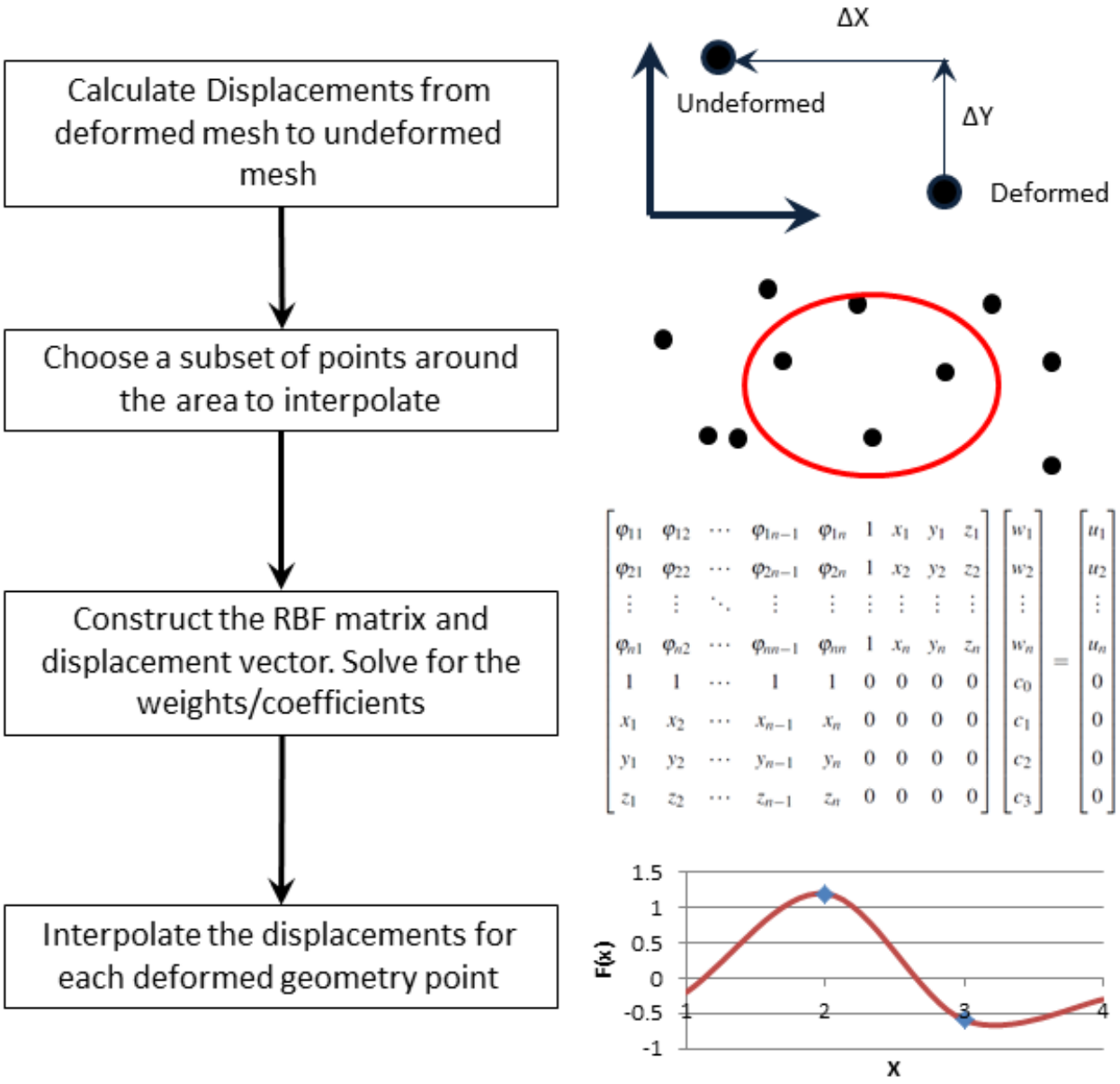

Figure 2.2: A flow chart illustrating the procedure for hot-to-cold mapping using radial basis function interpolation

2.2.4 RBF Method Results

RBF interpolation as a mapping scheme was found to be very accurate. When performed on a rotor of the 2^{nd} stage of an axial compressor, all errors were found to be less than 0.7% of the smallest displacement of the mesh nodes. They also found that if they decreased the shape factor to yield better accuracy (as opposed to increasing it for better stability) that a loss of smoothness was observed in the blade geometry.

2.3 Free Form Deformation

Sederberg and Parry [9] created a method for deforming three-dimensional objects in a free-form manner. This method consists of surrounding the model with a grid of control points. This control grid is then deformed, deforming the object with it in a smooth and continuous manner. This method is now used commonly in many engineering methods such as shape optimization in aerodynamics [10, 11]. Free-form deformation is commonly referred to as FFD.

2.3.1 FFD Method

FFD works by deforming a model using parametric basis functions. Traditionally, the basis function of choice is the Bernstein basis. The Bernstein basis is a basis for polynomials defined parametrically (called Bezier curves) for different polynomial degrees as in equation (2.6). The Bezier curve is constructed by placing control points, P, in space. If the control points are constructed in a grid in two dimensions, a parametric surface can be formed. This is then extended further to three parametric directions creating a trivariate parametric basis that defines a volume

$$P(t) = \sum_{i=0}^{n} B_i^n(t) P_i \tag{2.6}$$

where P_i is the coordinates of the control point and B_i^n is the basis function defined by equation (2.7)

$$B_i^n(t) = \binom{n}{i} (1-t)^{n-i} t^i, \; i = 1, 2, \ldots, n \tag{2.7}$$

Bernstein polynomials possess several desirable properties. First, these polynomials are smooth and infinitely differentiable. Second, the polynomials are coordinate system independent, meaning that they remain the same no matter the reference origin. Third, the polynomials must remain inside the convex hull of the control points. The polynomial also has the variation diminishing property which means that it can oscillate no more than the set of control points themselves [12].

The FFD method begins by constructing a parallelpiped of control points that encloses the body to be deformed. The grid of control points is layed out in three parametric directions usually referred to as the s, t, and u directions. Conventionally these align with the global x, y, and z directions. For every point in the body that then needs to be deformed, an s, t, and u coordinate is calculated, usually a number between the maximum and minimum values of s, t, and u (usually between 0 and 1). These values are used later to reposition the body's points.

The user then moves the control points to wherever is desired to deform the enclosed body. The s, t, and u coordinates of the model are then input into the trivariate Bernstein basis functions to calculate the new locations of the points. The points are then moved to their newly calculated locations. Since the basis functions are smooth, the resulting deformation is likewise smooth as well as possessing the other properties of the Bernstein basis. This method can be seen in Figure 2.3.

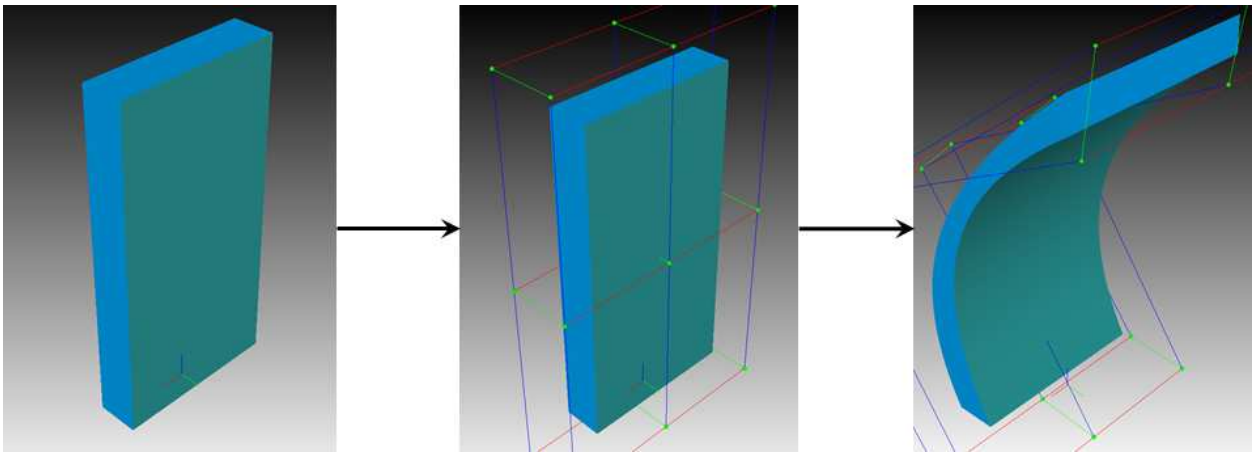

Figure 2.3: The process of free-form deformation. The model is imported, enclosed in a control grid, then deformed

2.3.2 Applying FFD to Hot-to-Cold Mapping

The general method for using FFD in hot-to-cold mapping is fairly straightforward. First, enclose the deformed blade geometry and deformed mesh with control points. Next, move the control points until the deformed mesh points are displaced as close as possible to their undeformed mesh counterparts. This process can be done automatically by means of an optimization technique such as least squares to minimize the distance between the deformed hot mesh points and cold mesh points. Lastly, calculate all the deformed geometry points' new locations using the basis functions.

This method is both simple, fast, and easy to implement. However, using simple FFD has many drawbacks. In order to increase the accuracy of an FFD mapping (accuracy being defined as the difference between a point's mapped location and its actual location), the user is forced to increase the order of the polynomial bases. This leads to increased computational time and increased oscillatory behavior (waviness) of the body. Both of these situations are undesirable.

2.3.3 Sculptor™and Arbitrary Shape Deformation

Sculptor™ [13] is a software made by Optimal Solutions that is used in engineering shape optimization. It has improved upon the free-form deformation technique in several ways. One improvement is to use a trivariate B-spline basis in place of the simple Bernstein basis.

B-splines are piecewise polynomials that are C_{n-1} continuous where n is the order of the polynomial used. The curve is a set of smaller Bezier curves that are all as continuous as possible with their neighbors. In the case of quadratic curves, this means that each curve is tangent to its neighbors. For cubic curves, this means that each curve is both tangent and curvature continuous to its neighbors. This pattern continues for all orders of the B-spline [12].

B-splines allow for greater local control of the shape of the curve because individual control points only effect the underlying Bezier curves in the locality of the point. Those Bezier curves farther away from the displaced control point are completely unaffected. The added benefit is that extra control points can be added without increasing the degree of the polynomial basis. Instead, added control points add extra Bezier curves to the definition of the B-spline.This helps to prevent high-order oscillations while adding greater local control to the curve.

The second major improvement is that sculptor allows for arbitrary grids of control points, not just simple parallelpipeds. Any shape can be done including curves and bends in the grid. This allows for even greater local control with the same number of control points as in a parallelpiped by defining the grid much closer to the surface of the part. These two features are what Sculptor™refers to as arbitrary shape deformation, also referred to as ASD.

Sculptor™uses these techniques along with the GRG optimization method (a gradient-based optimization method) to deform the hot shape into the desired cold shape. The optimizer moves the control points until the difference between the hot nodes and their corresponding cold nodes is within a specified tolerance. An arbitrarily accurate mapping can be achieved by refining the control grid to whatever level of accuracy is needed. This increase in accuracy comes at the cost of increased computational time, especially considering the expensive gradients that need to be calculated for the GRG method. Another cost of increased accuracy is an increase in oscillation of the basis functions. ASD helps to mitigate this effect when compared to FFD, but does not remove it completely.

CHAPTER 3. METHOD

3.1 Kriging Theory

Kriging is a process used in the field of Geostatistics to estimate spatial data. The method was developed by Georges Matheron for use in the mining industry where it was and still is commonly used to estimate ore distributions from a small set of boreholes. Kriging is now commonly used in many fields for many different purposes in many different fields such as hydrology, pollution, soil science, forestry, epidemiology, weather prediction, optimization, etc [14].

As with RBF interpolation, Kriging attempts to estimate a function $f\colon X \to Y$ with a surrogate function g that is as close to f as possible. This requires a two-step process. In the first step, the autocorrelation of f must be sampled and approximated with a function called the variogram. In the second step, the variogram is used to minimize the variance of the autocorrelated data providing a set of weights for interpolation in the form of $g(x) = \sum_{i=1}^{n} w_i(x)u_i$.

3.1.1 Random Functions

Kriging is based off of minimizing the error of what is referred to as a random function. Random functions are functions of two variables, denoted as $Z(x,\omega)$. Each of the sections $Z(x,\cdot)$ is a random variable (where $Z(x,\cdot)$ indicates that a value of ω is chosen as constant) and each of the sections $Z(\cdot,\omega)$ is referred to as a realization of the random function. The realization of the random function is referred to in the lowercase as $z(x)$ [15].

3.1.2 Autocorrelation and the Variogram

Autocorrelation is defined as the correlation between elements of a series and others from the same series that are separated by a given interval [16]. When applied to a function f, autocorrelation means how correlated are $f(x)$ and $f(x+h)$ where x is the original input to the function

and h is some separation distance. Most functions have some degree of autocorrelation. The higher the degree of autocorrelation, the less the change in the function in space. The lower the degree of autocorrelation, the greater the change in the function in space.

Three different but related functions are used to quantify the autocorrelation: the covariance, the correlogram, and the semivariogram. The covariance function is a direct measure of the autocorrelation. It is defined in equation (3.1). Under conditions of second-order stationarity (which will be discussed later), this function begins at the origin with the value of the global variance and monotonically decreases until it reaches a value of zero. The separation distance at this value is referred to as the range.

$$C(h) = \frac{1}{n}\sum_{i=1}^{n} z(x_i) \cdot z(x_i + h) - m_0 \cdot m_{+h} \tag{3.1}$$

The correlogram is related to the covariance function. They both have the same general trends (monotonically decreasing and reaching the value of zero at the range), but the correlogram has been scaled as seen in equation (3.2).

$$\rho(h) = \frac{C(h)}{\sqrt{\sigma_0 \cdot \sigma_{+h}}} \tag{3.2}$$

Here, the subscripts 0 and $+h$ refer to the values at x_i and at $x_i + h$ respectively.

The semivariogram, more commonly called the variogram, is different from the correlogram because it measures dissimilarity instead of similarity. The equation for the variogram can be seen in (3.3). Under second-order stationarity, this function begins at a value of zero and increases monotonically until it reaches the global variance at the range. The global variance value for a variogram is generally referred to as the sill value.

$$\gamma(h) = \frac{1}{2n}\sum_{i=1}^{n} [z(x_i + h) - z(x_i)]^2 \tag{3.3}$$

For a function to be second-order stationary, the property mentioned earlier, it must have a first moment and autocovariance that do not vary. In the context of this paper, it means that the mean and variance are spatially constant. Under these conditions, the covariance, correlogram, and variogram are related by equation (3.4). These relations show that the correlogram is simply the

covariance normalized by the covariance, and the variogram is just a reflection of the covariance. This is shown in Figure 3.1. While these conditions are generally violated to a degree, the processes built upon these assumptions will provide a good estimate so long as the deviation is not too large.

$$\rho(h) = \frac{C(h)}{C(0)}$$
$$\gamma(h) = C(0) - C(h) \tag{3.4}$$

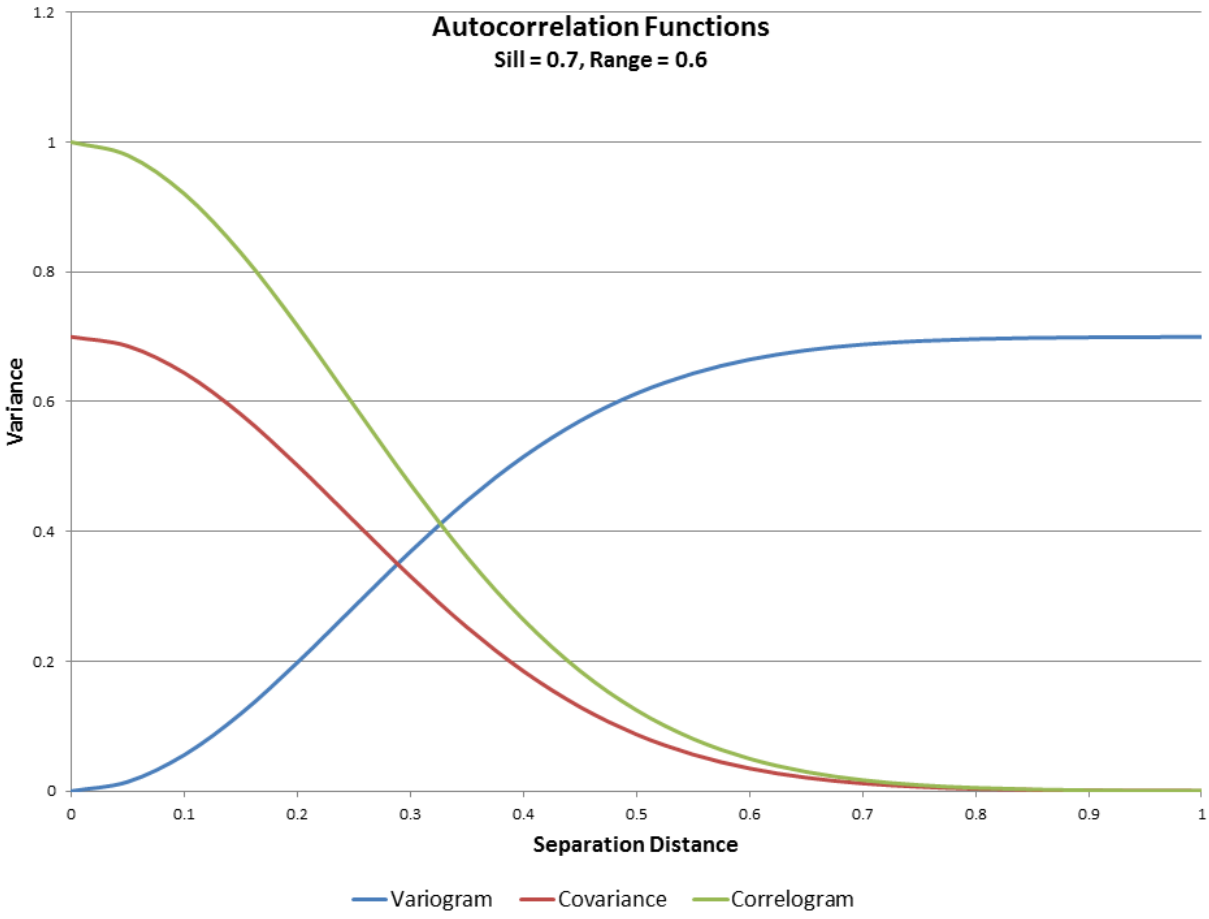

Figure 3.1: Examples of autocorrelation functions. The covariance and correlogram show relatedness whereas the variogram shows un-relatedness.

