ENHANCED CLASSIFICATION APPROACH WITH SEMI-SUPERVISED LEARNING FOR RELIABILITY-BASED SYSTEM DESIGN

A Dissertation
Presented to
The Academic Faculty

by

Jiten Patel

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Mechanical Engineering

Georgia Institute of Technology
August 2012

ENHANCED CLASSIFICATION APPROACH WITH SEMI-SUPERVISED LEARNING FOR RELIABILITY-BASED SYSTEM DESIGN

Approved by:

Dr. Seung-Kyum Choi, Chair
Mechanical Engineering
Georgia Institute of Technology

Dr. David W. Rosen
Mechanical Engineering
Georgia Institute of Technology

Dr. Richard W. Neu
Mechanical Engineering
Georgia Institute of Technology

Dr. Bruce R. Ellingwood
Civil and Environmental Engineering
Georgia Institute of Technology

Dr. Rafi L. Muhanna
Civil and Environmental Engineering
Georgia Institute of Technology

Date Approved: June 29, 2012

ACKNOWLEDGEMENTS

This doctoral dissertation would not have been possible without the help and support of the kind people around me and the excellent research facilities that Georgia Institute of Technology provides.

First of all, I would like to thank my advisor Dr. Seung-Kyum Choi without whose guidance, support and encouragement I could not have grown immensely as a researcher and individual over the last few years. He has always emphasized high quality results in research, presentations, and written documents as well as in my graduate course work. He has always encouraged me to put in my best in everything and expect the best results in the end.

I would also like to thank other members of my dissertation reading committee, Dr. David Rosen (ME), Dr. Rafi Muhanna (CEE), Dr. Rick Neu (ME) and Dr. Bruce Ellingwood (CEE) for taking the time to peruse my dissertation and suggest important improvements. Dr. David Rosen and Dr. Rafi Muhanna also served on my MS thesis committee earlier and their comments have been invaluable in providing the direction for my PhD research.

Georgia Institute of Technology in general and the Woodruff School of Mechanical Engineering in particular offers many courses which were responsible in creating the desire inside me to learn more. The flexible curriculum allowed me to take courses far and wide including courses in business, IP law and policy which helped me grow and become the person I am today. Few of the classes which have helped me understand the core formulations for this PhD dissertation were Computational Data Analysis: Machine Learning and Data Mining (CSE), Regression Analysis (ISYE), Finite

Element Analysis (ME) and Computer-Aided Engineering (ME). I am thankful to the instructors in these classes, Dr. Alexander Gray, Dr. Kwok-Leung Tsui, Dr. Suresh Sitaraman and Dr. David Rosen for providing me a platform where I could learn the core fundamentals upon which this dissertation is built.

I would like to thank all the students, past and present, of MaRC 264 who always volunteered to help me in times of need and encouraged/pushed/directed me in the right direction whenever I hit a wall. Specifically, I would like to thank Dr. Sungshik Yim who has many times contributed to the success of this dissertation indirectly by challenging my hypothesis and asking insightful questions. Dr. Dirk Schaefer has always helped me whenever I have knocked on his door. I am very thankful to him for being always kind and generous to me. My discussions with Dazhong Wu, Amit Jariwala, Patrick Chang, Andrew Hyder, Chenjie Wang, Jason Nguyen, Abhisek Kumar, Markus Rippel, Alex Ruderman, Mukul Singhee and Timothy Dietz have been very helpful and fun. Guys, you are great!

The Woodruff school has generously funded a major portion of my graduate school. The TI:GER NSF Fellowship from the Georgia Tech College of Management and the Sam Nunn Security Program MacArthur Foundation Fellowships have also funded a part of my research. I am grateful to all these sources for their generosity.

I am thankful to Dr. G. K. Ananthasuresh from Indian Institute of Science (IISc) and Dr. Girish Krishnan from University of Michigan for teaching me the core fundamentals of topology optimization and compliant mechanisms design, during my undergraduate years, which form a critical part of this dissertation.

Last but not least I would like to thank my parents for their unconditional love, wishes, patience and faith without which I would not have made it this far. They have inculcated me with the virtues of hard work, dedication, perseverance and belief in God since my early days. I exist because of them.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS **iii**

LIST OF FIGURES **x**

LIST OF TABLES **x**

LIST OF SYMBOLS AND ABBREVIATIONS **xv**

SUMMARY **xvii**

CHAPTER 1 INTRODUCTION **1**

1.1 Meso-Scale Structures (Mesostructures) 3

1.2 Macro-scale Structures 6

1.3 Topology Optimization 8

1.4 Uncertainty in Structural Design 12

1.5 Reliability-based Design Optimization 16

1.6 Reliability-based Topology Optimization 19

1.7 Discontinous Responses and Disjoint Failure Domains 20

1.8 Research Questions and Hypothesis 23

1.9 Current Research 26

1.10 Thesis Organization 27

CHAPTER 2 STATE OF THE ART IN STRUCTURAL RELIABILITY **30**

2.1 Methods for Design of Mesostructures 30

2.2 The Michell Analytical Approach 32

2.3 Deterministic Optimization Approach 33

2.3.1 Description of an Optimization Problem 33
2.3.2 Size Optimization 35
2.3.3 Shape Optimization 36
2.3.4 Topology Optimization 37
2.3.5 Topology Optimization of Truss Structures 41
2. 4 Reliability- based Optimization Approach 44

2.4.1 Formulation of RBTO 46
2.4.3 Stochastic Optimization 49
2.4.4 Structural Reliability Assessment 50
2.5 Sampling Methods 53

2.5.1 Monte Carlo Simulation 53
2.5.2 Latin Hypercube Sampling 55
2.5.3 Probability of Failure Calculation 56
2.6 Surrogate Modeling Techniques 58

2.6.1 Function Approximation 61
2.6.2 Regression 64
2.6.3 Moving Least Squares Method for Function Approximation 66
2.6.4 Classification 71
2.6.5 Artificial Neural Networks (ANN) 78
2.7 Summary 91

CHAPTER 3 MACHINE LEARNING FOR SURROGATE MODELING 93

3.1 Machine Learning 93

3.2 Supervised Learning 95

3.2.1 Function Approximation as Supervised Learning for Reliability Estimation 96
3.2.2 Classification as Supervised Learning for Reliability Estimation 98
3.2.3 Artificial Neural Network as a Superior Machine Learning Method 100

3.2.4 Classification using Probabilistic Neural Networks (PNN) 107
3.2.5 Limitations of Probabilistic Neural Networks (PNN) 114
3.3 Unsupervised Learning 118
3.3.1 Clustering as Unsupervised Learning 119
3.3.2 Maximum Likelihood Estimates for Mixture Models 121
3.3.3 Maximum Likelihood Estimates for Gaussian Mixtures 125
3.3.4 Clustering using Expectation Maximization (EM) Algorithm 130
3.4 Summary 133

CHAPTER 4 SEMI-SUPERVISED LEARNING FOR RELIABILITY ASSESSMENT 135

4.1 Semi-Supervised Learning (SSL) 135
4.2 Limitations of Probabilistic Neural Network 137
4.3 Mixture of Gaussians and Expectation-Maximization for SSL-1 142
4.4 SSL-1 for Automatic Selection of Smoothing Parameter of PNN 147
4.4.1 EM-like Algorithmic Framework for Training a SSL-1 based PNN 147
4.4.2 Proposed Framework for Reliability Estimation using PNN and EM 150
4.5 SSL-2 for Considering 'full' Covariance with PNN 152
4.6 Summary 162

CHAPTER 5 VALIDATION EXAMPLES 165

5.1 Continuous Problem in Two Dimensions 166
5.2 Disjoint Problem in Two Dimensions 169
5.2.1 Analysis Process 170
5.2.2 Comparison of SSL-2 with PNN 172

5.2.3 Performance of SSL-2 as Number of Unlabeled Data is Changed 175
5.2.4 Performance of SSL-2 as Number of Labeled Data is Changed 177
5.2.5 Comparison of SSL-2 with SVM 179
5.3 Ten Bar Truss Example 181

5.3.1 Comparison of PNN and SSL-1 183
5.3.2 Comparison of PNN and SSL-2 186
5.3.3 Example with Non-Gaussian Random Variables 189
5.4 Compliant Meso-Scale Gripper Mechanism Design for Biological Applications 191

5.4.1 Optimal Design of Compliant Mechanisms 192
5.4.2 Design of Complaint Gripper Mechanism for Biological Application 196
5.4.3 Deterministic and Reliability-based Topology Optimization Design Results 201
5.5 Summary 210

CHAPTER 6 CONCLUSION AND FUTURE WORK 211

6.1 Contributions 211

6.1.1 Extended Summary 211
6.1.2 List of Contributions and Explanation 215
6.2 Limitations 218

6.3 Future Work 221

REFERENCES 226

LIST OF TABLES

Table 1. 1 Organization of the dissertation 27

Table 3. 1Expectation Maximization Algorithm for Clustering of Unlabeled data.. …..131

Table 4. 1 EM algorithmic framework for SSL-1 [116]…….. ……..146

Table 4. 2 EM-like algorithmic framework for SSL-1 using PNN..... 148

Table 4. 3 EM algorithm for Gaussian mixture clustering 154

Table 5. 1 Confusion matrix for comparison between the conventional PNN and PNN with SSL……... ……..169

Table 5. 2 Labeled data sampled for training 171

Table 5. 3 Confusion matrix for comparison of the original PNN and SSL-2 174

Table 5. 4 No. of misclassifications when 10 labeled points are used and the number of unlabeled points is increased for SSL-2..... 176

Table 5. 5 Change in accuracy of SSL-2 as number of labeled data points is increased 178

Table 5. 6 Comparison of SSL-2 with PNN and SVM..... 180

Table 5. 8 Probability of failure values for tip displacement at node 2 using SSL-1 184

Table 5. 9 Probability of failure values for stress on truss member 3 using SSL-1..... 184

Table 5. 10 Probability of failure values for stress on truss member 4 using SSL-1..... 185

Table 5. 11 Probability of failure values for tip displacement at node 2 using SSL-2 ... 186

Table 5. 12 Probability of failure values for stress on truss member 3 using SSL-2..... 187

Table 5. 13 Probability of failure values for stress on truss member 4 using SSL-2..... 187

Table 5. 14 SSE comparison between PNN, SSL-1 and SSL-2 189

Table 5. 15 Uncertain variables and corresponding distributions..... 190

Table 5. 16 Probability of failure values for different displacement limit states..... 190

LIST OF FIGURES

Figure 1. 1 Cellular material structures (a) Natural (b) Artificial 4

Figure 1. 2 A non-intuitive input and output requirement satisfying compliant mechanism by Yin and Ananthasuresh 5

Figure 1. 3 Hoberman ball-popular toy 6

Figure 1. 4 Planer Hoberman mechanism [15] 7

Figure 1. 5 HALSA satellite launched by Japan in 1997 [16] 8

Figure 1. 6 Material structure as an arrangement of material and void 9

Figure 1. 7 Representation of building block with 5 design variables 10

Figure 1. 8 Taxonomy of reliability assessment methods 18

Figure 1. 9 Continuous failure domain example- suitable for regression approach 20

Figure 1. 10 Example of a disjoint failure domain limit state 22

Figure 2. 1 Unit cell approach to mesostructure analysis…….. ……31

Figure 2. 2 Michell truss design example from Ref.[55] 32

Figure 2. 3 The optimization procedure 34

Figure 2. 4 Size optimization of beam with six elements 36

Figure 2. 5 Shape optimization of beam with eight control points 37

Figure 2. 6 Topology optimization of beam using density design variable 38

Figure 2. 7 Topology optimization using area of cross-section design variable 40

Figure 2. 8 Initial design space discretization (a) Groundtruss (b) Unit cell 42

Figure 2. 9 Difference between reliability and robustness 45

Figure 2. 10 Reliability-based topology optimization procedure 48

Figure 2. 12 Basic concept of LHS: Two variables and five realizations 56

Figure 2. 13 Example of Probability of Failure Calculation using LHS 58

Figure 2. 14 RBTO procedure using sampling scheme 59

Figure 2. 15 RBTO procedure using a surrogate model and sampling scheme..... 61

Figure 2. 16 Moving least square approximation process[42]..... 68

Figure 2. 17 Weight functions 71

Figure 2. 18 Linearly separable and linearly non-separable hyperplanes respectively 75

Figure 2. 19 A simple neuron model with *n* inputs..... 80

Figure 2. 20 Linear transfer function 81

Figure 2. 21 Hard-limit transfer function..... 82

Figure 2. 22 Sigmoid transfer function 82

Figure 2. 23 A fully connected ANN configuration 84

Figure 2. 24 Internal connection between two layers of neurons 85

Figure 2. 25 Classification approach to probability of failure calculation 90

Figure 2. 26 Comparison of surrogate modeling techniques for reliability estimation 91

Figure 3. 1 Typical examples of US Zip Codes written by hand [101]……94

Figure 3. 2 Estimation of response function *f(x1)*..... 98

Figure 3. 3 Estimation of decision boundary *g(x1, x2) = 0* 100

Figure 3. 4 Comparison of various machine learning algorithms[41] 101

Figure 3. 5 Architecture of single hidden layer, feed-forward neural network 103

Figure 3. 6 Illustration of probabilistic classification process 104

Figure 3. 7 Magnified shapes of PDFs of both classes 106

Figure 3. 8 Architecture of probabilistic neural network..... 110

Figure 3. 9 Pattern Layer of PNN 111

Figure 3. 10 Summation layer of PNN 112

Figure 3. 11 Effect of different values of smoothing parameter on $f_A(x)$ in a one dimensional problem[106] 115

Figure 3. 12 Effect of different smoothing parameter values on PNN's performance[109] 116

Figure 3. 13 Unlabeled data is separated into two constituent Gaussian components after unsupervised learning 118

Figure 3. 14 Duda et. al. example on convergence characteristics of MLE [94]..... 123

Figure 4. 1 Architecture of Probabilistic Neural Networks……..…… ……..139

Figure 4. 2 Proposed framework for reliability estimation using PNN and EM 152

Figure 4. 3 Illustration of classification problem with Y axis as true classifier 156

Figure 4. 4 Illustration of four clusters created using EM 157

Figure 4. 5 Resulting decision space after Step 3 159

Figure 4. 6 Spherical Gaussian Kernels on labeled points from Figure 4. 3 160

Figure 4. 7 Classification boundary estimated by PNN algorithm 161

Figure 4. 8 Classification boundary estimated by the SSL-2 algorithm 162

Figure 5. 1 Data points classified into safe and unsafe regions…….. ……..... ….168

Figure 5. 2 True classification boundary 170

Figure 5. 3 Estimated classification boundary by PNN 172

Figure 5. 4 Estimated classification boundary with SSL-2 173

Figure 5. 5 Number of misclassifications with SSL-2 as no. of unlabeled data is varied 175

Figure 5. 6 Increase in accuracy as the number of labeled data is increased..... 179

Figure 5. 7 No. of misclassifications for SSL-2, PNN and SVM 181

Figure 5. 8 Ten-bar truss structure 182

Figure 5. 9 Classification boundary with displacement limit state function..... 188

Figure 5. 10 Design domain and problem specification for a compliant mechanism with input at P_1 and output at P_2 194

Figure 5. 11 Full Groundtruss with input and desired output 199

Figure 5. 12 Reduced Groundtruss that is considered for Topology Optimization 200

Figure 5. 13 Reduced groundtruss solution for gripper mechanism 202

Figure 5. 14 Wireframe representation of the deterministic solution 203

Figure 5. 15 Undeformed and deformed plot with displacement contours from ABAQUS using linear FEM..... 204

Figure 5. 16 Contours for the axial forces on deformed plot of gripper mechanism..... 204

Figure 5. 17 Reduced groundtruss solution for gripper mechanism for RBTO case..... 207

Figure 5. 18 Wireframe solution obtained from RBTO..... 208

Figure 5. 19 Undeformed and deformed plot with displacement contours from ABAQUS using linear FEM for RBTO 208

Figure 5. 20 Contours for the axial forces on deformed plot of gripper mechanism found using RBTO 209

LIST OF SYMBOLS AND ABBREVIATIONS

A_i = Area of cross section of each truss element

t = Thickness of elements

wid = Width of elements

d_i = Distance from a sampling point

$g_j(.)$ = Limit state function

K = Stiffness matrix of the structure

L_i = Length of each truss element

P = Internal pressure of the tank

r_i = Domain influence factor

$u(.)$ = Displacements of nodes

V^* = Final volume of the structure

$W(.)$ = Weight matrix

β_j = Regression coefficient

e = Error of the model equation

ξ_i = Gaussian random variable

$[X]_{nXm}$ = Matrix of input variables with n data points- each containing m dimensions

D = Set of data points

Y = Vector containing dependent variables (targets) for regression

y = Continuous label for function approximation problems

W = Vector containing dependent variables (targets) for classification

w = Class label

Σ	=	Covariance Matrix
σ^2	=	Variance
C	=	Denotes the current classifier
H	=	Vector of hidden labels w_i's
l	=	Number of labeled data points
u	=	Number of unlabeled data points
μ	=	Mean vector
γ	=	Posterior probability
Z	=	Number of classes or clusters
$\delta_z(x)$	=	Discriminant Function
$P(.)$	=	Probability value
$p(.)$	=	Probability density function
COV	=	Coefficient of Variation
s	=	Stress
ε	=	Strain
SE	=	Strain Energy
MSE	=	Mutual Strain Energy
F	=	Force
U	=	Displacements resulting from a force
F_d	=	One unit dummy force at the desired output location
V	=	Displacements resulting from the dummy load

SUMMARY

Traditionally design engineers have used the Factor of Safety method for ensuring that designs do not fail in the field. Access to advanced computational tools and resources have made this process obsolete and new methods to introduce higher levels of reliability in an engineering systems are currently being investigated. However, even though high computational resources are available the computational resources required by reliability analysis procedures leave much to be desired. Furthermore, the regression based surrogate modeling techniques fail when there is discontinuity in the design space, caused by failure mechanisms, when the design is required to perform under severe externalities. Hence, in this research we propose efficient surrogate modeling techniques that will enable accurate estimation of a system's response, even under discontinuity.

In Supervised Machine Learning, surrogate models can be trained with a set of training points for which the corresponding system responses are known. These *labeled* training points are expensive to get since the responses have to be evaluated for a combination of uncertain design variables, either through simulation or through tests. These combinations of uncertain design variables, called *unlabeled* data, are available in plenty since the Probability Distribution Function (PDF) information for the uncertain design variables are assumed to be known. We propose the combination of a few labeled and a large number of unlabeled data in order to construct superior surrogate modeling techniques, which come under the category of *Semi-Supervised Learning*. This superior performance is gained by combining the efficiency of Probabilistic Neural Networks

(PNN) for classification and Expectation-Maximization (EM) algorithm for treating the *unlabeled* data as *labeled* data with hidden labels.

Representative examples will be demonstrated where the proposed algorithms are shown to be effective in cases of linear, non-linear and discontinuous failure domains. Furthermore, the applicability of the proposed algorithms during the conceptual design stages is validated by reliability-based engineering design examples.

CHAPTER 1

INTRODUCTION

The advances in computational abilities have resulted in significant changes in the way complex engineering systems are designed. Development in computing software has made it possible to divide knowledge-based work, distribute it anywhere in the world, have it carried out and integrated again [1]. However, the introduction of these technologies is rapidly increasing the complexity of most engineered systems. Significant difficulties remain in understanding, designing, controlling and anticipating the normal and abnormal behaviors of the complex system. Furthermore, uncertainties in material properties, geometry, manufacturing processes and operational environments are critical at all scales (macro, meso, micro and nano scale). For example, during fabrication processes the typical surface finish and tolerances of geometric accuracy are on the order of tenths of microns during the fabrication processes [2], and the common microfabrication material (such as polycrystalline silicon) has 9~15% variation in its Young's modulus and tensile strength [3]. However, there are some anomalies and difficulties, which restrict the ability to accurately evaluate the systems responses for variable parameters after incorporating realistic descriptions of uncertainty for those parameters in complex engineering systems.

Many times the system's response is smooth and continuous (can be linear as well as highly nonlinear) in which case the system's response can be approximated using a regression based surrogate modeling technique such as Response Surface [4] Method (RSM) or Kriging [5, 6] method. In cases where the responses are not continuous and smooth, regression based methods cannot be used. Furthermore, given the larger scale of

modern complex systems, the traditional surrogate modeling techniques, based on a regression or function approximation approach, tend to utilize large computational resources in order to achieve the desired accuracy levels. As consumers demand for products that are better designed and last longer has increased, designers have to include more design variables in the product and have to consider more uncertain parameters that the product could be exposed to. Hence, even if computational power will be available at relatively cheaper price in the next few years, the increase in demand for complex designs by consumers will require designers to invent new and more accurate surrogate modeling techniques in order to consider design uncertainties.

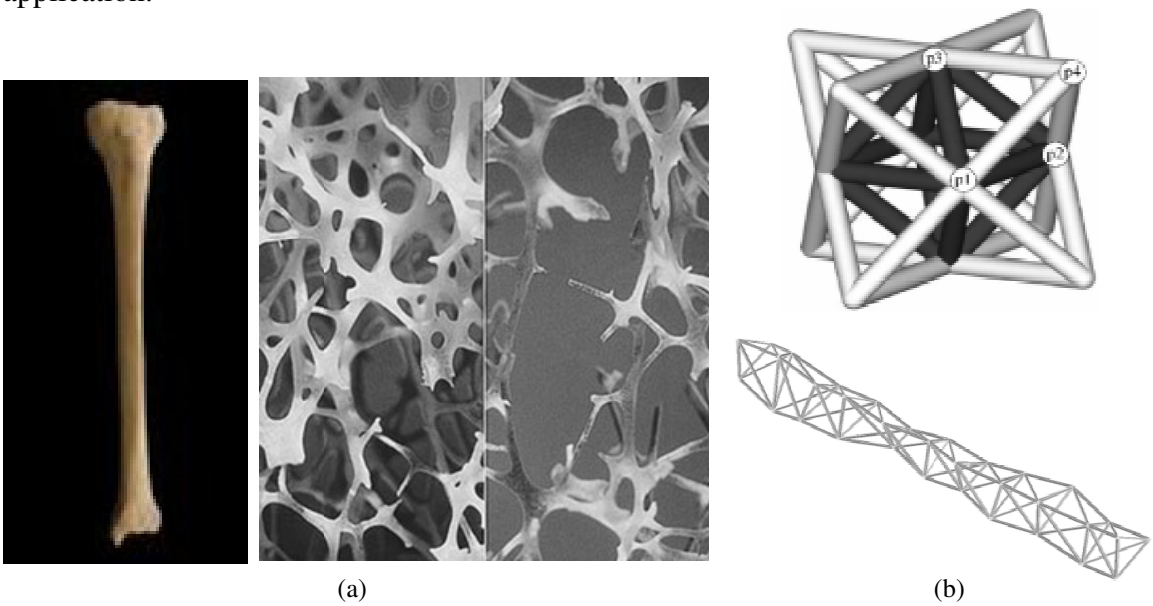
We start by describing typical complex systems which range from meso-scale (Section 1.1) to macro-scale (Section 1.2). In Section 1.3 we describe the Topology Optimization method which has been used extensively in the last few decades for the design of structures and systems at various scales. We describe the origins of uncertainty and typical ways of addressing them in Section 1.4, and in Section 1.5 we describe how uncertainty can be considered during the Topology Optimization process by incorporating reliability quantification in the design process. We discuss the typical problems that are generally encountered in the reliability estimation process in Section 1.6 which leads us to our research questions and hypotheses in Section 1.7. In Section 1.8 we relate the current research questions and hypotheses and discuss how the presented hypotheses could help answer the research questions. We finish this chapter with a discussion of the various chapters in this dissertation, and discuss how they relate to the research questions in Section 1.9.

1.1 Meso-Scale Structures (Mesostructures)

One of the most widespread trends in recent product development has been the copying of nature since nature has designed some of the most highly efficient systems for handling any condition in its environment. These natural systems utilize materials and structures capable of sensing the environment, processing data, responding, and adapting to the given condition. For instance, animal bones have been evolutionally optimized to support various loading conditions with minimum weight. The internal structure of bone can be considered a cellular structure, which can be used to strengthen, stiffen, and even create light-weight parts. The pursuit of engineering cellular materials is biologically inspired as shown in Figure 1. 1. The key advantages offered by cellular materials are high strength, energy absorption characteristics, and improved thermal and acoustic insulation properties accompanied by a relatively low mass. However, the use of advanced novel materials as primary structural elements is still a rarity, particularly in the industrial vehicle arena due to the difficulty with comprehensive understanding of uncertainties in system behavior.

Mesostructure materials are materials that have a characteristic cell length in the range of 0.1 to 10 mm. Small truss structures, honeycombs, and foams are examples of mesostructures [7]. The concept of mesostructured materials is motivated by the desire to put material only where it is needed for a specific application. Additive manufacturing processes are capable of fabricating the complex geometries inherent in cellular materials [8]. With the advancement of additive manufacturing technologies it is now possible to design custom mesostructures which have increased strength and low relative density when compared to the already available mesostructure materials [9]. For example

Seepersad et al. [10], designed the topology of extruded cellular material to find the best compromise between heat transfer and part strength in a structural heat transfer application.



(a) (b)

Figure 1. 1 Cellular material structures (a) Natural (b) Artificial

Instead of modeling these mesostructures as truss elements, if constituting members are modeled as beam elements and the structure is allowed to flex, the structure can be designed to hold certain paths of motion. These structures can then be used as mechanisms, which are called as Compliant Mechanisms [11], which obtain their range of motion by virtue of deflection when a certain magnitude of force is applied in a specified direction. These mechanisms have found particular applications in meso and micro scales in MEMS based applications where they can be used as force sensors and actuators [12]. A particular magnified instance of this mechanism is shown in Figure 1. 2, where an input displacement is magnified 50 times in a direction opposite of the natural intuitive direction of possible motion. Systematic design methods of these mechanisms

become important since it is very difficult to come up with intuitive designs that will give the same input and output relationships as shown in Figure 1. 2.

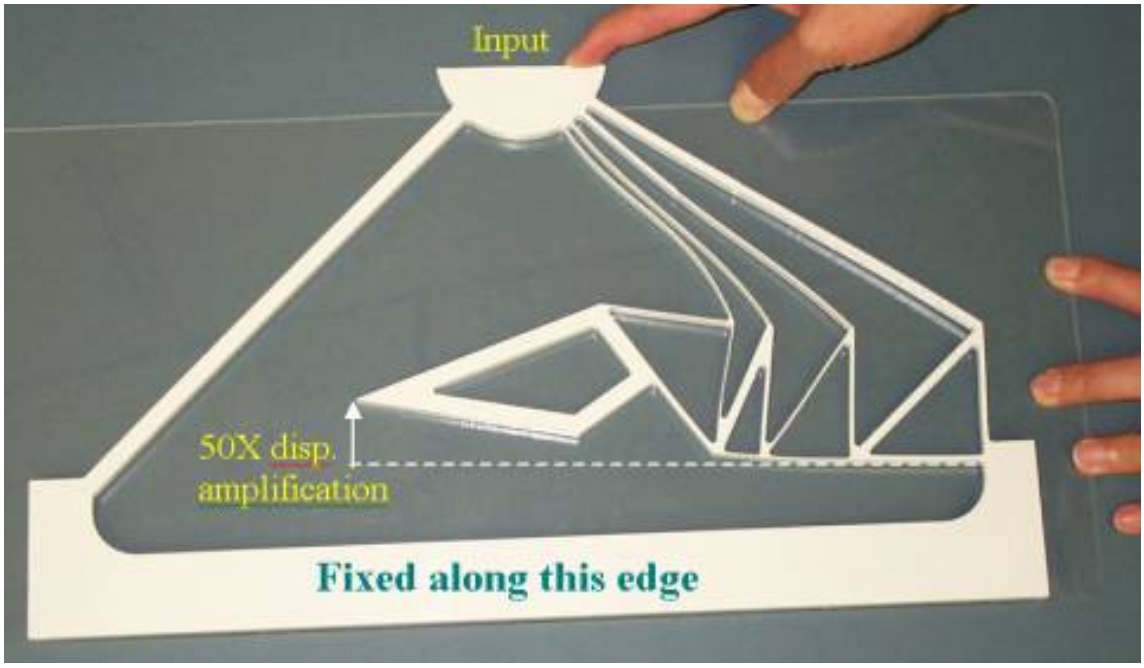


Figure 1. 2 A non-intuitive input and output requirement satisfying compliant mechanism by Yin and Ananthasuresh

The need for systematic design has led to the development of two approaches in the last decade. The first is the *pseudo rigid body model* [13] wherein a compliant mechanism is regarded as an assemblage of rigid links with joint springs. The second is the *topology synthesis method*, which entails generating an appropriate topology within a design region to achieve a single piece continuum with desired deformation and load bearing characteristics.

Topology optimization provides a systematic design method to design mesostructures (structures comprised of trusses) and compliant mechanisms (structures comprised of beams) where even non-intuitive input-output relationships can be realized

in a fixed design space. These structures can also be made to attain a high level of reliability by considering reliability constraints during the topology optimization process.

1.2 Macro-scale Structures

From the simple lazy tongs to the complex deployable space structures, foldable linkages consisting of only rigid bars and revolute joints (hinges) exhibit intriguing motion that is also aesthetically pleasing. Their applications range from consumer products and toys to architectural applications and massive deployable space structures. Two such examples are shown in Figure 1. 3 [14], Figure 1. 4 [15] and Figure 1. 5. In Figure 1. 3, we see the Hoberman's sphere—a popular toy in recent times (www.hoberman.com).

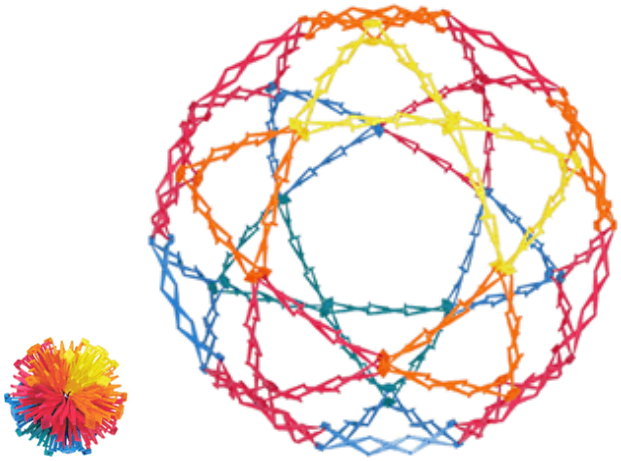


Figure 1. 3 Hoberman ball-popular toy

Its planar version is shown in Figure 1. 4, which has been used for a variety of applications in aerospace applications such as, in the case of deployable antennae mechanisms as shown in Figure 1. 5. These mechanisms have also been used as antennas and stadium roofs all over the world. The common underlying phenomenon in all these

mechanisms is that there are complex interactions between the links of these mechanisms leading to complex coupler curves for the linkages.

Figure 1. 4 Planer Hoberman mechanism [15]

These nonlinear coupler curves result from complex interactions and are more liable to change as a result of unexpected external influences such as foreign particles, changing wind velocities, human interference etc. and internal influences such as material properties, dislocation of the hinge points, etc. Hence, designers should account for these uncertainties in the design process, which may preempt the failures occurring from all these external and internal influences.

One of the common ways to design structures, in all different scales and configurations, is the topology optimization method. In order to account for these uncertainties, one can use a reliability constraint, which ensures the reliability of the system and will maintain a certain level during every iteration of the optimization process.

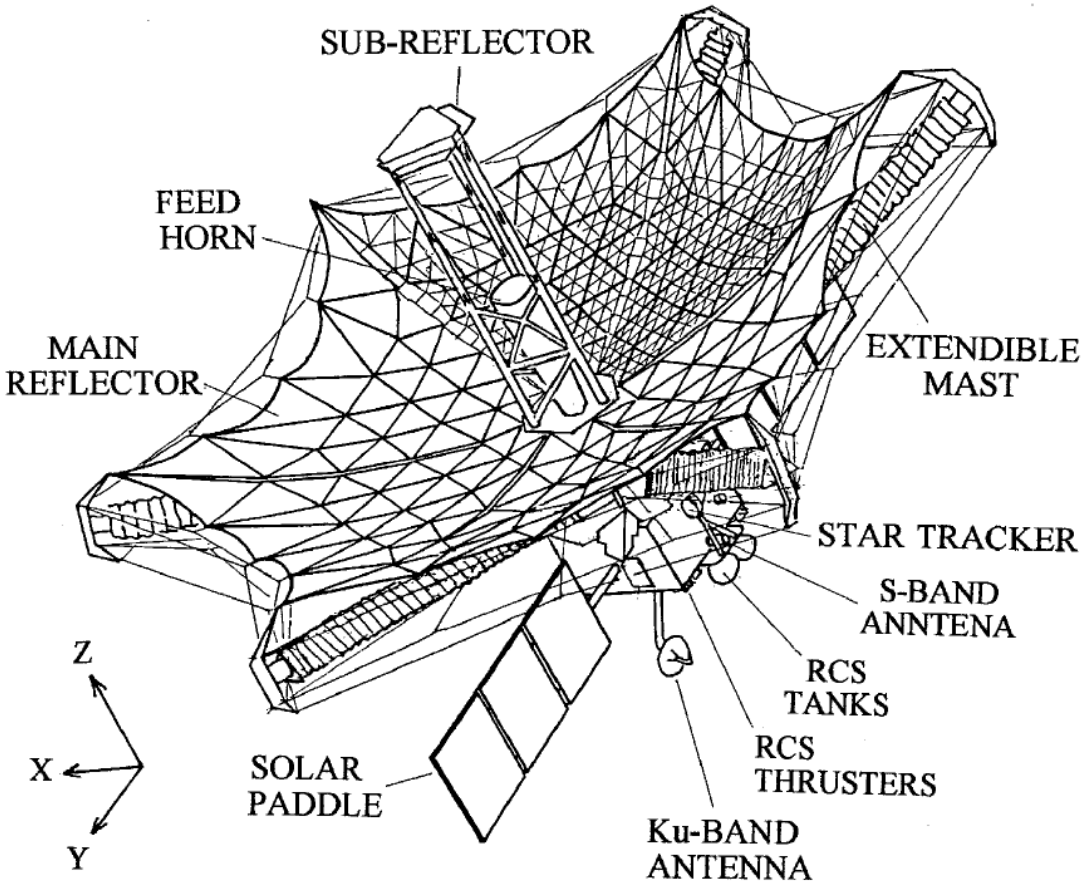


Figure 1. 5 HALSA satellite launched by Japan in 1997 [16]

1.3 Topology Optimization

Topology optimization is often referred to as layout optimization or generalized shape optimization [17]. Topology optimization operates on a fixed mesh of finite elements and defines a design variable, which is associated with each element in the mesh. The stiffest structure problem [18] has been posed as a compliance minimization problem for the design of truss structures. Developments in the computational analysis of structures and components, especially by means of the Finite Element Method (FEM), have made the process of designing specialized truss structures using the topology optimization method

possible. Bendsoe studied optimal shape design as a material distribution problem [19]. This method was adapted by various engineering fields for generating topologies for compliant mechanisms which have maximum displacement at a desired point [20, 21]. Many other applications of topology optimization are considered in the fields of material design for designing materials with prescribed macroscopic properties and recently in the field of biomechanics. In traditional topology optimization methods, it is assumed that the loading is prescribed and that a given amount of structural material is specified within a given 2D or 3D design domain with specified boundary conditions [22].

Research in the field of topology optimization of continuum structures began with the problem of generating optimal topologies in structural design in order to define the stiffest structures, which was explored by Bendsoe and Kikuchi [18]. Their strategy was to define the problem with a composite material represented by each element having material plus a void (hole) inside (Figure 1. 6).

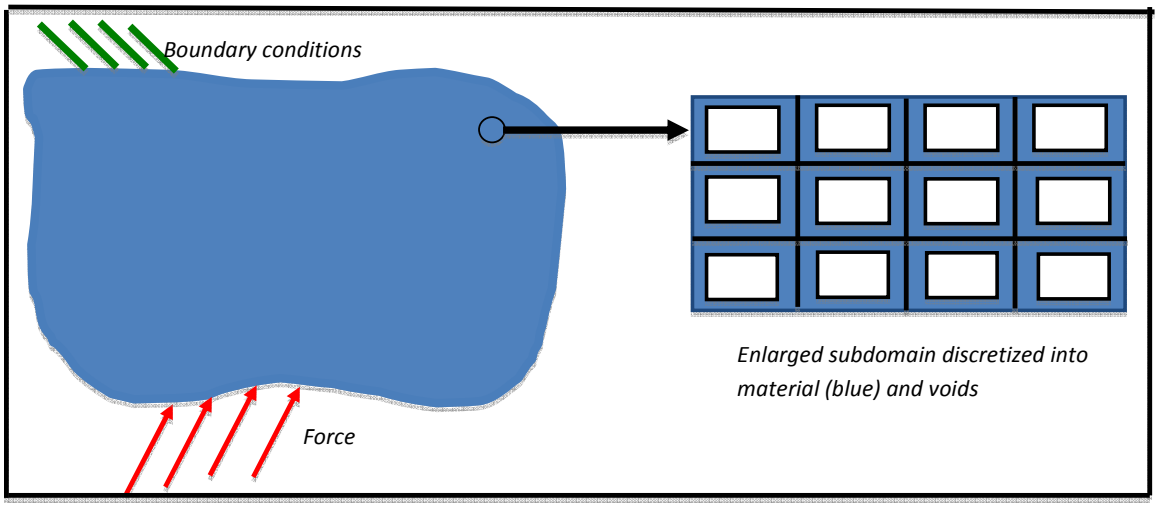


Figure 1. 6 Material structure as an arrangement of material and void

The building blocks can be rectangular in general and also be oriented at a certain angle θ to the horizontal as shown in Figure 1. 7. Here each building block is represented by five design variables namely, W1, W2, L1, L2 and θ. The material properties of each element are then dependent on the size and orientation of the void within the element according to a homogenization relationship. A sizing optimization is then performed to optimize the size/orientations of the voids of all the elements for a given objective function. Elements with large voids (low material density) will represent empty cells and the elements with small voids (high material density) indicate that material exists and hence that cell is a part of the structure. More details of this method can be found in [23].

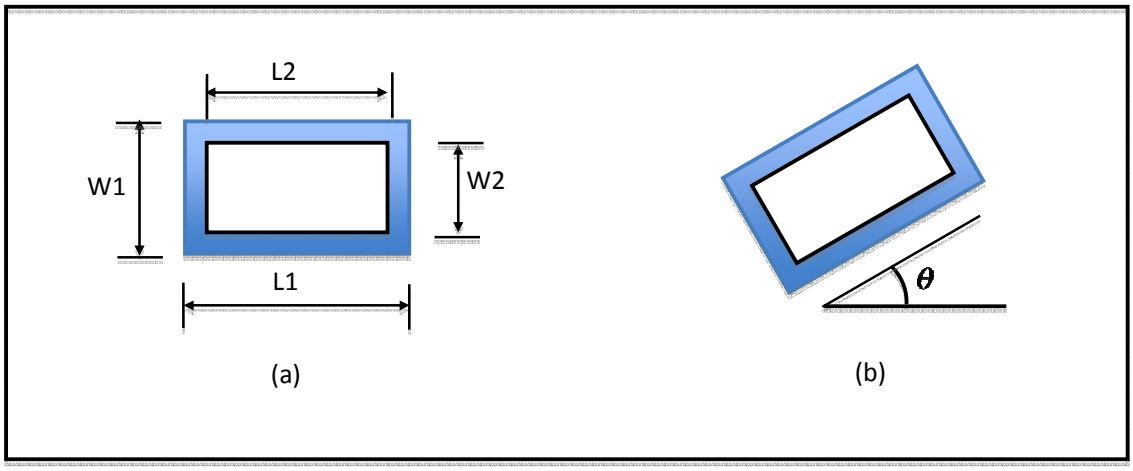


Figure 1. 7 Representation of building block with 5 design variables

An alternative but conceptually similar approach is to directly use the material density of each element (instead of voids) as the design variable. An empirical formula is required in this case instead of using the homogenization formulation. The topology optimization results from this formulation are reported to be similar to those obtained from the homogenization formulation [24].

Strang and Kohn [25] recommended the use of composites in structural optimization problems because the existence/ non-existence of building blocks results in a ill-posed minimization problem, where the optimal solution might be difficult to obtain. To solve this problem they suggest a "relaxation" of the problem where the material in the design domain is modelled as a composite with continously varying density which transforms the original problem into one that has a solution. Hence by modeling the material as a composite and then using material homogenization techniques to determine the composite's structural properties, a minimization problem is created which can be solved by common optimization algorithms.

Optimality criteria methods are typically used to solve the minimization problem created by this "relaxation". Specifically, an iterative redesign procedure modifies the initial design values until the design satisfies a set of optimality criteria. Even though the optimality criteria values are satisfied, there is no guarantee that the design solution is a global optimum. It has been shown that an optimal component design's shape depends upon both the initial material density values and the material microstructure model when using homogenization-based techniques.

Primarily there are two distinctions in topology optimization methods—discrete methods and continous methods. In this dissertation we will focus on the discrete topology optimization method and we explain more details of this method in Chapter 2.

1.4 Uncertainty in Structural Design

Uncertainty is a acknowledged phenomenon in the design processes. During a design optimization process the designer looks for a safe design that has the ability to perform according to the design specifications while it is exposed to various uncertainties. Traditionally safety factors were used to account for the uncertainties. However, use of safety factor does not usually lead to minimum cost designs for a given level of safety because different structural members or different failure modes require different safety factors.

In the traditional sense uncertainty has several connotations such as the degree of belief, lack of knowledge, inaccuracy, variability, etc. An accurate representation of uncertainty is crucial since the different representation of uncertainty may lead to different interpretations for the given system. Primarily the undertainty in a system can be divided into two categories—*Aleatory uncertainty* and *Epistemic uncertainty*. *Aleatory uncertainty* is also known as irreduciable uncertainty or the inherent uncertainty of the system whereas *Epistemic uncertainty* is the uncertainty that stems from lack of knowledge and data and as more information is gathered about the present state of the system the *Epistemic uncertainty* can be reduced. Conventionally, the probability density or frequency information is used to characterize *Aleatory uncertainty* and interval information is used to characterize *Epistemic uncertainty*. By definition, the Probability Density Function (PDF) represents the relative frequency of certain realizations for random variables where the first moment of the PDF indicates the most probably point whereas the tail regions indicate the less probable point.

Recently, *Aleatory uncertainty* has been accounted for by using probabilistic approaches which can give a safer design at a certain computational or experimental cost. These methods give an alternative to the designers who use the traditional safety factor design approach. However, these kind of processes require the statistical parameters for the design at hand which could be expensive to obtain because of either the computational cost of simulations or the cost of conducting failure tests in order to collect labeled data for estimating the parameters of the underlying PDFs. Hence, the probabilisic approaches require solving an expensive, complex optimization problem that needs robust formulations and efficient computational techniques for stable and accelerated convergence.

Probabilistic methods are used in reliability analysis by assuming that the amount of raw data available is sufficient to determine the probability density function and calculate other statistical inputs. However, in practical applications sufficient raw data might not be available due to restrictions in time, human and facility requirements and finances. To handle uncertainty with insufficient information, possibility-based (fuzzy set) methods have been recently introduced in the field of stochastic structural analysis and design optimization [26]. Additionally, Dempster- Shafer theory of evidence [27, 28], random set [29], probability bounds [30-32], imprecise probabilities [33], and convex model [34] are other methods that have been used to describe stochastic uncertainty. All of these methods have a variety of mathematical description although all of them are based on interval analysis [35]. Although the theory of fuzzy sets was introduced by Zadeh [36], the application of interval analysis in structural analysis is very recent. An interval analysis approach utilizing the finite element method was introduced

by Koyluoglu et al. [37] in order to deal with pattern loading and structural uncertainties. Recently, Muhanna and Mullen [38-40] formulated the development of interval based methods for fuzziness in continuum mechanics. These methods help to incorporate uncertain loads in static structural problems using an interval-based fuzzy finite element in the analysis.

In cases when sufficient data is available PDF information for the uncertain variables can be obtained, because of which, *Aleatory uncertainy* can be considered during the design process. Hence *Reliability* of a system can be considered at the conceptual design stages of the design proceses itself giving a confidence that the system will perform its function over a specified period of time and under specified service conditions. Note that, even in these conditions where PDF information is available, *Epistemic uncertainty* still exists since models, by definition, are not exact.

In this dissertation we limit our scope and discussion to the cases where enough data is available to estimate the PDF information for parameters. Hence, only *aleatory* uncertainty will be considered. When aleatory uncertainty is being considered, the behaviour of a structure in structural reliability analysis in probabilistic methods is measured by the performance function. The performance function is called the *limit state function* which is typically expressed as the difference between the *capacity* (e.g.,yield strength, displacement, allowable vibration level) and the demand on the system (e.g., stress, maximum allowable displacement, actual vibration). Note that the capacities and demands on the system will also be functions of the uncertain parameters.

The study of structural reliability is concerned with the calculation and prediction of the probability of limit-state violations at any stage during the structure's life. The probability of the occurance of an event such as a limit-state violation is a numerical measure of the chance of its occurring. Once the probabilty is detemined, the next goal is to choose design alternatives that improve structural reliability and minimize the risk of failure.

Reliability analysis methods can be broadly classified into two categories- analytical methods and simulation methods. While analytical methods are easy to use and are mostly limited to single failure modes, the simulation methods can acess complex limit state functions and can also handle multiple limit states together. Simulation approaches such as, Monte Carlo Simulation (MCS), are computationally intensive but unlike analytical methods which can only handle only linear limit state functions, they can handle any kind of limit state functions. Most real life applications exhibit multiple limit state functions and multiple failure modes and in most cases there is no prior information on the nonlinearity of the limit state function. Simulation based methods like MCS and LHS are the obvious choices in those scenarios. Since reliability analysis is an iterative process and using crude MCS is computationally expensive, researchers develop variants of MCS or other methods like response surface and other function approximation techniques that can replace a part of the reliability analysis computational process and obviate the need to repeatedly access the expensive computer models viz. FEM in case of structural optimization.

