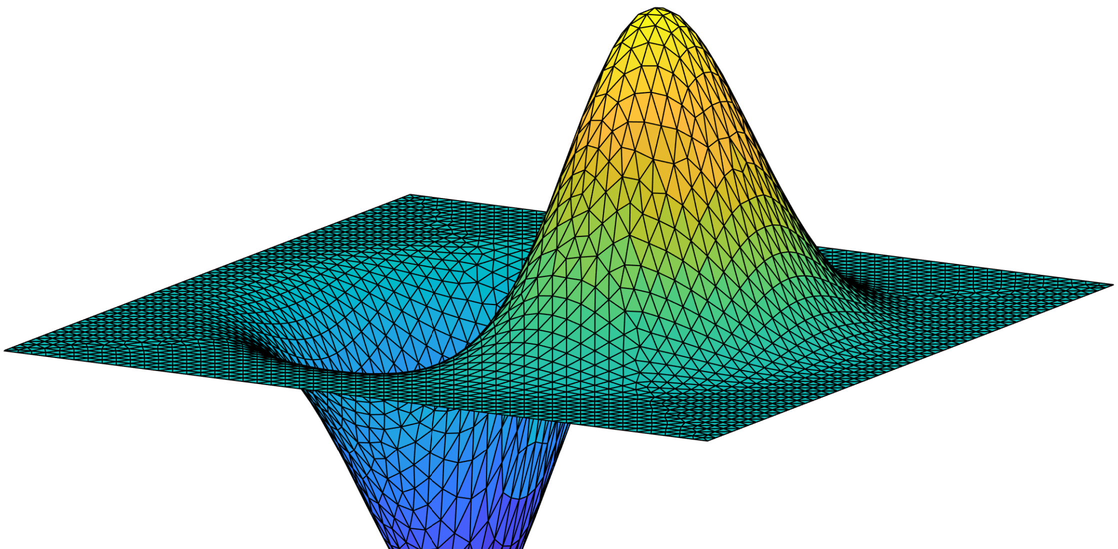
Local Enrichment Function Based Corrections Applied to Conventional RBF Mesh Deformation Methods

Bharadwaj Tallapragada



TU Delft

Local Enrichment Function Based Corrections Applied to Conventional RBF Mesh Deformation Methods

by

Bharadwaj Tallapragada

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Tuesday February 28, 2023 at 14:00.

Student number:	5323622	
Project duration:	April 11, 2022 – February 28, 2023	
Thesis committee:	Dr. ir. B.W. van Oudheusden,	TU Delft, chair
	Dr. ir. A. H. van Zuijlen,	TU Delft, supervisor
	Dr. ir. S.G.P. Castro,	TU Delft

Acknowledgements

This thesis draws a *meshy* curtain to my journey as a *Meshter* student at TU Delft. The journey was wrought with many challenges, especially with a pandemic raging at the beginning. Looking back at this period does fill me with immense joy because of the amazing people I met who made this journey definitely worth it.

First of all, I have to profusely thank Sander for being such a great mentor during the thesis. The patience and kind support you provided even after the number of times that I *meshed* up really meant a lot. Your insight and attention to detail were so helpful when I was stuck debugging my code. More often than not I always found the problem where you suggested it might be without even looking at the code. I really had a great time working with you over the last year.

I would also like to thank Michiel, Olaf, Friso and the entire Dimple/MuTech crew. I had an amazing time interning there and having the opportunity to stick around after that as well. I would like to think I picked up a few hands-on skills for performing experiments, but I guess you all will disagree quite vocally. I would also like to specifically thank Max, Nikhil and Sahana for making the coffee breaks really fun and for constantly checking up on my thesis work. Your constant reminders to ”not be weak” and get a double espresso probably made me wrap up my thesis faster.

I would be amiss not to thank my friends Loucas and Daniel. I think my life in Delft would have been very boring if not for you guys. The board game nights and bouldering over the last few months really pulled me through this stressful period. I would also like to thank Avinab for being an amazing cook and a great friend.

Finally, no words can express my gratitude towards my parents and brother. Your constant support and love over the last couple of years have kept me going when things looked tough. Without your support, I would have not made it this far.

Bharadwaj Tallapragada
Delft, February 2023

Contents

List of Figures

List of Tables

Nomenclature

Acronyms and Abbreviations

ALE	Arbitrary Eulerian Lagrangian
CFD	Computational Fluid Dynamics
CP	Compuact Support
CSM	Computational Structural Mechanics
FEM	Computational Fluid Dynamics
FSI	Fluid Structutre Interactions
FVM	Computational Fluid Dynamics
GCB	Grouping Circular Based
IB	Immersed Boundary
IMQB	Inverse Multiquadric Biharmonics
LU	Lower Upper
MLDR	Multilevel Subspace Method
MQB	Multiquadric Biharmonics
ONERA	Office National d’Etudes et de Recherches Aérospatiales
PDE	Partial Differential Equations
PoU	Partition of Unity
QB	Quadric Biharmonics
RBF	Radial Basis Function
RC	Recurrence Cholesky
TPS	Thin Plate Spline

Greek Symbols

α	Interpolation Coefficient
α_k	Determinant
β	Correction Constant
β_i	Weighting Coefficient
χ	Deformation
ϵ	Error
η	Efficiency
γ	Weighting Coefficient
λ_{ij}	Matrix element
ϕ	RBF
ξ	x/r

Non-Greek Symbols

$\mathcal{F}$	Functional
$\mathcal{T}$	Triangulation
A_k	Jacobian of element
R_l	Enrichment Function Support Radius
$\mathbf{n}$	Unit Normal Vector
d	Displacement
H	Enrichment Function
J	Imposed gradient
P	Polynomial
R	Support Radius
r	Distance
S	Deformation function
t	Time
u,v,w	Velocities in x,y and z directions
x	Node Location

Superscripts

new	Updated Variable
T	Transpose

Subscripts

∞	Freestream
b	Boundary Nodes
c	Control Point Subset
i, j, k	Node numbers
in	Internal Nodes
max	Maximum
opt	Optimal
ss	Size-skew
x, y, z	x,y,z direction

Executive Summary

A popular class of methods for solving Fluid-Structure Interactions (FSI) problems computationally are the moving mesh deformation methods. These methods generally involve deforming the fluid domain so that it conforms to the structure as it undergoes changes. An efficient and robust moving mesh technique involves interpolation of displacements using Radial Basis Functions (RBFs). But it is noticed that large deformations in the computational domain restrict the applicability of RBF methods due to resulting poor mesh quality. This further worsens the accuracy of the simulation by introducing additional numerical errors. The standard practice to overcome this deficiency has been to introduce the computationally expensive step of re-meshing the entire domain.

The current thesis aims to tackle this issue of poor mesh quality caused due to large deformations of the structure by introducing localized corrections in regions of poor quality. But deforming the mesh, then calculating the quality and then applying corrections would become an expensive operation of its own. Therefore, firstly a methodology is devised to predict the state of the mesh after the deformation apriori to performing the deformation step. It is found that the gradient information from the deformation functions can be utilized to accurately predict the various algebraic mesh quality metrics. The algorithm is tested on 1D and 2D meshes and grid convergence studies are performed to validate the predicted quality with the actual quality post-deformation. The theoretical framework for the 3D prediction algorithm is also laid out.

Once the mesh quality is predicted, based on a user-defined threshold, if the quality is deemed to be poor in a certain region then localized corrections are implemented to improve the mesh quality. These corrections are introduced in the form of enrichment functions on additional control points within the domain. The gradient of the deformation function is constrained using these functions such that a good quality and smooth mesh deformation is obtained post-deformation. Multiple enrichment functions are initially tested in 1D in a global sense. The findings from this analysis are then carried forward to analyze where to introduce these additional control points for local corrections and the nature of the correction itself.

Finally, the methodology is tested on 2D unstructured grids. It is found that the imposition of gradients is more complicated in the 2D sense as global information can not be used directly as in the 1D case. Therefore, a parametric study is performed to analyze if a logical conclusion can be made with regard to the imposition of the gradients. It is found that such a logical trend is not clear, therefore the strategy is modified such that the interpolation coefficients are imposed explicitly instead based on data available from eigen decomposition of the deformation gradients. It is found that this strategy performs admirably when compared to the results obtained from the standard RBF model. In conclusion, the method significantly improves the robustness of the mesh deformation process by ensuring that large deformations can be accommodated in the domain without huge computational costs.

1 Introduction

In the current chapter, the motivation behind selecting this particular topic for a thesis is provided along with a brief description of the background information required. Furthermore, the questions that are aimed to be answered at the end of the thesis are framed following which a brief overview of the structure of the report is presented.

1.1. Motivation

The study of FSI has been of paramount importance to engineers since scientists began to note that the complex interactions between a deforming body and a fluid could lead to catastrophic structural failures, as was noted in the infamous case of the Tacoma narrows bridge collapse, which occurred due to aeroelastic flutter caused by the development of Von Karman vortex streets[6]. Such interactions showcased that it is not always possible to study structural and fluid mechanics in a decoupled architecture. Furthermore, it was noted that the research on FSI is a critical necessity in various other domains as well, such as arterial flows in medical sciences[2, 4], wind turbine design[3, 24], flows in automobile and aircraft engines, partially filled liquid containers[37] and so on. Since experimental analysis is a very costly operation and sometimes is too complex to be implemented, it is necessary to develop robust and efficient computational methods that can accurately capture this complex multi-physical interaction and provide reliable results. Over the years, several different methods have been developed with this aim.

But it is noticed that a lot of challenges are involved when solving for large structural deformations due to issues with computational efficiency and poor mesh quality. The issue with poor mesh quality is especially critical as large deformations can often lead to inversion of cells within the domain thereby leading to the termination of simulations. One of the way around this problem is to introduce a remeshing step whenever the mesh quality becomes too low. But this method is computationally very expensive. Therefore, the objective of the current thesis is to develop a novel framework of predicting the quality of the mesh deformation without actually deforming the mesh, so that additional constraints can be introduced into the deformation step such that the mesh quality does not fall below a certain user-defined threshold.

1.2. Background

In order to solve an FSI problem, there are two approaches that can be undertaken, based on whether the solver performs structural as well as fluid mechanics calculations in a single framework, or whether a modular approach is utilized by integrating two separate CFD and CSM solvers, then coupling their solutions together using a pre-defined interface. The monolithic approach i.e. the singular framework approach leads to better computational efficiency as it does not involve any coupling algorithms between the solid and the fluid domain. On the other hand, mathematically it is preferred to solve the structural mechanics in a Lagrangian framework (FEM) where every node in the structural mesh is accounted as a structural particle and moves according to the deformation load applied. Whereas, the fluid domain is preferably treated under an Eulerian framework (FVM) as it is a purely conservative method i.e. since the fluxes are defined over the cell faces, which are common for adjacent cells, the conservation laws hold true even at a local level. Both these formulations are incompatible with each other. Furthermore, solving the structural mechanics involves differential equations with higher order in comparison to the Navier-Stokes equations leading to increased complexity. Hence, it is generally preferred to use modular or partitioned solvers while solving FSI problems as they allow the structural and fluid mechanics to be solved under two different grids. The results from one mesh to the other are transferred using a coupling algorithm along the pre-defined interface. The only disadvantage is convergence becomes difficult to achieve under certain conditions. Therefore partitioned solvers are generally preferred due to their greater flexibility. A depiction of the various facets of partitioned solvers is presented in Figure 1.1

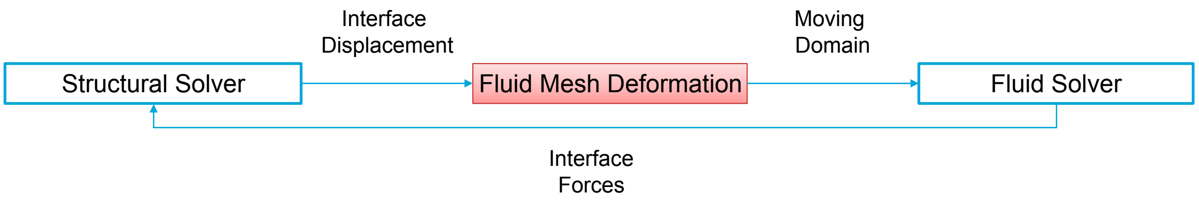


Figure 1.1: Steps involved in partitioned solvers for FSI

The objective of the current thesis is focused on the fluid mesh deformation part in the figure above. Since the deformation of a structure leads to a change in the fluid domain boundary as well, it is important that the structural displacements are accurately captured in the fluid mesh. Otherwise, at each time step a re-meshing algorithm needs to be applied, which would generate a completely new mesh with the new connectivity. This becomes more difficult when complex geometries are involved as they require an unstructured mesh, which usually requires significant user input in order to produce a good mesh. Furthermore, this approach would consume a lot of computational power and time, especially when large or well-refined domains are involved.

Therefore, in order to capture structural deformations on the same mesh and avoid having to re-mesh the entire domain, several methods have been developed which can be broadly classified into two categories: fixed mesh approaches and moving mesh approaches. As the names suggest, fixed mesh methods utilize a fixed background mesh for the fluid domain, over which the structural presence is administered using various methods. The moving mesh methods, on the other hand, are based on an ALE formulation[26, 15], where the fluid mesh deforms along with the solid body to conform to its new shape/location post-deformation. This conformity is maintained through a mapping algorithm, which is based on the location of the solid at the current time step. Based on whether the mesh is structured or unstructured, the technique utilized to perform this mapping might differ. Fixed mesh methods can perform well when the structure undergoes large displacements as a rigid body. But, they tend to perform poorly when the structure itself is deforming due to poor boundary layer cell quality, which is a major problem as capturing boundary layer dynamics accurately is of critical importance for FSI applications. Therefore, the current thesis will concentrate on the moving mesh approach. A succinct description of the computational effort required for several of the moving mesh approaches is provided in Table 1.1.

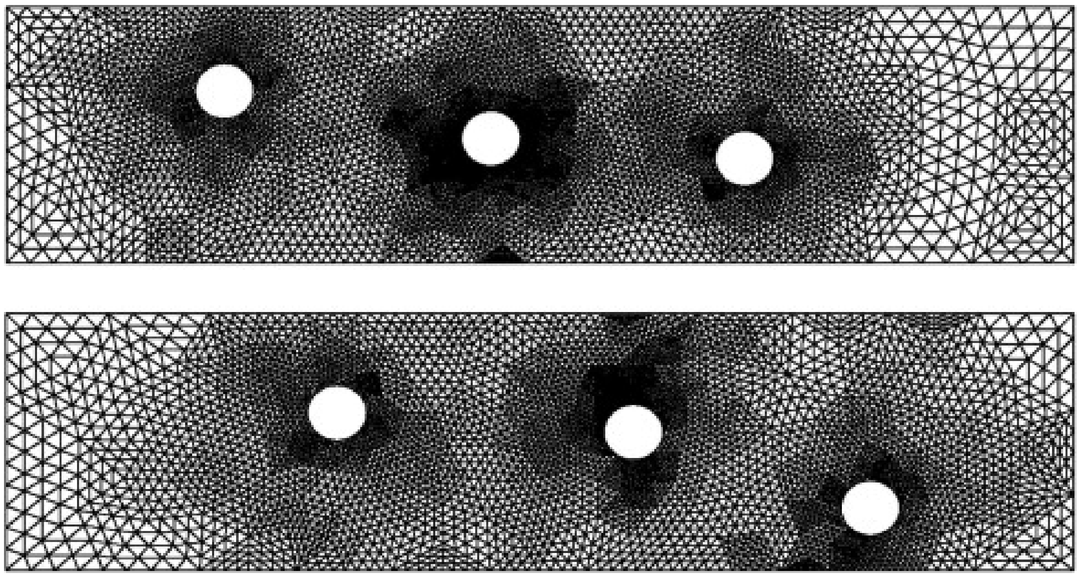


Figure 1.2: Remeshing at each time step as the particle moves within the domain. Top: at t, Bottom: at $t + \delta t$ [25]

	Method	Computational effort
1	Master-Slave coupling	
2	Lineal springs	Heavy $(\mathcal{O}(N_{\text{grid}})$
3	Torsional springs	Heavy $(\mathcal{O}(2 \cdot N_{\text{grid}}))$
4	Semi-torsional springs	Heavy $(\mathcal{O}(N_{\text{grid}}))$
5	Least Squares	Very Heavy
6	Solid Body Elasticity	Very Heavy
7	Laplacian Smoothing	Heavy $(\mathcal{O}(N_{\text{grid}}))$
8	Biharmonic operator	Heavy $(\mathcal{O}(2 \cdot N_{\text{grid}}))$
9	Radial basis function interpolation	Medium $(\mathcal{O}(N_{\text{boundary}}))$

Table 1.1: Computational cost of different moving mesh methods[40]

Among all the interpolation methods, the RBF formulation generates the most sparse system and hence, will be selected for further study in this report.

The fluid mesh is deformed after every time step based on the displacement of the structure from the previous time step or from its initial position i.e. based on the relative displacement method or absolute displacement method. Each method has its own advantages and disadvantages that need to be considered before implementation. Furthermore, the deformation step introduces several challenges as the truncation error varies with every time step due to variations in cell size and can also lead to instances with hanging nodes[10]. These hanging nodes. depicted in Figure 1.3, lead to a poor quality mesh after deformation as they do not have sufficient neighbouring nodes which leads to a mismatch in equilibrium balance. In some cases with large deformations, the cell itself might get inverted leading to negative volumes/degenerate cells, which have an adverse effect on the stability and accuracy of the solution. These considerations would be further discussed in chapter 2

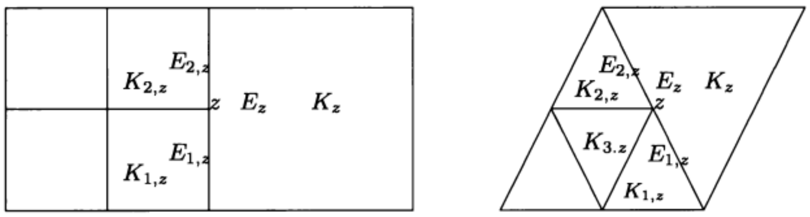


Figure 1.3: Elements depicting hanging nodes: Edge E_z is a hanging edge with two children $E_{1,z}, E_{2,z}$ [10]

The general remedy for poor quality meshes due to large deformations in the moving mesh set-up in the past has been to introduce remeshing with completely new connectivity from scratch[32, 31]. But in recent years, more efficient methods have been introduced such as a mixture of fictitious domain and ALE methods[14, 12], which provide greater robustness for large deformations.

Therefore, from the above discussion, it can be deduced that it is a very interesting challenge to develop a method of deforming meshes which will ensure a better quality of meshes when undergoing large deformations. But in order to do so in a computationally efficient manner, there is a lot of scope for further research. This leads to the possibility of implementing localized corrections to the deforming mesh using the RBF interpolation method whenever the quality of a cell goes below a certain threshold.

1.3. Research Objectives

In light of the literature study performed before the beginning of this project, the following research question, which forms the basis of this report is arrived at:

”When a structure undergoes large deformations or displacements within the domain, can the application of localized enrichment functions within the domain improve mesh deformation quality and computational efficiency?”

In order to answer the research question posed above, it is necessary to consider the multiple facets involved in the implementation of the localized correction algorithm. Therefore, it is important to frame sub-questions that will help ensure all the underlying considerations are accounted for:

1. How do the localized enrichment functions help the mesh deformation algorithm perform better in comparison to the regular RBF interpolation-based mesh deformation method?
   - Is there a significant gain in the final quality of the obtained mesh?
   - How significant is the gain in quality in comparison with the computational resources expended?
2. Which mesh quality metric should be utilized for setting the threshold?
   - Is it possible to predict when/where to perform the mesh quality calculation?
   - How cheap is this operation?
3. How much effect will there be on the accuracy of the final displacement due to localized interpolation of displacements?
   - Can control point reduction algorithms be applied to the local RBF correction algorithm without having any detrimental effect on the mesh quality?
4. How is the mesh quality and efficiency impacted when control point reduction algorithms are applied to the entire system?

- What is an acceptable level for the error tolerance and how does it impact the mesh quality and efficiency of the algorithm?

By framing these sub-questions it is possible to provide an in-depth performative analysis of the algorithm being developed in comparison to the original algorithm as well as its derivatives that have been developed to overcome the challenges associated with large deformations. Furthermore, the ALE method in itself is not the most efficient mesh deformation algorithm. Even if the developed algorithm provides very accurate results, it would not be acceptable at the cost of an exponential increase in computational time. Therefore, it is of paramount importance to ensure that the computational cost involved in order to implement this localized optimization is minimized.

1.4. Structure of Report

The current report is structured as follows: chapter 2 looks at the theoretical knowledge available in the literature on the RBF mesh deformation methods including the methodology, optimization routines and the various mesh quality metrics available. Chapter 3 delves into the methodology of predicting mesh quality without having to deform the mesh. Chapter 4 then goes on to explain in a 1D setup, how to utilize the predicted quality information in order to impose enrichment functions in the domain to constrain sharp gradients in the deformation. Chapter 5 then goes on to expand the enrichment methodology in 2D and the challenges associated with it. Finally, 6 provides the conclusions that could be drawn from the current project and gives suggestions on how the current work can be expanded in the future.

2
Theoretical Background

The current chapter provides a theoretical foundation required for the current thesis. Firstly the methodology of the conventional RBF interpolation-based mesh deformation is described in section 2.1. Then the types of RBFs that can be utilized in order to perform the interpolation are presented and an analysis is also presented on which functions perform this process optimally to fulfil the research objectives. Furthermore, the necessity of utilizing the polynomial function in the formulation and the choice of displacement method that would be ideal for the application of the current thesis is discussed.

Furthermore, section 2.2 provides a discussion on the various optimization algorithms available in the literature that can be utilized to make the mesh deformation process more streamlined. Special emphasis is put on the greedy algorithm developed by Rendall and Allen[38, 39] and its subsequent derivative algorithms to make sure that the efficiency of the current approach can be further improved. This is a critical section of the implementation as moving mesh methods are very computationally taxing when the grid size becomes large. Further adding a localized interpolation step will surely drive the efficiency lower, which might lead to a situation where the computational cost is not realistic with regards to the solution accuracy.

Finally, the mesh quality metrics that are best suited to qualify as the base criterion for implementing the localized corrections are discussed in section 2.3. The choice of the criterion is of critical importance to maintain high efficiency, as performing the large calculations involved in calculating mesh quality after each deformation step is not a feasible operation. Therefore, it is necessary to analyze which mesh properties have the most significant impact on the final quality of the mesh. An important consideration to make while looking at the mesh quality is that it is recommended to look at the minimum value of the selected qualifying criterion rather than considering the average quality over the entire mesh. The minimum quality metric is more significant because even a single degenerate element can cause the whole simulation to fail as demonstrated by deBoer, Schoot and Bijl[7].

2.1. RBF Interpolation Method for Mesh Deformation

The use of RBF based interpolation techniques has become quite popular for performing mesh deformations since it does not require any grid connectivity information[7]. Initially a projection method is used to transfer the known displacement values from the structural boundary points to the fluid mesh interface. Then the RBF interpolation method utilizes the boundary node displacement values at the interface to interpolate the displacements of the internal nodes of the fluid mesh. The RBFs operate in such a way that the displacement from the boundary points can be transferred to the internal points using an interpolation function, ’s’, which can be represented in the domain as a sum of basis functions, given as:

$$\mathbf{s}(\mathbf{x}) = \sum_{j=1}^{N_b} \boldsymbol{\gamma}_j \phi(\left\|\mathbf{x} - \mathbf{x}_{b_j}\right\|) + \mathbf{p}(\mathbf{x}), \qquad j \in [1, N_b] \tag{2.1}$$

where, $\mathbf{x}$ represents $[x, y, z]$ and x_{bj} are called as centres and represent the boundary points at which the displacements are already known. The RBF’s extend from these centres in a symmetric fashion. The function might be global or local in nature i.e. the function at these centres can affect all the internal points within the domain regardless of location or the function can exert influence only over a certain range determined by the support radius. The choice of the domain of influence has an impact on the smoothness of the final solution and is discussed in detail in subsection 2.1.1.

Furthermore, N_b in Equation 2.1 is the number of boundary points, p is a polynomial set incorporating the three spatial directions, ϕ is the chosen basis function and γ_i is a weighting coefficient. The polynomial coefficients, β_i and the weighting coefficients are evaluated based on the interpolation conditions, which are given as:

$$\mathbf{s}(\mathbf{x_{bj}}) = \mathbf{d_{bj}} \tag{2.2}$$

$$\sum_{j=1}^{N_b} \boldsymbol{\gamma}_j \mathbf{q}(\mathbf{x_{bj}}) = 0 \tag{2.3}$$

where $\mathbf{d_{bj}}$ is the discrete values of the displacements of the boundary nodes which is already known, $\mathbf{q}$ represents all the polynomials with degrees lower than or equal to the polynomial $\mathbf{p}$. The first interpolation condition ensures that the interface location on the fluid mesh coincides with the displacement values that are attained from the structural mesh. While the second interpolation condition ensures that the contribution of the shape functions does not represent anything that can be represented by just using the polynomial. Thereby, allowing a unique solution to be obtained for a given problem. For this to be possible, the interpolation matrix has to be positive definite[9], which implies that the RBF has to be positive definite, which is generally the case apart from some exceptions such as the thin plate spline. The degree of the polynomial is dependent on the type of basis function being used.

Using the above conditions, it is possible to calculate the values of the weighting coefficients and the polynomial coefficients by solving the system:

$$\begin{bmatrix} \mathbf{d_b} \\ 0 \end{bmatrix} = \begin{bmatrix} \phi_{bb} & P_b \\ P_b^T & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma}_i \\ \boldsymbol{\beta}_i \end{bmatrix}, \qquad i = x, y, z \tag{2.4}$$

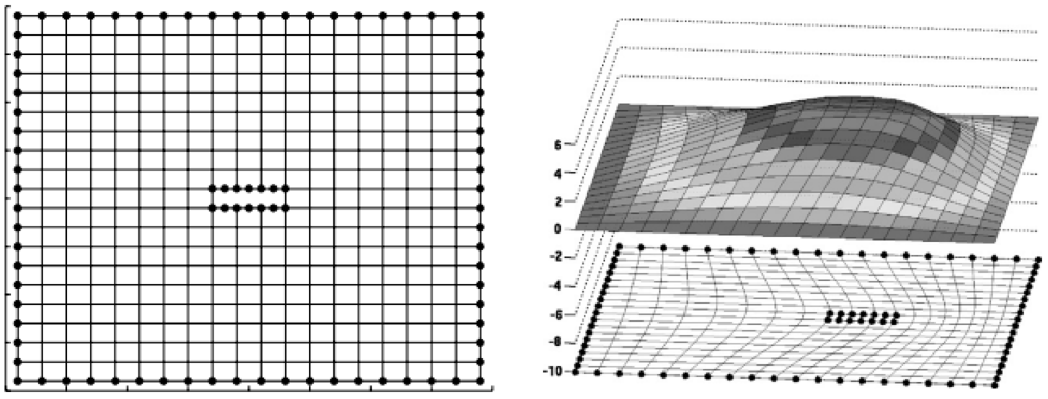
Here, ϕ_{bb} is the interpolation matrix of the size $N_b \times N_b$ comprising the evaluated basis functions and P is a $N_b \times 4$ matrix represented by $\begin{bmatrix} 1 & x_b & y_b & z_b \end{bmatrix}$. Therefore, in the end, the size of the system that has to be solved in Equation 2.4 is $(N_b + 4) \times (N_b + 4)$. This system can be easily solved using fast iterative methods[11] or using PoU approach[38] and is usually far more efficient than other moving mesh approaches, where generally matrix sizes of $N_{in} \times N_{in}$ are encountered, where N_{in} is the total number of interior nodes in the mesh.

After the polynomial and weighting coefficients are obtained, the displacements of all the internal mesh nodes can be calculated by applying the interpolation function to the internal nodes:

$$\mathbf{d}_{\mathbf{in_j}} = s(\mathbf{x}_{in_j}) = \sum_{j=1}^{N_b} \boldsymbol{\gamma}_i \phi(\left\|\mathbf{x}_{in_j} - \mathbf{x}_{bj}\right\|) + \mathbf{p}(\mathbf{x}_{in_j}), \qquad j \in [1, N_i] \tag{2.5}$$

Equation 2.5 can be written in the matrix form as,

$$\begin{bmatrix} \mathbf{d}_{\mathbf{in}} \\ 0 \end{bmatrix} = \begin{bmatrix} \phi_{inb} & P_{in} \end{bmatrix} \begin{bmatrix} \phi_{bb} & P_b \\ P_b^T & 0 \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{d}_{\mathbf{b}} \\ 0 \end{bmatrix}, \qquad i = x, y, z \tag{2.6}$$



(a) Initial mesh

(b) Interpolation function and final mesh

Figure 2.1: Visualization of RBF interpolation for mesh deformation[7]

2.1.1. Types of RBF interpolation functions

As mentioned earlier, the RBF interpolation functions can be broadly classified into two categories i.e. global support RBF and local support RBF. Functions with global support have an influence on all the mesh nodes within the domain which leads to a smoother solution. But this happens at the expense of efficiency as the underlying matrix to be solved becomes relatively dense. Therefore, generally it is preferred to utilize locally supported RBFs which have zero influence in the domain after a certain support radius. They can be represented as:

$$\phi(x) = \begin{cases} f(x) \geq 0, & 0 \leq x \leq 1, \\ 0, & x > 1 \end{cases} \tag{2.7}$$

$f(x)$ is generally constructed in such a way that it scales with the support radius of the function. As the support radius increases, the function becomes more smoother and hence, more accurate. On the other hand, it leads to denser matrix systems, thereby increasing computational costs.

Multiple studies have been conducted on the performance of various RBFs in order to analyze their effect on scattered data and mesh quality. A comprehensive study was performed by deBoer, van der Schoot and Bijl[7], who recorded the effectiveness of eight locally supported RBF's given by Wendland[46] along with six globally supported RBF's.

No.	Name	$f(\xi)$
1	CP C^0	$(1-\xi)^2$
2	CP C^2	$(1-\xi)^4(4\xi+1)$
3	CP C^4	$(1-\xi)^6\left(\frac{35}{3}\xi^2+6\xi+1\right)$
4	CP C^6	$(1-\xi)^8\left(32\xi^3+25\xi^2+8\xi+1\right)$
5	CTPS C^0	$(1-\xi)^5$
6	CTPS C^1	$1+\frac{20}{3}\xi^2-40\xi^3+15\xi^4-\frac{8}{3}\xi^5+20\xi^{z^2}\log(\xi)$
7	CTPS C_3^2	$1-30\xi^2-10\xi^3+45\xi^4-6\xi^5-60\xi^3\log(\xi)$
8	CTPS C_b^2	$1-20\xi^2+80\xi^3-45\xi^4-16\xi^5+60\xi^4\log(\xi)$

Table 2.1: RBF's with local support[46], studied in[7]

In Table 2.1, ξ represents r/R, implying that the RBFs are scaled with respect to the support radius chosen by the user.

No.	Name	Abbreviation	$f(x)$
9	Thin plate spline	TPS	$x^2\log(x)$
10	Multiquadric biharmonics	MQB	$\sqrt{a^2+x^2}$
11	Inverse multiquadric biharmonics	IMQB	$\sqrt{\frac{1}{a^2+x^2}}$
12	Quadric biharmonics	QB	$1+x^2$
13	Inverse quadric biharmonics	IQB	$\frac{1}{1+x^2}$
14	Gaussian	Gauss	e^{-x^2}

Table 2.2: RBF's with global support studied in [7]

In general, with regards to interpolation of scattered data, the multi quadratic biharmonic (MQ) performed best[22]. But, deBoer, Schoot and Bijl[7] found that in terms of mesh quality, five RBFs produced high-quality meshes after deformation. But, Wendland's C^2 continuous basis function with compact support (CP C^2)[46] and Thin Plate Spline(TPS) functions perform better in terms of efficiency. The functions are given as:

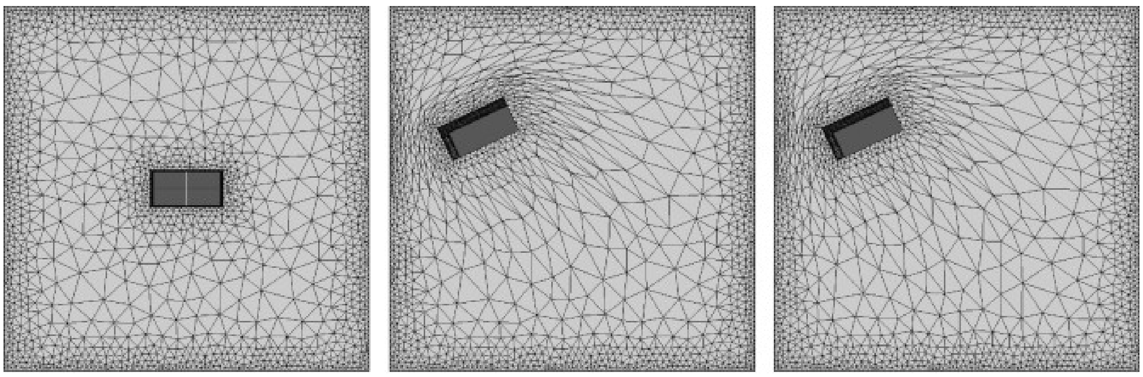
- CP C^2: $\phi(\xi) = (1-\xi)_+^4\,(4\xi+1)$
- TPS: $\phi(x) = x^2 log(x)$

The TPS function is a global basis function which leads to a dense system. Furthermore, evaluating the logarithmic function involves greater computational costs leading to lower efficiency of the algorithm compared to the CPC^2 function. This leads to the general preference of the CP C^2 function for mesh deformation methods due to greater efficiency as showcased in multiple studies[8, 38, 39]. Therefore, for the current study, it was decided to utilize the CP C^2 function for further analysis.

2.1.2. Polynomial function

As discussed in earlier sections, the polynomial function ensures that a unique solution is attained. This is especially useful when the structure undergoes rigid body translation or rotation within the domain, allowing the recovery of the exact movement. However, when the structure undergoes elastic deformation, the polynomial function does not provide any additional benefits and hence, can be neglected from the formulation, especially while using Wendland's C^2 RBF which is strictly positive definite, thereby making the interpolation condition represented in Equation 2.3 to be empty[5]. Furthermore, it was found by Rendall and Allen[38] that the polynomial function can cause far-field boundary nodes to deform, which are supposed to remain stationary.

Furthermore, as seen from Equation 2.4 the polynomial functions are present at the ends of the interpolation matrix, which increases the bandwidth of the entire matrix. This leads to the loss of sparseness that is obtained by the usage of locally supported RBFs. This leads to an overall loss in efficiency for very little gain in quality as shown in Figure 2.2, which depicts a prismatic block undergoing severe rigid translation and rotation[17]. Hence, it is prudent to ignore the addition of the polynomial function unless a globally supported function like the TPS is used, which leads to the interpolation matrix remaining dense regardless of the presence of the polynomial function. Since only locally supported basis functions will be used in the current study, it is safe to ignore the polynomial term in the formulation without an adverse impact.



(a) Initial mesh.