While any of the three functions can be used to build up an interpolation scheme, the variogram is generally chosen for several reasons that provide an advantage. First, the variogram

averages squared differences. This tends to filter out effects from a spatially varying mean (a condition that normally violates second-order stationarity). Second, the variogram can be used on a weaker condition called the intrinsic hypothesis where only the values of the first differences, that is $z(x) - z(x+h)$, are second-order stationary. In this case, the covariance and correlogram are undefined (since the variance increases continuously) whereas the variogram is.

3.1.3 Empirical Variograms

While the variogram can very rarely be found analytically, it can be sampled at discrete locations. For all known points, the entire set of possible pairs is generated. For each pair, the semivariance is calculated and plotted. This plot is commonly referred to as the variogram cloud as it contains a large number of data points in a very scattered fashion. This cloud is a rough estimate of the variogram itself, though it contains a considerable amount of noise [15].

Generally, to reduce the noise in the variogram cloud, the data is binned together and averaged into ranges of separation distance. Therefore, instead of a cloud of points at each specific h, the bins would represent all points in a range of $h \pm \delta h$. This tends to smooth out the noise in the variogram cloud (though noise is still present in this representation). This binned collection is commonly referred to as the empirical variogram. Figure 3.2 shows the variogram cloud and the empirical variogram for the function $f(x) = e^x$. The overlay in Figure 3.2 shows the scale of the empirical variogram as well as the magnitude of the noise in the variogram cloud.

The empirical variogram gives a good insight into the shape and behavior of the variogram. However, the empirical variogram cannot be evaluated at arbitrary locations. This requires a function to be fitted to it first [17].

3.1.4 Variogram Anisotropy

Variograms are generally considered to be isotropic meaning that the variogram value $\gamma(h)$ only depends on the magnitude of h i.e. the separation distance between points. When the variogram exhibits anisotropic behavior, $\gamma(h)$ will vary with the direction of h as well as its magnitude. This is usually seen as a change in the sill or the range of the variogram as seen in Figure 3.3.

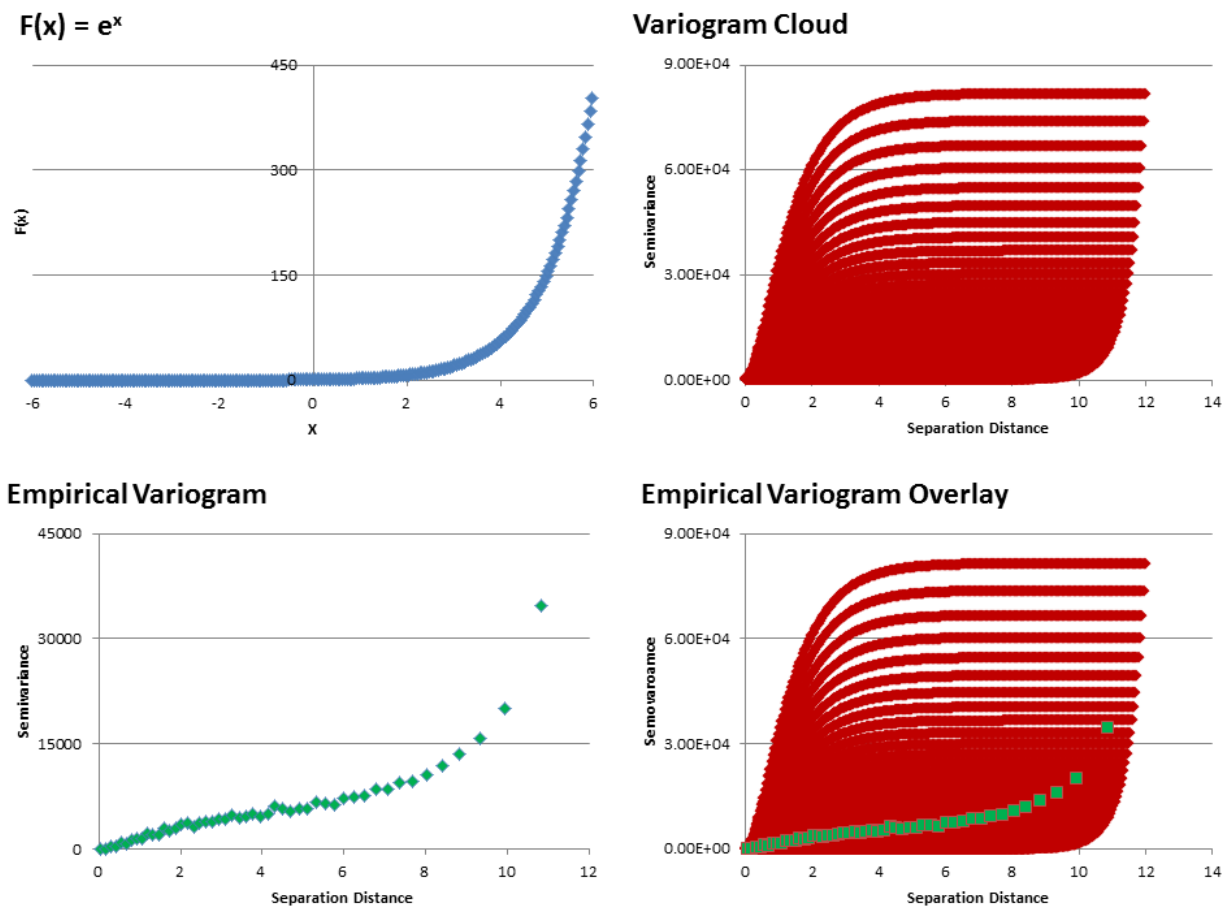

Figure 3.2: The process of sampling a variogram. The function in the top left is sampled. The semivariance is calculated for each possible pair creating the variogram cloud in the top right. The cloud is averaged into bins forming the empirical variogram in the bottom left. The bottom right shows an overlay of the two.

Some special cases of anisotropy can be modeled without too much difficulty such as geometric anisotropy and zonal anisotropy [15]. However, some cases of anisotropy are non-trivial to model. This problem is compounded when moving from two dimensions to three dimensions because of the additional directions for anisotropic behavior.

While an isotropic variogram is not always the correct model to use, Kriging generally yields good results despite having a less than ideal variogram model [15, 18]. Therefore, in order to simplify analysis, this work will only focus on the isotropic case, though the anisotropic case may be an area of further research.

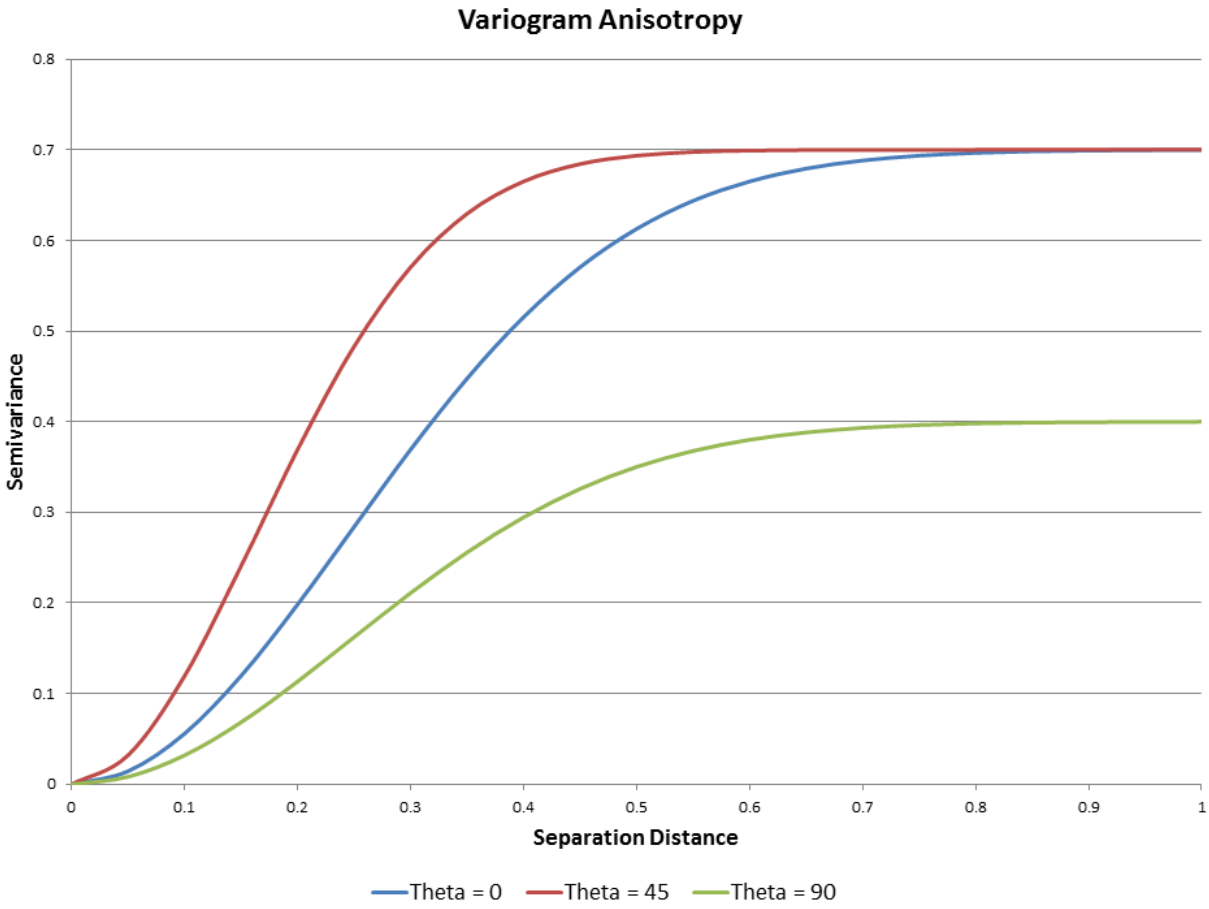

Figure 3.3: A 2-D variogram sampled along different directions. This variogram exhibits anisotropy where the range and sill vary depending on the sampling direction.

3.1.5 Variogram Fitting

Although a function model need only be non-negative definite to be considered a licit variogram, some shapes of variograms appear very often. A set of function models that are guaranteed to be non-negative definite that match these shapes has been developed and are commonly used. Some of these common functions can be seen in Figure 3.4. The definition for these functions can be seen in table 3.1. The sill, here is denoted by s and the range by r.

The power model is important for whenever the intrinsic hypothesis is assumed because it models the continuously increasing variance with increased separation distance. The cardinal sine model is important for functions that have periodic behavior, whereas the other models with sill parameters (spherical, exponential, and Gaussian models) are common among many non-periodic

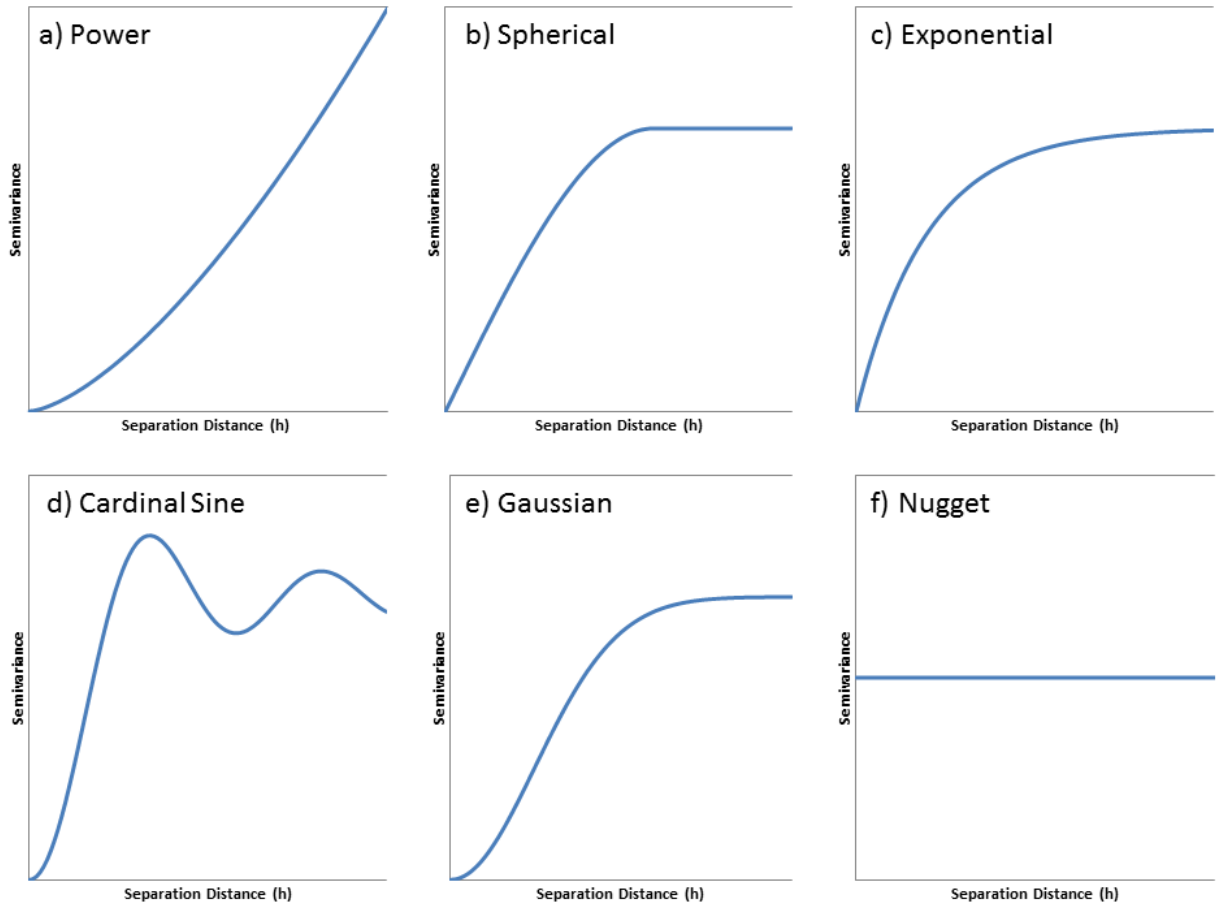

Figure 3.4: Different variogram models. From top left to bottom right: *a*) power, *b*) spherical, *c*) exponential, *d*) cardinal sine, *e*) Gaussian, *f*) nugget

Table 3.1: Equations and uses of different variogram models

Type	Equation	Common Uses
Power	$\gamma(h) = ah^b \quad where\ 0 < b < 2.0$	Modeling mild trends
Spherical	$\gamma(h) = \begin{cases} s \cdot \left[1.5(\frac{h}{r}) - 0.5(\frac{h}{r})^3\right] & if\ h \leq r \\ s & if\ h > r \end{cases}$	Sharp, linear behavior near origin
Exponential	$\gamma(h) = s \cdot \left[1 - e^{(\frac{-3h}{r})}\right]$	Sharp, linear behavior near origin
Cardinal Sine	$\gamma(h) = s \cdot \left[1 - r \cdot \frac{sin(\frac{h}{r})}{h}\right]$	Periodic behavior
Gaussian	$\gamma(h) = s \cdot \left[1 - e^{(\frac{-3h^2}{r^2})}\right]$	Smoothly vary behavior near origin
Nugget	$\gamma(h) = s$	Completely random behavior

functions. Each of these three have different behavior near the origin and at the range. The nugget model is simply used for functions that have no autocorrelation i.e. complete randomness.

Fitting a variogram has few rules to it; in fact, it is generally thought of as more of an art than a science. Usually a variogram model is chosen that best resembles the empirical variogram, then parameter values are manipulated until a "good-looking" fit is achieved. If a single variogram model does not yield good results, another model can be added to the current one. This is possible because the sum of any two licit variograms is also a licit variogram [17].

Some have tried to fit variograms using a scientific approach such as least squares estimation. These methods often require non-linear solution techniques—such as the Levenberg-Marquardt algorithm—since many variograms are not linear in their parameters. These methods have had varying levels of success in fitting variogram models to empirical variograms [15, 19]. This research creates a new method for this that will be outlined later.

3.1.6 Simple Kriging

Derivations for Simple and Ordinary Kriging can be found in many places [20, 21]. This work draws heavily on the derivation done by Chiles and Delfiner [15] for the Simple Kriging derivation found here and the Ordinary Kriging derivation in the following sections, though some of the steps have been simplified. The author suggests reading their derivation if greater clarity is needed.

In Simple Kriging, we assume that the mean of a random function $Z(x)$ is known and constant (that is constant in ω). We seek a function g that approximates the random function $Z(x)$ of the form:

$$g(x) = \sum_{i=1}^{n} w_i u_i + w_0 \tag{3.5}$$

where w_0 is a weight independent of the known points, and w_i is a weight corresponding to the i^{th} known point. We seek to minimize the mean square error of this function which can be written as:

$$E(g(x) - Z_0)^2 = Var(g(x) - Z_0) + [E(g(x) - Z_0)]^2 \tag{3.6}$$

We begin by seeking the value of w_0. The variance of the function does not depend on it's location. This means that it does not depend on w_0. Therefore, the bias term, $[E(g(x)-Z_0)]^2$, is the only term that is affected by w_0. The minimum of the mean square error will then be one where w_0 will minimize the bias term, that is:

$$w_0 = m_0 - \sum_{i=1}^{n} w_i m_i \tag{3.7}$$

This implies that the form of the estimating function is:

$$g(x) = m_0 + \sum_{i=1}^{n} w_i (Z_i - m_i) \tag{3.8}$$

To simplify, we create a variable $b(x) = Z(x) - m(x)$ to represent the zero-mean. We then estimate this variable using a linear sum $b^* = \sum_{i=1}^{n} w_i b_i$. In the end, we add the mean back in. This means that having a zero-mean and a known mean are the same so long as the non-zero mean is added back in at the end. Now, expanding out the variance term of the mean square error in terms of the covariance between two points, here denoted by σ_{ij}

$$E(g(x)-Z_0)^2 = \sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j \sigma_{ij} - 2\sum_{i=1}^{n} w_a \sigma_{a0} + \sigma_{00} \tag{3.9}$$

By taking the partial derivatives with respect to w_i and solving the root of the system, the Simple Kriging system of equations is found as in (3.10).

$$\sum_{j=1}^{n} w_j \sigma_{ij} = \sigma_{i0}, \;\; i = 1,2,\ldots,n \tag{3.10}$$

which means the variance is:

$$\sigma_{00} - \sum_{i=1}^{n} w_i \sigma_{i0} \tag{3.11}$$

The covariance function is commonly used for this method, but the variogram can be substituted if necessary by using the relationships shown in (3.4) [15].