1.5 Reliability-based Design Optimization

In deterministic design optimization, design solutions at the boundary of the design constraints are also considered leaving no latitude for variations in the design parameters. The resulting deterministic optimal solution will have an unknown level of safety against the uncertainties that are inherently present during the modeling and manufacturing phases of the product. Uncertainties in simulation-based design are inherently present and need to be accounted for in the design optimization process. Uncertainties may lead to high probability of failure, resulting from large variations in the performance characteristics of the system. Optimized deterministic designs determined without considering uncertainties can be unreliable and might lead to catastrophic failure of the product being designed. *Robust design optimization* and *reliability based design optimization* are methodologies that address these problems. The goal in robust design is to minimize the variations in the performance function. The goal in reliability-based design is to minimize the probability of failure while abiding to other performance constraints that that designer imposes on the design. Hence in order to maintain high market share it is extremely important that designers consider variations in the design of new products and systems so that products are resistant to failure while abiding to the performance requirements.

While using RBDO, the designer has to make a tradeoff between making the design more reliable or minimizing cost. The first step in RBDO is to characterize the important uncertain variables and the failure modes. In most engineering applications, the uncertainty is generally characterized using probability theory. Different statistical models can be used to describe the probability distribution function of the uncertain

variables. In case of Reliability-based Design Optimization (RBDO) the probability density function (PDF) should be known before starting the optimization process. The PDF is used to sample points using a Monte Carlo Simulation or a stratified sampling method, such as Latin Hypercube Sampling Scheme, to simulate uncertain data on the design. The different methods for PDF estimation can be classified as *Parametric*, *Non-Parametric* and *Semi-Parametric*. In Parametric method the PDF is assumed to be of a standard form (Gaussian, Weibull, Beta, etc.). The parameters of the assumed PDF can be estimated using Maximum Likelihood estimation (MLE) or Bayesian Estimation. The Non-Parametric methods include histogram based methods and the K-nearest neighbor methods [41]. In Semi-Parametric methods, the given density can be modeled as a combination of known densities. Mixture of Gaussian (MOG) is a well known method where a data set is assumed to come from different gaussian distributions and has been used for various machine learning applications as well such as *clustering*. The parameters for MOG can then be estimated either by using a gradient descent method or Expectation Maximization (EM) algorithm [41]. The EM algorithm will be explained in greater details in later chapters when we discuss the core of this dissertation.

While designing products with multiple failure modes it is important to justify the safety of the product with respect to each failure mode and also with respect to the overall system failure. In a RBDO formulation, the critical failure modes in deterministic optimization are replaced with constraints on probabilities of failure corresponding to each of the failure driven modes or with a single constraint on the system probability of failure. The reliability index, or the probability of failure corresponding to either a failure mode or the system, can be computed by performing a *probabilistic reliability analysis*.

Some of the techniques used in reliability analysis are the first order reliability method (FORM), second order reliability method (SORM), and Monte Carlo simulation (MCS) techniques. FORM and SORM are based on the Taylor series expansion and MCS/LHS are simulation based methods that can be used alone, or a solver substitution can be made using an appropriate surrogate modeling technique to reduce the computation.

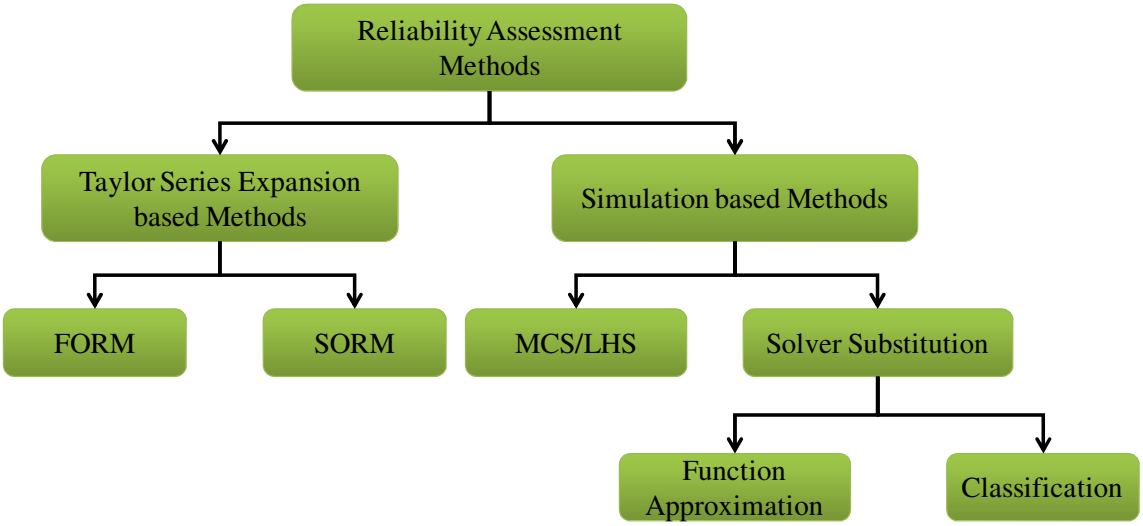


Figure 1. 8 Taxonomy of reliability assessment methods

Figure 1. 8 represents the taxonomy of the different reliability assessment methods that can be used to approximate the reliability constraint. The methods within solver substitution can be further classified into function approximation based methods or classification based methods. In this thesis the primary focus is on the application of classification based methods for reliability constraint approximation, so that discontinuous responses and disjoint failure domains can also be approximated.

1.6 Reliability-based Topology Optimization

Optimization algorithms traditionally have been solved using a deterministic approach where a design solution was obtained for specific force and boundary conditions. However, performing probabilistic analysis prior to the early stage of fabrication is critical to reduce cost, improve product quality, and provide a better understanding of failure mechanisms and sensitivity to process variation. With the high-powered digital computers, it has become feasible to find numerical solutions to realistic problems of large-scale, complex systems involving uncertainties in their behavior. This feasibility has sparked an interest in combining traditional optimization methods with uncertainty quantification measures. These new optimization techniques, which can consider randomness or uncertainty in the data, are known as *stochastic programming*, stochastic optimization, optimization under uncertainty, or reliability-based design optimization. These methods ensure robust designs that are insensitive to given uncertainties and provide the designer with a guarantee of satisfaction with respect to the uncertainties in the objective function, performance constraints, and design variables [42]. The use of integrated reliability analysis and topology optimization procedures, such as reliability-based topology optimization (RBTO) models as stated by Kharmanda et al. [43], yield structures that can possibly be more reliable than those produced by deterministic topology optimization methods. However, realistic representations of uncertainty and the improvement of the computational efficiency are still challenging in the existing methods [44, 45].

1.7 Discontinous Responses and Disjoint Failure Domains

The reliability analysis of complex structures is hindered by the implicit nature of the limit-state function. For their approximation, designers have traditionally used Response Surface Method (RSM) and more recently Artificial Neural Networks, which are essentially nonlinear approximation methods that can model highly non-linear behavior. Both these methods come into the broad category of Regression Approach.

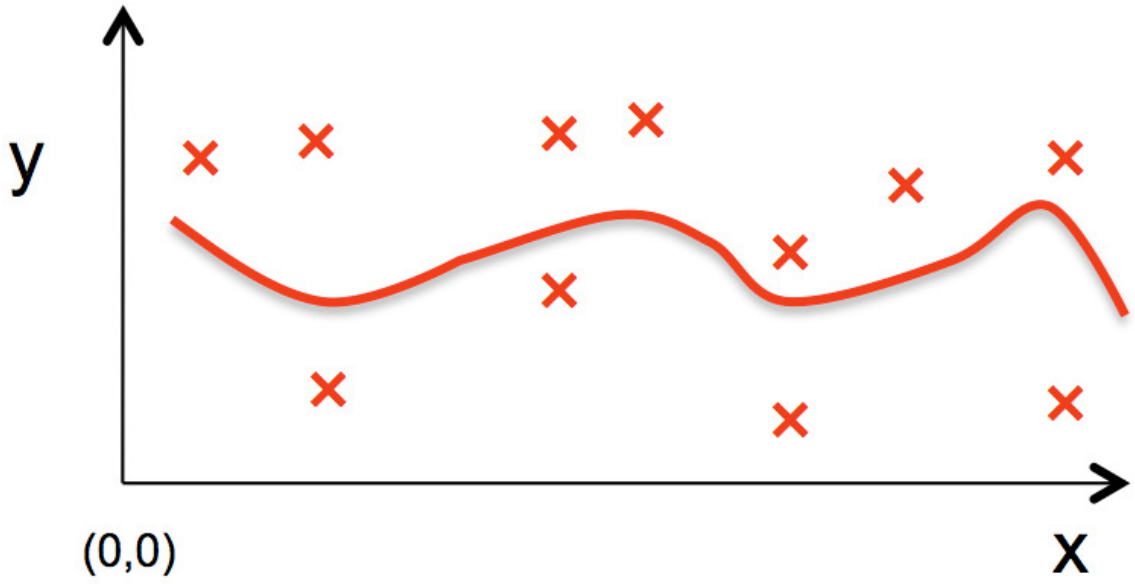


Figure 1. 9 Continuous failure domain example- suitable for regression approach

Figure 1. 9 shows design points in red, which belong to a continuous domain. Hence, a single function can be used to approximate the failure behavior which makes regression-based approaches suitable for surrogate modeling techniques.

A common problem faced in case of approximation using the regression approach is the inability of regression based methods to approximate discontinous functions. Limit state functions are continous and smooth as long as they are in the safe region or the

unsafe region. However, as a system transforms from the safe region to the unsafe region, the limit state function might not be continous anymore. For instance, if a displacement limit state function is posed such as in Eq. (1.1), the limit state function will be linear and continuous when the displacement is less than 0.009. Here x and y would represent the deterministic and uncertain parameters on which the displacement limit state function, $g(x, y)$, depends on.

$$g(x, y) = xy - 0.009 \tag{1.1}$$

Once the displacement is more than 0.009, the limit state function might not be linear anymore because of nonlinearities (geometrical or material) and after a while it could fail by different failure mechanisms such as fracture or buckling. At the point of irreversible failure, the limit state function and displacement value will take a much larger value and will show as an inflexion, which cannot be approximated with any of the regression based surrogate modeling technique.

Instead of approximating the value of the limit state function for reliability estimation, it is easier to estimate the sign of the limit state function. Note that if the sign of limit state function is positive, the displacement is greater than 0.009, the system is unsafe and if the displacement is less than 0.009 the system is safe. As discussed earlier, for reliability estimation, it is enough to find out how many times the limit state function has been violated. Hence, just estimating the sign corresponding to a particular set of input parametes is enough to estimate the reliability or probability of failure of a system. This task can be easily accomplished by a classification based surrogate modeling

technique. Even if the structure is completely in the safe or unsafe region or close to the failure boundary a classification based surrogate modeling technique can be used.

A simple disjoint failure domain is represented in Figure 1. 10. The red lines mark the boundary between the safe and unsafe regions in the design space. The red design points shown in the figure represent the unsafe designs and the green points represent the safe designs.

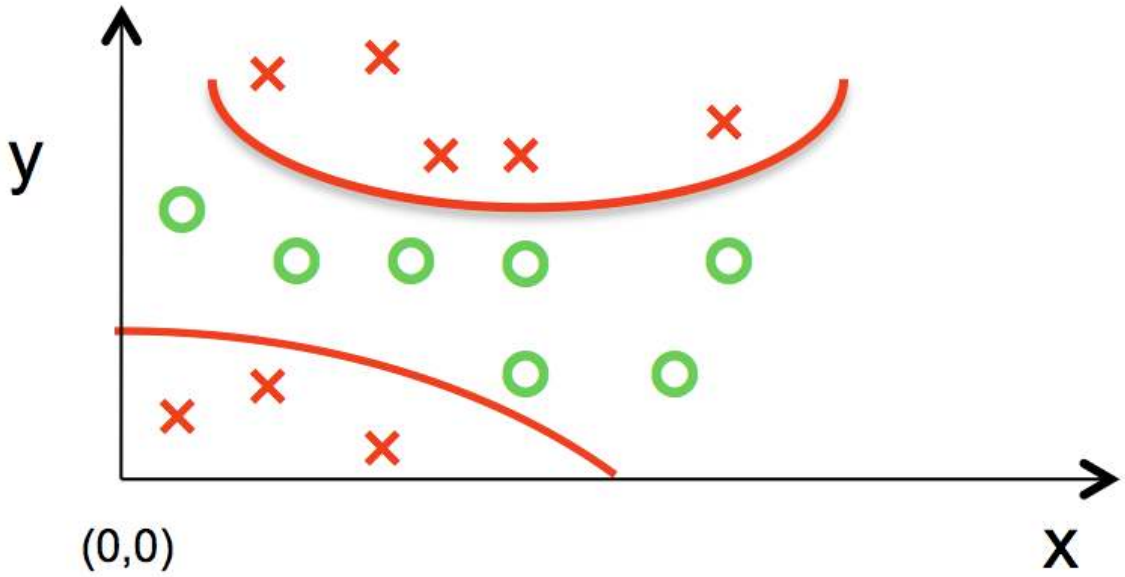


Figure 1. 10 Example of a disjoint failure domain limit state

Hence, in cases where the failure domains are disjoint, regression will not be suitable for estimating the failure behaviour of the design. A classification approach can be used in those cases for approximating the limit states and estimating the probability of failure.

In structural realiability analysis the designer would like to minimize the probabilty of failure as much as possible. Theoretically the probability value can't be zero. Hence, a low value such as 10^{-4} or 10^{-6} is chosen by a majority of designers as the

required probability of failure during a design optimization procedure. In order to reduce the computation cost in evaluating the reliability constraint during the optimization procedure a surrogate model is used by designers with the data obtained using a suitable experimental design procedure. Choi et. al. explored the application of response surface method [46] after Latin Hypercube Sampling (LHS) and Local Regression method [47] for the approximation of the limit state function during design optimization. These processes will still suffer from inability to approximate the probability of failure in disjoint failure domains since they are inherently adaptations and improvements upon the regression based surrogate modeling techniques.

1.8 Research Questions and Hypothesis

The current dissertation deals with the development of computationally efficient reliability estimation procedures for the design of complex systems for different scales and applications in the presence of uncertainty. We constrain our analysis of complex systems to mesostructures, which are meso-scale assemblies of truss or beam elements, assembled together to achieve a certain objective. Since these kinds of complex systems can be comprised of a large number of truss/beam elements, evaluating the response of these structures for a large number of uncertain input parameters using FEA is computationally inefficient. Ideally, we would like to estimate the reliability of these systems with a few representative data points. Hence, the primary goal of this dissertation is represented in the form of the following **Primary Research Question**:

Primary Research Question: *How can we design reliable engineering systems efficiently while minimizing the computational/experimental cost?*

Traditional reliability based methods are not computationally efficient since they have to evaluate the limit state function during every iteration of the optimization algorithm. A surrogate modeling technique can be used in those cases for reducing the computational requirement of the RBTO procedure.

In cases where the failure domain is discontinuous a regression-based surrogate modeling technique will be inadequate for use since regression can only approximate continuous domains. Another major concern in reliability-based designs is the need to deal with low probability of failures. The surrogate model should be able to estimate low values. Due to numerical stability issues, many surrogate modeling techniques can't be used for estimating responses whose values range in different orders.

The factors discussed above raise the following **Secondary Research Question-1**:

Secondary Research Question 1: *How can we accurately predict the quintessential responses obtained from engineering analysis for reliability estimation without requiring additional experimental cost?*

An answer to the secondary research question will successfully provide us a method that will enable the reliability estimation of all complex systems with less computational effort, irrespective of the kind of failure domain that influences it. The resulting surrogate modeling tool will enable the designer to quantify the reliability of a system without worrying about whether the structure is in the safe region or failure region or is in a transition phase between the two.

Finally, in order to answer the Primary Research Question, the above-mentioned surrogate modeling technique should be integrated into a framework which will enable the design of complex engineering systems. This problem can be posed as the **Secondary Research Question-2** as given below:

Secondary Research Question 2: *How can the proposed reliability estimation procedure be used for the design of an engineering system?*

We hypothesize that the answer to the two secondary research questions will enable us to answer the primary research question. In order to answer the Secondary Research Question-1, we hypothesize that including *unlabeled data* in the reliability estimation process can result in reduced computational cost of the overall reliability estimation process. Note that *unlabeled data*, in the case of reliability estimation problems, is the set of uncertain variables that are sampled from the corresponding PDFs for which the corresponding responses of the systems are unknown. More details about including *unlabeled* data in the reliability estimation process will be explained in Chapters 3 and 4. In order to answer the Secondary Research Question-2 we hypothesize that mesostructures are quintessential representations of complex engineering systems and usage of the proposed surrogate modeling technique for the design of mesostructures will prove the efficacy of the proposed surrogate modeling technique in engineering design problems.

1.9 Current Research

The intent of the current research is to explore the synthesis of optimized truss-like mesostructured materials where the loading, boundary conditions and geometry vary according to assumed statistical properties. In this research, a reliability-based synthesis framework is proposed to develop risk-minimized cellular structures that satisfy the performance criteria while specific loading, displacement and shape conditions are imposed. This is achieved by utilizing the stochastic *local regression* [47] procedure for approximating the failure behavior when the reliability constraint is linear in nature. In cases where the reliability constraints are nonlinear or discontinuous, an *artificial neural network* based *classification* technique is proposed which can be used to approximate the failure behavior. Classification based reliability analysis divides the failure domain into safe and unsafe regions and evaluates and classifies the data into one of the two classes, hence, eluding the need to evaluate the response.

The proposed algorithms include a simulation based risk estimation model that provides feedback to the design process and potentially improves the reliability of the meso-scale material structure. Thus, a reliability-based design technique will be integrated to mitigate the risk of structural failure via enhancements of conventional topology optimization techniques.

The following chapters describe important aspects of the algorithm and the solution principle for designing structural systems under uncertainty, which will result in the design of more reliable mesostructured materials.

1.10 Thesis Organization

This dissertation is organized as shown in Table 1. In Chapter 1 we introduce the concept of reliability-based design and how reliability estimation methods are important for the design of small scale as well as large scale systems. We also introduce the basic ideas behind topology optimization, uncertainty in design optimization and relibility based topology optimization. We ended this chapter by introducing the research questions and the corresponding hypotheses.

Table 1. 1 Organization of the dissertation

Chapter 1	Introduction
Chapter 2	State of the Art
Chapter 3	Machine Learning
Chapter 4	Semi-Supervised Learning for Reliability Estimation
Chapter 5	Validation Examples
Chapter 6	Conclusion and Future Work

In Chapter 2, we describe the state of current research in the fields of reliability estimation and design of mesostructures. In particular, we focus on surrogate modeling techniques and basics of the ones that are most prevalent now. We explain the basics of regression and logistic regression and show how classification is modeled as a regression problem where the probabilities are the dependent variables. A brief summary of different kinds of classification algorithms are also provided. The basics of Artificial

Neural Networks (ANNs) are also introduced in Chapter 2. Although, ANNs can be used for both regression or function approximation problems and classification problems, we focus on using ANNs for classification in this dissertation.

We dig deeper into classification techniques and introduce Probabilistic Neural Networks (PNNs) for classification in Chapter 3. We also introduce *clustering* methods for machine learning. Function approximation/regression, classification and clustering mark the three major categories of machine learning tasks undertaken by machine learning practitioners today.

We build on the explanation of machine learning methods in Chapter 3 and explain two different algorithms in Chapter 4, which are proposed in this dissertation for reducing the computational cost of reliability estimation processes. For this, we introduce *unlabeled* data (sampled points from corresponding PDFs) to the already available *labeled* data (sampled points from PDFs with the corresponding limit state function values). Together, labeled and unlabeled data comprise the training dataset for the surrogate model. This enables reduced computational cost since the computational cost of sampling unlabeled data in reliability-based design problems is almost negligible because the PDFs for uncertain variables are already available. The usage of *unlabeled* data with available *labeled* data in order to create better surrogate models is referred to as Semi-Supervised Learning (SSL). More details of these methods are discussed in Chapter 4.

Illustrative examples, which validate the efficacy of the proposed framework, are shown in chapter 5. The proposed methods are shown to work efficiently on continuous as well as discontinuous analytical examples. These examples will be used to validate out

hypothesis for the Secondary Research Question 1. Lastly, we provide an example where the design of a meso-scale compliant gripper is described which could be used for biological applications. This example will be used to validate the hypothesis to the Secondary Research Question 2.

In Chapter 6, we summarize the main points outlined in the thesis along with the advantages of the proposed framework. The limitations of the current research is discussed along with the suggestions of future work.

CHAPTER 2

STATE OF THE ART IN STRUCTURAL RELIABILITY

In this chapter, state of the art in research related to the design of customized reliable mesostructures is presented. This review is presented in six sections. Section 2.1 outlines the methods for design of customized mesostructures through the design of meso-scale truss structures. Section 2.2, describes the Michell analytical method for the design of light weight truss structures. Section 2.3 describes the deterministic optimization process as well as the topology optimization process that is widely used for design of truss structures. This lays the groundwork for section 2.4, where Reliability-based Design Optimization (RBDO) is described that can be used for the design of reliable mesostructures. Important concepts in the RBDO procedure, such as Sampling and Surrogate modeling techniques, are explained in sections 2.5 and 2.6 respectively. Section 2.7, conducts a gap analysis while section 2.8 restates the research questions and the corresponding hypothesis that were introduced in Chapter 1. Section 2.9 summarizes the chapter.

2.1 Methods for Design of Mesostructures

Currently the design and analysis of mesostructures are limited by the assumptions that they are made during the design phase of the mesostructure. Hence, designers use different analysis methods for different kinds of mesostructures. For example, extensive design and analysis have been performed by Ashby et al. [7, 48] .On similar lines, Wang and McDowell have studied the mechanics and behavior of metal honeycombs [49]. The present research focuses primarily on a subset of the analysis done by various researchers

in this field. The focus in this dissertation will be on mesostructures that have a periodically repeating element. These periodic repeating elements are assumed to be comprised of struts.

In related work, analysis was conducted by assuming that these struts are trusses, which implies that the elements of the mesostructure can only take axial loading in the form of either tension or compression and the elements are connected with pin joints. This analysis has been performed by Wallach and Gibson [50] on lattice structures which are subjected to axial loading conditions. Modeling of lattice structures as structures constituting of truss elements has resulted in results that have a percentage error ranging between 3% and 27%. A logical extension of the practice of modeling lattice structures using truss structures is to model these structures using beam or frame elements so that bending and shear stresses can also be considered. This extension has been explored by Chiras et al. [51]. Specifically, Johnston et al. [52] proposed a unit cell model (Figure 2. 1) in which structural members are modeled at beams. The model considers half struts, which are connected together at vertices to form a structure consisting of discrete "unit-trusses". Wang et al. [49, 53, 54] have used this unit cell truss approach successfully for the design of mesostructures.

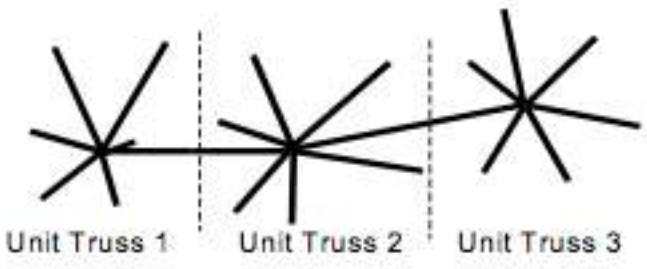
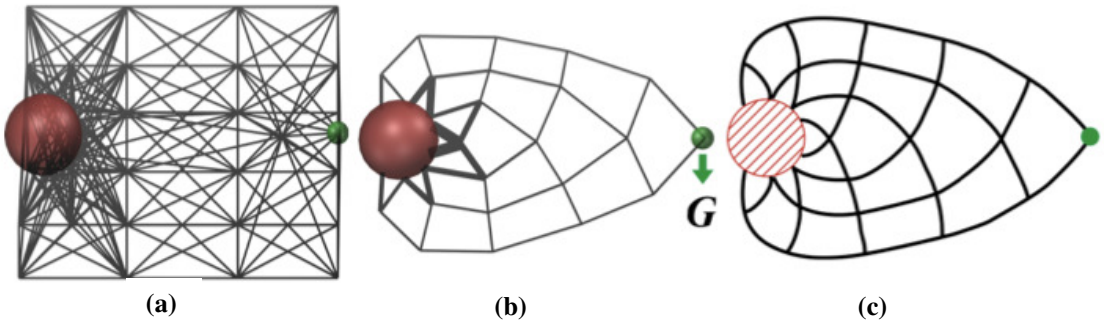


Figure 2. 1 Unit cell approach to mesostructure analysis

2.2 The Michell Analytical Approach

The Michell Truss [44] is a well-known minimum-weight planer truss designed to support a single load with anchors placed on a circle in the same plane [55]. This optimality criteria for least weight trusses with stress constraints and single load condition was derived in 1904 and was an extension of the least weight theorem derived by Maxwell in 1872 [56]. Michell gave several examples of least-weight trusses which included one example for a single point load between supports, one example for a point load and a circular support and a truss along a spherical surface. Figure 2. 2 shows a result obtained by Smith et al. where they start out with the design space discretization in the left and obtain the structure in Figure 2. 2 (b). The force is along the downward direction and is indicated by *G* in this figure. The optimum structure suggested by Michell is shown in Figure 2. 2(b).



(a) Intial design domain (b) Solution obtained by optimization procedure (c) Michell's truss

Figure 2. 2 Michell truss design example from Ref.[55]

Cox [57] applied Michell's and Maxwell's criteria to simple layout problems which included multiple forces. Cox also proved that for certain stress conditions Michell's trusses also minimize the compliance. This was further extended by Hegemier and Prager [58] in 1969 for various design conditions such as plastic collapse load, natural frequency

and stationary creep. However, the need for automated design procedures for design of structures with optimum layout for various objectives still exists. In spite of the analytical approaches provided by Michell, the solutions are limited to only two dimensions.

2.3 Deterministic Optimization Approach

2.3.1 Description of an Optimization Problem

An optimization problem seeks the maximum/minimum of a function $f(x)$ and the variable vector $X=(x_1,x_2,....,x_n)\in R^n$ that it depends on. Here f is called the *objective function* and $x_i,\ i\in 1,.....,n$ are the variables that determine the objective function and are typically called *design variables*. Any vector X in the n dimensional design space represents a single design where n represents the number of design variables in the optimization problem. It is important to note that the design variables can be either continuous or discrete. For example, a structure might have to be made using truss elements for a machine component. If the areas of cross-sections are taken as the design variables and trusses with certain cross-sections can only be purchased then the design variables should be considered as discrete. Since we can purchase any length of these truss elements or cut the purchased truss elements to desired lengths, the lengths can be considered as continuous variables.

In many of the design scenarios, the designer is posed with constraints in terms of geometry, performance, safety, cost and manufacturability. Some of these constraints might have an equality form. Owing to this, the number of independent dimensions in the

design space is reduced, from n, by the number of equality constraints. Along with this, the strict inequality constraints reduce the design space to a subset of R^n.

In the most general form, an optimization problem can be represented as:

$$\textbf{\textit{Minimize}} \quad f(x) \tag{2.1}$$

$$\textbf{\textit{Subject to}} \quad h_j(x) = 0, \qquad j = 1,2,.....,n_h \tag{2.2}$$

$$g_k(x) \le 0, \qquad k = 1,2,.....,n_g \tag{2.3}$$

$$x_i^l \le x_i \le x_i^u, \qquad i = 1,2,.....,n \tag{2.4}$$

where n_h, n_g and n are the number of equality constraints, inequality constraints and design variables, respectively. x_i^l and x_i^u are the lower and upper bounds on the design variable x_i. The implementation of a simple optimization procedure can be represented as shown in Figure 2. 3 below.

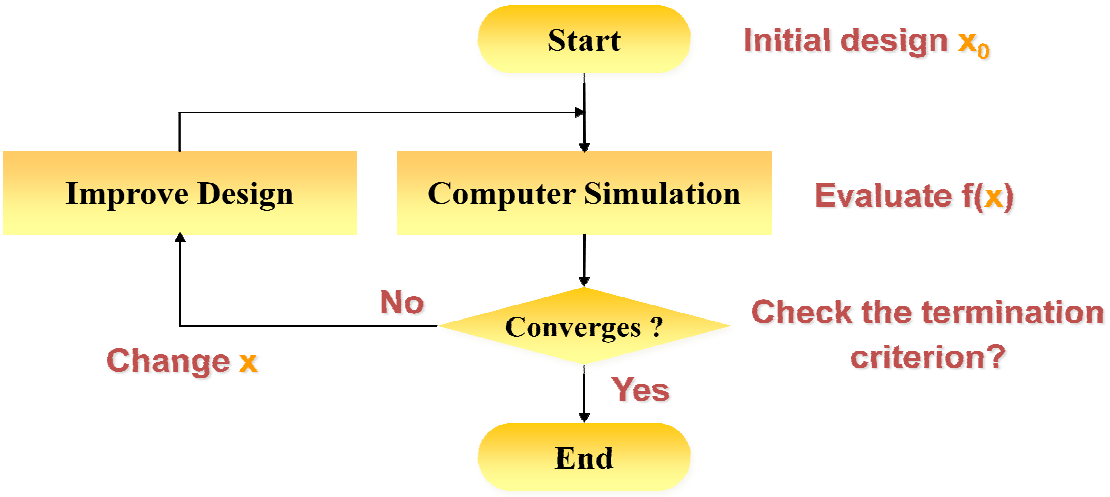


Figure 2. 3 The optimization procedure

If the objective function and all the constraints are linear functions of the design variables then the problem is termed a *linear optimization* problem. In a *nonlinear optimization* problem, either the objective function or at least one of the constraints is nonlinear function of the design variables. In general, structural optimization problems are nonlinear in nature. Further, design optimization can be classified into *size optimization*, *shape optimization* and *topology optimization*. A brief description of each type of optimization follows.

2.3.2 Size Optimization

In size optimization, the domain is fixed and does not change during the optimization process. Hence most of the time size optimization is performed in the final stages of the product design process.

The basic idea behind size optimization is explained with the help of Figure 2. 4. The figure shows a structure that can be discretized into six beam elements. For any given objective function, the design can be optimized for a better performance by altering the thickness of the six beam elements. Hence the thicknesses of the beam elements are considered as the design variables in this case. An important thing to note here is that although the answer from this procedure might be "optimal", changes to the beam element's shapes and the overall topology could possibly give a better result.

Figure 2. 4 Size optimization of beam with six elements

2.3.3 Shape Optimization

Shape, or geometrical, optimization is a somewhat more complex process. In case of shape optimization the topology[1] of the design is fixed whereas the shape is not fixed. The blue points shown in Figure 2. 5 can be used as control points to define the shape of the beam. The wider shape will mean more material usage in this case. Based on the designer's preference the eight variables can be changed that will define the location of the control points and the shape of the overall structure. Similarly, a collection of B-splines or Bezier curves can be used for the shape optimization of a cross-sectional shape. Shape optimization is generally performed during the initial stages of the design process. In general shape optimization can lead to better results than size optimization but again changes to a beam's topology could possibly lead to better results.

[1] Mathematically, two geometrical figures are said to have the same topology if they can be transformed from one to another through continuous transformations. A continuous transformation means pulling, stretching, twisting, bending or squashing without tearing or gluing points together.

Figure 2. 5 Shape optimization of beam with eight control points

2.3.4 Topology Optimization

Topology optimization has the complex features of both size and shape optimization. Topology optimization is often referred to as layout optimization or generalized shape optimization [19]. In this case, the design variables control the topology of the design. This is also the most general optimization procedure, as the size and shape of the structure are affected by the topology. The difficulty in implementing this procedure comes from its generality. Representing the topology of the structure is difficult and generally requires a large number of design variables. Topology optimization operates on a fixed mesh of finite elements and defines a design variable, which is associated with each element in the mesh. A common way of representing a topology optimization problem is to treat it as a configuration design problem where the design is treated as an assembly of a large number of "building blocks". The procedure begins by discretizing the design space into all possible identical building blocks. As the optimization process proceeds, various "building blocks" are allowed to disappear or reappear, which in turn alters the topology of the structure.

Figure 2. 6 represents a topology optimization problem with 72 design variables. In order to design the stiffest beam for a given amount of material, the whole design domain is divided into 72 building blocks. Typically, the target amount of material to be used in the final design is stated as a fraction of the total volume of the structure if all design variables were at their upper bound. As the optimization procedure proceeds, the blocks in white are the ones that are removed from the final design. The final optimized design only constitutes of the building blocks in blue.

Figure 2. 6 Topology optimization of beam using density design variable

In some classical methods of topology optimization a design variable value of 1 means the corresponding building block is present whereas a value of 0 means that it is not. One commonly used optimization procedure allows the design variables to take intermediate values between 0 and 1 and then introduces some form of penalty that steers the solution to discrete 0-1 values. The design problem for that design space can then be formulated as a standard sizing problem by modifying the stiffness matrix so that it depends continuously on a function which is interpreted as the density of material. This density function is the new design variable that can only take discrete values of 0 or 1. One

popular method, which has been used extensively for this 0-1 problem is the Solid Isotropic Material with Penalization (SIMP) model:

$$E_{ijkl} = \rho(x)^p E^0_{ijkl}, \qquad p > 1 \tag{2.5}$$

$$\int_\Omega \rho(x) d\Omega \leq V; \quad 0 \leq \rho(x) \leq 1, \qquad x \in \Omega \tag{2.6}$$

In Eq. (2.5) *ρ(x)* represents the "density" design function and for a given isotropic material E^0_{ijkl} represents the material properties. The density interpolates between the material properties 0 and E^0_{ijkl} as:

$$E_{ijkl}(\rho = 0) = 0, \qquad E_{ijkl}(\rho = 1) = E^0_{ijkl} \tag{2.7}$$

With this definition, if any material design has constituting density values equal to either 0 or 1 then it has a black-and-white pattern. In SIMP the value of *p* for Eq. (2.5) is chosen to be greater than 1 so that the intermediate values of density are unfavorable since the stiffness obtained is small when compared to the volume of the material [19]. In problems where the volume constraint is active Bendsøe and Sigmund [19] have recommended a value of $p \geq 3$ for a true 0-1 design. In effect, an interpolation scheme such as SIMP allows the designer to convert the optimal topology problem into a sizing problem on a fixed design domain.

Compared to many sizing and shape optimization problems, the topology optimization problem is different because the number of design variables for optimization is larger than in traditional structural optimization problems. For structural topology design problems the Method of Moving Asymptotes (MMA) [59, 60] and its "mother" method CONLIN [61] are mathematical programming methods that have been proved to

be effective. These methods are similar to Sequential Quadratic Programming (SQP) as well as Sequential Linear Programming (SLP) because they are used for solving non-linear and smooth optimization problems after the problem is broken down into a sequence of simpler approximate subproblems. These subproblems are separable and convex for MMA and CONLIN and are constructed based on the sensitivity information at the particular iteration point. These subproblems are solved by either a dual method or an interior point algorithm (primal-dual algorithm).

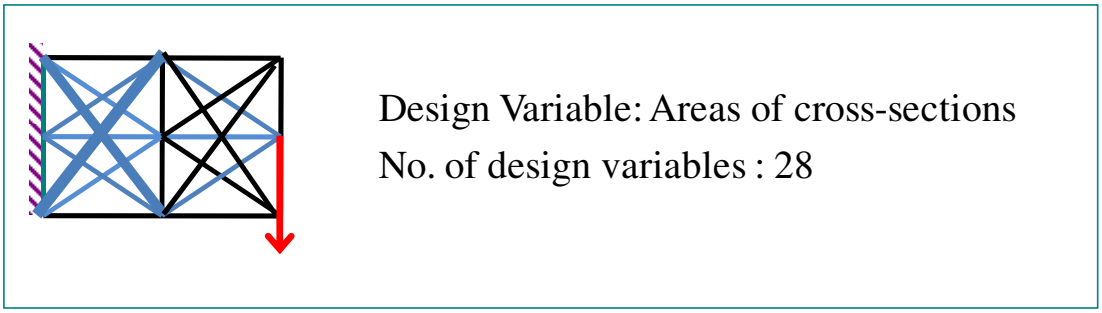


Figure 2. 7 Topology optimization using area of cross-section design variable

Similar to the design of the structure in Figure 2. 6 which consists of 72 building blocks (design variables), a truss structure can also be designed through the topology optimization method if individual truss elements are considered as design variables. Figure 2. 7 illustrates a scenario where a design space is discretized by using 28 truss members. The arrow represents external force on the design space. The cross-sectional areas are the design variables and the design variables are expected to converge to the lower bound or the upper bound of the allowed range. In this particular example, the truss elements represented in blue are the ones, which are remaining at the end of the topology optimization procedure, and the ones in black are the ones that converge to the lower bound. Gradient-based optimization procedures such as SQP, SLP and MMA can also be

used to solve topology optimization problems that are formulated with trusses as the design variables. Furthermore, evolutionary algorithms such as Genetic Algorithms [62] as well as Particle Swarm Optimization (PSO) [63] have also been used in solving truss based topology optimization problems by Hajela *et al.* [64] and Chu *et al.* [65] respectively. A comparison of different optimization algorithms for truss based topology optimization problems has been done by Chang [66]. The details of this truss based topology optimization method are explained in the following section.

2.3.5 Topology Optimization of Truss Structures

Topology optimization of trusses in the form of grid-like continua is a classical subject in structural design. Michell [44] pioneered the study of grid like continuum structures. The development of computationally efficient topology optimization methods is not only important for designing truss structures but also for the design of material structures. The optimization of the geometry and topology of trusses can be conveniently formulated with the so-called *ground structure* method [67]. The truss topology optimization problem is formulated so that the cross-sectional area of every possible truss element connecting the predefined nodes is a design variable. At the end of the optimization routine each of these truss members can either exist or vanish depending on the problem at hand. This is possible by defining the cross-sections as continously varying, owing to which the problem can be viewed as a standard sizing problem. This sizing reformulation is possible because the truss as a continuum geometrically is described as one dimensional. Thus for both planer and space trusses there are extra dimensions in physical space that can describe the extension of the truss as a true physical element of space, simplifying the basic modeling for truss topology design as compared to topology

design of three dimensional continuum structures [19]. Since the area of cross-sections were formulated as continous design variables, a non-zero (small) lower bound on the cross-sectional areas has to be imposed in order to have a positive definite stiffness matrix. Two different types of preliminary structures are shown in Figure 2. 8.

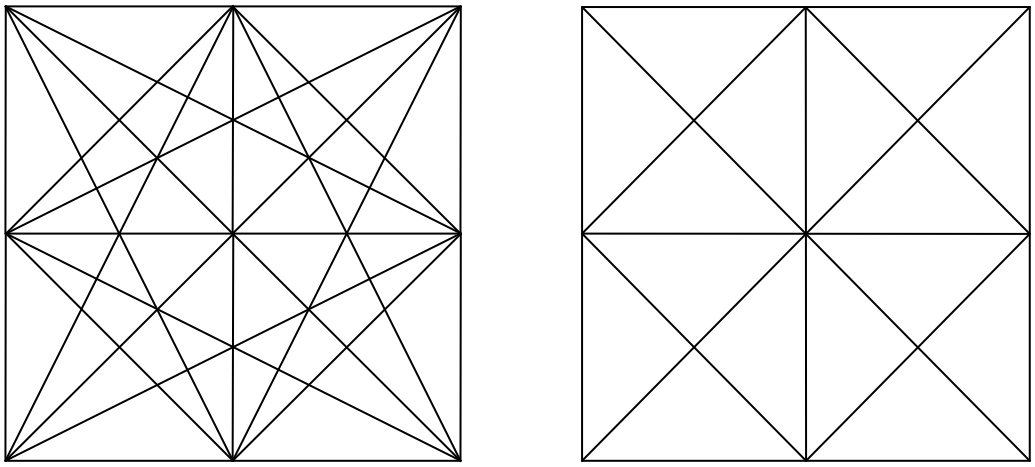


Figure 2. 8 Initial design space discretization (a) Groundtruss (b) Unit cell

Ground structure in Figure 2. 8(a) consists of three nodes along the length and the height of the design space. In this case each node is connected to every other node. Practically trusses can cross each other in space since they can be bolted together to lie in different planes. This kind of initial structure can be an effictive way to form the superset of all possible designs. In a rectangular ground structure with equal number of nodes on all sides, if there are n nodes in total then the number of truss elements in the design space is m, which is represented by Eq. (2.8) . The number of degrees of freedom equals $2n$ for a planer structure.

$$m = \frac{n(n-1)}{2} \tag{2.8}$$

In the unit cell each node is only connected to the most immediate neighbor making this kind of initial structure not as exhaustive as the ground structure. Nevertheless, these kind of initial structures are useful when the designer wants to keep the structure simple and easy to assemble from individual truss elements. These kind of structures can be advantageous while designing mesostructured materials. A simple formulation for topology optimization with areas of cross section A_i as the design variables for truss structure design for a *stiffest structure* [68] objective can be represented as

Minimize: *Mean Compliance* (2.9)

$$\textbf{\textit{Subject to:}} \sum_{i=1}^{N} A_i L_i - V^* \leq 0 \tag{2.10}$$

$$A_l \leq A \leq A_u \tag{2.11}$$

$$Ku = F \tag{2.12}$$

Eq. (2.9) represents the stiffest structure objective because a stiffest structure will have minimum mean compliance. Eq. (2.10) represents the volume constraint where L_i represents the length of each truss element and V^* represents the target volume of the final optimized structure. N represents the number of design variables. In most cases, many design variables (cross-sectional areas) converge to the lower bound. This enables the designer to remove those truss elements from the final design, hence modifying the overall topology of the design space. Eq. (2.11) shows the upper and lower bounds on the cross-sectional areas, A_l and A_u, which represent the lower and upper bounds for the design variable respectively. Eq. (2.12) represents the finite element method that is used to evaluate the objective function and other constraints.

2. 4 Reliability- based Optimization Approach

In topology optimization of truss structures, the objective functions are minimized and the constraints are satisfied in a deterministic sense with reference to nominal values of design variables and other structural parameters. Since most of the structures designed today are faced with uncertain forces, boundary conditions and material properties, it is important to consider these uncertainties in the conceptual design stage of the design process. Traditionally safety factors were used to account for the uncertainties. However, use of safety factors does not usually lead to optimal designs for a given level of safety because different structural members or different failure modes require different safety factors. Recently, probabilistic approaches have been coupled with design optimization methods in order to design structures that achieve the desired objectives even when uncertainties are present. Among all the probabilistic approaches used, *robust design optimization* aims at reducing the variability of structural performance caused by regular fluctuations in the design parameters. The practical concept of robust design was first proposed by Taguchi and a review of Taguchi methodology is given by Tsui [69]. In contrast to robust design optimization methods, reliability based design optimization (RBDO) [42] minimizes the objective function of the optimization problem while considering probabilistic constraints instead of the conventional deterministic constraints. Moreover the applicability of RBDO relies on the availability of precise probabilistic distribution of the stochastic parameters. Similar to RBDO, other studies based on interval set [38, 40] or fuzzy set [70] focus exclusively on structural safety with the motivation of avoiding system catastrophe in the presence of parameter uncertainties. The structural robustness is assessed by measure of the performance variability around the

mean and the structural reliability is based on the probability of failure occurrence, as shown in Figure 2. 9. The probability of failure of a structure is an indication of times the structure violates its permissible safety limits. The designer introduces the safety limits, which are termed limit states, into the RBDO framework.

However, realistic representations of uncertainty and the improvement of the computational efficiency are still challenging in the existing methods [44, 45]. In reliability-based design optimization problems, the designer can be faced with cases where the limit state function is highly nonlinear or discontinuous. Specifically, the use of classical approaches to assess the probabilities of failure is further limited in the disjoint failure region problems [71].

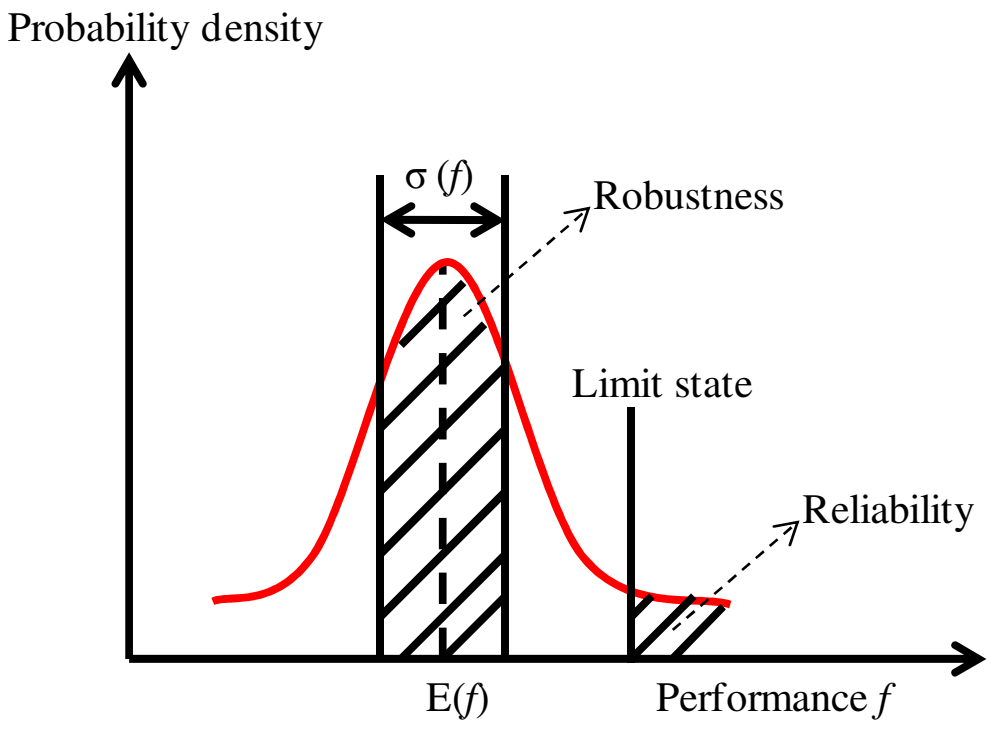


Figure 2. 9 Difference between reliability and robustness

2.4.1 Formulation of RBTO

In order to avoid catastrophic failure in structures, reliability analysis was integrated with topology optimization. The resulting Reliability-Based Topology Optimization (RBTO) [43] yields structures which are more reliable than those produced by deterministic topology optimization. Maute and Frangopol [72] applied RBTO to synthesize compliant mechanisms for MEMS base application. The level-set based topology optimization procedure [73] has also been combined with stochastic optimization techniques for RBTO [74]. Chen et al. [75] investigated the application of random field uncertainty for the robust shape and topology optimization using the level set method. The formation of RBTO is similar to that of deterministic topology optimization except for the reliability constraint, i.e. Eq. (2.14):

$$\textbf{\textit{Min/ Max:}} \quad f(b) \tag{2.13}$$

$$\textbf{\textit{Subject to:}} \quad P_j\left[g_j(b,\underline{x}) < 0\right] \le P_{Rj} \tag{2.14}$$

$$\sum_{i=1}^{N} A_i L_i - V^* \le 0 \tag{2.15}$$

$$b_l \le b \le b_u \tag{2.16}$$

$$Ku = F \tag{2.17}$$

where $f(.)$ represents the objective function, $g_j(.)$ represents the limit-state function, b is the vector of deterministic design variables, and $\underline{x}$ is the random vector, which can be random design variables or random parameters of the system. In Eq. (2.14), P_j [76]

denotes the probability of the event and the probability of failure, P_f, can be defined as $P_j\left[g_j(.)<0\right]$. P_{Rj} is the specified probability of failure (P_f) level. A_i is the cross-sectional area of the elements and L_i is the length of that particular element. V^* denotes the volume of material that can be used in the final design. b_l and b_u are the upper and lower bounds on the deterministic design variables, respectively. K is the global stiffness matrix, u is the global nodal displacement vector and F is the nodal load vector.