(b) Final mesh considering polynomial term. Relative aspect ratio = 0.276

(c) Final mesh discarding polynomial term. Relative aspect ratio = 0.252

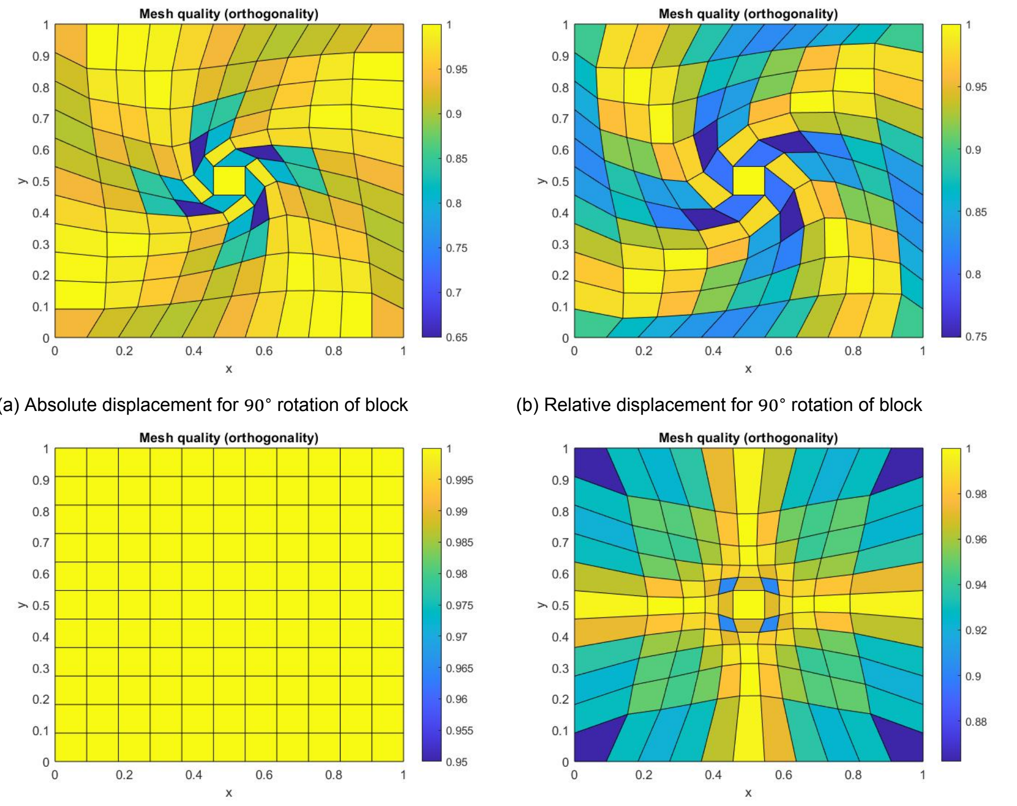
Figure 2.2: Effect of polynomial term [17]

2.1.3. Choice of displacement method

An important consideration to make when implementing the ALE formulation is whether the displacements are considered in an absolute sense or relative sense. As explained in the earlier sections, absolute displacement implies that the displacements of the internal nodes after each time step are based on the location of the same nodes in the initial original mesh. Whereas, for relative displacements, the movement of the nodes is based on the location of the same nodes in the previous time step. A visual representation of the mesh orthogonality of a rotating block for both methods after one cycle is represented in Figure 2.3.

Under the absolute displacement method, the interpolation matrix is built with regard to the initial position of the mesh nodes. This leads to the advantage that the interpolation matrix has to be constructed only once, which increases efficiency. This is especially useful when the structure undergoes periodic deformations as the mesh can always return to its initial configuration, which leads to a higher mesh quality over time.

Whereas, relative displacements generally provide a more accurate interpolation scheme when large displacements are involved. This happens because the interpolation is performed over a smaller displacement i.e. from the previous time step instead of the initial time step. But, it has to be noted that in cases of periodic motions, they perform poorly when compared to absolute displacements as small interpolation errors accumulate over each time step as shown in Figure 2.3. Therefore, the mesh may not return to its original configuration at the end of the periodic motion leading to lower mesh quality. But, in the case of non-periodic motions, relative displacements perform better than absolute displacements. Therefore, the relative displacement method will be utilized in the current study since the main area of interest is large deformations, which might not be necessarily periodic in nature.



(a) Absolute displacement for 90° rotation of block

(b) Relative displacement for 90° rotation of block

(c) Absolute displacement for periodic motion after one cycle

(d) Relative displacement for periodic motion after one cycle

Figure 2.3: Visualization of mesh orthogonality for a rotating block[13]

2.2. Optimization Strategies

2.2.1. Types of optimization strategies

The ALE-based moving mesh approaches are found to be very robust and accurate for small structural displacements and ensure that the conservation properties are always satisfied. But, problems arise whenever strong deformations or even topological changes of the interface lead to a degeneration of the computational mesh. To deal with large structural displacements, re-meshing has to be introduced, which leads to increased computational costs and completely new connectivity. Furthermore, topology changes are sometimes difficult to capture as well. The ALE methods are also computationally very expensive, especially in cases involving 3D meshes, which is not ideal. Several optimization strategies have already been developed to ensure that the time taken by these methods is reduced without loss in accuracy as discussed in this section. These optimization strategies can be coupled with parallel computing algorithms to further increase their efficiency as shown by Gao et al.[23], Wang et al.[44] and Gillebaart et al.[24].

A multi-step approach was presented by Floater and Iske[21] that sought to perform the interpolation on a coarse set of boundary nodes initially. This step was followed by interpolating on a refined set of selected boundary nodes, whilst further reducing the size of the support radius. These steps are repeated until a satisfactory accuracy level is obtained for the interpolation. The general algorithm for a multi-step approach is depicted in Figure 2.4. Wang et. al.[44] further developed this multi-step mesh deformation method for RBF interpolations. They utilized a 'double-edge' greedy supporting point selection algorithm using a multi-level subspace method (MLDR method) where the deformation procedure is divided into several classes which can be solved in parallel, thereby, increasing efficiency[45].

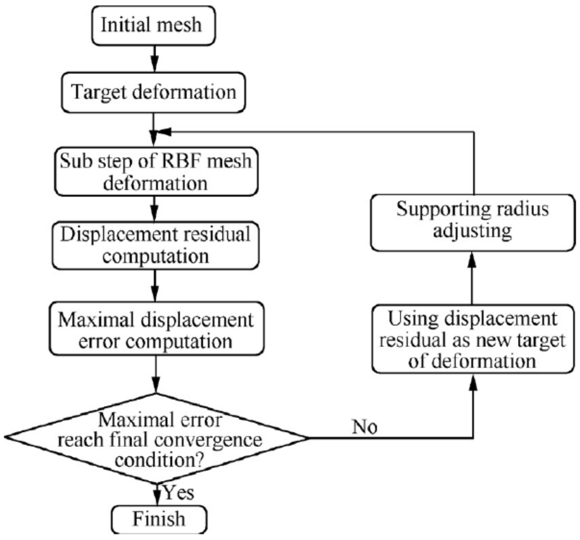

Initial mesh
Target deformation
Sub step of RBF mesh deformation
Displacement residual computation
Maximal displacement error computation
Maximal error reach final convergence condition?
Yes
Finish
No
Using displacement residual as new target of deformation
Supporting radius adjusting

Figure 2.4: Multi-step method algorithm for RBF interpolation based mesh deformation[44]

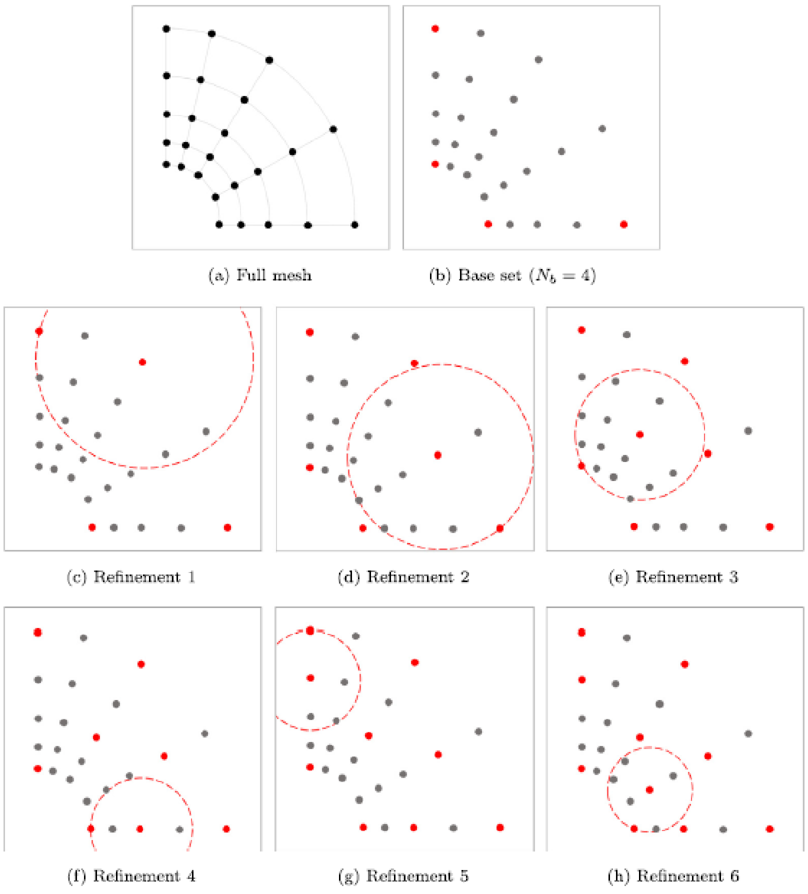


Figure 2.5: Multiscale optimization procedure[28]

Furthermore, a multi-scale method was devised by Kedward et al.[28]. In this method, an initial subset of boundary nodes is selected. After each iteration, a new boundary node is added to this set, which maximizes the separation distance of the active set of support nodes as shown in Figure 2.5. Even though this method leads to the addition of all boundary points into the active set, an increase in efficiency can be obtained by selecting an optimal support radius.

Volume node-based reduction algorithms can also be utilized to improve the efficiency of the RBF algorithm. These algorithms ensure that the internal nodes at which displacements need to be calculated are minimized. This can be done by restricting the deformation area by utilizing an enclosing box in which the nodes can be deformed and has zero displacements on its surface[34]. Furthermore, using restricted wall functions[47] or performing interpolations initially on a background cartesian grid[19] can reduce the overall data size.

2.2.2. Greedy Algorithm

The most common data reduction algorithm used for mesh deformation methods is the boundary node-based reduction algorithm which was developed by Jakosson and Amiognon[27]. This method was further developed by Rendall and Allen[38, 39], who proposed the most popular boundary-node-based data reduction algorithm in use, called the Greedy algorithm. This method is based on error-driven data reduction. The procedure begins with constructing a support set of nodes on the basis of a randomly selected set of boundary nodes. It then goes on to solve the interpolation matrix and then computes the interpolation error. The node which comprises the maximum interpolation error is then added to the support set. The error is generally calculated as the difference between the interpolated value and the known boundary point displacements. This process is repeated until the error goes below the pre-defined acceptable tolerance in the final solution. Then all the internal nodes are moved to their new locations based on the interpolated displacement values.

The inversion of the interpolation matrix in order to calculate the weighting coefficients every time a control point is added to the support set was found to be a costly operation. Therefore, Rendall and Allen further developed this ”full point” greedy method in order to form the greedy ”one point” algorithm. In this method, the inversion step is not carried out after the addition of every control point, but instead, an approximate correction is applied based on the maximum error, and the calculated displacements are updated. The correction constant is based on the value of the basis function at the maximum error point and is given by:

$$\boldsymbol{\beta} = \frac{f_{max}}{\phi(0)} \tag{2.8}$$

Using this, the interpolation coefficient of the maximum error point is approximated as:

$$\boldsymbol{\alpha}_{max}^{new} = \boldsymbol{\alpha}_{max} + \boldsymbol{\beta} \tag{2.9}$$

Therefore, the basis function can be corrected at the rest of the internal nodes as:

$$f_{in}^{new}(r) = f_{in}(r) + \boldsymbol{\beta}\phi(\|\boldsymbol{x} - \boldsymbol{x}_{imax}\|) \tag{2.10}$$

After performing this step, the errors are re-calculated based on the corrected displacements and known displacements. This process is repeated iteratively until the error goes below the tolerance level. Due to the application of an approximate correction, this method is not as accurate as the full point greedy algorithm. Therefore, generally, a combination of both these methods is used where the one-point algorithm is implemented, but after every, say n iterations, the full point approach is used to calculate the displacements. Thereby, ensuring that the interpolation coefficients are calculated correctly every once in a while so that the cumulative error reduces. The efficiency of this method in comparison to the full point and one point methods individually is presented in Figure 2.6, where the full update is performed after ten iterations.

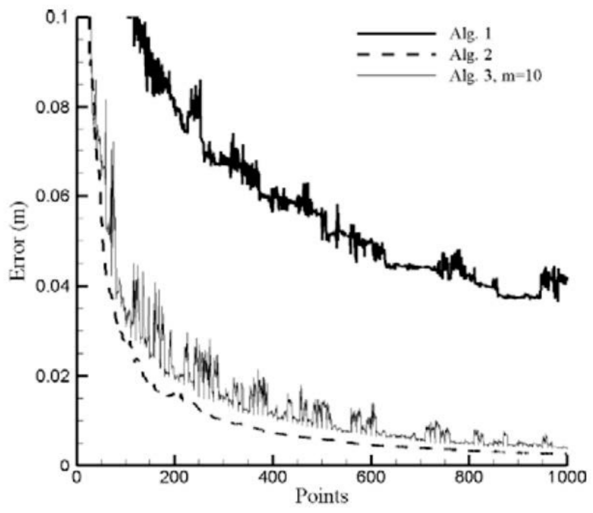


Figure 2.6: Convergence study of error for the three greedy algorithms[38]

The total computational cost of the greedy algorithm is estimated to be $\mathcal{O}(N_c^4 + N_c^2 N_b)$, where N_c is the support nodes and N_b is the total boundary nodes[20]. The first part involving the fourth-order term corresponds to the computational costs associated with solving the algebraic interpolation system, while the second part is associated with the calculation of the interpolation errors at the boundary nodes. Therefore, in order to have an efficient algorithm, N_c, should be minimized.

2.2.3. Customized greedy algorithms

In order to reduce the size of the support set, N_c, Li et al.[30] developed a modified greedy model which would select the support set nodes based on error peaks. The nodes with error peaks above a certain user-defined threshold would be added to the support set, which would increase the efficiency as it allows the addition of multiple points at the same time while ensuring that multiple points are not selected from the same region as shown in Figure 2.7. The initial boundary displacement is calculated based on the selected support set as,

$$\mathbf{d_b}^{optimized} = \boldsymbol{\phi}_{bc}\boldsymbol{\phi}_c^{-1}\mathbf{d_c} \tag{2.11}$$

where d_s are the displacement values of the boundary nodes in the support set, $d_b^{optimized}$ is the approximated displacements of all the boundary nodes and the subscript ”c” represents the support set. The interpolation errors are, therefore, calculated as,

$$\epsilon = \mathbf{d}_b - \mathbf{d}_b^{optimized} \tag{2.12}$$

Once the error is reduced to an acceptable level by incorporating additional points in the support set, the new optimal support set is used to calculate the interpolation coefficients of the RBF, shown in Equation 2.4.

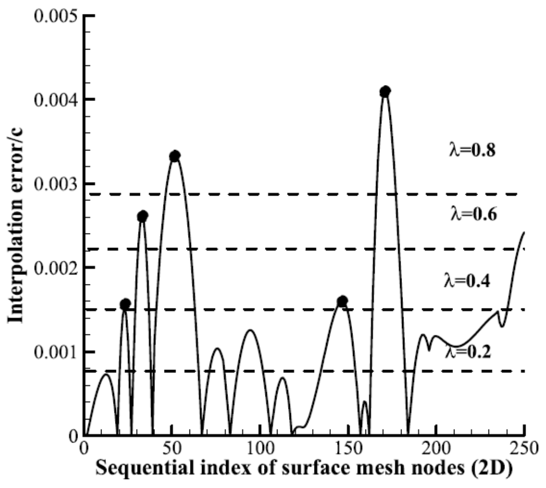


Figure 2.7: Selection of support set nodes based on user-defined error threshold[30]

A summary of the computational cost of a few custom greedy optimization algorithms in order to build the initial RBF model, given in Equation 2.11 while selecting the nodes for the support set is provided in Table 2.3.

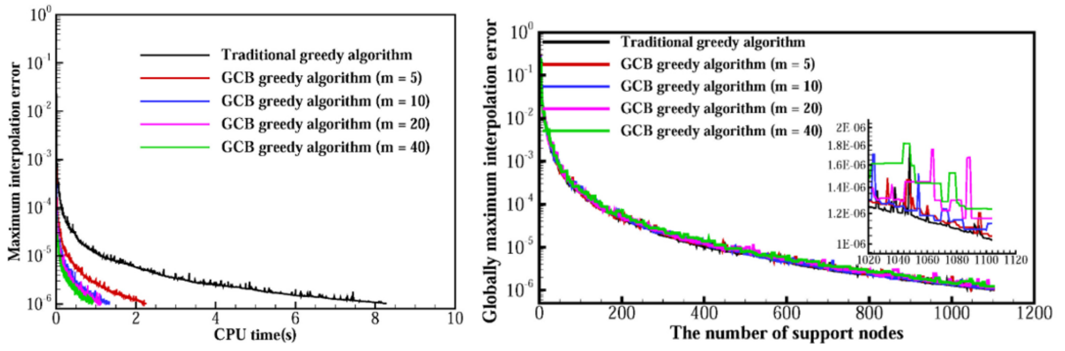
Method	**Computational Cost**
Greedy[38]	$\frac{N_c^4}{4}\left[1+O\left(\frac{1}{N_c}\right)\right]$
MLDR [45]	$\frac{N_c^4}{4\mathrm{M}}\left[1+O\left(\frac{\mathrm{M}}{N_c}\right)\right]$
Error threshold based Greedy[30]	$\frac{N_c^4}{4L}\left[1+O\left(\frac{L}{N_c}\right)\right]-\frac{(L-1)^2}{4L}$

Table 2.3: Computational cost of several optimization methods[30]

where M is the average number of support points specified at each level[45] and L is the assumed average number of nodes added to the support set after every iteration[30].

Furthermore, methods to reduce the order of the cost associated with solving the algebraic interpolation matrix system were developed by multiple authors. Selim et al.[41] utilized an incremental Lower-Upper (LU) decomposition algorithm. This method reduced the order of the first term of the computational cost of the conventional greedy algorithm from $N_c^4 \rightarrow N_c^3$. Fang et al.[18] suggested the implementation of a recurrence Cholesky (RC) decomposition scheme which further reduced the order of the first term to $N_c^3/2$ with the use of parallel computation. Generally, N_b is far greater in size than N_c, therefore, Strofylas et al.[43] proposed a multi-grid clustering algorithm that would reduce the number of boundary nodes, thereby leading to greater efficiency. But, this model is complicated in nature, which reduces its applicability to mesh deformation methods and hence, will not be in the scope of the current study.

Furthermore, Fang et. al.[20] developed a grouping-circular-based(GCB) algorithm which divides all the boundary nodes into m groups. It calculates the local interpolation error of each group and approximates it as the global interpolation error. This allows the utilization of all the boundary nodes in order to calculate the error. This method leads to the reduction in the interpolation error calculation cost from $O(N_c^2 N_b) \rightarrow O(N_c^3)$. The performance of the GCB algorithm in comparison with the traditional greedy algorithm is provided in Figure 2.8.



(a) Maximum interpolation error histories with regards to computational time

(b) Global maximum interpolation error histories with regards to support node-set size

Figure 2.8: Convergence study of conventional greedy and GCB greedy algorithm[20]

Therefore, with the aim of reducing the computational costs associated with the RBF interpolation, especially when a local interpolation, associated with optimizing the local mesh quality, has to be performed as well. It was decided to utilize the GCB algorithm developed by Fang et al.[20] as it accounts for all the boundary nodes, thereby increasing accuracy while being highly efficient at the same time. Additionally, the greedy algorithms can be coupled with a volume node reduction method as well to further increase efficiency. The greedy algorithm would reduce the number of control points selected, while the volume node reduction will ensure an optimized mesh update step[35].

2.3. Mesh Quality Metrics

In this section, the mesh quality metrics that are best suited to perform as a base criterion for implementing the localized corrections are discussed. The choice of the criterion is of critical importance to maintain high efficiency, as performing large calculations after each deformation step is not a feasible operation. Therefore, it is necessary to look into which mesh properties have the most significant impact on the final quality of the mesh. An important consideration to make while looking at the mesh quality is that it is recommended to look at the minimum value of the selected qualifying criterion rather than considering the average quality over the entire mesh. The minimum quality metric is more significant because even a single degenerate element can cause the whole simulation to fail as demonstrated by deBoer, Schoot and Bijl[7].

One of the primary objectives of the current research is to ensure that the mesh quality does not deteriorate when the structure is subjected to large deformations. Initially, when no deformation is applied, it is assumed that the mesh produced has optimal quality and is sufficient to resolve complex boundary layer phenomena as well. However, as the mesh deforms, the size and angle of the element begins to change as well. In order to define the quality of the mesh, a parameter is required that can track these fundamental metrics. In order to do so, a set of Jacobian matrices are used to quantify the quality of the mesh[29]. The Jacobian matrix comprises of basic information such as size, orientation, shape and skewness. These qualities are especially important because, in case of large deformations, the cells become highly skewed and can even lead to negative volumes which is not desirable at all. Furthermore, the calculation of the Jacobian is a robust method which can be applied to any mesh, be it structured or unstructured.

The Jacobian matrix of a cell node is a $d \times d$ matrix, where d is the dimension of the mesh element. It is denoted by $\mathbf{A_k}$, where $k = 0, 1, 2, ..., n-1$ represents the node numbers. The numbering of the nodes along with the Jacobians of commonly used cell types has been presented below.

Jacobian for triangular cell

Jacobian for tetrahedral cell

$$\mathbf{A_k} = \begin{bmatrix} x_{k+1} - x_k & x_{k+2} - x_k \\ y_{k+1} - y_k & y_{k+2} - y_k \end{bmatrix} \quad , \mathbf{A_k} = (-1)^k \begin{bmatrix} x_{k+1} - x_k & x_{k+2} - x_k & x_{k+3} - x_k \\ y_{k+1} - x_k & y_{k+2} - y_k & y_{k+3} - y_k \\ z_{k+1} - z_k & z_{k+2} - z_k & z_{k+3} - z_k \end{bmatrix}$$

Jacobian for quadrilateral cell

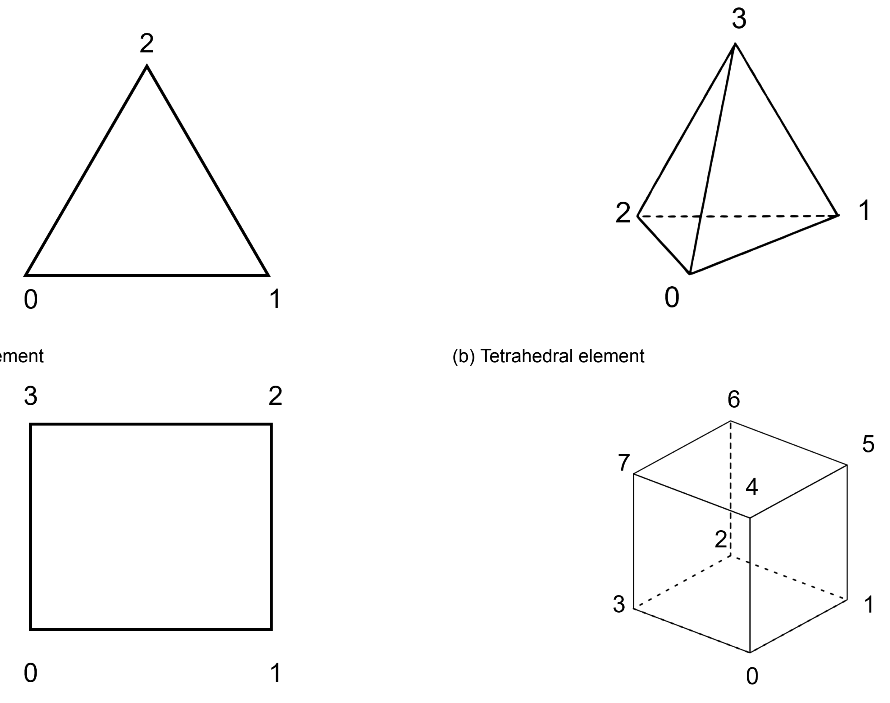
Jacobian for hexahedral cell

$$\mathbf{A_k} = \begin{bmatrix} x_{k+1} - x_k & x_{k+a} - x_k \\ y_{k+1} - y_k & y_{k+a} - y_k \end{bmatrix} \quad , \mathbf{A_k} = \begin{bmatrix} x_a - x_k & x_b - x_k & x_c - x_k \\ y_a - x_k & y_b - y_k & y_c - y_k \\ z_a - z_k & z_b - z_k & z_c - z_k \end{bmatrix}$$

The indices (a,b,c) are taken modulo n, except for the hexahedral cell which depend on k and vary for the showcased example cell as follows:

Node (k)	0	1	2	3	4	5	6	7
a	1	2	3	0	7	4	5	6
b	3	0	1	2	5	6	7	4
c	4	5	6	7	0	1	2	3

Table 2.4: Index numbering convention for Jacobian matrices



(a) Triangular element

(b) Tetrahedral element

(c) Quadrilateral element

(d) Hexahedral element

Figure 2.9: Numbering convention for construction of Jacobian[33]

The determinant of the Jacobian matrices, $\mathbf{A_k}$, is found to be directly proportional to the area of the cell in 2D and volume of the cell in 3D, therefore $\alpha_k = det(\mathbf{A_k})$. Furthermore, metric tensor matrices can be obtained using $\mathbf{A_k^T A_k}$. These tensor metrics are found to be symmetric with either three or six (depending on the element dimension being 2D or 3D) distinct elements represented by λ_{ij}^{k}, where i,j = 1,2...,d.

2.3.1. Relative size metric

Using the parameters obtained above, several mesh quality metrics can be defined. The relative size metric of the cell denotes the changes in the element size. It can be defined as:

$$f_{size} = min(\tau, 1/\tau) \tag{2.13}$$

where, $\tau = \sum_{k=0}^{n-1} \alpha_k/\alpha_k^0$, represents the ratio of the current cell size to the initial cell size. If the value of $f_{size} = 1$, then the element size has not changed from its initial size and if $f_{size} = 0$, then the element area becomes zero, making the cell degenerate.

2.3.2. Skew metric

Furthermore, the skew metric can be utilized to identify any distortion in the elements. It can be defined as follows for different elements:

$$\begin{aligned}
&\text{Triangular:} && f_{\text{skew}} = \frac{\sqrt{3}\alpha}{\lambda_{11}+\lambda_{22}-\lambda_{12}} \\
&\text{Tetrahedral:} && f_{\text{skew}} = \frac{3(\alpha\sqrt{2})^{2/3}}{\frac{3}{2}(\lambda_{11}+\lambda_{22}+\lambda_{33})-(\lambda_{12}+\lambda_{23}+\lambda_{13})} \\
&\text{Quadrilateral:} && f_{\text{skew}} = \frac{4}{\sum_{k=0}^{3}\sqrt{\lambda_{11}^{k}\lambda_{22}^{k}/\alpha_k}} \\
&\text{Hexahedral:} && f_{\text{skew}} = \frac{8}{\sum_{k=0}^{7}\sqrt{\lambda_{11}^{k}\lambda_{22}^{k}\lambda_{33}^{k}/\alpha_k}}
\end{aligned} \tag{2.14}$$

If $f_{skew} = 1$, then all the angles within the element are equal and if $f_{skew} = 0$, then the element becomes degenerate.

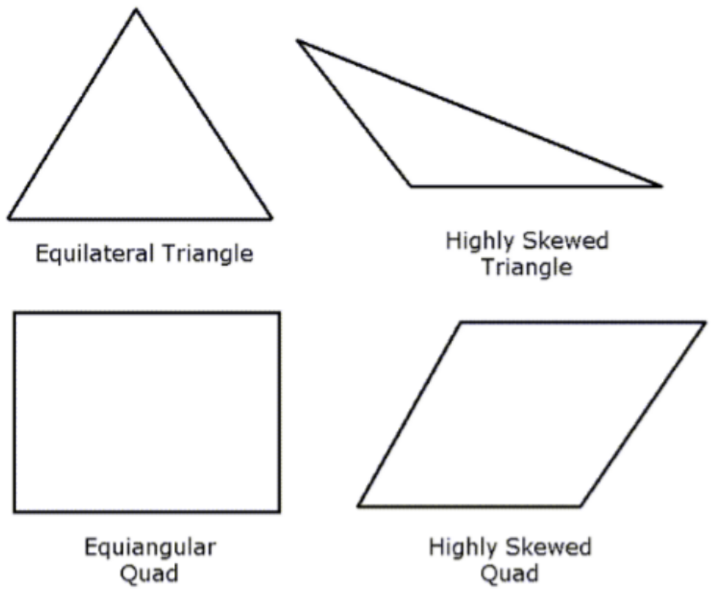


Figure 2.10: Skewed elements for triangular and quadrilateral elements[16]

2.3.3. Size-skew metric

In order to keep track of the change in cell size and skewness, a new parameter called size-skew[29] is defined as:

$$f_{ss} = \sqrt{f_{size}} \cdot f_{skew} \tag{2.15}$$

Variation in the volume of the cell has a relatively lower impact on the quality of the cell, hence f_{size} is introduced under a square root. This allows the cell skewness to attain a greater weightage. If $f_{ss} = 1$, then it is implied that there is no variation in cell size and all the angles in the cell are still equal. But, when $f_{ss} = 0$, then the element becomes degenerate. therefore, it is critical to ensure the value of f_{ss} remains as close as possible to one.

Furthermore, since the size-skew metric can track changes in size and skewness at the same time, it is more robust in nature and hence, will be utilized as the qualifying criterion for the implementation of the localized RBF corrections to the deformed mesh. The size-skew metric calculated for the deformed mesh is highly dependant on the original support radius of the interpolation function as well. The greater the support radius, the more optimal the quality of the mesh, but as discussed in subsection 2.1.1, there is a significant trade-off associated with the efficiency of the algorithm with the support radius of the interpolation function.

3 Mesh Quality Prediction

From the information gathered from the previous chapter on mesh quality metrics, it can be seen that in order to calculate the quality of a mesh it is necessary to have the nodal information of the mesh, i.e. the location of the nodes is required to construct the Jacobian matrices. But in cases where the meshes have to be deformed drastically due to large deformations, it is probable that the quality of the mesh post-deformation is going to be quite low. This might result in having to re-mesh the entire domain to improve the quality. Hence, the deformation step itself leads to a waste of computational time since the final quality is not known apriori.

However, the function that is being used to displace the nodes in the mesh is already known. It will be showcased in the current chapter that using the gradients of the deformation function, a reasonable estimate can be made with regards to the mesh quality in the next time-step without having to actually deform the mesh. An important thing to note is that all the deformations performed in the current chapter will be performed over a single time step in order to test the robustness of the algorithm in extreme conditions. Additionally, it will be assumed that the initial mesh is ”perfect”, and hence can be used as a qualifying standard for the quality at future time steps. The upcoming sections will provide a detailed description of the general approach, followed by validation on 1D and 2D meshes. Finally, a brief discussion is provided on how to expand the method to 3D as well.

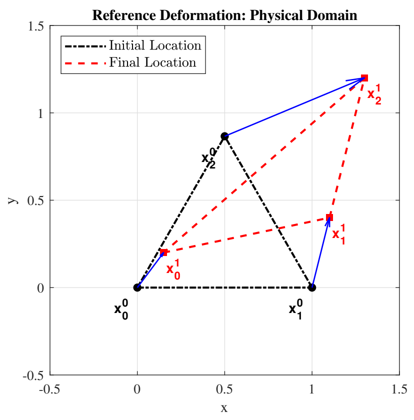
3.1. Approach

The current section describes the overall strategy to predict the algebraic mesh quality measures discussed in section 2.3 on a simplical triangular element for convenience's sake. The element comprises of three nodes as showcased in Figure 3.1a. The location of the nodes of the cell at the initial time-step, t^n is given by $\mathbf{x}_k^n$, $\mathrm{k} = 0, 1, 2$, where:

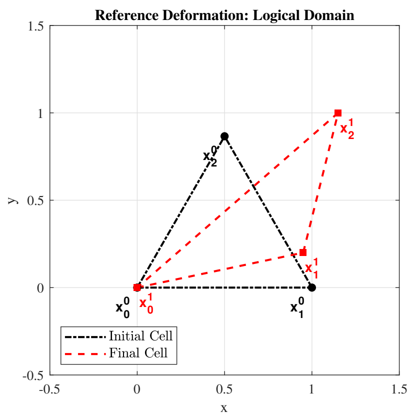
$$\mathbf{x}_k^n = \begin{bmatrix} x \\ y \end{bmatrix}_k^n \tag{3.1}$$

Now, if the element is within the support radius of an RBF control point, then it will undergo a deformation such that its new location and displacement at t^{n+1} can be represented as:

$$\mathbf{x}_k^{n+1} = \mathbf{x}_k^n + \mathbf{S}(\mathbf{x}_k^n) \tag{3.2}$$



(a) Deformation of a reference cell at the first time-step from the initial location in the physical domain.



(b) Deformation of the cell in the logical domain, where $\mathbf{x}_0$ lies on the origin

Figure 3.1: Reference deformation of an element in the mesh.

Consider an affine transformation of the physical elements onto a logical domain where the node, $\mathbf{x}_0$ always lies on the origin as shown in Figure 3.1b. The location of the nodes in this logical domain can be obtained by subtracting $\mathbf{x}_0$ from the original nodal locations.

$$\mathbf{x}_1^{n+1} - \mathbf{x}_0^{n+1} = (\mathbf{x}_1^n + \mathbf{S}(\mathbf{x}_1^n)) - (\mathbf{x}_0^n + \mathbf{S}(\mathbf{x}_0^n))$$

$$\implies \mathbf{x}_1^{n+1} - \mathbf{x}_0^{n+1} = \underbrace{(\mathbf{x}_1^n - \mathbf{x}_0^n)}_{part1} + \underbrace{(\mathbf{S}(\mathbf{x}_1^n) - \mathbf{S}(\mathbf{x}_0^n))}_{part2}$$

Part 2 of the above equation can be approximated using Taylor's series at an arbitrary location, ξ, in the interval $[\mathbf{x}_0^n, \mathbf{x}_1^n]$ as:

$$\mathbf{S}(\mathbf{x}_1^n) = \mathbf{S}(\mathbf{x}_\xi^0) + \mathbf{S}(\mathbf{x}_\xi^0)\nabla(\mathbf{x}_1^n - \mathbf{x}_\xi^n) + \dots$$

$$\mathbf{S}(\mathbf{x}_0^n) = \mathbf{S}(\mathbf{x}_\xi^0) + \mathbf{S}(\mathbf{x}_\xi^0)\nabla(\mathbf{x}_0^n - \mathbf{x}_\xi^n) + \dots$$

$$\implies (\mathbf{S}(\mathbf{x}_1^n) - \mathbf{S}(\mathbf{x}_0^n)) \approx \mathbf{S}(\mathbf{x}_\xi^0)\nabla(\mathbf{x}_1^n - \mathbf{x}_0^n)$$

Therefore,

$$\Delta\mathbf{x}_{10}^{n+1} \approx \Delta\mathbf{x}_{10}^n + (\mathbf{S}(\mathbf{x}_\xi^0)\nabla)\Delta\mathbf{x}_{10}, \qquad \xi \in [\mathbf{x}_0^n, \mathbf{x}_1^n] \tag{3.3}$$

$$\Delta\mathbf{x}_{20}^{n+1} \approx \Delta\mathbf{x}_{20}^n + (\mathbf{S}(\mathbf{x}_\xi^0)\nabla)\Delta\mathbf{x}_{20}, \qquad \xi \in [\mathbf{x}_0^n, \mathbf{x}_2^n] \tag{3.4}$$

The expressions obtained in Equation 3.3 and Equation 3.4 can be directly substituted in the expression to calculate the Jacobian of the cell at $\mathbf{x}_0$ given below:

$$\mathbf{A}_0 = \begin{bmatrix} \Delta\mathbf{x}_{10} & \Delta\mathbf{x}_{20} \end{bmatrix} \tag{3.5}$$

Thereby providing a first order approximation for the Jacobian at a future time-step based on the gradient of the deformation function in the previous time-step. The upcoming sections will now concentrate on utilizing this approach to approximate the algebraic mesh quality metrics discussed in chapter 2 for real meshes comprising of varying densities of elements in 1D and 2D.

3.2. 1D Mesh Quality Prediction

In the case of 1D deformation, the mesh quality metric that needs to be monitored is the size metric as discussed in section 2.3. In a 1D mesh, the nodes just have one translational degree of freedom. An example of a 1D grid is provided in Figure 3.2.