3.1.7 Simple Kriging Example

As an example of Simple Kriging, assume the case of two known points in one dimension, (2.0, 1.2) and (3.0, -0.6). While a variogram can not be fitted to this data (only one point exists in the variogram cloud), let's assume that a Gaussian variogram with range = 1.0 and sill = 0.7 presents a good fit to the data. The covariance function in this case is simply $0.7 - \gamma(h)$. We begin by constructing the covariance matrix:

$$\begin{bmatrix} 0.7 & 0.061 \\ 0.061 & 0.7 \end{bmatrix} \tag{3.12}$$

where the inverse is:

$$\begin{bmatrix} 1.432 & -0.071 \\ -0.071 & 1.432 \end{bmatrix} \tag{3.13}$$

This matrix is then used to solve for the weights for each interpolation. To interpolate at the point x = 2.25, we construct a vector of covariance values:

$$\begin{bmatrix} 0.58 \\ 0.129 \end{bmatrix} \tag{3.14}$$

where the first value is the covariance with the first known point and the second value is the covariance with the second known point. Notice that the first value is larger than the second value indicating that the first point is more related to our point of interest (x = 2.25) than the second point.

Now the weights are solved for by multiplying the covariance matrix inverse by the covariance vector yielding:

$$\begin{bmatrix} 1.432 & -0.071 \\ -0.071 & 1.432 \end{bmatrix} \begin{bmatrix} 0.58 \\ 0.129 \end{bmatrix} = \begin{bmatrix} 0.821 \\ 0.144 \end{bmatrix} \tag{3.15}$$

Multiplying these weights by the y-values of the known points and adding back in the known mean (in this case 0.0) yields the Simple Kriging estimate, $f(2.25) \approx 0.8998$. The complete Simple Kriging solution for this two-point example can be seen in Figure 3.5.

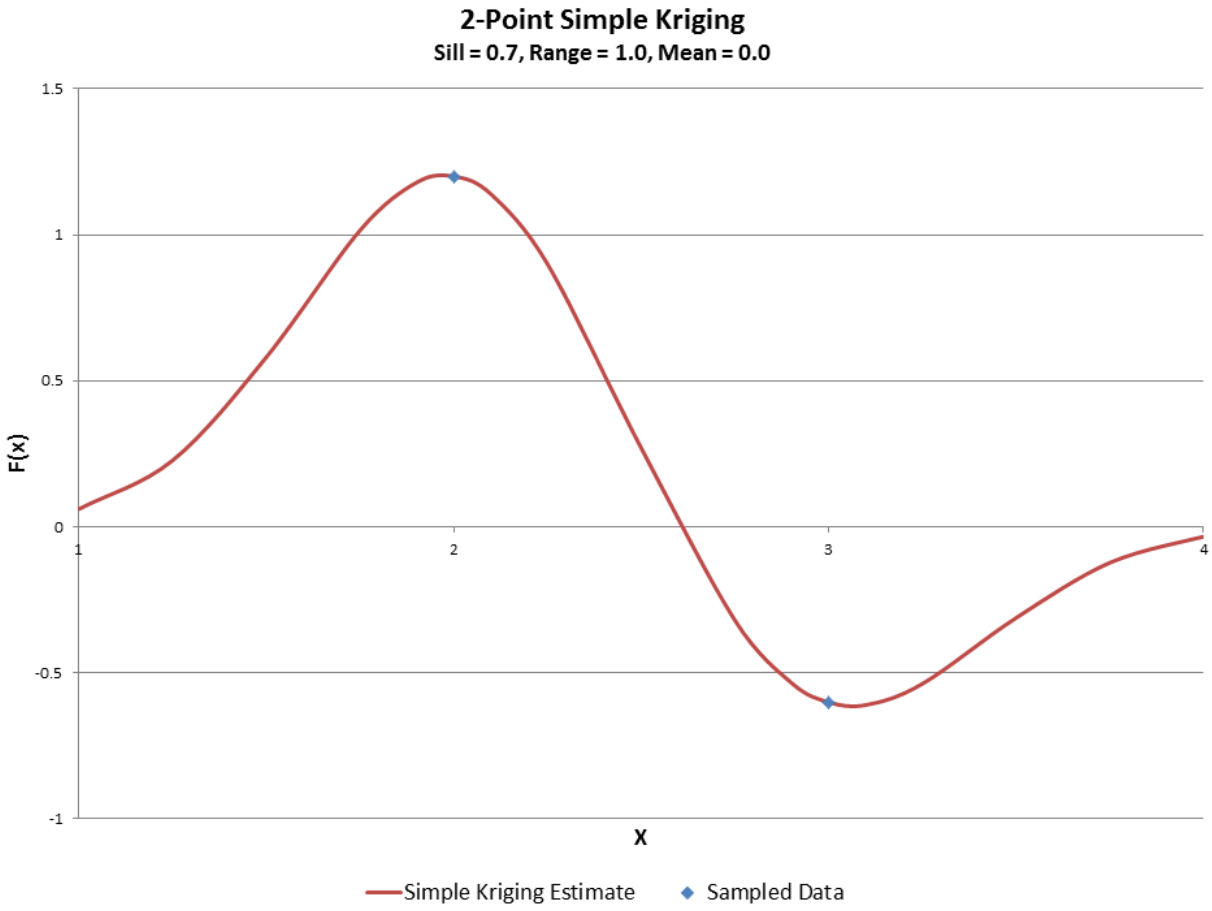

Figure 3.5: The Simple Kriging Solution for a two-point example and a Gaussian Variogram (range = 1.0, sill = 0.7). The estimate is smooth, interpolates at known points, and tends toward the mean (in this case 0.0) at distances far from known data.

3.1.8 Ordinary Kriging

This derivation, once again, draws heavily from the derivation by Chiles and Delfiner [15] with simplifications and notational differences that the author found appropriate.

In Ordinary Kriging, the same general process for deriving the weights is used, but this time the mean is not assumed to be known (though is still constant in ω). Instead, the weights are assumed to sum to 1, that is $\sum_{i=1}^{n} w_i = 1$ and w_0 is assumed to be zero. This minimizes the bias term and we are left to minimize the variance as in (3.9). To solve this equation with the constraint on the weights mentioned, Lagrange multipliers are generally used (3.16).

$$Q = Var(g(x) - Z_0) + 2\lambda(\sum_{i=1}^{n} w_i - 1) \tag{3.16}$$

By taking the partial derivative of this system with respect to the weights and the Lagrange multiplier and solving for the roots results in the linear system shown in equation (3.17).

$$\begin{aligned} \sum_{j=1}^{n} w_j \sigma_{ij} + \lambda = \sigma_{i0}, \;\; i = 1, 2, \ldots, n \\ \sum_{i=1}^{n} w_i = 1 \end{aligned} \tag{3.17}$$

Which when put in terms of the variogram γ is:

$$\begin{aligned} \sum_{j=1}^{n} w_j \gamma_{ij} - \lambda = \gamma_{i0}, \;\; i = 1, 2, \ldots, n \\ \sum_{i=1}^{n} w_i = 1 \end{aligned} \tag{3.18}$$

This means the variance can be estimated by:

$$\sum_{i=1}^{n} w_i \gamma_{i0} + \lambda \tag{3.19}$$

3.1.9 Ordinary Kriging Example

Once again we will cover a simple, two-point example. Our known data is (2.0, 1.2) and (3.0, -0.6). A Gaussian variogram with range = 1.0 and sill = 0.7 is also assumed as before. We begin by constructing the variogram matrix:

$$\begin{bmatrix} 0.0 & 0.665 & -1.0 \\ 0.665 & 0.0 & -1.0 \\ 1.0 & 1.0 & 0.0 \end{bmatrix} \tag{3.20}$$

where the inverse is:

$$\begin{bmatrix} -0.752 & 0.752 & 0.5 \\ 0.752 & -0.752 & 0.5 \\ -0.5 & -0.5 & 0.333 \end{bmatrix} \tag{3.21}$$

This matrix is then used to solve for the weights for each interpolation. To interpolate at the point x = 2.25, we construct a vector of variogram values:

$$\begin{bmatrix} 0.1197 \\ 0.5705 \\ 1.0 \end{bmatrix} \tag{3.22}$$

where the first value is the variogram with the first known point and the second value is the variogram with the second known point. The third value is simply 1.0 to account for the weights summing to 1.0. Notice that the second value is larger than the first value indicating that the second point is less related to our point of interest (x = 2.25) than the first point.

Now the weights are solved for by multiplying the variogram matrix inverse by the variogram vector yielding:

$$\begin{bmatrix} -0.752 & 0.752 & 0.5 \\ 0.752 & -0.752 & 0.5 \\ -0.5 & -0.5 & 0.333 \end{bmatrix} \begin{bmatrix} 0.1197 \\ 0.5705 \\ 1.0 \end{bmatrix} = \begin{bmatrix} 0.8389 \\ 0.1611 \\ -0.0125 \end{bmatrix} \tag{3.23}$$

Multiplying these weights by the y-values of the known points (and ignoring the Lagrange multiplier—the last value) yields the Ordinary Kriging estimate, $f(2.25) \approx 0.91$. The complete Ordinary Kriging solution for this two-point example can be seen in Figure 3.6. Notice that the behavior between the two known points is essentially the same as with Simple Kriging, but outside

these points the solution tends toward the average of the known data (0.3) instead of the known mean in Simple Kriging (0.0).

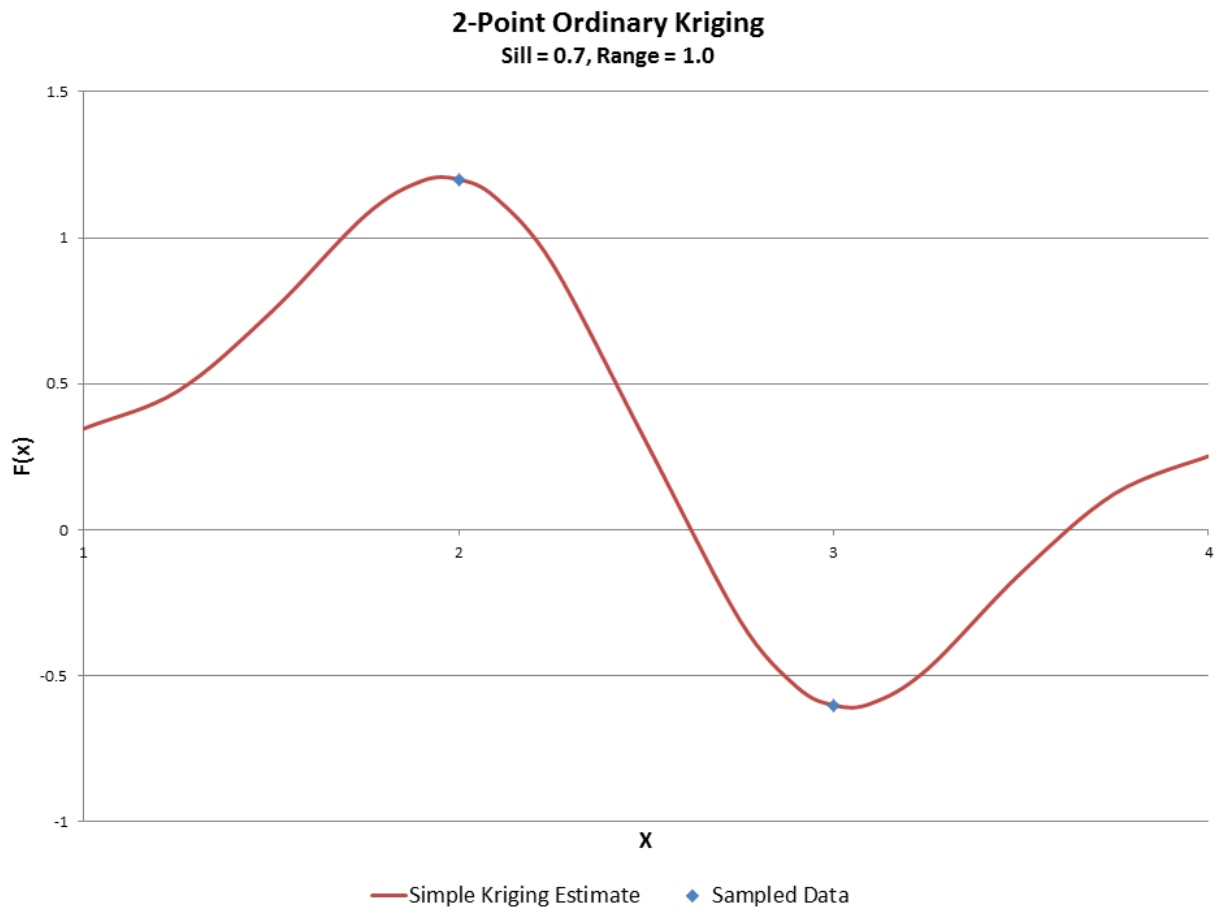

Figure 3.6: The Ordinary Kriging Solution for a two-point example and a Gaussian Variogram (range = 1.0, sill = 0.7). The estimate is smooth, interpolates at known points, and tends toward the mean of the known data (in this case 0.3) at distances far from known data.

3.1.10 Other Kriging Methods

Other methods for Kriging exist that work off of other assumptions. Universal Kriging, like Ordinary Kriging, assumes the mean is unknown, but that there is a strong presence of a drift meaning the mean varies strongly in space. The mean is calculated as a function of the points coordinates. Kriging with an external drift is another method that makes the same assumptions,

however the mean is calculated with external parameters, not its coordinates. Further forms such as Log-normal Kriging and Co-Kriging also exist [15].

3.2 Hot-to-Cold with Kriging

Performing hot-to-cold mapping using Kriging is very similar in the technique for hot-to-cold with radial basis functions. The method, however, is more complex because the variogram must be analyzed before Kriging can be performed. The following sections lay out how it has been accomplished in this research.

3.2.1 Data Organization

Because Kriging requires frequent look-up of mesh points in a close vicinity to the geometry point that is being mapped, a fast and efficient structure is required to store both sets of points. This is accomplished by implementing an octree. An octree stores points in a set of nested "boxes" represented by branch nodes and leaf nodes.

A branch node in an octree is represented as a box in three-space. This box further subdivides itself into eight new child nodes, each with an equal portion of the volume of the parent branch node. Each of these eight new nodes may also be a branch node, or they may be leaf nodes. The branch nodes are then used as an organizational system for all the nodes. For each query given to the octree, the query will then be redistributed to the corresponding nodes lower in the structure.

Leaf nodes are the bottom level of the structure. These also are represented as a box in three-space. However, leaf nodes, unlike branch nodes, do not have child nodes. Instead, they contain the points that need to be stored in list structures with a pre-set maximum capacity. Once the leaf nodes reach their maximum capacity, a split occurs where the leaf node turns itself into a branch node and redistributes its points into its newly created child nodes.

This nested method of storing points allows point look-up in $O(n + log_8(m))$ time, where n is the capacity of the leaf node and m is the total number of nodes, both branch and leaf, in the structure. A diagram of a quadtree, the two dimensional analog of an octree, can be seen in Figure 3.7.

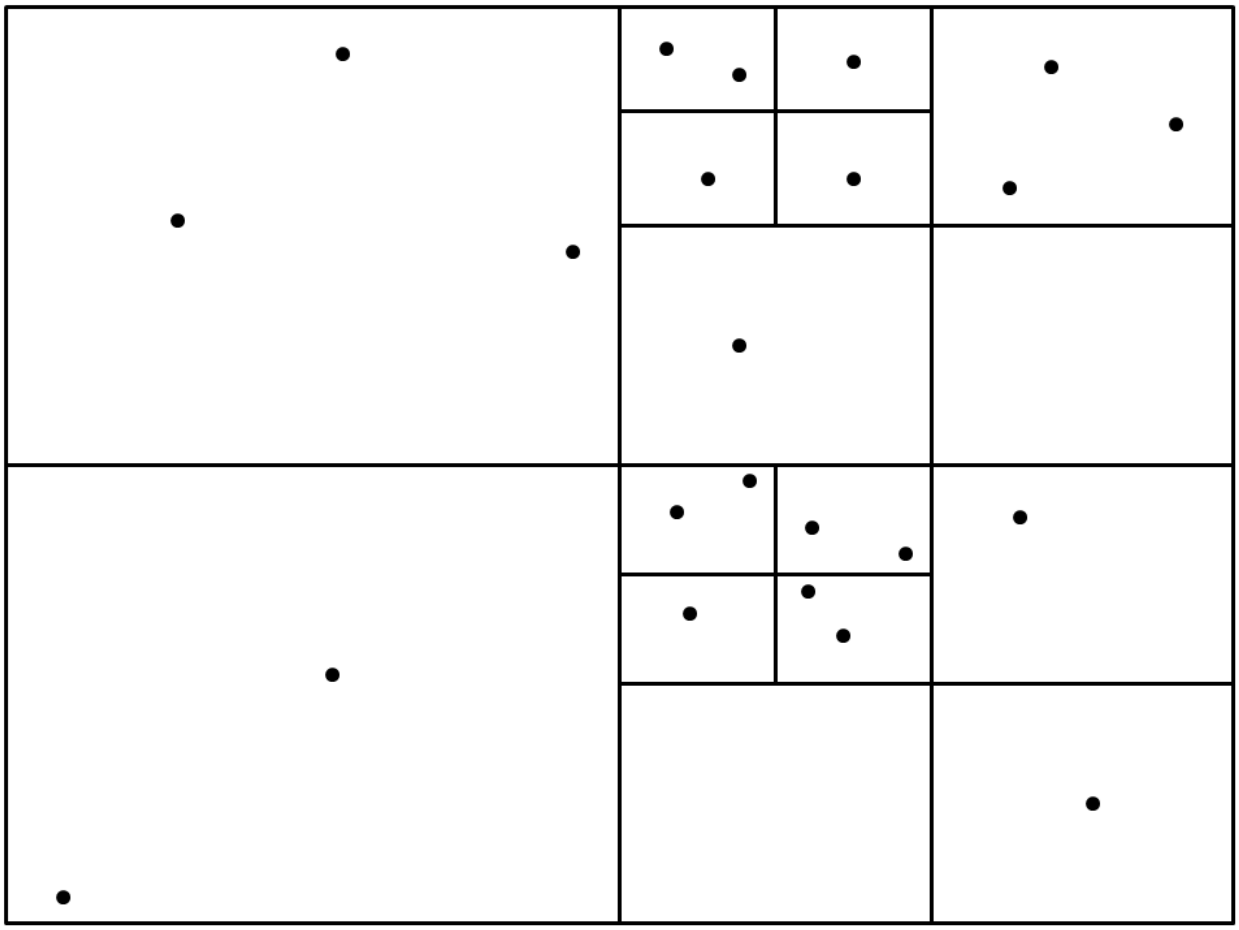

Figure 3.7: A visualization of a quadtree. Points are sorted into octants in an octree just as they are sorted into quadrants in a quadtree

For this research, the leaf nodes were given no limits on the number of geometry points that could held in their lists. The mesh point list, however, was constrained to only hold 10% of the number of points that would be used for the interpolation. This means that when Kriging was performed, mesh points from the leaf node itself as well as surrounding leaf nodes would be used. This gives a rough guarantee that the geometry points are within 10% of the center of the mesh point cluster used for Kriging.