Typically, the cross-sectional areas are taken as the design variables. In that case, V^* is given as a fraction of the maximum possible volume of the structure, *i.e.*, in the case where all the design variables go to their upper bound, A_u. Hence, Eq. (2.15) represents the volume constraint. Within every iteration in the optimization processes, the finite element analysis, Eq. (2.17), is invoked and the information required by the objective function is evaluated. Due to the nature of the reliability constraint, Eq. (2.14), in the RBTO problem, it is critical to consider realistic uncertainty representation schemes to conduct accurate reliability assessment. The process of evaluation of the reliability constraint is explained in Section 2.5.4. Figure 2. 10 represents the Reliability-based design optimization procedure. Apart from the objective function and the constraints that are dealt with in the deterministic optimization procedure, the evaluation of the reliability constraint is an important step in RBDO. The reliability estimation is represented inside the serrated box in Figure 2. 10. The reliability constraint introduces randomness in the optimization procedure which enables the method to consider all the variations during the design process.

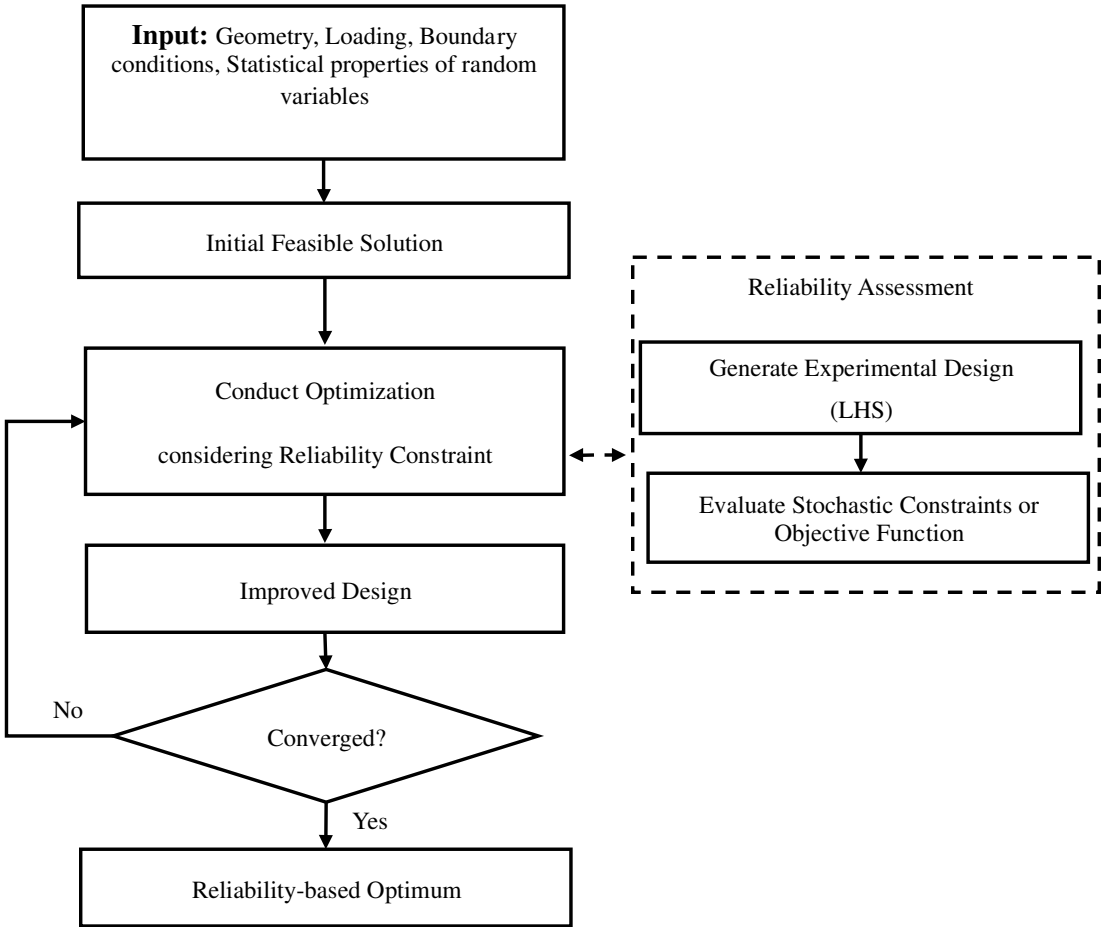


Figure 2. 10 Reliability-based topology optimization procedure

Owing to the stochastic nature of the reliability constraint this optimization process can also be called *stochastic optimization* process. Consequently, evaluation of the reliability constraint increases the computational requirement of the procedure drastically. This computational requirement is caused by the need to evaluate the FEM method multiple times in order to estimate the responses that are required to evaluate the reliability constraint. Hence a surrogate model, which can approximate the response of the FEM procedure, can be used as a way of reducing the computational requirement of the overall procedure. The surrogate model can be constructed after conducting a suitable *experimental design* such as Latin Hypercube Sampling method. The state of the art in

stochastic optimization and structural reliability assessment is discussed in Sections 2.5.3 and 2.5.4 respectively.

2.4.3 Stochastic Optimization

With the emergence of high power digital computers, it has become feasible to combine randomness or uncertainty in the optimization process and hence design large-scale complex systems. These methods are known as *stochastic programming* or *stochastic optimization* methods. These methods help the designer arrive at robust and reliable designs that are insensitive to given uncertainties and hence ensure a guarantee of satisfaction with respect to the uncertainty in the objective function, performance constraints and design variables. In this dissertation, Sequential Quadratic Programming (SQP) has been chosen as the optimization algorithm of choice. Although various other methods including genetic algorithms can be used for this problem, SQP was chosen because of its ability to converge faster and provide equally appropriate solutions for the RBTO problem. SQP has also been shown to give good results in case of large scale nonlinear problems [77]. However, genetic algorithms and other evolutionary algorithms should be used in cases involving discontinuous problems since a gradient-based method such as SQP cannot provide globally optimal solutions for these problems.

Optimization under uncertainty, by its very nature is more expensive than solving deterministic problems. The computational cost of stochastic optimization problems turns out to be extremely high in many cases. This limitation has encouraged researchers to introduce and adapt efficient schemes to represent uncertainty in the optimization procedure. A common approach for treating the computationally expensive objective

function and the constraints is to build relatively inexpensive surrogate models using approximation techniques. The choice of surrogate-based optimization can be reasonable in typical engineering applications. Choi et al. [46] introduced a formulation that combines Polynomial Chaos Expansion (PCE) and Analysis of Variances (ANOVA) within the framework of LHS which can be effective in estimating the responses of large-scale uncertain structural problems. Specifically, to represent variability in stochastic constraints or objective functions, fluctuating components are introduced and approximated in this method. Many other function approximations techniques can be used in order to approximate the variability in the model that can help reduce the computational requirement of the optimization procedure drastically.

2.4.4 Structural Reliability Assessment

Reliability is defined as the probability that a system will perform its function over a specified amount of time and under specified service conditions. Primarily, reliability-based optimal design consists of minimizing an objective function while satisfying reliability constraints. The reliability constraints are based on the failure probability corresponding to each failure mode or a single failure mode decreasing the system failure. In case of structural optimization, the structure is under the influence of loads and boundary conditions and the response depends on the stiffness and mass properties. The responses that are critical for the reliability of the structure such as critical location of stresses, resonant frequencies, displacements etc. are called *limit-state*. The probability of violation of the limit state is a metric for quantifying the reliability of the structure under consideration. Once the limit state has been violated, the structure is believed to have undergone failure for the sake of calculations. By determining the number of times

the structure failed out of the number of evaluations the probability of failure can be determined. Once the probability has been determined, the next step will be to choose design alternatives that improve structural reliability and minimize the risk of failure.

Generally the limit state indicates the margin of safety between the resistance and the load of structures. The limit-state function, $g(.)$, and probability of failure, P_f, can be defined as

$$g(X) = R(X) - S(X) \tag{2.18}$$

$$P_f = P[g(.) < 0] \tag{2.19}$$

where R is the resistance and S is the loading of the system. Both $R(.)$ and $S(.)$ are functions of random variables X. Here $g(.) = 0$ represents the failure surface. $g(.) < 0$ and $g(.) > 0$ represent the failure region and safe region respectively.

The mean of the limit state $g(.)$ can be expressed as in Eq. 2.20, where μ_R and μ_S represent the means of R and S respectively.

$$\mu_g = \mu_R - \mu_S \tag{2.20}$$

The standard deviation of *g(.)* is

$$\sigma_g = \sqrt{\sigma_R^2 + \sigma_S^2 - 2\rho_{RS}\sigma_R\sigma_S} \tag{2.21}$$

where, ρ_{RS} is the correlation coefficient between R and S, and σ_R and σ_S are the standard deviations of *R* and *S*, respectively. The *safety index* or *reliability index* is then defined as [42]

$$\beta = \frac{\mu_g}{\sigma_g} = \frac{\mu_R - \mu_S}{\sqrt{\sigma_R^2 + \sigma_S^2 - 2\rho_{RS}\sigma_R\sigma_S}} \tag{2.22}$$

The safety index indicates the distance of the mean of the margin of safety from *g(.)=0*. The idea behind the safety index is that the design is more reliable if μ_g is farther from the limit state surface.

For a special case, if the resistance *R* and the loading *S* are assumed to be normally distributed and uncorrelated, then the probability density function of the limit-state function can be represented as

$$f_g(g) = \frac{1}{\sigma_g\sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{g-\mu_g}{\sigma_g}\right)^2\right] \tag{2.23}$$

The *probability of failure* can then be represented as

$$P_f = \int_{-\infty}^{0} f_g(g)dg \tag{2.24}$$

For a multidimensional case, the generalization of Eq. (2.24) becomes

$$P_f = P[g(X) \le 0] = \int \ldots \int f_X(x_1, \ldots, x_n) dx_1 \ldots dx_n \tag{2.25}$$

where *g(X)* is the *n*-dimensional limit-state function and $f_X(x_1,...,x_n)$ is the joint probability density function of all relevant random variables *X*.

Due to the curse of dimensionality in the probability of failure calculation in Eq. (2.25) numerical methods can be used to simplify the numerical treatment of the integration process. The Taylor series expansion is often taken to make the limit state *g(X)=0,* linear. This is the basis of the First order reliability method (FORM) [78] and Second order reliability method (SORM) [79]. Other strategies have also been used in the past for probabilistic analysis for designing reliable structures. Stochastic Finite Element method [80, 81], sampling methods and stochastic expansions [82] are some of the most commonly used methods for conducting reliability analysis.

2.5 Sampling Methods

In this research the efficient use of sampling methods for design of reliable material structures is explored. The basic advantage of sampling methods is that the probabilistic information or mathematical solution of a problem can be obtained by direct use of experiments.

2.5.1 Monte Carlo Simulation

Monte Carlo methods were originally practiced under more generic names such as statistical sampling, and the name is a reference to the famous casino in Monaco. The methods use of randomness and iterative procedure is similar to a casino's activities. In Monte Carlo Sampling (MCS) [83] the *inverse transform method* is used to generate random variables with specified probability distributions. This method can be applied to

variables for which the cumulative distribution function has been obtained from direct observation, or where an analytic expression for the inverse cumulative function, $F^{-1}(.)$, exists [42].

Let $F_X(x_i)$ be the Cumulative Distribution Function (CDF) of random variable xi. Since the value of CDF can only lie between 0 and 1, $F(.)$ has a value between 0 and 1. If u is the uniformly distributed random variable that is generated using MCS then the inverse transfer method is used to equate u to $F_X(x_i)$ as follows:

$$F_X(x_i) = u \tag{2.26}$$

or

$$x_i = F_X^{-1}(u) \tag{2.27}$$

This method can be applied to variables for which a cumulative distribution function has been obtained from experiments or where an expression for the inverse cumulative function exists. The process starts with the random number generator producing random numbers between 0 and 1 based on randomly selected seed values. The corresponding CDF value of the uniform distribution and target distribution can easily be obtained using the random numbers that were generated. The final step is to obtain the random number for the target PDF using Eq. (2.27).

Monte Carlo sampling for reliability estimation can be expensive if low probability of failures are being estimated. In order to make MCS less computationally expensive sometimes variance reduction techniques are integrated. Latin Hypercube

Sampling [84] is an excellent variance reduction technique that reduces the computational requirement for the simulation as well as increasing the accuracy with the same number of runs.

2.5.2 Latin Hypercube Sampling

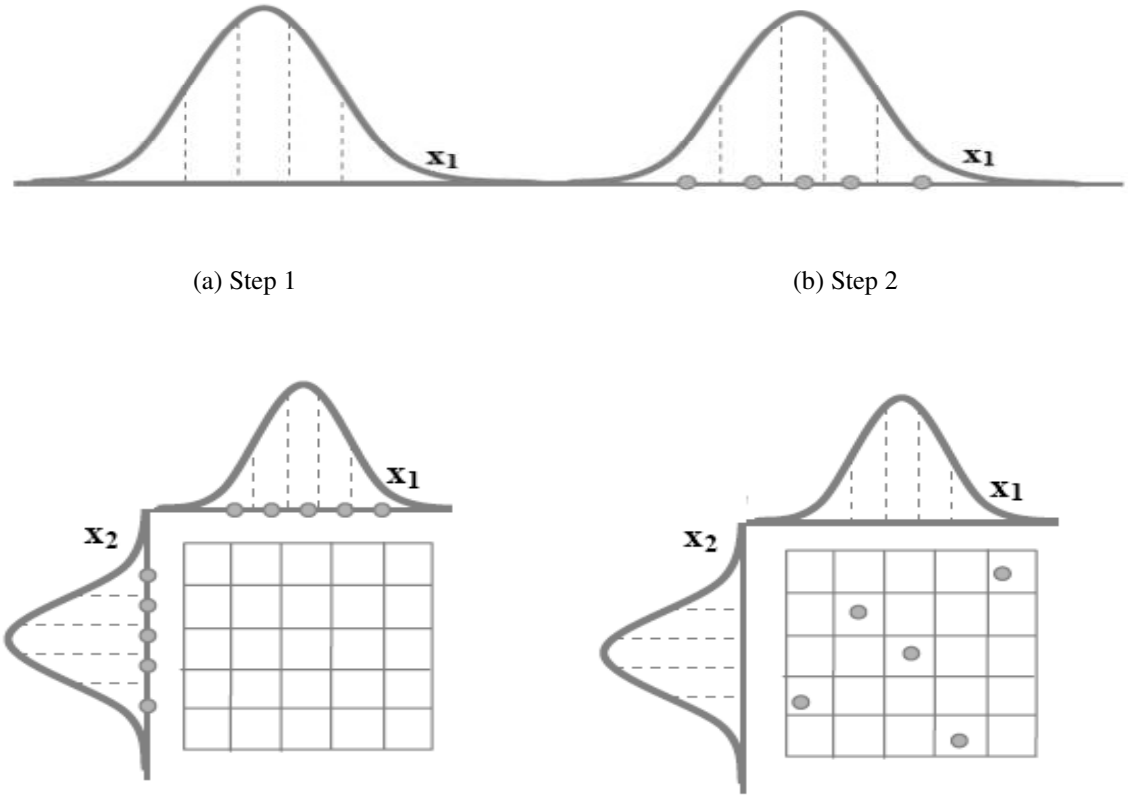
Latin Hypercube Sampling (LHS), also known as the stratified sampling technique, represents a multivariate sampling method that can be used in the reliability estimation problems. In LHS, the distribution for each random variable can be subdivided into *n* equal probability intervals or bins. Each bin has one analysis point. There are n analysis points, randomly mixed, so each of the *n* bins has *1/n* of the distribution probability. Figure 2. 11 shows the basic steps for the general LHS method, which are:

Step 1: Divide the distribution for each variable into *n* non-overlapping intervals on the basis of equal probability.

Step 2: Select one value at random from each interval with respect to its probability density.

Step 3: Repeat steps (1) and (2) until you have selected values for all random variables, such as *x1*, *x2*,…, *xk*.

Step 4: Associate the *n* values obtained for each xi with the n values obtained for the other $xj \neq i$ at random.



(a) Step 1 (b) Step 2

(c) Step 3 (d) Step 4

Figure 2. 11 Basic concept of LHS: Two variables and five realizations

The regularity of probability intervals on the probability distribution function ensures that each of the input variables has all portions of its range represented, resulting in relatively small variance in the estimates. At the same time, the analysis is much less computationally expensive. The LHS method also provides flexible sample sizes while ensuring stratified sampling; i.e., each of the input variables is sampled at *n* levels.

2.5.3 Probability of Failure Calculation

The sampling methods can be used to calculate the probability of failure where the limit state function involves complex functions, and direct evaluation of the limit state is not possible. The following steps are taken to calculate the probability of failure P_f:

Step 1: Generate a sampling set of random variables according to the corresponding probability density functions.

Step 2: Set the mathematical model of the limit-state, which can determine failures for the drawing samples of the random variables.

Step 3: The simulation is executed and for each run the limit state is evaluated.

Step 4: If the limit-state function *g(.)* is violated, the structure or the structural element has "failed".

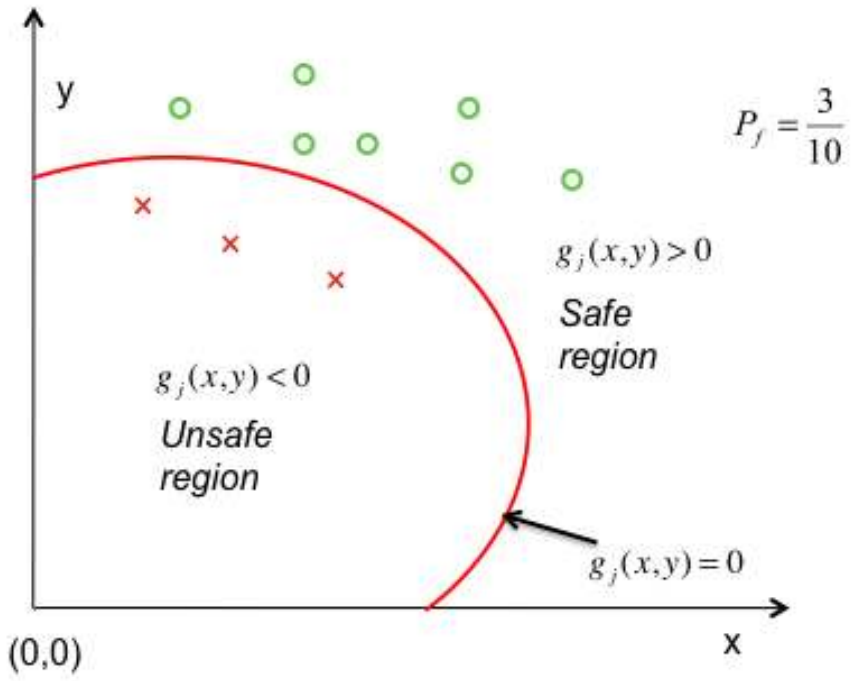
Step 5: The trial is repeated many times to guarantee convergence of the statistical results.

Step 6: If *N* trials are conducted, the probability of failure is given approximately by

$$P_f = \frac{N_f}{N} \tag{2.28}$$

where N_f is the number of trials for which the limit state function is violated out of the *N* experiments conducted.

An example is illustrated in Figure 2. 12. Here 10 data points are generated using LHS procedure. For each data point, *g(.)* is evaluated to check if the corresponding point belongs to the safe region or the unsafe region. The safe and the unsafe region are depicted in the figure. In this example, 3 points are assumed to be in the unsafe region. Hence the probability of failure for this case would be 0.3.



2.6 Surrogate Modeling Techniques

A primary challenge of stochastic analysis is to discover rigorous ways to forecast the low probability of failure, which is critical to reliability constraints. Simulation based methods evaluate the limit state function a number of times in order to calculate the probability of failure. In case of reliability-based designs, the probability of failure has to be calculated in every iteration, making it a computationally expensive procedure.

A common simplification is to approximate the system response using relatively inexpensive surrogate modeling techniques. Figure 2. 13 represents the Reliability-based design optimization procedure based on sampling schemes. Apart from the objective function and the constraints that are dealt with in the deterministic optimization procedure, the evaluation of the reliability constraint is an important step in RBDO and RBTO. The evaluation of the reliability constraint requires the evaluation of responses by

FEM for every sample created using LHS. The procedure of evaluation of the reliability constraint using sampling methods was explained in Section 2.5. Since random points are sampled for evaluating the reliability constraint, the evaluation of the reliability constraint entails the evaluation of FEM responses for each sample. Consequently, evaluation of the reliability constraint increases the computational requirement of the procedure drastically because of the computational requirement of evaluation of FEM. Hence a surrogate model can be created with a selected small number of sample points which can be used to approximate the limit state function or the reliability constraint. This model can be used to reduce the computational requirement of the overall RBTO procedure, assuming that FEM procedure is the major contributor to the computational requirement.

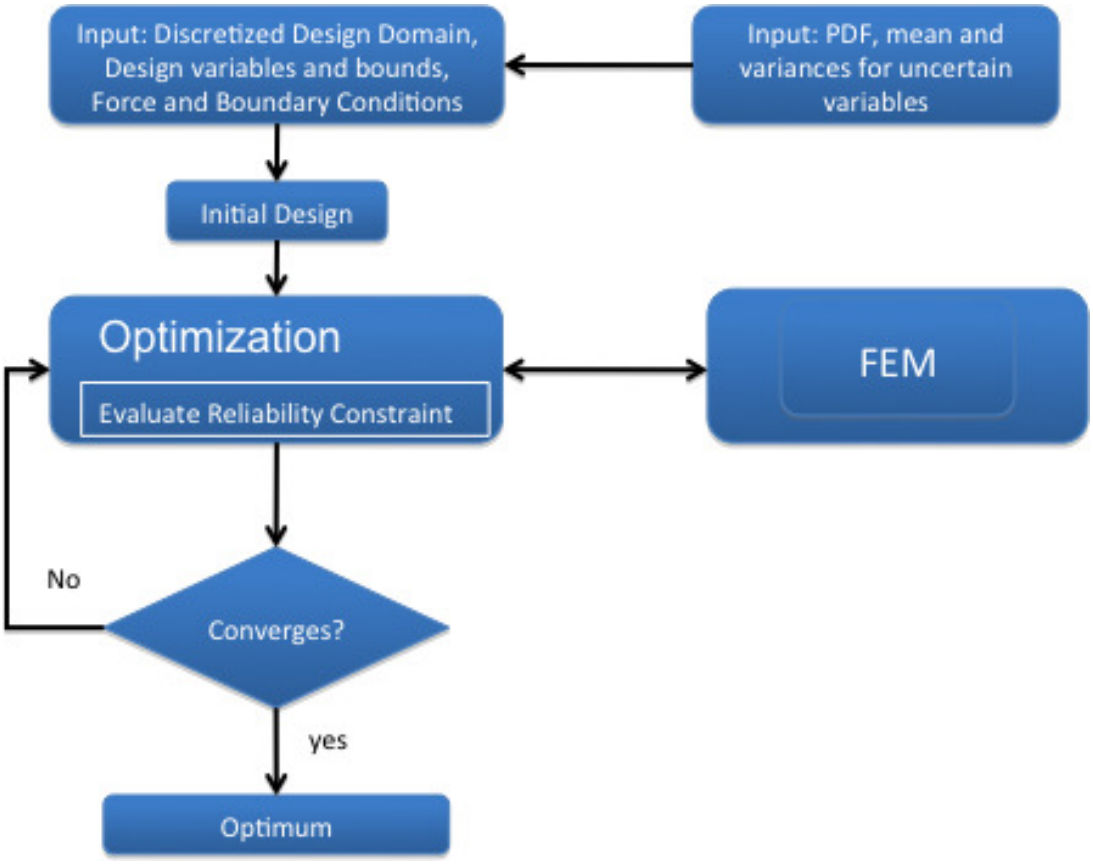


Figure 2. 13 RBTO procedure using sampling scheme

The modified RBTO procedure is illustrated in Figure 2. 14. The designer provides the Probability Distribution Function (PDF) information for modelling the variability or uncertainty that the structural system could be exposed to during its lifetime. This variability could come from variable external forces on the system as well as variable boundary conditions and material properties of the structures in the system. In order to calculate the probability of failure (P_f) value, samples are generated from the PDF function and the corresponding responses are evaluated by using FEM. Unlike the procedure depicted in Figure 2. 13, the procedure shown in Figure 2. 14 uses a few representative data points, which are generated using Latin Hypercube Sampling. The FEM is then used to calculate the responses for the samples created by LHS. This dataset, comprising of the samples and the responses can then be used to formulate the surrogate model. The next sections explain the different surrogate models that have been used for the RBDO and RBTO procedures. Specific advantages and disadvantages of using each method are also explained in these sections.

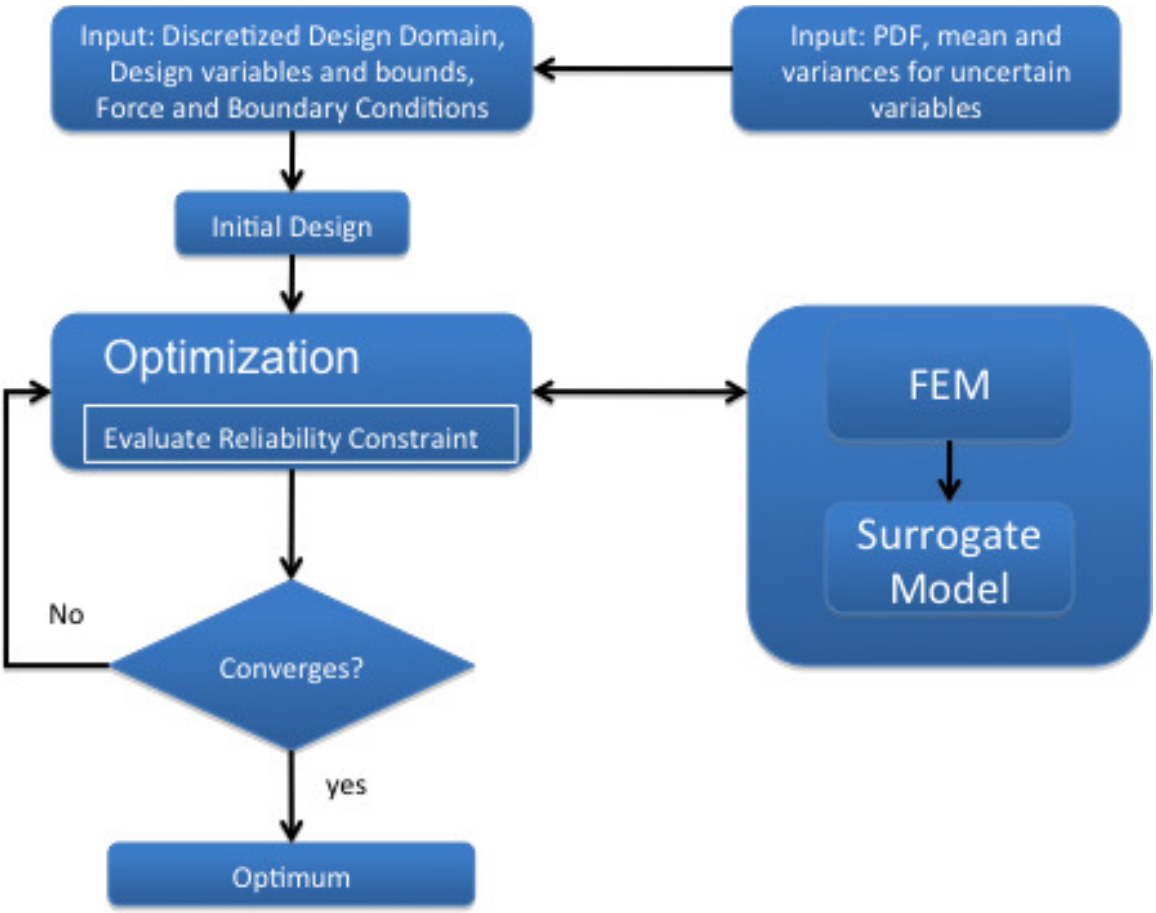


Figure 2. 14 RBTO procedure using a surrogate model and sampling scheme

2.6.1 Function Approximation

Function approximations play a major role in iterative solutions and optimization of large-scale structures. For many structural optimization problems, evaluation of the objective function and constraints requires the execution of costly finite element analysis for displacements, stresses or other structural responses. The optimization process may require the evaluation of the objective function and the constraints hundreds or thousands of times. For example in case of the RBDO method, for every iteration of the optimization procedure the probability of failure has to be calculated using Eq. (2.14), which can require the finite element analysis of the structure *N* times in order to evaluate

the limit state function. In order to reduce the computational requirements, an experimental design like LHS scheme is used to generate a small number of samples of input data and the response is obtained from the finite element analysis. This data is used to construct a surrogate model that can then be evaluated using *N* samples generated using any sampling scheme to evaluate the reliability constraint.

Some of these techniques can be used as a black box (viz. Neural Network based methods), whereas for some of the methods (viz. regression and response surface techniques) it is important to have knowledge of the inherent physics of the problem. Furthermore, Artificial Neural Networks (ANN) has the added advantage that they can be used either for function approximation or for *classification*.

In the case of both function approximation and classification, the goal is to establish a relationship between the inputs and the outputs to a system. The inputs are called independent variables and the outputs are called dependent variables since they are expressed as a function of the independent variables. In case of function approximation, the dependent variables are continuous and can take any real value whereas in case of classification, the dependent variables are discrete or *categorical*. A simple scenario below will help illustrate the difference between regression and classification:

Today, stock investor Tom wants to decide whether to buy or sell the stock of company A. In order to make this decision he wants to consider three factors: the historical stock price, the number of shareholders and the historical Dow Jones Industrial Average value. A classification model can be formulated using these three factors as the input variables and the decision buy/sell as the output. Buy can be represented as 1 and sell can be

represented as -1 in the model. Hence the output or dependent variable is categorical in this case. Consider another case where Tom wants to estimate the most likely stock price of company A in the next day. In this case the dependent variable will be the stock price of A tomorrow and the independent variables will remain the same as in the last case. Note that the dependent variable in this case is allowed to be continuous in this case. Hence Tom can use a classification technique in the first case and a function approximation technique in the later.

In both function approximation and classification, the independent variable will be represented by the symbol X. If X is a vector its components can be accessed by subscripts X_j. Continuous outputs, as in case of function approximation, will be represented by Y and discrete outputs, as in case of classification, will be represented by W. Observed values will be shown in lower case; hence the ith observed value of X will be noted as x_i. The surrogate modelling task can be summarized as: given the value of an input vector X, make a good prediction of the output Y, denoted by $\hat{Y}$ (pronounced "y-hat"). If Y takes values in $\Re$, then so should $\hat{Y}$; likewise for categorical outputs, $\hat{W}$ should take values in the same set ω associated with W [41].

The following sections give a brief description of the regression method, moving Least Squares (MLS) local regression method and artificial neural networks (ANN) methods for both regression and classification.

2.6.2 Regression

Regression[2] is a method that maps the relationship between a response variable *Y* and a covariate or independent variables *X* [85]. The covariate is also called *predictor variable* or *feature* or *independent variable*. One way of expressing the relationship between *X* and *Y* is through the regression function which is expressed as:

$$r(x) = E(Y|X = x) = \int yf(y|x)dy \tag{2.29}$$

The goal in regression is to estimate the regression function $r(x)$ from the data of the form $(Y_1, X_1), ..., (Y_n, X_n)$ where the *Y*'s correspond to the particular *X*'s. If the function *r* is linear then this process is called linear regression and if the function is non-linear then it is called *non-linear* or *parametric*. In general, a parametric function is global in nature and all the data points are used to evaluate the function. For simple linear regression, the regression function can be represented as:

$$r(x) = \beta_0 + \beta_1 x \tag{2.30}$$

If we make the further assumption that $Var(Y|X) = \sigma^2$ does not depend on *x*, the Eq. (2.30) can be rewritten as:

$$Y_i = \beta_0 + \beta_1 X_i + \varepsilon_i \tag{2.31}$$

where, the expectation of the errors ε_i, $E(\varepsilon_i|X_i) = 0$ and the variance of the errors, $Var(\varepsilon_i|X_i) = \sigma^2$.

[2] The term "Regression" is due to Sir Francis Galton (1822-1911) who noticed that tall and short men tend to have sons with heights closer to the mean. He called this "regression towards the mean".

For the linear function represented in Eq. (2.30), the intercept β_0, the slope β_1 and the variance σ^2 are the unknown parameters, which needs to be estimated. The fitted line is

$$\hat{r}(x) = \hat{\beta}_0 + \hat{\beta}_1 x \tag{2.32}$$

The fitted or predicted values are $\hat{Y}_i = \hat{r}(X_i)$ and the errors (residuals) are defined as

$$\hat{\varepsilon}_i = Y_i - \hat{Y}_i = Y_i - (\hat{\beta}_0 + \hat{\beta}_1 X_i) \tag{2.33}$$

The goodness of fit for the regression process can be defined by calculating the residual sum of squares,

$$RSS = \sum_{i=1}^{n} \hat{\varepsilon}_i^2 \tag{2.34}$$

The estimated values of the intercept β_0, the slope β_1 can be represented below as:

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} (X_i - \bar{X}_n)(Y_i - \bar{Y}_n)}{\sum_{i=1}^{n} (X_i - \bar{X}_n)^2} \tag{2.35}$$

$$\hat{\beta}_0 = \bar{Y}_n - \hat{\beta}_1 \bar{X}_n \tag{2.36}$$

And the unbiased estimate of the error variance σ^2 can be represented as

$$\hat{\sigma}^2 = \left(\frac{1}{n-2} \right) \sum_{i=1}^{n} \hat{\varepsilon}_i^2 \tag{2.37}$$

In general it is advisable to include more covariates or predictor variables so that the regression model is able to estimate the values of the dependent variables more

efficiently. However, after a certain point, the limits of efficiency are reached. As a rule of thumb, if we consider a variety of models with different number of predictor variables, the model with least number of predictor variables should be chosen as the final model. This heuristic follows from Ockham's Razor [86-90] principle in statistics where it is suggested that when various models are compared, the model with the least number of terms should be selected. Various model selection models in statistics such as Akaike information criterion (AIC) [41] and Bayesian Inference Criteria (BIC) [41] can be connected back to the Ockham's Razor principle as well. Specifically for regression analysis, stepwise fit or forward and backward elimination methods [4], can also be used for appropriate model selection.

2.6.3 Moving Least Squares Method for Function Approximation

To achieve a high quality surrogate model, the local regression model, namely Moving Least-Squares (MLS) method [91] can be used.

The main advantage of the MLS method is that the regression coefficients are not constant, but rather parameter dependent. This quality allows the data analysis to not be constrained to a specific global function in order to fit a model to the data. Instead, the fitting segments spawn a local-global approximation allowing the data to acclimate to the function over a wide range of parameters. The main idea of local regression is to fit curves and surfaces to localized subsets of the data by a multivariate smoothing procedure with moving processes.

The details of MLS process are shown in Figure 2. 15. In the first step we define the local domain based on the domain influence factor or the bandwidth, r. In the second step an

approximation is estimated at the point x_i. This process can then be repeated at different calculation points by moving the local domain. Therefore, the regression coefficients of the MLS are not constant but a function of the calculation position or location.

A linear regression model can be written as

$$y(x) = \beta_0 + \beta_1 p_1(x) + ... + \beta_k p_k(x) + \varepsilon \tag{2.38}$$

where $p_j(x)$, j = 0,1,2,…,k, are the basis polynomials of order k, β_j are the regression coefficients, and ε, the error of the model equation, is assumed to be normally distributed with mean zero and variance σ_e^2. Eq. (2.38) can be expressed in matrix notation for n sample values of x and y as

$$Y = X\hat{\beta} + e \tag{2.39}$$

where

$$Y = \begin{bmatrix} y_1 \\ y_2 \\ : \\ y_n \end{bmatrix} \quad X = \begin{bmatrix} 1 & p_1(x_1) & p_2(x_1) & . . & p_k(x_1) \\ 1 & p_1(x_2) & p_2(x_2) & . . & p_k(x_2) \\ : & : & : & . . & : \\ 1 & p_1(x_n) & p_2(x_n) & . . & p_k(x_n) \end{bmatrix} \quad \hat{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ : \\ \beta_k \end{bmatrix} \quad \text{and} \quad e = \begin{bmatrix} \varepsilon_1 \\ \varepsilon_2 \\ : \\ \varepsilon_n \end{bmatrix}$$

Here, the simplest polynomial model is the monomials of x^k, i.e.,

$$p^T(x) = [1, x, x^2, ..., x^k].$$

The coefficients can be calculated using a least square formulation. The regression coefficients can be represented as

$$\hat{\beta} = (X^T X)^{-1} X^T Y \tag{2.40}$$

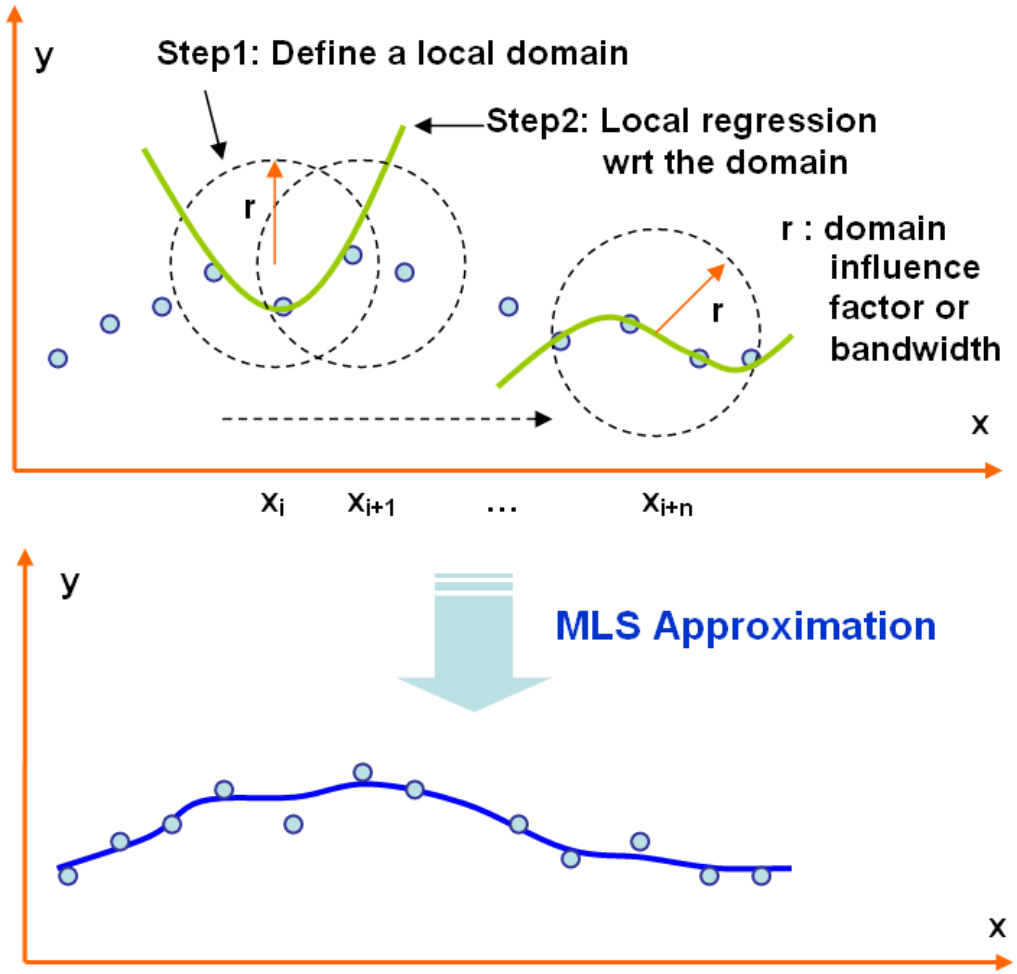


Figure 2. 15 Moving least square approximation process[42]

The estimated target values and the errors are given by

$$\hat{Y} = X\hat{\beta} \text{ and } e = Y - \hat{Y} \tag{2.41}$$

The weight matrix $W(x)$ is also present in the equation for the coefficient matrix in the case of a Moving Least-Squares (MLS) approximation. The regression coefficient vector, $b(x)$, can be calculated as

$$b(x) = \left[X^T W(x) X\right]^{-1} X^T W(x) Y \tag{2.42}$$

where X is a n x m matrix of the levels of the regressor variables, Y is a n x 1 vector of the responses, and $W(x)$ is a non-zero diagonal matrix given by

$$W(x) = \begin{bmatrix} w_1(x-x_1) & 0 & \dots & 0 \\ 0 & w_2(x-x_2) \dots & & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & w_n(x-x_n) \end{bmatrix} \tag{2.43}$$

Hence, the estimates for the MLS model $u^h(x)$ can be represented as follows

$$u^h(x) = \sum_{j=0}^{k} p_j(x) b_j(x) = p^T(x) b(x) \tag{2.44}$$

The weight matrix in Eq. (2.43) is a function of the location or position of x and there are several types of weighting functions. The exponential, canonical and spline functions are widely used as weight functions and are represented as

Exponential weight function

$$w_i(x-x_i) = w(d_i) = \begin{cases} \exp(-(d_i/r_i)^2), & if \quad d_i/r_i \le 1 \\ 0, & if \quad d_i/r_i > 1 \end{cases} \tag{2.45}$$

Conical weight function

$$w(d_i) = \begin{cases} 1-(d_i/r_i)^2, & if \quad d_i/r_i \le 1 \\ 0, & if \quad d_i/r_i > 1 \end{cases} \tag{2.46}$$

(c) Spline weight function

$$w(d_i) = \begin{cases} 1 - 6(d_i / r_i)^2 + 8(d_i / r_i)^3 - 3(d_i / r_i)^4, & if \quad d_i / r_i \le 1 \\ 0, & if \quad d_i / r_i > 1 \end{cases} \tag{2.47}$$

where $d_i = \|x - x_i\|$ is the distance from the sample point x_i to x, and r_i is the smoothing parameter or the bandwidth. The smoothing parameter is an important factor, depending on which the function approximation can widely vary.

Figure 2. 16 depicts the three types of the weight functions discussed in this section. It is important to note that the shape of the fitted curve is not critically sensitive to the precise selection of the weight function. However, the careful adjustment of the domain influence factor of the weight function is critical so that the interval should contain enough data points to obtain the regression coefficients. This is important in order to avoid the singularity of the weight matrix.

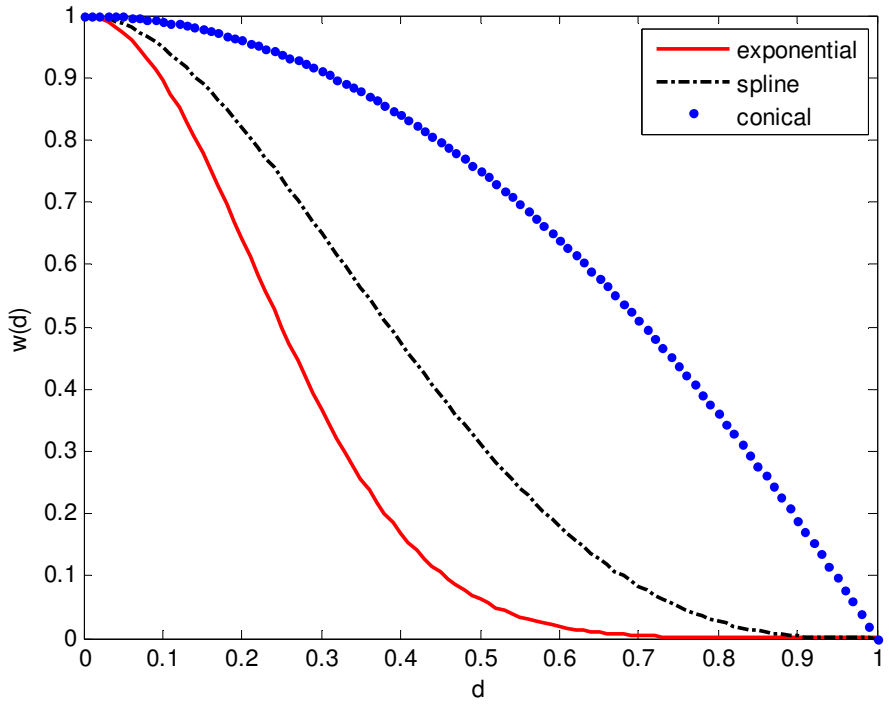


Figure 2. 16 Weight functions

2.6.4 Classification

Classification is a method for estimating a categorical variable given that it depends on a set of independent variables. These categorical variables represent a set of "classes" such as "*small*", "*medium*" and "*large*" or "*safe region*" and "*failure region*" or "*success*" and "*failure*". These are often represented by a single binary digit or bit as 1 or 0, or else by -1 and 1. These numerical codes or categorical variables are sometimes referred to as *targets* since a classification model has to target these variables given the set of independent variables.

For a set of two classes *W*, the simplest approach to classification is to denote the binary coded target as *Y* and then treat it as a continuous output, as in the case of function

approximation. In this case the task will be to estimate the response $\hat{Y}$ which will be in the range [0,1] and we can assign to $\hat{W}$ the class label according to whether $\hat{y} > 0.5$. This approach will also generalize to multi-class cases.