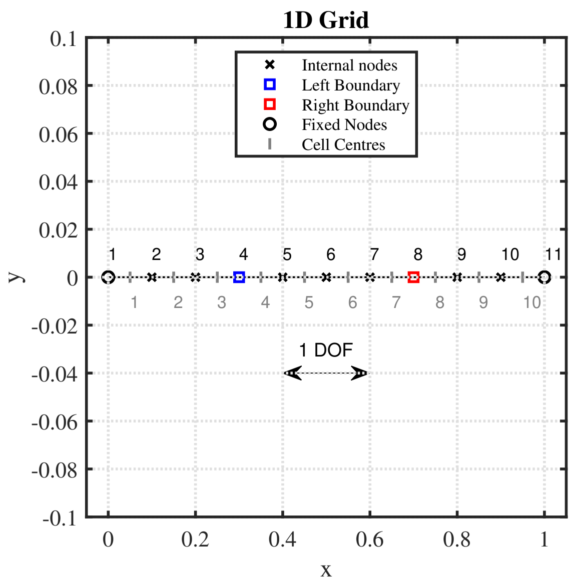


Figure 3.2: Example of a 1D Grid with 2 static boundaries and 2 moving boundaries. Nodes have one translational degree of freedom. Cell centres are depicted in grey.

Since it is assumed that the initial mesh quality is ”perfect” in the current method, the size metric of any cell at the next time step can be defined as the ratio of the cell size at that time step and the cell size at the initial time step:

$$f_s^{(1,0)} = \frac{\mathbf{x}_{i+1}^1 - \mathbf{x}_i^1}{\mathbf{x}_{i+1}^0 - \mathbf{x}_i^0} = \frac{\Delta\mathbf{x}_{i+1}^1}{\Delta\mathbf{x}_{i+1}^0} \tag{3.6}$$

Here, Δx basically represents the size of the cell created by two adjacent nodes. Therefore, the quality for each cell can be stored at the cell centre of that particular cell. Now applying the approach discussed in section 3.1 to the above equation for a sample deformation showcased in Figure 3.3, Equation 3.6 can be re-written as:

$$\begin{aligned} f_s^{(1,0)} &= \frac{(\mathbf{x}_{i+1}^0 + \mathbf{S}(\mathbf{x}_{i+1}^0)) - (\mathbf{x}_i^0 + \mathbf{S}(\mathbf{x}_i^0))}{\mathbf{x}_{i+1}^0 - \mathbf{x}_i^0} \\ &= \frac{\Delta\mathbf{x}_{i+1}^0 + (\mathbf{S}(\mathbf{x}_{i+1}^0) - \mathbf{S}(\mathbf{x}_i^0))}{\Delta\mathbf{x}_{i+1}^0} \end{aligned}$$

Since the function performing the mesh deformation is known to be real and continuous in the computational domain, Taylor's expansion can be utilized to re-write the above equation as discussed in section 3.1 as

$$f_s^{(1,0)} \approx \frac{\Delta\mathbf{x}_{i+1}^0 + \{[\cancel{\mathbf{S}(\mathbf{x}_{i+\xi}^0)} + \nabla\mathbf{S}(\mathbf{x}_{i+\xi}^0)(\mathbf{x}_{i+1} - \mathbf{x}_{i+\xi}) + ...] - [\cancel{\mathbf{S}(\mathbf{x}_{i+\xi})} + \nabla\mathbf{S}(\mathbf{x}_{i+\xi}^0)(\mathbf{x}_i - \mathbf{x}_{i+\xi}) + ...]\}}{\Delta\mathbf{x}_{i+1}^0}$$

Since $\Delta\mathbf{x}$ is generally very small, higher order terms can be neglected without significant effect to the accuracy of the approximation.

$$f_s^{(1,0)} = \frac{\Delta \mathbf{x}_{i+1}^0 + \nabla \mathbf{S}(\mathbf{x}_{i+\xi})^0 \Delta \mathbf{x}_{i+1}^0}{\Delta \mathbf{x}_{i+1}^0}$$
$$\implies f_s^{(1,0)} = 1 + \nabla \mathbf{S}(\mathbf{x}_{i+\xi})^0 \tag{3.7}$$

It has to be noted that calculating the gradient at $\xi = 0.5$, i.e. at the centre of the two nodes should lead to higher accuracy of the approximation due to the smaller magnitude of the numerical error. Furthermore, the cell centres are already known in the mesh and hence would simplify the evaluation process without having to define additional points. Therefore, it is decided that the gradients of the deformation function should be calculated at the cell centres in the current work. Furthermore, for convenience's sake here onwards the gradient calculated at the cell centre will just be mentioned as $\nabla \mathbf{S}$.

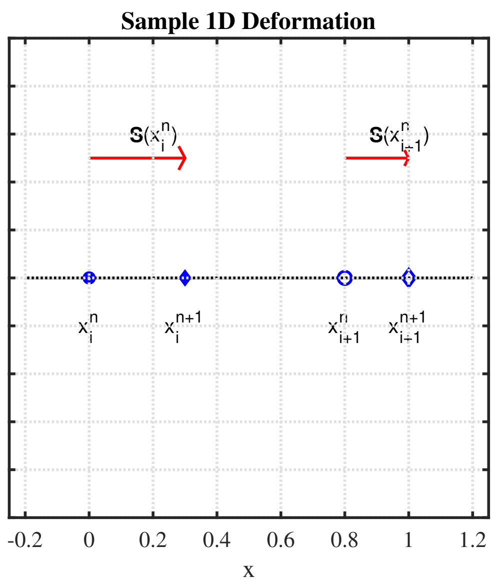


Figure 3.3: Sample 1D deformation using a deformation function, $\mathbf{S}$.

If the deformation process occurs in multiple time steps, then the quality at the time step (n+1) can be defined as:

$$f_s^{(n+1,0)} = \frac{\Delta \mathbf{x}^{n+1}}{\Delta \mathbf{x}^0} \qquad , n \in \mathbb{W} \tag{3.8}$$

But as discussed before, nodal information at the $(n+1)^{th}$ time-step is not available before the deformation is performed. Therefore, the above equation needs to be rewritten in a form that removes its dependence on the requirement of nodal data. Hence, the above equation is re-arranged as:

$$f_s^{(n+1,0)} = \frac{\Delta \mathbf{x}^{n+1}}{\Delta \mathbf{x}^n} \cdot \frac{\Delta \mathbf{x}^n}{\Delta \mathbf{x}^{n-1}} \dots \frac{\Delta \mathbf{x}^2}{\Delta \mathbf{x}^1} \cdot \frac{\Delta \mathbf{x}^1}{\Delta \mathbf{x}^0} = \prod_{i=0}^{n+1} f_s^{(i+1,i)} \tag{3.9}$$

Now applying Taylor's expansion to the above equation:

$$f_s^{(n+1,0)} = \frac{\Delta \mathbf{x}^n + \nabla \mathbf{S}^n \cdot \Delta \mathbf{x}^n}{\Delta \mathbf{x}^n} \cdot \frac{\Delta \mathbf{x}^{n-1} + \nabla \mathbf{S}^{n-1} \cdot \Delta \mathbf{x}^{n-1}}{\Delta \mathbf{x}^{n-1}} \dots \frac{\Delta \mathbf{x}^0 + \nabla \mathbf{S}^0 \cdot \Delta \mathbf{x}^0}{\Delta \mathbf{x}^0} \tag{3.10}$$

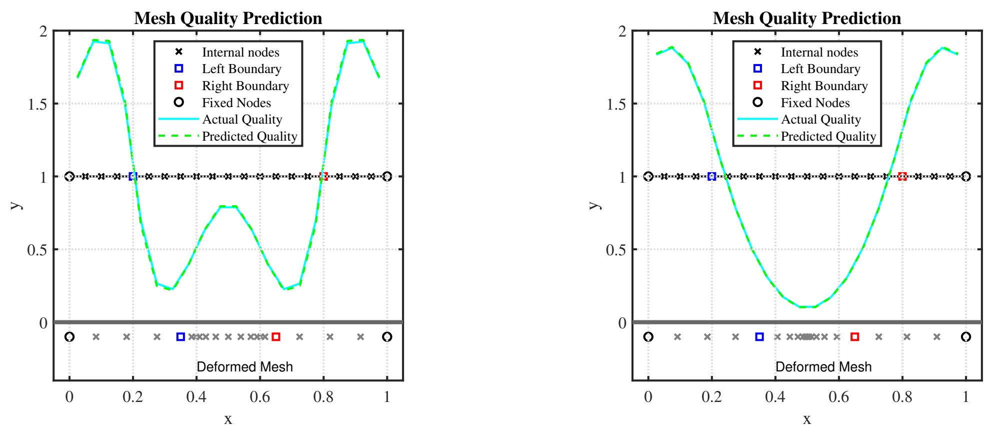
Only first-order terms of the Taylor expansion are considered in the above equation as $\Delta\mathbf{x}$ is very small.

$$\implies f_s^{(n+1,0)} = \frac{\cancel{\Delta\mathbf{x}^n}(1+\nabla\mathbf{S}^n)}{\cancel{\Delta\mathbf{x}^n}} \cdot \frac{\cancel{\Delta\mathbf{x}^{n-1}}(1+\nabla\mathbf{S}^{n-1})}{\cancel{\Delta\mathbf{x}^{n-1}}} \cdots \frac{\cancel{\Delta\mathbf{x}^0}(1+\nabla\mathbf{S}^0)}{\cancel{\Delta\mathbf{x}^0}} \tag{3.11}$$

$$\implies f_s^{(n+1,0)} = \prod_{i=0}^{n}(1+\nabla\mathbf{S}^i) \tag{3.12}$$

Therefore, from Equation 3.12,it can be seen that the mesh quality can be analytically derived based on just the gradient information of the mesh deformation function, $\mathbf{S}$. Hence, the dependence of the mesh quality on the nodal information at the future time step is completely eliminated. This ensures that the computationally expensive step of deforming the mesh does not need to be performed before realizing that the quality is low. Then having to again go back to the mesh configuration in the previous time step in order to implement the local enrichment algorithm. The only computational cost associated with the mesh quality prediction is the storage of the gradient information of the RBF at the cell centres. This implies that the size of the RBF gradient matrix will be $N \times N$, where N is the number of cells in the domain. Generally, low quality cells appear in localized clusters post the deformation step, so the theory of optimizing routines for RBF methods discussed in chapter 2 can be implemented to reduce the size of the matrix and improve efficiency. This will be discussed in the upcoming chapters.

It is to be noted that the analytically derived value is still an approximation of the actual mesh quality as the higher-order terms are ignored in the Taylor expansion. The comparison of the predicted mesh quality with the actual mesh quality for a reference grid with twenty cells is showcased in Figure 3.4. The two boundaries move towards each other by a magnitude of 1.5. It is seen that there is no major effect on the accuracy of the prediction since the higher-order terms are negligibly small. The deformation depicted is performed over a single time step. It can also be seen that the accuracy of the predicted quality is independent of the support radius, r. Furthermore, an interesting observation from Figure 3.4 is that when the cells are stretching then $f_s > 0$, whereas when there is contraction $f_s < 0$. The mesh quality crosses the perfect value, $f_s = 1$ at the boundary nodes as based on their movement, one side is always contracting while the other is expanding.



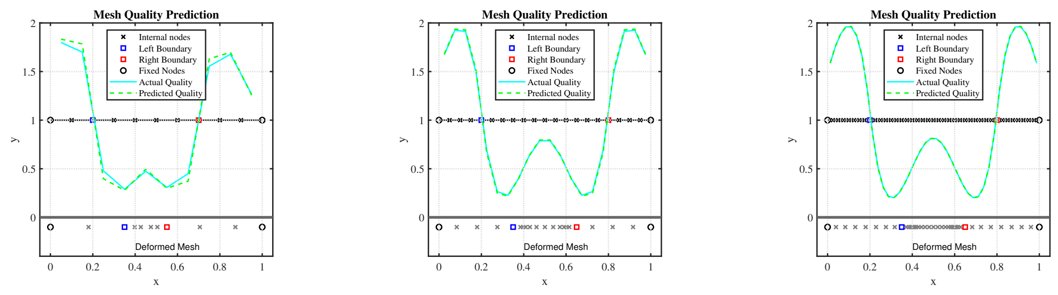
(a) Support radius of RBF, r = 0.4

(b) Support radius of RBF, r = 1

Figure 3.4: Comparison of the 1D Mesh quality prediction with the actual mesh quality at the $(n+1)^{th}$ time-step for a grid with 20 cells.

3.2.1. Grid Independence Study for Mesh Quality Prediction

In order to ensure that the mesh quality prediction is robust, different grids with varying cell densities (N) are tested as shown in Figure 3.5. It can still be seen that the quality prediction algorithm forecasts the quality with considerable accuracy.



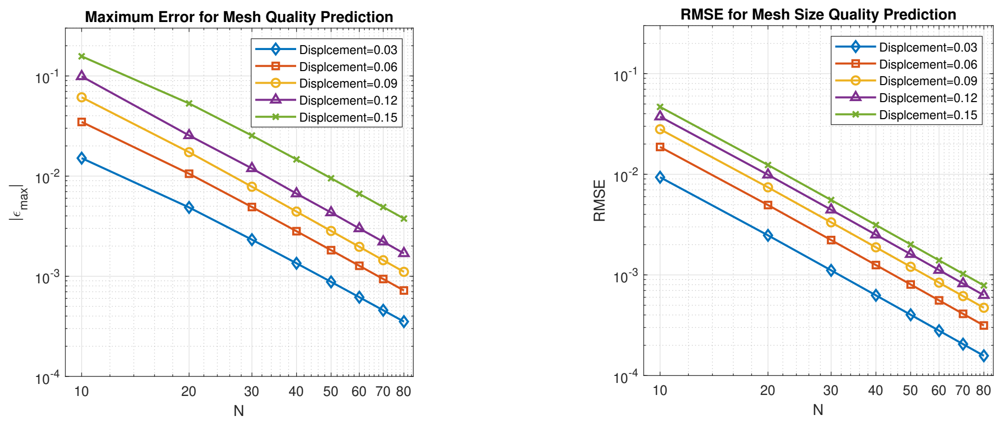
(a) N = 10 (b) N = 20 (c) N = 40

Figure 3.5: Comparison of the 1D Mesh quality prediction with the actual mesh quality at the $(n+1)^{th}$ time-step. Support radius of RBF, r = 0.4. Boundary displacement magnitude = 0.15 units.

In order to quantify the variation in the actual quality and the predicted quality, the maximum error in the prediction of the mesh quality is depicted in Figure 3.6a with variation in the number of cells. The relative error in the prediction is calculated as:

$$\epsilon = \left(\frac{f_s^{predicted} - f_s^{actual}}{f_s^{actual}} \right) \tag{3.13}$$

It can be seen that the magnitude of the maximum relative error in the domain goes down as the value of N increases due to a reduction in the discretization error. Furthermore, the RMSE in the mesh quality prediction is showcased in Figure 3.6b. From these plots, it can be clearly concluded that the error in the quality prediction is negligible and therefore, this methodology can in fact be used to find regions of poor quality in the domain in the future time step without having to actually deform the mesh regardless of the meshing resolution.



(a) Maximum error in cell quality prediction. (b) RMSE of the predicted mesh quality

Figure 3.6: Comparison of the 1D Mesh quality prediction with the actual mesh quality at the $(n+1)^{th}$ time-step. The deformation is performed in a single time step. Error decreases as the number of nodes are increased.

The effect of the deformation magnitude on the accuracy of the prediction is also showcased in Figure 3.6.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(f_s^{predicted} - f_s^{actual})^2}{N}} \tag{3.14}$$

It can be seen that the maximum error in the domain and the RMSE of the entire mesh increases as the displacement magnitude increases. This trend can again be attributed to the fact that the higher-order terms are being neglected in the Taylor expansion. With increasing displacements, the magnitude of these higher-order terms also increases. Thereby, resulting in a larger error. But with an increase in the number of cells, the value of $\Delta\mathbf{x}$ reduces thereby lowering the error. It can be noticed that the rate of convergence of the discretization error is quadratic in nature. Hence, it can be concluded that the mesh quality prediction algorithm is robust enough to accurately predict the quality irrespective of the deformation magnitude.

3.3. 2D Mesh Quality Prediction

In the case of 1D grids, the prediction is simple to perform due to the size of the cells being the only relevant parameter in play. But when it comes to 2D grids, the level of complexity increases as the nodes now have two translational DOFs. This means that the predictive algorithm needs to keep track of variations in size, skewness and aspect ratio, along with orientational information such as rotation and shape of the cell.

As discussed in section 2.3, the Jacobian matrix, $\mathbf{A}_k$ of a cell can be used in order to keep track of these qualities at the $(n+1)^{th}$. But the Jacobian is again dependent on the nodal information which is not available at the current time-step. Therefore, the methodology devised in section 3.1 eliminates the requirement of the locational information of the nodes in the Jacobian. This is possible by performing the Taylor expansion in 2D. Since the dimensions of the Jacobian is dependent on the shape of the cell, the formulation for the predicted quality will change with respect to varying element geometries. In the current thesis, two of the most common element shapes in 2D meshes are selected for analysis, i.e. triangular meshes which are mostly used in unstructured mesh cases and quadrilateral meshes, which are more common for structured mesh cases.

3.3.1. Triangular Unstructured Grids

For a triangular cell as shown in Figure 3.7b, the Jacobian, $\mathbf{A}_k$, has the dimensions of 2×2 and is given at the time-step, (n+1) as:

$$\mathbf{A}_k^{n+1} = \begin{bmatrix} x_{k+1} - x_k & x_{k+2} - x_k \\ y_{k+1} - y_k & y_{k+2} - y_k \end{bmatrix}^{n+1} \tag{3.15}$$

where, $k = 0, 1, 2$ and $\alpha = det(\mathbf{A}_k)$ represents twice the area of the triangular element. Now consider that $\mathbf{x}$ represents the x and y location of a particular node such that:

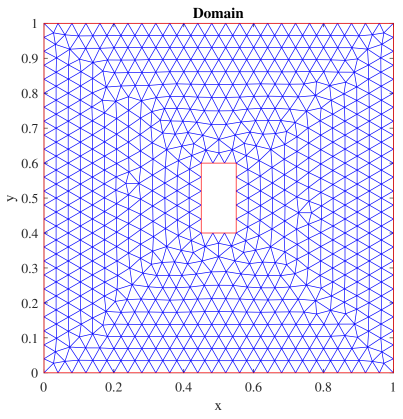
$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{3.16}$$

This implies that Equation 3.15 can be written as

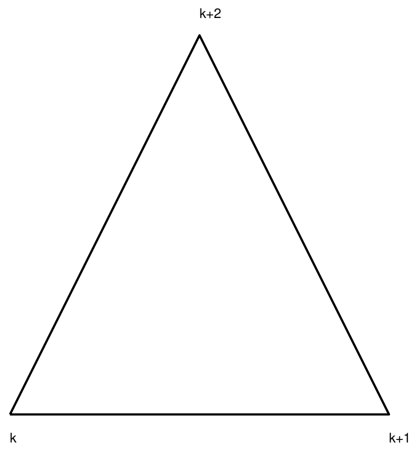
$$\mathbf{A}_k^{n+1} = \begin{bmatrix} \mathbf{x}_{k+1} - \mathbf{x}_k & \mathbf{x}_{k+2} - \mathbf{x}_k \end{bmatrix}^{n+1} \tag{3.17}$$

Consider the first term of the above equation. It can be re-written using Taylor expansion as:

$$\begin{aligned}(\mathbf{x}_{k+1} - \mathbf{x}_k)^{n+1} &= (\mathbf{x}_{k+1} - \mathbf{x}_k)^n + (\mathbf{S}(\mathbf{x}_{k+1}) - \mathbf{S}(\mathbf{x}_k)^n \\ \implies (\Delta\mathbf{x}_{k+1})^{n+1} &= (\Delta\mathbf{x}_{k+1})^n + \boldsymbol{\nabla}\mathbf{S}^n_{cc} \cdot (\Delta\mathbf{x}_{k+1})^n \end{aligned} \tag{3.18}$$



(a) Triangular mesh



(b) Triangular cell

Figure 3.7: Example of a triangular unstructured grid

Therefore, substituting the value of $\Delta\mathbf{x}_{k+1}$ in the above equation and doing the same for $\Delta\mathbf{x}_{k+2}$, the Jacobian, $\mathbf{A}_k$ can be written as:

$$\mathbf{A}_k^{n+1} = \begin{bmatrix} \Delta x_{k+1} & \Delta x_{k+2} \\ \Delta y_{k+1} & \Delta y_{k+2} \end{bmatrix}^n + \begin{bmatrix} \frac{\partial \mathbf{S}_x}{\partial x} & \frac{\partial \mathbf{S}_x}{\partial y} \\ \frac{\partial \mathbf{S}_y}{\partial x} & \frac{\partial \mathbf{S}_y}{\partial y} \end{bmatrix}^n \begin{bmatrix} \Delta x_{k+1} & \Delta x_{k+2} \\ \Delta y_{k+1} & \Delta y_{k+2} \end{bmatrix}^n \tag{3.19}$$

$$\implies \mathbf{A}_k^{n+1} = \left[\mathbb{I} + \begin{bmatrix} \frac{\partial \mathbf{S}_x}{\partial x} & \frac{\partial \mathbf{S}_x}{\partial y} \\ \frac{\partial \mathbf{S}_y}{\partial x} & \frac{\partial \mathbf{S}_y}{\partial y} \end{bmatrix}^n \right] \begin{bmatrix} \Delta x_{k+1} & \Delta x_{k+2} \\ \Delta y_{k+1} & \Delta y_{k+2} \end{bmatrix}^n \tag{3.20}$$

Here the identity matrix component represents the initial quality of the mesh. If we assume that the initial quality of the mesh is optimal or that is the target mesh quality that can be attained, then in order to save some computational cost, it would be possible to ignore the identity matrix. But, as discussed in section 2.3, the size-skew metric is the most ideal metric to characterize the quality of a cell in 2D. This metric is given as:

$$f_{ss} = f_s \cdot \frac{\sqrt{3}\alpha}{\boldsymbol{\lambda}_{11} + \boldsymbol{\lambda}_{22} - \boldsymbol{\lambda}_{12}} \tag{3.21}$$

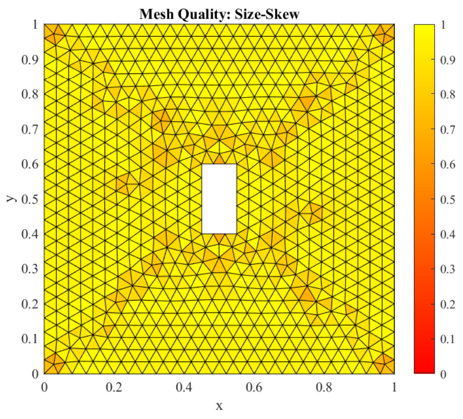
where, $f_s = min(\alpha/w, w/\alpha)$. Here, 'w' is the reference area of an ideal cell. The denominator in the above equation refers to the elements present in the matrix:

$$\boldsymbol{\lambda} = \mathbf{A}_k^T\mathbf{A}_k = \begin{bmatrix} \boldsymbol{\lambda}_{11} & \boldsymbol{\lambda}_{12} \\ \boldsymbol{\lambda}_{21} & \boldsymbol{\lambda}_{22} \end{bmatrix} \tag{3.22}$$

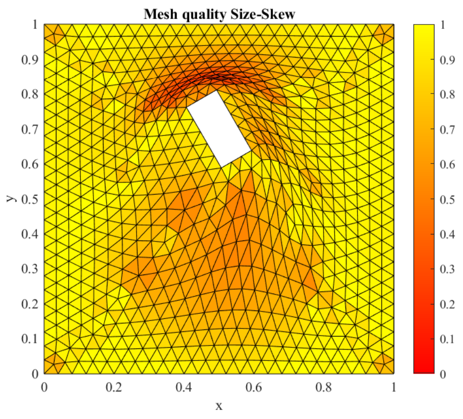
Therefore, it is noticed that if the initial quality of the mesh is neglected, then the magnitudes of the terms in $\boldsymbol{\lambda}$ is very small. Therefore, the quality prediction becomes undefined. Hence it is not possible to disregard the initial quality of the mesh from the formulation.

Furthermore, in the gradient matrix, the principle diagonal terms give information with regard to the size of the cell i.e. whether the cell is stretching or contracting in a particular dimension. While the off-diagonal terms provide information with regard to skewing. If the off-diagonal terms are equal, it implies that the cell is rotating which is not a major issue in most cases. Depending on the type of mesh deformation, each of these terms might have a varying level of importance which can be parameterized by the user. For example, stretching of a cell at the same rate in all directions in the case of a structured quadrilateral grid is not as important as contraction which can lead to inverted elements in future time steps.

In order to test the prediction algorithm, the mesh depicted in Figure 3.7a, is deformed using a regular RBF deformation. The structure is translated in the y-direction by 0.2 units and rotated by 30 degrees in the anti-clockwise direction. The actual mesh quality is plotted on the undeformed and deformed grid in Figure 3.8.



(a) Undeformed grid



(b) Deformed Grid

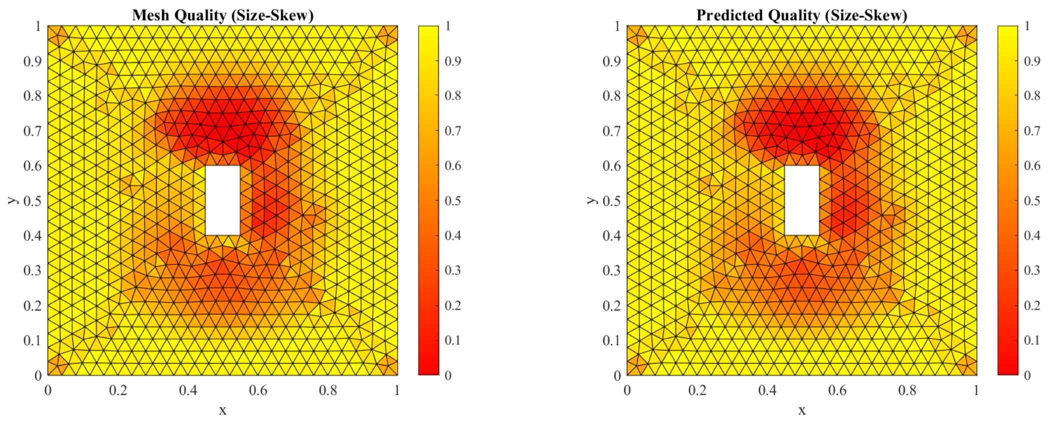
Figure 3.8: Actual 2D Mesh quality at the $(n+1)^{th}$ time-step.

Now in order to compare the performance of the 2D mesh quality prediction, the mesh quality at $(n+1)^{th}$ time-step from the predicted algorithm before performing the deformation, and the actual quality of the mesh post-deformation is compared in Figure 3.9. The qualities are plotted on the undeformed grid in order to clearly visualize which cells from the initial reference grid are deteriorating. It can be clearly seen that the qualities obtained from the prediction algorithm do not vary much from the actual quality.

Grid Independence Study for Mesh Quality Prediction

In order to test the robustness and accuracy of the algorithm with varying mesh sizes, a grid independence study is conducted. Figure A.1 showcases the qualitative comparison between the actual mesh quality and the predicted quality. It is seen that all the trends from the actual mesh quality are accurately captured by the predicted quality.

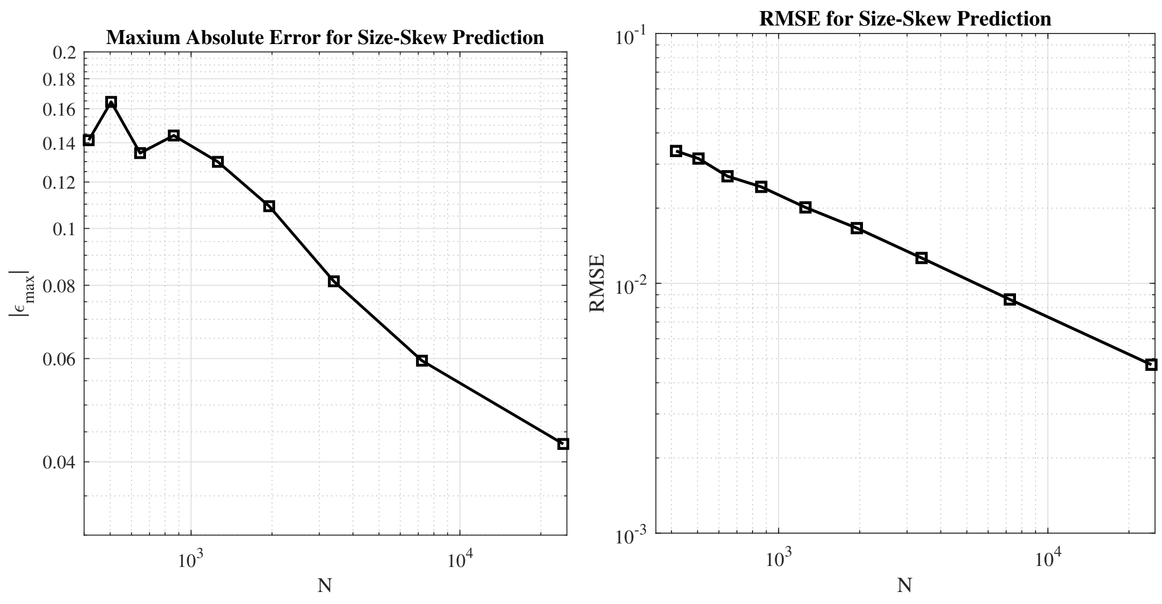
But to quantify the accuracy of the prediction, the maximum relative error and the RMSE of the predicted mesh quality are plotted in Figure 3.10 for a varying number of cells. The error is calculated for a 0.2 unit displacement in the positive y-direction and a rotational magnitude of 30°. It is noticed that the rate of convergence of the error is quadratic in nature. The error in the domain reduces as the number of cells increase since Δx reduces. Hence, the prediction becomes increasingly accurate with an increase in N. But, it has to be noted that for even coarse meshes, the magnitude of the maximum error and RMSE are well within acceptable limits.



(a) Actual Quality, N = 1386

(b) Predicted Quality, N = 1386

Figure 3.9: Comparison of the 2D Mesh quality prediction with the actual mesh quality at the $(n+1)^{th}$ time-step. Support radius of RBF, r = 0.4.



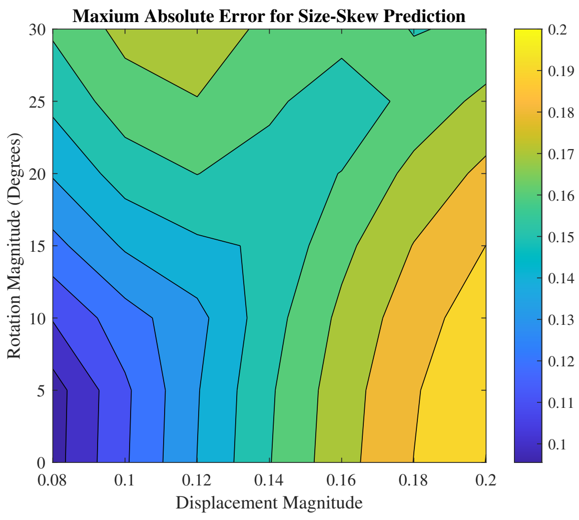
(a) Maximum absolute error in cell quality prediction

(b) Variation in RMSE with number of cells

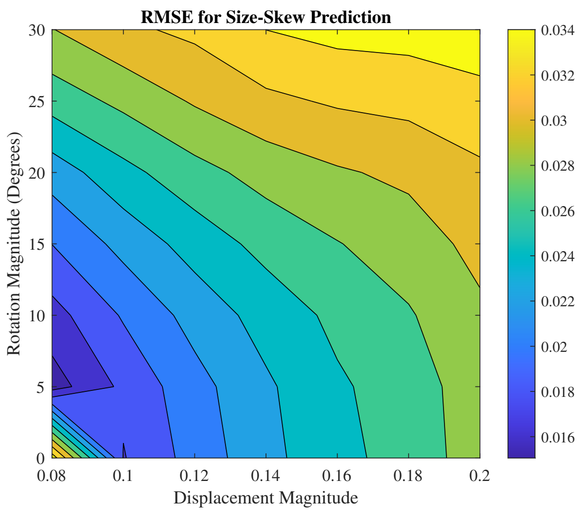
Figure 3.10: Variation in maximum relative error and RMSE with the number of cells. Error decreases as the number of cells are increased.

Effect of Deformation Magnitude on Mesh Quality Prediction

To ensure that the mesh quality prediction can accurately capture cases where large deformation magnitudes are prevalent, the methodology is tested on the coarsest grid with N = 416 over a range of displacements and rotation magnitudes in a single time-step. The minimum edge length in the mesh was 0.04 units. The error in the mesh quality prediction increases as the magnitude of deformation is increased as shown in Figure 3.11. But the errors obtained are very low in magnitude. Therefore, it can be safely concluded that the mesh prediction methodology works regardless of the deformation magnitude.



(a) Variation in maximum absolute error with deformation magnitude



(b) Variation in RMSE with various deformation magnitudes

Figure 3.11: Variation in maximum error and RMSE with displacement and rotational magnitudes at N = 416.

3.3.2. Quadrilateral Structured Grids

Unlike the triangular cells, the quadrilateral cells are non-simplical and hence their shape can not be determined by the use of a single Jacobian matrix[29]. The Jacobian matrix, $\mathbf{A}_k$ defined by at each node of the quadrilateral is given as:

$$\mathbf{A}_k^{n+1} = \begin{bmatrix} x_{k+1} - x_k & x_{k+3} - x_k \\ y_{k+1} - y_k & y_{k+3} - y_k \end{bmatrix}^{n+1} \tag{3.23}$$

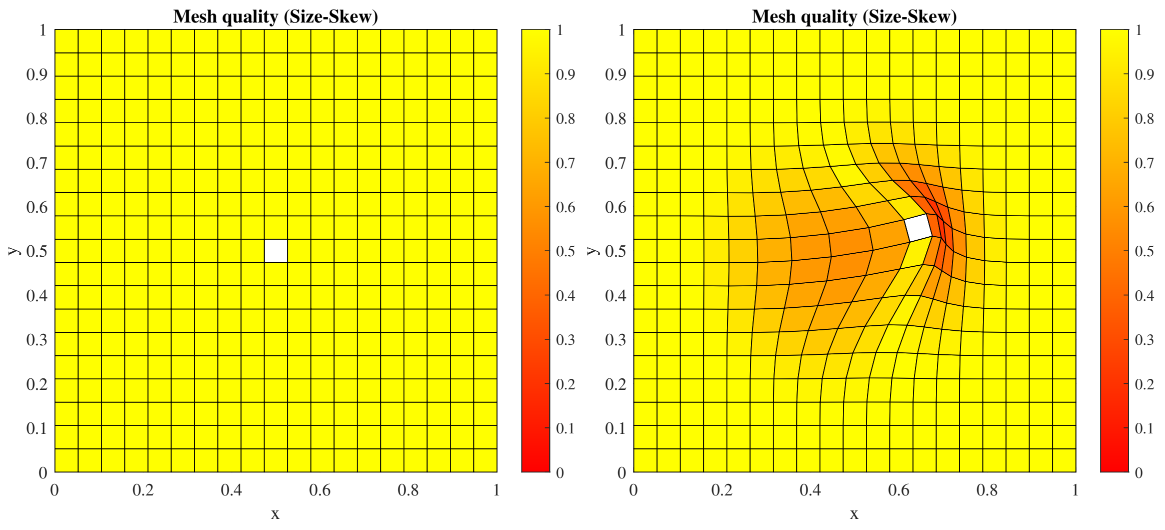
where, $k = 0, 1, 2, 3$ and the area of the quadrilateral is given by $(\alpha_k + \alpha_{k+2})/2$ with $\alpha_k = det(\mathbf{A}_k)$. This area of the quadrilateral is independent of the node. But the Jacobian itself is not independent of the node, four matrices are obtained by taking the formulation $\mathbf{A}_k^T\mathbf{A}_k$. But since the Jacobian of the quadrilateral cell also has the dimension of 2×2 like the triangular Jacobian with the only difference being the numbering of the nodes, Equation 3.20 can be used to formulate the Jacobian at each node.