3.2.2 Variogram Analysis and Fitting

In order to create an accurate and automated fit to the empirical variogram, a new method for variogram fitting was formulated. Instead of choosing a select variogram model from user input

and fitting the model using a least squares technique, this research created an optimization-based fit using a wide palette of different models.

The method works by creating a nested model variogram. Since the sum of any two licit variograms is also a licit variogram, many different models were simply summed together. This allowed for the computer to fit the variogram with little knowledge of the variogram's actual shape. The optimizer was then used to determine the coefficients for this nested model that would minimize the sum of the squared differences—essentially solving the least squares criterion numerically instead of analytically. Should the optimizer deem that one of the many individual models in the nested model was unnecessary, it would simply select the sill parameter for that model to be zero. The optimizer chosen for this research was a genetic algorithm because of it's robustness and ease of implementation.

The genetic algorithm was modeled after the implementation by Balling [22].The genetic algorithm works by creating chromosomes that are combinations of different parameters called genes. The genes in this case are floating point numbers that represent the coefficients of the nested variogram model. The genes are allowed to vary between an upper and lower bound specified.

A starting generation of chromosomes is generated by assembling the boundary values into as many combinations as possible and then scattering several randomly generated chromosomes into the population. At this point a fitness value is determined by calculating the sum of the squared differences between the empirical variogram values and those predicted by the chromosome's coefficients. The lower the sum of squares, the fitter the chromosome is considered.

A new generation of chromosomes is then created by selecting a father and mother chromosome and switching their genes into two new chromosomes. The father and mother are chosen by selecting a random group of chromosomes and selecting the fittest from this small group. Child chromosomes then will mutate some genes into a new value at random. Once mutated, the child chromosome's fitness is calculated as it was for its parents.

This process is repeated until a new generation the size of the old generation is created. The last step is for all the chromosomes, both old and new, to be sorted by their fitness value. The fittest chromosomes will then survive until the next iteration. This process of reproducing, mutating, and selecting is repeated until a sufficiently fit chromosome is found i.e. a good variogram model has been fitted. This process can be seen in Figure 3.8.

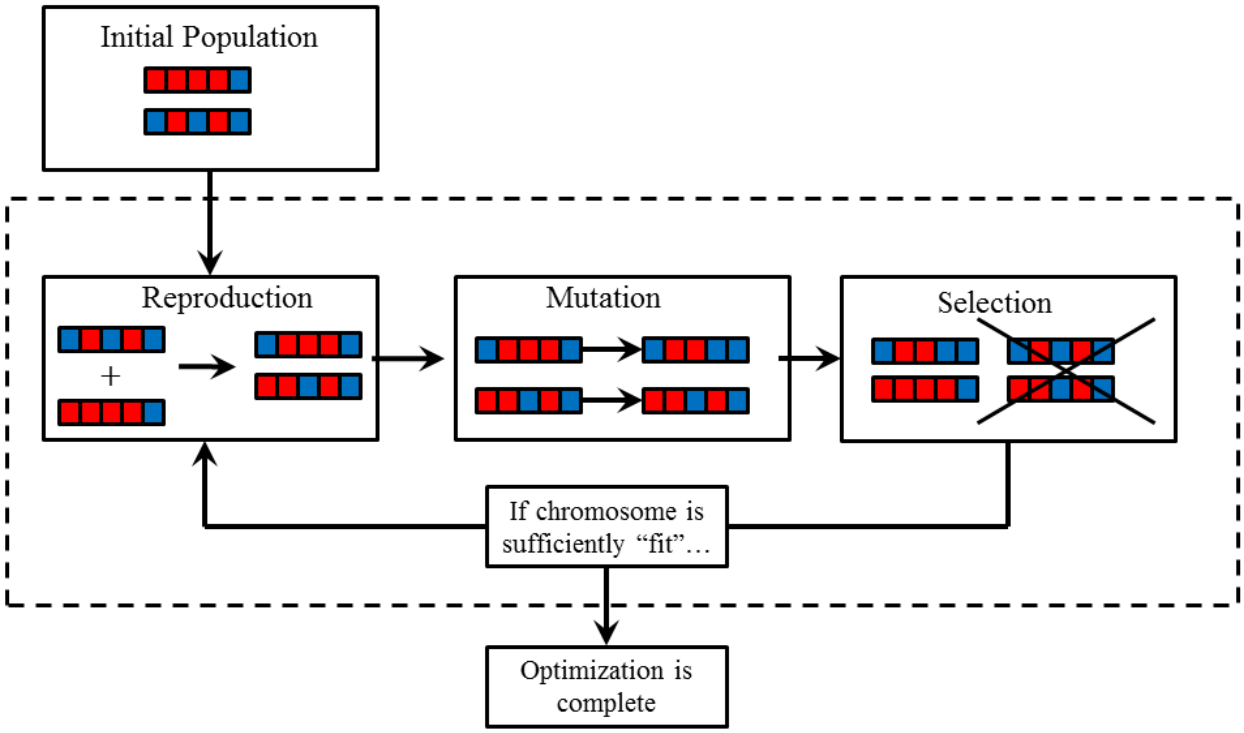

Figure 3.8: A flow chart illustrating the process of a genetic algorithm

With this fitting procedure, the program only needs to obtain a cloud of mesh points surrounding the area to be interpolated in. Pairs are then made and binned to create the empirical variogram. Once the empirical variogram is sampled, general bounds are placed on the coefficients of the nested model. The genetic algorithm is then run on the empirical variogram using these bounds. This returns the coefficients (the parameters of the chromosome) to use in the fitted variogram model.

3.2.3 Interpolating Displacements

Once the variogram was calculated for a given direction, estimating the displacement was a simple application of the Ordinary Kriging equations in (3.18). These equations were decomposed into their matrix form as seen in equation (3.24).

$$\begin{bmatrix} \gamma_{11} & \gamma_{12} & \cdots & \gamma_{1n} & -1 \\ \gamma_{21} & \gamma_{22} & \cdots & \gamma_{2n} & -1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \gamma_{n1} & \gamma_{n2} & \cdots & \gamma_{nn} & -1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \\ \lambda \end{bmatrix} = \begin{bmatrix} \gamma_{01} \\ \gamma_{02} \\ \vdots \\ \gamma_{0n} \\ 1 \end{bmatrix} \tag{3.24}$$

For each leaf node in the octree and for each direction, the matrix of variogram values was assembled. If the matrix was ill-conditioned then singular value decomposition was calculated for the matrix, otherwise the LU decomposition was used as the default method. Then, for each point, the variogram vector was constructed and the system was solved for the weights. Next, the weights were used to calculate both the displacement as well as the variance associated with it (as a local measure of accuracy/confidence). While the decomposition is $O(n^3)$, solving the linear system is only $O(n^2)$. Therefore, the decomposition of the variogram matrix is reused for interpolating each geometry point to save computational time.

3.2.4 Method Summary

The many parts of the process of geometry mapping with Kriging can now be summarized into a simple system that can be implemented on a computer. This work implemented the system using Java™, though any coding language should be just as effective. This process, as explained in the following paragraphs, can be seen in Figure 3.9.

First, mesh data is read into the computer. Deformed mesh points are associated with their displacements (that is, the distance between the deformed mesh point and its corresponding undeformed mesh point). These deformed mesh points are loaded and organized in an octree for fast and efficient look-up.

Next, geometry data is read into the computer. These points are also loaded into the octree for convenient and efficient access during the mapping.

At this point, an empirical variogram is sampled around each leaf node of the octree using a point cloud with ten times as many deformed mesh points than are in the leaf node itself. An empirical variogram is created for each displacement direction making a total of three empirical variograms.

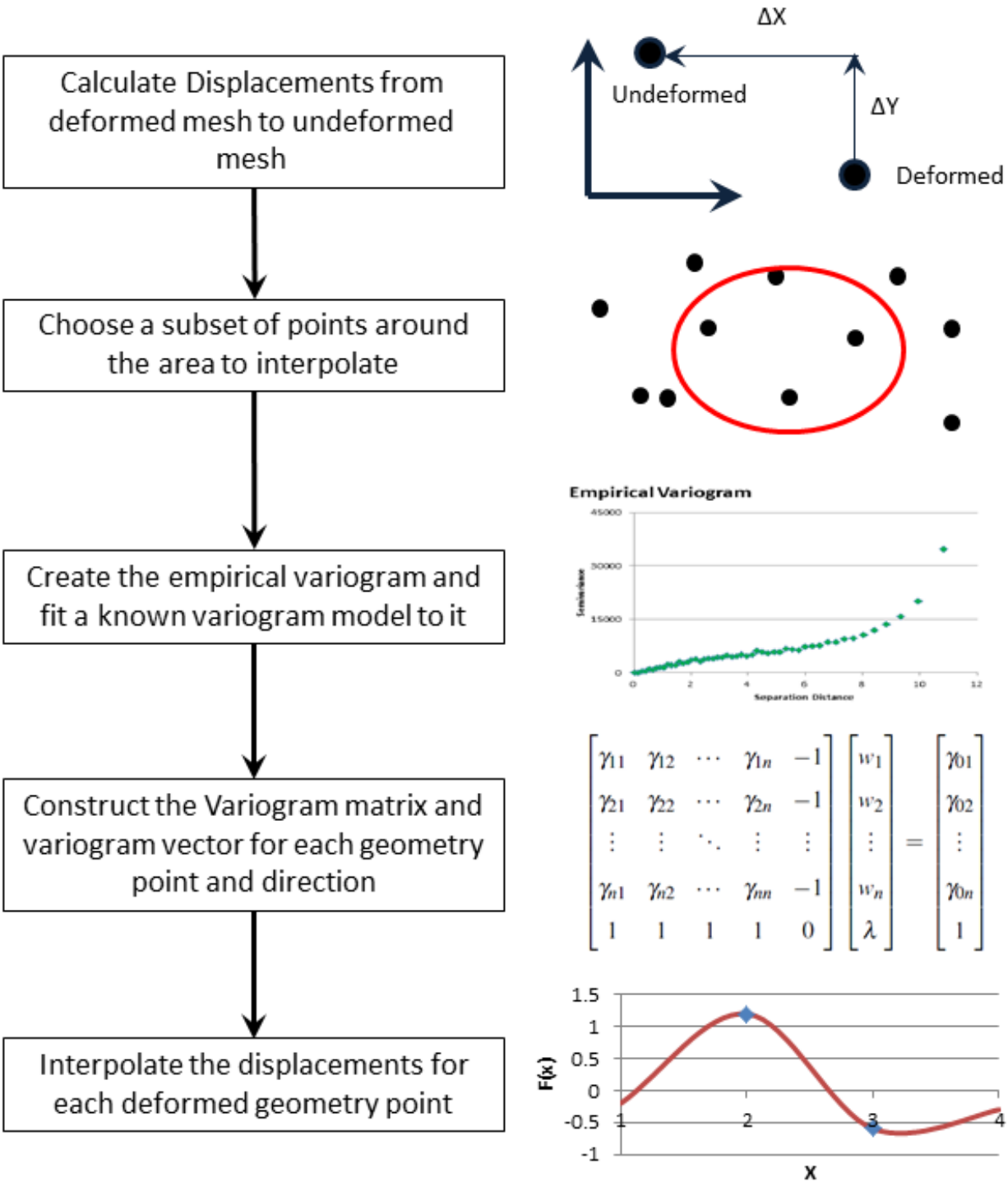

Figure 3.9: A flow chart illustrating the procedure for hot-to-cold mapping using Kriging

Each empirical variogram is then fed into a genetic algorithm to fit the nested variogram model to the empirical variogram. This nested model consists of a nugget model, a linear model, a near-quadratic power model ($ah^{1.99}$), a Gaussian model, and a cardinal sine model summed together.

Once a fitted function has been determined, the Ordinary Kriging system of equations is solved to determine the weights for each geometry point. The weights are then used to interpolate the displacement in each direction and to calculate a standard deviation for the interpolation.

Lastly, the interpolated displacements are added to the original coordinates of the deformed geometry to create the undeformed geometry. These are then written to a file for use by other systems.

CHAPTER 4. RESULTS

4.1 Testing Methodology

The three methods posed were all evaluated and compared by three different metrics: accuracy, speed, and smoothness. Although error estimation is a feature of Kriging (in the form of a standard deviation), this was not comparable to the other methods since they did not posses any similar feature. It is, however, investigated and discussed as a means of estimating error. Also, code complexity was not compared since the implementation of Sculptor™'s ASD is proprietary.

Two models were generated for formulating results. For tuning the Kriging process and quantifying the variogram fitting procedure, a 2 in. long cantilever beam with a 1/4 in. by 1 in. rectangular cross section was created. Another model, this one a generic rotor blade, was generated for testing the method itself and comparing it with the other 2 methods. The rotor blade was modeled after the GOE531 airfoil [23] using 25 different section curves. The curves were scaled to give the blade a taper using the formula $c = 1.2 - 0.1z$, where c represents the chord length of the section. The curves were also rotated to give the blade a slight twist using the formula $\theta = \frac{\pi z^2}{16}$ where θ represents the rotation angle about the centroid of the section. Both models were fixed to the origin i.e. the base section was at $z = 0.0$ and the tip section at $z = 2.0$. The blade was created in Siemens NX using a through curve operation. A 0.0125" fillet was applied along the trailing edge of the model. The models can be seen in Figure 4.1.

Both of these models were solved using ANSYS™ with a linear static model. The models were meshed using 20-node brick elements. The meshes were refined until the total displacement converged. For the cantilever beam model, the converged solution had approximately 5k nodes. For the blade model, 41k nodes were needed. These were each solved for three different load cases: an 80,000 rpm centripetal load, a 10 psi pressure load applied on the suction surface, and the combination of these two. These loads, while arbitrary, may be comparable to a blade in a turbocharger. Meshes and solutions for these models can be seen in appendix A.

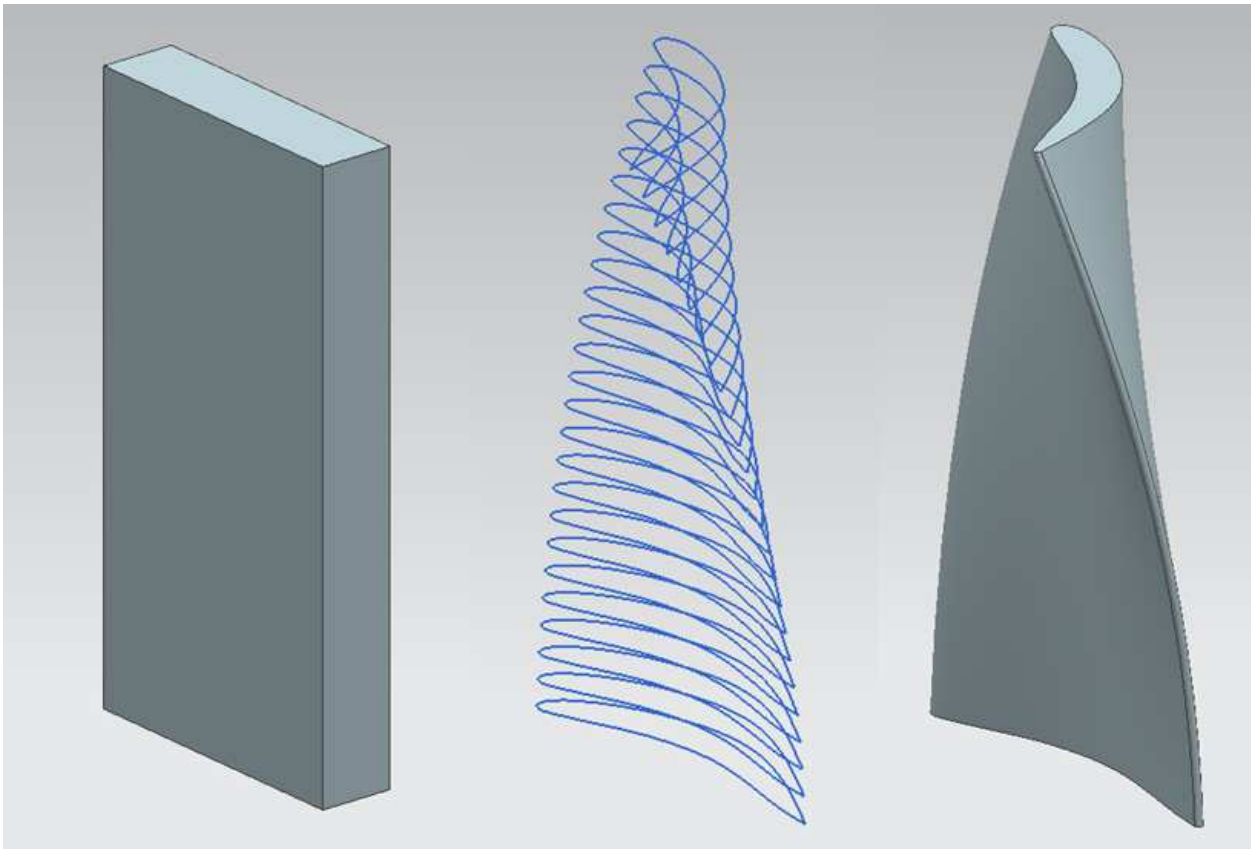

Figure 4.1: Trimetric views of the models used for testing. From left to right: *a*) the cantilever beam, *b*) section curves defining the rotor blade, and *c*) the rotor blade

By nature, accuracy is difficult to quantify in hot-to-cold mapping problems because no solution is known beyond the actual finite element solution contained in the mesh. The actual solution is only known at these discrete locations—knowledge of the actual solution at all locations along the surface, which is what is needed, is not. Therefore, to provide a solution to compare to, a second finite element solution was formulated using a much denser mesh (39k nodes for the cantilever beam and 303k nodes for the blade). Since the FEA solution of the dense mesh was essentially the same as the solution for the original mesh, it can be used as an actual solution for the mapping at this increased number of discrete locations. Therefore, this new dense mesh was used as a pseudo-geometry file. Once the pseudo-geometry had been mapped using Kriging, RBF mapping, or ASD, the result could then be compared to the actual finite element solution, and the error could be calculated as $Error = abs(FEADisplacement - EstimatedDisplacement)$.