Consider a multi-class case where there are *Z* classes with labels 1,2,...,*Z*, and the estimated linear model for the *z*th response variable is given by $\hat{y}_z(x) = \hat{\beta}_{z0} + \hat{\beta}_k^T x$. For classes *z* and *l*, the set of points for which $\hat{y}_z(x) = \hat{y}_l(x)$ will represent the decision boundary between the two classes. This set of points represents an affine set or hyperplane[3] which can be represented by the set:

$$\left\{x : \left(\hat{\beta}_{z0} - \hat{\beta}_{l0}\right) + \left(\hat{\beta}_z - \hat{\beta}_l\right)^T x = 0\right\} \tag{2.48}$$

In case where there are multiple classes, the input space is divided into regions of constant classification which enables the representation of multiple class boundaries with piecewise hyperplane decision boundaries. This method of classification, which is derived from function approximation methods, is a member of a group of methods that model *discriminant functions* $\delta_z(x)$, for each class, and then classify *x* to the class with the largest value for its discriminant function [41]. Similarly, the posterior probability $P(W = z | X = x)$ can also be modeled using a discriminant function. Once the posterior probability is calculated, the instance *x* can be assigned to the class corresponding to the largest value of posterior probability. This classification rule is the direct statement for

[3] By definition a hyperplane passes through the origin whereas an affine set need not. Common terminology ignores this distinction and refers generally to hyperplanes.

the *Bayes Decision Rule* [41] which states that an instance x should be assigned a class label corresponding to the class with the highest probability, $P(W = z|X = x)$ value.

It is evident that if either $\delta_z(x)$ or $P(W = z|X = x)$ are linear in *x*, then the decision boundaries will be linear. In order to perform classification, all we need now is a linear monotonic transformation for $\delta_z(x)$ or $P(W = z|X = x)$ so that we can form a linear classification boundary. A commonly used monotone transformation is the *logit* transformation: $\log[P/(1-P)]$, which is the basis for *Logistic Regression* method for classification. From the logit transformation it follows for a 2 class situation:

$$\log \frac{P(W = 1|X = x)}{P(W = 2|X = x)} = \beta_0 + \beta^T x \tag{2.49}$$

Since $P(W = 2|X = x) = 1 - P(W = 1|X = x)$, we can derive that

$$P(W = 1|X = x) = \frac{\exp(\beta_0 + \beta^T x)}{1 + \exp(\beta_0 + \beta^T x)} \tag{2.50}$$

and

$$P(W = 2|X = x) = \frac{1}{1 + \exp(\beta_0 + \beta^T x)} \tag{2.51}$$

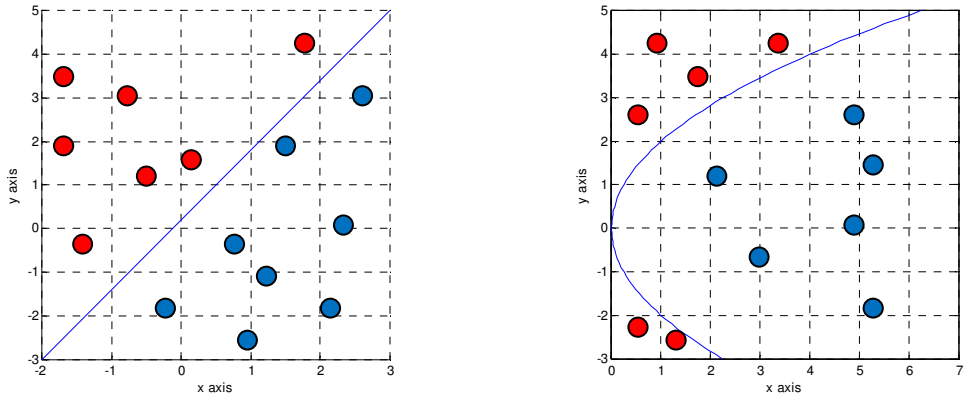
For logistic regression, the decision boundary is represented by the points for which the log-odds represented in Eq. (2.49) are zero, and it is represented by the hyperplane $\{x|\beta_0 + \beta^T x = 0\}$.

Logistic regression is one of the simplest linear classifiers which directly model the posterior probability $P(W = z|X = x)$. Other linear classifiers draw separating hyperplanes in $\Re^m$ so that the data can be separated into different classes as well as possible. *Perceptrons* [92] achieve this by forming linear combination of input features which return the sign. Depending on whether the sign is positive or negative *x* can be classified as an instance belonging to either class 1 or class 2. Perceptrons can be considered as a special case of s*eparating hyperplanes* classification methods which are linear. However, the disadvantages of perceptrons can be summarized in three points:

When the data is linearly separable there can be many solutions and which one is found depends on the initial solution provided. Figure 2. 17(a) shows an example of linearly separable data points and a linear hyperplane that classifies the two classes. Note that many different hyperplanes can classify the data points.

Perceptrons converge in a finite number of steps; but this "finite" number of steps can be very large. The smaller the gap between the two classes, the longer the time required to find the optimum linear combination.

When data is not linearly separable, the algorithm will not converge and go in cycles. Figure 2. 17(b) shows an example of linearly non-separable hyperplane.



(a) Linearly Separable **(b) Linearly Non-separable**

Figure 2. 17 Linearly separable and linearly non-separable hyperplanes respectively

In order to estimate linearly non-separable decision boundaries multilayer perceptrons were invented, which can be considered as a starting point for more complex learning algorithms such as Artificial Neural Networks (ANN). Multilayer perceptrons are feedforward neural networks which require minimum training time but tend to be inefficient when the decision boundaries tend to be complex non-linear functions. In these scenarios the backpropagation neural networks can be used, which will be explored in the next section.

The concept of *optimal separating hyperplanes* is attributed to Vapnik [93], who stated that optimal separating hyperplanes separate two classes and maximize the distance to the closest point from either class. This forms the basis for many separating hyperplane classifiers such as *Support Vector Machines* (SVMs). SVMs choose a hyperplane so that the distance from it to the nearest data points on each side (each class) is maximized. If such a hyperplane exists, it is known as the *maximum-margin*

hyperplane. The nearest data points on either side of the hyperplane are in turn called as *supports* [94].

Logistic regression, separating hyperplanes methods and other linear classification methods fall into the broad category of *discriminative classifiers* [95] since the focus of these models is to estimate the posterior probability $P(W = z|X = x)$ or the discriminant function $\delta_z(x)$. Clearly, the posterior probability can also be estimated by using the Bayes theorem as follows:

$$p(W = z|X = x) = \frac{p(X = x|W = z)P(W = z)}{\sum_{\omega} p(X = x|W = z)P(W = z)} \tag{2.52}$$

And for a case with just two classes, the denominator can be written as:

$$p(x) = p(X = x|W = z_1)P(W = z_1) + p(X = x|W = z_2)P(W = z_2) \tag{2.53}$$

When the Bayes theorem is used for estimating the posterior probability, $P(W = z)$ can be calculated by finding the ratio of number of training data points in each class and the total number of data points. A multivariate Gaussian distribution can be used for calculating $p(x|w)$,

$$p(X = x \mid W = z) = N(x : \mu_w, \Sigma_w) = \frac{1}{(2\pi)^{m/2}|\Sigma_w|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu_w)^T \Sigma_w^{-1}(x - \mu_w)\right) \tag{2.54}$$

where μ_w and Σ_w represent the mean vector and covariance matrix of the Gaussian distribution (Normal Distribution), respectively. Bayes' formula, expressed in Eq. (2.52) can also be expressed informally by saying that:

$$posterior = \frac{likelihood \times prior}{evidence} \tag{2.55}$$

Baye's formula shows that by observing the values of x we can convert the prior probability $p(W=z)$ to the posterior probability – the probability that the state of nature being $W=z$ given that the feature value x has been measured [94]. $p(X=x|W=z)$ will be called the *likelihood* of $W=z$ with respect to x since it is assumed that all things being equal, the category $W=z$ for which $p(X=x|W=z)$ is large is more "likely" to be the true category. Notice from Eq. (2.55) that the product of the likelihood and the prior probability is the most important in determining the posterior probability, and the evidence factor $p(x)$ in the denominator is merely a normalization factor that guarantees that the sum of the posterior probabilities sum up to 1.

Note that when the Bayes Theorem is used for classification, the only undetermined parameters are the mean and covariance values in Eq. (2.54). The classfier is as good as the estimations of means and covariance values are, which affects the quality of the likelihood estimate $p(X=x|W=z)$. All the classification methods that use Bayes Theorem for computing the posterior probability value after calculating the likelihood value are called *generative classifiers* since they focus on estimating the class conditional probability first and then classify the data point based on which class has the largest value of class conditional probability. Generative classifiers will be dealt with in more detail in Chapter 4 during the discussion of Semi-Supervised Learning (SSL).

2.6.5 Artificial Neural Networks (ANN)

Artificial Neural Networks (ANNs) are processing devices (algorithms or actual hardware) that are loosely modeled after the neuronal structure of the mammalian cerebral cortex but on much smaller scales. A large ANN might have hundreds or thousands of processor units, whereas a mammalian brain has billions of neurons with a corresponding increase in magnitude of their overall interaction and emergent behavior. Neural networks have been used for a variety of applications in the past. Some of them are in Machine Learning [96] and data mining, which include:

Having a computer program itself so that the programmer doesn't have to write the code by himself. This is achieved by learning from a set of examples.

Optimization- Given an objective function and constraints, how do we find an optimal solution?

Classification- How to group patterns of data into classes? For example the United States Postal Service uses a neural network based scanning system to recognize the zip code on addresses.

Associative memory- Recalling a memory based on a partial match, which is analogous to case based reasoning.

Regression- It has been proved that neural networks have an ability to approximate any function given the optimal number of *neurons* in the network.

Because of their robust nature and versatility, ANN's find application in a variety of fields [97]. They have been applied in

1) Signal processing: suppress line noise, with adaptive echo canceling, blind source separation.

2) Control: e.g. in backing up a truck, cab position, rear position, and match with the dock get converted to steering instructions. Manufacturing plans for controlling automated machines.

3) Robotics: navigation, vision control.

4) Pattern recognition, i.e. recognizing handwritten characters

5) Medicine: Storing medical records based on case information

6) Speech recognition and production, which helps reading texts aloud.

7) Vision based applications like face recognition, edge detection and visual search engines.

8) Business: Rules for mortgage decisions are made based on the old decisions that produced good results

9) Financial applications: time series analysis, stock market prediction

10) Data Compression: speech signal, image and faces.

11) Game playing: chess, pacman etc.

The simplest computational element for a neural network is called a *neuron*. A neuron can receive inputs from other neurons or from external source. Each input to a neuron has an associated **weight** *w*, which can be modified to model synaptic learning. The weighted inputs are then summed to form the net input for the activation function *f*. A neuron computed summation function f of the weighted inputs is given by:

$$y = f\left(\sum x_i w_i\right) \tag{2.56}$$

The output from this neuron can be input into another neuron for making a network. There can be neurons in parallel or series making different layers of neurons that can make a complex network that is able to approximate any function. Most of the times the number of layers and the number of neurons in each layer has to be decided based on the problem at hand. A simple neuron model can be represented as shown in Figure 2. 18.

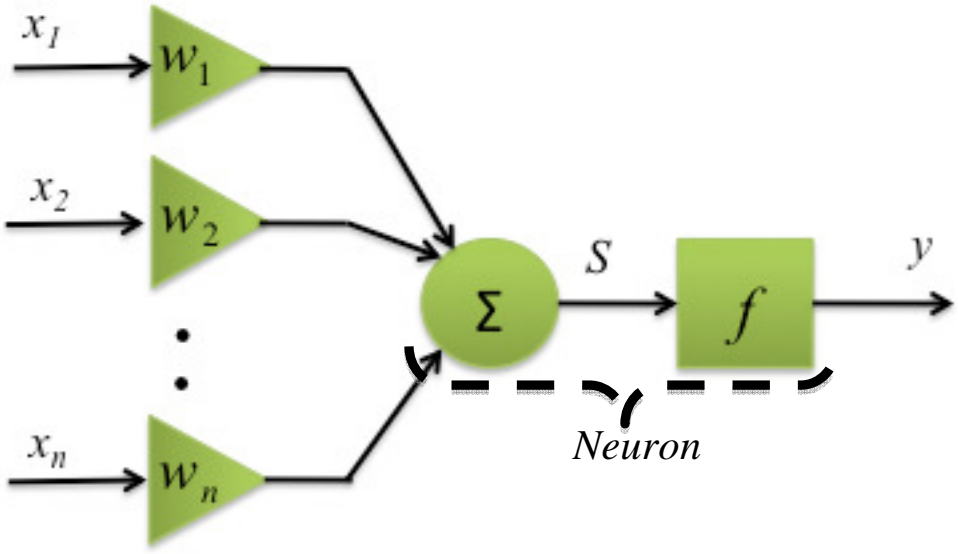


Figure 2. 18 A simple neuron model with *n* inputs

In Figure 2. 18 the weighted sum $\sum w_i x_i$ is called the net input to neuron unit *i* which is referred to as net_i or the sum *S*.

The function f in Eq. (2.56) is referred to as the unit's *activation function* or *transfer function*. For the simplest case, f is the identity function and the unit's output is just its net input. The neuron in that case would be called a linear neuron. The *Hard-Limit transfer function* and the *Sigmoid transfer function* are the two other most used transfer functions. Each of these transfer functions is shown below with red color. The values of all the transfer functions range from -1 to +1. The simplest transfer function is the linear transfer function which is shown in Figure 2. 19. The neurons of this type are used in linear filters as linear approximators. These transfer functions also have been heavily used for function approximations or regression based problems.

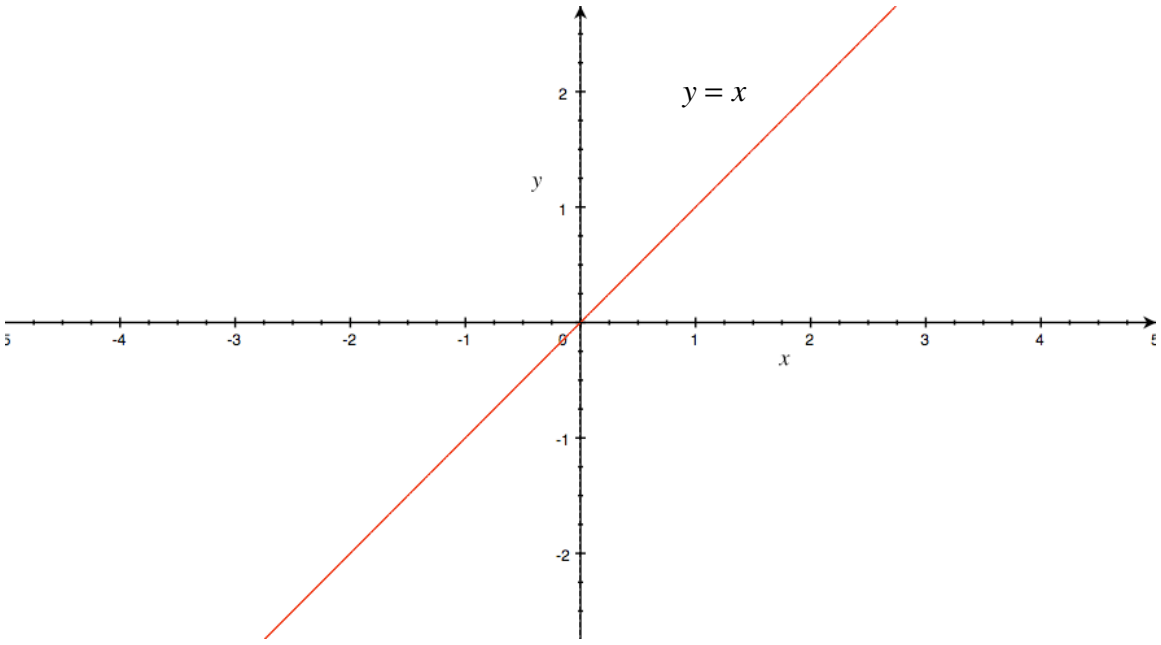


Figure 2. 19 Linear transfer function

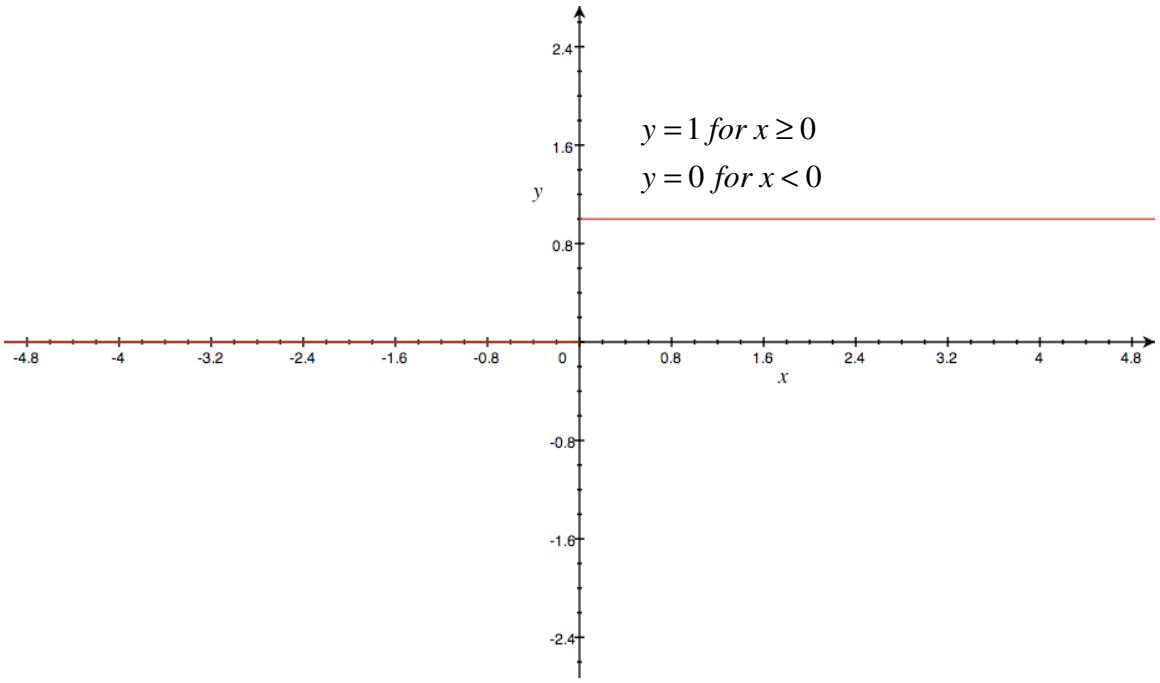


Figure 2. 20 Hard-limit transfer function

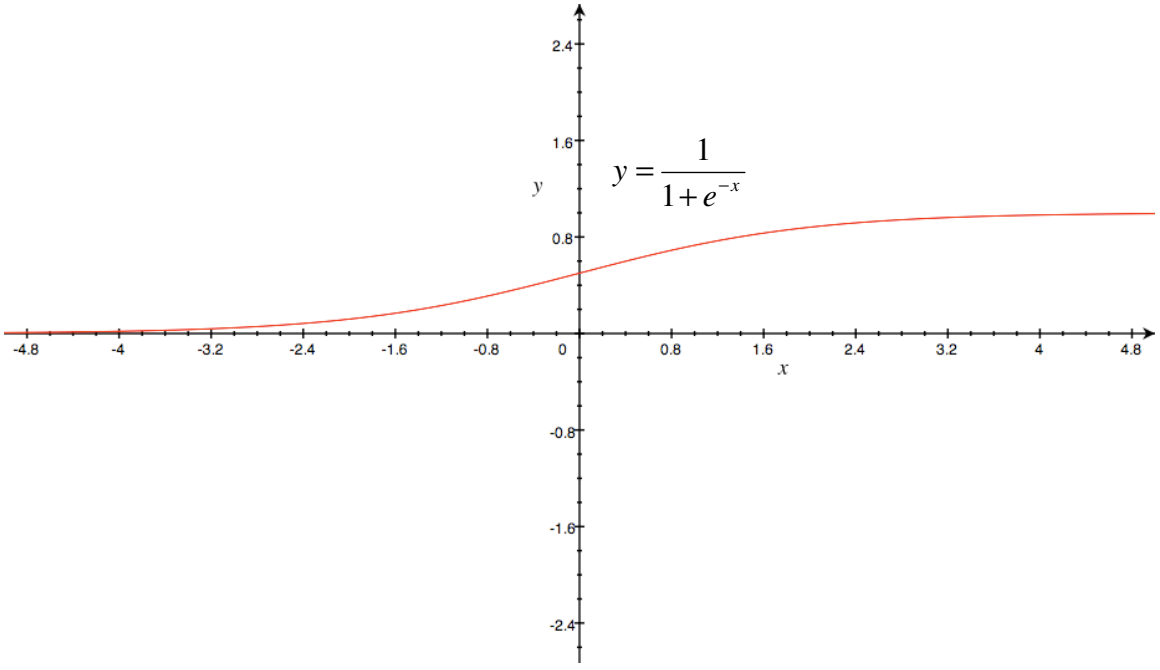


Figure 2. 21 Sigmoid transfer function

The Hard-limit transfer function shown in Figure 2. 20, which limits the output of the neuron to either 0, if the net input argument x is less than 0, or 1, if *x* is greater than or

equal to 0. This function is generally used in classification problems pertaining to perceptrons.

The Sigmoid transfer function is differentiable, which makes it suitable for use in backpropagation networks. A plot of the sigmoid transfer function is shown in Figure 2. 21. In general there are many different types of ANNs and usually there is no single architecture that is suitable for all problems. The main types of ANN architectures widely used are *competitive learning*, the *Boltzmann machine*, the *Hopfield network* and the *back propagation network* [98]. The back propagation network is the most popular type due to its simplicity and ease of use. Its name comes from the way it "back-propagates" the error that occurs during the training process.

A back propagating neural network consists of multiple interconnected processing elements belonging to different layers. In the BP algorithm, learning is carried out using a set of input training patterns propagated through a network consisting of an input layer, one or more hidden layers and an output layer as shown in [98] Figure 2. 22. The hidden layers represent complicated associations between patterns and propagated data in a feed-forward manner from the input towards the output layer. The number of neurons and the number of hidden layers play an important factor in determining the ability of the network to model complex relationship between inputs and outputs. In general, increasing the number of neurons and number of hidden layers increases the ability of the network to model nonlinear relationships, which also increases the training time for the network. The number of nodes in the hidden layer(s) is usually selected as the mean value of the number of the input and output nodes plus the input nodes [99]. More sophisticated networks use "dynamic node pruning" or "node growing" in intermediate layer(s).

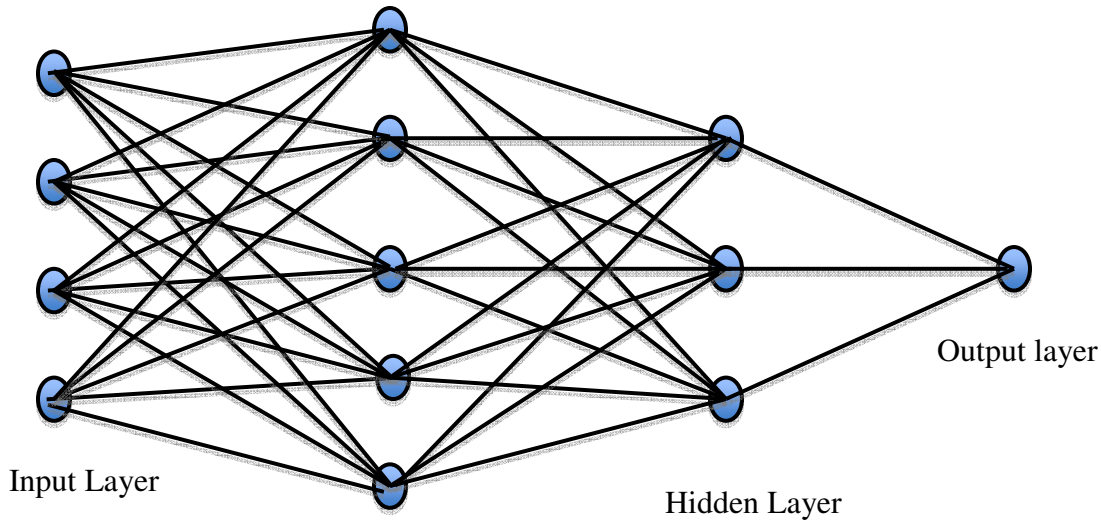


Figure 2. 22 A fully connected ANN configuration

Most of the neural networks use the gradient descent algorithms, such as least squares, in order to correct the values of the weight connections. This comes as an optimization problem where the difference between the computed and desired output values is minimized. The correction step of the weights mentioned above is generally called as the *delta rule.* Once the network has "learned", it produces different outputs for every set of different inputs it evaluates.

Figure 2. 23 shows the connection between two layers of neurons. Let $w_{p,ij}$ be the connection weight between the i neuron in the q(source) layer and the j neuron in the p(target) layer. Let the input signal transmitted from the i neuron of the layer q to the nodes of the target layer p be called $\text{net}_{q,\,i}$, and the output produced at the j neuron of the layer p be $\text{net}_{p,\,j}$. The exterior inputs x_i corresponds to $\text{net}_{\text{q, i}}$ for the input layer.

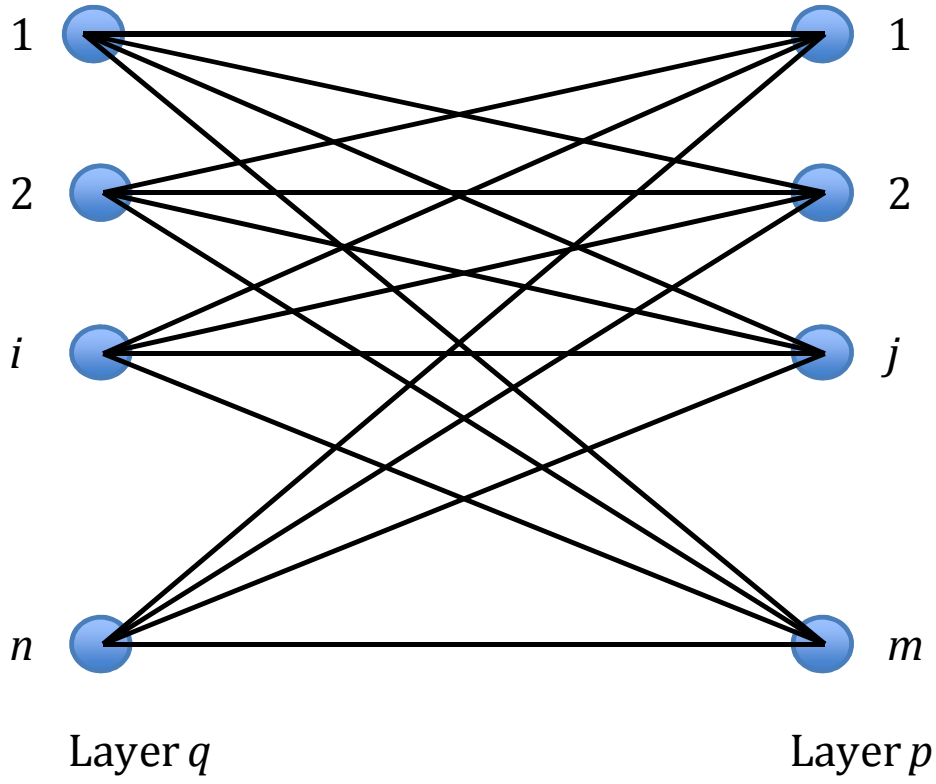


Figure 2. 23 Internal connection between two layers of neurons

In a typical neuron, the output signal is produced only if the incoming signal is strong enough to simulate the neuron. This output is simulated with NN by

$$\text{Out}_{p,j} = f(\text{net}_{p,j}) \tag{2.57}$$

where f is an activation function which produces the output at the j neuron of the p layer. The activation function used in this research is the commonly used sigmoid function

$$f(net_{p,j}) = \frac{1}{1+e^{-(net_{p,j}+b_{p,j})}} \tag{2.58}$$

where $b_{p,j}$ is a bias parameter which acts as a function shifting term that improves the overall network accuracy. Bias parameters can be learned during the training in the same

manner as the other weights. Any random values can be assigned to the weights and bias and during the backpropagation and correction phase the values are improved as the procedure continues. One major advantage of the sigmoid function is that it can handle small as well as large input values. At the output the error can be calculated as the difference between the expected and the actual output value

$$err_{k,j} = tar_{k,i} - out_{k,i} \tag{2.59}$$

where $tar_{k,i}$ and $out_{k,i}$ are the target (expected) and the observed outputs for the node i of the output layer k respectively. The following relationship is used to evaluate the weight changes in the output layer that are related to the input signals.

$$\Delta w_{k,ji} = \eta \delta_{k,i} out_{p,j} \tag{2.60}$$

where η denotes the learning rate coefficient usually selected between 0.01 and 0.9 and $out_{p,j}$ denotes the output of the hidden layer p. Here, η is analogous to the step size parameter in gradient-based optimization algorithms.

The term $\delta_{k,i}$ is the result of the multiplication of the derivative of the activation function, for the neuron in question, with the error signal that is represented as in Eq. (3.61).

$$\delta_{k,j} = df(net_{k,i}) err_{k,i} \tag{2.61}$$

The derivative of the sigmoid function is given by

$$df(net_{k,i}) = out_{k,i}(1 - out_{k,i}) \tag{2.62}$$

This method can be repeated until the desired error level is reached for the training set. This type of training mentioned above is called *supervised learning*. Only a brief description of backpropagation neural networks was given in the previous section. More detailed explanation of back propagation network and other kind of networks can be found in Ref. [41]

In order for the back propagation algorithm to give satisfactory results the training data has to be chosen carefully. A sufficient number of input data properly distributed in the design space together with the output data resulting from the finite element analysis is required to producing satisfactory results in structural optimization problems.

In order to predict accurate structural analysis outputs, the ANN has to be trained properly, which encompasses three tasks:

1) Selecting the proper training set

2) Finding a suitable network architecture

3) Determining the appropriate values of the characteristic parameters such as the training rate

An important limitation of ANN is that there are no rules for determining the efficient training set, architecture or the training rate. Frequently, the designer has to rely on past experience to determine the appropriate characteristics for the data in hand. Most of the times a "*hit and trail*" approach is used which might not lead to good solutions all the time. Hence, a method is required which will alleviate this problem.

In this research, in order to reduce the computational requirements of the reliability-based design procedures, a special kind of ANN called as Probabilistic Neural Networks (PNN) is used. PNN can be used for a classification task only and fits our purpose of reliability estimation.

To summarize, Function Approximation and Classification are the two different ways a designer can estimate the reliability of a system when considering only the sampling based approaches. ANNs are capable of doing function approximation as well as classification but PNNs can only do classification. The methods of reliability estimation using ANNs are given below for both function approximation as well as classification.

Function Approximation Approach- In case randomness is introduced in a design variable x and the output from the FEA is y , which, is used to calculate the limit state, x is the input to the ANN and y is the expected output. A network is trained that can accurately estimate the response y for an input x. The output y can then be used to calculate the limit state and check if it satisfies the safety criteria. By counting the number of times the limit state has been violated, the probability of failure of the structure can be calculated.

This method will be useful to approximate the limit state value in cases where the limit state is highly nonlinear. The disadvantage of this process lies in the fact that there is no set procedure to decide on the characteristics of the ANN such as the learning rate, number of neurons etc. Another major disadvantage of this procedure is that function approximation/regression gives unsatisfactory (wrong) results if the underlying limit state

function is discontinuous. Even in such cases, the regression approach will give us a value for y for which the corresponding x value does not exist in the *neighborhood* of the training dataset. The *classification* approach could be beneficial in this case.

Classification Approach- Classification is used in case we have to classify the inputs into different classes. In order to determine the probability of failure we have to determine if, for the inputs x, the structure has failed or not. Then the ratio of the number of times the structure failed and the total number of input data gives us the probability of failure. Hence, it should be sufficient to determine if the structure has failed for the input x_i. This implies that it would be sufficient to classify an input x_i into either of two classes i.e., pass or fail.

This procedure starts with evaluating the limit state for each of the training data point x_i and evaluating the limit state for each of them and checking if the structure has failed or not and assigning a corresponding class to it. This data is supplied to ANN and a network is created which classifies the test data into either of the two classes. By counting the number of elements in the fail class, the probability of failure can be calculated. This procedure is illustrated in Figure 2. 24.

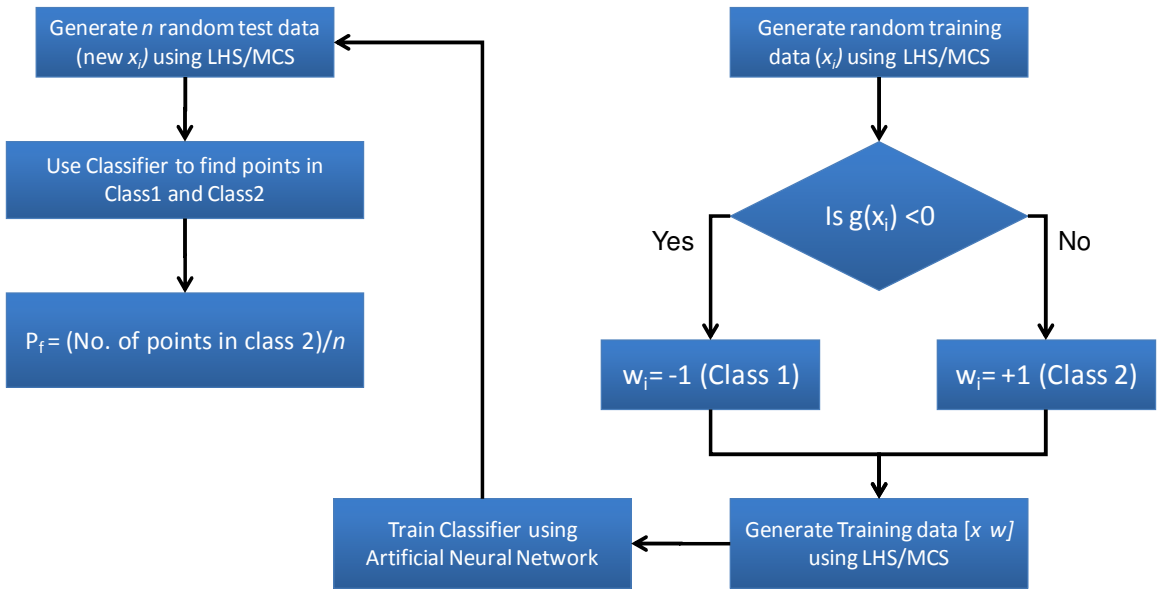


Figure 2. 24 Classification approach to probability of failure calculation

Both the regression approach and the classification approach can be used for estimating the probability of failure for structural reliability assessment. Specifically Artificial Neural Networks can be used for both the regression approach and the classification approach. The back propagation neural networks can be used for both the classification and well as the regression approach. However, if there is a choice between considering function approximation based methods or classification based methods, classification methods give good results and remain the surrogate modeling method of choice when the designer wants to use a surrogate modeling technique as a black-box. In cases where the data is not linearly separable Probabilistic Neural Networks (PNN) can be used to classify the data. A comparison of function approximation and classification based methods for reliability estimation is shown in Figure 2. 25.

	Function Approximation	Classification
Continuous Limit States		
One limit State function	☒	☒
More limit state functions	☒	☒
Discontinuous Limit States		
One limit State function	☐	☒
More limit state functions	☐	☒
Applicable Methods		
Taylor series based (FORM, SORM, etc.)	☒	☐
Simulation based (MCS/LHS, etc.)	☒	☒

Figure 2. 25 Comparison of surrogate modeling techniques for reliability estimation

It is clear from Figure 2. 25, that a classification based surrogate modeling technique applied to an MCS/LHS can be applicable to most cases. This architecture is also applicable in situations where the limit state functions are discontinuous[100]. Hence this will be the chosen architecture for the present research.

2.7 Summary

In this chapter, the state of art in design approaches for mesostructures, reliability-based design approaches and surrogate modeling techniques were presented. Various methods for the design of mesostructures were examined and the Reliability-based Topology Optimization (RBTO) procedure was idenified as the procedure of choice for the systematic design of mesostructures. Based on the benefits and limitations of previous research, a classification-based approach for surrogate modeling was shown to be

appropriate for reliability estimation of a wide variety of engineering systems. Most importantly, the responses from these systems need not be continuous. The reliability levels of systems with discontinuous responses can also be evaluated using a classification framework. Many classification frameworks exist today and all the methods have their advantages and disadvantages. A brief description of machine learning algorithms and the relevant algorithms that can be used for classification will be explained in Chapter 3. The usage of the proposed Semi-Supervised Learning (SSL) algorithm for reliability estimation will be introduced in Chapter 4 followed by design examples in Chapter 5.

CHAPTER 3

MACHINE LEARNING FOR SURROGATE MODELING

In this chapter we introduce the general field of machine learning and explore the various opportunities it provides for surrogate modeling. In Section 3.1, we introduce the different data types and definitions that we will be using for the rest of the discussion in this chapter and the following chapters. In Section 3.2 and 3.3, we will introduce supervised and unsupervised learning— which represent two major types of machine learning tasks. We summarize this chapter in Section 3.4. The foundations introduced in this chapter will be used to explain Semi-Supervised Learning (SSL), which forms the core of this dissertation.

3.1 Machine Learning

"We are drowning in information and starving for knowledge."

–Rutherford D. Roger

In the age of faster computers and cheap data storage infrastructure, we have more data than anyone can possibly interpret alone. With the advent of computers and the information age, statistical problems have exploded in both size and complexity. The field of "data mining" was born in order to solve these challenges posed by the areas of data storage, searching, and organization. These challenges have also created the field of "bioinformatics" in order to deal with statistical and computational problems in the field of statistics and computational biology. Most of the times the statistician's job is to interpret "what the data says" but when dealing with large amount of data the

requirement changes from just “interpretation” to “automatic interpretation”. This has created the fields of “machine learning” and “pattern recognition” where the concern is automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories [101].

Machine learning tasks can be challenging and can originate in a variety of fields. One of the most popular usages of machine learning is by the USPS for automatic recognition of handwritten US zip codes. Typical examples of US zip codes written by hand are shown in Figure 3. 1. Here each digit corresponds to a 28 X 28 pixel image which can be represented by a vector x comprising 784 real numbers. The task is to make a machine that can take each vector x as input and that will post the most likely estimate of the original identity of the number. Hence the output of the machine will be in the form of the digits 0, 1,…., 9. This is a problem that can be tackled, with some difficulty, by using handcrafted rules or heuristics for distinguishing the digits based on the shaped of strokes. However, in practice this approach leads to a proliferation of rules and their exceptions, which leads to poor results.

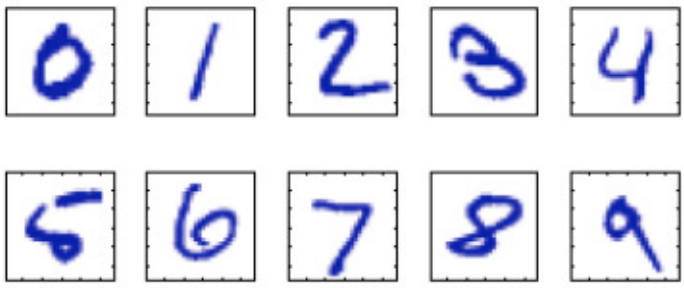


Figure 3. 1 Typical examples of US Zip Codes written by hand [101]

Another way of tackling this problem will be by adopting a machine learning approach where a large set of n digits can be taken to tune the adaptive parameters of a model. These *n* digits will form the *training set*. The corresponding categories for each of these digits in the training set are known in advance. These ten categories ranging from 0—10 form the *target vector*, which represents the identity of the corresponding digit. The result of running this machine learning algorithm can be expressed as a function *y(x)* which takes a new digit image *x* as input and that generates an output vector *y*, encoded in the same way as the target vectors. This process of learning the function *y(x)* is called the training process where only the training data is used. Once the function is learnt, it can then determine the identity of new digit images. These new digit images which do not have an associated digit identity on them comprise the *test set*. In this particular example of pattern recognition, the output vector *y* was categorical, however as discussed in Chapter 2, the output vector *y* can also be continuous, in which case the machine learning algorithms are essentially performing function approximation. Hence, whether a machine-learning algorithm is chosen to perform classification or function approximation depends on the training data, which is comprised of a training set and a target vector. Both of these methods fall under the broad scope of supervised learning.

3.2 Supervised Learning

Let $X = (x_1,...,x_n)$ be a set of n instances (or data points), such that $x_i \in \chi$ for all $i \in [n] := \{1,...,n\}$. In case of pattern recognition machine learning problems, X can also be called as *patterns*. In most cases during training of the machine learning algorithm, it is assumed that points are drawn i.i.d. (independently and identically distributed) from any

distribution of χ. In general, X is an $n \times p$ matrix, which contains n instances, each containing p features.

In the case of supervised learning, the goal is to learn a mapping from x to w given a training set made of pairs (x_i, w_i). Here, $w_i \in \omega$ can be termed labels or targets of the examples x_i. Hence such (*instance*, *label*) pairs are called *labeled* data, while instances without labels are called *unlabeled* data [102]. The standard requirement is that the pairs, (x_i, w_i), are sampled i.i.d. from any distribution which ranges over $\chi \times \omega$. The mapping can be evaluated through the prediction of the performance on training examples. Hence supervised learning is the process of learning from labeled data alone whereas unsupervised learning is the process of learning from unlabeled data alone. The most popular methods of the supervised learning are regression and classification. Regression is a form of supervised learning when the labels are continuous (w_i are continuous values in $\Re$ and we denote them by y). On the contrary, in classification w_i can only take finite discrete values in $\Re$.

3.2.1 Function Approximation as Supervised Learning for Reliability Estimation

Consider the simple scenario where the reliability of a structural system is to be evaluated using sampling based methods. Hence the steps in evaluating the reliability will be as follows:

Step 1: Consider all the parameters in the design and decide on the fixed design variables and the uncertain design variables.

Step 2: Sample *n1* points using MCS/LHS for training from the distribution of the uncertain design variables. These points are the unlabeled points *x1*.

Step 3: Evaluate the responses, *y*, of the sampled points along with the fixed design variables by using Finite Element Analysis (FEA). Together, *x1* and *y*, form the labeled point (*x1, y*).

Step 4: Use a machine learning technique to model the variation in *y* as *x1* varies (Figure 3. 2). This model, *f(x1)*, can now be used for finding the response *y* for any number of newly sampled points representing *x1*.

Step 5: Sample *n2* points using MCS/LHS and evaluate the responses *y* corresponding to these points.

Step 6: Evaluate the limit state function *g(x1)* to find whether the structure has failed or not.

Step 7: The probability of failure of the structure is the ratio of number of times the structure has failed and *n2*.

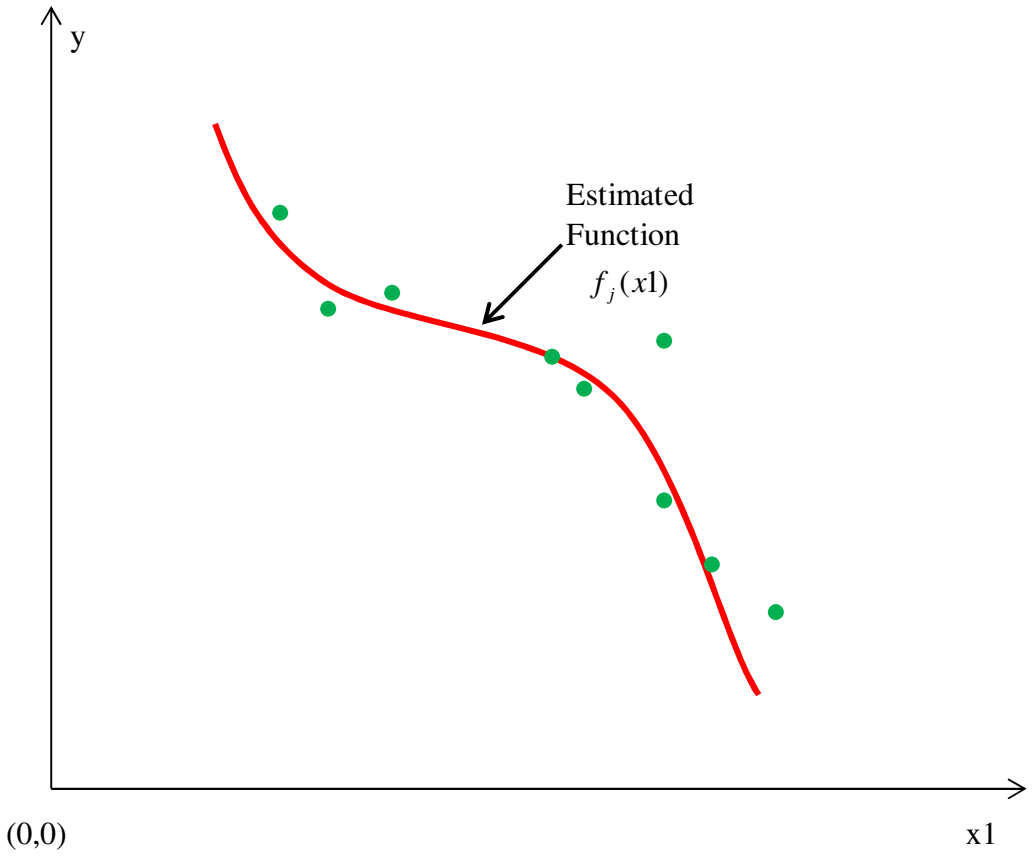


Figure 3. 2 Estimation of response function *f(x1)*

3.2.2 Classification as Supervised Learning for Reliability Estimation

In contrast to function approximation technique, the reliability estimation process using classification techniques is used for estimating whether the structures has failed or not, instead of estimating the response of the structure. In this case the structure is "classified" as either safe or failed—hence the model only is required to estimate the discrete labels—safe or unsafe for each of the input. The steps in estimating the reliability of a structural system is as follows:

Step 1: Consider all the parameters in the design and decide on the fixed design variables and the uncertain design variables.

Step 2: Sample *n1* points using MCS/LHS for training from the distribution of the uncertain design variables. These points are the unlabeled points *x1*.

Step 3: Evaluate the limit state function *g(x1,x2)* (Figure 3. 3) after estimating the responses of the structure using FEA. If *g(x1,x2)* is positive for a particular unlabeled data point, the point is in the failure region and if *g(x1,x2)* is negative then the point lies in the safe region. If the limit state function is positive, the label of the point can be set as 1 and if the limit state function is negative, the label can be set as 0. Together,(*x1,x2)* and the labels, 0 or 1, form the labeled point (*x1,x2, w*). Note that as mentioned earlier in Chapter 2, for classification the labels are discreet and are denoted by *w*.

Step 4: Use a machine learning technique to model the variation in *w* as *x1* varies (Figure 3. 3). This model, $\hat{g}(x1, x2)$, can now be used for finding the response *w* for any number of newly sampled points representing *x1* by simply determining the sign of the decision boundary function $\hat{g}(x1, x2)$.