Using this formulation the size-skew metric can be predicted at the $(n+1)^{th}$ time-step, as discussed in section 2.3 using the equation:

$$f_{ss} = f_s \cdot \frac{4}{\sum_{k=0}^{3} \frac{\left(\sqrt{\lambda_{11}^k \lambda_{22}^k}\right)}{\alpha_k}} \tag{3.24}$$

where, $f_s = min(\alpha/w, w/\alpha)$. Here, 'w' is the reference area of an ideal cell. The denominator in the above equation refers to the elements present in the matrix, $\mathbf{A}_k^T\mathbf{A}_k$.

The mesh quality prediction algorithm is tested on a structured 2D quadrilateral mesh as depicted in Figure 3.12 utilizing the regular RBF interpolation-based mesh deformation algorithm. A square-shaped structure is displaced by 0.15 units in the positive x-direction and a rotation of 15° is applied to it in the anti-clockwise direction with the rotation centre located outside the structure at (x,y) = (0.3,0.5). The actual mesh quality calculated post-deformation is showcased in Figure 3.12 on the undeformed and deformed grid respectively.

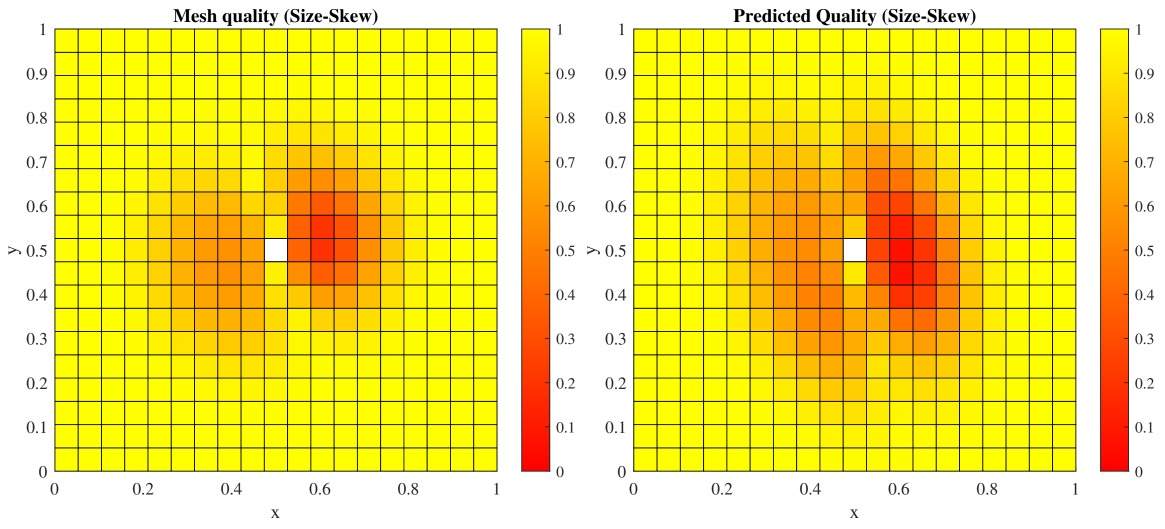


(a) Undeformed grid

(b) Deformed Grid

Figure 3.12: Actual 2D Mesh quality at the $(n+1)^{th}$ time-step.

The predicted mesh quality obtained using the formulation in Equation 3.20 and Equation 3.24 without deforming the mesh is compared with the actual mesh quality obtained post-deformation in Figure 3.13. Again the qualities are plotted on the undeformed grid in order to qualitatively visualize what the magnitudes of the mesh quality metric are in a clear fashion. It can be seen that the predicted mesh quality varies slightly from the actual mesh quality. The prediction algorithm showcases a larger effect on the surrounding of the block than is actually observed. Therefore, in order to see the reason behind this variance a grid convergence study is performed and the mesh qualities are compared quantitatively.



(a) Actual Quality, N = 361

(b) Predicted Quality, N = 361

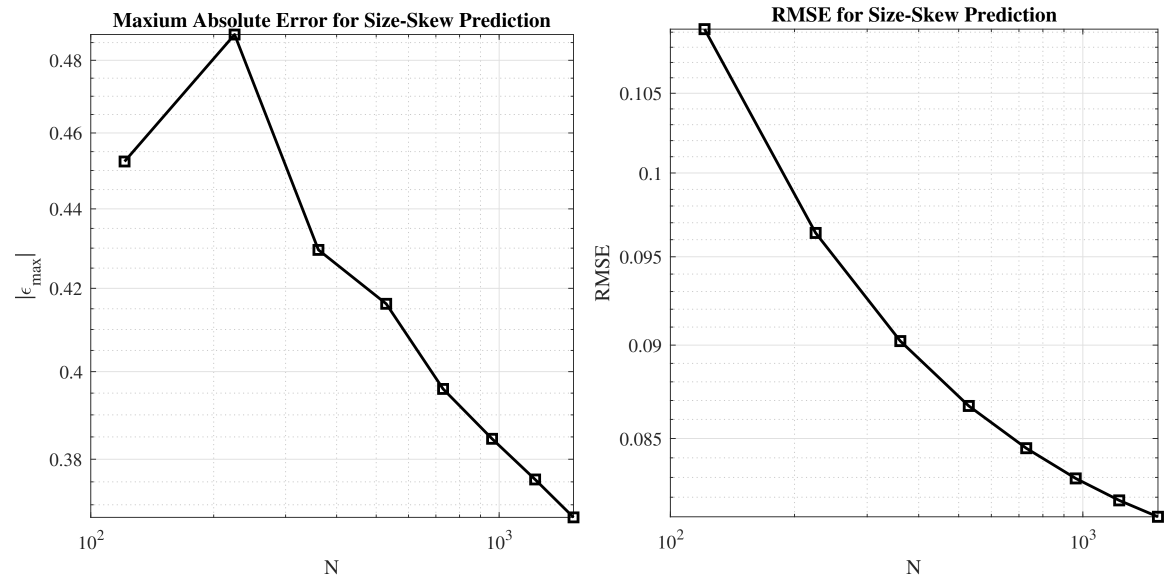
Figure 3.13: Comparison of the predicted 2D Mesh quadrilateral mesh quality with the actual mesh quality at the $(n+1)^{th}$ time-step. Support radius of RBF, r = 0.4.

Grid Independence Study

The qualitative plots for the grid convergence study are provided in Figure A.2. It is noticed that the quality prediction follows the trends of the actual quality relatively well. But slight discrepancies in the magnitudes of the prediction are noticed. In order to quantitatively analyze the difference between the predicted and the actual mesh quality, the maximum absolute error and the RMSE for the mesh quality prediction is depicted in Figure 3.14 for the various number of cells, N. It can be seen that the magnitude of both errors decreases as the number of cells in the domain increase. This is to be expected as the discretization error reduces. But unlike the triangular mesh, the rate of convergence of the maximum relative error and the RMSE are different.

Even though the formulation of the mesh quality prediction for both triangular and quadrilateral grids is similar, differences are noticeable in terms of the magnitudes of the errors in the prediction. The prediction algorithm functions in a far more accurate manner for the triangular grid than the quadrilateral grid. This can be attributed to the way in which the size-skew metric is calculated for both shapes. This metric for triangular cells, being simplical in nature, is dependent only on one Jacobian matrix, whereas, for the quadrilateral cell, the Jacobian at all four nodes has to be taken into account. This leads to the magnitude of the discretization error in the calculation of the size-skew metric being at least four times the magnitude of that observed in the triangular cell.

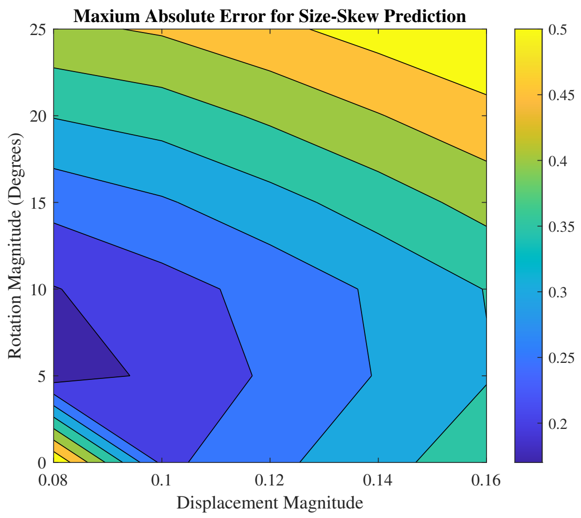
On the other hand, even if the magnitudes of errors are larger when compared to the triangular cell, the prediction is still accurate up to the first decimal point even for the large cases of rotational and translational deformations as shown in Figure 3.15 for the RMSE and accurate up to second decimal place for the maximum relative error. This level of accuracy is more than sufficient since the goal of the quality prediction algorithm is to estimate where bad quality cells might appear in order to introduce new RBF control points as discussed in chapter 4.



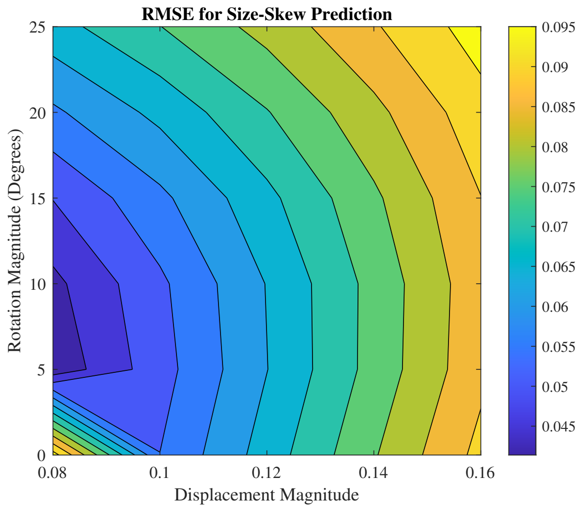
(a) Maximum relative error in cell quality prediction

(b) Variation in RMSE with number of cells

Figure 3.14: Variation in maximum error and RMSE of the cell mesh quality prediction with respect to the number of cells in the domain



(a) Variation in maximum relative error with various deformations



(b) Variation in RMSE with various magnitudes of deformations

Figure 3.15: Variation in maximum error and RMSE of the cell mesh quality prediction with respect to various translational and rotational deformation magnitudes for the coarsest grid, N = 121.

3.4. Discussion

3.4.1. 3D Mesh Quality Prediction

It is noticed that the step from 2D mesh quality prediction to 3D is not complicated. The only difference in the formulation is that the dimension of the Jacobian matrix increases by the order of one. The principal diagonal elements of the predicted Jacobian still represent how the volume of the cell changes in a particular direction while the off-diagonal terms represent the skewing of the cell along a particular axis.

It is to be noted that the differences observed between the triangular and quadrilateral mesh quality predictions are expected to be observed in the tetrahedral and hexahedral mesh quality prediction as the tetrahedral cell is simplical in nature and its properties can be defined by the use of a single Jacobian whereas the hexahedral cell is non-simplical,

Tetrahedral Unstructured Grids

The Jacobian of the tetrahedral cell can be described using the equation below as discussed in section 2.3.

$$\mathbf{A}_k^{n+1} = \begin{bmatrix} x_{k+1} - x_k & x_{k+2} - x_k & x_{k+3} - x_k \\ y_{k+1} - y_k & y_{k+2} - y_k & y_{k+3} - y_k \\ z_{k+1} - z_k & z_{k+2} - z_k & z_{k+3} - z_k \end{bmatrix}^{n+1} \tag{3.25}$$

Expanding the formulation obtained in Equation 3.20 to three dimensions as showcased below,

$$\Delta \mathbf{x}_{k+1}^1 = \Delta \mathbf{x}_{k+1}^0 + \nabla \mathbf{S}(\mathbf{x}_{k+1}^0) \Delta \mathbf{x}_{k+1}^0 \tag{3.26}$$

The predicted Jacobian matrix for time-step (n+1) is derived as:

$$\mathbf{A}_k^{n+1} = \begin{bmatrix} \Delta x_{k+1} & \Delta x_{k+2} & \Delta x_{k+3} \\ \Delta y_{k+1} & \Delta y_{k+2} & \Delta y_{k+3} \\ \Delta z_{k+1} & \Delta z_{k+2} & \Delta z_{k+3} \end{bmatrix}^n + \begin{bmatrix} \frac{\partial S_x}{\partial x} & \frac{\partial S_x}{\partial y} & \frac{\partial S_x}{\partial z} \\ \frac{\partial S_y}{\partial x} & \frac{\partial S_y}{\partial y} & \frac{\partial S_y}{\partial z} \\ \frac{\partial S_z}{\partial x} & \frac{\partial S_z}{\partial y} & \frac{\partial S_z}{\partial z} \end{bmatrix}^n \begin{bmatrix} \Delta x_{k+1} & \Delta x_{k+2} & \Delta x_{k+3} \\ \Delta y_{k+1} & \Delta y_{k+2} & \Delta y_{k+3} \\ \Delta z_{k+1} & \Delta z_{k+2} & \Delta z_{k+3} \end{bmatrix}^n \tag{3.27}$$

Furthermore, from section 2.3, the size-skew metric for the tetrahedral cell is defined as:

$$f_{ss} = f_s \cdot \frac{3(\alpha\sqrt{2})^{\frac{2}{3}}}{\frac{3}{2}(\boldsymbol{\lambda}_{11} + \boldsymbol{\lambda}_{22} + \boldsymbol{\lambda}_{33}) - (\boldsymbol{\lambda}_{12} + \boldsymbol{\lambda}_{23} + \boldsymbol{\lambda}_{13})} \tag{3.28}$$

where, $\boldsymbol{\lambda}_{ij}$ represent the elements of the matrix $\mathbf{A}^T\mathbf{A}$ and α represents the determinant of the Jacobian $\mathbf{A}_k$.

Hexahedral Structured Grids

As mentioned previously, the hexahedral cell is similar to the quadrilateral cell and is in fact non-simplical in nature. Therefore, the jacobian has to be calculated at all eight nodes of the hexahedron in order to predict the quality of the cell accurately. The Jacobian of the hexahedron is given as:

$$\mathbf{A}_k^{n+1} = \begin{bmatrix} x_{k+1} - x_k & x_{k+3} - x_k & x_{k+4} - x_k \\ y_{k+1} - y_k & y_{k+3} - y_k & y_{k+4} - y_k \\ z_{k+1} - z_k & z_{k+3} - z_k & z_{k+4} - z_k \end{bmatrix}^{n+1} \tag{3.29}$$

The formulation of the predicted Jacobian is in fact same as the formulation for the tetrahedral cell-derived in Equation 3.27. Therefore using the Jacobian, the symmetric matrix $\boldsymbol{\lambda}$ can be obtained as, $\boldsymbol{\lambda} = \mathbf{A}^T\mathbf{A}$. Therefore, for the eight Jacobian matrices, $\mathbf{A}_k$, eight $\boldsymbol{\lambda}^k$ matrices are obtained. Using which, the size-skew metric can be predicted using the formula:

$$f_{\mathrm{ss}} = f_s \cdot \frac{8}{\sum_{k=0}^{7} \left(\left(\sqrt{\boldsymbol{\lambda}_{11}^k \boldsymbol{\lambda}_{22}^k \boldsymbol{\lambda}_{33}^k}\right) / (\alpha_k)\right)^{2/3}} \tag{3.30}$$

Therefore it can be seen that the algebraic mesh quality metrics discussed in chapter 2 can be easily predicted without having to deform the mesh. The algorithm is validated in the case of 1D and 2D meshes, while the theoretical foundation for 3D meshes is laid out. The 3D case can not be validated due to time constraints and will be left as a recommendation for future work.

4 Local Enrichment Correction Algorithm: 1D Model

The methodology of the local enrichment function-based mesh deformation improvement method will be first expounded in a concise manner for a 1D setup in the current chapter, before being expanded into multiple dimensions in future chapters. The current chapter initially discusses the framework and the motivation behind the implementation of the local enrichment correction methodology. Following this, technical details with regard to the methodology will be presented in terms of where and what kind of corrections need to be performed in order to improve the minimum quality of the mesh. The results presented in the current chapter will showcase the variation between the mesh quality obtained using just the standard RBF method and the localized enrichment correction algorithm such that a clear distinction can be made in terms of the effectiveness of the developed algorithm. An important point to note is that all the results showcased in the current chapter are for deformations which have been performed over a single step in order to

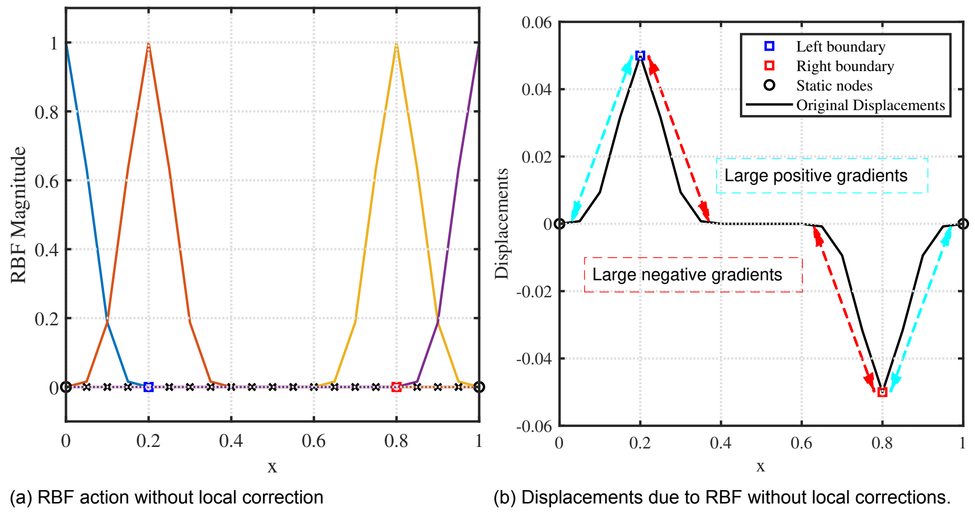
4.1. Approach

As discussed in chapter 2, the RBF interpolation works on the basis of interpolating the prescribed displacements at the boundary nodes to the internal nodes of the mesh within a certain radius. But the standard RBF interpolation method is not robust enough to handle large displacements to the structure within the domain. Therefore a mesh quality prediction algorithm was introduced in chapter 3, such that the regions with poor quality are preempted before performing the intensive deformation step. After identifying such problematic regions in the domain, it needs to be ensured that the mesh does not contain elements with quality below a certain user-defined threshold post the deformation step. Therefore, it would seem to be viable to introduce additional control points in these regions upon which additional enrichment functions can be imposed in order to constrain the deformation of the internal nodes, hence improving the minimal quality of the mesh. These additional control points would be fictitious in nature and would be utilized only as a computational constraint in the deformation algorithm. In simpler terms, this means that the additional control points would have no physical presence in the domain, unlike the original boundary nodes, which drive the movement of the internal nodes due to the presence of the RBF on them. The goal of applying these local enrichment corrections is to smoothen the overall deformation process, in order to avoid sharp gradients with respect to the deformation magnitudes between the nodes in the domain.

But then two important questions arise in order to smoothen these deformation gradients:

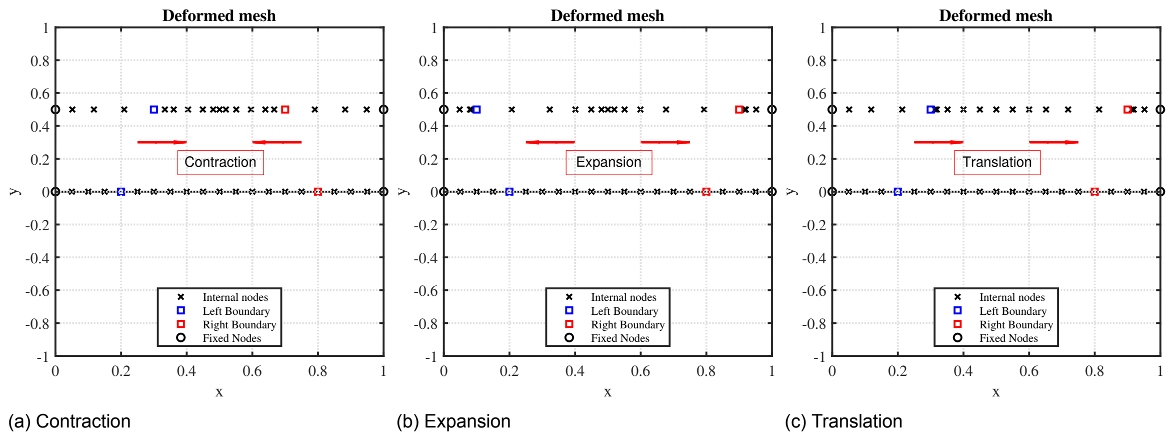
1. Where should the control points be introduced?
2. What kind of function should be acting from these control points?

In order to answer the two questions posed above, first a valid test setup needs to be defined on which the corrections can be performed in order to validate the choices for the answers. An example of a domain with deformations involving sharp gradients is showcased in Figure 4.1. The small RBF support radius of 0.2 is selected explicitly to ensure that the deformation is not very smooth, thereby leading to low magnitudes of the minimum quality in order to test the robustness of the algorithm being developed. Therefore, an academic test case for the 1D setup of the localized RBF correction algorithm is developed comprising twenty one nodes out of which two are static boundary nodes and two are moving boundary nodes as depicted in Figure 4.2. The setup can be tested for three general cases representing the phenomena of contraction, expansion and translation of the boundaries.



(a) RBF action without local correction (b) Displacements due to RBF without local corrections.

Figure 4.1: Qualitative depiction of sharp deformation gradients due to compact support RBF's without local correction. RBF support radius, R = 0.2.



(a) Contraction (b) Expansion (c) Translation

Figure 4.2: Test case setup shown for three possible motions of boundaries in 1D.

These cases represent the most fundamental movement of nodes that lead to a reduction in mesh quality in 1D. But it can be noticed that by studying one of the three cases, the other cases are automatically covered because if there is contraction or expansion occurring in some region of the domain, then some other part in the domain has to automatically be expanding or contracting as seen from Figure 4.2. Therefore, the contracting setup is chosen to be studied in depth over the other two cases in the current chapter as that particular case has the highest likelihood of producing inverted elements due to the nodes crossing over each other. This is especially apparent in cases where large deformations are involved. An additional point to note is that in the current chapter, whenever the mesh quality is mentioned, it refers to the size metric since parameters such as skewing, rotation, aspect ratio and so on require the presence of more than just 1D. A size metric of one implies an ideal cell while more or less than that would imply expansion and contraction respectively.

4.2. Local RBF Control Point Location

The location at which the additional control point is supposed to be added and the shape of the enrichment function is contingent on whether the correction being performed is local in nature or global. For a global function, since the current test case exhibits symmetric motion of the boundary nodes with respect to the centre of the domain, it would make sense to add the control point at the centre of the domain.

Furthermore, it has been showcased in the previous chapter that the mesh quality can be accurately predicted at the $(n+1)^{th}$ time-step without having to deform the mesh to the new location. Since the aim of the methodology is to improve mesh quality using enrichment functions in regions where poor quality is predicted, a logical guess would be to add the control points directly in such regions.

Additionally, for the 1D mesh quality prediction test case, it was noticed that the mesh quality always crosses the value of one (most ideal quality) at the location of the boundary point. The reasoning behind this is that based on the boundary movement the nodes on one side of the boundary are always contracting, whereas, on the other side they will be expanding as can be seen from Figure 3.4. Now from the two cases, it is known that a contracting setup is more critical for mesh quality as the plausibility of cell inversions occurring is higher. Therefore, it would also seem to be a plausible idea to add the control point at the location of the closest boundary point to the poor-quality region. The reasoning is that the enrichment function can then specifically target the contraction region on one side and the expansion region on the other side thereby improving the overall quality of the mesh post-deformation.

Both hypotheses are tested in the current chapter. First by adding local enrichment control points only at the boundary nodes and then adding them only at the locations where the worst quality is observed.

4.3. Nature of Correction

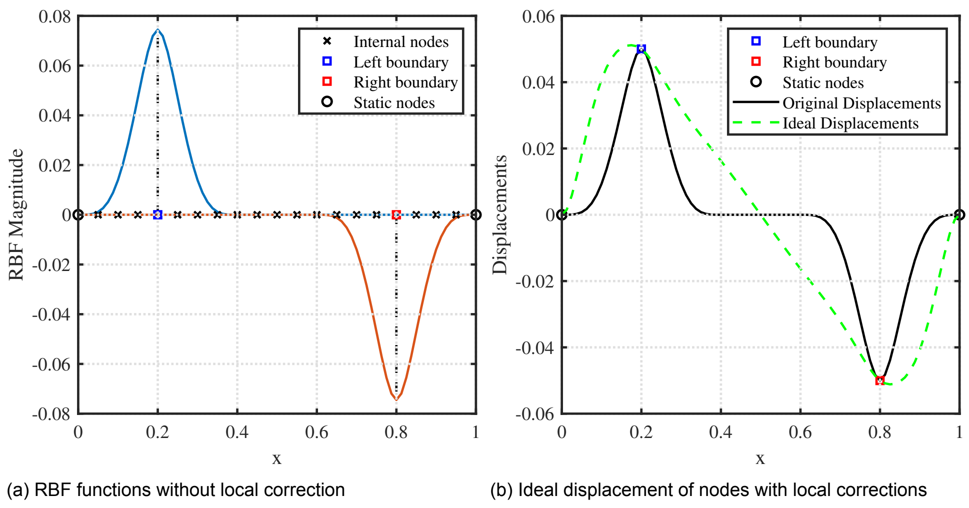
Now that the possible locations for the addition of the control points are known, the nature of the enrichment function needed to improve the mesh quality can be looked upon in the current section. The enrichment function being used can be either local in nature or global in nature. The choice of the nature of correction is highly dependent on the initial mesh parameters. Global mesh corrections would make sense when the size of the grid is relatively small. If not, the interpolation matrix would become very dense and would lead to high computational costs, thereby making the method unviable. Furthermore, in terms of a 1D setup, the most ideal deformation would involve the internal nodes having a linear spread between the moving boundary nodes as shown in Figure 4.3. It can be seen from Figure 4.3b the ideal displacements have a constant gradient between the two boundaries, thereby avoiding the sharp gradients observed in the original displacements. This would ensure that the minimum quality in the domain is higher, even if the average quality goes down relatively.

In the previous section, it was hypothesized that the addition of the enrichment function at either the boundary nodes or the worst quality location could leave to an improvement in the overall mesh quality. From Figure 4.3a, it can be clearly deduced that the gradient of the deformation is equal to zero at the location of the boundary node. Whereas, the highest gradient of deformation will be observed at the worst-quality locations. Therefore, it can be hypothesized that if the ideal gradient of the linear interpolation observed is imposed as a constraint at the location of the control point i.e. at the boundary node or the worst quality location, then it would lead to a more smooth deformation of the internal nodes.

For a global function, this is relatively simple as the displacement of the moving boundary nodes and their initial locations are already known. The gradient, '$\mathbf{J}$', on which the internal nodes in between have to lie upon can be calculated as:

$$\mathbf{J} = \frac{D_{b_1} - D_{b_2}}{x_{b_1} - x_{b_2}} \tag{4.1}$$

where D_b represents the displacement of the boundary nodes and x_b represents the initial location of the boundary nodes. It can be clearly seen from Figure 4.3b that the ideal displacements have a negative gradient. Therefore, the value of 'q' will also be less than zero at the centre of the domain where the control point is added to control the deformations.



(a) RBF functions without local correction (b) Ideal displacement of nodes with local corrections

Figure 4.3: Linear variation in nodal displacements

This linearity in the displacement of the internal nodes in critical locations where low mesh quality is predicted can be achieved in several ways. Even in the case of local corrections, it is still possible to stitch up multiple enrichment functions in the domain which would allow this almost linear behaviour in the displacement of the internal nodes. But in order to save computational cost, it would be ideal to construct the enrichment function such that it would also be able to have a very localized effect if necessary. This would allow for a sparse system matrix to be formulated. Since Wendland's C^2 function[46] performs well with compact support and is already being utilized as the RBFs, it would be ideal to construct the enrichment function using a similar formulation. This would lead to the added advantage that the data already available in the deformation process can be utilized without having to define new functions.

4.3.1. Global Correction

In the case of global correction in 1D, only one control point needs to be added which should have the desired effect on the entire domain. Additionally, in the previous chapter on mesh quality prediction, it was noticed that the magnitude of the minimum mesh quality of the grid is highly dependent on the support radius of the RBF at the boundary nodes. Larger support radii lead to a smoother deformation thereby increasing the overall minimal quality when compared to compact support radii. In order to find the ideal function, multiple variations of the C^2 functions are tested with a global support radius. The idea is to find an efficient implementation in the global sense first and then dissect the effect of that function using multiple compact support enrichment functions.

Looking at the test case, it can be seen that the nodes between the static and moving boundaries are expanding, while the nodes between the moving boundaries are contracting. The introduced enrichment function should essentially constrain both these effects to a considerable extent to preserve quality. This can be done by imposing an enrichment function which prescribes a certain gradient to the deformation at the location of correction. To obtain a linear trend in the displacements as compared to the deformation in Figure 4.2a, the imposed function should have a positive gradient between the static and boundary nodes and a negative gradient between the moving boundaries as seen from the ideal deformation in Figure 4.3b. The global correction is introduced at a control point in the centre of the domain as the movement of the boundary nodes is symmetric in nature. Three variants of the C^2 function that fulfil the specified requirements are provided as follows:

1. $H_1(\mathbf{x}, R_l) = (\mathbf{x} - \mathbf{x}_c)\phi(r, R_l)$
2. $H_2(\mathbf{x}, R_l) = \frac{\partial \phi(r,R_l)}{\partial x}$
3. $H_3(\mathbf{x}, R_l) = \frac{\partial \phi(r,R_l)}{\partial x}\phi(r, R_l)$

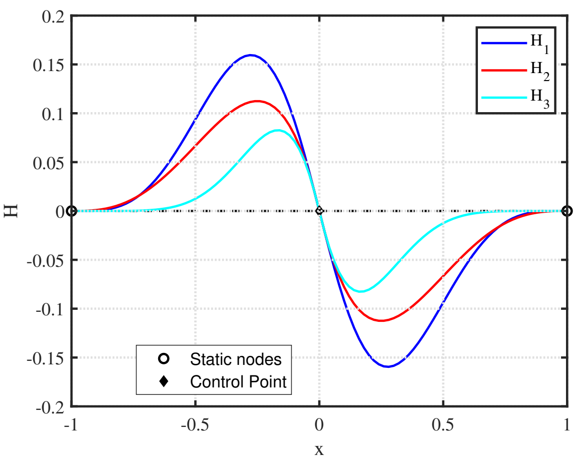


Figure 4.4: Depiction of enrichment function shapes in 1D, $R_l = 1$.

where, $\phi(r, R_l)$ represents the original C^2 function, r is the distance between the RBF control points and 'R_l' is the support radius of the enrichment function. All three functions introduced are dependent on this support radius, thereby making them compact and hence making the correction efficient. Also, x_c represents the set of fictitious enrichment control points and the original boundary nodes. $(\boldsymbol{x} - \boldsymbol{x}_c)$ is a linear function that can introduce the linearity.

Therefore, a coupled architecture can be formulated incorporating the boundary conditions from the actual mesh deformation and the imposed gradient condition within the same system (as no additional polynomial term is being utilized). Therefore the system that needs to be solved can be represented as:

$$\begin{bmatrix}\mathbf{d}_b \\ \mathbf{J}\end{bmatrix} = \begin{bmatrix}\phi_{bc} & H_{bc} \\ \nabla\phi_{bc} & \nabla H_{cc}\end{bmatrix}\begin{bmatrix}\boldsymbol{\gamma}_\mathbf{x} \\ \boldsymbol{\beta}_\mathbf{x}\end{bmatrix} \tag{4.2}$$

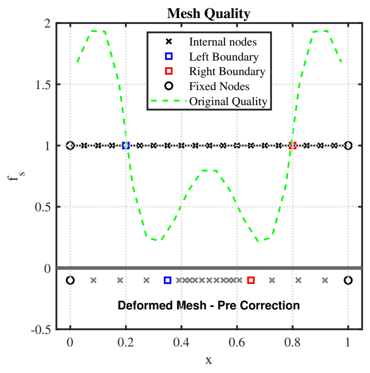
where $'\mathbf{J}'$ represents the imposed condition on the gradient of the deformation at the location of the enrichment control point as given in Equation 4.1. $\boldsymbol{\gamma}_\mathbf{x}$ and $\boldsymbol{\beta}_\mathbf{x}$ are the interpolation coefficients. The above conditions when applied to the chosen test case depicted in Figure 4.2a yields:

$$\begin{bmatrix}0 \\ \mathbf{d_b} \\ -\mathbf{d_b} \\ 0 \\ \mathbf{J}\end{bmatrix} = \begin{bmatrix}\phi_{11} & \phi_{12} & \phi_{13} & \phi_{14} & H_{1c} \\ \phi_{21} & \phi_{22} & \phi_{23} & \phi_{24} & H_{2c} \\ \phi_{31} & \phi_{32} & \phi_{33} & \phi_{34} & H_{3c} \\ \phi_{41} & \phi_{42} & \phi_{43} & \phi_{44} & H_{4c} \\ \boldsymbol{\nabla}_x\phi_{1c} & \boldsymbol{\nabla}_x\phi_{2c} & \boldsymbol{\nabla}_x\phi_{3c} & \boldsymbol{\nabla}_x\phi_{4c} & \boldsymbol{\nabla}_x H_{cc}\end{bmatrix}\begin{bmatrix}\boldsymbol{\gamma}_\mathbf{1} \\ \boldsymbol{\gamma}_\mathbf{2} \\ \boldsymbol{\gamma}_\mathbf{3} \\ \boldsymbol{\gamma}_\mathbf{4} \\ \boldsymbol{\beta}_\mathbf{x}\end{bmatrix} \tag{4.3}$$

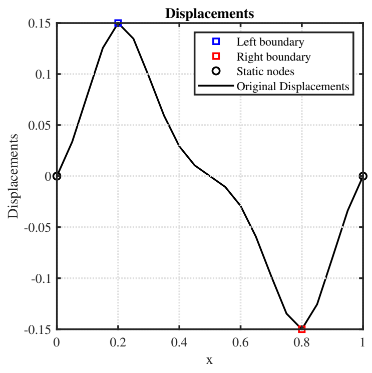
Once the interpolation coefficients γ_x and β_x are calculated, the displacements of the internal nodes can then be calculated using the formulation:

$$\left[\mathbf{d}_{in}\right] = \left[\phi_{in,b} \quad H_{in,c}\right]\begin{bmatrix}\boldsymbol{\gamma}_\mathbf{x} \\ \boldsymbol{\beta}_\mathbf{x}\end{bmatrix} \tag{4.4}$$

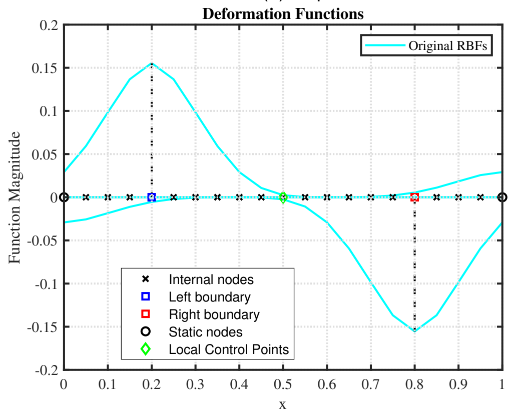
Now that the system matrix has been defined, the effect of the enrichment function-based correction on the final mesh quality will be compared to the mesh quality using the standard RBF method on the domain defined in Figure 4.2a. The standard RBF has a radius of 0.4, while the enrichment function has a support radius of 1. The boundary nodes move towards each other by a magnitude of 0.15. The deformation using the standard RBF method is depicted in Figure 4.5 for reference.



(a) Mesh Quality



(b) Displacement of nodes



(c) Effect of RBFs

Figure 4.5: Deformation obtained using standard RBF model, $R = 0.4$.

Three plots will be showcased for each enrichment function: the first depicts the mesh quality using both methods on the undeformed grid along with a depiction of the deformed grid, the second will showcase the displacement magnitudes of the nodes within the domain and the final plot will showcase the shapes of the RBFs and the enrichment functions to provide clarity on the effect of every function acting on the domain.