The speed of the methods was computed as the actual run-time of the program. For the RBF mapping and Kriging programs, multi-threaded operation was employed for the major features of the operation, namely the solution to the linear systems for weight calculation and vector operations involved in the interpolation. While Sculptor™'s implementation of ASD is proprietary, it is a reasonable guess that it is only single-threaded judging by the computers processor usage while running the program. This research, however, is not aimed at optimizing other methods of hot-to-cold mapping. Therefore, ASD will be compared to the other methods regardless of the number of processor cores that Sculptor™uses in its implementation.

The smoothness of the mappings will be more of a qualitative comparison. To compare, the actual geometry file (not to be confused with the pseudo-geometry file) will be mapped. Next, the mapped blade geometry will be reconstructed in Siemens NX using a through curve operation on all the section curves with the 0.0125 in. trailing edge fillet applied afterward. Smoothness will then be evaluated by inspection of the model, checking for any wrinkles or divots in the surface.

4.2 Kriging Results

For the Kriging process, this research first tested the methodology on the cantilever beam model whereas the other methods were only run on the rotor blade model. This extra comparison was done because 1) well-known solutions exist for cantilever beams, 2) calculations and tests could be done much quicker on the lower node count model, and 3) variogram models were likely to be similar for the cantilever beam and rotor blade, making fine-tuning the fitting process ideal with the beam model.

4.2.1 Cantilever Beam

The empirical variogram was first sampled at several locations along the beam for all three load cases and for all three displacement directions. Several variogram models were common at all locations. In the directions of large displacements, e.g. the direction of the pressure load, a power model shape would generally be present. While at times a single power variogram model with varying coefficient and exponent was able to produce a good fit, a nested model with two separate power variograms, each with pre-determined exponents (1.0 and 1.99), tended to obtain a better

fit as seen in 4.2. While the fit shown is not perfect, it is sufficient for the purposes of Kriging, considering that the process has been found to work well despite sub-optimal variograms as was discussed in section 3.1.4.

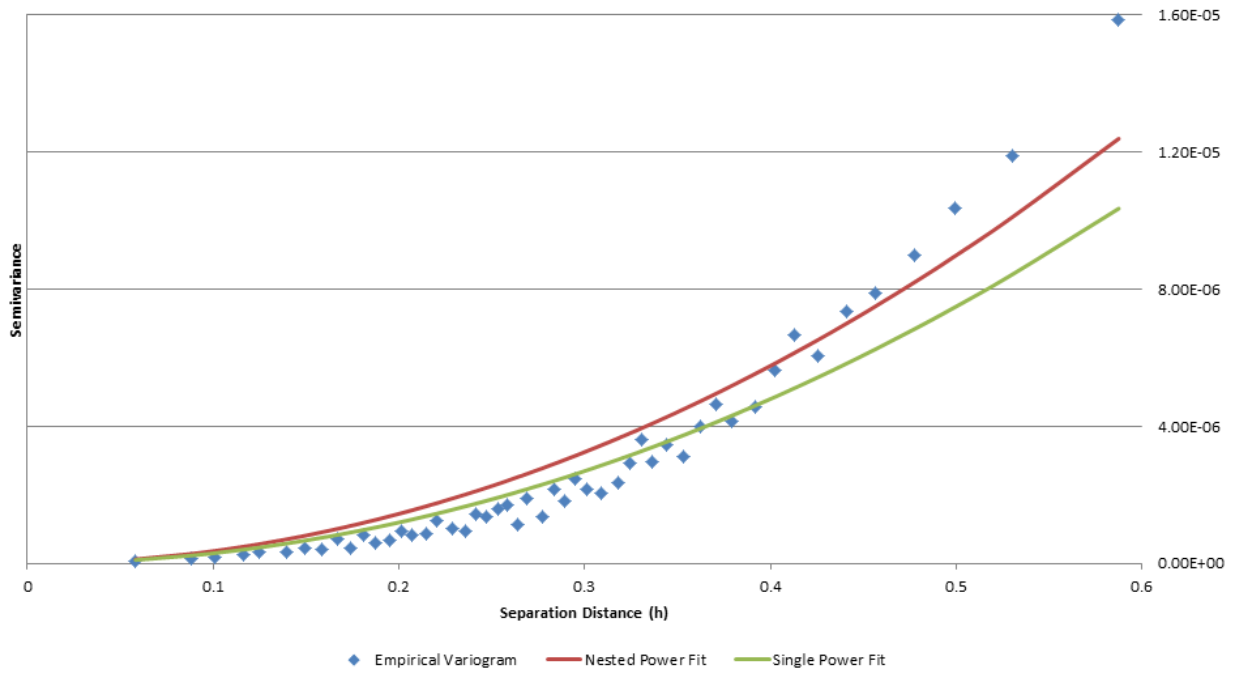

Figure 4.2: An example of an empirical variogram with two attempted fits: a single power law model and a nested power law model

In the direction of small displacements, a Gaussian shape was the most prevalent and provided good fits as in Figure 4.3 It could be argued that a spherical or exponential model might provide a better fit in some cases. However, in order to keep the dimensionality of the fitting optimization small, only the Gaussian model was chosen. This could be a point of future research, especially considering the near-origin behavior of the variogram is the most important [15].

A nested variogram model was created using the three models discussed. The genetic algorithm would then alter the coefficients and the range parameters to find the best fit to the empirical variograms. If only a Gaussian model was necessary then the algorithm would simply optimize to the nested model with zeros as the coefficients for the power models. Similarly, if only a power model was necessary then the Gaussian model's coefficient would be optimized to zero. This proved to be a robust method for determining an ideal variogram model for use in Kriging.

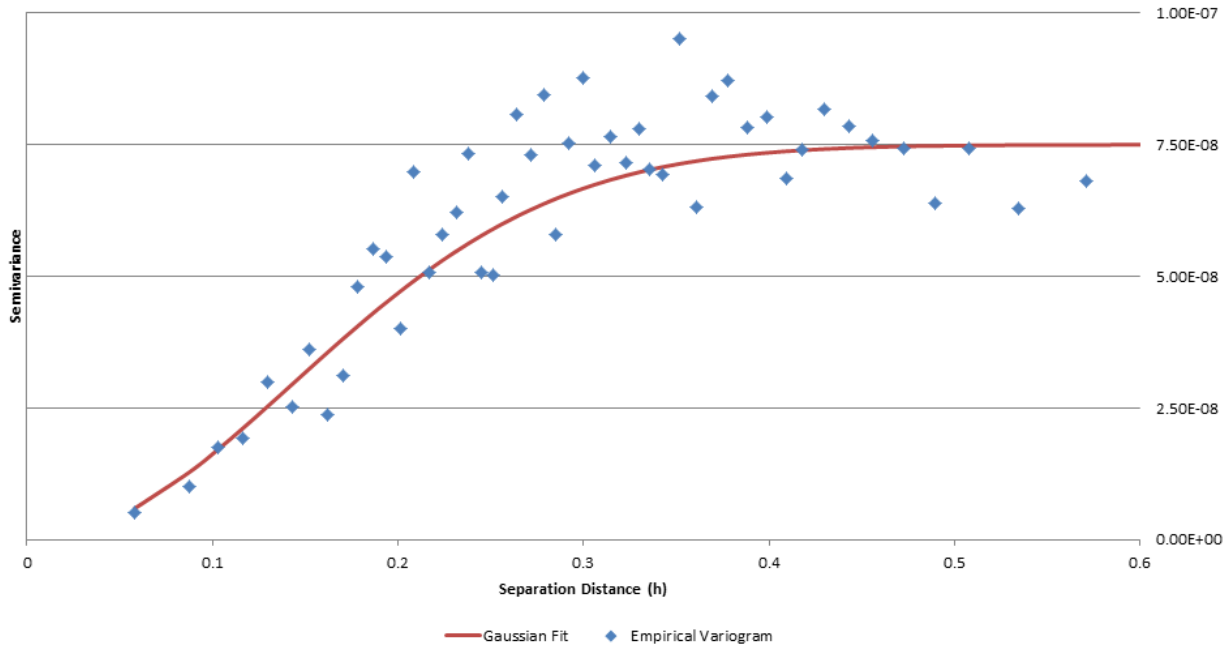

Figure 4.3: An example of an empirical variogram with a Gaussian model fitted to it

Kriging with this model and variogram fitting procedure produced very good results. The mapping results for all three load cases were similar, which would indicate that the process is indeed independent of the load case. Because all results were similar only the combined pressure and centripetal load will be discussed in terms of the evaluation criteria.

For the combined load case, the maximum displacement was 0.0224 in. The mapping for this case was performed in under a minute while using 500 mesh points for each interpolation. Smoothness for this model was not considered since the model itself is not smooth. The maximum error, which occurred just beyond the fixed end, was 2.2×10^{-4} in.. Once beyond the fixed end of the beam, the error dropped even further to an average value of 1.2×10^{-5} in.. Both of these numbers were well within the defined tolerance for this method. A chart of these errors can be seen in Figure 4.4.

A result of interest was observed concerning the Kriging standard deviation. While roughly 95.3% of the the mapped points had errors below three standard deviations (slightly lower than the expected 99.5%), almost all of the 4.7% that had larger errors also had very small standard deviation estimates. In fact, for the most part the error was between 0.0 in. and 3.5×10^{-5} in. regardless of standard deviation. This seems to suggest that while useful information, the standard

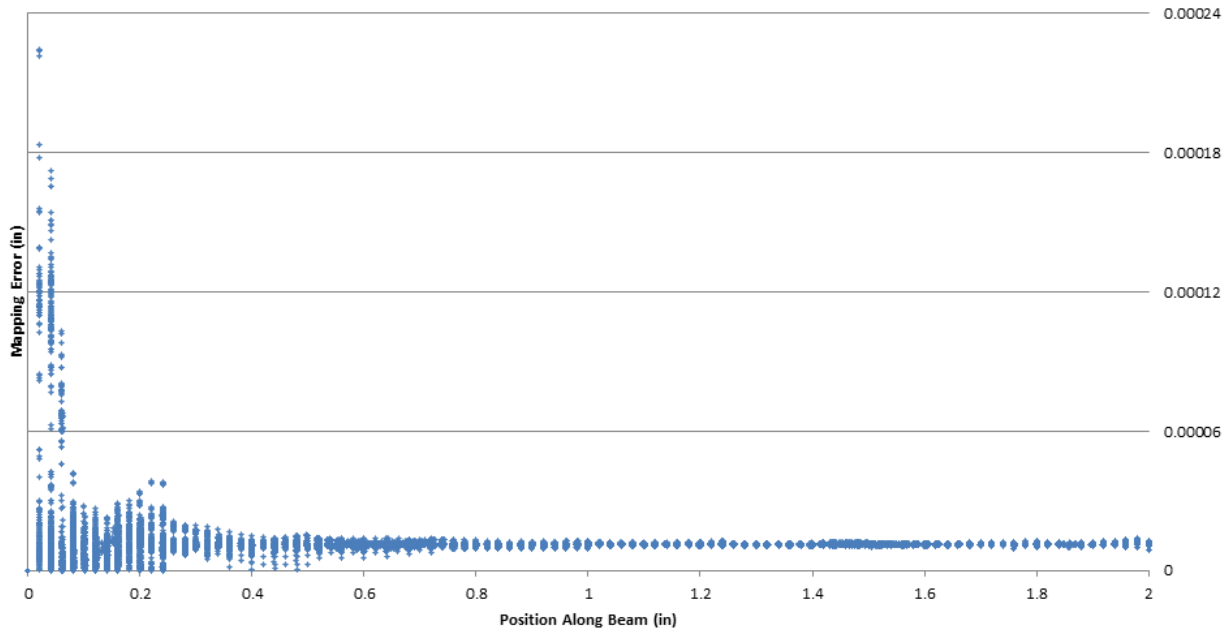

Figure 4.4: A plot of the z-position vs mapping error for the cantilever beam model

deviation may be no more than an order of magnitude predictor of error. This can be seen in Figure 4.5

4.2.2 Rotor Blade

The empirical variogram was sampled again on the rotor blade model at several locations for each load case. The variograms, for the most part, appeared similar to the variograms observed with the cantilever beam, as expected. However, two key deviations were seen in several empirical variograms. One difference was noted that some models reached a sill level, as in the Gaussian model, then continued to rise afterward like the power model. This type of empirical variogram could still be fitted from the nested model used on the cantilever beam. An example can be seen in Figure 4.6

However, on some models, a bit of cyclic behavior was observed. This is seen where the variance normally is monotonically increasing with separation distance, instead begins to decrease slightly before increasing again in an periodic manner. This type of variogram was best fitted using a cardinal sine model as seen in Figure 4.7. This model was simply added to the previous nested model and fit as before with good results.

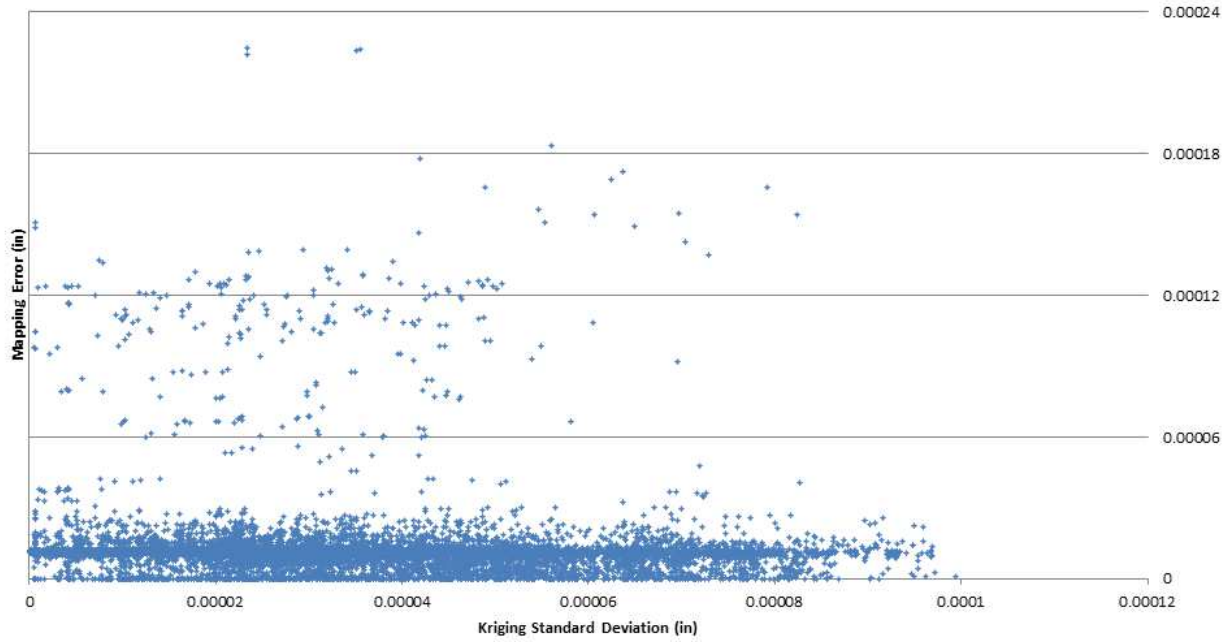

Figure 4.5: A plot of the Kriging standard deviation vs mapping error for the cantilever beam model

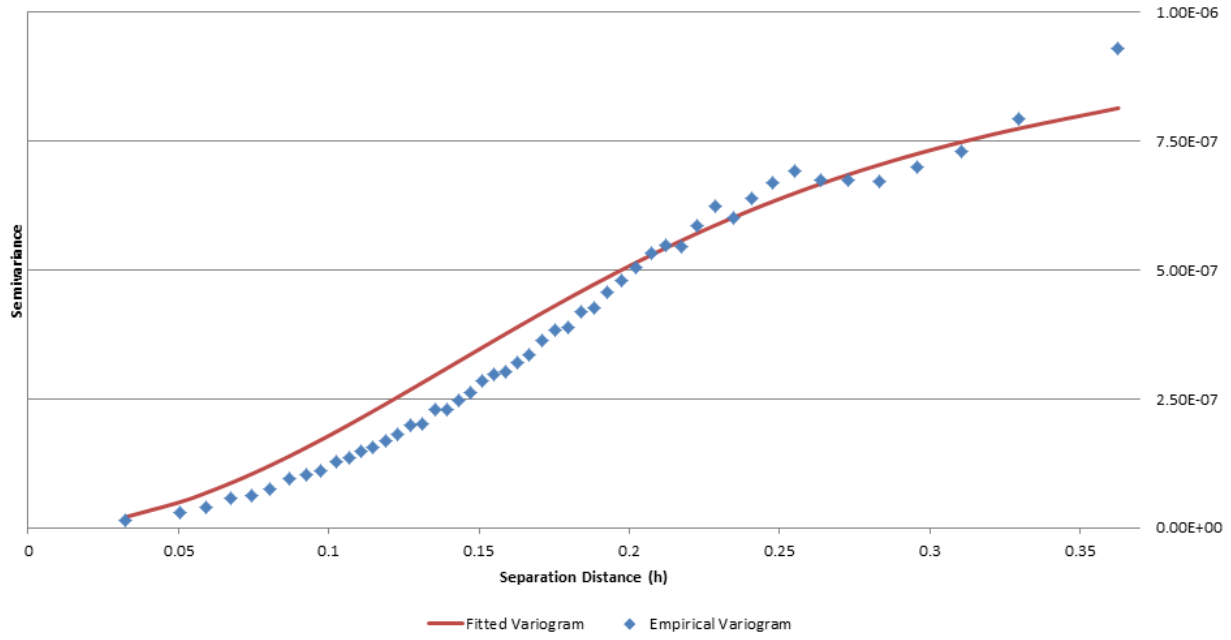

Figure 4.6: An example of an empirical variogram with a combined gaussian and power model fitted to it

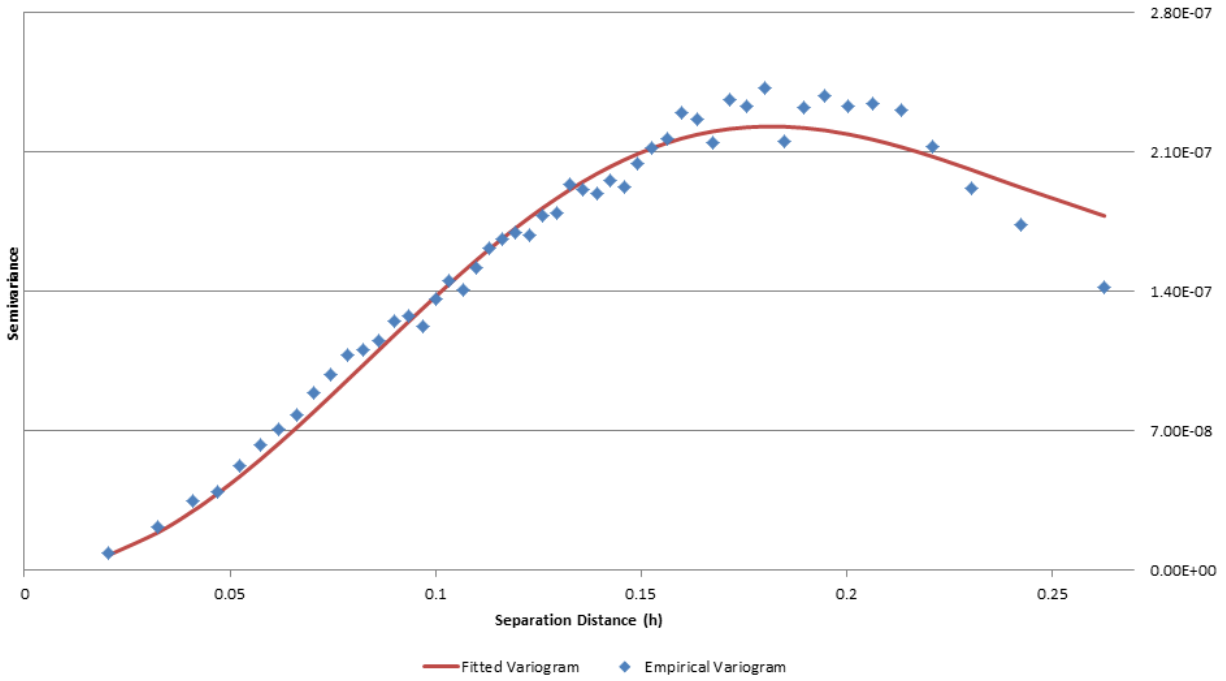

Figure 4.7: An example of an empirical variogram with a cardinal sine model fitted to it

As before, results were similar for all load cases reaffirming that the process was independent of load case. Only the combined load case is discussed here because of the results similarity. For this model and load case, the maximum displacement was 0.162 in..