Step 5: Sample *n2* points using MCS/LHS and evaluate the responses *y* corresponding to these points.

Step 6: Evaluate the limit state function $\hat{g}(x1, x2)$ to find whether the structure has failed or not.

Step 7: The probability of failure of the structure is the ratio of number of times the structure has failed and *n2*.

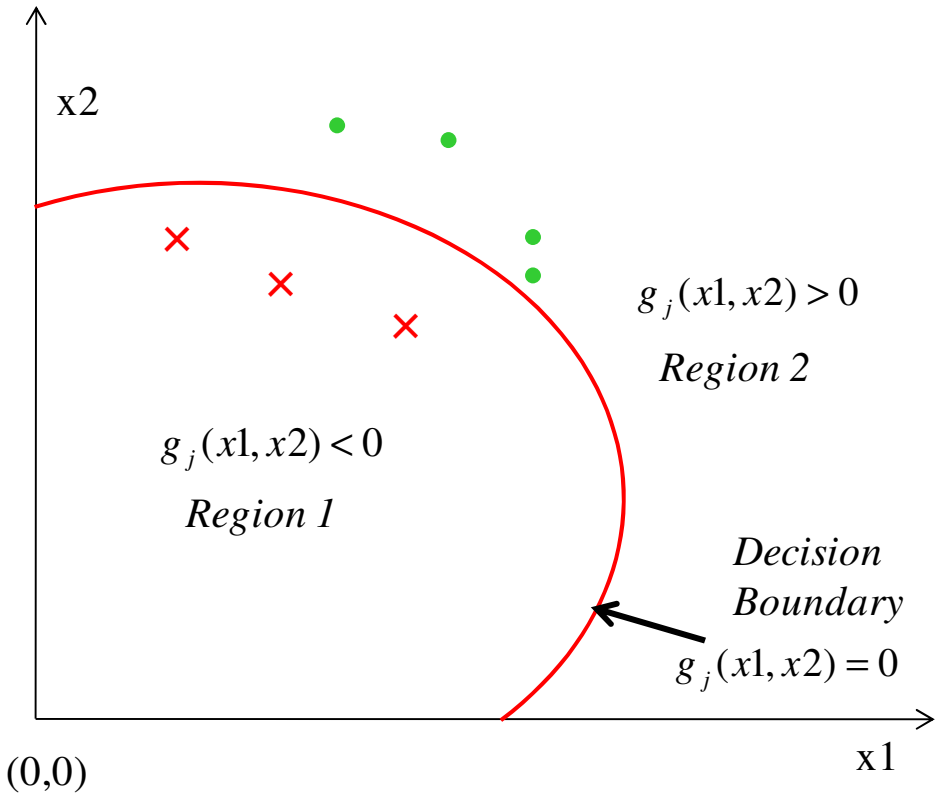


Figure 3. 3 Estimation of decision boundary *g(x1, x2) = 0*

3.2.3 Artificial Neural Network as a Superior Machine Learning Method

The term neural network has evolved to encompass a large class of models and learning methods. Many times researchers have used Neural Networks as a black box in order to conduct function approximations or classifications. Because it is easier to control the complexity of neural networks by increasing the number of hidden layers—inherently it is easier to increase the nonlinearity of the models, which allow ANNs to approximate functions seamlessly. A brief comparison of various machine-learning algorithms is given in Figure 3. 4. It shows that ANNs have superior predictive power and superior ability to extract linear combinations of features. On the flip side ANNs should not be used as "*off the shelf*" machine learning algorithms since they can be highly sensitive to

outliers as well. In order to use ANNs the number of hidden layers has to be fixed before training the algorithm. If the number of layers is less then ANN's responses resemble simple linear models for regression and classification. If a high number of hidden layers are used, ANNs resemble highly non-linear models for regression (such as local regression) and classification (such as k-Nearest Neighbors). Until now there is no simple rule for choosing the right number of hidden layers. The number of hidden layers typically varies between 5-100 for different problems from different domains [41].

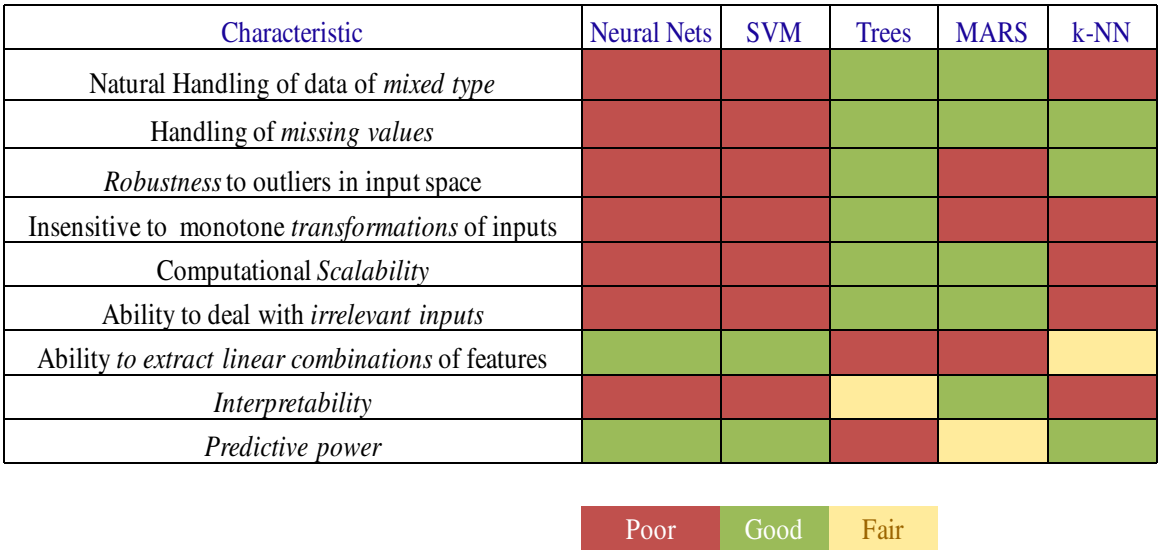


Figure 3. 4 Comparison of various machine learning algorithms[41]

One of the ways to surmount this problem is to fix the number of layers and then vary other parameters in the network. We propose different methods of doing it for reliability estimation problems in Chapter 4. Here we continue to the discussion of ANNs and introduce the idea of Probabilistic Neural Networks in Section 3.2.4 which will be building block for our proposed methods in Chapter 4.

The details of neural networks, particularly the back propagation neural networks were previously explained in Chapter 2. Neural Networks can be made to perform functions approximation as well as classification. Any kind of continuous function can be used for performing function approximation; but the sigmoid function has been used as an activation function extensively because of its smooth properties as well as its appropriateness for back propagation networks. However, a major complaint with back propagation training is the high computational time requirement with them. Feed-forward neural networks can be used in those scenarios where the computational time required in a training process is the critical factor; which is also true in case of reliability based design processes.

We introduce a simpler ANN, feed-forward neural network, in Figure 3. 5. This neural network has z neurons at the top, with the z th unit modeling the probability of class z. There are Z target measurements Y_z, $z = 1,\ldots,Z$, each being coded as 0-1 variable for the z th class. Derived features in hidden layers, H_p are created from linear combinations of the inputs, and then the target Y_z is modeled as a function of linear combinations of the H_p.

$$H_p = f(\alpha_{0p} + \alpha_p^T X),\ p = 1,\ldots, P \tag{3.1}$$

$$Y_z = \beta_{0z} + \beta_z^T H,\ z = 1,\ldots, Z \tag{3.2}$$

In case of radial basis function neural networks, Gaussian radial basis functions are chosen for *f(.)* in Eq.(3.1).

$$f(x) = \exp(-\frac{x^2}{2\sigma^2}) \tag{3.3}$$

A more general form of writing Eq. (3.3) will be in the form of the general Gaussian distribution.

$$f(x) = \exp(-\frac{(x-\mu)^2}{2\sigma^2}) \tag{3.4}$$

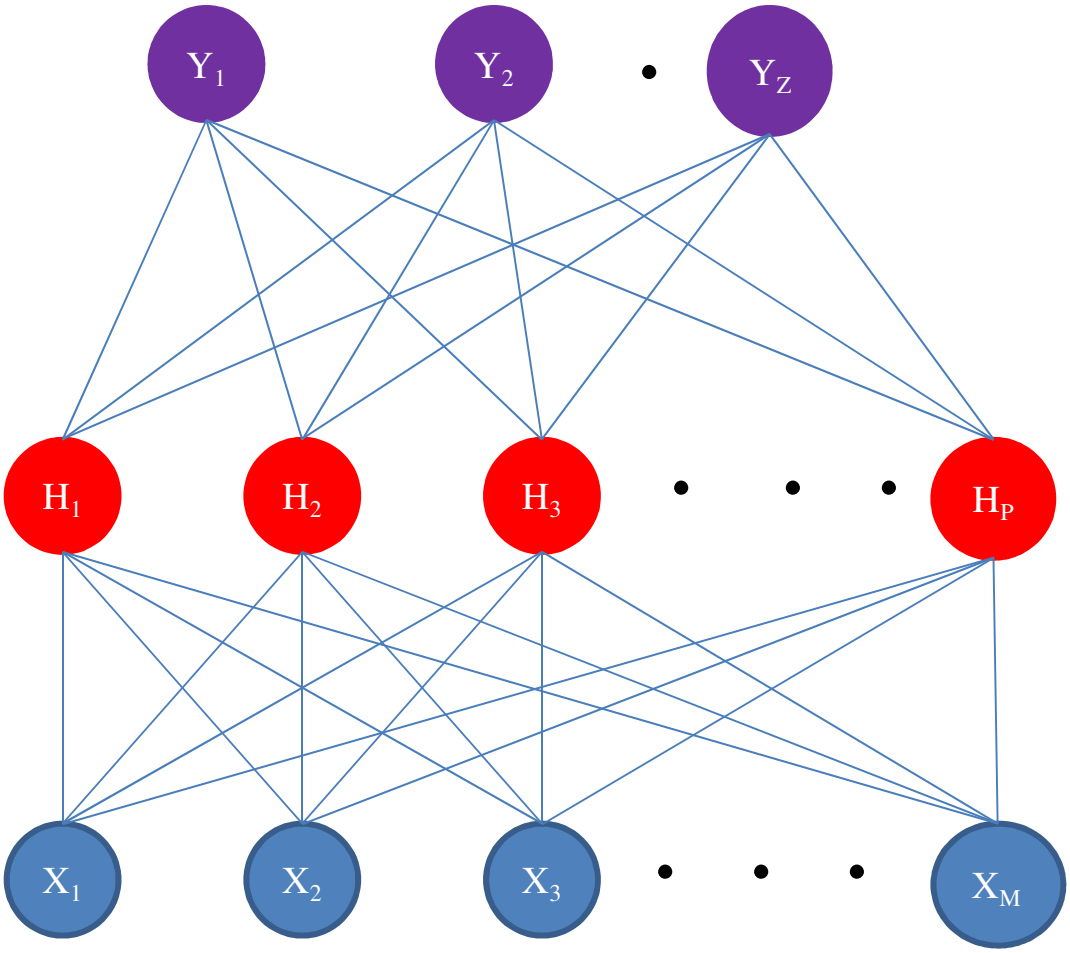


Figure 3. 5 Architecture of single hidden layer, feed-forward neural network

Note that in the above equation, if it is assumed that the Gaussian kernel is centered at a labeled training point, we can evaluate the Gaussian basis function for a test point x_{Te}. The value $f(x_{Te})$ is now an indicator of how far the test point lies from a training point '*probabilistically*'. If we denote the training point as x_{Tr}, then Eq.(3.4) takes the form of Eq.(3.5)

$$f(x_{Te}) = \exp(-\frac{(x_{Te} - x_{Tr})}{2\sigma^2}) \tag{3.5}$$

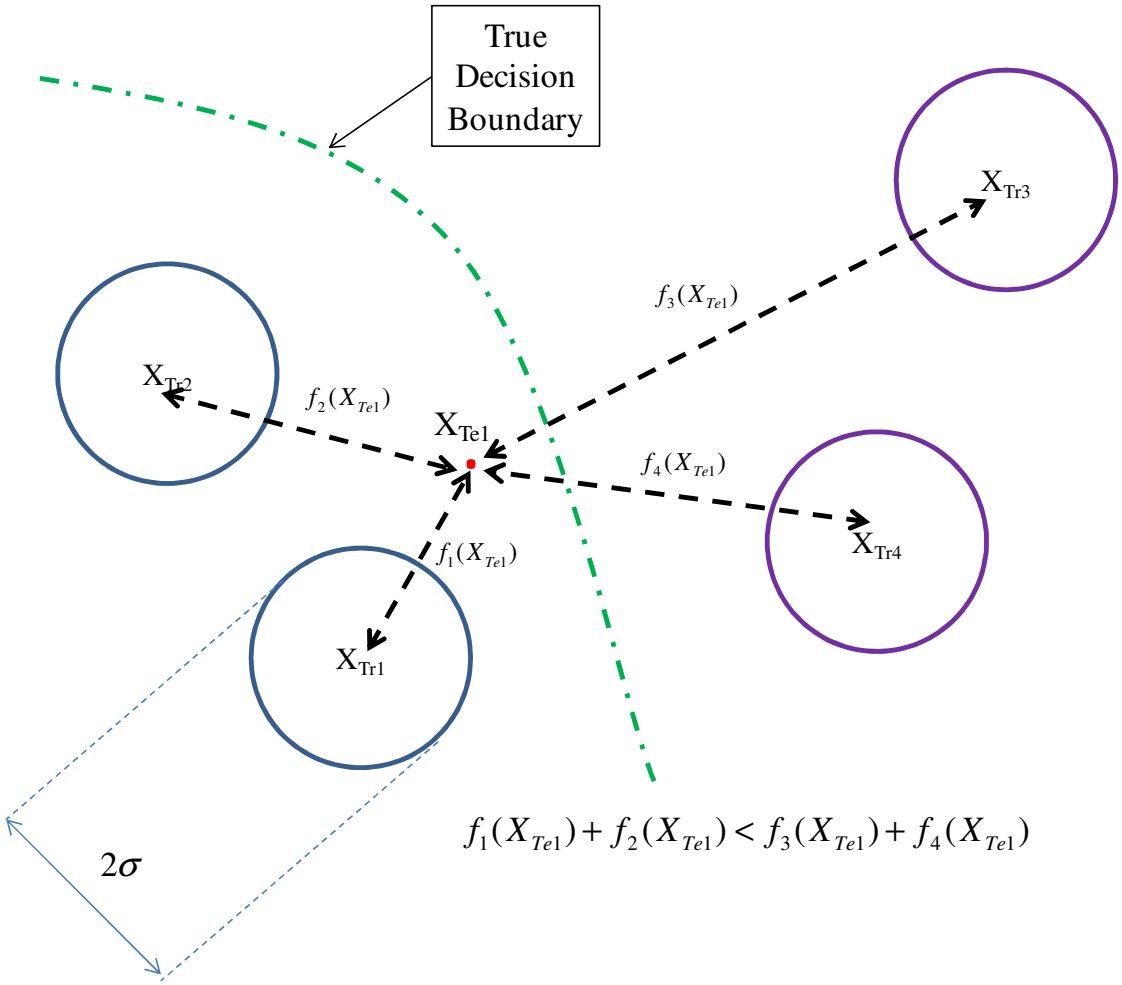


Figure 3. 6 Illustration of probabilistic classification process

If we take the '*probabilistic*' distance as a measure of classifying a test point into a class, we can add the probabilistic distance of a test point from all the training points belonging to either class and assign the test point to a class that has the lowest total. This total is an indicator of how far the test point is from each class. This is the basic idea behind classification by all feed-forward radial basis function neural networks. This process is represented in Figure 3. 6.

An analogous way of looking at the classification idea explained above is to think about it as a two-step process. In the first step, the PDF function of each class is modeled as a combination of Gaussian distributions. In other words, the PDF of each class is the sum of the spherical Gaussian distributions that are centered at the labeled points belonging to that particular class. This process is stated with estimated PDFs of Classes 1 and 2 in Figure 3. 7.

Once the PDFs are estimated, the test point is evaluated for each PDF to find the probability that the test point belongs to a particular class. The class that has the higher probability is the label for the test point. In the particular case shown in Figure 3. 7 the test point is classified to class 1 and given the corresponding label.

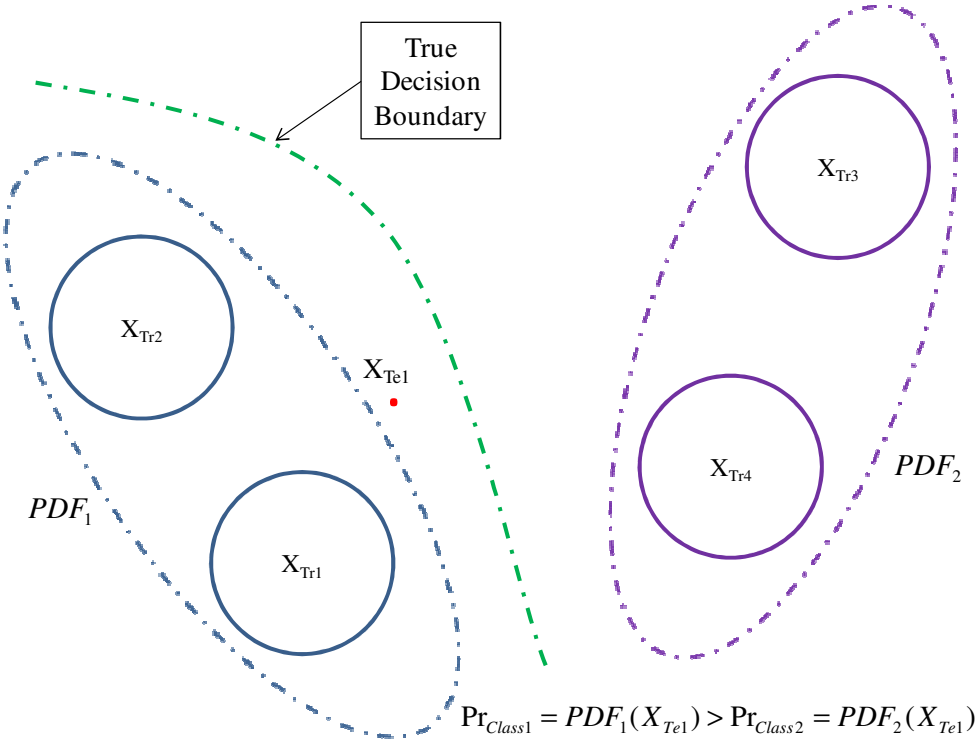


Figure 3. 7 Magnified shapes of PDFs of both classes

As mentioned in Chapter 2, generative classifiers model the PDFs of each class and then find the probability of a test point belonging to each class using these PDFs. The test point is given a label of the class that has the maximum posterior probability. Hence, the radial-basis function ANNs are generative classifiers and are capable of modeling any kind of complex decision boundary since the PDF functions can take complex shapes. In this research we primarily focus on Probabilistic Neural Networks, which are special kinds of radial-basis function feed-forward ANNs. A detailed description of these ANNs is provided in the next section.

3.2.4 Classification using Probabilistic Neural Networks (PNN)

The PNN has been successfully used for diverse pattern recognition applications such as image recognition, texture recognition, signal processing, finance, and biomedical applications [103, 104]. The PNN is a pattern classifier that combines the widely used Bayes decision rule with the Parzen nonparametric estimator [105] for the estimation of probability density functions of different classes [106]. Unlike other neural network architectures, PNN is relatively simple to implement and the network is easily interpretable. PNN comes into the category of *generative classifiers* [95] since it estimates the class conditional PDFs before assigning a data point to a class with maximum class conditional probability value. The decision rules for generative classifiers use Bayes decision rule in order to reduce the "expected risk" in pattern classification [107].

Consider an instance x_i that belongs to either of the two classes A or B. If a decision of whether x_i belongs to class A or class B has to be made based on the data represented in the m-dimensional vector $X^T=[X_1\ X_2 \ldots\ X_j \ldots\ X_m]$, the Bayes decision rule is represented by

$$x_i \in A \text{ if } h_A l_A f_A(x_i) > h_B l_B f_B(x_i) \tag{3.6}$$

$$x_i \in B \text{ if } h_A l_A f_A(x_i) < h_B l_B f_B(x_i) \tag{3.7}$$

where $f_A(X)$ and $f_B(X)$ are class conditional PDFs for categories A and B, respectively. l_A is the loss function associated with the decision that x_i belongs to class B when x_i actually belongs to class A, and l_B is the loss function associated with the decision that x_i belongs to class A when x_i actually belongs to class B. h_A is the a priori probability of occurrence

of patterns from category *A*, and $h_B = 1 - h_A$ is the a priori probability of occurrence of patterns from category *B*. h_A is simply the ratio of the number of patterns from Class A in training data and the total number of patterns in the training data.

In general, a quadratic loss function can be used for many classification algorithms. For this scenario the quadratic loss function can be given as

$$l = (h_A - P)^2 + (h_B - P)^2 \tag{3.8}$$

where $l = l_A$ or l_B correspond to $P = P_A$ or P_B. Here $P_A + P_B = 1$ and $P_A, P_B \in \{0,1\}$, depending on whether the data point belongs to class *A* or class *B*.

The Bayes Decision boundary is given by

$$f_A(X) = Qf_B(X) \tag{3.9}$$

where $f_A(X)$ and $f_B(X)$ are the class conditional probabilities of class *A* and *B*, respectively.

And the constant, $Q = \frac{h_B l_B}{h_A l_A}$ (3.10)

In theory, the decision boundary represented by Eq. (3.9) can be fairly complex since there is no restriction on the densities except for the conditions that all PDFs must adhere to. These conditions imply that all PDFs should be non-negative and integrable, and their integral over the whole domain should equal one. A similar Bayes Decision rule can be established for many category problems as well [106].

In cases when the a priori probabilities are equal to each other and the loss functions are assumed to be the same, Bayes rule classifies an input pattern to the class that has its class conditional PDF greater than the class conditional PDF of the other class

for that input pattern. Hence, the effectiveness of this procedure depends on the accuracy of the PDF estimation. The first step is the computation of the PDFs, $f_A(X)$ and $f_B(X)$, in order to compute the decision boundary. The procedure of construction of a family of estimates of the PDF, $f(X)$, was shown by Parzen and Cacoullos [108]. It extends Parzen's results to the case where a multivariate kernel is a product of univariate kernels. The PDFs can be computed using a Parzen window considering a multivariate kernel. When a Gaussian kernel is used, the multivariate estimate for class-conditional PDF of class A can be expressed as

$$f_A(X) = \frac{1}{(2\pi)^{m/2}|\Sigma|^{1/2}} \frac{1}{n_A} \sum_{i=1}^{n_A} \exp\left[-\frac{(X - X_{TAi})^T (X - X_{TAi})}{2\sigma^2} \right] \tag{3.11}$$

where X is the vector to be classified, $f_A(X)$ is the value of the PDF of category A at point X, n_A denotes the number of patterns in category A, m is the dimensionality of the training patterns, X_{TAi} is the i^{th} training pattern from category A, and K is the covariance matrix or smoothing parameter. Note that in case of PNN the covariance matrix is 'spherical' or of the form $\Sigma = \sigma^2 I$. It is important to note that the task assigned to the classifier is to classify the dataset X (the test data) after it is trained using the training dataset X_T. X_T is comprised of X_{TA} and X_{TB} corresponding to whether the data point belongs to class A or class B. This information is available apriori since X_T is the training dataset. The class-conditional PDF of class A, $f_A(X)$ can be determined by summing the multivariate Gaussian distributions centered at each training sample. However, the kernel function chosen to compute the PDFs is not limited to being Gaussian. Choosing a different kernel would not lead to a worse classifier [106]. The PDF functions from Eq. (3.11) can be

used to compute the class conditional probabilities in the summation layer of the PNN for *X*.

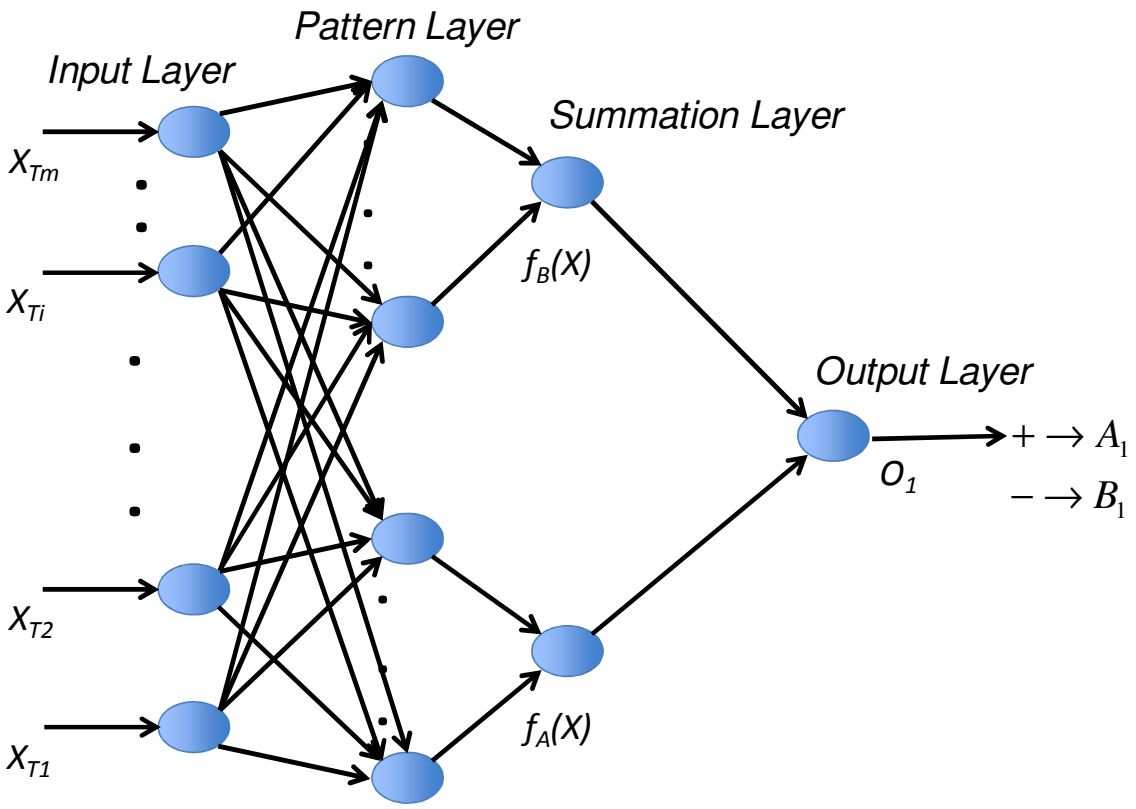


Figure 3. 8 Architecture of probabilistic neural network

Figure 3. 8 shows the architecture of a PNN for classifying the vector *X* into two classes—*A* and *B*. It consists of four different layers, including the input layer, pattern layer, summation layer and the output layer. Assuming that the training dataset X_T consists of *n* data points, each containing *m* dimensions, the PNN network will have *m* neurons in the input layer, *n* neurons in the pattern layer and 2 neurons in the summation layer. The input units are merely distribution units that provide the same input values to all the pattern units.

As shown in Figure 3. 8, the second layer of the PNN architecture is the pattern layer which can be interpreted as shown in Figure 3. 9. In the pattern layer, the first step of training the network is to set the weights in the pattern units equal to the training dataset X_T.. Each pattern unit then forms a dot product of the input to pattern layer, vector X with the weight vector X_{Ti}, namely, $Z_i = X \cdot X_{Ti}$. Then, the nonlinear operation depicted in Eq. (3.11) can be performed on Z, before passing the output of this step to the summation unit (Figure 3. 10).

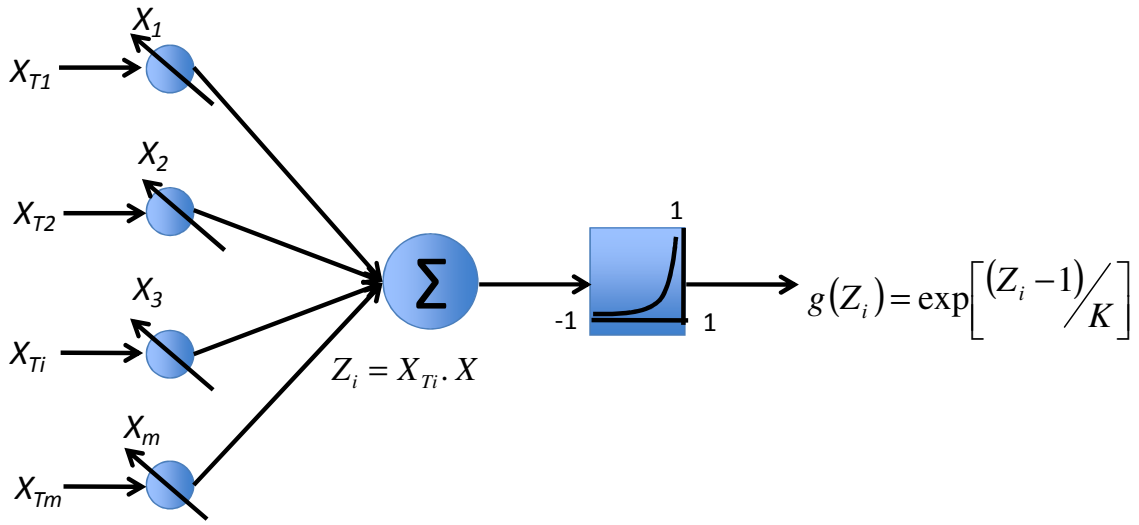


Figure 3. 9 Pattern Layer of PNN

In contrast to the sigmoid transfer function [41] that is generally used for backpropagation neural networks, the transfer function used in this PNN is the exponential function, namely, $g(Z_i) = exp\ [(Z_i-1)/\Sigma]$. If both X and X_{Ti} are normalized to unit length, the nonlinear transfer function can be expressed as

$$g(Z_i) = \exp\left[-\frac{(X_i - X_T)'(X_i - X_T)}{2\Sigma}\right] \tag{3.12}$$

Once the transfer function is calculated, the outputs corresponding to the class *A* and *B* can be summed together in the summation layer to compute the PDF using the Parzen window method according to Eq. (3.11).

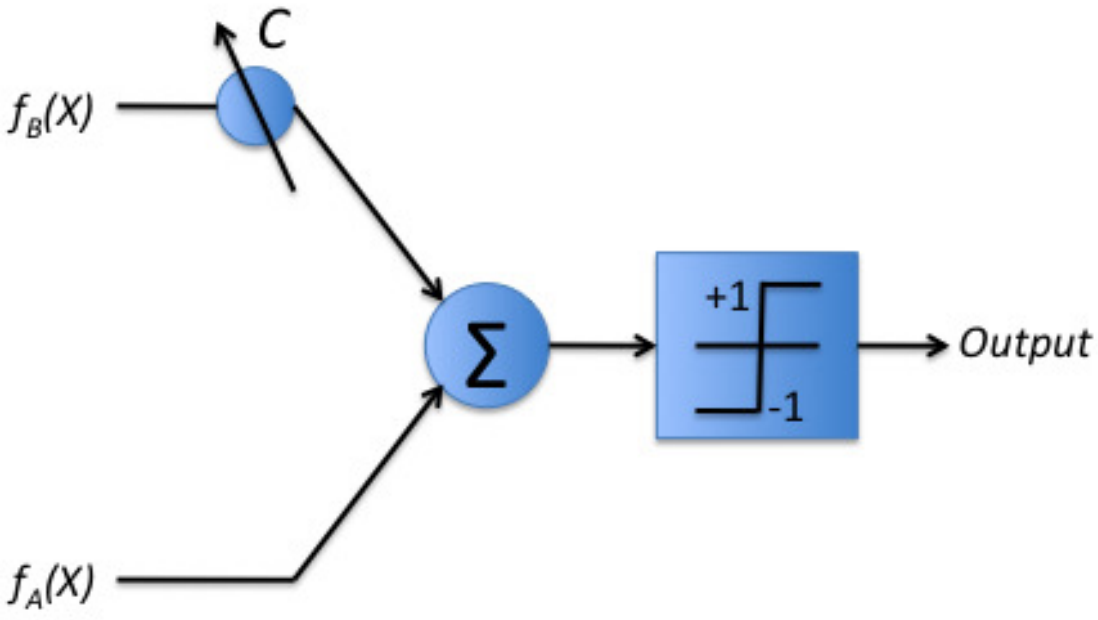


Figure 3. 10 Summation layer of PNN

Figure 3. 10 represents the summation layer which is used to calculate the class or category PDFs from Eq. (3.11). This step involves connecting the pattern unit's output to the appropriate summation unit. Every training pattern requires a separate neuron (in the pattern unit) which has one unique connection to a neuron in the summation layer. Each neuron in the summation layer corresponds to a different class and it sums the inputs that correspond to that particular class from the neurons in the pattern layer. The same pattern units can be grouped by different summation units to provide additional pairs of categories and additional bits of information in the output vector. This is illustrated in Figure 3. 8 for the case where all the pattern layers are grouped into two categories for the classification of *X* into one of the two classes, *A* or *B*. Furthermore, once the class-

conditional PDFs $f_A(X)$ and $f_B(X)$ are computed, the Bayes decision criteria can be evaluated by the following equations

$$d(X) = class\ A \text{ if } f_A(X) + Qf_B(X) > 0 \tag{3.13}$$

$$d(X) = class\ B \text{ if } f_A(X) + Qf_B(X) < 0 \tag{3.14}$$

$$\text{and } Q = -\frac{h_B l_B}{h_A l_A} \cdot \frac{n_A}{n_B} \tag{3.15}$$

where n_{A_k} and n_{B_k} are the number of training patterns from category *A* and the number of training patterns from category *B*, respectively.

It can be seen from Eq. (3.15) that *Q* is the ratio of a priori probabilities divided by the ratio of samples and multiplied by the ratio of losses. Thus, if the number of training samples from categories *A* and *B* are in proportion to their a priori probabilities, $Q = -l_B / l_A$. The final ratio, *Q*, cannot be calculated from the statistics of the training samples alone, but only by the significance of the decision. If there are no strong reasons for biasing the decision, then *Q* can be simplified to -1. Substituting this value of Q in Eq. (3.13) and Eq. (3.14), the decision boundary changes to a comparison of class conditional posterior probabilities for a test pattern.

$$d(X) = class\ A \text{ if } f_A(X) - f_B(X) > 0 \tag{3.16}$$

$$d(X) = class\ B \text{ if } f_A(X) - f_B(X) < 0 \tag{3.17}$$

Note that as per our discussion in the previous section and in Chapter 2, a PNN is a generative classifier because the classification is divided into two steps. In the first step the class-conditional PDF is calculated for both the classes and then the probability

values for both classes are calculated by using the class-conditional PDFs. The test point is then classified to the class, which has the higher probability.

3.2.5 Limitations of Probabilistic Neural Networks (PNN)

There are two limitations when using PNN for classification. The first limitation is posed by the inability of the user to know the optimum value of smoothing parameter (σ^2) when using the classical PNN using Eq. (3.11). The width of the Gaussian kernel determines how much influence a particular Gaussian kernel, centered at a labeled data point, has on a test point. A particular example is given by Specht [106] that demonstrates the different shapes that $f_A(x)$ (class-conditional PDF of class A) can take with different values of σ (Shown in Figure 3. 11). Furthermore, Specht [109] also showed that PNN's results are comparable to what is obtained by the k-nearest neighbor (k-NN) algorithm [101] when the value of smoothing parameter tends to zero. Figure 3. 11 also shows that as the smoothing parameter values are small multimodal Gaussian distribution is possible for class-conditional PDF and as the smoothing parameter value is large the PDF is Gaussian, irrespective of the underlying true PDF of the labeled dataset. In the case of the lower value of smoothing parameter, PNN will have a very local, nonlinear nature where the decision boundary is highly influenced by the local characteristics of the decision surface whereas as the value is high, PNN is insensitive to the local characteristics of the decision surface and tends to approximate the global nature of the decision criteria. Either of these could be the right decision boundary—unfortunately, the level of optimum local/global nature of the classifier cannot be estimated when labeled data are difficult to obtain.

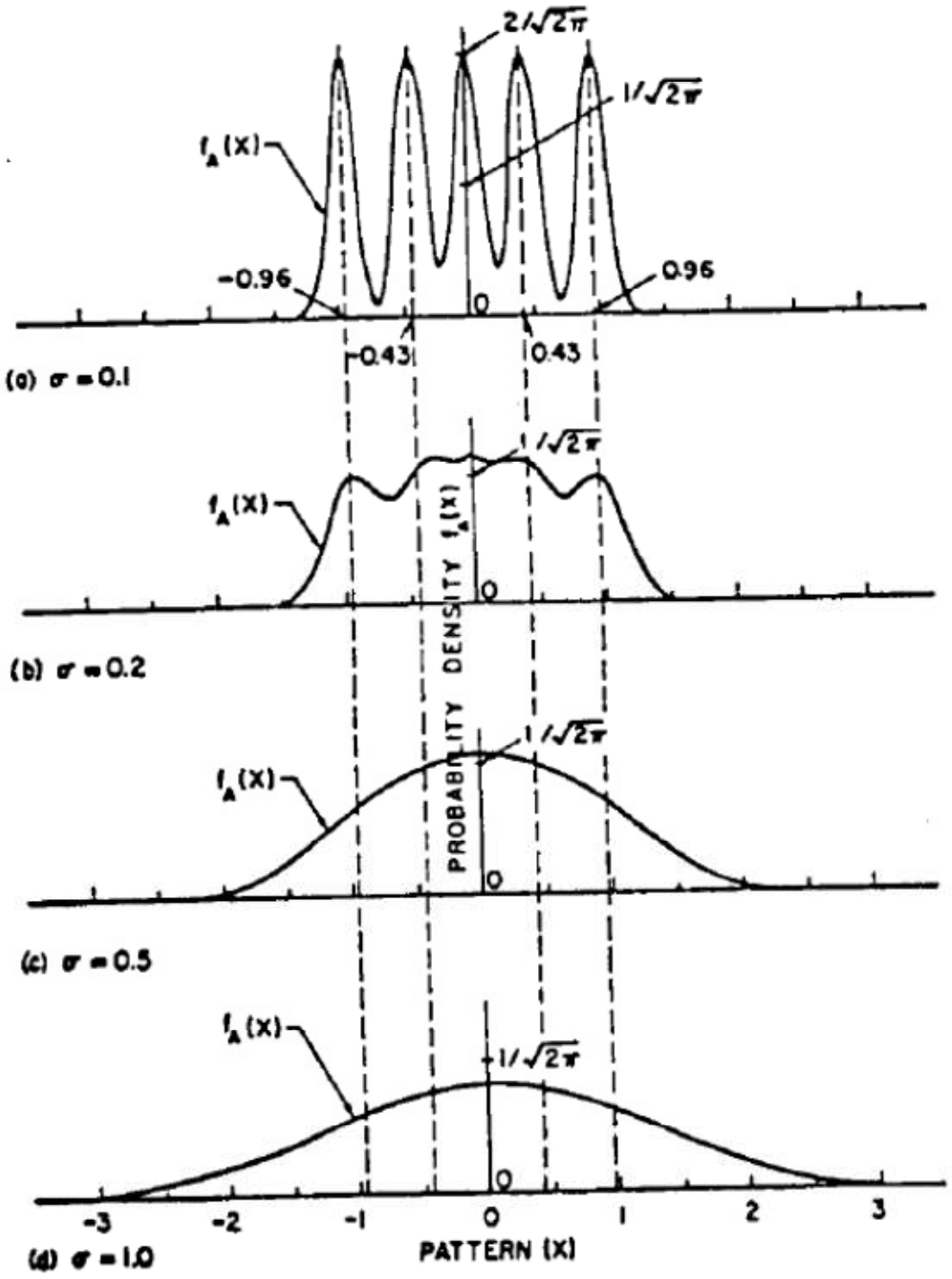


Figure 3. 11 Effect of different values of smoothing parameter on $f_A(x)$ in a one dimensional problem[106]

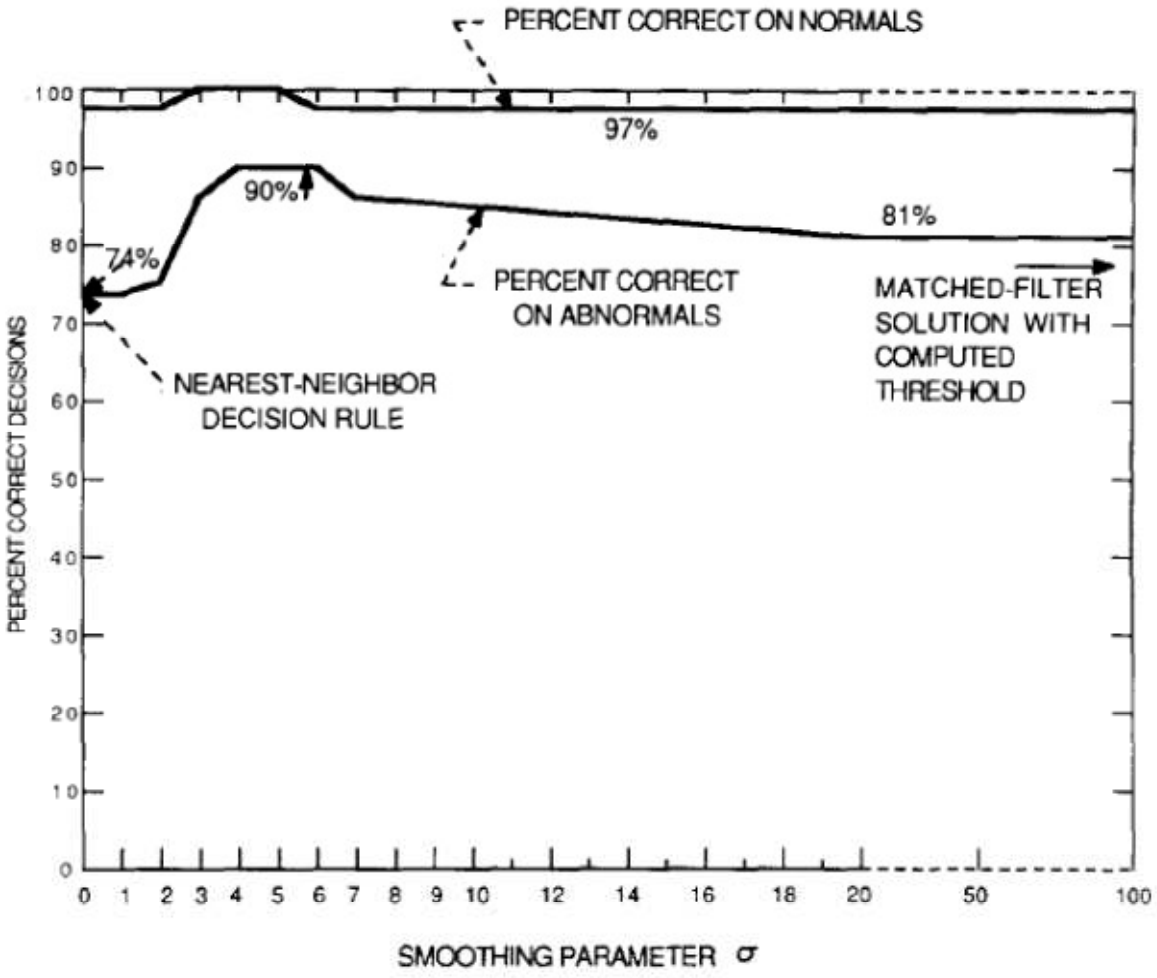


Figure 3. 12 Effect of different smoothing parameter values on PNN's performance[109]

The second limitation of the classical form of PNN is that the covariance matrix is 'spherical'. In other words, the covariance matrix is of the form $\Sigma = \sigma^2 I$, where I represents an identity matrix of size m. Assuming a spherical covariance matrix is a simple way to estimate a PDF function when the real PDF function is unknown. Our tests show that the 'spherical covariance' assumption is a safe one to make when a large number of labeled data is available. But when a limited number of labeled data is available, as in case of reliability estimation problems, PNN can give spurious results.

We hypothesize that these two problems can be alleviated by relieving the 'spherical' covariance assumption and allowing 'full' covariance matrices when computing the class conditional PDFs of classes in PNN. An example for 'spherical' covariance matrix and 'full' covariance matrices are shown in Eqs.(3.18) and (3.19) respectively.

$$\Sigma = \sigma^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{3.18}$$

$$\Sigma = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 \\ \sigma_{12}^2 & \sigma_{22}^2 \end{bmatrix} \tag{3.19}$$

If we allow a 'full' covariance matrix in the computation of the class-conditional PDFs, the Gaussian Kernel shown in Eq. (3.11) takes a more general form as shown in Eq. (3.20).

$$f_A(X) = \frac{1}{(2\pi)^{m/2}|\Sigma|^m} \frac{1}{n_A} \sum_{i=1}^{n_A} \exp\left[-\frac{1}{2}(X - X_{TAi})^T \Sigma^{-1} (X - X_{TAi}) \right] \tag{3.20}$$

We hypothesize that in the presence of unlabeled data, more information about the real PDF can be gathered from the unlabeled data itself, which may lead to better estimates of covariance matrices for each training pattern. Hence, in the presence of unlabeled data, we can release this constraint of 'spherical' covariance matrix and assume a 'full' covariance matrix so that each dimension in our dataset is allowed to have different variance values and cross-dependencies between different dimensions are allowed. The question that arises is how can unlabeled data be used to estimate the parameters of the 'full' covariance matrix when a large amount of unlabeled data is present? The answer is

provided by '*unsupervised learning*', which is a machine learning algorithm which only uses unlabeled data.

3.3 Unsupervised Learning

The goal of unsupervised learning is to discover an interesting structure in the '*unlabeled*' data *X*. Quantile estimation, clustering, outlier detection and dimensionality reduction are some of the methods that are used for unsupervised learning [110]. A simple clustering example is shown in Figure 3. 13, where a set of unlabeled data is grouped into two Gaussian clusters and the true PDFs from which the data is sampled are assumed to be unknown.