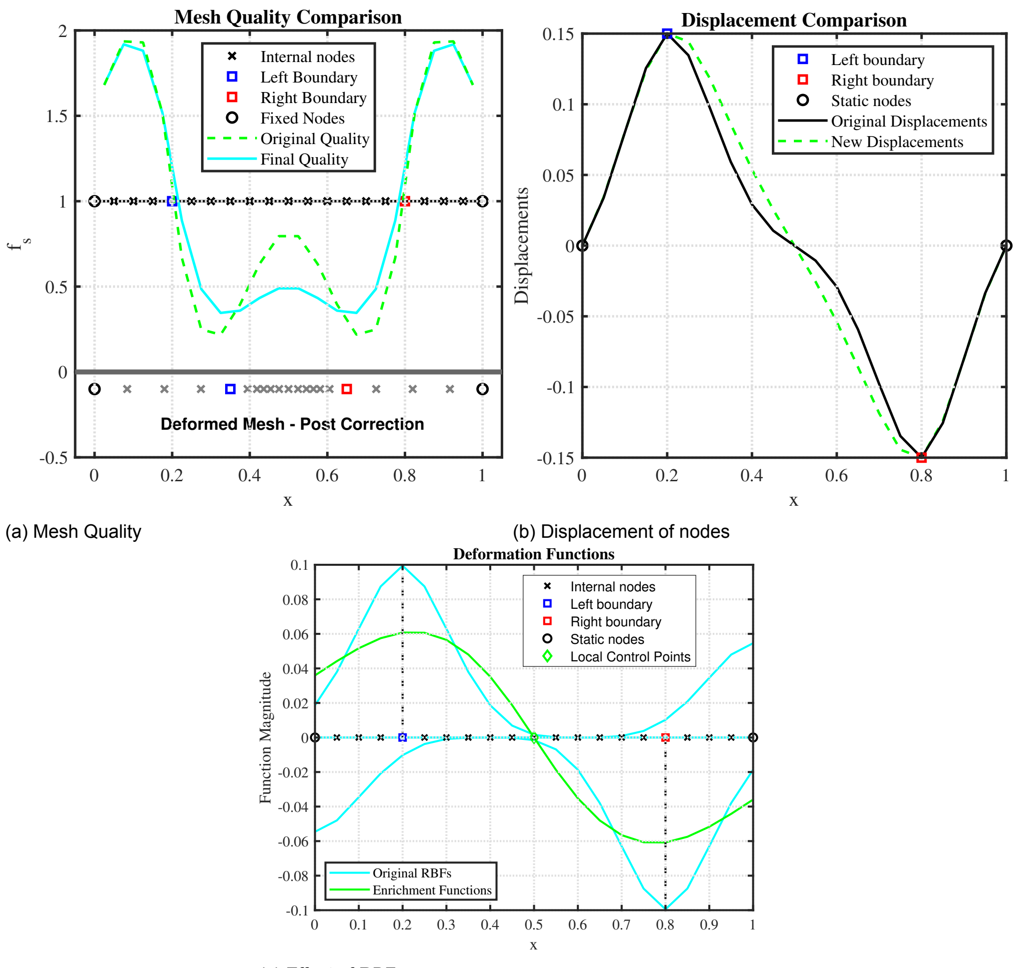
1. $H(\mathbf{x}, R_l) = (\mathbf{x} - \mathbf{x}_c)\phi(r, R_l)$

The enrichment function defined can be expanded as:

$$H_1(\mathbf{x}, R_l) = (\mathbf{x} - \mathbf{x}_c)\left(1 - \frac{r}{R_l}\right)_+^4 \left(4\frac{r}{R_l} + 1\right) \tag{4.5}$$

The effect of the enrichment function $H_1(\mathbf{x}, R_l)$ when applied at the centre of the domain is presented in Figure 4.6. Setting the value of '$\mathbf{J}$' in Equation 4.3 to the ideal slope value actually provides good results i.e.

$$\mathbf{J} = \frac{D_{b_1} - D_{b_2}}{x_{b_1} - x_{b_2}} = \frac{0.15 - (-0.15)}{0.2 - 0.8} = -0.5$$



(a) Mesh Quality (b) Displacement of nodes

(c) Effect of RBFs

Figure 4.6: Effect of global enrichment function, $H_1(\mathbf{x}, R_l)$ on mesh deformation, $R = 0.4, R_l = 1$

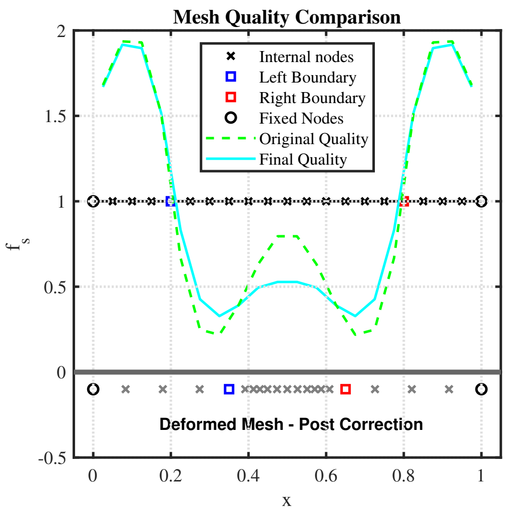
It can be clearly seen from Figure 4.6a that there is a considerable improvement in the minimum quality when compared to the standard RBF method. The average quality of the cells does go down, but its effect is not so significant. The displacements of the internal nodes do actually fall into a linear pattern as intended as seen from Figure 4.6b. Furthermore, it is seen from Figure 4.6c that the gradient of the deformation functions is indeed constrained to the imposed value while ensuring that the boundary conditions are also satisfied. Hence, this function can be utilized for improvements in the deformation quality in the future.

2. $H_2(\mathbf{x}, R_l) = \frac{\partial \phi(r, R_l)}{\partial x}$

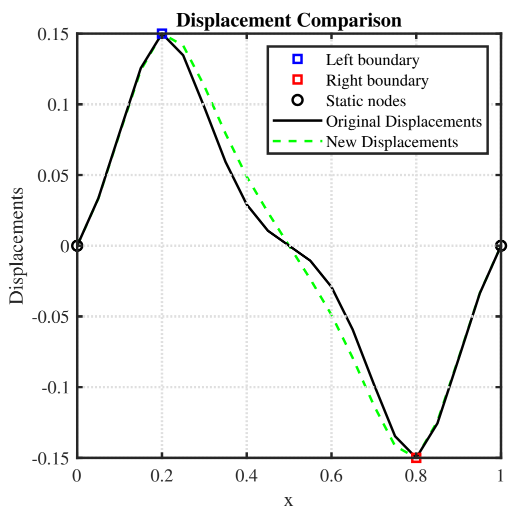
The enrichment function defined can be expanded as:

$$H_2(\mathbf{x}, R_l) = \frac{-20(\mathbf{x} - \mathbf{x}_c)(R_l - r)^3}{R_l^5} \tag{4.6}$$

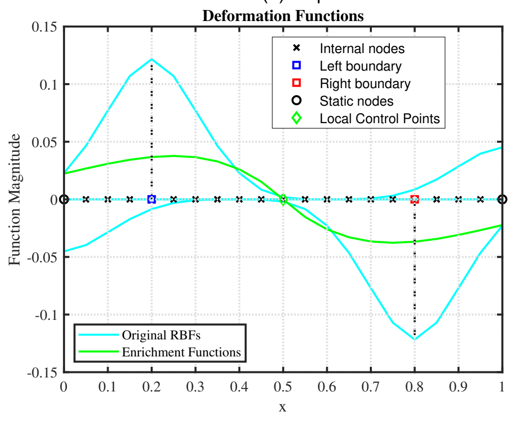
The effect of the enrichment function $H_2(\mathbf{x}, R_l)$ when applied at the centre of the domain is presented in Figure 4.7. Setting the value of $\mathbf{J}$ of -0.5 in Equation 4.3 to the ideal slope value produced improved results as well when compared to the original standard RBF mesh deformation as seen from Figure 4.7a. The displacements of the internal nodes also fall into a linear pattern.



(a) Mesh Quality



(b) Displacement of nodes



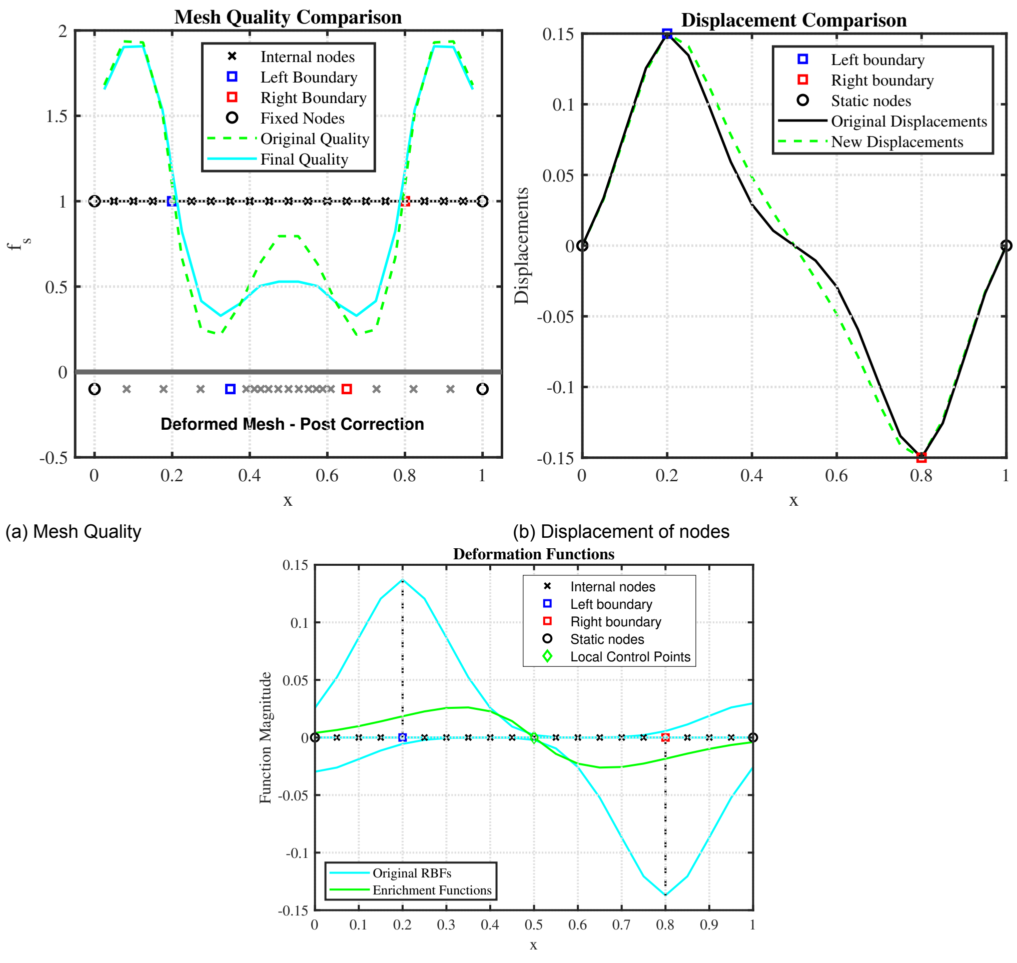
(c) Effect of RBFs

Figure 4.7: Effect of global enrichment function, $H_2(\mathbf{x}, R_l)$ on mesh deformation, $R = 0.4, R_l = 1$

3. $H(\mathbf{x}, R_l) = \frac{\partial \phi(r,R_l)}{\partial x}\phi(r, R_l)$

$$H_3(\mathbf{x}, R_l) = \frac{-20(\mathbf{x}-\mathbf{x}_c)^2(R_l - r^3}{R_l^5}\left(1-\frac{r}{R_l}\right)_+^4\left(4\frac{r}{R_l}+1\right) \tag{4.7}$$

The effect of the enrichment function $H_3(\mathbf{x}, R_l)$ when applied at the centre of the domain is presented in Figure 4.8. Setting the value of $\mathbf{J}$ in Equation 4.3 to the ideal slope value of -0.5 again improves the minimal quality seen in the domain. Similar to the other functions, the displacements are also made to be linear in the critical contraction region.



(a) Mesh Quality

(b) Displacement of nodes

(c) Effect of RBFs

Figure 4.8: Effect of global enrichment function, $H_3(\mathbf{x}, R_l)$ on mesh deformation, $R = 0.4, R_l = 1$

Discussion

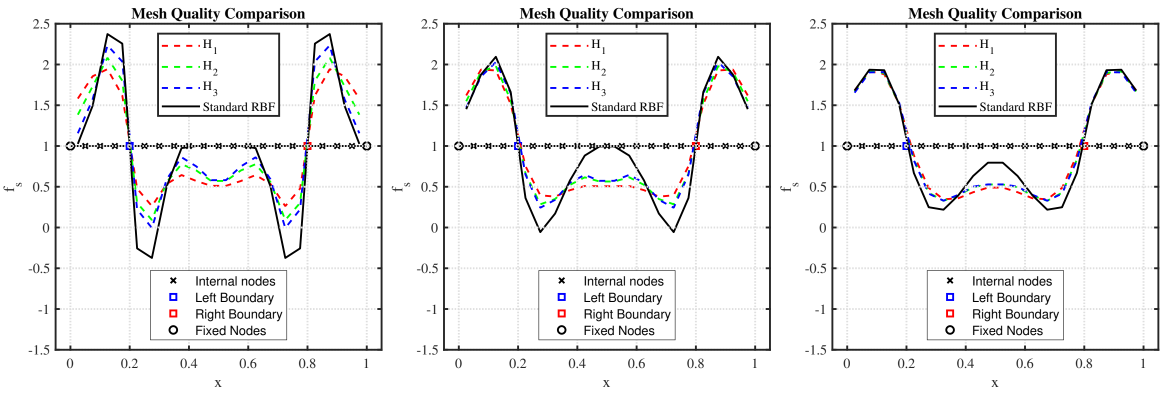
The three functions do perform similarly when the original RBF support radius is large. Therefore, the efficacy of each function was tested by lowering the support radius of the original RBF, thereby introducing sharper gradients to the deformations. The results from the comparison are quantified in Table 4.1 and are depicted in Figure 4.9.

After analysing the results from the three test functions, it can clearly be concluded that the first function provides the best results and hence can be utilized to further analyze the mesh improvements when subjected to multiple compact support enrichment functions to further improve the efficiency as well. Therefore, from here on, whenever the function, $H(\mathbf{x}, R_l)$ is mentioned, the first function from this sub-section will be used to obtain all the results unless specifically stated otherwise.

$$H(\mathbf{x}, R_l) = (\mathbf{x} - \mathbf{x}_c)\left(1 - \frac{r}{R_l}\right)_+^4 \left(4\frac{r}{R_l} + 1\right) \tag{4.8}$$

RBF Support Radius	Minimum Quality Standard RBF	Enrichment Function	Minimum Quality Enrichment Corrections
0.2	-0.373	H_1	0.262
		H_2	0.091
		H_3	-0.006
0.3	-0.056	H_1	0.379
		H_2	0.284
		H_3	0.242
0.4	0.2185	H_1	0.346
		H_2	0.328
		H_3	0.329

Table 4.1: Efficacy of the enrichment functions with variation in RBF support radius, R: Enrichment function support radius, $R_l = 1$, displacement magnitude = 0.15.



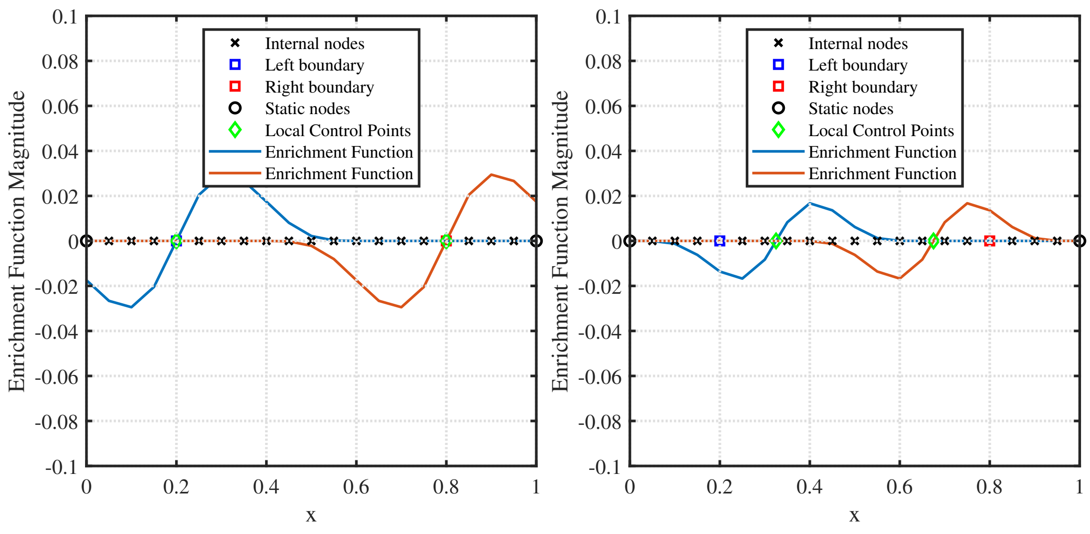
(a) RBF support radius, R = 0.2 (b) RBF support radius, R = 0.3 (c) RBF support radius, R = 0.4

Figure 4.9: Comparison of the three enrichment functions ($R_l = 1$) for varying RBF support radii. Deformation magnitude = 1.5.

4.3.2. Local Correction

It has been discussed previously that global functions often lead to dense system matrices which are computationally very expensive to solve, even if they provide smooth deformations. This severely restricts their usage in practical applications. Furthermore, global functions can lead to the formation of an ill-conditioned system matrix due to the accumulation of a lot of truncation errors and can cause numerical instabilities. Therefore, it is necessary to formulate a method to mimic the advantages that were obtained using the global support correction using compact support functions.

This can be achieved utilizing the same $H(\mathbf{x}, R_l)$ defined in Equation 4.8. Instead of having a single global enrichment function, two compact support functions can be "stitched" together to provide a similar result. The larger the support radius of these local enrichment functions, the closer they come to the results attained by the global function. It has already been discussed in section 4.2 that for compact support enrichment functions the two hypothetical locations at which a control point can be added are the worst quality locations and the boundary locations respectively. Two compact support functions are introduced at these locations and are depicted in Figure 4.10 for reference. It can be seen that the cumulative shape of both the functions combined resembles the global function shape as seen in Figure 4.4 in the region between the moving boundaries. The optimal support radius to be utilized would actually depend on the computational constraints and the magnitude of the quality improvement desired by the user and hence would be subjective to the test case being analyzed.



(a) Enrichment function at boundary nodes, $R_l = 0.4$ (b) Enrichment functions at worst quality locations, $R_l = 0.4$

Figure 4.10: Multiple compact support enrichment functions can be stitched to resemble the single global enrichment correction

The cumulative function shape does differ in regions close to the static nodes when compared to the global function and would introduce an unnecessary correction in the opposite of the desired direction. But the magnitude of this correction is relatively small, so it does not have a significant impact on the overall quality obtained from the correction function. Also, it can be noticed that unlike the global functions, where the ideal slope could be specified at the centre of the domain leading to a linear distribution, the gradients $\mathbf{J}$ need to be specified at the locations of the local control points, where it is not possible to use the ideal slope formation specified in Equation 4.1. Furthermore, as discussed in section 4.2, for the 1D case it is plausible to introduce the control points at the boundary node or at the worst quality locations. For each case, the optimal value of $\mathbf{J}$ needs to be calibrated individually. Both hypotheses are tested in the current analysis to see which would be more beneficial for overall improvement in the mesh deformation algorithm.

But first, the formulation provided in Equation 4.3 needs to be expanded to incorporate the additional correction function. Therefore, the RBF system to obtain the interpolation coefficients $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ can be written as:

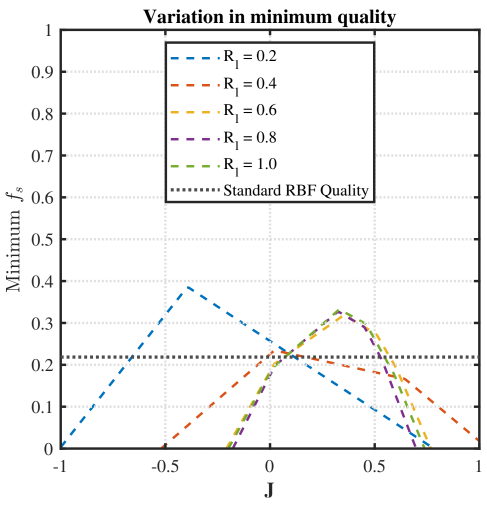
$$\begin{bmatrix} 0 \\ \mathbf{d_b} \\ -\mathbf{d_b} \\ 0 \\ \mathbf{J}_1 \\ \mathbf{J}_2 \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} & \phi_{14} & H_{1c_1} & H_{1c_2} \\ \phi_{21} & \phi_{22} & \phi_{23} & \phi_{24} & H_{2c_1} & H_{2c_2} \\ \phi_{31} & \phi_{32} & \phi_{33} & \phi_{34} & H_{3c_1} & H_{3c_2} \\ \phi_{41} & \phi_{42} & \phi_{43} & \phi_{44} & H_{4c_1} & H_{4c_2} \\ \nabla_x\phi_{1c_1} & \nabla_x\phi_{2c_1} & \nabla_x\phi_{3c_1} & \nabla_x\phi_{4c_1} & \nabla_x H_{c_1c_1} & \nabla_x H_{c_2c_1} \\ \nabla_x\phi_{1c_2} & \nabla_x\phi_{2c_2} & \nabla_x\phi_{3c_2} & \nabla_x\phi_{4c_2} & \nabla_x H_{c_2c_2} & \nabla_x H_{c_1c_2} \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma}_1 \\ \boldsymbol{\gamma}_2 \\ \boldsymbol{\gamma}_3 \\ \boldsymbol{\gamma}_4 \\ \boldsymbol{\beta}_{x_1} \\ \boldsymbol{\beta}_{x_2} \end{bmatrix} \tag{4.9}$$

The displacements of the internal nodes can then be calculated using the formulation:

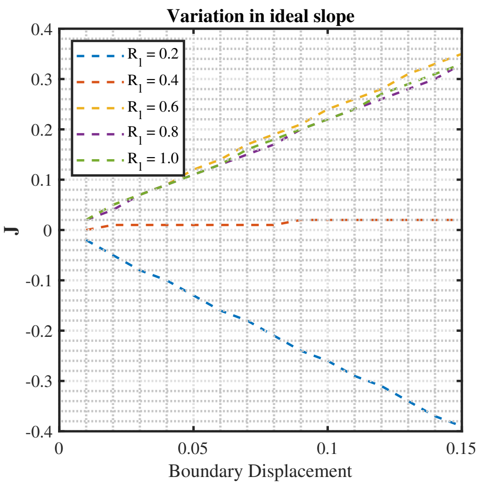
$$[\mathbf{d}_{in}] = [\phi_{in,b} \quad H_{in,c_1} \quad H_{in,c_2}] \begin{bmatrix} \boldsymbol{\gamma}_x \\ \boldsymbol{\beta}_x \end{bmatrix} \tag{4.10}$$

Control Points at Boundary Nodes

Initially, the control points are introduced at the boundary nodes and the analysis is performed. It can be seen from the example showcased in Figure 4.10a that to mimic the global function, the gradient imposed for the enrichment function at the control points (boundary nodes in this case) needs to be positive. Furthermore, since the boundary nodes are equidistant from the centre of the domain in the current test setup, it would make sense to impose the same magnitude of the gradient at both locations as well such that the correction is symmetric and linear between the moving boundaries. Since the gradient is the function of the support radius of the enrichment function, this implies that the same support radius should be used for both functions as well. But the magnitude of the ideal gradient that needs to be imposed at the boundary nodes to attain linearity in the movement of the internal nodes between the moving boundaries is unknown. In order to ascertain the optimal value of the prescribed gradients, a parametric study is performed to analyze what value of prescribed gradients provides the optimal deformation as shown in Figure 4.11. The same test setup parameters that were used for the global correction test case are utilized here. The support radius of the RBFs is set as 0.4. The variation in the ideal imposed gradient for varying support radius of the enrichment function is shown in Figure 4.11a. The imposed gradients are varied with a step size of 0.01. There is only a small range of values for the imposed gradients which actually makes the final mesh quality better than the quality obtained from the standard RBF method. It can be noticed that the trend remains similar irrespective of the support radius. Additionally, the variation in the ideal gradient with respect to deformation magnitude is also presented in Figure 4.11b to ascertain whether a generic trend could be identified for the imposed gradient. The boundary displacements are varied with a step size of 0.01. It is noticed that the ideal prescribed gradients vary linearly with respect to the deformation magnitude.



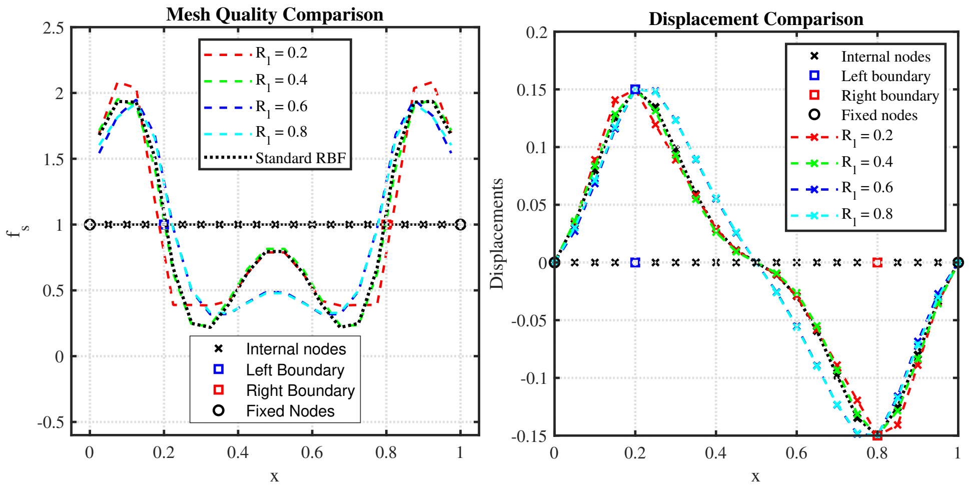
(a) Variation in minimum quality with respect to imposed gradient value for boundary displacement, $\mathbf{d_b} = 0.15$.



(b) Variation in optimal prescribed gradient value with respect to the deformation magnitude

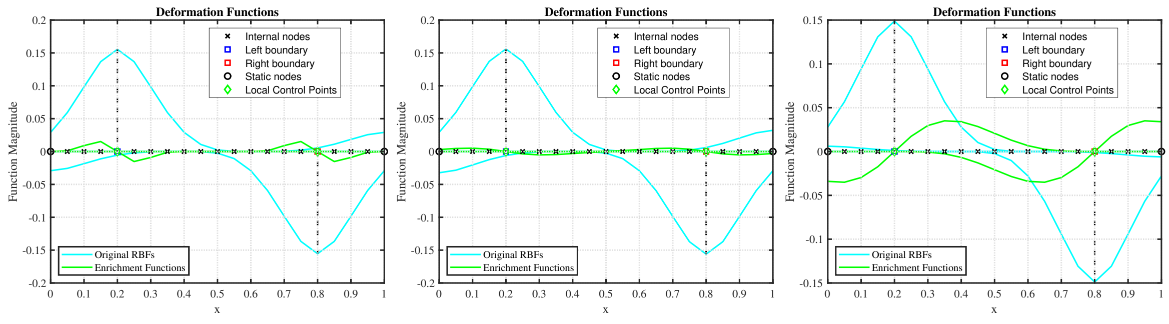
Figure 4.11: Parametric analysis of prescribed gradient values for ideal mesh deformation with control points at boundary nodes, R = 0.4

It is observed that for $R_l < 0.4$, the ideal gradient that needs to be imposed is negative. This can be attributed to the fact that the support radius of the enrichment function is so small that it influences a very small portion of the contraction region as seen in Figure 4.13a. Therefore, its influence is not based on mimicking the global function at all. The trough of the function is attained before the worst quality location. This allows the sharp gradients due to the standard RBF to be controlled by the positive gradient of the enrichment function after the trough leading to a slight improvement in mesh quality. For $R_l = 0.4$, no improvement in the mesh quality is observed at all. This can be attributed to the enrichment function not being able to influence the worst-quality region in a positive manner as its peak is always over-arched by the sharpest gradient region of the RBF, thereby making any correction performed to be an over-correction which worsens the quality. Finally, for $R_l > 0.4$, there is an improvement in the mesh quality and the displacements in the contraction region do become linear as seen from Figure 4.12b, but as the support radius keeps increasing, the two functions cumulatively become more or less a global function. Therefore, the system matrix becomes denser leading to increased computational costs.



(a) Minimum quality comparison for different support radius of the enrichment functions.
(b) Variation in nodal displacements for different support radius of the enrichment functions

Figure 4.12: Comparison of the performance of enrichment function at boundary nodes with an optimal imposed gradient from the parametric study: R = 0.4,$\mathbf{d_b}$ = 0.15.



(a) R_l = 0.2.
(b) R_l = 0.4
(c) R_l = 0.6

Figure 4.13: Comparison of the optimal shapes of the RBFs and enrichment functions for different enrichment support radius. RBF support radius, R = 0.4,$\mathbf{d_b}$ = 0.15.

More importantly, it is noticed that trying to copy the effect of the global enrichment function by imposing the gradient at the boundary node is not a convenient strategy because originally the gradient of the RBFs at the boundary node is equal to zero, by forcing the gradient to adhere to the imposed value at the boundary nodes leads to an over-shoot in the displacement of the internal nodes closest to the boundary nodes as seen from Figure 4.12b. Furthermore, it is not easy to tell what gradient needs to be imposed at the boundary node to attain the ideal gradient value at the centre of the domain as observed in the global enrichment function case. Therefore, since the gradient at the boundary node does not represent the underlying problematic region in the mesh, this approach can be concluded to be unsuitable for performing the local corrections.

Control Points at Worst Quality Locations

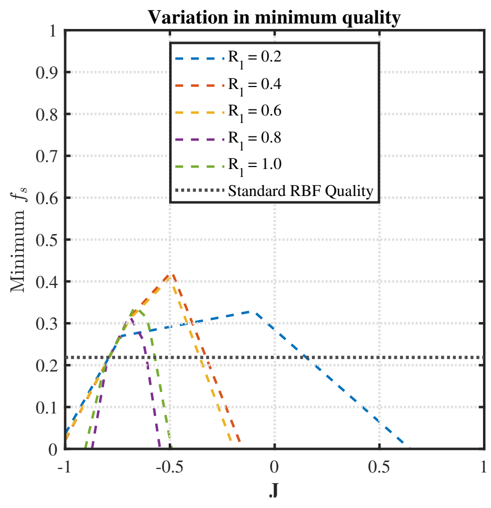
Since it was showcased in chapter 3 that the quality of the mesh in the domain is a function of the gradient of the deformation. Therefore by controlling the gradient at the worst-quality location, the quality at that location can be controlled as well. Therefore the second hypothesis is tested by introducing the local enrichment control points directly at the locations where the worst quality itself is predicted. This means that the control points will be placed at the locations where the mesh quality is being sampled if it is below the user-defined threshold. In the current scenario, as the mesh qualities are being predicted at the cell centres, the additional control points will coincide with them. The algorithm first checks whether the predicted quality is under the threshold at the sampled location and whether it is a local minimum or not, and then applies the correction function there if the minima condition is satisfied.

$$\frac{\partial f_{s_{predicted}}}{\partial x} = 0 \ \ \& \ \ \frac{\partial^2 f_{s_{predicted}}}{\partial x^2} > 0 \tag{4.11}$$

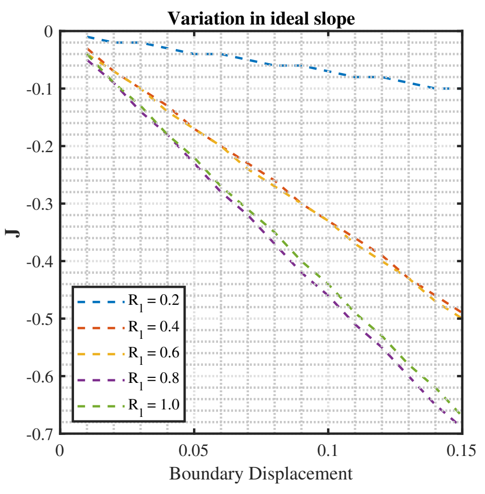
This ensures that if multiple cells are present in proximity to each other with quality predicted to be lower than the threshold, the control point will be added at the worst quality location. But if there are multiple cells in proximity with the same magnitude of bad quality, then the control point is added at only one of those locations, thereby ensuring a smooth and efficient correction.

But in order to understand what gradient needs to be imposed at such locations, first a similar parametric study is performed with respect to the effect of the prescribed gradient with varying support radius and magnitude of displacements as in the previous case. The results from the study are presented in Figure 4.14. It is observed that, unlike the previous case, there is clear consistency in the values of the gradients that need to be imposed. It is clear that the ideal gradient that needs to be imposed should be negative. Furthermore, the value of the imposed gradients $\mathbf{J}_1, \mathbf{J}_2$ in Equation 4.9 are again the same since both the enrichment functions being applied have the same support radius and the displacements are symmetric as discussed previously. The magnitudes again vary linearly with respect to the magnitude of the deformation as seen from Figure 4.14b. It is observed from Figure 4.14a that for compact support radius of the enrichment function i.e. $R_l = 0.2 \rightarrow 0.6$, the ideal gradient used in the global corrections i.e. $\mathbf{J} = -0.5$ provides very good results. Therefore it can be said that for compact support corrections using enrichment functions at the worst quality location in the mesh, the ideal gradient can be imposed based on the already available information about the movement of the boundary nodes. The resulting mesh quality and displacements of the internal nodes using $(\mathbf{J}_1, \mathbf{J}_2 = -0.5)$ is showcased in Figure 4.15 for different support radius of the enrichment function.

It can be seen from Figure 4.16a that there is an improvement in mesh quality for all the support radii tested. But for $R_l = 0.2$, the correction is not very smooth due to the sharp nature of the gradients of the enrichment function itself, thereby not allowing for a smooth deformation that is observed in the other cases. Compact supports do in fact end up performing better than the global corrections as global functions can lead to over corrections as seen from Figure 4.14b for $R_l = 0.8$, where the internal node closest to the boundary node is displaced way more than it needs to be due to the large magnitude of the enrichment function. Therefore it can be said that by placing compact support enrichment functions at the worst quality location and imposing the ideal gradient using data from the boundary nodes, the desired smooth and linear translation can be obtained in the mesh deformation.

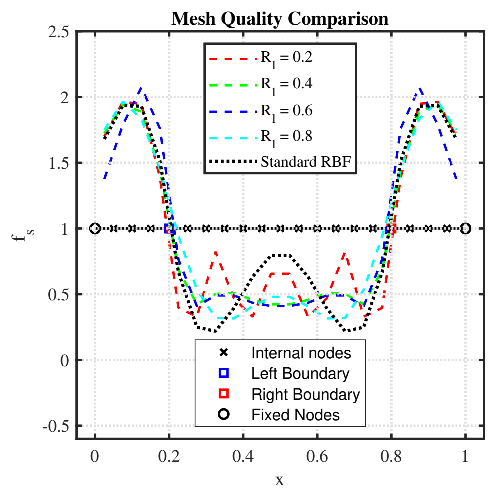


(a) Variation in minimum quality with respect to imposed gradient value for boundary displacement, $\mathbf{d_b} = 0.15$.