The mapping speed was once again good for even this larger model, taking only 14 minutes when using 500 mesh points with each interpolation. When using only 250 mesh points, the time was almost cut in half at 8 minutes. Further reduction in points used for each interpolation to 150 and 75 took only 4 minutes. This suggests that while the matrix decomposition in Kriging dominates for large numbers of mesh points used (an $O(n^3)$ process), for smaller values other operations dominate the execution speed. The most likely part to dominate at these levels is the genetic algorithm because its computational cost does not vary with the number of mesh points used.

The displaced model exhibited an excellent level of smoothness as seen in Figure 4.8. This figure shows the blade from only one angle and zoom level, though inspection at all angles was performed on the blade. As the number of mesh points used in interpolating was reduced, the smoothness of the model suffered. At 250 mesh points, the model began to develop a wrinkle

along the trailing edge near the free end. At 75 mesh points, this wrinkle was more pronounced as can be seen in Figure 4.9. The area that the wrinkle appeared is of interest. This location is likely the most difficult to map. Not only does it have a high level of curvature, low mesh density, and thin features, but it also is a point of extrapolation instead of interpolation. Because the trailing edge of the geometry has a small fillet placed on it, the tip point of each section curve lies outside of the mesh.

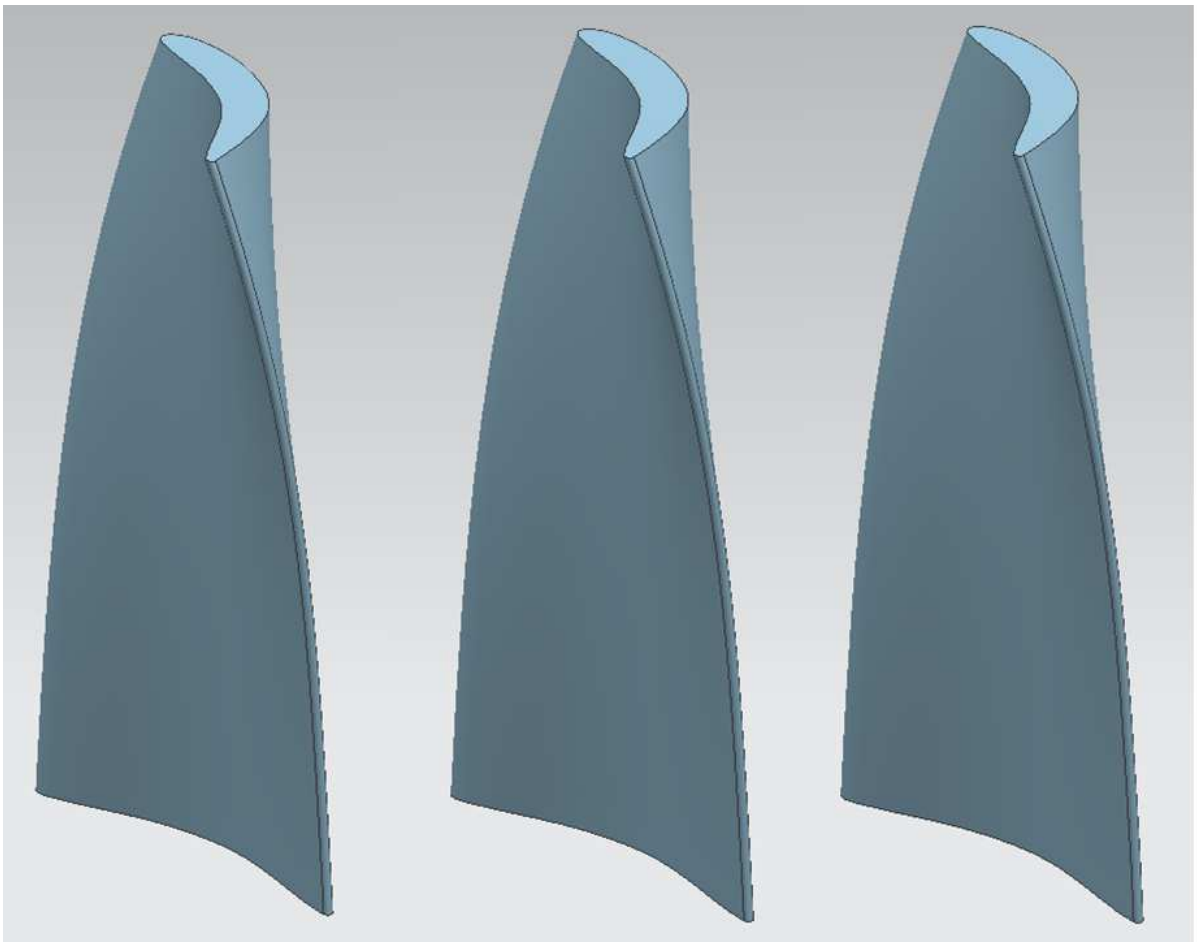

Figure 4.8: Rotor blade geometry mapped using *a*) Kriging, *b*) RBF mapping, and *c*) ASD

The blade's error differed greatly from that of the beam model. Instead of the largest errors being observed at the base, on this model the largest errors were observed from the midpoint of the beam until the tip. The maximum error was at a value of 9.84×10^{-4} in. at approximately 7/8 of the length of the blade as seen in Figure 4.10. The larger error may possibly be attributed

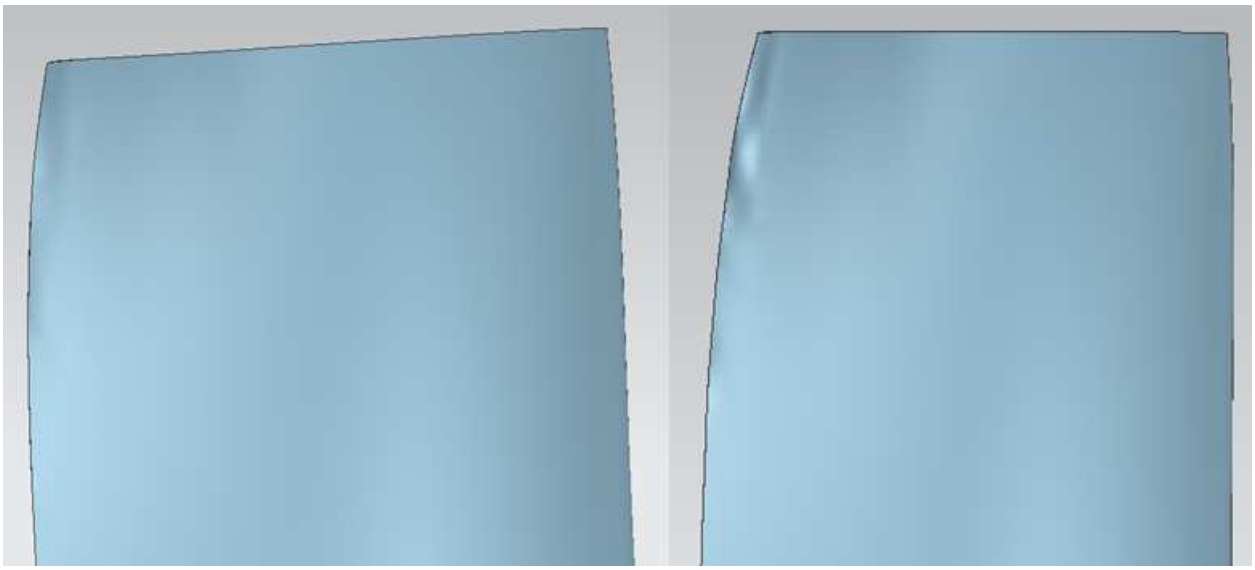

Figure 4.9: Wrinkles along the trailing edge of the mapped rotor blade with 250 and 75 mesh points used in each interpolation. The wrinkles grow more pronounced as the number of mesh points is reduced

to anisotropic effects arising from the increasing rotation angle though this was not investigated as it is beyond the scope of this research. Even though a larger maximum error was observed, the average error was still very small at 1.2×10^{-5} in.. Both of these values were still within the tolerance specified, though the maximum observed error was close to violating this constraint.

The standard deviation estimate with the blade model behaved just as it had with the cantilever beam. The only notable deviation was that 99.5% of all points had errors below three standard deviations. This lines up exactly with the value that would be expected from an actual standard deviation from observed data. This can be seen in Figure 4.11.

4.3 RBF Mapping Results

Like with Kriging, 500 mesh points were used for each interpolation with the RBF method. This process gave excellent speed results, finishing in only 7 minutes. A good, smooth shape was once again achieved with this method. The mapped blade can be seen in Figure 4.8. As with Kriging (see Figure 4.9), the shape began to develop wrinkles when low numbers of mesh points were used in each interpolation.

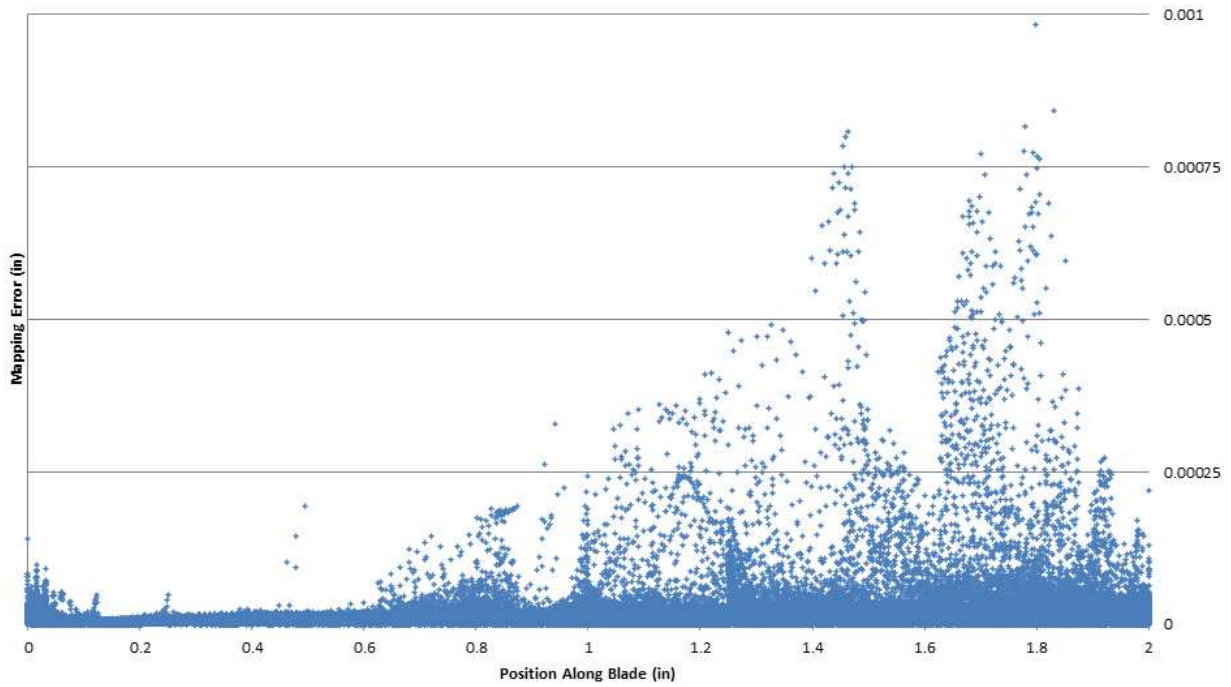

Figure 4.10: A plot of the z-position vs mapping error for the rotor blade model when mapped using Kriging

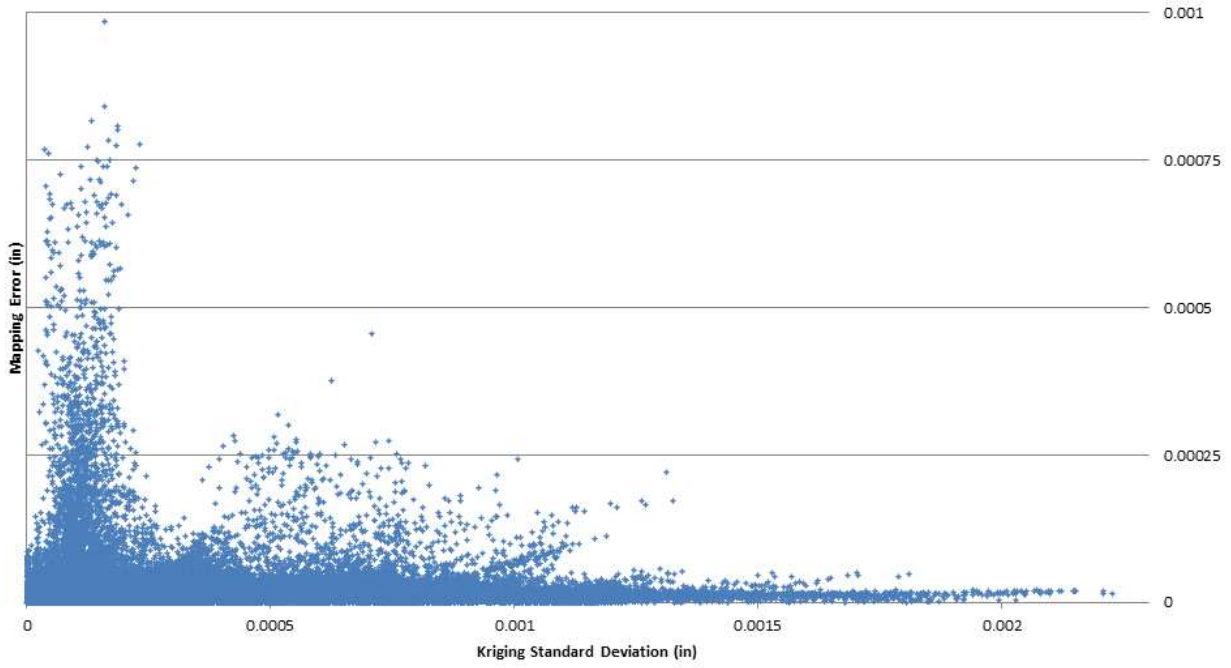

Figure 4.11: A plot of the Kriging standard deviation vs mapping error for the rotor blade model

RBF mapping had excellent accuracy in performing the mapping. The maximum error in this case was only 1.2×10^{-4} in., almost a full order of magnitude better than the Kriging result. The average error, however, was similar to the Kriging result at 1.1×10^{-5} in.. This can be seen in Figure 4.12. The largest pocket of error is seen near the fixed end of the blade as well as another, smaller pocket near the tip. This makes intuitive sense because less information is known for the interpolation at the fixed and free ends (the sphere of mesh points for interpolating is truncated at these areas).

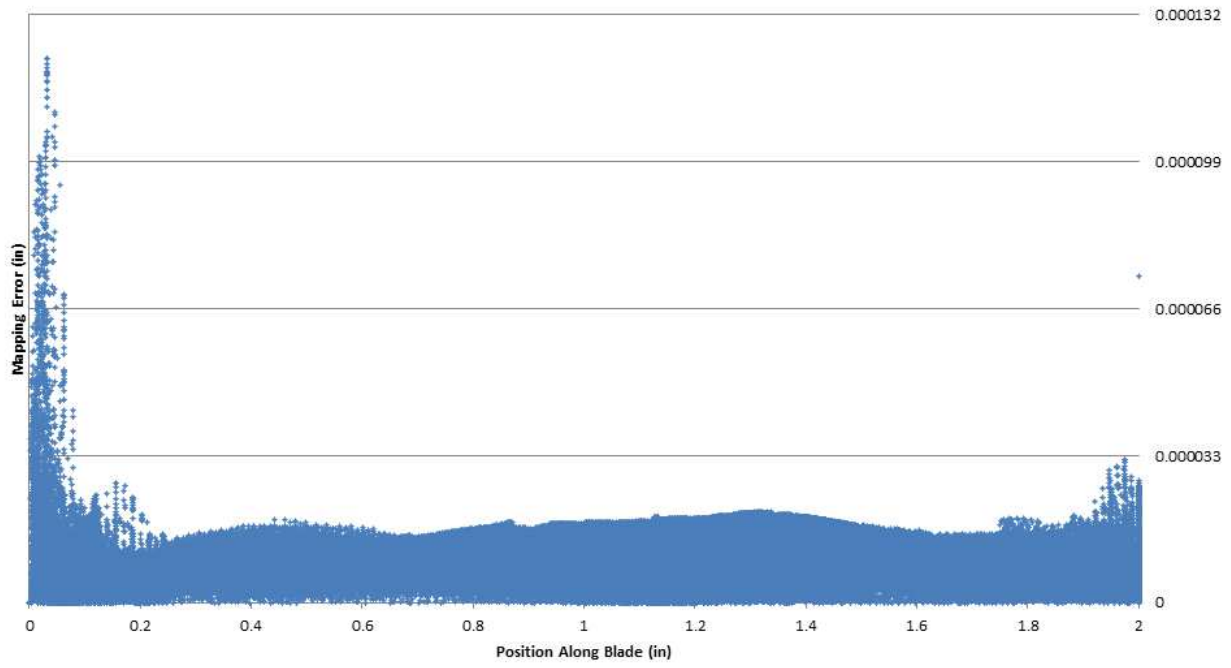

Figure 4.12: A plot of the z-position vs mapping error for the rotor blade model when mapped using RBF mapping

4.4 Sculptor™Shape Matching Results

Shape matching was performed using Sculptor™'s ASD method. Using a 4x4x7 control grid (with 7 being along the z-axis), Sculptor™was able to converge to a solution in 30 minutes, by far the slowest of the methods. This can likely be attributed to the cost of computing gradients of the objective function, in this case the sum of the differences between hot and cold nodes,

using the forward difference method. Each node in the control grid introduces three new variables thus adding three new derivatives to calculate. With 288 derivatives to calculate by evaluating the difference at 41k nodes at each iteration, it is easy to see why this process is much slower.