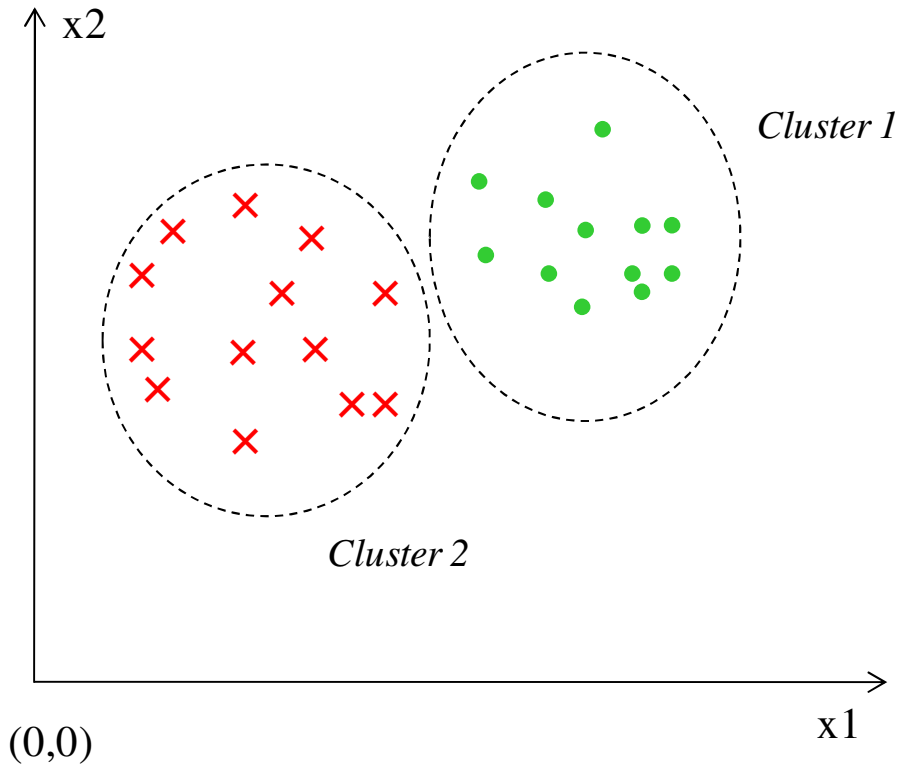


Figure 3. 13 Unlabeled data is separated into two constituent Gaussian components after unsupervised learning

In order to estimate the parameters of Gaussians, *clustering*-unsupervised learning algorithms can be used where the task is defined as given that the shapes of the components are Gaussians, what are the best estimates of the Gaussian parameters? The following sections elaborate on this form of machine learning algorithm.

3.3.1 Clustering as Unsupervised Learning

In clustering, there is no explicit teacher, and the system forms clusters or "natural groupings" of the input patterns [94]. "Natural" is always defined explicitly or implicitly in the clustering system itself, and given a particular set of patterns or cost function; different clustering algorithms lead to different clusters. In most of the clustering algorithms, before running the algorithm, the user has to hypothesize the number of different clusters and the shape of the clusters ahead of time. Once the number of clusters and the parameters of the shape of the clusters are initialized, the unsupervised learning algorithms can learn the optimum values of the shape parameters of the clusters. With this, learning the shape of the clusters essentially boils down to finding the parameters of a PDF. If an assumption is made that the set of unlabeled patterns come from a mixture of Gaussian PDFs, the problem of estimating the PDF essentially boils down to finding the mixing parameters, means and covariance of each of the component Gaussian PDFs.

In the case of a 2-class problem such as the problem of probability of failure estimation, the data can be assumed to be sampled from two different Gaussian PDFs. However, the correspondence of data points to particular Gaussian PDFs is unknown. In order to define the two Gaussian PDFs, it is essential to estimate the means and variances of these individual distributions. In addition to the means and variances, a mixing

parameter would be required to properly define the density of the whole dataset so that the mixing parameter λ defines the density of the data according to

$$f(x) = \lambda N(x;\mu_1,\Sigma_1) + (1-\lambda) N(x;\mu_2,\Sigma_2), \qquad \lambda \in [0,1] \tag{3.21}$$

Once the parameters (λ, μ_1, Σ_1 and μ_2, Σ_2) are determined, the density estimation process is complete. Note that λ is the mixing parameter for the two Gaussian PDFs. In case of a mixture of more than two Gaussian PDFs, the mixture parameters for all the Gaussian PDFs should add up to 1. This problem of parameter estimation is a classical problem in statistics and it can be approached in several ways. Two commonly used procedures for this are the *maximum likelihood* estimation (MLE) and *Bayesian* estimation. Even though the results obtained from both methods have been reported to be nearly identical in most cases the approaches are conceptually different [94]. MLE and several other methods view the parameters as quantities whose values are fixed but unknown. In MLE the best estimate of the parameters is defined as the one that maximizes the probability of obtaining the samples actually observed. On the other hand Bayesian methods assume that the parameters are random variables having some kind of priori distribution. New observations convert this priori distribution to posterior density, which helps us revise our opinion of the true values of the parameters. Each additional sample sharpens the posteriori density function causing it to peak near the true values of the parameters. More details of Bayesian Learning can be found in Ref. [94]. In this research we will take the MLE approach to parameter estimation since MLE is computationally more efficient than Bayesian methods and the models predicted by MLE are easier to interpret [94]. Furthermore, unlike least square minimization based

parameter estimation where the errors are ssumed to come from a standard normal distribution, MLE doesn't assume any inherent distribution for the errors.

3.3.2 Maximum Likelihood Estimates for Mixture Models

Suppose that we have a collection of samples forming a set $D = \{X_1, ..., X_u\}$ of u unlabeled samples drawn independently from the mixture density consisting of two components

$$p(X \mid \theta) = \sum_{i=1}^{2} p(X \mid w_i, \theta_i) P(w_i) \tag{3.22}$$

where the full parameter vector $\theta = (\theta_1, \theta_2)$ is fixed but unknown. A density function resembling Eq. (3.22) is called a mixture density and the conditional densities $p(X \mid w_i, \theta_i)$ are called the *component densitites*, and the *prior probabilities* $P(w_i)$ are called the *mixing parameters*. Recall that Eq. (3.22) resembles Eq. (3.21) when the component densities are assumed to be Gaussian PDFs. In that case the unknown parameter vector will consist of the mean and covariance of each component. Our problem is to use the information provided by the training samples to obtain good estimated for the unknown parameter vectors θ_1 and θ_2 associated with each known component $p(X \mid w_i, \theta_i)$.

To simplify the solution of this problem, it is generally assumed that the parameters for different components are functionally independent. This permits working with each component independently. Once independence is assumed the joint probability of the data can be expressed as

$$p(D \mid \theta) = \prod_{j=1}^{u} p(X_j \mid \theta) \tag{3.23}$$

If viewed as a function of θ, $p(D \mid \theta)$ is called the likelihood of θ with respect to the set of samples. By definition, the maximum likelihood estimate of θ is the value $\hat{\theta}$ that maximizes $p(D \mid \theta)$. This estimate of θ agrees with or supports the actual observed training samples. Since Eq. (3.23) represents a product of many terms, it is easier to work the the logarithms for analytical purposes. Since logarithm is monotonically increasing, the $\hat{\theta}$ that maximizes the log-likelihood also maximizes the likelihood. An example from Duda et. al. [94] is shown in Figure 3. 14, to illustrate the behavior of likelihood and log-likelihood (represented by $l(\theta)$).

The top picture in the figure shows several training points in one dimension, where it is assumed that the points are drawn from a Gaussian distribution with unknown mean but fixed variance. Only four of the many Gaussian distributions are shown in dashed lines. The middle picture shows the likelihood function as it varries with resepect to the mean and the value, $\hat{\theta}$ for the mean, that maximizes the log likelihood is shown in the bottom picture.

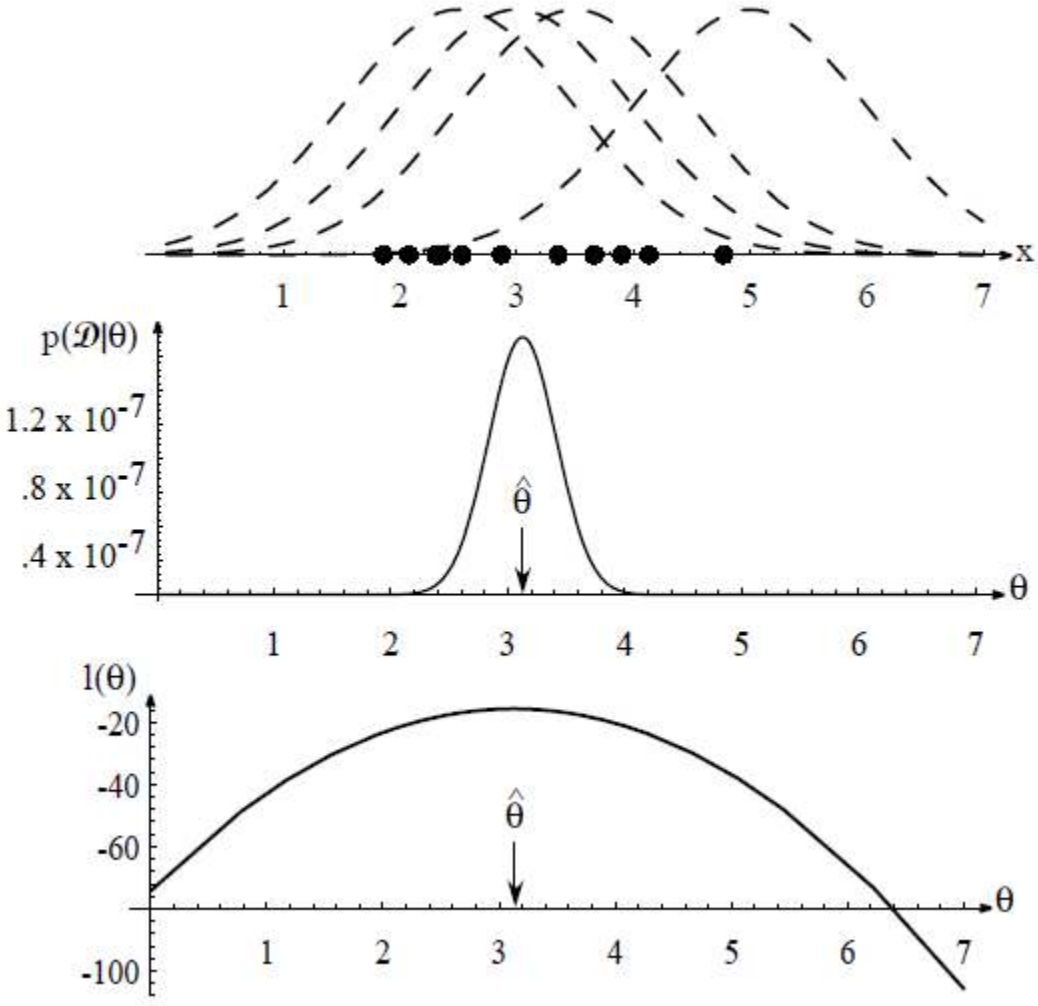


Figure 3. 14 Duda et. al. example on convergence characteristics of MLE [94]

If we assume that $p(D \mid \theta)$ in Eq. (3.23) is differentiable function of θ then we can derive some necessary conditiona for $\hat{\theta}$. If we take logarithm on both sides of Eq. (3.23) then we have

$$l = \sum_{j=1}^{u} \ln p(X_j \mid \theta) \tag{3.24}$$

where l represents the logarithm of the likelihood. If we want to find the gradient of l with respect to each component of θ, θ_k, (where k = 1 or 2 for a 2 component mixture) the gradient $\nabla\theta_k l$ can be written as

$$\nabla\theta_k l = \sum_{j=1}^{u} \frac{1}{p(X_j \mid \theta)} \nabla\theta_k \left[\sum_{i=1}^{2} p(X_j \mid w_i, \theta_i) P(w_i) \right] \tag{3.25}$$

If we assume that the elements of θ_j and θ_k are functionally independent if $j \neq k$, and we introduce the posterior probability from Eq. (3.26) we get a different way of writing the log-likelihood as shown in Eq. (3.27).

$$P(w_k \mid X_j, \theta) = \frac{p(X_j \mid w_k, \theta_k) P(w_k)}{p(X_j \mid \theta)} \tag{3.26}$$

$$\nabla\theta_k l = \sum_{j=1}^{u} P(w_k \mid X_j, \theta) \nabla\theta_k \ln p(X_j \mid w_k, \theta_k) \tag{3.27}$$

Since the gradients of l calculated in Eq. (3.27) must be equal to zero where l is maximized, the following condition follows

$$\sum_{j=1}^{u} P(w_k \mid X_j, \hat{\theta}) \nabla\theta_k \ln p(X_j \mid w_k, \hat{\theta}_k) = 0 \text{ for } k = 1,2. \tag{3.28}$$

since for this case only two clusters are considered. The posterior probability in Eq. (3.28) can be stated as

$$P(w_k \mid X_{j,} \hat{\theta}) = \frac{p(X_j \mid w_k, \hat{\theta}_k) \hat{P}(w_k)}{\sum p(X_j \mid w_i, \hat{\theta}_i) \hat{P}(w_i)} \tag{3.29}$$

Next, if we consider that the prior probabilities are also unknown and they satisfy the two conditions, $\sum_{j=1}^{2} P(w_j) = 1$ and $P(w_j) \geq 0$, then it can also be shown that for maximum likelihood estimate

$$\hat{P}(w_j) = \frac{1}{u} \sum_{j=1}^{u} \hat{P}(w_j \mid X_i, \hat{\theta}) \tag{3.30}$$

We can interpret Eq. (3.30) by visualizing that the maximum-likelihood estimate of the probability of a category is the average over the entire data set of the estimate derived from each sample. Hence each sample is weighted equally when calculating the maximum likelihood estimate of the prior probability.

Eqs (3.28-3.30) can be used to estimate the parameters of a mixture of any kind of distributions. For this research we will only need to study the MLE for Gaussian mixtures, which is explained in the next section.

3.3.3 Maximum Likelihood Estimates for Gaussian Mixtures

In case of mixture of Gaussians Eqs. (3.28-3.30) can be used in order to estimate the unknown parameters of the component multivariate normal distributions, $p(X \mid w_k, \theta_k) \sim N(\mu_k, \Sigma_k)$. The following three cases can arise when we study mixture of Gaussians PDF models:

- Case 1: Mean, μ_k (unknown); Covariance, Σ_k (known); Prior Probabilities, $P(w_k)$ (known) and No. of clusters, Z (known);

- Case 2: Mean, μ_k (unknown); Covariance, Σ_k (unknown); Prior Probabilities, $P(w_k)$ (unknown) and No. of clusters, Z (known);
- Case 3: Mean, μ_k (unknown); Covariance, Σ_k (unknown); Prior Probabilities, $P(w_k)$ (unknown) and No. of clusters, Z (unknown);

Case 1 is the simplest since only one parameter is unknown whereas Case 2 is involved but useful in most of the conditions that we will be using in this research. Case 3 is not solvable using maximum likelihood methods and not useful for the conditions that are considered for unsupervised learning in the present research. Hence, the methods used to solve the problems involving Case 3 will not be considered in the present research.

Case1: In this case the only unknown values in the parameter vector θ_k are the components of μ_k. The likelihood of a multivariate normal distribution will be given by

$$\ln p(X \mid w_i, \mu_i) = -\ln\left[(2\pi)^{m/2}\left|\Sigma_k\right|^{1/2}\right] - \frac{1}{2}(X-\mu_k)^T \Sigma_k^{-1}(X-\mu_k) \tag{3.31}$$

and its derivative will be given by

$$\nabla\mu_k \ln p(X \mid w_k, \mu_k) = \Sigma_k^{-1}(X-\mu_k) \tag{3.32}$$

Hence, according to Eq. (3.28), the maximum likelihood estimate $\hat{\mu}_k$ must satisfy

$$\sum_{j=1}^{u} P(w_k \mid X_j, \hat{\mu})\Sigma_k^{-1}(X_j - \hat{\mu}_k) = 0 \text{, where } \hat{\mu} = (\hat{\mu}_1, \hat{\mu}_2) \tag{3.33}$$

for a mixture of two Gaussians.

Rearranging the above equation, we obtain the expression for $\hat{\mu}_k$

$$\hat{\mu}_k = \frac{\sum_{j=1}^{u} P(w_k \mid X_j, \hat{\mu}) X_j}{\sum_{j=1}^{u} P(w_k \mid X_j, \hat{\mu})} \tag{3.34}$$

Notice that it is intuitively easy to explain the result obtained in Eq. (3.34). In essence, this equation says that the maximum likelihood estimate for μ_k is merely a weighted average of all the unlabeled samples. The weight for the *j*th sample is an estimate of how probable it is that X_j belongs to the *k*th cluster. More specifically, if $P(w_k \mid X_j, \hat{\mu})$ is 1.0 for some of the samples and 0 for others then $\hat{\mu}_k$ would be the mean of those samples that are estimated to belong to the *k*th cluster. Unfortunately, Eq. (3.34) does not explicitly give the value of μ_k but if we happen to get good initial estimates, $\mu_k(0)$ for the unknown means, Eq. (3.34) gives us the following iterative scheme for improving the estimate:

$$\hat{\mu}_k(t+1) = \frac{\sum_{j=1}^{u} P(w_k \mid X_j, \hat{\mu}(t)) X_j}{\sum_{j=1}^{u} P(w_k \mid X_j, \hat{\mu}(t))} \tag{3.35}$$

This method is just like any other hill-climbing algorithm and doesn't guarantee convergence to global maxima. If the overlap between component densities is small, then the coupling between classes will be small and convergence will be fast. However, even when convergence occurs, we can only be sure than the gradient is zero [94]. Furthermore, if the model is misplaced (the wrong number of clusters is guessed) then the log-likelihood might decrease.

Case 2: In this case all the parameters, mixing parameters (prior probabilities), means and covariances, are unknown. This is the more general case which is frequently encountered in practice. Since all the parameters are unknowns, if no constraints are placed on the covariance matrix, the maximum-likelihood principle yields singular solutions[94]. Since singular solutions are of no interest, we can conclude that the maximum-likelihood principle fails for this class of normal mixtures. However, empirically it has been shown that meaningful solutions can still be found if we restrict the attention to the largest of the finite local maxima of the likelihood function. Assuming that the likelihood function is well behaved at the local maxima of the likelihood function, we can use Eqs. (3.28-3.30) to obtain estimates for μ_k, Σ_k and $P(w_k)$. Note that when we include the elements of Σ_k, in the elements of the parameter vector θ_k, we only need to include half of the off-diagonal elements since a covariance matrix is symmetrical. Again, it is easier to estimate the parameters of Σ_k^{-1} rather than the parameters of Σ_k.

Similar to Eq. (3.31) the log likelihood of the multivariate normal distribution is given by

$$\ln p(X \mid w_i, \mu_i) = -\ln\left[(2\pi)^{m/2}\left|\Sigma_k\right|^{1/2}\right] - \frac{1}{2}(X-\mu_k)^T \Sigma_k^{-1}(X-\mu_k) \tag{3.36}$$

In the next step we can differentiate this equation to find the maximum likelihood estimates of the mean and the inverse covariance matrix. Let us denote the p th element of X_j by $x_p(j)$, the pth element of μ_k by $\mu_p(k)$ and the pqth element of Σ_k by $\sigma_{pq}(k)$. The

differentiation of log likelihood with respect to mean and different elements of covariance matrix is given below.

$$\nabla \mu_k \ln p(X_j \mid w_k, \theta_k) = \Sigma_k^{-1}(X_j - \mu_k) \tag{3.37}$$

and

$$\frac{\partial \ln p(X_j \mid w_k, \theta_k)}{\partial \sigma^{pq}(k)} = \left(1 - \frac{\delta_{pq}}{2}\right)\left[\sigma_{pq}(k) - \left(x_p(j) - \mu_p(k)\right)\left(x_p(j) - \mu_p(k)\right)\right] \tag{3.38}$$

where δ_{pq} is the Kronecker delta. Substituting these results in Eq. (3.28), which is restated below, we can derive the expressions for maximum likelihood estimates of prior probabilities, means and covariances as shown in Eq. (3.39-3.43).

$$\sum_{j=1}^{u} P(w_k \mid X_j, \hat{\theta}) \nabla \theta_k \ln p(X_j \mid w_k, \hat{\theta}_k) = 0 \text{ for } k = 1,2. \tag{3.28}$$

$$\hat{P}(w_k) = \frac{1}{u} \sum_{j=1}^{u} \hat{P}(w_k \mid X_j, \hat{\theta}) \tag{3.39}$$

$$\hat{\mu}_k = \frac{\sum_{j=1}^{u} \hat{P}(w_k \mid X_j, \hat{\theta}) X_j}{\sum_{j=1}^{u} \hat{P}(w_k \mid X_j, \hat{\theta})} \tag{3.40}$$

$$\hat{\Sigma}_k = \frac{\sum_{j=1}^{u} \hat{P}(w_k \mid X_j, \hat{\theta})(X_j - \mu_k)(X_j - \mu_k)^T}{\sum_{j=1}^{u} \hat{P}(w_k \mid X_j, \hat{\theta})} \tag{3.41}$$

where,

$$\hat{P}(w_k \mid X_j, \hat{\theta}) = \frac{p(X_j \mid w_k, \hat{\theta}_k)\hat{P}(w_k)}{\sum_{i=1}^{2} p(X_j \mid w_i, \hat{\theta}_i)\hat{P}(w_i)} \tag{3.42}$$

We can rewrite Eq. (3.42) by substituting the expression for multivariate normal distribution, which simplifies Eq. (3.42) to Eq. (3.43)

$$\hat{P}(w_k \mid X_j, \hat{\theta}) = \frac{\left|\hat{\Sigma}_k\right|^{-1/2} \exp\left[-\frac{1}{2}(X_j - \hat{\mu}_k)^T \hat{\Sigma}_k^{-1}(X_j - \hat{\mu}_k)\right]\hat{P}(w_k)}{\sum_{i=1}^{2}\left|\hat{\Sigma}_i\right|^{-1/2} \exp\left[-\frac{1}{2}(X_j - \hat{\mu}_i)^T \hat{\Sigma}_i^{-1}(X_j - \hat{\mu}_i)\right]\hat{P}(w_i)} \tag{3.43}$$

It is simple to interpret Eqs. (3.39-3.43). If we consider the extreme case when $\hat{P}(w_k \mid X_j, \hat{\theta})$ is 0.0 when X_j is from Class w_1 and 0.1 otherwise, $\hat{P}(w_1)$ is the fraction of samples from w_1, $\hat{\mu}_1$ is the mean of those samples and $\hat{\Sigma}_1$ is the covariance matrix of those samples. When $\hat{P}(w_k \mid X_j, \hat{\theta})$ is between 0.0 and 1.0 all the samples have a contribution on these estimates for both the clusters.

The problems involved in solving these implicit equations in Case 2 is the same as the problems involved in Case 1 with the added complication of having to deal with singular solutions. Hence in order to solve these equations the Expectation-Maximization Algorithm is used which will be explained in the next section.

3.3.4 Clustering using Expectation Maximization (EM) Algorithm

The Expectation-Maximization algorithm [111] is a powerful iterative procedure for finding MLE of parameters in statistical models where the model performance depends on unobserved latent variables. These latent variables are sometimes called Hidden labels

and denoted by *H*, and are nothing but $\hat{P}(w_k \mid X_j, \hat{\theta})$ values, which are calculated in Eqs. (3.42-3.43). As described in the previous section these latent variables act as the weights which are used to calculate new values of means and covariances for different clusters. As should be expected, the sum of latent variables for both the clusters should equal 1.

$$\hat{P}(w_1 \mid X_j, \hat{\theta}) + \hat{P}(w_2 \mid X_j, \hat{\theta}) = 1 \tag{3.44}$$

Table 3. 1Expectation Maximization Algorithm for Clustering of Unlabeled data

Step1	*Input: Unlabeled data* $X = \{x_1, ..., x_u\}$
Step2	*Initialize t = 0 and input initial model parameter* $\theta^0 = [\mu_k, \Sigma_k, P(w_k)]$.
Step3	*Repeat until the convergence of* $p(X \mid \theta^t)$: ***i.*** ***E-Step:*** *compute* $\hat{P}(t+1)(w_k \mid X_j, \hat{\theta}(t)) = \frac{\left\|\hat{\Sigma}_k(t)\right\|^{-1/2} \exp\left[-\frac{1}{2}(X_j - \hat{\mu}_k(t))^T \hat{\Sigma}_k^{-1}(t)(X_j - \hat{\mu}_k(t))\right]\hat{P}(t)(w_k)}{\sum_{i=1}^{2}\left\|\hat{\Sigma}_i(t)\right\|^{-1/2} \exp\left[-\frac{1}{2}(X_j - \hat{\mu}_i(t))^T \hat{\Sigma}_i^{-1}(t)(X_j - \hat{\mu}_i(t))\right]\hat{P}(t)(w_i)}$ (3.45) ***ii.*** ***M-Step:*** *find* θ^{t+1} *that maximizes* $p(X \mid \theta^t)$ $\hat{\mu}_k(t+1) = \frac{\sum_{j=1}^{u}\hat{P}(t+1)(w_k \mid X_j, \hat{\theta}(t))X_j}{\sum_{j=1}^{u}\hat{P}(t+1)(w_k \mid X_j, \hat{\theta}(t))}$ (3.46) $\hat{\Sigma}_k(t+1) = \frac{\sum_{j=1}^{u}\hat{P}(t+1)(w_k \mid X_j, \hat{\theta})(X_j - \mu_k(t+1))(X_j - \mu_k(t+1))^T}{\sum_{j=1}^{u}\hat{P}(t+1)(w_k \mid X_j, \hat{\theta})}$ (3.47) $\hat{P}(t+1)(w_k) = \frac{1}{u}\sum_{j=1}^{u}\hat{P}(t+1)(w_k \mid X_j, \hat{\theta})$ (3.48) ***iii.*** *t = t+1*

Step 4	*Output* θ^t

The EM algorithm alternates between the Expectation step (E-Step) and the Maximization step (M-step). In the E-Step, the expectation of the log-likelihood is evaluated using the current estimate for the latent variables, and in the M-Step, the parameters of the model that maximize the log-likelihood are maximized. Hence, the EM algorithm is an iterative algorithm which locally maximizes the likelihood function, $p(X \mid \theta)$. The steps in EM algorithm along with the modified implicit equations, which were derived in the last section, are present in Table 3. 1. Note that as mentioned before as will all hill-climbing algorithms the Expectation-Maximization Algorithm doesn't guarantee convergence to a global maxima. The results always depend on the initial starting point and the problem of multiple solutions always exists. Furthermore, the repeated computation and inversion of the covariance matrix can be quite time consuming in many cases.

Most of the problems encountered can be circumvented if it is possible to assume that the covariance matrix is diagonal which reduces the number of unknown variables drastically. If this assumption seems to simplify the underlying problem by a large amount, the covariance matrices of all the clusters can be assumed to be the same, in which case most of the problems with EM can still be circumvented with relatively less computational requirement.

Even with all these inherent problems the EM algorithm has become the de-facto method for finding the parameters of a model that maximizes the log likelihood since it

guarantees that the likelihood value of the converged solution will be greater than the initial starting solution. If the initial estimates are good, a rapid convergence can be seen. This is generally the case when a large number of unlabeled data is available. The EM algorithm also gives a way to maximize the log likelihood function without the need to use a brute-force function maximization technique such as Newton Raphson Method. In many cases a gradient-based function maximization technique such as Newton Raphson Method would not work because the log-likelihood function is non-convex [112].

In the case of reliability estimation, the cost of obtaining unlabeled data is minimal. As mentioned in the earlier chapters the major part of the cost comes from the need to obtain labels using FEA. Hence using EM for clustering with only unlabeled data is not a considerable computational cost increment for problems involving reliability estimation.

3.4 Summary

In this chapter we introduced the concepts of supervised and unsupervised learning and how labeled and unlabeled data can be used to train models in supervised learning and unsupervised learning respectively. The Probabilistic Neural Network (PNN) was introduced as a superior supervised learning method and we discussed the limitations of PNN. Particularly it was mentioned that PNN assumes a 'spherical' Gaussian centered at each labeled data, which hinders its ability to provide highly accurate results with less number of labeled data. Since reliability estimation problems incur high computational cost, there is a scarcity of labeled data, but a large number of unlabeled data can be accessed. Unsupervised learning was introduced as a class of methods, which can be used

to learn surrogate models when only unlabeled data is present. Specifically, the Mixture of Gaussians (MOG) model was studied, whose parameter's maximum likelihood estimates can be estimated using an alternate hill-climbing method called as the Expectation-Maximization Algorithm. In Chapter 4 we will build on this foundation to explain the Semi-Supervised Learning (SSL) algorithms that were developed as a part of this research in order to accurately estimate the reliability of a system when few labeled data are available.

CHAPTER 4

SEMI-SUPERVISED LEARNING FOR RELIABILITY ASSESSMENT

The procedure for implementing semi-supervised learning for reliability assessment is explained in this chapter. This chapter extends the discussion on machine learning from the last chapter and describes the procedure of application of some of those techniques for the special domain of reliability based design. Specifically we focus on Semi-Supervised Learning (SSL), which can be regarded broadly as a combination of Supervised and Unsupervised Learning. PNN will be chosen as the supervised learning method for SSL and the Expectation-Maximization (EM) algorithm will be chosen for unsupervised learning.

In Section 4.1 we explain the basics of Semi-Supervised Learning (SSL) and in Section 4.2 we recap the important concepts and the limitations of PNN. We provide a brief description of Mixture of Gaussians and EM algorithm as it applies to SSL in Section 4.3 and in Section 4.4 we explain the integration of PNN with the EM algorithm for a preliminary SSL algorithm which enables automatic selection of the smoothing parameter for PNN. We call this algorithm SSL-1. We explain a more generalized version of an SSL algorithm which allows full freedom to covariance matrices during the training of PNN in Section 4.5. We summarize the chapter in Section 4.6.

4.1 Semi-Supervised Learning (SSL)

Supervised learning is the machine learning task of inferring a function from supervised training data which consists of training examples. In supervised learning, each example is

a pair consisting of an input (typically a vector) and a desired output value. A supervised learning algorithm analyzes the training data and produces a function which is called a classifier (if the output is discrete) or a regression function (if the output is continuous). The inferred function should predict the correct output value for any valid input. This kind of training data is called labeled data and the desired values are called labels. In *unsupervised learning*, the primary goal is to estimate the parameters of a mixture model when the correspondence of individual data points to the mixture components is unknown. In order to estimate the parameters of the mixture model, the training dataset does not need to include the labels, as in case of supervised learning. Hence, the training data for unsupervised learning is called unlabeled data.

SSL is a type of machine learning task that utilizes both labeled and unlabeled data. SSL can be considered to be halfway between supervised and unsupervised learning. Traditional classifiers and other supervised learning algorithms use only labeled data for training. However, obtaining training data often requires tremendous effort since the process of labeling individual instances is either time consuming or computationally intensive. On the contrary, unlabeled data can be readily generated without requiring additional experimentation or simulation. SSL achieves this additional improvement in accuracy by augmenting a small number of labeled data with a large amount of unlabeled data in order to train a better classifier. In SSL, the dataset X can be divided into two parts, X_l and X_u. For the first part $X_l := (x_1,...,x_l)$, the labels $W_l := (w_1,...,w_l)$ are provided, whereas for the second part $X_u := (x_{l+1},...,x_{l+u})$, the labels are unknown. For the sake of generality, the dataset [X_l, W_l] will be called *labeled* dataset and the dataset [X_u] will be called *unlabeled* dataset for the rest of the dissertation. Typically, the *smoothness*

assumption [113] should be valid in order to perform SSL. The smoothness assumption for supervised learning states that if two points, x_1 and x_2, are close, then the corresponding outputs, w_1 and w_2, are also close to each other. Hence, if two instances, x_1 and x_2 belong to the same cluster, the corresponding labels, w_1 and w_2, are likely to be close too. On the contrary, if both instances are linked by a low density region, then their labels are more likely to be different.

In general, the addition of unlabeled data can improve the classifier accuracy in the case of generative classifiers. Recall that generative classifiers approximate the PDF distributions of all classes during the training process and a test point is predicted to belong to one class based on it's higher probability value when compared to probability values corresponding to other classes. These probabilities are calculated by evaluating the PDFs for the test data point. Unlabeled data contain the PDF information of all data collectively. Hence a large number of unlabeled data should help in the estimation of PDFs of all classes during the training process of generative classifiers. By learning how the data from each class is distributed, we may decompose the mixture into individual classes. One way of modeling the data as a mixture of PDFs is by assuming that the data comes from a Mixture of Gaussian (MOG) PDFs. The EM algorithm is used to compute the parameters of the constituent Gaussian PDFs in this dissertation. The following section illustrates the details of MOG and the EM algorithm.

4.2 Limitations of Probabilistic Neural Network

Consider a pattern x_i that belongs to either class *A* or class *B*. If a decision of whether x_i belongs to class *A* or class *B* has to be made based on the data represented in the *m*-dimensional vector $X^T=[X_1\ X_2 \ldots\ X_j \ldots\ X_m]$, the Bayes decision rule is represented by

$$x_i \in A \text{ if } n_A f_A(x_i) > n_B f_B(x_i) \tag{4.1}$$

$$x_i \in B \text{ if } n_A f_A(x_i) < n_B f_B(x_i) \tag{4.2}$$

where $f_A(X)$ and $f_B(X)$ are class conditional PDFs for categories A and B, respectively. n_A and n_B represent the number of patterns in class A and class B of the training data. Since there are only two classes, it follows that the total number of labeled patterns in the training data $n = n_A + n_B$. A similar Bayes Decision rule analogous to Eqs. (4.1-4.2) can be established for many category problems as well.

In Figure 4. 1 we show the architecture of a PNN for classifying the vector X into two classes—A and B. It consists of four different layers, input layer, pattern layer, summation layer and the output layer. If the training dataset X_T consists of n data points and 2 classes, each containing m dimensions, the PNN network will have m neurons in the input layer, n neurons in the pattern layer and 2 neurons in the summation layer.

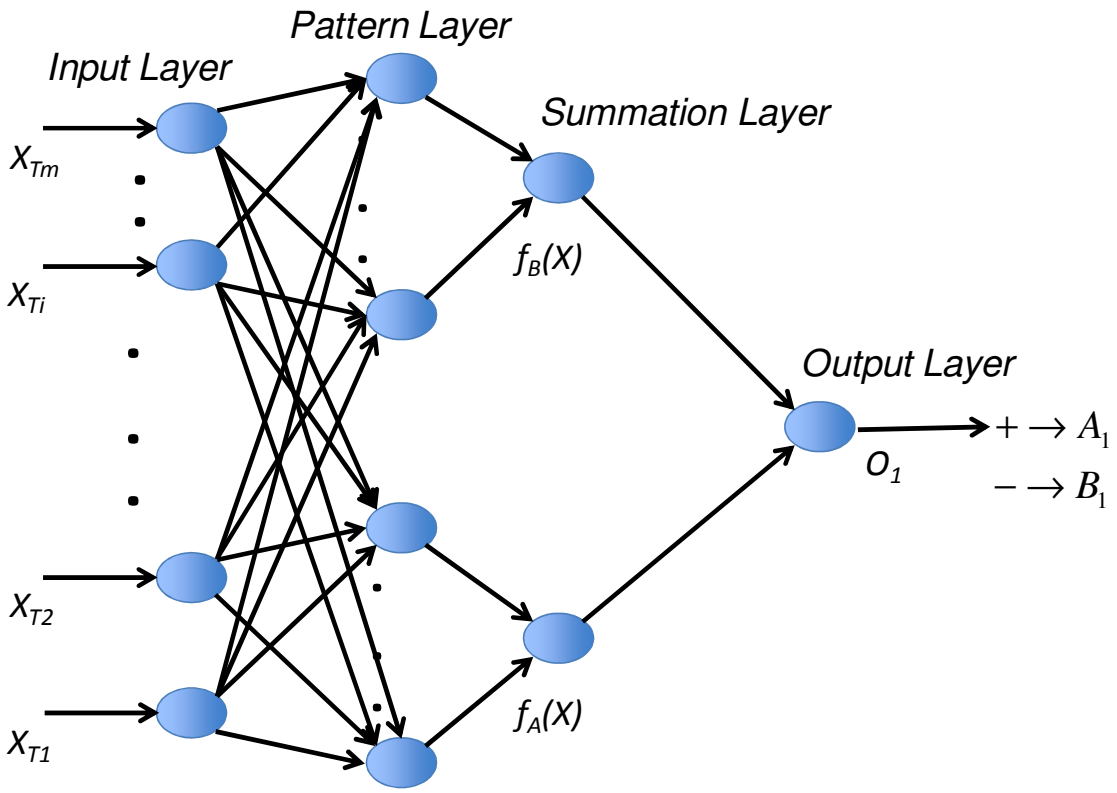


Figure 4. 1 Architecture of Probabilistic Neural Networks

From Eqs. (4.1-4.2) it is clear that the effectiveness of this procedure depends on the accuracy of the PDF estimation. A procedure for construction of a family of estimates of the PDFs, *f(X),* was shown by Parzen [105] and Cacoullos [114] . PNN utilizes the Parzen window method for computing the PDF function while considering a multivariate kernel. When a Gaussian kernel is used, the multivariate estimate for PDF of class *A* can be expressed as

$$f_A(X) = \frac{1}{(2\pi)^{m/2}|\Sigma|^m} \frac{1}{n_A} \sum_{i=1}^{n_A} \exp\left[-\frac{(X - X_{TAi})^T (X - X_{TAi})}{2\sigma^2} \right] \tag{4.3}$$

where X is the vector to be classified, $f_A(X)$ is the value of the PDF of category A at point X, n_A denotes the number of patterns in category A, m is the dimensionality of the training

patterns, X_{TAi} is the i^{th} training pattern from category A, and Σ is the covariance matrix or smoothing parameter. The task assigned to the classifier is to classify the dataset X after it is trained using the training dataset X_T. X_T is comprised of X_{TA} and X_{TB} corresponding to whether the data point belongs to class A or class B. This information is available apriori since X_T is the training dataset. $f_A(X)$ can be determined by summing the multivariate Gaussian distributions centered at each training sample corresponding to class A.

To summarize, a Probabilistic Neural Network is the neural network representation of a Parzen window classifier where the kernels density functions for each labeled pattern corresponding to a class are summed together to form the PDF of that class. Each of these kernels is assumed to be Gaussians with 'spherical' covariance matrices. The reader should remember from the discussion in Chapter 3 that 'spherical' covariance for a two dimensional Gaussian is of the form:

$$\Sigma = \sigma^2 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.4}$$

and a 'full' covariance for the two dimensional Gaussian case is of the form:

$$\Sigma = \begin{bmatrix} \sigma_{11}^2 & \sigma_{12}^2 \\ \sigma_{12}^2 & \sigma_{22}^2 \end{bmatrix} \tag{4.5}$$

For a generalized 'spherical' covariance case the covariance matrix is of the form $\Sigma = \sigma^2 I$, where I represents an identity matrix of size m.

Evaluating the PDFs for all classes and subsequently using the Bayes decision rule enables the classification of a test pattern. In both PNN and Parzen window classifier with Gaussian Kernel functions, the smoothness parameter (also called window, width or

covariance) can be thought of as a value that quantifies the influence that a particular labeled pattern has on a test data point. This smoothing parameter is the value of σ^2 in Eq. (4.4). If the window has a high value then the influence is lower (exponential term in Eq.(4.3)), and if the window has a lesser value PNN behaves similar to a k-nearest neighbor (k-NN) classifier [98].

Assuming a spherical covariance matrix is a simple way to estimate a PDF function when the real PDF function is unknown. Hence in cases, when it is prior knowledge that the classification boundary is not very complex, the spherical assumption should give good results too. However, in case of 'spherical' covariance assumption also an automatic method for estimation of smoothing parameter, σ^2, is required. We achieve this feat by our proposed SSL-1 algorithm.

In cases when there is no prior knowledge about the simplicity of the classification boundary, no such assumption should be made. We hypothesize that in the presence of unlabeled data, more information about the real PDF can be gathered from the unlabeled data itself which will lead to better estimates of covariance matrices for each training pattern. Hence, in the presence of unlabeled data and no prior knowledge about simplicity of the classification boundary, we can release the constraint of 'spherical' covariance matrix and assume a 'full' covariance matrix so that each dimension in our dataset is allowed to have different variance values and cross-dependencies between different dimensions is also allowed. We achieve this feat by our proposed SSL-2 algorithm.

We start out by discussing the basics of Semi-Supervised Learning and Mixture of Gaussians in the next section which is foundational for functioning of SSL-1. Following this we explain the proposed SSL-1 and SSL-2 algorithms.

4.3 Mixture of Gaussians and Expectation-Maximization for SSL-1

Consider the scenario when the objective is to classify a data point x_i into one of the two classes, w_1 and w_2. In order to achieve this objective probabilistically, we have to assign a label to x_i which maximizes the *posterior* probability, $p(w|x)$. Given a data point, this conditional probility value specifies the probability of this data point belonging to either class. In order to minimize classification error, the best strategy is to always classify x_i into the class for which this posterior probability value is greater. In order to conduct classification using a *generative* [95] model, $p(w|x)$ can be estimated using the Bayes rule as shown in Eq. (4.6) where the summation in the denominator is over all class labels.

$$p(w|x) = \frac{p(x|w)\,p(w)}{\sum_{\omega} p(x|w)\,p(w)} \tag{4.6}$$

A multivariate Gaussian distribution can be used for for calculating $p(x|w)$,

$$p(x \mid w) = N(x : \mu_w, \Sigma_w) = \frac{1}{(2\pi)^{m/2}\left|\Sigma_w\right|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_w)^T \Sigma_w^{-1} (x-\mu_w) \right) \tag{4.7}$$

where μ_w and Σ_w represent the mean vector and covariance matrix, respectively.

The class conditional PDF, $p(w|x)$ is dependent on the Gaussian parameters μ_w and Σ_w in Eq. (4.7). It can be concluded that training a good classifer amounts to estimating good

values for μ_w and Σ_w. In order to determine these parameters, Maximum Likelihood Estimate (MLE) can be used. Given training data X, the MLE is

$$\hat{\theta} = \begin{bmatrix} \hat{\mu}_w \\ \hat{\Sigma}_w \end{bmatrix} = \arg\max_{\hat{\theta}} p(X \mid \theta) = \arg\max_{\hat{\theta}} \log p(X \mid \theta) \tag{4.8}$$

When $X = \{(x_i, w_i)\}_{i=1}^{l}$, the log likelihood in Eq.(4.8) can be easily rewritten as

$$\log p(X \mid \theta) = \log \prod_{i=1}^{l} p(x_i, w_i \mid \theta) \tag{4.9}$$

and

$$\log \prod_{i=1}^{l} p(x_i, w_i \mid \theta) = \sum_{i=1}^{l} \log p(w_i \mid \theta)\, p(x_i \mid w_i, \theta) \tag{4.10}$$

Since $p(x, w) = p(w)\, p(x \mid w)$, $p(x, w)$ represents the joint distribution of instances and labels. Eq. (4.9) represents the fact that the probability of a set of i.i.d events is the product of individual probabilities. Hence, finding the MLE is the process of solving Eq. (4.8) as an optimization problem. In the case of a 2-class problem such as the problem of probability of reliability estimation, the data can be assumed to be sampled from two different Gaussian PDFs. However, the correspondence of data points to particular PDF is unknown. In order to define the two Gaussians, it is essential to estimate the means and variances of these individual distributions. In addition to the means and variances, a mixing parameter would be required to properly define the density of the whole dataset so that the mixing parameter λ defines the density of the data according to

$$f(x) = \lambda N(x; \mu_1, \Sigma_1) + (1 - \lambda) N(x; \mu_2, \Sigma_2), \qquad \lambda \in [0,1] \tag{4.11}$$

Once the parameters (λ, μ_1, Σ_1 and μ_2, Σ_2) are determined, the density estimation process is complete. For the case when the data is composed of only labeled data points, it is possible to obtain analytical expressions for the MLE. A detailed derivation of estimating the MLE for the two class Gaussian mixture for the case of only labeled data is available in Ref. [41].

If the training data X consists of both labeled and unlabeled data such that $X = \{(x_1, w_1),...,(x_l, w_l), x_{l+1},...,x_{l+u}\}$, the log likelihood function can then be defined as

$$\log p(X \mid \theta) = \log\left(\prod_{i=1}^{l} p(X_i, w_i \mid \theta)\prod_{i=l+1}^{l+u} p(X_i \mid \theta)\right) \tag{4.12}$$

$$= \sum_{i=1}^{l} \log P(w_i \mid \theta)p(x_i \mid w_i, \theta) + \sum_{i=l+1}^{l+u} \log p(x_i \mid \theta) \tag{4.13}$$

The apparent difference between Eq. (4.10), which uses only labeled data (supervised learning), and Eq. (4.13), which used both labeled and unlabeled data, is the second term in Eq. (4.13). In the second term of Eq. (4.13), $p(X \mid \theta)$ is called the *marginal probability* [85], which is given by

$$p(X \mid \theta) = \sum_{w=1}^{2} p(X, w \mid \theta) = \sum_{w=1}^{2} P(w \mid \theta)\, p(X \mid w, \theta) \tag{4.14}$$

for the 2-class problem. Hence Eq. (18) can be rewritten as

$$\log p(X \mid \theta) = \sum_{i=1}^{l} \log P(w_i \mid \theta)p(X_i \mid w_i, \theta) + \sum_{i=l+1}^{l+u} \log \sum_{w=1}^{2} P(w_i \mid \theta)\, p(X_i \mid w_i, \theta) \tag{4.15}$$

The marginal probabilty is the probability of generating x from any of the classes. The marginal probablities in Eq. (4.13) account for the fact that the presence of unlabeled data is known *apriori* but it is impossible to know which class each instance belongs to. Hence, for mixture models, SSL is different from supervised learning by the expression for the log likelihood function. According to Eq. (4.13), the solution of the optimization problem would need to fit both the labeled and unlabeled data. It is possible to model the unknown labels $,(w_{l+1},...,w_{l+u})$, for the unlabeled data, as hidden variables. These hidden variables can be represented as $H=\{w_{l+1},...,w_{l+u}\}$ and will be denoted as hidden data from here on. The presence of hidden variables makes the log-likelihood in Eq. (4.13) non-convex and hard to optimize since it is impossible to obtain an analytical solution for the MLE, $\hat{\theta}$. The MLE can be computed using standard root finding techniques such as the Newton Raphson Method on Eq. (4.9), or using a specialized algorithm such as the Expectation-Maximization Algorithm. However, the EM algorithm is the preferred method in most cases since it is guaranteed to achieve a maximum in a finite number of iterations [85]. However, it is not guaranteed to be the global maximum.