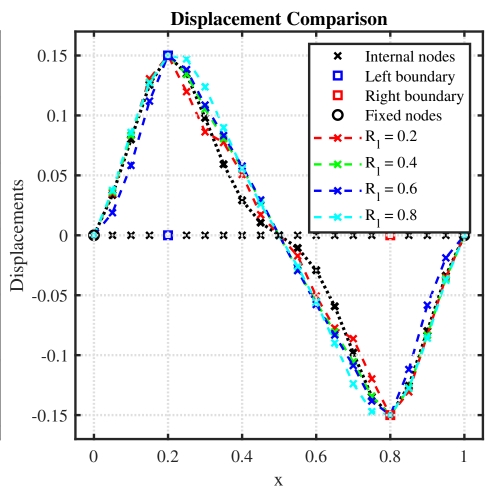


(b) Variation in optimal prescribed gradient value with respect to the deformation magnitude

Figure 4.14: Parametric analysis of prescribed gradient values for ideal mesh deformation with control points at worst quality locations, R = 0.4

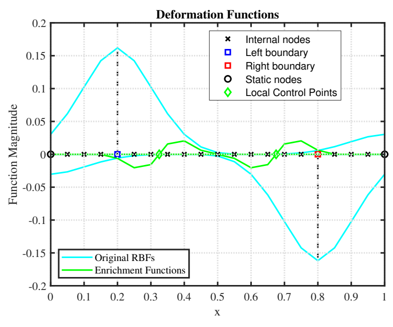


(a) Minimum quality comparison for different support radius of the enrichment functions.

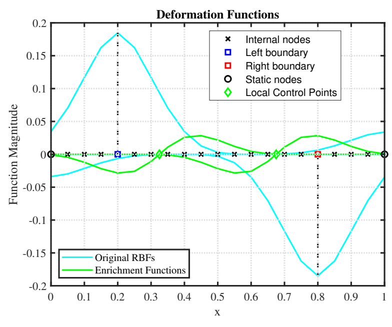


(b) Variation in nodal displacements for different support radius of the enrichment functions

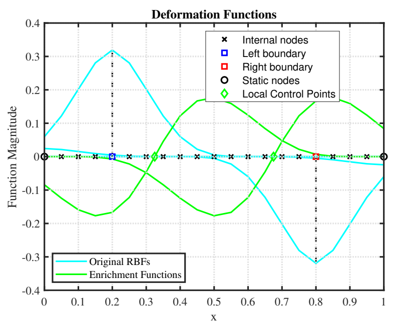
Figure 4.15: Comparison of the performance of enrichment function at boundary nodes with an optimal imposed gradient from the parametric study: R = 0.4,$\mathbf{d_b} = 0.15$.



(a) $R_l = 0.2$.



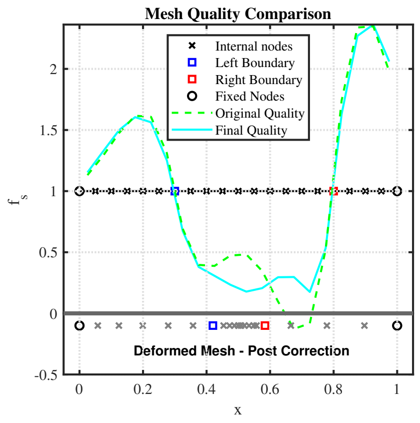
(b) $R_l = 0.4$



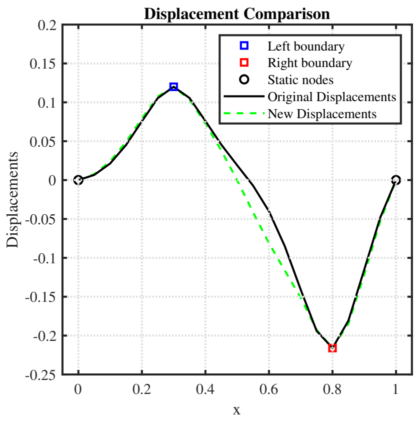
(c) $R_l = 0.6$

Figure 4.16: Comparison of the optimal shapes of the RBFs and enrichment functions for different enrichment support radius. RBF support radius, R = 0.4,$\mathbf{d_b} = 0.15$.

In order to test the robustness of the method for cases where the motion is not symmetric. The location of the boundary nodes is changed along with their magnitudes of displacements. The setup is still targeted at correcting the regions of contraction since that is the most critical region in 1D deformations as discussed in section 4.1. The results from this analysis are showcased in Figure 4.17, Figure 4.18 and Figure 4.19 which actually tests the setup where the contraction region actually lies between the boundary and static nodes. But even for such a movement, the algorithm is able to accurately assign optimal gradients for the deformation at the worst quality location to improve the final mesh quality.

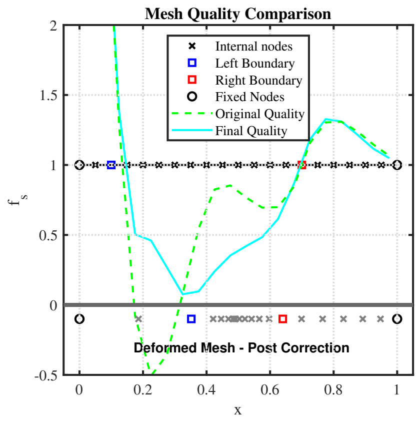


(a) Mesh quality comparison

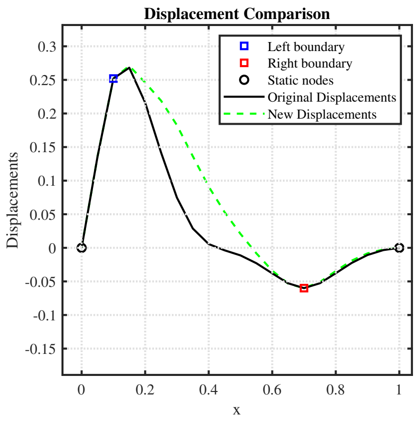


(b) Displacement comparison

Figure 4.17: Local Enrichment correction results: $R = 0.4, \mathbf{d}_{\mathbf{b}_1} = 0.12, \mathbf{d}_{\mathbf{b}_2} = -0.216, R_l = 0.4, \mathbf{J}_1 = \mathbf{J}_2 = -0.672$.



(a) Mesh quality comparison

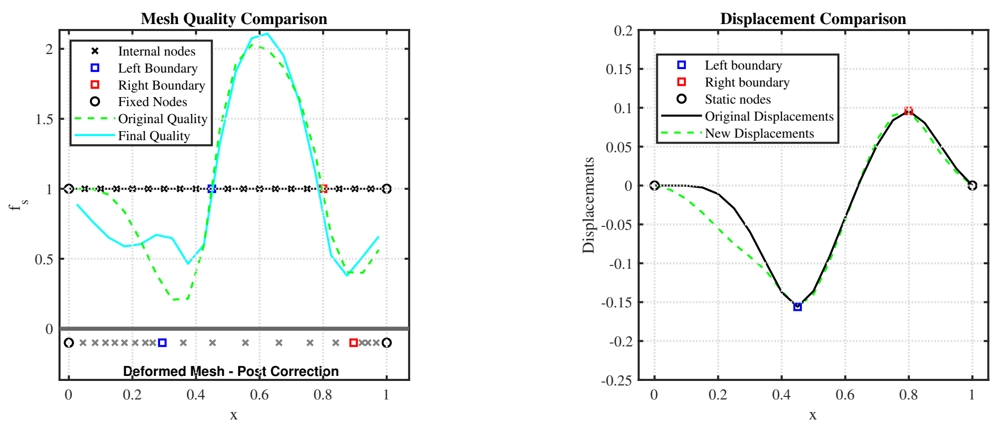


(b) Displacement comparison

Figure 4.18: Local Enrichment correction results: $R = 0.4, \mathbf{d}_{\mathbf{b}_1} = 0.252, \mathbf{d}_{\mathbf{b}_2} = -0.06, R_l = 0.4, \mathbf{J}_1 = \mathbf{J}_2 = -0.52$.

4.4. Discussion

In summary, the concept of enrichment function-based corrections to improve the mesh quality was introduced in the current chapter. It was showcased that a single global enrichment function or multiple local support enrichment functions can be utilized to improve the quality of the deformation. This is done by introducing additional control points either at the centre of the domain (for global) or at the worst quality locations (for local). The system matrix for the deformation was modified by imposing an ideal gradient at the location of these control points, thereby obtaining a smooth and linear deformation. For the global correction, it was obvious in terms of what gradient



(a) Mesh quality comparison (b) Displacement comparison

Figure 4.19: Local Enrichment correction results: $R = 0.4, \mathbf{d}_{\mathbf{b}_1} = -0.156, \mathbf{d}_{\mathbf{b}_2} = 0.096, R_l = 0.4, \mathbf{J}_1 = -0.3466, \mathbf{J}_2 = -0.48$.

to impose, but since global functions lead to denser system matrices which are not efficient to solve, local functions were stitched together to try and mimic the global action. Parametric studies were performed in order to ascertain the ideal gradient for the local enrichment functions. It was found that the boundary data could actually be used in a similar fashion to the global correction to obtain good mesh quality post-deformation. Furthermore, the methodology was showcased to be robust enough to work well for various magnitudes and types of deformation for the boundary nodes. A flow chart describing all the steps involved in the implementation of the enrichment function-based correction algorithm is shown in Figure 4.20.

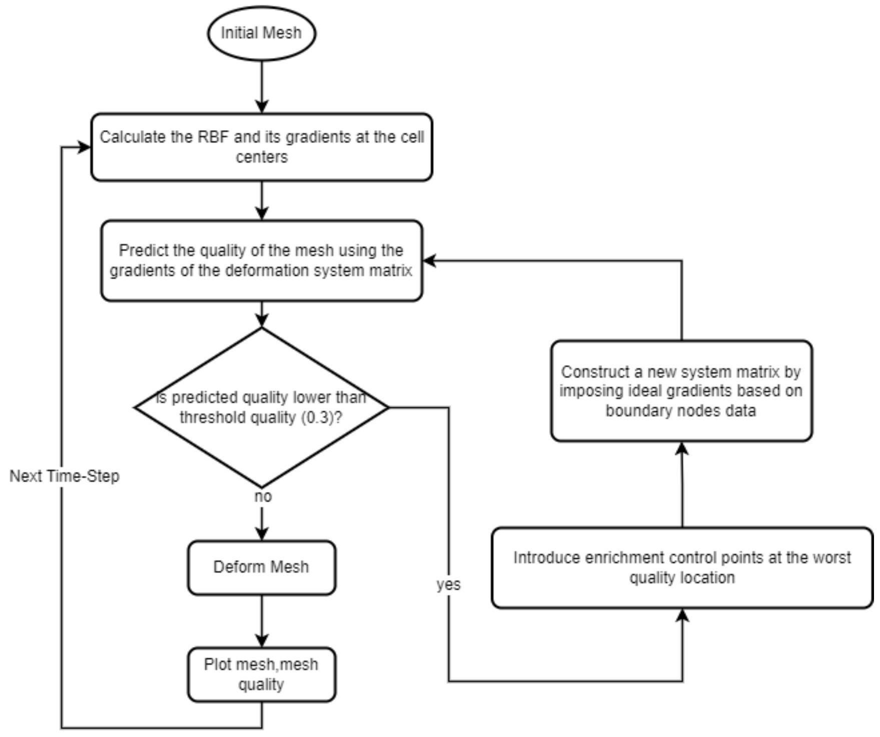

Initial Mesh
Calculate the RBF and its gradients at the cell centers
Predict the quality of the mesh using the gradients of the deformation system matrix
Is predicted quality lower than threshold quality (0.3)?
no
yes
Deform Mesh
Plot mesh,mesh quality
Next Time-Step
Introduce enrichment control points at the worst quality location
Construct a new system matrix by imposing ideal gradients based on boundary nodes data

Figure 4.20: Flow chart depicting the steps involved in the local enrichment function-based correction algorithm.

5 Local Enrichment Function Algorithm: 2D Model

The current chapter deals with expanding the methodology discussed for the local corrections in 1D in the previous chapter to 2D. The challenges associated with doing so are discussed in depth and a pathway is devised in order to overcome said challenges. In order to do so, an in-depth analysis of what the gradients of the deformation function mean is explained, followed by how to extract useful information from them in order to improve the mesh quality. Finally, the methodology is tested on three test cases which encompass all the facets involved in the deformation of a 2D domain.

5.1. Extrapolation of 1D Framework in 2D

5.1.1. Enrichment Function

In the 1D setup in the previous chapter, it was apparent that a linear interpolation around the worst quality location would lead to an ideal increase in the minimum quality of the mesh post-deformation. In order to achieve this result, a control point is introduced into the domain which constrains the gradients of the deformations of the internal nodes by the application of an enrichment function. The global enrichment function which provided ideal results is depicted in Figure 5.1a.

If the same methodology is extrapolated to 2D, then it would appear that when the enrichment function is introduced at the worst quality location, it should have a planar action around its vicinity in order to achieve an ideal deformation in that region. In 1D, the corrections have to be made in only one direction whereas, in 2D, the planar correction has to be accounted for two principle directions as shown in the example in Figure 5.1b. A simple way to ideally perform the local correction would be to split enrichment functions into its two underlying components, each performing corrections in either of the orthogonal directions. Therefore, the enrichment function used in the 1D scenario can be expanded for a 2D grid as:

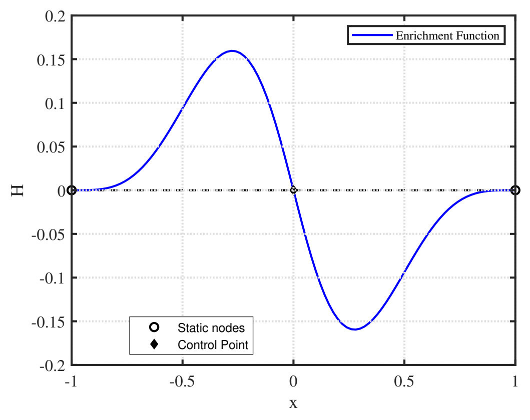
$$\mathbf{H} = H_x\hat{\mathbf{i}} + H_y\hat{\mathbf{j}} \tag{5.1}$$

where H is the enrichment function and

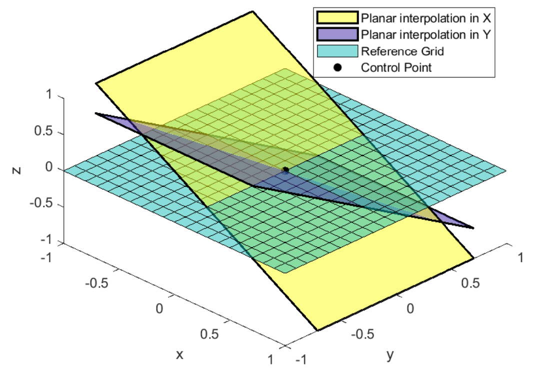
$$H_x = (x - x_c)\phi(,R_l) \tag{5.2}$$

$$H_y = (y - y_c)\phi(r,R_l) \tag{5.3}$$

where, (x_c, y_c) are the location of the control points and (x, y) are locations in the domain. $\phi(r, R_l)$ represents the RBF i.e. Wendland's CPC^2 function as presented in Table 2.1.



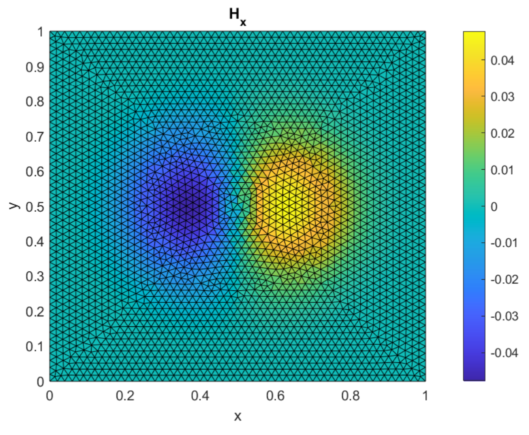
(a) Enrichment function in 1D



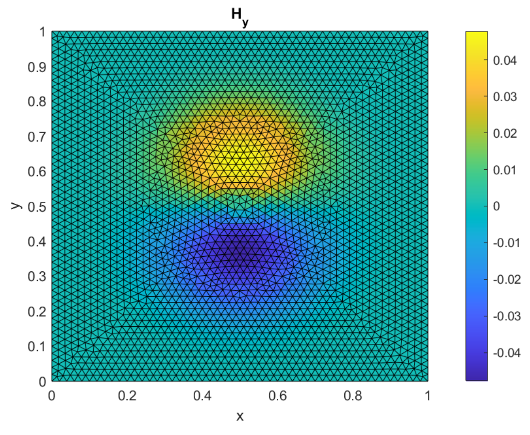
(b) Reference planar action around the worst quality location

Figure 5.1: Rationale behind extrapolation of 1D framework

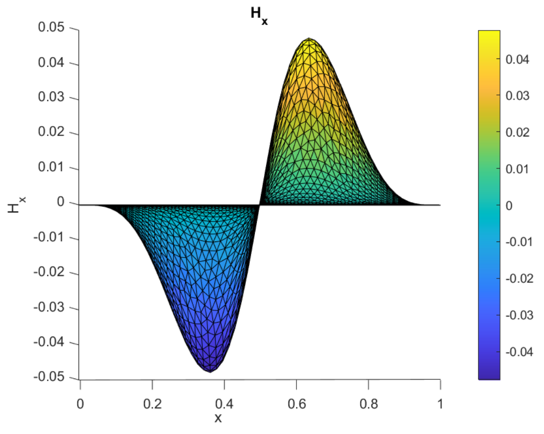
Suppose the control point is to be added at the centre of a reference computational domain. If the principle directions in which the corrections have to be performed are assumed to be in the positive x and y directions respectively. Then the enrichment functions will have an effect as showcased in Figure 5.2. The magnitude of the enrichment function will depend on the support radius, which is chosen to be 0.5 for this particular example.



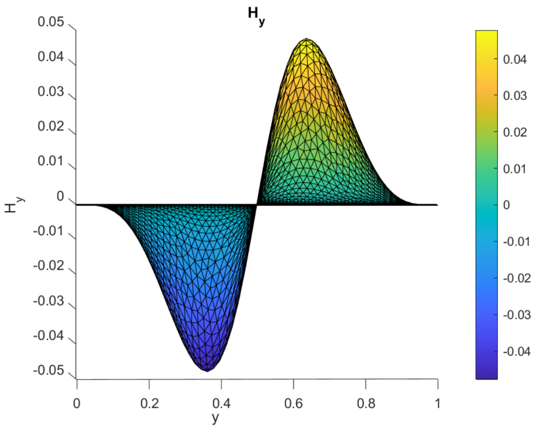
(a) Enrichment function in 1D



(b) Enrichment function in 2D



(c) Projection of enrichment function (H_x) in xz plane



(d) Projection of enrichment function (H_y) in yz plane

Figure 5.2: Depiction of the 2D enrichment function

5.1.2. Coupled Architecture

Now that the functions that are required to perform the corrections are defined, these enrichment functions need to be incorporated into the RBF system such that the local corrections can be performed. In the 1D case, it was found that the ideal way to obtain the interpolation coefficients $\boldsymbol{\gamma}_x, \boldsymbol{\beta}_x$ in order to control the quality of the deformation was by coupling the boundary conditions and imposing the gradient of the deformation at the location of the control point into a single system as shown below:

$$\begin{bmatrix} \mathbf{d}_b \\ \mathbf{J}_S \end{bmatrix} = \begin{bmatrix} \phi_{bb} & H_{bc} \\ \nabla\phi_{bc} & \nabla H_{cc} \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma}_x \\ \boldsymbol{\beta}_x \end{bmatrix} = \begin{bmatrix} \mathbf{S}_x \\ \nabla\mathbf{S}_x \end{bmatrix} \tag{5.4}$$

It has to be noted that in 1D, all the nodes are essentially moving along a single principal direction, therefore the enrichment function also has to act only along one principal direction. Hence, the gradient that needs to be imposed to constrain the deformation is a one-dimensional vector and has just a single value $\mathbf{J}_S$ as denoted in Equation 5.4. But in the 2D case, not only does the magnitude of the gradient have to be imposed, but the direction in which the gradient is imposed also has to be defined. Since there are two principle directions in 2D, $\mathbf{J}_S$ would actually be a matrix of size 2×2 and all the components of the gradient can be represented by the Jacobian matrix of the interpolation function, where:

$$\mathbf{J}_S = \begin{bmatrix} \frac{\partial S_x}{\partial x} & \frac{\partial S_y}{\partial x} \\ \frac{\partial S_x}{\partial y} & \frac{\partial S_y}{\partial y} \end{bmatrix} = \begin{bmatrix} \mathbf{J}_{11} & \mathbf{J}_{12} \\ \mathbf{J}_{21} & \mathbf{J}_{22} \end{bmatrix} \tag{5.5}$$

It is important to note that the $\mathbf{J}_S$ matrix is just the Jacobian of the interpolation function and not the Jacobian of the cell in the mesh, which is represented by $\mathbf{A}_k$ as discussed in chapter 3. This method could in fact be generalized for N dimensions, where the size of the imposed Jacobian would be $N \times N$, which is why a single value is obtained when $N = 1$. Therefore, the final evaluation matrix for the 2D case can be represented as:

$$\begin{bmatrix} \mathbf{d}_x & \mathbf{d}_y \\ \frac{\partial \mathbf{S}_x}{\partial \mathbf{x}} & \frac{\partial \mathbf{S}_y}{\partial \mathbf{x}} \\ \frac{\partial \mathbf{S}_x}{\partial \mathbf{y}} & \frac{\partial \mathbf{S}_y}{\partial \mathbf{y}} \end{bmatrix} = \begin{bmatrix} \phi_{bb} & H_{x_{bc}} & H_{y_{bc}} \\ \nabla_x\phi_{bc} & \nabla_x H_{x_{cc}} & \nabla_x H_{y_{cc}} \\ \nabla_y\phi_{bc} & \nabla_y H_{x_{cc}} & \nabla_y H_{y_{cc}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma}_x & \boldsymbol{\gamma}_y \\ \boldsymbol{\alpha}_x & \boldsymbol{\alpha}_y \\ \boldsymbol{\beta}_x & \boldsymbol{\beta}_y \end{bmatrix} \tag{5.6}$$

For the 1D deformation, it was quite intuitive what the gradient of the deformation that needs to be imposed as the condition should be based on the deformation magnitude and location of the boundary nodes. But in 2D, such a simplification is not possible because the deformation of the boundary induces not just translations in the directions of the principle components, but also induces rotation and skewing of elements. Furthermore, the boundary themselves are not discrete nodal locations anymore but comprise a larger set of nodes which are continuous in terms of their connectivity. Therefore, the imposition of the four parameters for the Jacobian matrix is not so clear. In order to understand the behaviour of the most optimal parameters that need to be imposed, the test case used in chapter 3 is chosen for performing the deformations for the 2D methodology as showcased in Figure 5.3. The domain and internal block specifications are presented in Table 5.1.

Mesh Details	
Nodes	751
Static Nodes	100
Moving Nodes	16
Spacing	0.04
Domain Dimensions	
Height	1 unit
Width	1 unit
Centre	(0.5,0.5)
Inner Block Dimensions	
Height	0.2 unit
Width	0.1 unit
Initial Mesh Quality	
Mean Quality	0.94
Minimum Quality	0.67

Table 5.1: Specifications of the 2D domain.

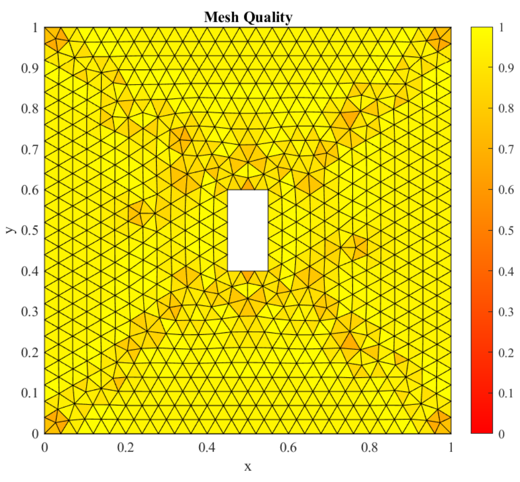


Figure 5.3: 2D test case comprising a square domain with an internal rectangular block.

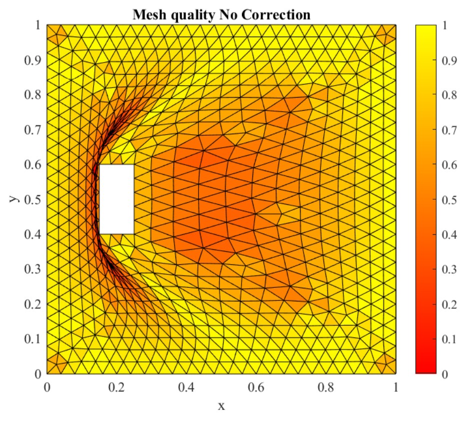
Initially in order to simplify the complexity of the problem, a simple deformation involving pure translation of the block along the principle x-direction is selected such that the setup of the problem becomes similar to the 1D problem whilst still retaining the 2D properties at other places as the RBF function peters out. But as discussed previously, unlike the 1D problem it is not possible to use global information that is already known in order to impose the optimal gradient values. Therefore, a parametric study is initially performed over a support radius, $R_0 = 0.2 \rightarrow 1$ for the enrichment function in order to identify if there is a particular pattern amongst the most optimal imposed gradient values. But first, the prescribed deformation parameters and the quality obtained using the regular RBF method are presented in Table 5.2. As discussed in the previous chapters, all deformations are performed over a single time-step. The deformation using the regular RBF method is showcased in Figure 5.4a. The variables of interest that were varied during the parametric seep are also presented in Table 5.3:

Prescribed Motion	
x displacement	-0.3 units
y displacement	0 units
Rotation	0 °
Regular RBF Mesh Quality	
Mean Quality	0.94
Minimum Quality	0.67

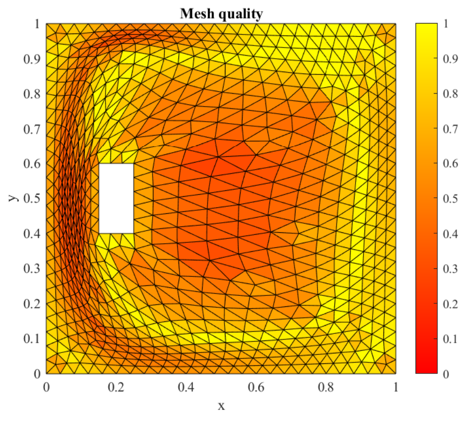
Table 5.2: Parameters for the prescribed deformation and final mesh quality for the regular RBF method

Variable	Varied	Lower Bound	Upper Bound	Step Size
RBF Radius	×	0.5	0.5	-
Enrichment Function Radius	✓	0.2	1	0.2
$\mathbf{J}_{11}$	✓	-3	3	0.2
$\mathbf{J}_{12}$	✓	-3	3	0.2
$\mathbf{J}_{21}$	✓	-3	3	0.2
$\mathbf{J}_{22}$	✓	-3	3	0.2

Table 5.3: Variables of interest that were varied during the parametric study



(a) Original deformation of the block using the regular RBF method, R = 0.5.

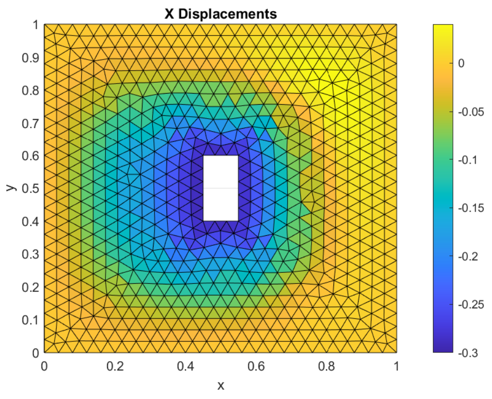


(b) Optimal deformation obtained using the enrichment functions, R = 0.5 and R_0 = 0.8.

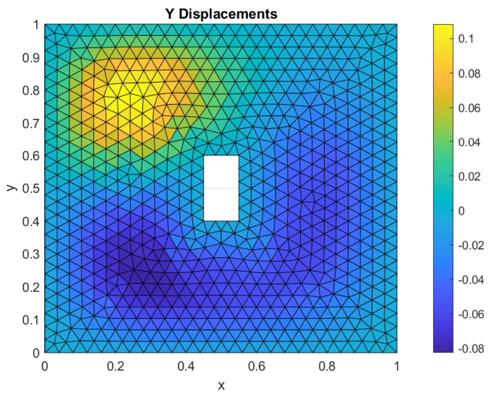
Figure 5.4: Comparison of the deformation of the mesh using regular RBF and local correction RBF methods.

The resulting optimal deformation from the parametric study is showcased in Figure 5.4b. It is observed that using the local RBF methodology, it is possible to improve the minimum quality of the mesh from a degenerate value of -0.25 to 0.26, which is a considerable improvement. Furthermore, it is noticed that even though the deformation is based purely in the X direction, the result from the most optimal local RBF correction method implies that needs to be imposed also perform alterations in the Y direction as shown in Figure 5.5. This is an unusual result because the deformation has been simplified to a pure 1D translation in a 2D domain. Therefore, a reasonable guess to have made for the imposed gradients would be that the corrections in the Y direction are not necessary, thereby implying that $\mathbf{J}_{21}$ and $\mathbf{J}_{22}$ should be equal to zero. But that does not seem to be the case.

In order to better understand the results obtained from the parametric study, response plots of the minimal quality, location of minimal quality and the imposed gradients with respect to the support radius of the enrichment function are showcased in Figure 5.6. It can be seen as expected, as the support radius of the enrichment function increases, the minimal quality increases as the effect of the enrichment function becomes smoother. Additionally, the location of the minimal quality is clearly seen to be constrained to a specific region and there are no sudden shifts in terms of the worst quality location being moved to someplace across the domain. But unfortunately, no discernible trend can be extracted in terms of the imposed gradients that would provide the most optimal mesh deformation. While there is a noticeable improvement in terms of the minimum quality of the mesh, the ideal imposed gradients in fact fluctuate a lot based on the support radius. Hence, it would appear that the methodology prescribed in the 1D formulation might not be very suitable for more complex setups.

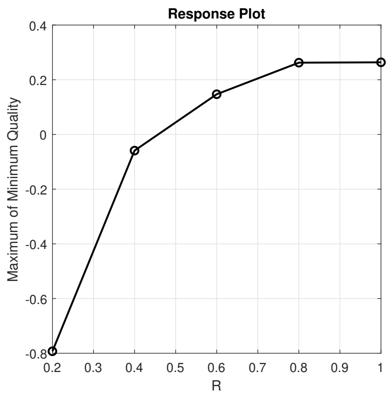


(a) Optimal X displacements obtained using enrichment function

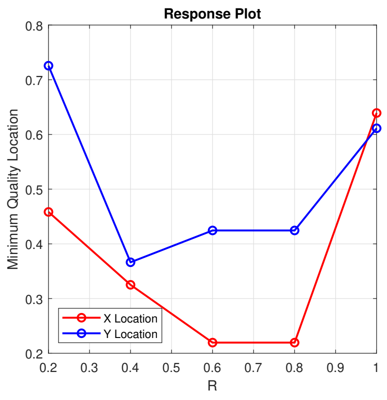


(b) Optimal Y displacements obtained using enrichment function

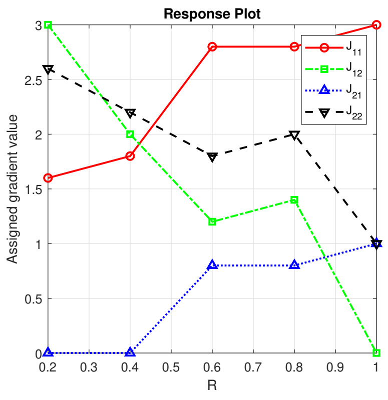
Figure 5.5: Displacements observed for the most optimal formulation of the enrichment function



(a) Response plot of minimal quality with respect to support radius



(b) Response plot of minimal quality with respect to support radius



(c) Response plot of minimal quality with respect to support radius

Figure 5.6: Response plots of various parameters with respect to support radius of the enrichment function

5.1.3. Discussion

Even for a simplified motion i.e. a pure translation in one direction which should be theoretically very similar to the actual 1D test case since there is no nodal deformation in the y-direction, it seems that there is no global trend observable in terms of what the ideal imposed gradient values should be. But it can be clearly seen that even if the condition that needs to be imposed is not clear, the parametric study does showcase that there can be considerable improvement achieved in the final mesh quality using such enrichment functions in the domain. Therefore, a plausible course of action would point towards somehow utilizing the known data in the local region of poor mesh quality instead of depending on a ”global” condition that would immediately fix the mesh.

The location at which the control point needs to be added is known from the mesh quality prediction algorithm discussed in chapter 3. Furthermore, the predicted Jacobian values at the control point are also known. Therefore what needs to be seen is whether these predicted Jacobian values leave some sort of clue in terms of what the ideal prescribed gradients should be in order to constrain the mesh deformation process.

5.2. Interpretation of Predicted Jacobians in 2D

As discussed in the previous section, a clear pattern could not be discerned using a parametric study in order to determine the optimal gradients that need to be imposed as a constraint for the deformation. But since the location at which the enrichment function needs to be added i.e. the location at which the worst quality is predicted is already known along with the nature of the enrichment function that needs to be imposed as discussed in the previous section. The logical step would be to analyze the predicted values of the gradients at the location of the control point and look to somehow counteract/control these predicted gradients in such a manner that a poor-quality region is not developed at another location because of trying to make the original worst quality location perfect. Therefore, an important question that needs to be answered is whether:

”Based on the predicted values of the gradient of the deformation can something be said regarding how the particular cell is going to skew or rotate or compress or expand?”

Now the gradients of the deformation by themselves can not be expected to provide a clear picture of the shape or size or orientation of a cell in the mesh. The gradients only provide information in terms of how the cell will change in the next time step compared to the previous time step. Therefore it is important to incorporate how the cell actually is in the current time-step in order to accurately extract the geometrical data of the cell as explained in chapter 3. It was showcased that:

$$\mathbf{A}_k^{n+1} = (\mathbb{I} + \mathbf{J})\mathbf{A}_k^n = \mathbf{B}\mathbf{A}_k^n \tag{5.7}$$

where $\mathbf{A}_k^{n+1}$ would give the exact representation of the geometrical properties of the cell. But it can be seen that the quality of the cell in the future time-step is inherently dependent on the matrix **B** as the current methodology assumes that the initial mesh is ideal in nature and no improvements need to be performed on it, thereby implying that $\mathbf{A}_k^n$ need not be optimized.

Hence, in order to answer the posed question the **B** matrix needs to be interpreted. Since there are four parameters governing the quality of any particular cell in the mesh. It becomes important to understand what information can be deciphered from each value of the **B** matrix in terms of how the cell is eventually going to deform. To do so, various matrix decompositions could be utilized from available literature in geometry morphing and matrix animations[1, 42] to extract useful data. Some of the most popular choices for matrix decompositions include the Singular Value Decomposition (SVD), QR decomposition, Polar decomposition and Eigen decomposition. The framework behind each decomposition and their advantages and disadvantages are briefly discussed in the upcoming subsections.

5.2.1. Singular Value Decomposition (SVD)

SVD is a very powerful decomposition method generally utilized for dimensionality reduction, data-driven generalizations and geometrical transformations. When it comes to geometrical transformations, the SVD can be used to map a system of interest into a new coordinate system where it would be easier to interpret the data. In the present problem at hand, SVD can be utilized to decompose the **B** matrix as follows:

$$\mathbf{B} = \mathbf{R}_\alpha \Sigma \mathbf{R}_\beta^T \tag{5.8}$$

where $\mathbf{R}_\alpha$ and $\mathbf{R}_\beta$ are orthogonal matrices and represent the set of left singular and right singular vectors respectively. While Σ is a diagonal matrix. In geometrical terms, the decomposition basically implies that firstly $\mathbf{R}_\beta$ rotates the geometry to its principle axes, then Σ scales the geometry followed by which $\mathbf{R}_\alpha$ rotates the system back to its original coordinate system[1]. This implies that the **B** matrix can be decomposed into matrices which provide data in terms of just the scaling of the original cell and the rotation of the cell, which would be very useful in order to constrain unwanted effects such as excessive rotations or contractions.

One of the main advantages of SVD is that it always exists and is always unique. But a fundamental disadvantage with this decomposition is that the rotation matrices can be factored into orthogonal matrices in an infinite number of ways[42]. Therefore, even for the slightest perturbation in the values of the **B** matrix, the rotation angle that can be obtained can be completely different. Furthermore, the computational cost involved in computing the SVD is very high. Hence, due to its instability and computational expense, the SVD can not be used in the current application.