Smoothness, as with the other methods, was very good as seen in Figure 4.8. This logically follows since the basis functions for the mapping are inherently smooth. The error, however, was not as good as the other methods shown here in Figure 4.13. The maximum error was a little better than with Kriging at 7.6×10^{-4} in., though the average error was almost 4 times worse than the other methods at 4.7×10^{-5} in.. Figure 4.13 helps to explain why the error was larger in many places. Bumps can be seen in the chart which lie in-between planes of control points. Spikes are also seen at the edges where fewer control points have influence on the geometry points to map in those regions. Local refinement of the control grid could be done to remove these spikes. The average error could also be reduced by refining the entire grid and theoretically drive the error to almost zero. This, however, would require more time for each iteration, going beyond the 30 minute constraint for this research.

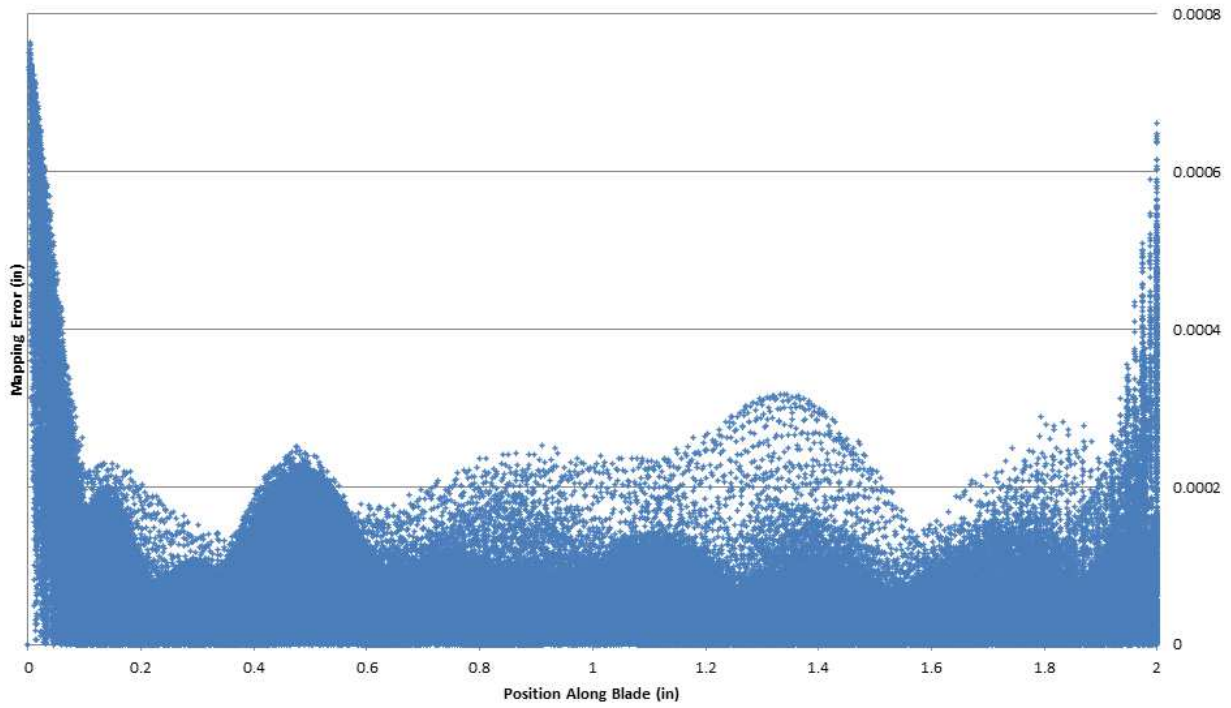

Figure 4.13: A plot of the z-position vs mapping error for the rotor blade model when mapped using ASD

4.5 Comparisons

As discussed above, each method fulfilled the constraints of the research for accuracy, smoothness, and time. Kriging was the only one to provide an error estimation, however, as discussed, this was only useful as an order of magnitude approximation to the error. Though Kriging was similar in construction to RBF mapping, it required more time to execute because of it could not reuse the decomposition of the variogram matrices for each displacement direction ($O(n^3)$ for each matrix decomposition), nor could it reuse the weights for each point ($O(n^2)$ for each linear solve).

RBF mapping performed the best in all the categories. Its ability to reuse matrix decompositions for each direction as well as reusing weights for each point gave it superior execution speed.

ASD performed well though was very time consuming. It did have one advantage over all the other methods in that it could guarantee arbitrary levels of accuracy if given enough time to solve.

CHAPTER 5. CONCLUSION

5.1 Summary

The purpose of this research was to develop a hot-to-cold mapping process that could 1) map the geometry using only generic inputs, 2) map points within tolerance, 3) map the points in a smooth manner, 4) perform the mapping in under 30 minutes, and 5) provide an estimation of the error of the mapping.

Kriging was implemented as a method to accomplish these goals. A genetic algorithm was implemented to fit the variogram for use in Kriging. An octree data structure was also implemented to allow for fast look-up of spatial data. Kriging was used to interpolate the displacements of points for a hot-to-cold mapping on both a cantilever beam and a generic rotor blade model. RBF mapping was implemented from existing research. RBF mapping and Arbitrary Shape Deformation (ASD) were used on the same generic rotor blade model and compared to the Kriging method.

5.2 Findings

Kriging was found to be a viable method for hot-to-cold geometry mapping as defined by this research's criteria. The method used only the generic inputs of a hot (deformed) mesh, cold (undeformed) mesh, and hot (deformed) geometry file. The other two methods examined, RBF mapping and ASD, also required only these inputs. The other criteria were also met as described in the following sections.

5.2.1 Mapping Within Tolerance

Kriging was able to map all the points within the specified tolerance (on the order of 0.001–0.0001 in.). It's maximum error was far from ideal, coming just within tolerance and at the worst level of all three processes. However, it's average error was excellent and comparable to the best

method of the three (RBF mapping). Error was also found to increase with a decrease in the number of mesh nodes used in each interpolation. While ASD had a slightly lower maximum error, its error on average was much larger than that of Kriging. RBF mapping far outperformed either method in regards to accuracy.

5.2.2 Mapping Smoothness

Kriging was capable of creating a smooth geometry output. It was found that large numbers of mesh nodes used with each interpolation helped to smooth out the shape. As smaller numbers of mesh nodes were used, wrinkles began to develop along the trailing edge. Further reduction resulted in more pronounced wrinkles at this location. RBF mapping also exhibited this same behavior with small numbers of mesh nodes used. ASD was inherently smooth due to the basis functions used for its mapping.

5.2.3 Mapping Time

Kriging performed the mapping within the allotted 30 minutes, actually finishing in only 14 minutes. Examination of the process showed that its execution time was dominated by three matrix decompositions, leading to cubic growth of the execution time with an increasing amount of mesh nodes used in each interpolation. Therefore, a fine balance needs to be reached between smoothness, accuracy, and execution time when determining the number of mesh nodes to use in each interpolation. RBF mapping was twice as fast and dominated by only one matrix decomposition. ASD took the full 30 minutes to perform the mapping. This process was dominated by the gradient calculations used for the optimization. These gradient calculations grow with the size of the mesh as well as the size of the control grid.

5.2.4 Mapping Error Estimation

While Kriging provided a standard deviation as an estimate of error, this feature proved to be less useful than hoped. The standard deviation estimate behaved globally as would be expected with 99.5% of error values below 3 standard deviations. However, locally (on a point-by-point basis) the standard deviation behaved opposite of what was expected. Large values of the standard

deviation corresponded with small errors, and small standard deviations corresponded with large errors. It was concluded that, in its current form, the standard deviation was only useful as an order of magnitude estimate of error. The other methods did not provide any estimation of the error.

5.3 Recommendations for Future Research

While Kriging was found to be a viable option for hot-to-cold geometry mapping, it was matched or outperformed by RBF mapping in every category of comparison. It also failed to improve on ASD in many categories. This process, however, has many areas that it can be improved upon.

This research found that an isotropic approximation to the variogram gave sufficient results for performing the mapping. However, the research also found that the variogram, in reality, had a large degree of anisotropy. This research also found that the anisotropy for the given problem required a non-trivial solution. By researching into methods for approximating these anisotropies, the accuracy of the Kriging process could be increased dramatically. This would not only give a more accurate solution, but would also allow for few mesh points to be used with each interpolation, thus improving execution speed.

A genetic algorithm was used for the fitting of variogram functions to the empirical variogram. This method was found to be very robust and simple to implement, however it requires much more computation time than more traditional methods of fitting. While not the dominate factor in execution speed, it was a large contributor to the overall execution time. Research into different methods for automatically fitting functions to the empirical variogram at a lower cost while maintaining robustness would improve this methods execution time.

Lastly, this research implemented Ordinary Kriging which assumes that while the mean is not known, no trend or drift is associated with it. This assumption was almost certainly violated, but power variogram elements were used to approximate this with good results. More complex models of Kriging, such as Universal Kriging, would be better able to model this behavior. Universal kriging includes a trend model into the interpolation, much like the linear polynomial that was used in RBF mapping which may account for the better results found with RBF mapping. Research into the best implementation of Kriging for hot-to-cold geometry mapping would be able to determine a more accurate model for use.

REFERENCES

[1] Mahajan, A. J., and Stefko, G. L., 1993. "An iterative multidisciplinary analysis for rotor blade shape determination." In *AIAA/SAE/ASME/ASEE 29th Joint Propulsion Conference and Exhibit*, Vol. 29. 1, 3

[2] Denton, J. D., 2010. "Some limitations of turbomachinery cfd." In *Proceedings of the ASME Turbo Expo 2010: Power for Land, Sea and Air*, Vol. 7, pp. 735–745. 1

[3] Abumeri, G. H., and Chamis, C. C., 1996. "T/best: a computer code for assessing the benefits of advanced aerospace technologies." *Advances in Engineering Software,* **28**, pp. 231–238. 1, 4

[4] Brown, K. W., 1992. Structural tailoring of advanced turboprops (stat) Tech. Rep. 191017, Pratt Whitney, East Hartford, Conneticut. 1, 4

[5] Paldi, F., 2007. "Conversion of 'hot' geometry into 'cold' geometry using the finite element method program ansys®." Master of science, Engineering and Computational Engineering, Technical University of Berlin. 2

[6] Persson, A., and Runsten, J., 2011. "Cfd investigation of a 3.5 stage transonic axial compressor including real geometry effects." Master of science, Department of Applied Mechanics, Chalmers University of Technology. 5, 7, 8, 9, 10

[7] Landon, M., 2012. Hot to cold tubine blade shape matching `http://gosculptor.com/HottoColdTurbineBlade.html`. 5

[8] Schaback, R., 2007. A practical guide to radial basis functions `http://num.math.uni-goettingen.de/schaback/teaching/sc.pdf` Chapter of a future book. 9, 10

[9] Sederberg, T. W., and Parry, S. R., 1986. "Free-form deformation of solid geometric models." In *SIGGRAPH '86 Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, Vol. 20, pp. 151–160. 12

[10] Duvigneau, R., 2006. Adaptive parameterization using free-form deformation for aerodynamic shape optimization Tech. Rep. 5949, Institut National De Recherche En Informatique Et En Automatique. 12

[11] Andreoli, M., Ales, J., and Desideri, J.-A., 2003. Free-form-deformation parameterization for multilevel 3d shape optimization in aerodynamics Tech. Rep. 5019, Institut National De Recherche En Informatique Et En Automatique. 12

[12] Sederberg, T. W., 2014. Computer aided geometric design `http://cagd.cs.byu.edu/~557/text/cagd.pdf`, October. 13, 14

[13] Landon, M., 2014. Chief technology officer, January Private communication. 14

[14] Clark, I., 2015. What is kriging http://www.kriging.com/whatiskriging.html, March. 16

[15] Chiles, J.-P., and Delfiner, P., 2012. *Geostatistics: Modeling Spatial Uncertainty*., 2 ed. John Wiley & Sons. 16, 19, 20, 23, 24, 26, 30, 40

[16] Press, O. U., 2015. Oxford online dictionary http://www.oxforddictionaries.com/us/definition/american_english/autocorrelation, March. 16

[17] Bohlin, G., 2005. Variogram analysis http://people.ku.edu/~gbohling/cpe940/Variograms.pdf. 19, 23

[18] Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P., 2007. *Numerical Recipes*., 3 ed. Cambridge University Press. 20

[19] Brunell, R. M., 1992. An automatic procedure for fitting variograms by cressie's approximate weighted least squares criterion Tech. rep., Southern Methodist University, October. 23

[20] Long, A., 1999. Geostatistical mapping techniques http://ceadserv1.nku.edu/longa//modules/geostats/lec/latex2html/latex2html.html. 23

[21] Isaaks, E. H., and Srivastava, R. M., 1990. *An Introduction to Applied Geostatistics*. Oxford University Press. 23

[22] Balling, R. J., 2012. *Computer Structural Optimization*., Vol. 2 BYU Academic Publishing. 32

[23] airfoiltools.com, 2015. Goe 531 airfoil http://airfoiltools.com/airfoil/details?airfoil=goe531-il. 37

APPENDIX A. ANSYS SOLUTIONS FOR TEST MODELS

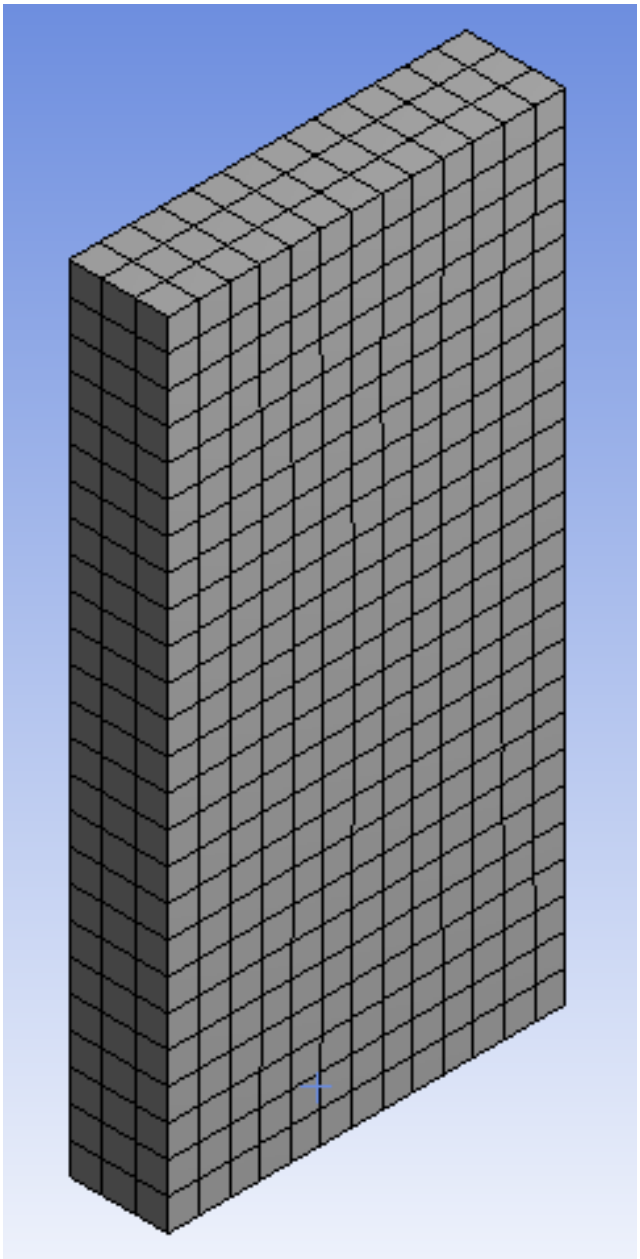

Figure A.1: Mesh used for solutions with the cantilever beam model. Elements are 20-node bricks

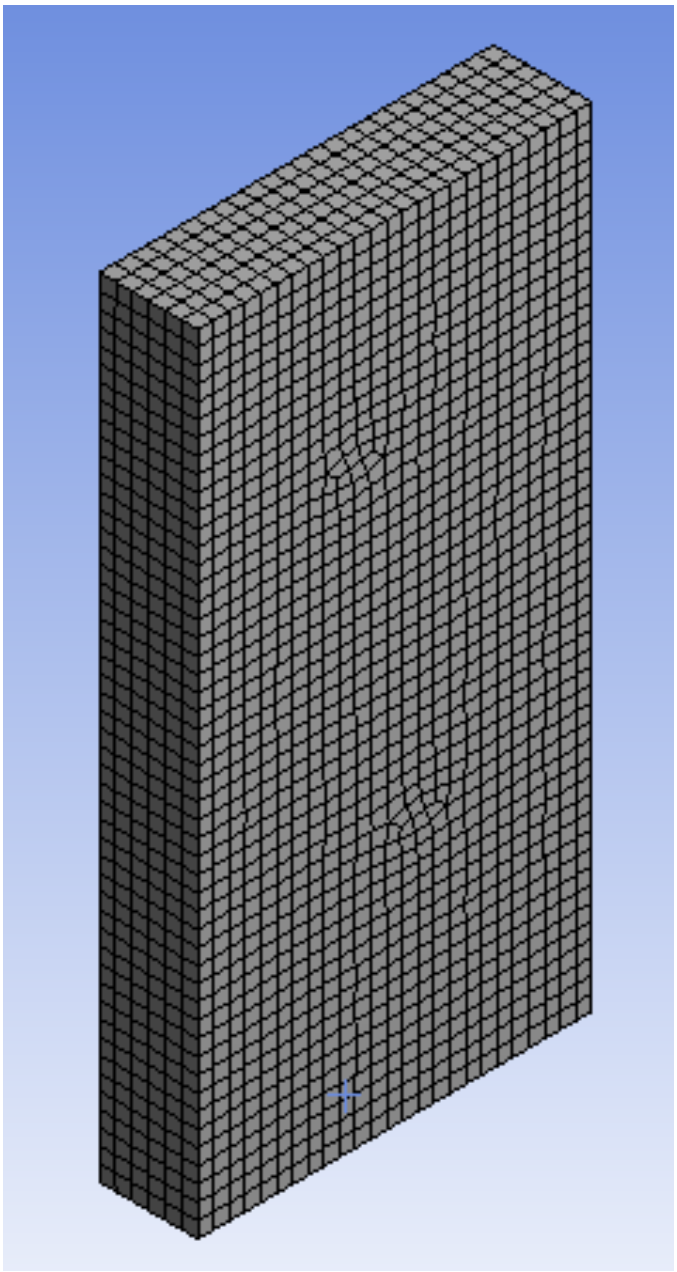

Figure A.2: Fine mesh used as psuedo-geometry for the cantilever beam model. Elements are 20-node bricks