The EM algorithm is a powerful iterative procedure for finding the MLE of parameters in statistical models where the model performance depends on unobserved latent variables. The EM algorithm alternates between the Expectation step (E-Step) and the Maximization step (M-step). In the E-Step, the expectation of the log-likelihood is evaluated using the current estimate for the latent variables, and in the M-Step, the parameters of the model that maximize the expected log-likelihood (found in the E-Step), are maximized. Hence, the EM algorithm is an iterative algorithm which locally maximizes $p(X \mid \theta)$. Table 4. 1 presents the basic EM algorithm which is used to

optimize the log likelihood for the case when both labeled and unlabeled data are present (SSL).

Table 4. 1 EM algorithmic framework for SSL-1 [116]

Step1:	*Input: Labeled and unlabeled data* $X = \{(x_1, w_1),...,(x_l, w_l), x_{l+1},...,x_{l+u}\}$
Step2:	*Initialize t = 0 and input initial model parameter* θ^0.
Step3:	*Repeat until the convergence of* $p(X \mid \theta^t)$: *i.* *E-Step: compute* $q^t(H) \equiv p(H \mid X, \theta^t)$ *ii.* *M-Step: find* θ^{t+1} *that maximizes* $\sum_H q^t(H) \log p(X, H \mid \theta^{t+1})$ *iii.* $t = t+1$
Step4:	*Output* θ^t

In Table 4. 1, $q^t(H)$ is the hidden label distribution which can be visualized as assigning '*soft labels*' to the unlabeled data according to the current model parameters, θ^t. Soft labels are probability values for the data points belonging to each class. Hence, they can also be considered as temporary labels on the data point. Typically, generative classifers assign a soft label to a data point before classifying it to the class for which the soft label is maximum. Therefore, the EM algorithm can be modified to suit any generative classifier and incorporate both labeled and unlabeled data in the framework. The following section illustrates the development of a modified EM algorithm into a PNN classifier to improve its accuracy.

4.4 SSL-1 for Automatic Selection of Smoothing Parameter of PNN

In the proposed method, a SSL algorithm for PNN is developed to facilitate the reliability assessment process of the complex systems for the first time. In order to use unlabeled data for improving a classifier in the context of the SSL, a general form of clustering can be considered. By treating the class labels of the unlabeled data as missing values, the EM algorithm can be used to train a better classifier. The EM algorithm consists of two steps, namely Expectation (E-step) and Maximization (M-step). The EM algorithm is an iterative algorithm where the E-step computes the expected value of '*unavailable*' data using the current estimation procedure. The M-step then updates the current estimation procedure based on all the data including both the labeled and unlabeled data. This procedure is continued until the satisfaction of the convergence. The following sections describe the details of the proposed method.

4.4.1 EM-like Algorithmic Framework for Training a SSL-1 based PNN

When the aim is to train a PNN classifier using both labeled and unlabeled data by adapting the EM algorithm, the first step is to train a PNN classifier using only the labeled data. This step 1 requires the estimation of PNN parameters, namely, θ. The PNN parameter, θ, constitutes of the mixing parameter λ, mean μ, and covariance K for the individual clusters. This trained PNN with the labeled data is then used to calculate the labels of the unlabeled data in the E-step. The expected labels are treated as real labels for the unlabeled data in the M-step. In the M-step all the data is used to calculate the parameters, θ, for the new PNN classifier. This E-step and M-step are repeated until convergence. A proposed algorithm for training the PNN based on the EM process is summarized in Table 4. 2.

An available dataset, X, will consist of labeled dataset, X_l, and unlabeled dataset, X_u, such that $X = X_l \cup X_u$. Specifically, the dataset, X, will have the form $X = \{(x_1, w_1),...,(x_l, w_l), x_{l+1},...,x_{l+u}\}$, where w_i's represent the class labels. The task of the classification process is to train a PNN classifer using the dataset, X, for predicting the class labels of unseen unlabeled data points.

Table 4. 2 EM-like algorithmic framework for SSL-1 using PNN

Step1:	*Input: Labeled and unlabeled data* $X = \{(x_1, w_1),...,(x_l, w_l), x_{l+1},..., x_{l+u}\}$
Step2:	*Initialize t = 0 and* $\theta^0 = \{\lambda_j^0, \mu_j^0, \Sigma_j^0\}_{j\in\{1,2\}}$ *to the MLE. Here j = 1, 2 corresponds to class A and B respectively.*
Step3:	*Train initial PNN classifier C usinglabeled data.*
Step4:	*Begin Loop.*
Step5:	*Iterate until CCM converges while classifying the unlabeled data* X_u *with the current classifier, C*
Step6:	*Step:Use current classifier C, to evaluate the classification scores and predict class labels,* w_i, *for each unlabeled data point. These class labels constitute the Hidden data* $H = \{w_{l+1},...,w_{l+u}\}$. *Hence, for all unlabeled instances* $i \in \{l+1,...,l+u\}$, *class* $j \in \{1,2\}$, *compute* $CCM_i = [f_A(x_i)\ \ f_B(x_i)]$ *and* $\gamma_{ij} = p(w_j \mid x_i, \theta^t) = \dfrac{\lambda_j^t CCM_{ij}}{\Sigma_{j=1}^2 \lambda_j^t CCM_{ij}}$ (4.16) *For labeled instances, define* γ_{ij} *=1 if* w_i*=j, and 0 otherwise*
Step7:	*M-Step:Compute MLE,* θ^{t+1} *using the current* γ_{ij}. *Perform following for* $j \in \{1,2\}$,

	$$s_j = \sum_{i=1}^{l+u} \gamma_{ij} \quad (4.17)$$ $$\mu_j^{t+1} = \frac{1}{s_j} \sum_{i=1}^{l+u} \gamma_{ij} x_i \quad (4.18)$$ $$\Sigma_j^{t+1} = \frac{1}{s_j} \sum_{i=1}^{l+u} \gamma_{ij} \left(x_i - \mu_j^{t+1}\right)\left(x_i - \mu_j^{t+1}\right)^T \quad (4.19)$$ $$\lambda_j^{t+1} = \frac{s_j}{l+u} \quad (4.20)$$
Step8:	*Rebuild the PNN classifier C using current values of parameters θ , based on both labeled and unlabeled data X and H, while using the class lebels, $w_{i,}$ calculated for unlabeled data in the E-Step*
Step9:	*t=t+1*
Step10:	*End Loop.*
Step11:	*Output:* $\{\lambda_j, \mu_j, \Sigma_j\}$ *and current classifier C*

As described in Table 4. 2, the process begins with training an initial classifier, *C*, from only the labeled dataset, X_l. Next, the current classifier is used for estimating the labels of the unlabeled dataset, X_u. The temporarily assigned class labels for the unlabeled data are considered to be hidden and can constitute the hidden data $H = \{w_{l+1}, ..., w_{l+u}\}$. During this estimation process for the hidden data, the scores ($f_A(x_i)$ and $f_B(x_i)$ for x_i such that $x_i \in X_u$) of each data point, x_i, corresponding to class *A* and class *B*, will be denoted as the *class conditional membership* (CCM) values of the data point. The scores are calculated in the summation layer of the PNN as shown in Eq. (4.3).

Then these scores are stored in a CCM matrix; *CCM(u,2)* for the case of only two classes, *A* and *B*. The CCM matrix is normalized according to Eq. (4.16) in order to compute the weights, γ_{ij}. The class labels of x_i such that $i \in [l+1, l+u]$ are also estimated using the current classifier, *C*. This step is the E-Step in this proposed method. The weights and the class labels are used to calculate the parameters of the new classifier. In the case of PNN, the diagonal covariance matrix obtained can be used to update the values of the smoothing parameter of the PNN. For the two classes case, the covariance matrix has a size of 2. The first diagonal element is used as the smoothing parameter for the patterns that are summed at the first neuron of the summation layer of PNN. Similarly, the second element of the covariance matrix is used as the smoothing parameter of the patterns that are summed at the second neuron of the summation layer. This process trains the SSL based PNN. All the data corresponding to both the labeled and unlabeled data is used to train a new classifier, *C*. This step constitutes the M-Step of the procedure. Both the E-Step and M-Step are iterated until the convergence of the CCM Matrix is achieved. The convergence of the *CCM* also ensures the convergence of the *maximum likelihood estimator*. Once convergence of the *CCM* is achieved, the current classifier, *C,* can be used to predict the labels of unseen, unlabeled data points.

4.4.2 Proposed Framework for Reliability Estimation using PNN and EM

Figure 4. 2 illustrates the proposed framework for the reliability assessment using a classification process based the developed learning algorithm for the PNN. First, the user defines the limit state function with the corresponding random variables along with the PDF information. For the training data, $l + u$ data points are to be sampled, where l is the number of labeled data points and u is the number of unlabeled data points. After

sampling l points from the given PDFs, the limit states/responses of these points can be evaluated using FEM, analytical formulas or other black-box techniques in order to label these points as either safe or unsafe. In the next step, u numbers of points are sampled; however, the limit states for these points will not be evaluated since they can be used later for the SSL process. An initial estimate of smoothing parameter, Σ, is used to train a PNN classifier to the labeled data. In the Expectation Step (E-Step) the current classifier computes the probabilistically weighted class labels for the unlabeled data.

The intermediate CCM matrix computed is stored for checking the convergence of the EM iterations. The probabilistically weighted class labels computed in the E-Step are then used to compute the MLE estimates of the PNN parameters. These parameters are used to train a new classifier with both the labeled and unlabeled data. This constitutes the Maximization Step (M-Step) of the framework. The E-Step and M-Step are iterated until the convergence of the CCM Matrix. Once the CCM Matrix converges, the updated PNN can be used to compute the probability of failure (P_f) by calculating the number of points in the failure class (Class 2) and dividing it by the number of points generated using MCS/LHS.

$$P_f = \frac{\text{No. of points in class 2}}{n} \tag{4.21}$$

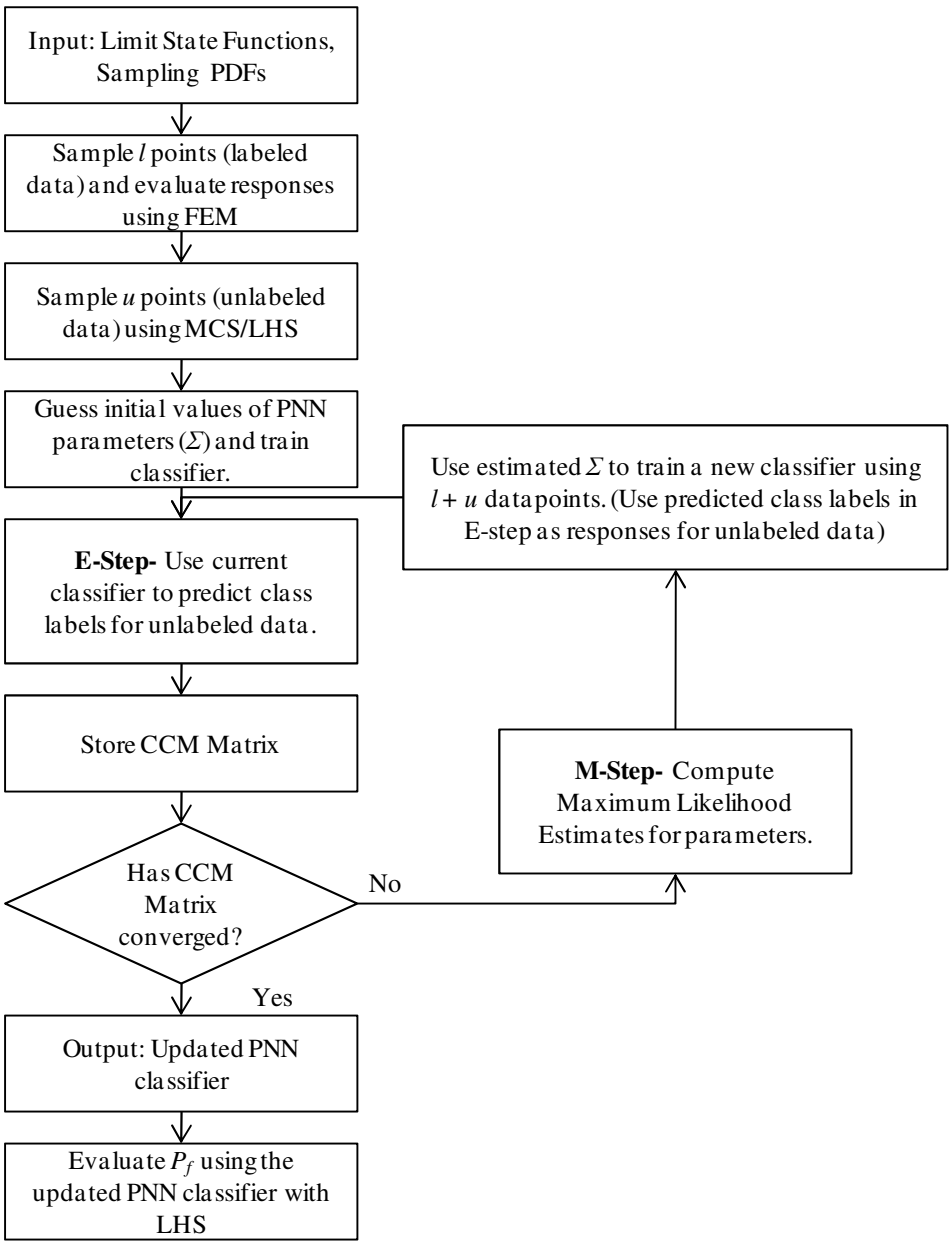


Figure 4. 2 Proposed framework for reliability estimation using PNN and EM

4.5 SSL-2 for Considering 'full' Covariance with PNN

The major computational cost of the reliability estimation process is the evaluation of responses using the Finite Element Analysis. Hence a surrogate model is used for estimating the responses from Finite Element Analysis. A Semi-Supervised Learning based method provides an avenue for drastic reduction of computational expense of the

procedure by reducing the number of labeled data points required for training a classifier. Furthermore, since the PDF functions for uncertain design variables are available, practically an unlimited number of unlabeled data points can be generated at relatively no computational cost. However, a limited number of labeled data points in many nonlinear and disjoint failure domain problems can put an upper limit on classification accuracy. With a combination of EM algorithm and Bayes theorem we propose the addition of a large number of unlabeled patterns which will enable relaxation of the spherical covariance assumption of PNN and increase the classification accuracy when the available number of labeled patterns is small. The steps in building such a classifier are explained in this section. Simultaneously, we also show a classification example where the Y-Axis is the true classification boundary so that the process is easier to visualize.

Step1:EM for clustering

The available dataset, X, will consist of labeled dataset, X_l, and unlabeled dataset, X_u, such that $X = X_l \cup X_u$. Specifically, the dataset, X, will have the form $X = \{(x_1, w_1),...,(x_l, w_l), x_{l+1},...,x_{l+u}\}$, where w_i's represent the class labels. In the case of reliability estimation, w can only take the values of either 0 or 1.

In Step 1 we ignore the labels of labeled data and combine the data with unlabeled data. Hence, we end up with $l+u$ unlabeled data, which we cluster around l mean values with the EM algorithm. The l labeled data points that we have initially can be taken as the starting mean values for the EM algorithm. The initial prior probabilities and covariances for all clusters can be taken as same. After the convergence of the EM algorithm we end up with l clusters according to which the whole dataset can be defined. In the following

the labels for clusters will be labeled by the symbol g_j where $1 \le j \le l$. The output from the EM algorithm are the prior probabililties and means and covariance values of the l clusters. The details of the EM algorithm is provided in Table 4. 3 [115].

Table 4. 3 EM algorithm for Gaussian mixture clustering

Step1:	*Input: data,* $X = \{x_1,...,x_l, x_{l+1},...,x_{l+u}\}, \mu_j$ and Σ_j for $1 \le j \le l$
Step2:	*Initialize t = 0*
Step3:	*Repeat E-Step and M-Step until the convergence of* $\log p(X \mid \theta^t)$: ***E-Step:****Compute the posterior probabilities for* $1 \le i \le l+u$ $\gamma_{ij} = p(g_j \mid x_i, \theta^t) = \frac{\lambda_j^t \lvert\Sigma_j\rvert^{-1/2} \exp\left\{-\frac{1}{2}(x_i-\mu_j)^{'}\Sigma_j^{-1}(x_i-\mu_j)\right\}}{\sum_{j=1}^{l} \lambda_j^t \lvert\Sigma_j\rvert^{-1/2} \exp\left\{-\frac{1}{2}(x_i-\mu_j)^{'}\Sigma_j^{-1}(x_i-\mu_j)\right\}}$ (4.22) ***M-Step:****Compute MLE,* θ^{t+1} *using the current* γ_{ij}. $s_j = \sum_{i=1}^{l+u} \gamma_{ij}$ (4.23) $\mu_j^{t+1} = \frac{1}{s_j} \sum_{i=1}^{l+u} \gamma_{ij} x_i$ (4.24) $\Sigma_j^{t+1} = \frac{1}{s_j} \sum_{i=1}^{l+u} \gamma_{ij} \left(x_i - \mu_j^{t+1}\right)\left(x_i - \mu_j^{t+1}\right)^T$ (4.25) $\lambda_j^{t+1} = \frac{s_j}{l+u}$ (4.26)

Step4:	*Output all* λ_j, μ_j *and* Σ_j.

Consider the simple example shown in Figure 4. 3 where the Y-Axis is the true classification boundary and four labeled points are available for training a classifier. After conclusion of Step-1 of our proposed SSL method we obtain four Gaussians which are shown in Figure 4. 4. Gaussian 1 is in one class and Gaussians 2 and 4 are in another class whereas Gaussian 3 is present in both classes. Note that since the EM algorithm is sensitive to the initial solution, the Gaussians might vary with each run. In the next steps, all the Gaussians are pooled together and their weighted contribution is used as the kernel at each labeled point.

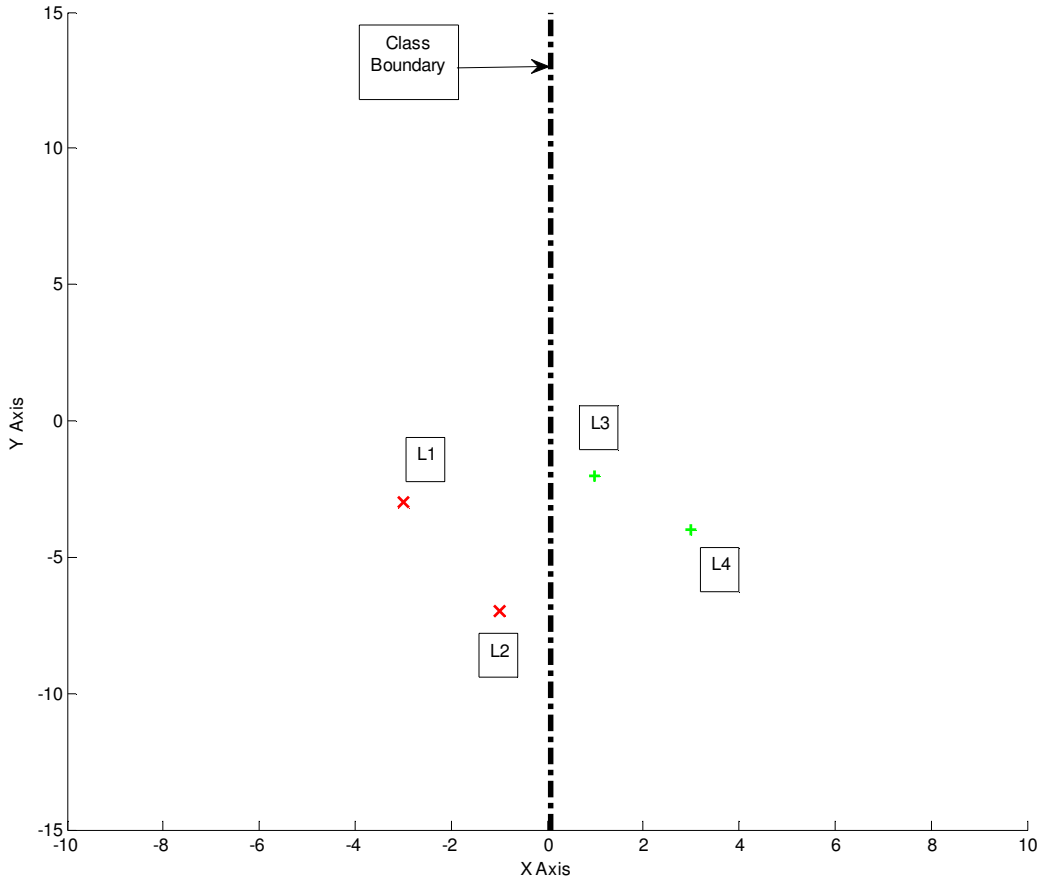


Figure 4. 3 Illustration of classification problem with Y axis as true classifier

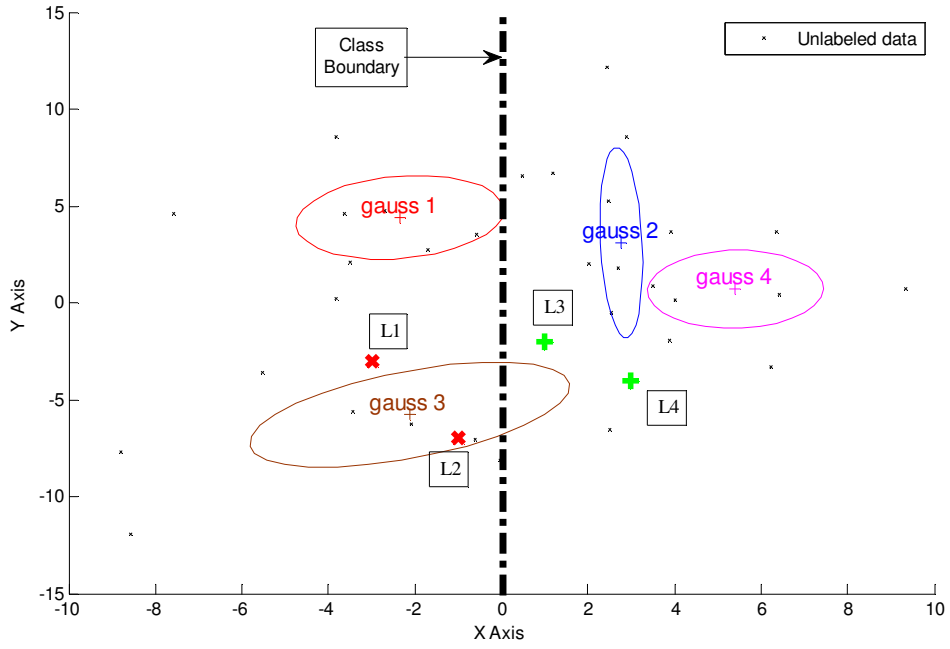


Figure 4. 4 Illustration of four clusters created using EM

Step2:Estimate Posterior Probabilities for labeled patterns

In step 1 we calculated the prior probabilities and the means and covariances of the l Gaussians from which our original dataset can be sampled. In this step we compute the posterior probabilities for all the labeled data points to belong to each of the l Gaussian clusters. All the posterior probabilities are stored in a l X l matrix. The following equation can be used to calculate the posterior probabilities using the Bayes Theorem:

$$\pi_{ij} = \frac{\lambda_j \left|\Sigma_j\right|^{-1/2} \exp\left\{-\frac{1}{2}(x_i - \mu_j)'\Sigma_j^{-1}(x_i - \mu_j)\right\}}{\sum_{j=1}^{l} \lambda_j \left|\Sigma_j\right|^{-1/2} \exp\left\{-\frac{1}{2}(x_i - \mu_j)'\Sigma_j^{-1}(x_i - \mu_j)\right\}} \text{ for } 1 \le i, j \le l \qquad (4.27)$$

Conceptually these posterior probabilities give a measure of closeness of the labeled data from the cluster centers. Hence the higher the posterior probability of a labeled point for a

given cluster, the higher the probability that the labeled data point was generated from that particular Gaussian cluster.

Step3:Estimate Probabilistic Distance of Test Pattern from Labeled Patterns

In this step we consider the test patterns, X_{Te} , which will have the same number of dimensions as the training data. We compute the kernel functions for each of the test data considering the covariance matrices of the Gaussian clusters and the labeled data points as the kernel centers. These distances are then added together after being weighted by the posterior probabilities computed from Eq. (4.27). This is shown in Eqs. (4.28-4.29). Adding the weighted distances corresponding the the labeled points belonging to class A and class B, we can estimate the class conditional PDF for each class. The reader is invited to notice the similarity of this process with the original PNN process.

$$f_A(X_{Te}) = \sum_{i=1}^{n_A} \sum_{j=1}^{l} \pi_{ij} \exp\left\{ -\frac{1}{2} (X_{Te} - X_{Trj})' \Sigma_j^{-1} (X_{Te} - X_{Trj}) \right\} \tag{4.28}$$

$$f_B(X_{Te}) = \sum_{i=1}^{n_B} \sum_{j=1}^{l} \pi_{ij} \exp\left\{ -\frac{1}{2} (X_{Te} - X_{Trj})' \Sigma_j^{-1} (X_{Te} - X_{Trj}) \right\} \tag{4.29}$$

Note that n_A and n_B are the number of labeled patterns that belong to Class A and Class B respectively and $n_A + n_B = l$. The term $\left(X_{Te} - X_{Trj}\right)' \sum_j^{-1} \left(X_{Te} - X_{Trj}\right)$ can also be recognized as the square of Mahalanobis [101] distance.

If we consider the example problem shown in Figure 4. 3 again, Step 3 partitions the Gaussian clusters identified by the EM algorithm which are shown in Figure 4. 4. After the partition, the resulting decision space is shown in Figure 4. 5. After the partition

of the Gaussians only the part that belongs to a certain class is used for calculation of the class conditional PDFs. This is eventually what Eqs. (4.28-4.29) accomplishes.

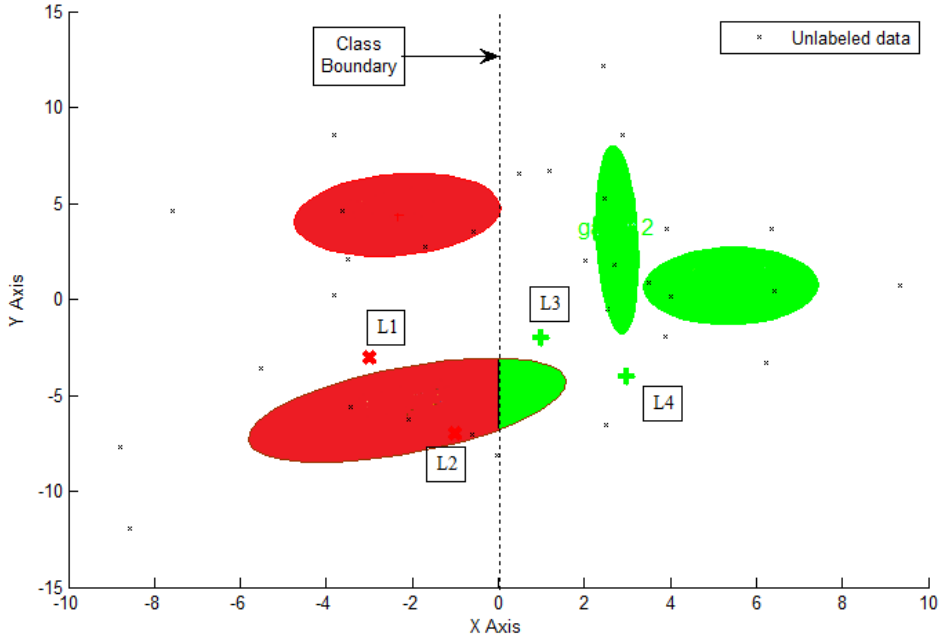


Figure 4. 5 Resulting decision space after Step 3

Step4:Bayes Decision Rule Implementation

After the class conditional PDF values are computed, the Bayes decision rule can be used to classify a test data in one of the two classes according to the following equations:

$$X_{Te} \in A \text{ if } n_A f_A(X_{Te}) > n_B f_B(X_{Te}) \tag{4.30}$$

$$X_{Te} \in B \text{ if } n_A f_A(X_{Te}) < n_B f_B(X_{Te}) \tag{4.31}$$

Notice that the method adopted in Step 3 and Step 4 are similar to the PNN process but instead of associating a particular 'spherical' covariance matrix with all labeled data, we pool all the different 'full' covariance matrices and weight them based on the posterior probabilities computed from Step 2. This process enables more accurate

computation of the covariance matrices and doesn't restrict the covariance at a particular labeled data point to a particular value; instead, the covariance at a labeled point is a weighted average of the pooled covariance matrices.

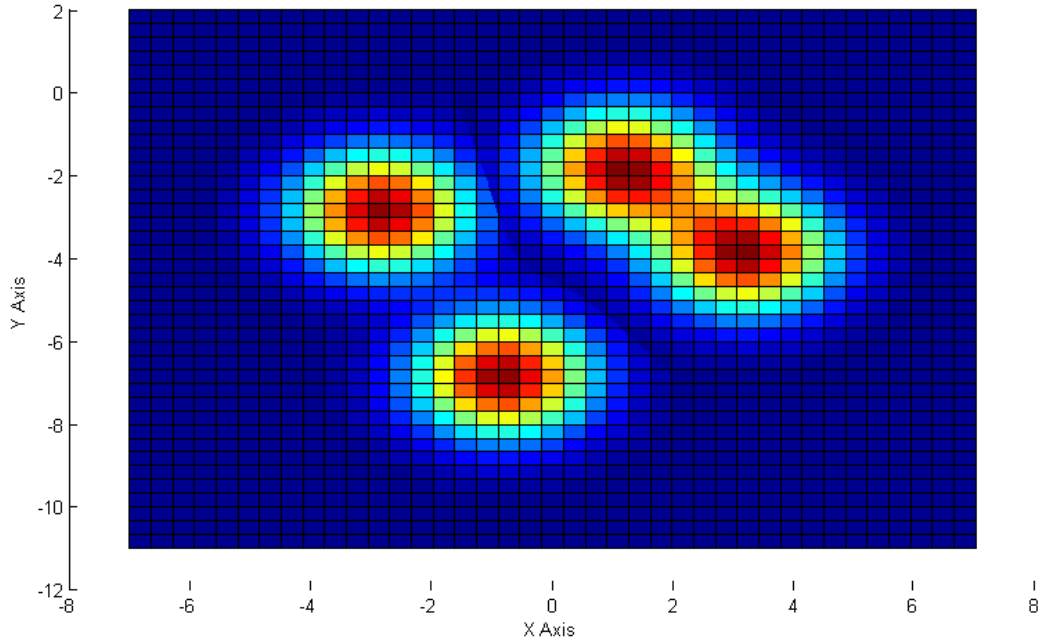


Figure 4. 6 Spherical Gaussian Kernels on labeled points from Figure 4. 3

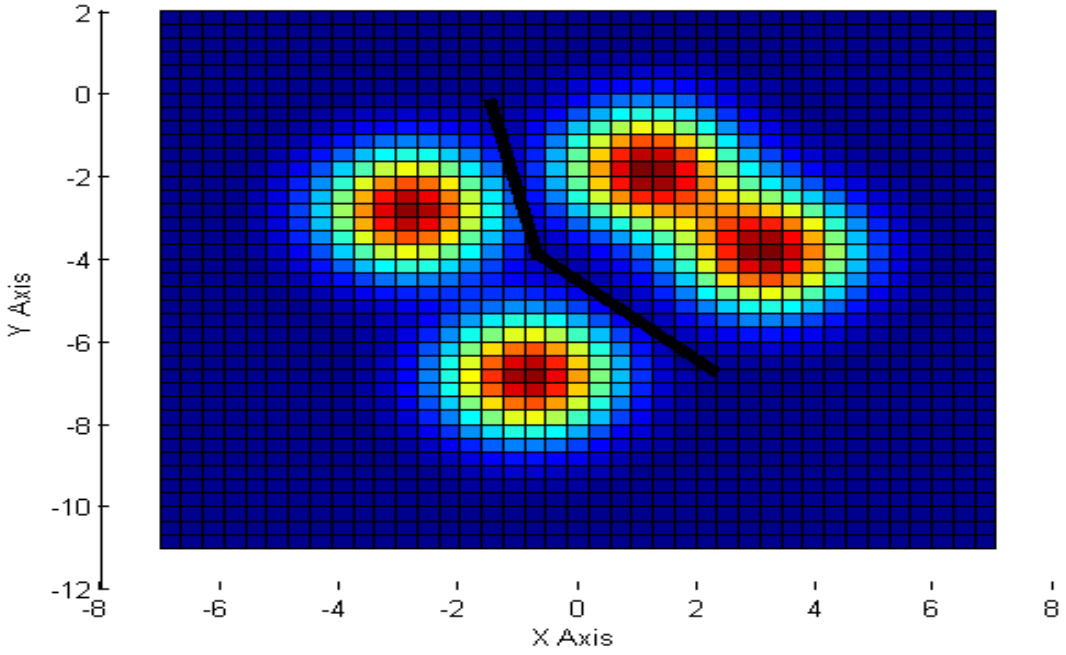


Figure 4. 7 Classification boundary estimated by PNN algorithm

When the PNN is used for training with the four labeled data points of Figure 4. 3, 'spherical' covariance Gaussians with identity covariance matrices are used as kernels as shown in Figure 4. 6 and the estimated classification boundary is shown in Figure 4. 7. Clearly, the estimated classification boundary is linear but doesn't resemble the true classification boundary (Figure 4. 5). In contrast to the performance of original PNN the decision boundary obtained by the proposed SSL algorithm is illustrated in Figure 4. 8. The decision boundary is closer to the Y-Axis than the decision boundary estimated by the PNN algorithm. Note that even though the EM algorithm results in different clusters in different runs, our experiments show that there is little variation in the final decision boundary predicted after the conclusion of all steps of the proposed SSL algorithm.

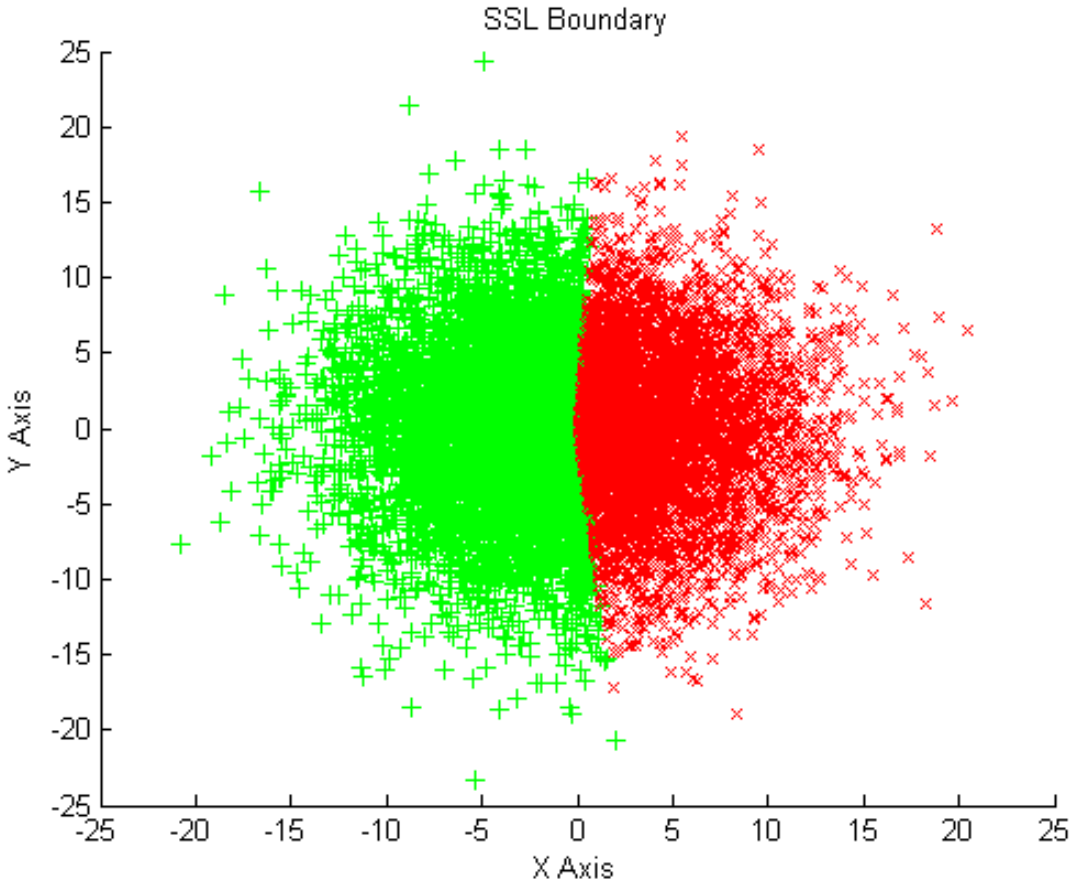


Figure 4. 8 Classification boundary estimated by the SSL-2 algorithm

4.6 Summary

In this chapter we established the core formulations which will enable us to validate the hypotheses to our research questions in Chapter 5. We introduced two Semi-Supervised Learning (SSL) algorithms, SSL-1 and SSL-2, for increasing the accuracy of PNN when less number of labeled data is available for training a PNN. SSL-1 algorithm assumes that the Gaussian kernels are centered at each labeled point. Furthermore, all the Gaussian kernels are 'spherical', or have covariance of the form $\Sigma = \sigma^2 I$, with the same value of σ^2. SSL-2 algorithm is more general and assumes that Gaussian kernels can take

different elliptical shapes and orientations and need not be centered at the labeled points. The Gaussian covariance matrices in SSL-2 are 'full' which allows all the parameters in the covariance matrices to have different values allowing the Gaussian kernel ellipses to have different shapes, sizes and orientations.

The SSL-1 algorithm resembles the basic EM algorithm since it also consists of Expectation and Maximization steps. Before the Expectation and Maximization steps a classifier is trained by assuming a 'smoothing parameter' or σ^2 value for PNN. The current classifier is used to estimate the labels of unlabeled data in the expectation step. These labels are denoted 'soft-labels' since they will have a value between 0 and 1. These soft-labels corresponding to unlabeled data and the 'hard-labels', which is either 0 or 1, corresponding to the labeled data are combined together and used for estimating the new parameters of the smoothing parameter in the Maximization step. The Expectation and Maximization steps are iterated until convergence. The final smoothing parameter is the only required value for the PNN classifier.

In the first step of the SSL-2 algorithm, the labeled and unlabeled data are combined together after ignoring the labels of labeled data. The combined data set is then used for unsupervised learning with the basic Expectation-Maximization algorithm, while allowing for 'full' covariance matrices. We initialize the number of clusters in EM as the number of labeled data. The outputs from EM algorithm are prior probabilities and means and covariance matrices for the component clusters. The second step involves finding the posterior probability of each of these clusters at the labeled points. These posterior probabilities are a measure of *responsibilities* of each Gaussian Cluster at each of the labeled point. In the third step, we calculate the weighted average of the Gaussian kernels

based on Mahalanobis distance of the test point from each of the labeled point. The weights used with the Gaussian kernels based on Mahalanobis distances are the responsibilities that were calculated in the second step. In the fourth and final step, we use the Bayes decision rule to assign a label to the test point based on which class had a higher probability for that test point.

SSL-2 algorithm was shown to approximate the true classifier, the Y-Axis, more accurately than the basic PNN algorithm. In Chapter 5 advanced numerical examples will demonstrate the superiority of both SSL-1 and SSL-2 over PNN in quintessential classification problems that might occur in reliability estimation problems. We will also show how the proposed algorithms can be used during the design of complex engineering system.

CHAPTER 5

VALIDATION EXAMPLES

In the previous chapter we proposed SSL-1 and SSL-2, Semi-Supervised Learning (SSL) algorithms, which use labeled and unlabeled data in order to reduce the computational cost of the overall reliability estimation process because overall they will require lesser number of labeled points in order to achieve the same level of accuracy as a PNN trained with only labeled data. In order to validate that the level of accuracy achieved using these algorithms is significantly higher than PNN we will test these algorithms with two analytical examples. In Section 5.1 we describe an analytical problem which is linear in nature and resembles reliability estimation problems where the failure domain is continuous. This example is used to validate the efficacy of the SSL-1 algorithm. The second example is described in Section 5.2 where the failure domain is disjoint. This example is used to validate the efficacy of the SSL-2 algorithm. Specifically, we demonstrate how the performance of SSL-2 changes as the number of unlabeled and labeled data is changed. Then, we demonstrate that SSL-2 performs better than Support Vector Machines (SVM) as the number of labeled data is increased. In section 5.3 we validate both the SSL methods, SSL-1 and SSL-2, with the help of a ten-bar example which is a classical problem in reliability estimation and has been used by various authors in order to validate their methods [42]. We take up the task of using these methods for the design of complex structural systems in Sections 5.4. In Section 5.4 we will show the usage of topology optimization methods for the design of a meso-scale compliant gripper that can be used for biological applications. A buckling limit state function is used for formulating the reliability constraint for the case of Reliability-based Topology

Optimization. The resulting mechanism is compared to the mechanism obtained by a Deterministic Topology Optimization procedure in order to validate our hypothesis to the Secondary Research Question 2.

5.1 Continuous Problem in Two Dimensions

In this example, the applicability and usefulness of the proposed SSL architecture using PNN with the novel learning algorithm will be demonstrated. Consider a limit state function [116] with two random variables,

$$g(u_1, u_2) = -3.8 + \exp(u_1 - 1.7) - u_2 \tag{5.1}$$

where u_1 and u_2 are assumed to be Gaussian random variables ~ N ([0, 0], 100) .

This example consists of two phases, namely the training phase and test phase. The training phase will incorporate learning with the SSL framework where unlabeled data will be used to improve the accuracy of the PNN classifier with the aid of the proposed EM algorithm. In the training phase of the PNN, both labeled and unlabeled data is used. For creating the labeled dataset, 20 samples of u_1 and u_2 are generated using LHS with an assumption of Gaussian distribution ~ N ([0 0], 100). For these 20 data points, the response in Eq. (5.1) is evaluated. Accordingly, the system is considered to be safe if $g(u_1, u_2) < 0$, and the system is considered to have failed if $g(u_1, u_2) > 0$. The points in the safe region can be given a class label of $w_i = 0$ (class 1), and the points in the unsafe region can be given a class label of $w_i = +1$ (class 2). These 20 data points constitute the labeled dataset. This labeled dataset will be augmented with 400 unlabeled data points which are sampled with the same Gaussian distribution as the labeled data.

The SSL-1 algorithm is then used to train the PNN with the proposed modified EM algorithm. The algorithm starts with training an initial classifier with a smoothing parameter value of 2. After 10 iterations of the proposed algorithm with both labeled and unlabeled data, the CCM matrix (matrix composed of probability values for each class) converged and the resultant classifier had a smoothing parameter value of 7.12. This concludes the training phase. For the test phase, 10,000 data points were generated with the Gaussian distribution ~ N ([0 0], 100) using LHS in order to test the classifier.

Figure 5. 1 shows the test data plotted in the decision space where the limit state function is illustrated. The space below the limit state function represents the safe region and the space above the limit state represents the failure region. The test data points classified by the SSL algorithm as safe are represented in circles and those classified as unsafe/failures are represented by dots.

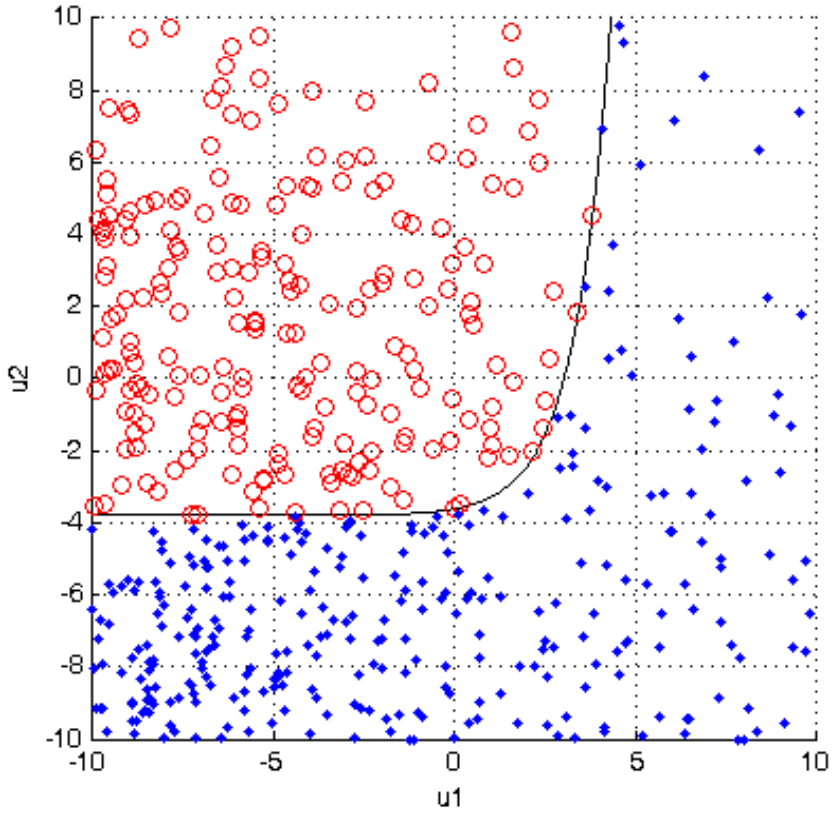


Figure 5. 1 Data points classified into safe and unsafe regions

The proposed method, based on SSL-1 classifier, has an accuracy of 88.94%, whereas the original PNN Classifier has an accuracy of 64.97%. This shows a 36.89% improvement in classification accuracy. These results are summarized in a *confusion matrix* [110] in Table 5. 1. Table 5. 1 shows that out of the 10,000 data points on which the original PNN algorithm was tested, 4421 points were in the safe region, whereas 5579 were in the failure region. The algorithm predicted that only 3924 points where in the safe region and 6076 points were in the failure region. Furthermore, out of the 3924 points, 1503 points actually belonged to the failure region. Hence there were 1503 misclassifications. Similarly, there were 2000 misclassifications in the prediction of failure class. Similar analogy also applies to the SSL-1 algorithm section of the confusion matrix where there

were 482 and 624 misclassifications in prediction of safe and failure class respectively. Table 5. 1 and Figure 5. 1 depict the efficacy of the proposed SSL-1 based classification method for limit state approximation, which has been shown here to perform better than the original PNN method.