5.2.2. QR Decomposition

The QR decomposition basically decomposes an input matrix into two parts as shown below:

$$\mathbf{B} = \mathbf{QR} \tag{5.9}$$

where $\mathbf{Q}$ is an orthogonal matrix which represents the rotational data and $\mathbf{R}$ is a lower triangular matrix that represents the scaling data. The advantage of the QR decomposition is that it is also always unique as the SVD, but it is also stable and efficient. But the main challenge with using QR decomposition in order to ascertain how a particular cell is deforming is that it is not invariant to the coordinate basis being used[42]. That means that if the deformation needs to be mapped onto another coordinate system for the sake of convenience, then the orthogonal rotation matrix obtained in both these systems would be completely different, thereby reducing its physical significance.

5.2.3. Polar Decomposition

The polar decomposition also decomposes an input matrix into two components as shown below, where $\mathbf{Q}$ is the orthogonal rotational part and $\mathbf{S}$ is the symmetric positive definite part which represents scaling.

$$\mathbf{B} = \mathbf{QS} \tag{5.10}$$

The advantages of the polar decomposition include that the factorized forms are always unique, coordinate independent and stable, unlike the SVD and QR decompositions. The decomposition can be performed in a simple and efficient manner as well. The polar decomposition essentially factors out the closest orthogonal matrix to **B** in the Frobenius norm sense[36] i.e:

$$||\mathbf{B} - \mathbf{Q}||_F^2 < ||\mathbf{B} - \mathbf{X}||_F^2 \quad \forall \mathbf{X} \in \mathbf{Q}(n), \mathbf{X} \neq \mathbf{Q} \tag{5.11}$$

But when the determinant of **B** is less than zero, then the closest rotation actually becomes a reflection. Thereby indicating that the cell becomes inverted when $\det(\mathbf{B}) < 0$ as shown below:

$$\mathbf{Q}_{validcell} = \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix}, \qquad \mathbf{Q}_{invalidcell} = \begin{bmatrix} a & b \\ b & -a \end{bmatrix} \tag{5.12}$$

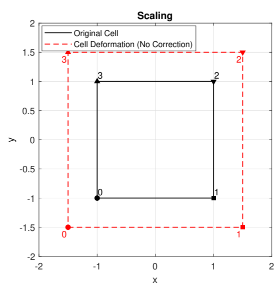
where $a^2 + b^2 = 1$. Though the polar decomposition does not in fact provide any useful information in terms of what is causing the cell to invert. Even if the reason for inversion is a very high magnitude of contraction with no rotation at all, the rotation matrix will be obtained as a reflection matrix. For an inverted element, simply flipping the nodes i.e. correcting for the scaling can lead to a valid cell. But how to improve the quality of a low-quality skewed/rotated cell is still an unresolved question, especially since the input matrix for the decomposition does not represent the nodal locations in the mesh. Therefore, at the current stage, polar decomposition is suggested as a valid idea for further research in the future.

5.2.4. Eigen Decomposition

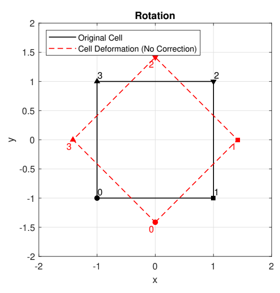
Finally, one of the most common and simple decompositions in use is the eigen decomposition. The eigen decomposition basically factors the input matrix into a set of eigen values and eigen vectors as shown below:

$$\mathbf{Bv} = \lambda\mathbf{v} \tag{5.13}$$

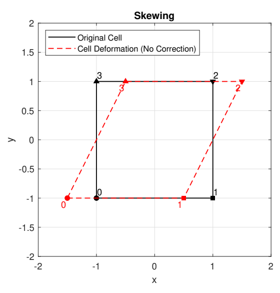
where λ represents the eigen values and **v** represents the eigen vectors. The eigen vectors in the current scenario basically represent the principle directions along which the cell is scaled. The eigen values give the magnitude of the scaling and hence can actually be referred to as the scaling factors. If either of the scaling factors is negative, it basically implies that the original principle direction has reversed and hence the cell becomes inverted. The major disadvantage of using eigen decomposition is that it does not provide any valuable data on the rotation or shearing of the cell. But it is extremely useful in determining how the cell is scaling and hence, would be the most efficient way to identify if a particular cell has inverted or not.



(a) Scaling



(b) Rotation



(c) Skewing

Figure 5.7: Generic deformations of a reference quadrilateral cell.

5.2.5. Discussion

As discussed in the previous sub-sections, it is definitely possible to decompose a given input matrix into multiple factors representing fundamental operations such as scaling and rotations. These properties have the advantage that they are generic to all types of cells irrespective of shape or type. But an essential drawback that was noticed in the decompositions was that there does not seem to be a way to explicitly factor out the properties such as skewness, orthogonality and so on which are specific to the type or orientation of the cell. Although it is seen from the SVD, QR and polar decompositions that the shear can be represented as a rotation and a scaling operation. Furthermore, even when clear data can be extracted in factors of scaling and rotation, it is not always intuitively easy to grasp which part needs to be corrected.

Therefore, it is decided that for the current thesis, it would make sense to concentrate on ensuring that inverted cells in the domain are corrected as a priority over specifically targeting individual factors involved in the worsening of mesh quality. In order to achieve this aim, it would make the most sense to work with eigen decompositions as they are relatively inexpensive to compute when compared to the other decompositions and provide good information with regard to how the cell is scaling along its principle components. Furthermore, since the main priority is to prevent inversions, it would also make sense to concentrate on the size metric for the mesh quality which is based on the determinant of the Jacobian of the cell and is computationally less expensive to compute.

5.3. Method of Explicit Iterations

Now that valid geometrical interpretations can be made from the available gradient information at the location where the enrichment function needs to be added, it is necessary to come up with a framework to perform the local corrections. As shown from the parametric study in section 5.1, it is not possible to assign optimal gradients as an interpolation condition as was done in the 1D setup. Therefore, an explicit scheme is developed in order to perform the corrections iteratively.

5.3.1. Eigen Corrections

What this essentially means is that once the mesh quality drops below a certain user-defined threshold, the gradient matrix at that location where the control point needs to be added, **J** is obtained from the mesh quality prediction algorithm as discussed in chapter 3. Then the **B** matrix is calculated and its determinant is calculated in order to see how to improve the mesh quality at that location. If $\det(\mathbf{B} < 0)$, it means the cell is inverted and implies that one of the eigen values is negative as seen from Equation 5.14. Therefore, a positive scaling factor is introduced as the eigen value so that the determinant becomes positive. In the case that both eigen values are negative, the cell might not be identified through the determinant, but it is still invalid due to the flipping of its orientation. This can occur in extreme cases of distortion, for example when a cell is being squeezed in every direction such that it is flipped as shown in Figure 5.8.

Now a new **B** matrix is calculated while ensuring that the eigen vectors are still the same so that the principle axes of the cell are not modified. This would ensure that the orientation of the cell itself would not change, just that excessive contraction of the cell is curtailed.

$$\det(\mathbf{B}) = \prod \lambda = \lambda_1 \cdot \lambda_2 \tag{5.14}$$

But if the cell does not face any problems with inversions, just that its quality is low due to excessive deformations in one of the principle directions. Then, the corrections performed are aimed at reducing the ratio between the two scaling factors. Therefore,

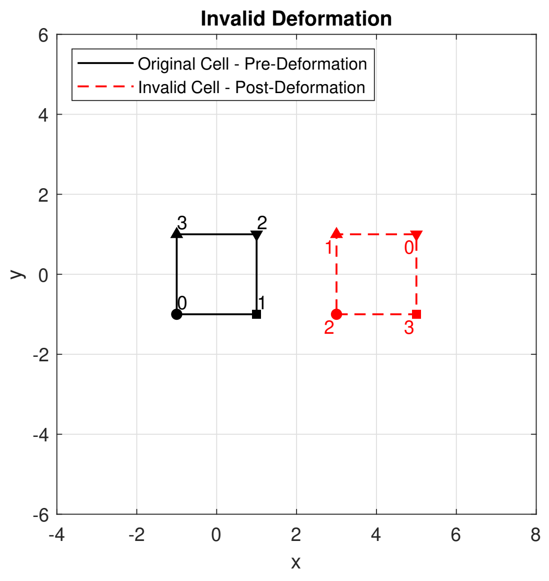


Figure 5.8: Invalid deformation even if the determinant is positive

using the eigen corrections, it is possible to construct a new gradient matrix $\mathbf{J}_{new}$ at the location of the control point such that the quality at that location is improved. A brief description of this methodology is presented in algorithm 1.

Algorithm 1 Methodology for eigen corrections

```
Require: J
Ensure: $\det(\mathbf{B}) > 0$
  $\mathbf{B} = \mathbb{I} + \mathbf{J}$
  $[\mathbf{V}, \lambda] = eig(\mathbf{B})$
  if $\det(\mathbf{B}) \leq 0$ then
      for i ←1, 2 do
          while $\lambda(i,i) <= 0$ do
              $\lambda(i,i) \leftarrow 0.1$                    ▷ The eigen value is made positive
          end while
      end for
  else
      if $\lambda(1,1) < \lambda(2,2)$ then                 ▷ Reduce ratio between the scaling
          $\lambda(1,1) \leftarrow \lambda(1,1) + 0.1$
      else
          $\lambda(2,2) \leftarrow \lambda(2,2) + 0.1$
      end if
  end if
  $\mathbf{B}_{new} = \mathbf{V}\lambda\mathbf{V}^{-1}$
  $\mathbf{J}_{new} = \mathbf{B}_{new} - \mathbb{I}$
  $\mathbf{dJ} = \mathbf{J}_{new} - \mathbf{J}$
```

5.3.2. Explicit Iterations

It was noticed in the 1D setup that performing a single correction with ideal gradients imposed at the location of the control points would always ensure a good-quality mesh. But in the current case, the corrections being made to the gradients are not ideal and hence provide no guarantee that the overall quality of the mesh is improved. It is possible that the mesh quality around the local region of the control point improves, but can eventually lead to the worsening of the mesh quality in some other regions.

Therefore, a methodology is developed to perform these corrections in an iterative manner. In order to understand how these iterative corrections are performed, first it is important to decouple the system presented in Equation 5.6 and ignore the imposed condition of satisfying gradients. The goal is to find the interpolation coefficients $\boldsymbol{\gamma}_x, \boldsymbol{\gamma}_y$ such that the mesh quality is improved while satisfying the boundary conditions as shown below:

$$\begin{bmatrix} \mathbf{d}_x & \mathbf{d}_y \end{bmatrix} = \begin{bmatrix} \phi_{bb} & H_{x_{bc}} & H_{y_{bc}} \end{bmatrix} \begin{bmatrix} \boldsymbol{\gamma}_x & \boldsymbol{\gamma}_y \\ \boldsymbol{\alpha}_x & \boldsymbol{\alpha}_y \\ \boldsymbol{\beta}_x & \boldsymbol{\beta}_y \end{bmatrix} \tag{5.15}$$

Therefore, the entire system can be re-written in order to find appropriate interpolation coefficients $\boldsymbol{\gamma}_x, \boldsymbol{\gamma}_y$ as follows:

$$\begin{bmatrix} \boldsymbol{\gamma}_x & \boldsymbol{\gamma}_y \end{bmatrix} = \underbrace{\phi_{bb}^{-1}}_{\text{Part 1}} \left[\underbrace{\begin{bmatrix} \mathbf{d}_x & \mathbf{d}_y \end{bmatrix}}_{\text{Part 2}} - \underbrace{\begin{bmatrix} H_{x_{bc}} & H_{y_{bc}} \end{bmatrix}}_{\text{Part 3}} \underbrace{\begin{bmatrix} \boldsymbol{\alpha}_x & \boldsymbol{\alpha}_y \\ \boldsymbol{\beta}_x & \boldsymbol{\beta}_y \end{bmatrix}^i}_{\text{Part 4}} \right], \qquad \text{i = iteration} \tag{5.16}$$

From the above equation, it can be seen that for any $\boldsymbol{\alpha}, \boldsymbol{\beta}$, it is always possible to obtain $\boldsymbol{\gamma}$ such that the boundary conditions are met. However, improving the mesh quality in the local region around the control point requires $\boldsymbol{\alpha}, \boldsymbol{\beta}$ to be chosen appropriately. Now from the eigen corrections, the new gradient matrix at the control point $\mathbf{J}_{new}$ was calculated which ensured that the cell quality was improved compared to the original $\mathbf{J}$ matrix. Therefore, the difference between $\mathbf{J}_{new}$ and $\mathbf{J}$ i.e. the magnitude by which the gradients have to be corrected can be imposed as the interpolation conditions $\boldsymbol{\alpha}, \boldsymbol{\beta}$ such that the problem with cell inversion or low quality is curtailed.

$$\mathbf{dJ} = \mathbf{J}_{new} - \mathbf{J} = c \begin{bmatrix} \boldsymbol{\alpha}_x & \boldsymbol{\alpha}_y \\ \boldsymbol{\beta}_x & \boldsymbol{\beta}_y \end{bmatrix} \tag{5.17}$$

To increase the rate of convergence to the optimal quality, the calculated interpolation coefficients were multiplied with a constant over-relaxation factor. But since improving the cell quality in a local region can lead to worsening quality in other regions, it becomes important to predict the quality based on the newly obtained interpolation coefficients. If it is found that there are still cells with minimum quality lesser than the user-defined threshold, then another control point would be added in the new location which would curtail the gradient of deformations in that region in the next iteration. Therefore, with every iteration, the number of control points would increase and hence the size of the matrices in part 3 and part 4 of Equation 5.16 would increase in order to account for the gradient correction at each location of the control point.

Another beneficial factor for this methodology is that part 1 and part 2 of Equation 5.16 do not change since the displacements are already known and the RBF matrix needs to be computed only once. Furthermore, ϕ_{bb} is not singular and is symmetric, thereby ensuring that it is always easily invertible. In order to gain a complete understanding of the entire flow process from the beginning, the methodology for performing the explicit corrections is presented in Figure 5.9.

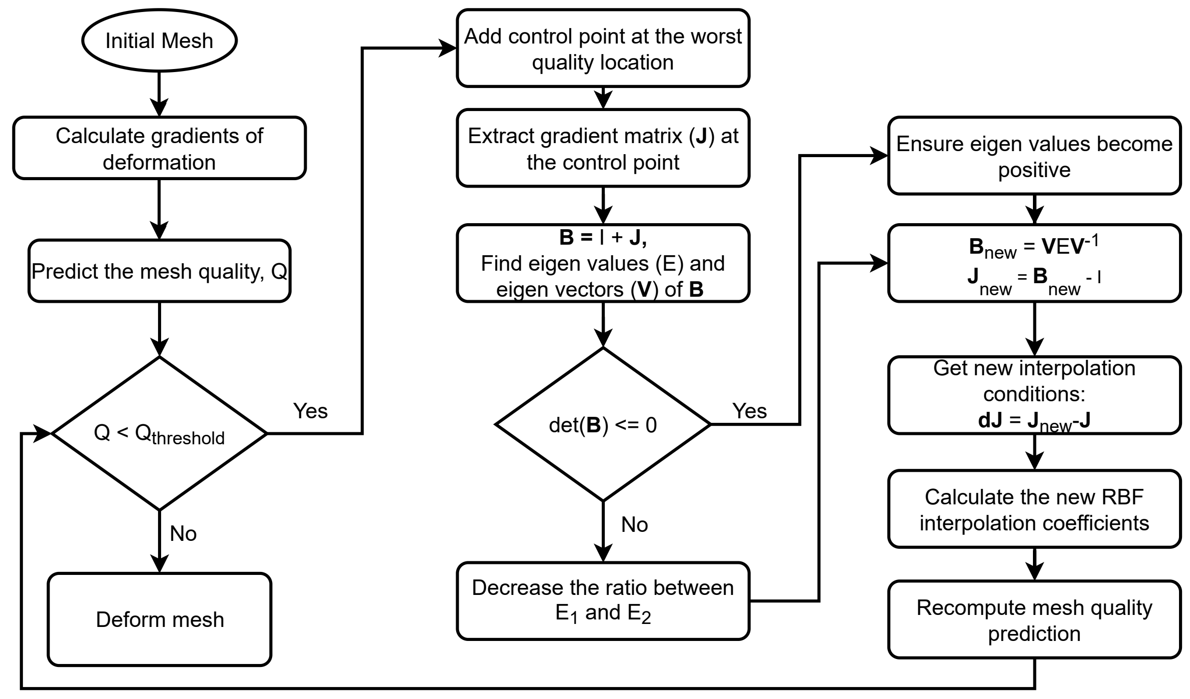

Initial Mesh
Calculate gradients of deformation
Predict the mesh quality, Q
Q < Q_threshold
Yes
No
Deform mesh
Add control point at the worst quality location
Extract gradient matrix (J) at the control point
B = I + J, Find eigen values (E) and eigen vectors (V) of B
det(B) <= 0
Yes
No
Decrease the ratio between E1 and E2
Ensure eigen values become positive
B_new = VEV^-1
J_new = B_new - I
Get new interpolation conditions: dJ = J_new-J
Calculate the new RBF interpolation coefficients
Recompute mesh quality prediction

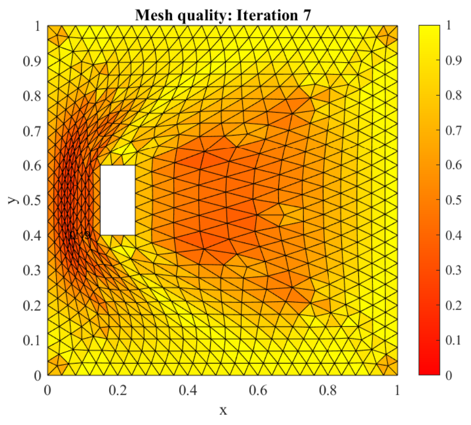
Figure 5.9: Flow chart for the 2D local correction methodology

5.4. Results

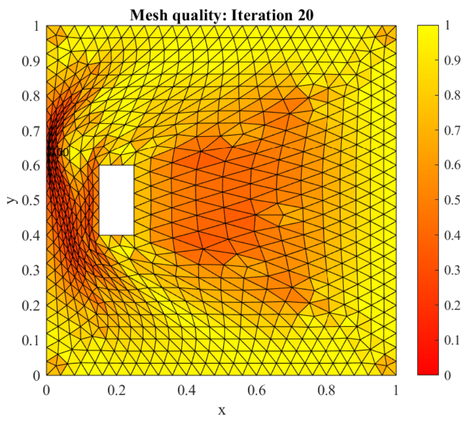
5.4.1. Translation in X direction

Now that the methodology to perform the local corrections has been discussed in the previous section, it is necessary to see how the method performs in terms of improving the mesh quality. Therefore, the same test case that was used in the parametric study in section 5.1 is again utilized and the properties of the motion can be obtained from Table 5.2. The user-defined threshold for the minimum quality is set as 0.2 for the current scenario, implying that if the predicted quality exceeds this value, the mesh will be deformed. Now in many cases of deformations, it may not be possible to achieve the threshold quality due to the severity of the deformation involved. Therefore, to ensure that the algorithm does not get into an infinite loop of trying to constantly improve the mesh to no avail, the number of control points that can be added is curtailed to 20. Thereby, implying that a maximum of 20 corrections or explicit iterations will be performed. Furthermore, the support radius for the RBF and the enrichment function for all the results discussed in this section is set as 0.5. The results from the method of explicit iterations are showcased in Figure 5.10.

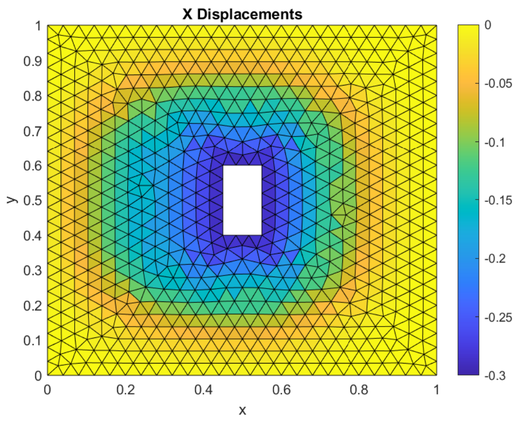
It is observed that the best minimal mesh quality of 0.19 is achieved at the seventh iteration using this method. After which the quality fluctuates a lot and gets considerably worse. It is also noticed from Figure 5.10e that the convergence between the predicted mesh quality and the actual mesh quality worsens from the seventh iteration onwards. This anomaly can be chalked up to the extreme case of the translation involved where it is not possible to further improve the mesh locally. Due to this multiple control points are added at the same location which ends up counteracting each other thereby leading to incorrect predictions of the mesh quality as well. But overall, if the user-defined threshold was not set so high or the magnitude of deformation was not so severe, the method does seem to improve the overall mesh quality as shown for other cases in the upcoming sections. A comparison of the original mesh quality post-deformation using the regular RBF method, the parametric study for imposed gradients and the method of explicit iterations is provided in Table 5.4.



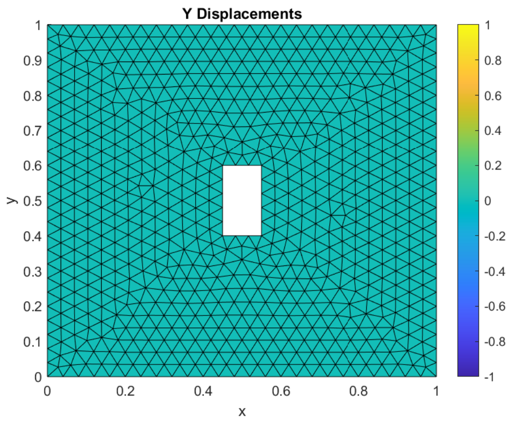
(a) Best quality of the mesh observed on iteration 7.



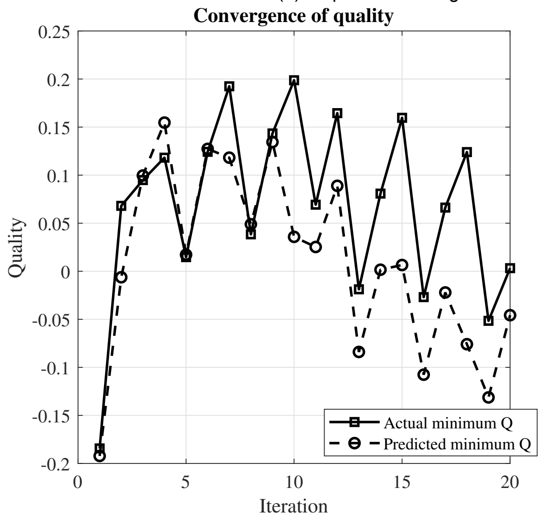
(b) Quality of the mesh after 20 iterations



(c) Displacement magnitudes in X direction on the seventh iteration



(d) Displacement magnitudes in Y direction on the seventh iteration



(e) Convergence properties of the predicted and actual mesh quality

Figure 5.10: Results obtained for the mesh deformation from the method of explicit iterations.

Regular RBF method	
Minimum Quality	-0.25
Imposed Gradients: Parametric study	
Minimum Quality	0.26
Method of Explicit Iterations	
Minimum Quality	0.19

Table 5.4: Comparison of the mesh qualities using the various methods

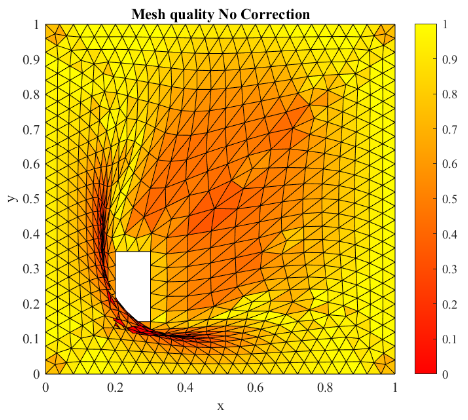
5.4.2. Translation in X and Y directions

Now that it is seen that for a simple case of 1D translation, the method of explicit iterations does perform commendably even when encountering large deformations. It is interesting to see the performance of the algorithm when the deformation takes place in both directions. Therefore, firstly the same test case is subjected to translations in both X and Y directions according to the parameters presented in Table 5.5.

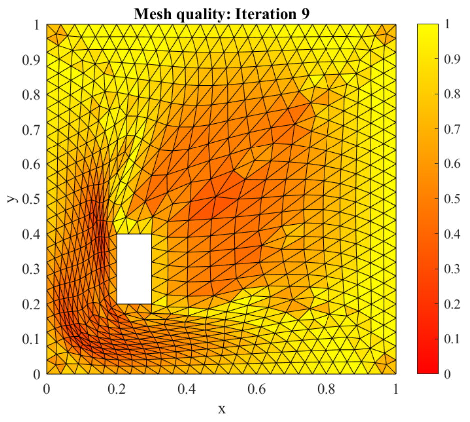
Prescribed Motion	
X displacement	-0.25 units
Y displacement	-0.25 units
Rotation	0 °
Support Radius	
RBF	0.5 units
Enrichment function	0.5 units

Table 5.5: Parameters for the prescribed deformation and final mesh quality for the regular RBF method

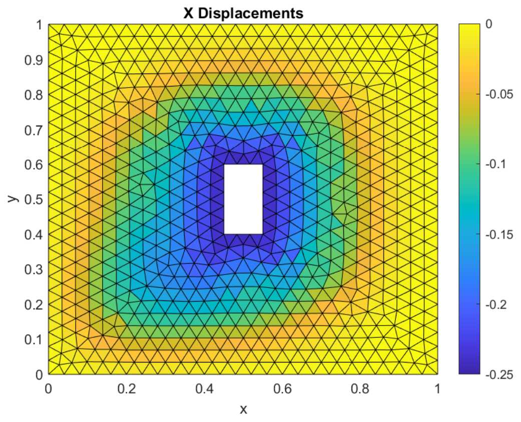
The user-defined threshold quality is still maintained at 0.2 and the maximum number of iterations remains at 20. It is observed that using the regular RBF method, the original mesh deformation has a minimum quality of -0.53 while using the explicit iteration method it can be improved to approximately 0.2 within 14 iterations as seen from Figure 5.11 and Figure 5.12. From the results, it is quite evident that the method of explicit iterations is able to accurately tackle the problematic regions in the mesh regardless of how convoluted a cell inversion is. It is able to improve the mesh quality without violating the boundary conditions of the domain. The locations at which the enrichment functions are being added are showcased in Figure 5.12b and their cumulative effect within the domain is depicted in Figure 5.11e and Figure 5.11f. Furthermore, it is interesting to note that if the threshold value is reduced to a lower margin, then the number of iterations reduce considerably. This would be very beneficial to cases with extreme deformations where low-quality elements are expected and are not considered a huge issue unless encountering invalid elements. For an actual FSI simulation, the low mesh quality might require a greater number of iterations for the fluid flow solver to converge, but the complicated and computationally expensive step of re-meshing can be avoided.



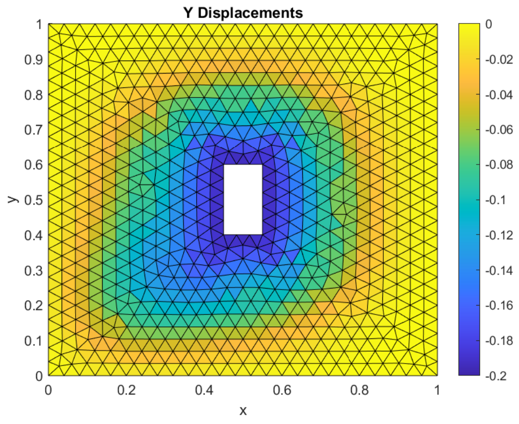
(a) Mesh quality using the regular RBF method.



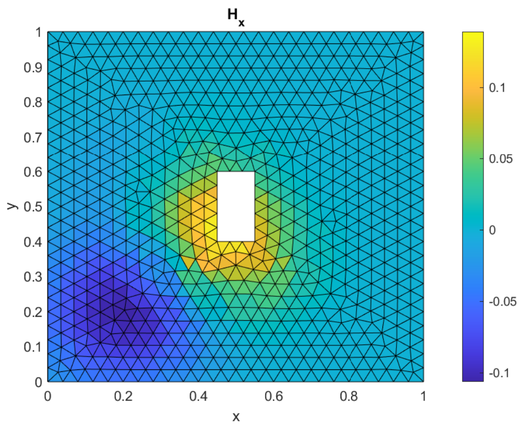
(b) Best quality of the mesh observed on iteration 9.



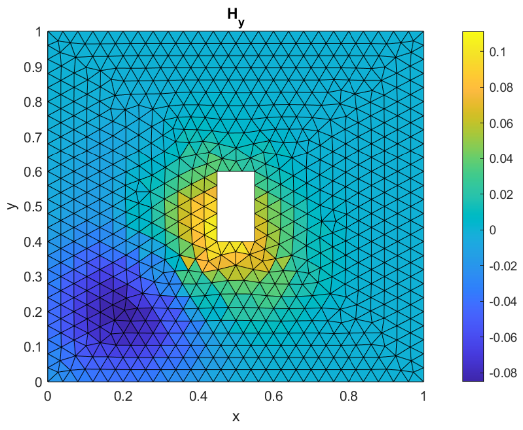
(c) Displacement magnitudes in X direction



(d) Displacement magnitudes in Y direction

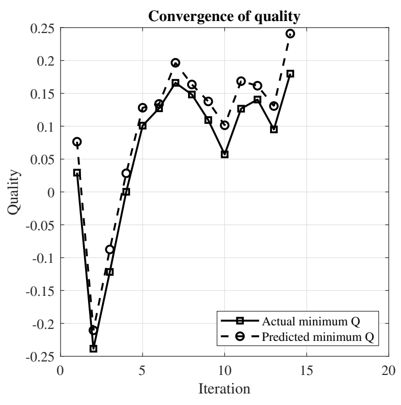


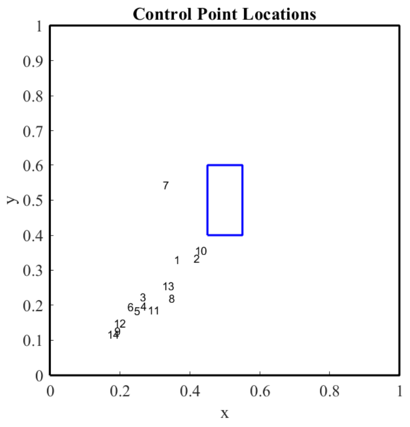
(e) Enrichment function, H_x magnitude



(f) Enrichment function, H_y magnitude

Figure 5.11: Results obtained for the mesh deformation from the method of explicit iterations.





(a) Convergence properties of predicted and actual mesh quality (b) Locations at which the control points are added

Figure 5.12: Convergence properties and control point locations

5.4.3. Validation: Translation and Rotation

Now that it has been shown that the method of explicit iterations works for 2D motions as well, it would be interesting to validate the quality of the deformation with other test cases available in the literature. So it is decided to compare the resulting quality of the mesh post-deformation with the test case used by de Boer et al.[7] and Mathew[33] due to readily available data with regards to the meshing and deformation parameters. The test case used in this case was initially used by de Boer et al. to compare the performance of various kinds of RBFs on mesh deformation and then by Mathew to compare the performance of the sliding boundary mesh deformation method. The parameters for the mesh and deformation are provided in Table 5.6. The initial mesh is showcased in Figure 5.13.

Mesh Details	
Nodes	751
Static Nodes	100
Moving Nodes	16
Spacing	0.04
Domain Dimensions	
Height	1 unit
Width	1 unit
Centre	(0.5,0.5)
Inner Block Dimensions	
Height	0.04 unit
Width	0.2 unit
Prescribed Motion	
X displacement	-0.2 units
Y displacement	-0.2 units
Rotation	60 °

Table 5.6: Parameters for the mesh and prescribed deformation

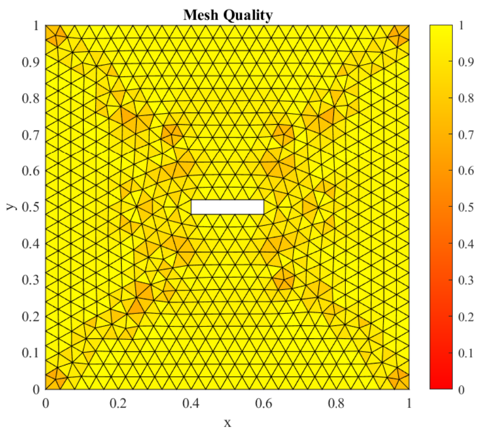
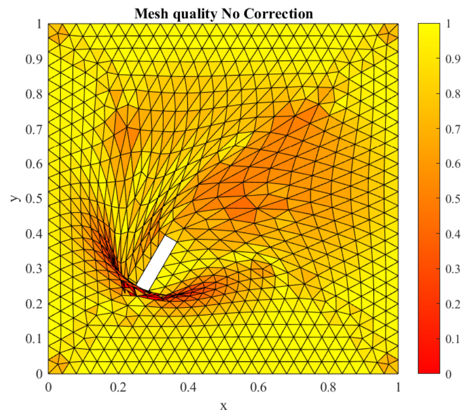
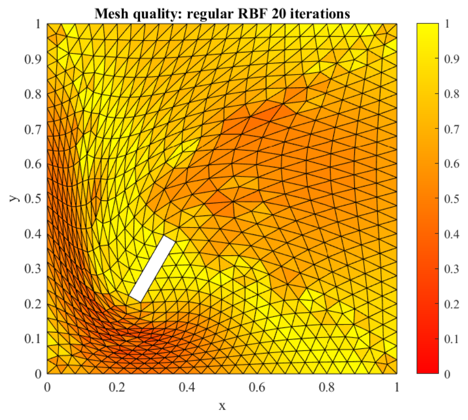


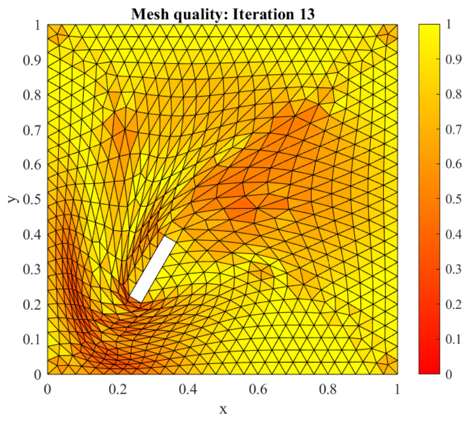
Figure 5.13: Initial mesh for the translation with rotation case.



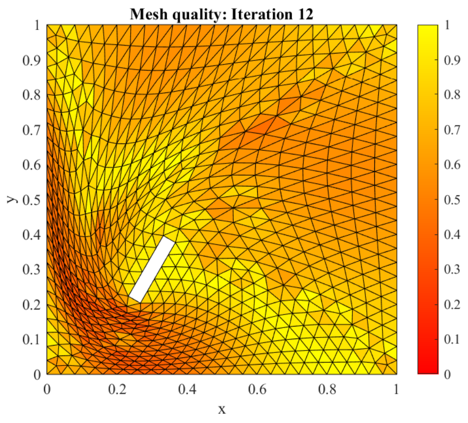
(a) Mesh quality using the regular RBF method: Time-steps = 1, R = 2.5



(b) Mesh quality using the regular RBF method: Time-steps = 20, R = 2.5



(c) Mesh quality using the method of explicit iterations - 13 iterations: R = 0.5, R_l = 0.5



(d) Mesh quality using the method of explicit iterations - 12 iterations: R = 2.5, R_l = 0.5

Figure 5.14: Comparison of mesh quality for regular RBF and method of explicit iterations for various deformation parameters: translation + rotation.

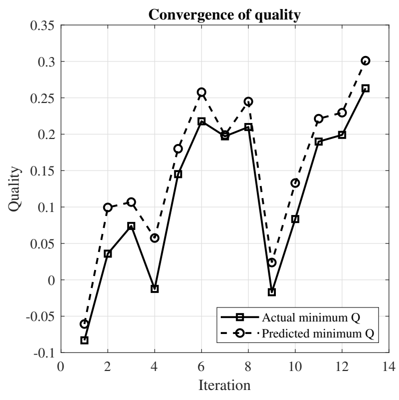
It has to be noted that even though the mesh and deformation parameters are similar for both the methods, there are still some differences in terms of the support radius in use and the number of time-steps utilized to perform the deformation as depicted in Table 5.7.