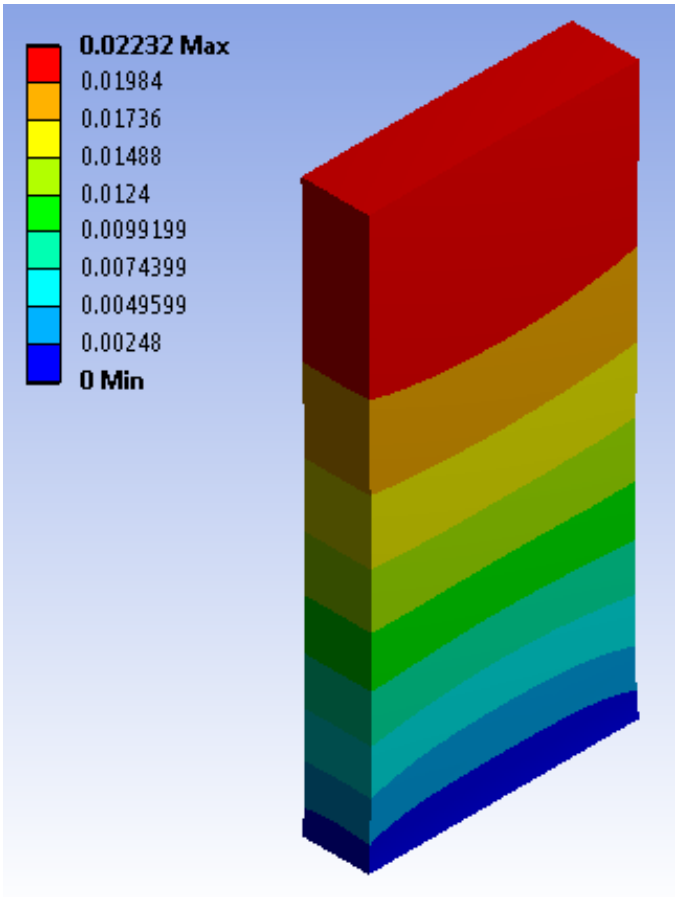

Figure A.3: Total displacement solution for the centripetal load case

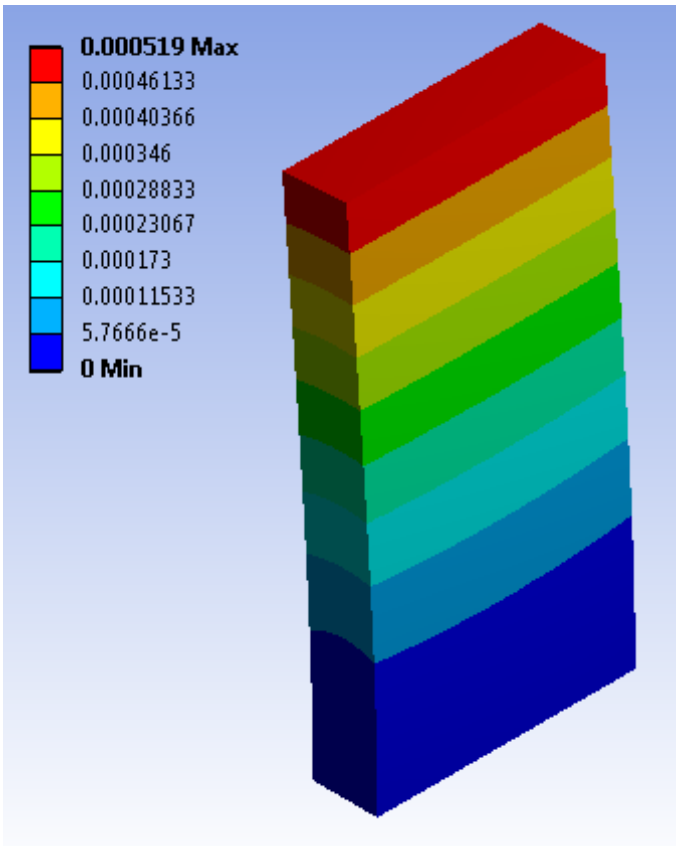

Figure A.4: Total displacement solution for the uniform pressure load case

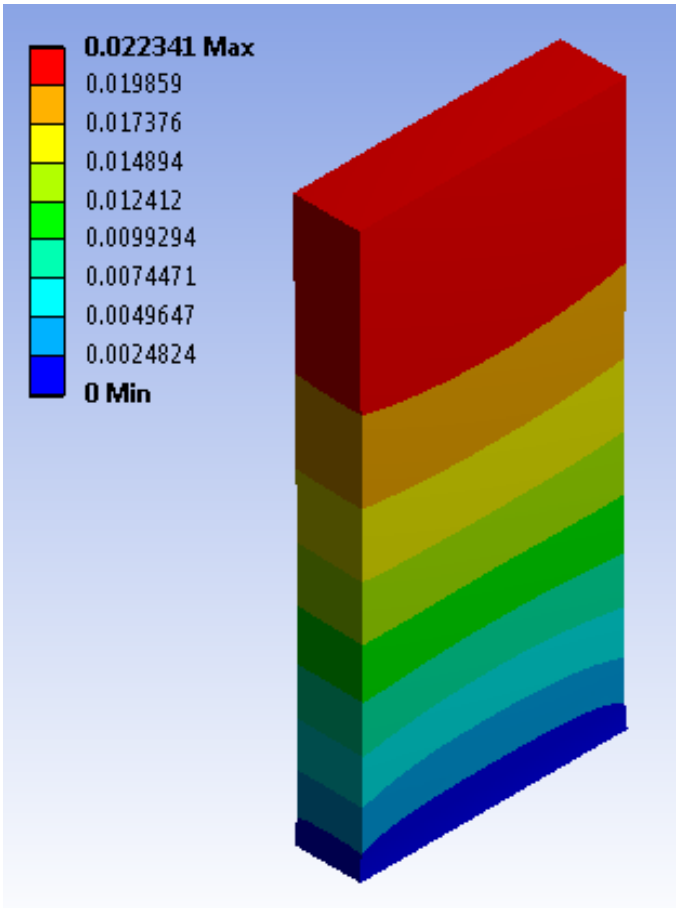

Figure A.5: Total displacement solution for the combined load case

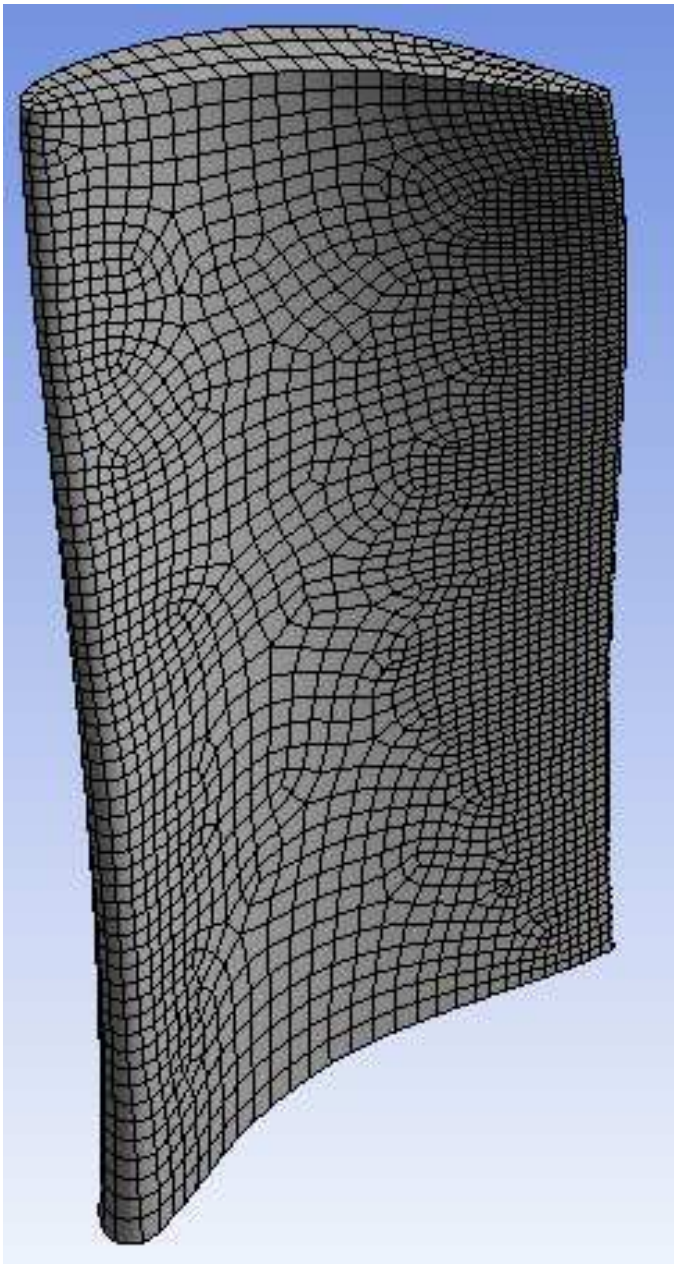

Figure A.6: Mesh used for solutions with the rotor blade model. Elements are 20-node bricks

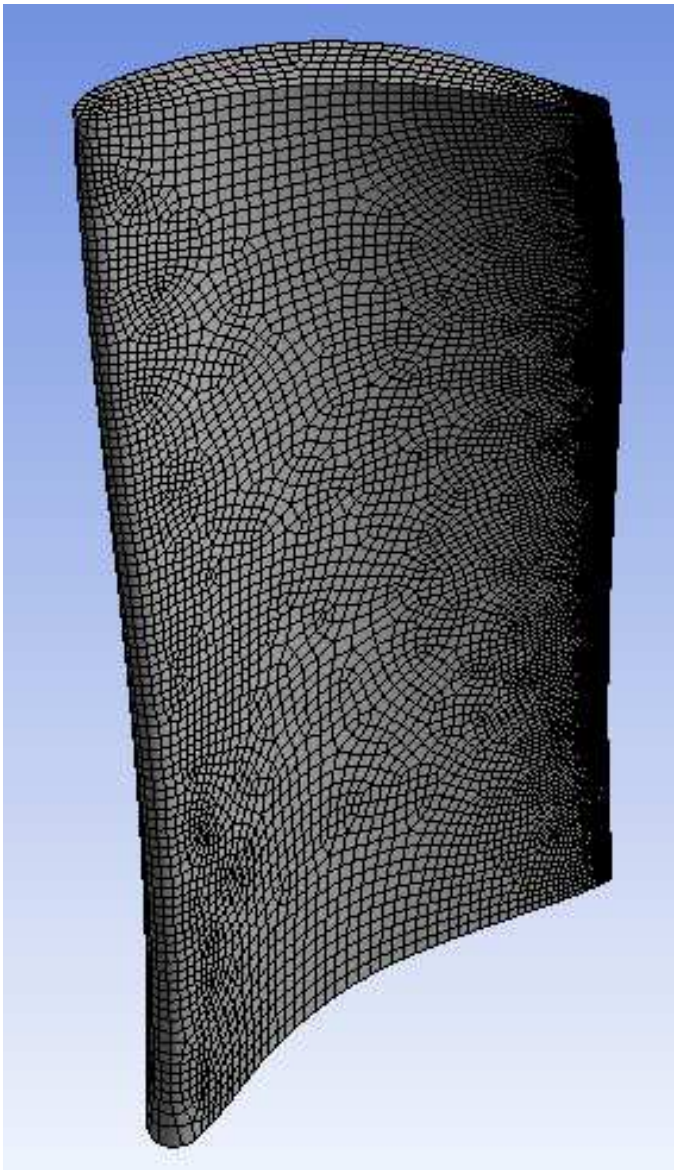

Figure A.7: Fine mesh used as psuedo-geometry for the rotor blade model. Elements are 20-node bricks

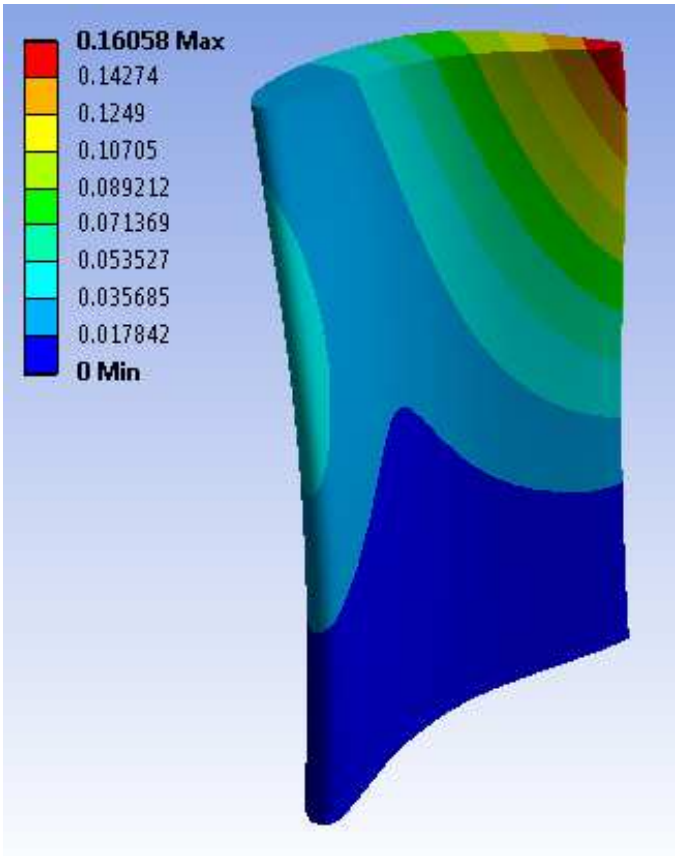

Figure A.8: Total displacement solution for the centripetal load case

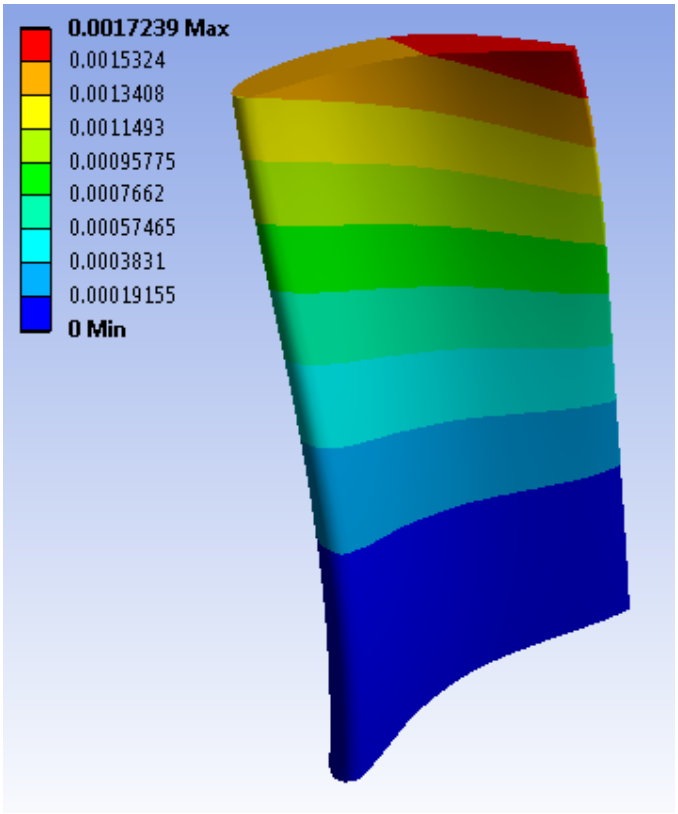

Figure A.9: Total displacement solution for the uniform pressure load case

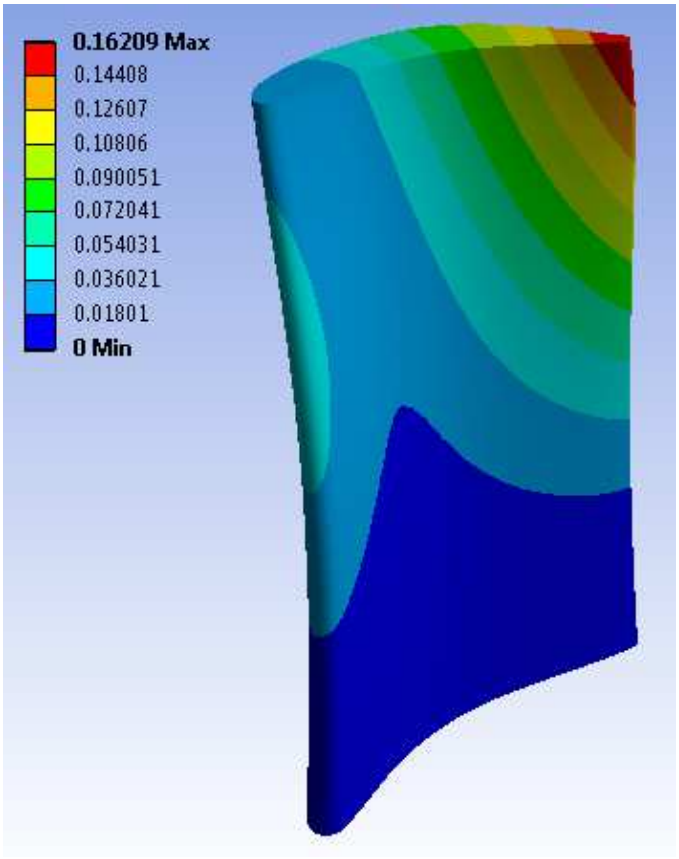

Figure A.10: Total displacement solution for the combined load case