Method	Deformation Function	Support radius	Time Step	Iterations	Quality
Standard RBF	RBF	2.5	1	-	-0.57
	RBF		20		0.27
Method of explicit iterations	RBF	0.5	1	13	0.27
	Enrichment function	0.5			
	RBF	2.5		12	0.27
	Enrichment function	0.5			

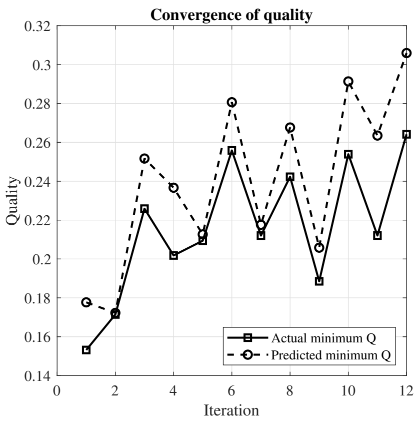
Table 5.7: Differences in input parameters between standard RBF and method of explicit iterations

For the current test case, the threshold minimum quality was raised such that the number of iterations taken by the method of explicit iterations in order to provide the final mesh quality obtained using the regular RBF method when the deformation is performed in 20 time steps could be tracked. The difference in qualities of the final mesh can be seen from Figure 5.14 and Table 5.7. It is observed that when a single time-step deformation is performed for the regular RBF method, it leads to an invalid mesh even while using a global support radius. Whereas if the number of time steps is increased to 20, then the deformation is way smoother and the minimum quality in the mesh is observed to be 0.27. But if a single time-step is used with the method of explicit iteration to correct for the low-quality elements, then it is observed that it takes 13 iterations for the mesh quality to reach the same value as the regular RBF method in 20 iterations. But on the flip side, it is noticeable that the deformation is no longer as smooth. This can be attributed to the low values of the support radius selected for the latter. This was done to see how well the enrichment functions perform whilst simultaneously trying to ensure that the system to be solved does not become too dense. As the support radius of the RBF was increased, the deformation with the explicit iteration was indeed observed to be smoother as seen in Figure 5.14d.

Furthermore, there is no discrepancy observed between the predicted mesh quality and actual mesh quality as shown in Figure 5.15. The locations at which the enrichment control points are added along with the order in which they are added are depicted in Figure 5.16.

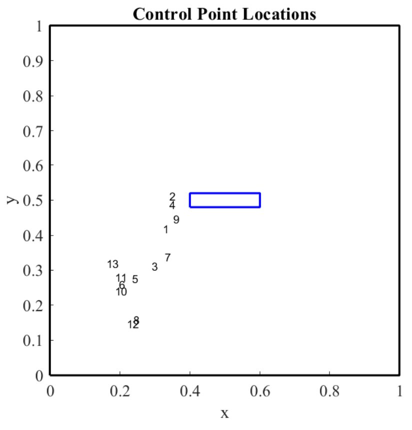


(a) R = 0.5, R_l = 0.5

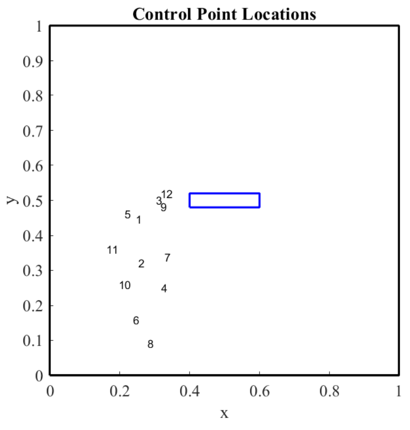


(b) R = 2.5, R_l = 0.5

Figure 5.15: Convergence properties of predicted and actual mesh quality : translation + rotation



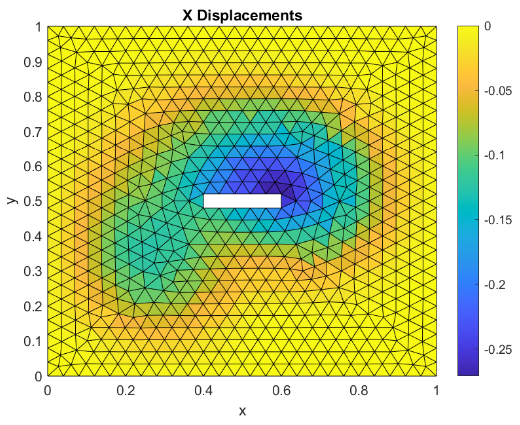
(a) R = 0.5, R_l = 0.5



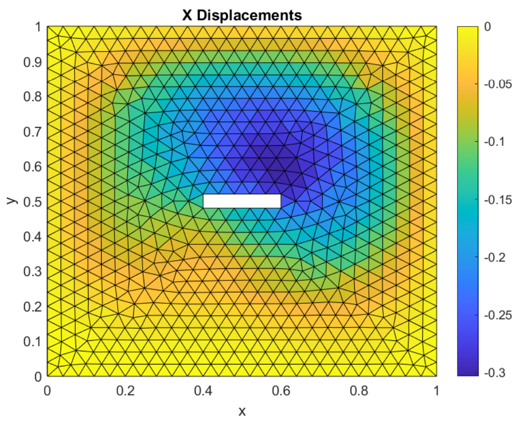
(b) R = 2.5, R_l = 0.5

Figure 5.16: Location of enrichment control points: translation + rotation

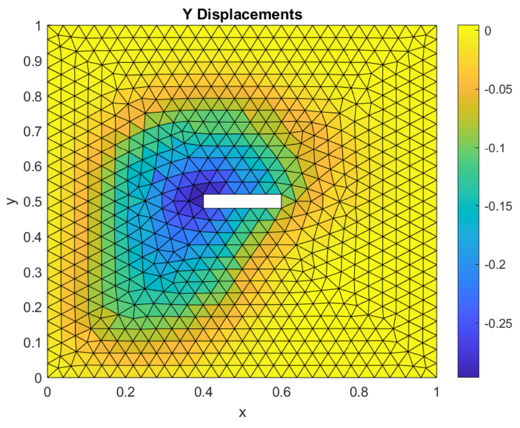
The displacements and the cumulative action of the enrichment functions are depicted in Figure 5.17 and Figure 5.18.



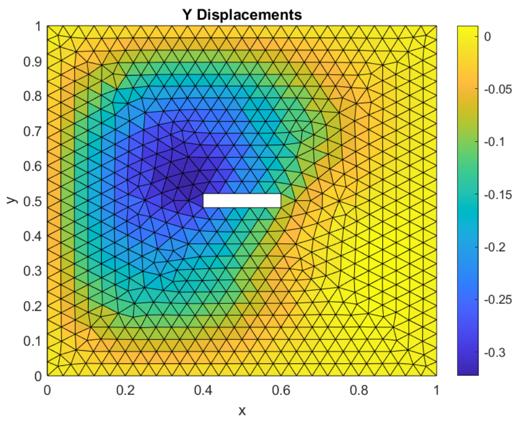
(a) Displacement magnitudes in X direction



(b) Displacement magnitudes in Y direction

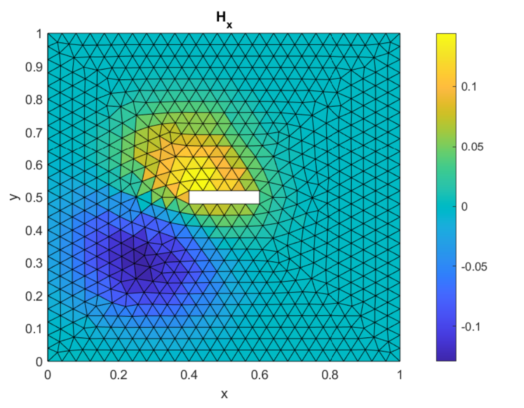


(c) Enrichment function, H_x magnitude

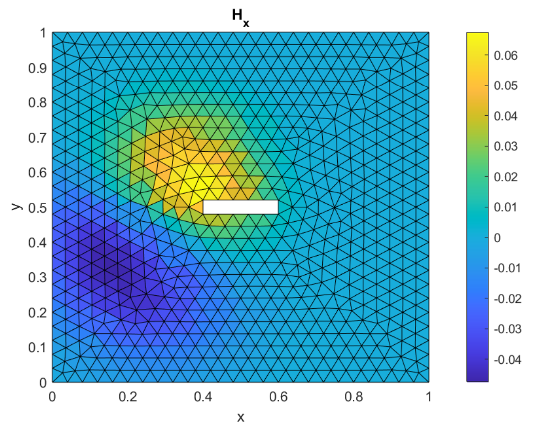


(d) Enrichment function, H_y magnitude

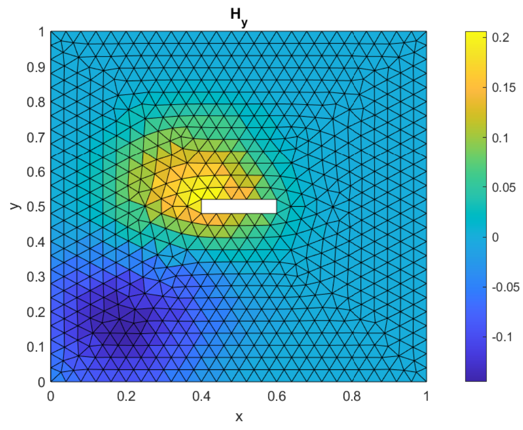
Figure 5.17: Results obtained for the mesh deformation from method of explicit iterations: translation + rotation: $R = 0.5, R_l = 0.5$



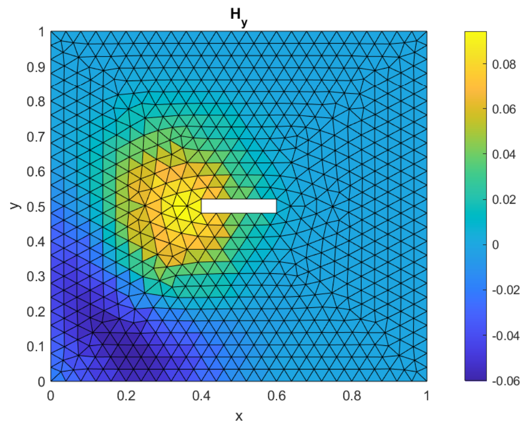
(a) Convergence properties of predicted and actual mesh quality



(b) Locations at which the control points are added



(c) Enrichment function, H_x magnitude



(d) Enrichment function, H_y magnitude

Figure 5.18: Convergence properties and control point locations: translation + rotation: R = 0.5, $R_l = 0.5$

5.5. Discussion

In the current section, the results obtained in this chapter will be summarised. Initially, it was attempted to extrapolate the 1D methodology directly to the 2D setup. Since it was not possible to intuitively predict the optimal gradients to impose, a parametric study was conducted to ascertain a pattern in the ideal gradients that need to be imposed with respect to varying support radius for the enrichment function. But no useful data regarding the gradients could be gathered from the parametric study. Therefore, it was decided to concentrate on the data that was easily available i.e. the gradients at the location of the control point. It was hypothesized that by decomposing the data from these gradients, a logical step could be taken in order to improve the mesh quality. Various matrix decompositions were analysed for this effect and the eigen decomposition was chosen for the current study. Using the eigen decomposition, local corrections were made to the problematic cells such that extreme deformations of the cells were contained. These corrections were performed in an iterative manner such that an enrichment function would be added in each iteration to control the gradient of deformation until the minimum quality exceeds the user-defined threshold or a certain number of iterations is reached.

The methodology was initially tested on an unstructured grid in which the internal block translated along the X direction. it was found that the mesh quality could be considerably improved, but it is possible that if the user-defined threshold is too large or the deformation is too extreme, then the enrichment functions could eventually make the mesh quality worse as well. Therefore, it is necessary to set a reasonable threshold for the algorithm to abort. In the current setup, a limit was set on the number of explicit iterations that could be performed. Furthermore, 2D deformations were also tested where the block was translated along both X and Y directions. Considerable improvement in the mesh quality was observed for a relatively low number of iterations. Finally, another test case available in the literature was chosen as a validation case. The original setup utilised a global RBF system. Therefore, the performance of the method of explicit iterations was analysed while using a compact system such that the computational expense is reduced$(R, R_l = 0.5)$. It was found that the latter achieved a similar level of minimum quality in the domain far more quickly even though the deformation was not as smooth. But if the support radius of the original RBF functions is increased, then the final deformation became smoother as well$(R = 2.5, R_l = 0.5)$.

6 Conclusions and Recommendations

6.1. Conclusions

In the current thesis work, localized enrichment function-based corrections are performed to the mesh deformation algorithm in order to improve the quality of the mesh post-deformation. The corrections are performed whenever the mesh quality falls below a certain threshold. In order to save computational cost, the mesh quality is predicted based on the gradients of the RBF deformation function. It was found that the quality could be predicted to a high degree of accuracy. Grid convergence studies were performed on 1D meshes, and for unstructured and structured 2D meshes. Furthermore, a theoretical framework was provided for expanding the prediction algorithm to 3D meshes in the future. Next, the methodology for the application of the enrichment functions was discussed in detail for a 1D case. In terms of mesh quality in 1D, it was found that the contraction case was most likely susceptible to cell inversions thereby invalidating the domain. Therefore, this setup was chosen to be studied in depth.

It was observed that two critical factors come into play in order to perform the corrections. The first is the location at which the correction has to be performed and the second is the nature of the correction that is performed. Initially, global corrections were performed to identify useful information which could be utilised in the case of local corrections as well. It was found that by imposing a gradient as an additional condition apart from the boundary conditions for the deformation, the mesh quality could be considerably improved. In the case of local corrections, it was quickly hypothesized that the two probable locations where the corrections had to be performed ought to be either the worst quality location or the boundary node location as these are the locations where the most useful data is known. But upon further study, it was found that adding the enrichment function at the worst quality location provided far better results and hence, was chosen for performing further study.

Multiple functions were tested to see their suitability to control the gradient of the deformation at the control point location. It was found that the function provided below performed most optimally in terms of improving the mesh quality as the magnitude of the function itself is zero at the control point, while the gradient is not, thereby making it ideal.

$$\mathbf{H} = (\vec{\mathbf{x}} - \vec{\mathbf{x}}_c) \cdot \phi(\|\vec{\mathbf{x}} - \vec{\mathbf{x}}_c\|, R_l) \tag{6.1}$$

Furthermore, the enrichment functions were utilized to make corrections in 2D. But it was observed that, unlike the 1D case, there was no readily available global gradient information that could be used to impose an additional condition. Therefore, it was deemed necessary to modify the 1D methodology such that instead of imposing the gradients on the RHS, the interpolation coefficients for the enrichment function would be imposed instead. This imposition was based upon the corrections applied to the eigen values of the gradient matrix such that the cell where the control point is being added is always improved in terms of quality. But it was found that it was not enough to perform a single correction as the imposed coefficients are not representative of the ideal deformation.

Hence, the method of explicit iterations was introduced where a control point would be added at the predicted worst quality region after every iteration in order to ensure that threshold quality would be reached after a certain number of iterations. The methodology was showcased to work successfully for different deformations in the 2D domain. Although it was observed that the method did run into problems when the deformation magnitudes were too extreme, which would lead to the addition of control points in the same location multiple times leading to the undermining of the enrichment functions at that location. This would in turn lead to wrong predictions in the mesh quality and hence, the actual worst quality and the predicted worst quality would begin to diverge in terms of both magnitude and location. A comparison of the explicit iteration method in a single time step was also compared with the regular RBF method using multiple time steps and it was found that the method did provide suitable results efficiently.

In conclusion, it is good to now go back and see if the questions raised at the beginning of the project are answered or not:

1. Q. How do the localized enrichment functions help the mesh deformation algorithm perform better in comparison to the regular RBF interpolation-based mesh deformation method?

   Ans. The localized enrichment functions improve the minimum quality of the mesh by a considerable margin by smoothing the gradients of the deformation. They do so by imposing an ideal gradient as a deformation condition as seen in the 1D case or by adding multiple enrichment functions in an iterative manner till quality is improved as seen in the 2D case. There is a significant improvement in quality without too much computational cost.

2. Q. Which mesh quality metric should be utilized for setting the threshold?

   Ans. The mesh quality metric that needs to be imposed is the size metric for both the 1D and 2D cases. For the 1D case, it is the only possibility. For the 2D case, it is also possible to work with size-skew metric, but since the corrections are based on eigen decompositions which give scaling information about the principle axes, it would make sense to consider the area of the cell as the qualifying metric which is given by the determinant. Furthermore, it is possible to predict these values with reasonable accuracy.

3. Q. How much effect will there be on the accuracy of the final displacement due to localized interpolation of displacements?

   Ans. No adverse effect is observed in terms of the local enrichments causing problems with satisfying the boundary conditions. This can be attributed to the fact that for calculating the interpolation coefficient, the only major concern would be inverting the RBF system matrix, (But since there is no polynomial function involved), it is symmetric and non-singular, thereby ensuring accurate boundary displacements.

4. Q. How is the mesh quality and efficiency impacted when control point reduction algorithms are applied to the entire system?

   Ans. This question could not be satisfactorily explored due to time limitations in the current thesis and hence would be recommended for future work.

6.2. Recommendations for Future Work

With regards to future work, there are three key recommendations that are laid out to be explored, which could not be performed due to time constraints:

6.2.1. Optimization of the algorithm

While a proof of concept has been shown with regards to the robustness and accuracy of the local enrichment function method. There is still considerable scope for improvement for optimizing the efficiency of the algorithm. The optimization can be subdivided into two parts:

Mesh quality prediction

In the current thesis, the mesh quality is being predicted for all the cells in the mesh which is a waste of computational resources because most of the time, the poor-quality regions are very localized in the domain. Therefore, a suggestion for improvement in this regard is to use a grouping-based mesh quality prediction algorithm. In such a method, the entire domain would be split into multiple groups where each would comprise randomly selected cells in the domain. Then the mesh quality would be predicted over each group and if the quality observed is less than the threshold value, then the enrichment functions would be added. Then the predictions would be performed on the other groups such that it is ensured that the enrichment function does not worsen the quality of the cells in the other groups. This would also increase the efficiency, as statistically, it is very likely that a problematic region would be identified in a few trials unless the size of the group is made too small. Therefore, a study should also be performed in order to find what would the ideal size of the groups be such that the accuracy and efficiency of the predictions are optimized.

Control point reduction

Furthermore, the current thesis utilizes all the boundary nodes to perform the deformation step. This is not necessary as there are many control point reduction algorithms available in literature which allow a lesser number of points to be selected whilst maintaining accuracy as discussed in chapter 2.

6.2.2. 3D implementation of the enrichment functions

Another important recommendation for future work is to expand the current methodology to 3D meshes. The methodology for both the mesh quality prediction and the addition of enrichment functions in the domain has already been explained theoretically in the current thesis. What remains to be done is actually implementing the methodology on a 3D test case in order to ensure that everything works smoothly. If not, what challenges are encountered which were not considered in the current theoretical framework need to be explored.

6.2.3. Evaluation of shear/rotation data from gradients

The current thesis does introduce various matrix decompositions in order to gain a geometrical understanding utilizing the gradient matrix of a cell in the mesh. But most of these decompositions return some anomalous properties when concerning rotations and shears which is not entirely consistent with what can be seen on the mesh. Decompositions such as the SVD, QR and polar have found wide use in preserving geometrical features of matrix animations, but these methods mostly involve the usage of the Jacobian of the cell i.e. the nodal data is readily available at the initial and final stages. What would be an interesting area to research would be to see if a matrix decomposition can somehow be tweaked such that based on the gradient matrix as the input, it would be able to provide some sort of insight into how the mesh will deform.

Bibliography

[1] Marc Alexa, Daniel Cohen-Or, and David Levin. “As-rigid-as-possible shape interpolation”. In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH ’00* (2000). DOI: 10.1145/344779.344859. URL: http://dx.doi.org/10.1145/344779.344859.

[2] Alessandra M. Bavo et al. “Fluid-Structure Interaction Simulation of Prosthetic Aortic Valves: Comparison between Immersed Boundary and Arbitrary Lagrangian-Eulerian Techniques for the Mesh Representation”. In: *PLOS ONE* 11.4 (2016), e0154517. DOI: 10.1371/journal.pone.0154517.

[3] Y. Bazilevs, M.-C. Hsu, and M.A. Scott. “Isogeometric fluid–structure interaction analysis with emphasis on non-matching discretizations, and with application to wind turbines”. In: *Computer Methods in Applied Mechanics and Engineering* 249 (2012), pp. 28–41. DOI: 10.1016/j.cma.2012.03.028.

[4] Y. Bazilevs et al. “Isogeometric Fluid–structure Interaction Analysis with Applications to Arterial Blood Flow”. In: *Computational Mechanics* 38.4-5 (2006), pp. 310–322. DOI: 10.1007/s00466-006-0084-3.

[5] Armin Beckert and Holger Wendland. “Multivariate interpolation for fluid-structure-interaction problems using radial basis functions”. In: *Aerospace Science and Technology* 5.2 (2001), pp. 125–134. DOI: 10.1016/s1270-9638(00)01087-7.

[6] K. Y. R. Billah and Robert H. Scanlan. “Resonance, Tacoma Narrows bridge failure, and undergraduate physics textbooks”. In: *American Journal of Physics* 59 (1991), pp. 118–124.

[7] A. de Boer, M.S. van der Schoot, and H. Bijl. “Mesh deformation based on radial basis function interpolation”. In: *Computers & Structures* 85.11-14 (2007), pp. 784–795. DOI: 10.1016/j.compstruc.2007.01.013.

[8] Frank M. Bos, Bas W. van Oudheusden, and Hester Bijl. “Radial basis function based mesh deformation applied to simulation of flow around flapping wings”. In: *Computers & Fluids* 79 (2013), pp. 167–177. DOI: 10.1016/j.compfluid.2013.02.004.

[9] Martin Buhmann. *Radial Basis Functions: Theory and Implementations*. Cambridge University Press, 2009.

[10] C. Carstensen and Jun Hu. “A unifying theory of a posteriori error control for nonconforming finite element methods”. In: *Numerische Mathematik* 107.3 (2007), pp. 473–502. DOI: 10.1007/s00211-007-0068-z.

[11] Eric Darve. “The Fast Multipole Method: Numerical Implementation”. In: *Journal of Computational Physics* 160.1 (2000), pp. 195–240. DOI: 10.1006/jcph.2000.6451.

[12] J. De Hart et al. “A computational fluid-structure interaction analysis of a fiber-reinforced stentless aortic valve”. In: *Journal of Biomechanics* 36.5 (2003), pp. 699–712. DOI: 10.1016/s0021-9290(02)00448-7.

[13] C. Delaporte, S. Jawahar, and B. Tallapragada. “Fluid-Structure Interaction: Assignment 1”. In: *AE4117 Course Assignment: TU Delft* (2021).

[14] N. Diniz dos Santos, J.-F. Gerbeau, and J.-F. Bourgat. “A partitioned fluid–structure algorithm for elastic thin valves with contact”. In: *Computer Methods in Applied Mechanics and Engineering* 197.19-20 (2008), pp. 1750–1761. DOI: 10.1016/j.cma.2007.03.019.

[15] J. Donea, S. Giuliani, and J. P. Halleux. “An arbitrary lagrangian-eulerian finite element method for transient dynamic fluid-structure interactions”. In: *Computer Methods in Applied Mechanics and Engineering* 33 (1-3 Sept. 1982), pp. 689–723. ISSN: 0045-7825. DOI: 10.1016/0045-7825(82)90128-1.

[16] MB-Editor. *Skewness Of Mesh Structures In ANSYS® Meshing(Illustrated Expression)*. Jan. 2022. URL: https://mechanicalbase.com/skewness-of-mesh-structures-in-ansys-meshing/.

[17] O. Estruch et al. “A parallel radial basis function interpolation method for unstructured dynamic meshes”. In: *Computers & Fluids* 80 (2013), pp. 44–54. DOI: 10.1016/j.compfluid.2012.06.015.

[18] Hong Fang et al. “An efficient radial basis functions mesh deformation with greedy algorithm based on recurrence Choleskey decomposition and parallel computing”. In: *Journal of Computational Physics* 377 (2019), pp. 183–199. DOI: 10.1016/j.jcp.2018.10.029.

[19] Hong Fang et al. “Efficient mesh deformation based on Cartesian background mesh”. In: *Computers & Mathematics with Applications* 73.1 (2017), pp. 71–86. DOI: 10.1016/j.camwa.2016.10.023.

[20] Hong Fang et al. “Efficient mesh deformation using radial basis functions with a grouping-circular-based greedy algorithm”. In: *Journal of Computational Physics* 433 (2021), p. 110200. DOI: 10.1016/j.jcp.2021.110200.

[21] Michael S. Floater and Armin Iske. “Multistep scattered data interpolation using compactly supported radial basis functions”. In: *Journal of Computational and Applied Mathematics* 73.1-2 (1996), pp. 65–78. DOI: 10.1016/0377-0427(96)00035-0.

[22] Richard Franke. “Scattered Data Interpolation: Tests of Some Method”. In: *Mathematics of Computation* 38.157 (1982), p. 181. DOI: 10.2307/2007474.

[23] Xiang Gao et al. “Efficient and Robust Parallel Mesh Motion Solver Using Radial Basis Functions”. In: *Journal of Aerospace Engineering* 31.3 (2018), p. 04018019. DOI: 10.1061/(asce)as.1943-5525.0000874.

[24] T. Gillebaart. *Towards Efficient Fluid-Structure-Control Interaction for Smart Rotors*. Tech. rep. PhD Thesis. DOI: 10.4233/uuid:c078909a-a39c-47b5-8ca6-2cbc9e04486e.

[25] S. Haeri and J. S. Shrimpton. “On the application of immersed boundary, fictitious domain and body-conformal mesh methods to many particle multiphase flows”. In: *International Journal of Multiphase Flow* 40 (Apr. 2012), pp. 38–55. ISSN: 0301-9322. DOI: 10.1016/J.IJMULTIPHASEFLOW.2011.12.002.

[26] Thomas J.R. Hughes, Wing Kam Liu, and Thomas K. Zimmermann. “Lagrangian-Eulerian finite element formulation for incompressible viscous flows”. In: *Computer Methods in Applied Mechanics and Engineering* 29 (3 Dec. 1981), pp. 329–349. ISSN: 0045-7825. DOI: 10.1016/0045-7825(81)90049-9.

[27] S. Jakobsson and O. Amoignon. “Mesh deformation using radial basis functions for gradient-based aerodynamic shape optimization”. In: *Computers & Fluids* 36.6 (2007), pp. 1119–1136. DOI: 10.1016/j.compfluid.2006.11.002.

[28] L. Kedward, C.B. Allen, and T.C.S. Rendall. “Efficient and exact mesh deformation using multiscale RBF interpolation”. In: *Journal of Computational Physics* 345 (2017), pp. 732–751. DOI: 10.1016/j.jcp.2017.05.042.

[29] Patrick M. Knupp. “Algebraic mesh quality metrics for unstructured initial meshes”. In: *Finite Elements in Analysis and Design* 39.3 (2003), pp. 217–241. DOI: 10.1016/s0168-874x(02)00070-7.

[30] Ch.-N. Li et al. “An efficient multiple point selection study for mesh deformation using radial basis functions”. In: *Aerospace Science and Technology* 71 (2017), pp. 580–591. DOI: 10.1016/j.ast.2017.09.047.

[31] Raphaël Loubère et al. “ReALE: A reconnection-based arbitrary-Lagrangian–Eulerian method”. In: *Journal of Computational Physics* 229.12 (2010), pp. 4724–4761. DOI: 10.1016/j.jcp.2010.03.011.

[32] L.G. Margolin and Mikhail Shashkov. “Second-order sign-preserving conservative interpolation (remapping) on general grids”. In: *Journal of Computational Physics* 184.1 (2003), pp. 266–298. DOI: 10.1016/s0021-9991(02)00033-5.

[33] M Mathew and A van Zuijlen. “Mesh Deformation Using Radial Basis Function Interpolation With Sliding Boundary Nodes”. MA thesis. Aug. 2019. URL: http://resolver.tudelft.nl/uuid:75f3e25a-b991-4e85-ab8a-736a32383bb5.

[34] Andreas K. Michler. “Aircraft control surface deflection using RBF-based mesh deformation”. In: *International Journal for Numerical Methods in Engineering* 88.10 (2011), pp. 986–1007. DOI: 10.1002/nme.3208.

[35] Myles Morelli, Tommaso Bellosta, and Alberto Guardone. “Efficient radial basis function mesh deformation methods for aircraft icing”. In: *Journal of Computational and Applied Mathematics* 392 (2021), p. 113492. DOI: 10.1016/j.cam.2021.113492.

[36] Julian Panetta. *Polar Decomposition and the Closest Rotation*. Tech. rep. Feb. 2015. URL: https://julianpanetta.com/#notes.

[37] S. Rebouillat and D. Liksonov. “Fluid–structure interaction in partially filled liquid containers: A comparative review of numerical approaches”. In: *Computers & Fluids* 39.5 (2010), pp. 739–746. DOI: 10.1016/j.compfluid.2009.12.010.

[38] T.C.S. Rendall and C.B. Allen. “Efficient mesh motion using radial basis functions with data reduction algorithms”. In: *Journal of Computational Physics* 228.17 (2009), pp. 6231–6249. DOI: 10.1016/j.jcp.2009.05.013.

[39] T.C.S. Rendall and C.B. Allen. “Reduced surface point selection options for efficient mesh deformation using radial basis functions”. In: *Journal of Computational Physics* 229.8 (2010), pp. 2810–2820. DOI: 10.1016/j.jcp.2009.12.006.

[40] Derek Risseeuw. “Fluid Structure Interaction Modelling of Flapping Wings”. PhD thesis. Jan. 2019.

[41] Mohamed M. Selim, Roy P. Koomullil, and Ahmed S. Shehata. “Incremental approach for radial basis functions mesh deformation with greedy algorithm”. In: *Journal of Computational Physics* 340 (2017), pp. 556–574. DOI: 10.1016/j.jcp.2017.03.037.

[42] Ken Shoemake and Tom Duff. “Matrix animation and polar decomposition”. In: *Graphics Interface* (Sept. 1992), pp. 258–264. URL: http://kucg.korea.ac.kr/seminar/2006/src/PA-06-13.pdf.

[43] Giorgos A. Strofylas, Georgios N. Lygidakis, and Ioannis K. Nikolos. “An agglomeration strategy for accelerating RBF-based mesh deformation”. In: *Advances in Engineering Software* 107 (2017), pp. 13–37. DOI: 10.1016/j.advengsoft.2017.02.004.

[44] Gang Wang, Xin Chen, and Zhikan Liu. “Mesh deformation on 3D complex configurations using multistep radial basis functions interpolation”. In: *Chinese Journal of Aeronautics* 31.4 (2018), pp. 660–671. DOI: 10.1016/j.cja.2018.01.028.

[45] Gang Wang et al. “Improved Point Selection Method for Hybrid-Unstructured Mesh Deformation Using Radial Basis Functions”. In: *AIAA Journal* 53.4 (2015), pp. 1016–1025. DOI: 10.2514/1.j053304.

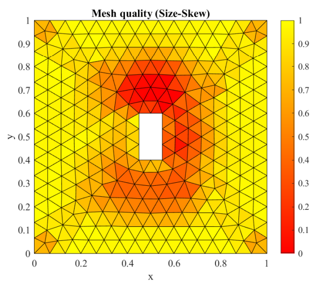
[46] H. Wendland. *Konstruktion und Untersuchung radialer Basisfunktionen mit kompaktem Träger*. 1996. URL: https://books.google.nl/books?id=4AG5HAAACAAJ.

[47] Liang Xie and Hong Liu. “Efficient mesh motion using radial basis functions with volume grid points reduction algorithm”. In: *Journal of Computational Physics* 348 (2017), pp. 401–415. DOI: 10.1016/j.jcp.2017.07.042.

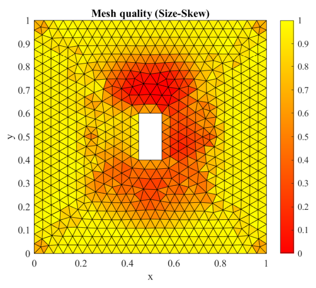
A

Mesh Quality Prediction - Convergence Study

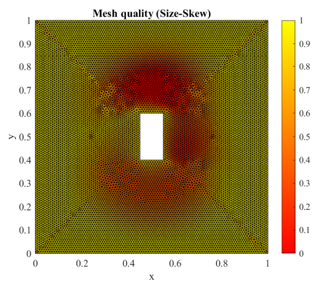
A.1. Trigonal Grid



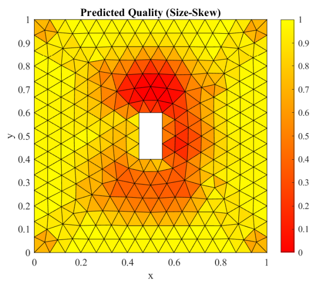
(a) N = 416



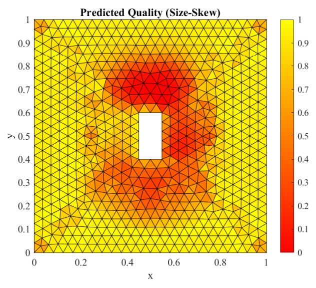
(b) N = 1258



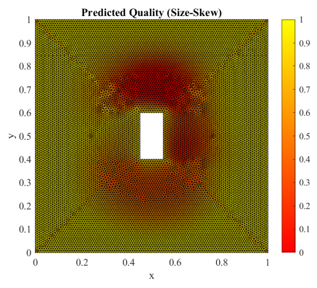
(c) N = 24242



(d) N = 416



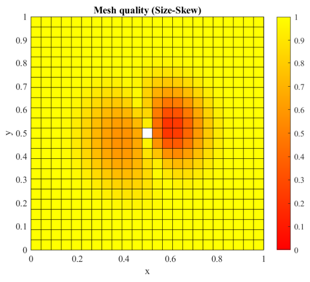
(e) N = 1258



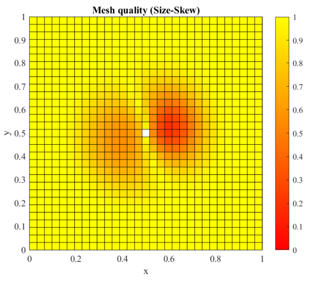
(f) N = 24242

Figure A.1: Comparison of the 2D Mesh quality prediction (d,e,f) with the actual mesh quality (a,b,c) at the $(n+1)^{th}$ time-step. Support radius of RBF, r = 0.4

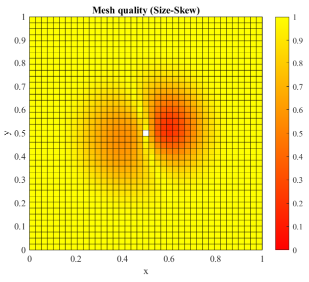
A.2. Quadrilateral Grid



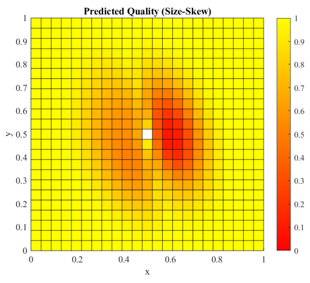
(a) N = 529



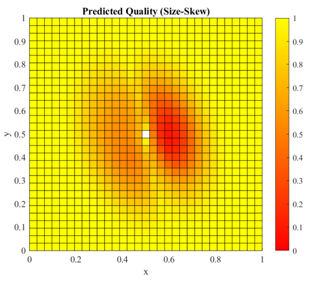
(b) N = 961



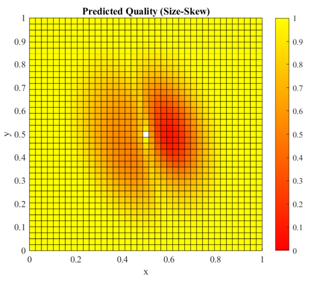
(c) N = 1521



(d) N = 529



(e) N = 961



(f) N = 1521

Figure A.2: Comparison of the 2D Mesh quality prediction (d,e,f) with the actual mesh quality (a,b,c) at the $(n+1)^{th}$ time-step. Support radius of RBF, r = 0.4