Table 5. 1Confusion matrix for comparison between the conventional PNN and PNN with SSL

Confusion Matrix		Original PNN			SSL-1		
		Predicted Class		Total	Predicted Class		Total
		Safe	Failure		Safe	Failure	
Actual Class	Safe	2421	2000	4421	5565	624	6189
	Failure	1503	4076	5579	482	3329	3811
Total		3924	6076	10000	6047	3953	10000

In the next example we validate the efficiency of the SSL-2 algorithm on a disjoint failure domain problem.

5.2 Disjoint Problem in Two Dimensions

In this example, the applicability and usefulness of the SSL-2 algorithm will be demonstrated with an analytical problem which is a representative disjoint failure domain problem. Consider a limit state function with two random variables,

$$g(x_1, x_2) = x_1 x_2 - 4 \tag{5.2}$$

where x_1 and x_2 represent the random variables. The task assigned to classifiers is to estimate the classification boundary accurately. The classification boundary is the points where $g(x_1, x_2)$. The function represented in Eq. (5.2) is a rectangular hyperbola when represented in the x_1-x_2 dimensions. A graphical representation of the $g(x_1, x_2)$

function is shown in Figure 5. 2 where the classification boundary is the boundary between the region containing the points marked in circles and the region marked with points marked in squares.

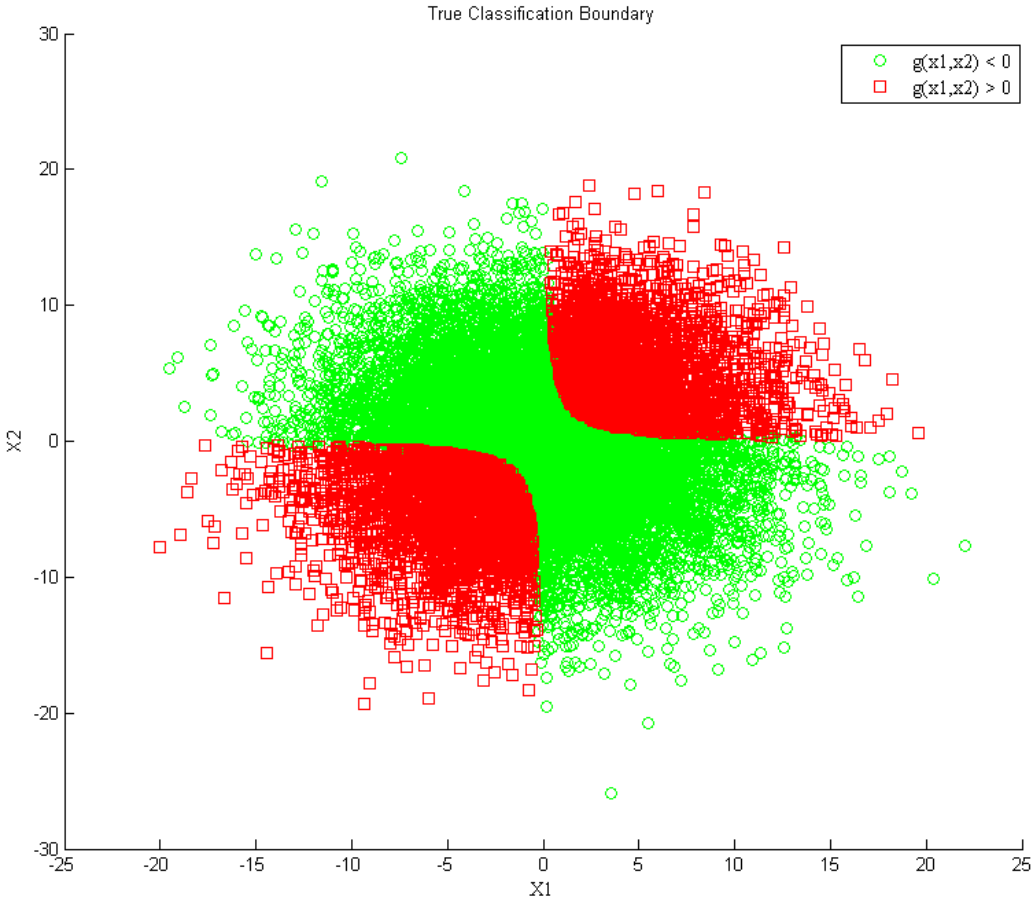


Figure 5. 2 True classification boundary

5.2.1 Analysis Process

This example consists of two phases, namely, training phase and test phase for the conventional PNN algorithm as well as the proposed SSL-2 algorithm. During the training phase, PNN will be trained using labeled data only whereas the SSL-2 algorithm will be trained using both the labeled and unlabeled data. Both the algorithms will be

tested on a common dataset. As mentioned earlier, one of the underlying assumptions of the proposed SSL based algorithm is that the labeled and unlabeled data points are sampled from the same Gaussian distributions. Hence, the same distributions are used for sampling both the labeled and unlabeled data points. The training labels for the labeled dataset are calculated using Eq. (5.2), where the sign of $g(x_1, x_2)$ gives the corresponding label. The points in the region where $g(x_1, x_2) < 0$ is given a class label of $w_i = 0$ (class *A*) and the points where $g(x_1, x_2) > 0$ is given a class label of $w_i = 1$ (class *B*). In Figure 5. 2 the region with circular points represent the region of class A and the region with square points represent the region of class B.

In total 10 labeled, 200 unlabeled points, and 10,000 test points are sampled. The labels for the 10 labeled and 10,000 test points are calculated using Eq. (5.2). All the labeled, unlabeled and test datasets are sampled from the bivariate Gaussian distribution ~ *N* ([0, 0], [30, 0; 0, 30]). The labeled data used for training is shown in Table 5. 2. As shown in the table, the training dataset contains equal number of points from each class.

Table 5. 2 Labeled data sampled for training

x1	x2	g(x1,x2)	Labels
1.7604	-3.9767	-11.001	0
1.8462	-8.7452	-20.145	0
-4.5722	-0.5416	-1.5237	0
-3.6530	1.8321	-10.693	0
-0.5531	-7.5897	0.1979	1
-6.2169	-5.6503	31.1274	1
-0.5569	-8.4132	0.6853	1
-5.8578	-9.7298	52.9952	1
6.8602	-2.6037	-21.862	0
9.6106	5.4285	48.1711	1

After demonstrating the advantage of the SSL2 algorithm in comparison to PNN in Section 5.2.2, we will demonstrate the performance variance in SSL-2 algorithm as the number of unlabeled data is changed in Section 5.2.3. We will also compare the SSL-2 algorithm with Support Vector Machines (SVM) in Section 5.2.4.

5.2.2 Comparison of SSL-2 with PNN

A smoothing parameter value of 1.00 was chosen to train the conventional PNN algorithm. The decision boundary estimated by PNN in this case is shown in Figure 5. 3. In this case, PNN assumes the shape of a linear classifier and it is unable to realize the discontinuity in the decision space for the given small number of labeled data points. Out of the 10,000 test data points, PNN misclassified in 2293 cases.

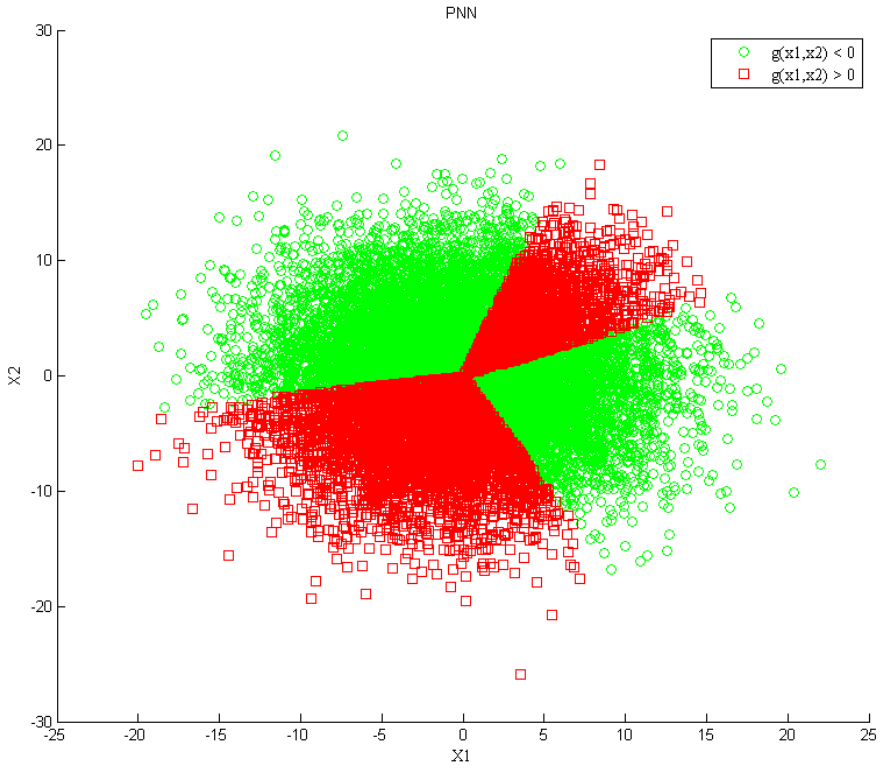


Figure 5. 3 Estimated classification boundary by PNN

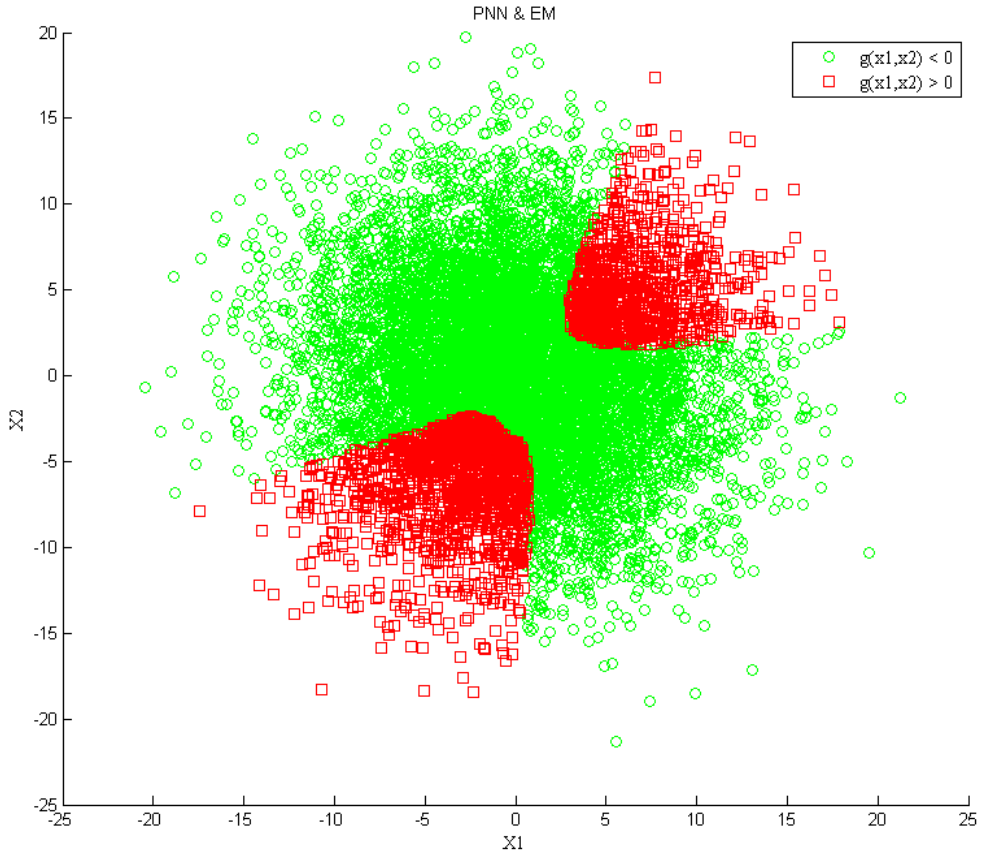


Figure 5. 4 Estimated classification boundary with SSL-2

In the SSL case, the set of 10 labeled data was used along with the set of 200 unlabeled data for the EM algorithm in the Step 1 of proposed algorithm. After conclusion of the training process where 10 different 'full' covariance matrices are learned and a fraction of each of these 10 covariance matrices is associated with each labeled data point during the classification process, the classifier was tested on the same test dataset containing 10,000 points. The average misclassifications were found to be 1723. Note that because the EM algorithm was used as a starting step for our algorithm, the final classifier performance will vary based on the initial starting point and convergence of EM. In our experiments, the number of misclassifications varied from

1187 in the best case scenario and 2189 in the worst case scenario with an average misclassification of 1723 in 10 runs. Note that even the worst case performance of our proposed algorithm (2189 misclassifications) is better than the performance of PNN algorithm (2293 misclassifications). The results from this test phase for 1723 misclassifications in the proposed algorithm are represented in the form of a confusion matrix in Table 5. 3. The decision boundary is also presented in Figure 5. 4.

Table 5. 3 Confusion matrix for comparison of the original PNN and SSL-2

Confusion Matrix		Original PNN			SSL-2		
		Predicted Class		Total	Predicted Class		Total
		Label 0	Label 1		Label 0	Label 1	
Actual Class	Label 0	3792	1104	4886	4012	874	4886
	Label 1	1189	3925	5114	849	4265	5114
Total		4971	5029	10000	4861	5139	10000

The confusion matrix has been divided into two sections in order to compare the results derived from the conventional PNN algorithm and the proposed SSL based PNN algorithm. The matrix is comprised of three columns. The second column represents the performance of the conventional PNN algorithm and the third column represents the performance of the proposed algorithm. Within the second column the off diagonal elements 849 and 874 represent the number of misclassifications. Hence the PNN model has a misclassification percentage of 22.93%. In comparison, the PNN with SSL algorithm has a misclassification rate of 17.23%. Hence the SSL based algorithm results in a 24.86% reduction in the misclassification rate. We want to restate that we compared the average performance of the proposed algorithm with the performance of PNN

algorithm. In case the number of labeled dataset is increased, the performance of both the algorithms increases at a proportional rate.

These results (Table 5. 3) confirm that the SSL based classification procedure provides sufficient accuracy compared to the results from the conventional PNN algorithm for this disjoint failure domain problem.

5.2.3 Performance of SSL-2 as Number of Unlabeled Data is Changed

In order to study the change in performance level of the proposed SSL-2 algorithm as the number of unlabeled data is varied, the same 10 labeled data shown in Table 5. 2 were used along with unlabeled data sampled using the bivariate Gaussian distribution ~ N ([0, 0], [30, 0; 0, 30]).

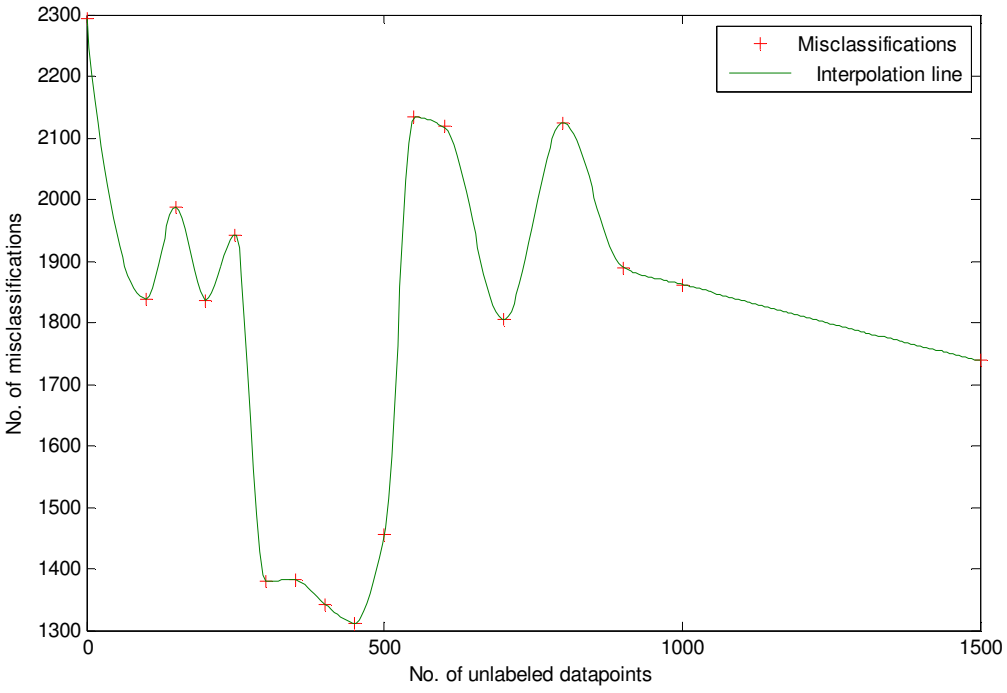


Figure 5. 5 Number of misclassifications with SSL-2 as no. of unlabeled data is varied

In order to test the accuracy of the algorithms, the algorithms were tested on 10,000 test data points, which were also sampled from the same Gaussian distribution. The number of resulting misclassifications quantifies the accuracy of the algorithm. Hence, the more accurate iteration is the one with the minimum number of misclassifications. The variation in the number of misclassifications as more unlabeled data is added is shown in Figure 5. 5. The exact values of the misclassifications are shown in Table 5. 4.

Table 5. 4 No. of misclassifications when 10 labeled points are used and the number of unlabeled points is increased for SSL-2

Unlabeled Data	Misclassifications
0	2293
100	1839
150	1988
200	1836
250	1943
300	1380
350	1382
400	1343
450	1312
500	1456
550	2134
600	2118
700	1804
800	2124
900	1889
1000	1862
1500	1738

Even when just 100 unlabeled data is added, the number of misclassifications decreases from 2293 to 1839 representing a 17.44% increase in accuracy. Maximum increase in accuracy occurs when 450 unlabeled data are added to the existing pool of 10 labeled

data. In this case there are only 1312 misclassifications representing a 42.78% increase in accuracy from the case when unlabeled data are not used at all (case of PNN). In case of SSL, the maximum number of misclassifications occur when 800 unlabeled data are used. In this case there are 2124 misclassifications, which is a 7.37% improvement over the case when only labeled data is used for PNN. Hence it can be concluded that SSL-2 will always improve the accuracy of PNN.

From the above discussion, an important question that arises is how many unlabeled data points should be included when conducting SSL-2 so that we obtain maximum accuracy? Note that the first step in SSL-2 is the Expectation Maximization algorithm where the number of required clusters is equal to the number of labeled data points. A popular rule of thumb for the EM algorithm is to have at least $2n^2$ number of unlabeled data points for clustering, where n represents the number of data clusters. According to this rule of thumb, when we have 10 labeled data points (10 clusters for EM algorithm), we should use at least 200 unlabeled data points. However, based on the results obtained from this example, we recommend using $4n^2$ data points for maximum accuracy with SSL-2.

5.2.4 Performance of SSL-2 as Number of Labeled Data is Changed

It is generally expected that as the number of labeled points is increased, the accuracy of the machine learning algorithms will increase. In this section we demonstrate that the accuracy of the proposed SSL-2 algorithm also increases as the number of labeled data is increased. Based on the inference drawn from the last section, we will sample $4n^2$ unlabeled data points when n number of labeled points are used for training. The labeled, unlabeled and test data points are sampled using the same method that was used in the

last section. The number of labeled data, unlabeled data, the number of misclassifications and the corresponding accuracy are represented in Table 5. 5.

Table 5. 5 Change in accuracy of SSL-2 as number of labeled data points is increased

Labeled	Unlabeled	Misclassifications	% Accuracy
5	100	2689	0.7311
6	144	2502	0.7498
7	196	2005	0.7995
8	256	1926	0.8074
9	324	1724	0.8276
10	400	1540	0.846
15	900	1464	0.8536
20	1600	1330	0.867
25	2500	1220	0.878
30	3600	1091	0.8909
40	6400	911	0.9089
50	10000	869	0.9131
60	14400	553	0.9447
70	19600	550	0.945

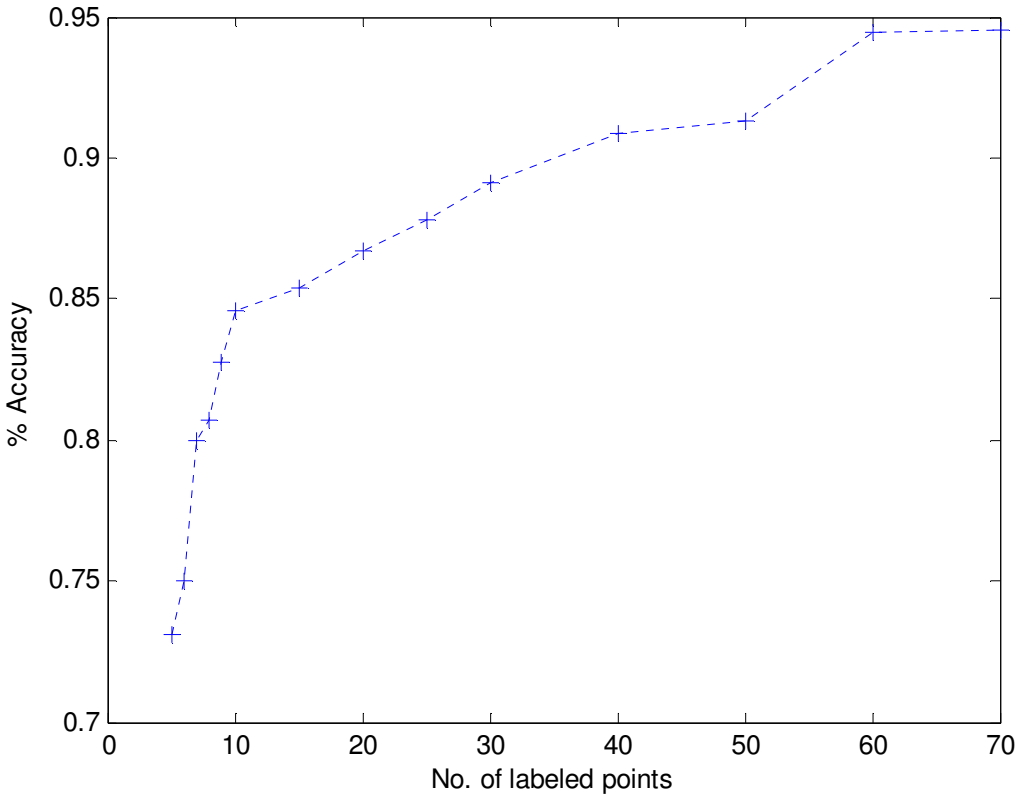


Figure 5. 6 Increase in accuracy as the number of labeled data is increased

The accuracy is represented as the number of correct classifications from the 10,000 test data points that were sampled earlier. The variation of accuracy with the number of labeled points is also shown in Figure 5. 6. Hence it can be concluded that for this example, the accuracy of the SSL-2 algorithm converges at 95% with respect to the test date. For different problems the accuracy will converge to different levels, hence, the benefit of adding additional labeled points will decrease drastically after a certain point.

5.2.5 Comparison of SSL-2 with SVM

Table 5. 6 shows the number of misclassifications for SSL-2, PNN and Support Vector Machines (SVM) when trained with 10, 20, 30, 40 and 50 labeled data points. The same set of 10 labeled points that were used in Sections 5.2.1 were used in this comparison

also. It is evident from this comparison that SSL-2 gives better results in all cases since the number of misclassifications for SSL-2 is the least in all cases. The number of misclassifications is plotted against the number of labeled data used in each case in Figure 5. 7. These results show that SSL-2 performs better than SVM as well as PNN as the number of labeled data points is increased.

Table 5. 6 Comparison of SSL-2 with PNN and SVM

Labeled Data	Unlabeled Data	SSL-2	PNN	SVM
50	10000	869	2081	917
40	6400	911	1560	1329
30	3600	553	1787	1983
20	1600	1606	4190	2062
10	400	1343	2293	2011

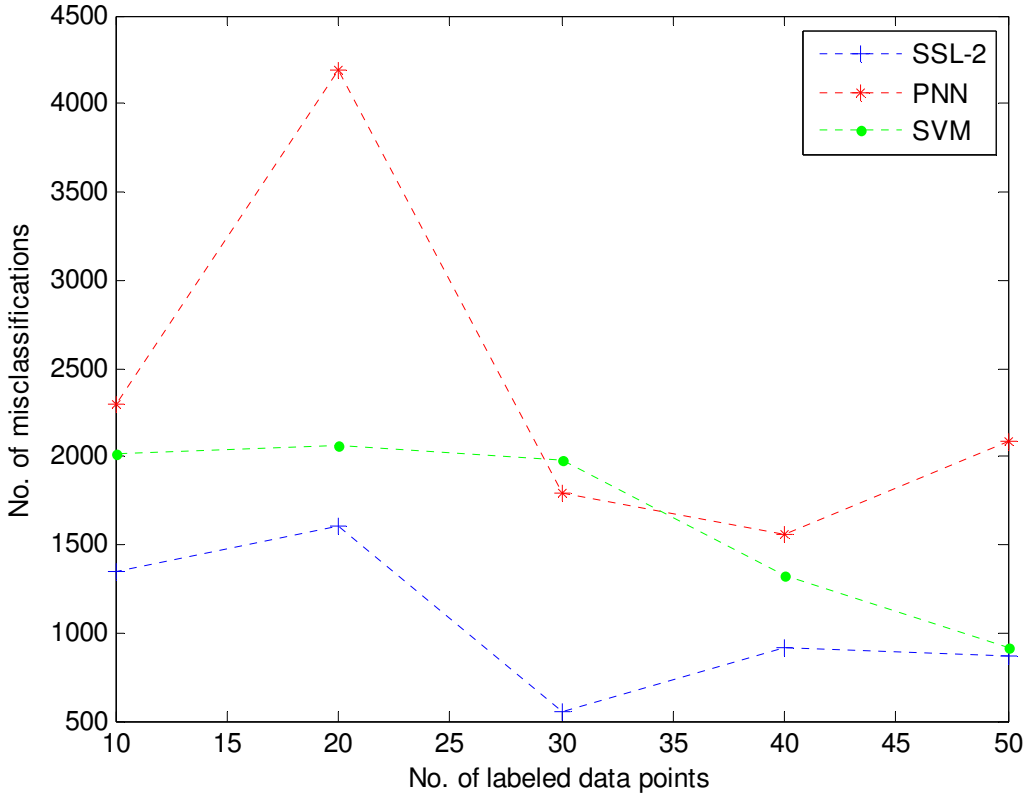


Figure 5. 7 No. of misclassifications for SSL-2, PNN and SVM

5.3 Ten Bar Truss Example

The ten-bar truss structure shown in Figure 5. 8 is used to validate the reliability estimation capability of the proposed algorithms. Since the horizontal, vertical, and diagonal members are cut from three different aluminum rods, the cross-sectional areas A1, A2 and A3 are considered design variables. These three variables can be potential sources of uncertainties. Therefore, the cross-sectional areas of the horizontal members, A1, vertical members, A2, and diagonal members, A3, are assumed random. The random quantities of these three variables are considered as having Gaussian distributions with the mean values of 13, 2 and 9 in^2, respectively. The coefficient of variation (COV) is 0.1

for all the three variables. Young's modulus, material density, length and load are assumed deterministic:

- Force: P = 100,000 lb
- Length: L = 360 in
- Young's modulus: $E = 10^7$ psi (for Aluminum)
- Material density: ρ = 0.1 lb/in^3
- Allowable stress for members 3 and 7: s_{allow} = 20,000 psi
- Allowable tip displacement at Node (2): d_{allow} = 4.0 in

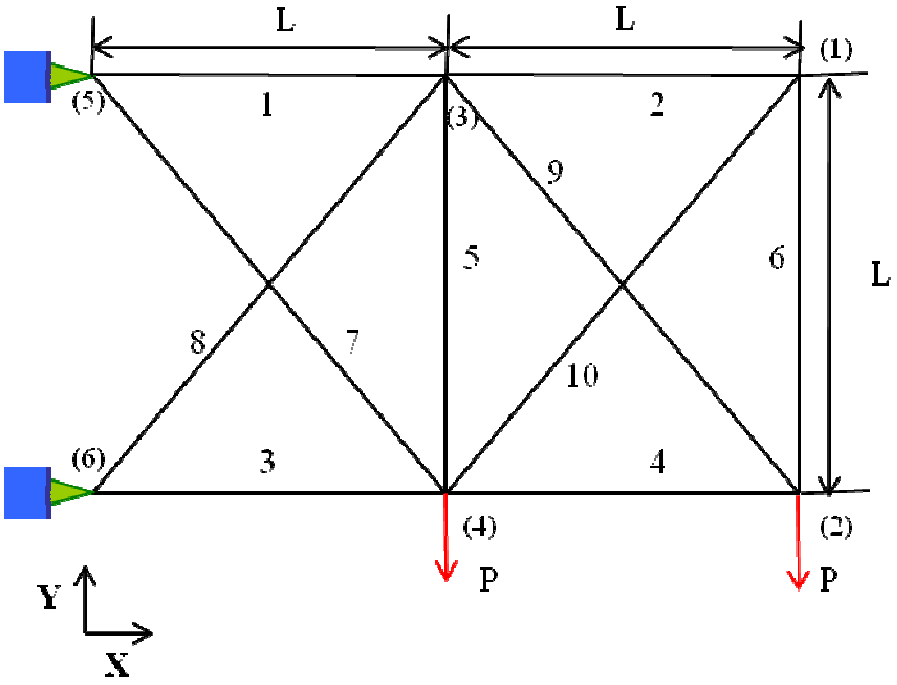


Figure 5. 8 Ten-bar truss structure

The reliability analysis is conducted to check the statistical characteristics of the displacement and stress-constraints at the optimum design. To set up the limit-state

function, the following closed-form analytical expressions are derived using the finite element analysis procedure. The limit state function for the tip displacement at Node (2) is given by

$$g_{disp} = \frac{PL}{A_1 A_3 E}\left\{\frac{4\sqrt{2}A_1^3\left(24A_2^2 + A_3^2\right) + A_3^3\left(7A_1^2 + 26A_2^2\right) + }{D_T}\right.$$

$$\left.\frac{4A_1A_2A_3\left\{\left(20A_1^2 + 76A_1A_2 + 10A_3^2\right) + \sqrt{2}A_3\left(25A_1 + 29A_2\right)\right\}}{D_T}\right\} - d_{allow} = 0 \qquad (5.3)$$

where $D_T = 4A_2^2\left(8A_1^2 + A_3^2\right) + 4\sqrt{2}A_1A_2A_3\left(3A_1 + 4A_2\right) + A_1A_3^2\left(A_1 + 6A_2\right)$

The limit-state functions for the stress in member 3 and member 7 are given as

$$g_3 = \frac{P}{A_1}\left\{2 + \frac{A_1A_2A_3\left(2\sqrt{2}A_1 + A_3\right)}{D_T}\right\} - s_{allow} = 0 \qquad (5.4)$$

and

$$g_7 = \frac{P}{A_3}\left\{2 + \frac{\sqrt{2}A_1A_2A_3\left(2\sqrt{2}A_1 + A_3\right)}{D_T}\right\} - s_{allow} = 0 \qquad (5.5)$$

5.3.1 Comparison of PNN and SSL-1

Different values were used for allowable constraints (s_{allow} and d_{allow}) to find the corresponding P_f values using Monte-Carlo Simulation (MCS), PNN and SSL-1. The probability of failure values calculated using SSL-1 are compared to the MCS values which were found to be the same as reported in Ref. [42]. The P_f values are also

calculated using the original PNN algorithm with 100 labeled data points. In order to calculate the P_f value using SSL-1, 20 different combinations of cross-sectional areas were sampled in each case using LHS. These values were used to compute the limit state function using Eqs. (5.3-5.5). The sampled dataset and the responses computed using Eqs (5.3-5.5) constitute the labeled dataset. This labeled dataset was augmented using 800 unlabeled data points that were sampled using the same distribution as the labeled data.

Table 5. 7 Probability of failure values for tip displacement at node 2 using SSL-1

Tip displacement at Node 2			
Displacement Limit	MCS(1,000,000)	PNN	SSL-1
3.0	0.9996	0.9982	0.9996
3.5	0.8442	0.8238	0.8177
3.7	0.5244	0.5000	0.5012
4.0	0.1764	0.1639	0.1622
4.5	0.0072	0.0061	0.0058

Table 5. 8 Probability of failure values for stress on truss member 3 using SSL-1

Stress in Member 3			
Stress Limit	MCS(1,000,000)	PNN	SSL-1
15000	0.7741	0.7739	0.7732
16124	0.5010	0.5000	0.5000
20000	0.0259	0.0263	0.0241
21000	0.0100	0.0103	0.0098
22000	0.0034	0.0045	0.0041

Table 5. 9 Probability of failure values for stress on truss member 4 using SSL-1

Stress in Member 4			
Stress Limit	MCS(1,000,000)	PNN	SSL-1
17223	0.49918	0.5123	0.5023
20000	0.07932	0.0801	0.0811
21000	0.03456	0.0341	0.0343
22000	0.0142	0.0145	0.0138

The results obtained using SSL are compared with MCS and PNN in Table 5. 7 and Table 5. 8 and Table 5. 9 for the displacement limit state at node 2 and stress limit states in truss members 3 and 4, respectively. Comparing the values show that SSL-1 gives conformal and acceptable results while reducing the computational requirement drastically. In order to compare the accuracy of SSL-1 in comparison to PNN, we can assume that the values obtained by using MCS are the true probability of failure values. With this assumption, the Sum of Square Error (SSE) for SSL-1 is 0.0014 whereas that for PNN is 0.0012 in Table 5. 7. Similarly, in Table 5. 8 the SSE for SSL-1 is 5.72×10^{-6} and that for PNN is 2.53×10^{-6}. In Table 5. 9 the SSE for SSL-1 is 1.313×10^{-5} and SSE for PNN is 1.7304×10^{-4}. These results show that SSL shows conformal results with PNN. Note that these results have been obtained while using 100 labeled data points using PNN and just 20 labeled data points for SSL. Since most of the computational requirement in reliability-based design processes lies in the reliability estimation process, which in turn depends on the FEM process for determining the responses, it can be concluded that SSL-1 results in drastic reduction in computational requirement of the

reliability estimation process. The added advantage in using SSL-1 over PNN is that SSL-1 results in an 80% reduction in computation cost than the original PNN algorithm.

5.3.2 Comparison of PNN and SSL-2

To find the corresponding P_f values Monte-Carlo Simulation (MCS), PNN and SSL-2 were conducted. The P_f values are calculated using the original PNN algorithm with 20 labeled data points. These values were used to compute the limit state function using Eqs. (5.3-5.5). The sampled dataset and the responses computed using Eqs. (5.3-5.5) constitutes the labeled dataset.

Table 5. 10 Probability of failure values for tip displacement at node 2 using SSL-2

Tip displacement at Node 2			
Displacement Limit	MCS(100,000)	PNN	SSL-2
3.0	0.9996	0.9982	0.9990
3.5	0.8442	0.8238	0.8441
3.7	0.5244	0.5000	0.5240
4.0	0.1764	0.1639	0.1761
4.5	0.0072	0.0061	0.0074

Table 5. 11 Probability of failure values for stress on truss member 3 using SSL-2

Stress in Member 3			
Stress Limit	MCS(1,000,000)	PNN	SSL-2
15000	0.7741	0.7739	0.7739
16124	0.5010	0.5000	0.5009
20000	0.0259	0.0263	0.0260
21000	0.0100	0.0103	0.0100
22000	0.0034	0.0045	0.0036

Table 5. 12 Probability of failure values for stress on truss member 4 using SSL-2

Stress in Member 4			
Stress Limit	MCS(1,000,000)	PNN	SSL-2
17223	0.49918	0.5123	0.5001
20000	0.07932	0.0801	0.07935
21000	0.03456	0.0341	0.0371
22000	0.0142	0.0145	0.0147

The results obtained using SSL-2 are compared with MCS and PNN in Table 5. 10, Table 5. 11 and Table 5. 12 for displacement limit state at node 2 and stress limit states on members 3 and 4. The same set of labeled and unlabeled data used for the results in Table 5. 7, Table 5. 8 and Table 5. 9 were also used for this section. Comparing the values show that SSL-2 gives accurate results while requiring less number of labeled data points.

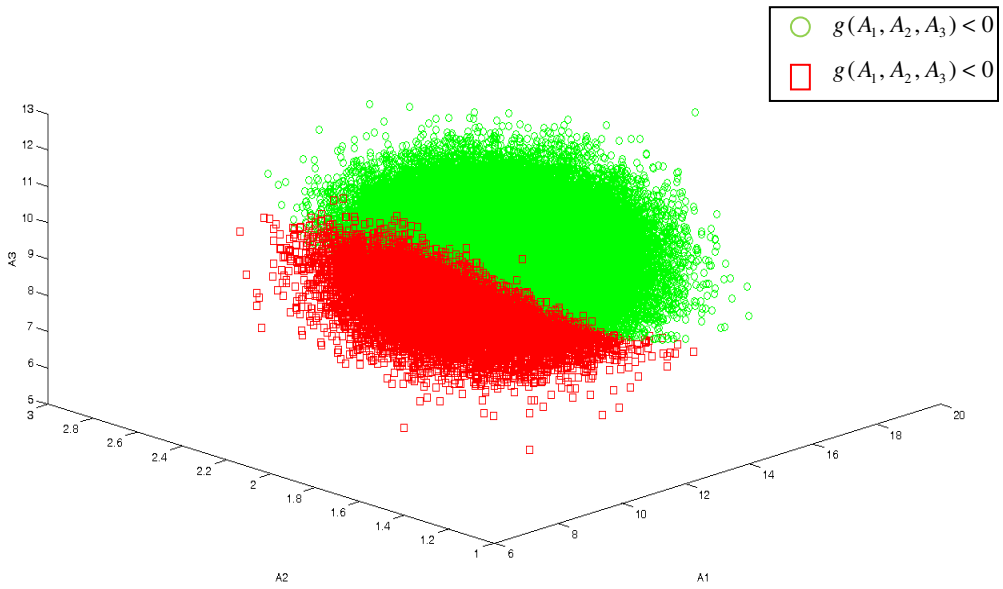


Figure 5. 9 Classification boundary with displacement limit state function

In order to compare the accuracy of SSL-2 in comparison to PNN, we can assume that the values obtained by using MCS are the true probability of failure values. With this assumption in Table 5. 10, the Sum of Square Error (SSE) for SSL-2 is 6.6 X 10^{-7} whereas that for PNN is 0.0012. Similarly, in Table 5. 11 the SSE for SSL-2 is 1.0 X 10^{-7} and that for PNN is 2.53 X 10^{-6}. In Table 5. 12 the SSE for SSL-2 is 7.5489 X 10^{-6} and SSE for PNN is 1.7304 X 10^{-4}. These results show that SSL-2 shows superior results than PNN. Note that these results have been obtained while using 100 labeled data points using PNN and just 20 labeled data points for SSL-2. As in the case of SSL-1, in the case of SSL-2 the set of labeled data is augmented with 800 unlabeled data, which are sampled from the same distribution. An increased accuracy in both the methods should be expected as the number of labeled data is increased. The predicted classification boundary by the proposed SSL algorithm on 10,000 test data is shown in Figure 5. 9. As expected the classification boundary seems to be a linear hyperplane.

The results from this reliability estimation problem are summarized in Table 5. 13. The SSE values obtained for PNN using 100 labeled data points are compared with the values obtained by SSL-1 and SSL-2 when 20 labeled data points are used with 800 unlabeled data points. The table shows that the results obtained by using PNN and SSL-1 are similar whereas the results obtained by SSL-2 are more accurate than both PNN and SSL-1.

Table 5. 13 SSE comparison between PNN, SSL-1 and SSL-2

SSE of PNN, SSL-1 and SSL-2 for Limit State Functions			
Limit State Function	PNN	SSL-1	SSL-2
Displacement at Node 2	0.0012	0.0014	6.6×10^{-7}
Stress in Member 3	2.53×10^{-6}	5.72×10^{-6}	1.0×10^{-7}
Stress in Member 4	1.7304×10^{-4}	1.313×10^{-5}	7.5489×10^{-6}

These results demonstrate that both SSL-1 and SSL-2 can reduce the computational requirement of the reliability estimation process since both these methods take 80% less labeled data points for obtaining comparable or better results than PNN.

5.3.3 Example with Non-Gaussian Random Variables

In the previous sections, we assumed that all the uncertain parameters are represented by Gaussian distributions. However, in many cases uncertainties are represented by non-Gaussian distributions. Hence, in a complex structural system uncertain variables will be represented by a combination of different distributions.

In order to validate that the proposed SSL-2 algorithm will be effective when the uncertainties are represented by non-Gaussian distributions, we will assume that the

uncertain variables are represented by the distributions represented in Table 5. 14. Unlike the previous sections of this example, here we are assuming that the force, *P*, and Young's Modulus, *E*, are also random variables.

Table 5. 14 Uncertain variables and corresponding distributions

Random Variable	Distributions	Parameters	
A1	Uniform	[10,16]	
A2	Lognormal	Mean=10.1757	Std. Dev. = 92.8255
A3	Uniform	[6,12]	
P	Weibull	Shape=2.6901	Scale=1.1246 X 10^5
E	Gamma	Alpha=6.25	Beta=1.6 X 10^6

Using these uncertainty descriptions, 20 labeled data points were sampled for training the original PNN algorithm. SSL-2 algorithm was trained using these 20 labeled samples and 32,000 unlabeled data points. The labels of the samples were estimating the values of displacement limit state which is given in Eq. (5.3). The probability of failure estimated by using 1,000,000 samples of MCS, PNN while using 20 labeled data points and SSL-2 while using 20 labeled data points and 32,000 unlabeled data points are represented in.Table 5. 15 Probability of failure values for different displacement limit states

Tip displacement at Node 2			
Displacement Limit	MCS(1,000,000)	PNN	SSL-2
3.0	0.6386	0.5933	0.5942
3.5	0.5410	0.4173	0.4956
3.7	0.5057	0.3975	0.4636
4.0	0.4529	0.3846	0.4295
4.5	0.3758	0.4160	0.4144

Specifically, assuming that the probability of failure values calculated by using MCS are the true values, the Sum of Square Errors (SSE) value for PNN is 0.0353 and the SSE value for SSL-2 is 0.0078. Hence it can be concluded that SSL-2 predicts more accurate values of probability of failure values than PNN in cases where the uncertainties are sampled from different distributions. Since it is highly likely that in modern structural systems, the uncertain parameters might not be Gaussians and the uncertain parameters of the system are sampled from different systems, this example validates that the reliability of these complex structural systems can also be predicted by using the SSL-2 algorithm.

5.4 Compliant Meso-Scale Gripper Mechanism Design for Biological Applications

A mechanism is a mechanical device used to transfer motion, force or energy. A traditional rigid-body mechanism consists of rigid links connected at movable joints. Since energy is conserved between the input and the output (neglecting friction losses), the output force may be much larger than the input force, but the output displacement is much smaller than the input displacement. Like mechanisms, structures may also consist of rigid links connected at joints, but relative rigid-body motion is not allowed between the links.

A compliant mechanism also transfers motion, force or energy but unlike rigid body mechanisms complaint mechanisms gain at least some of their mobility from the deflection of flexible members rather than from movable joints only.

Compliant mechanisms can be considered for use in a particular application for a variety of reasons. The advantages of compliant mechanisms can be considered in two

categories: cost reduction (part-count reduction, reduced assembly time, and simplified manufacturing processes) and increased performance (increased precision, increased reliability, reduced wear reduced weight, and reduced maintenance). An advantage of compliant mechanisms is the potential for the dramatic reduction in the number of parts required to accomplish a specified task. Some mechanisms can be manufactured from an injection moldable material and be constructed of one piece. Compliant mechanisms also have fewer movable joints, such as pin (turning) and sliding joints. This results in reduced wear and need or lubrication. These properties make them easy to use in harsh conditions and places which are not easily accessible. Reducing the number of joints can also increase the mechanism precision, because backlash can be reduced and eliminated.

Because compliant mechanisms rely on the deflection of flexible members, energy is stored in the form of strain energy in the flexible members. This stored energy is similar to the strain energy in a deflected spring, and the effects of springs may be integrated into a complaint mechanism's design. In this manner, energy can easily be stored or transformed, to be released at a later time or in a different manner. It is possible to realize a significant reduction in weight by using compliant mechanisms rather than their rigid-body counterparts. This may be a significant factor in aerospace and other applications. Complaint mechanisms have also benefited companies by reducing the weight and shipping costs of consumer products.

5.4.1 Optimal Design of Compliant Mechanisms

As explained in the last section, topology optimization is one of the systematic methods for synthesizing compliant mechanisms with distributed compliance [117]. This method operates on a fixed finite element mesh of either continuum or discrete elements to

optimally distribute material in the designable region and thus defining a topology. In other words, each element is associated with a design variable that defines the element size or its contribution to the entire topology. The converged optimization result is supposed to drive the value of all the design variables either close to the lower and upper limits thus defining a definite topology. These homogenization-based methods were introduced by Bendsoe and Kikuchi [18] for designing topologies with maximum stiffness having a finite volume of material. This method was adapted by Ananthasuresh [117], Frecker et al. [20], Sigmund [118], Saxena [21] and others for generating topologies which have maximum displacement at a desired point. Displacement Amplifying Compliant Mechanisms (DaCMs) and compliant grippers have been designed using this approach. The objective function that needs to be minimized for designing compliant mechanisms from topology optimization is usually based on a trade-off between flexibility at a particular point to achieve deformation and stiffness to support the external load. This can be effectively captured by the following formulation

$$\text{Minimize} \quad MSE / SE \tag{5.17}$$

$$\text{where} \quad MSE = \int s^T \varepsilon_d \, d\Omega \tag{5.18}$$

$$\text{and} \quad SE = \int \frac{1}{2} s^T \varepsilon \, d\Omega \tag{5.19}$$

The symbols used above are explained next. Referring to Figure 5. 10, *SE* is the *strain energy*, Ω is the elastic continuum under the applied load F_{in} ; *MSE* is the *mutual strain energy* due to applied and unit dummy loads, F_{in} and F_d ; ε and s are the strain and the stress fields due to the load F_{in}, applied at point P_1 and ε_d is the strain field due to the unit

dummy load F_d applied at point P_2. It should be noted that *MSE* is numerically equal to the displacement of point P_2 in the direction of F_d due to the applied load, F_{in}. Saxena and Ananthasuresh [21] proposed an optimality property that emerges from this objective function and any general objective function of the type $f_1(MSE) + f_2(SE)$ and $f_1(MSE) / f_2(SE)$. The mutual strain energy (*MSE*) of each element signifies the contribution of that element towards the displacement of the desired output point, for a force applied at the input point. At the same time the strain energy (SE) of each element signifies the contribution of that element towards the stiffness at the input side. These two energy measures are conflicting as one demands stiffness and the other flexibility.

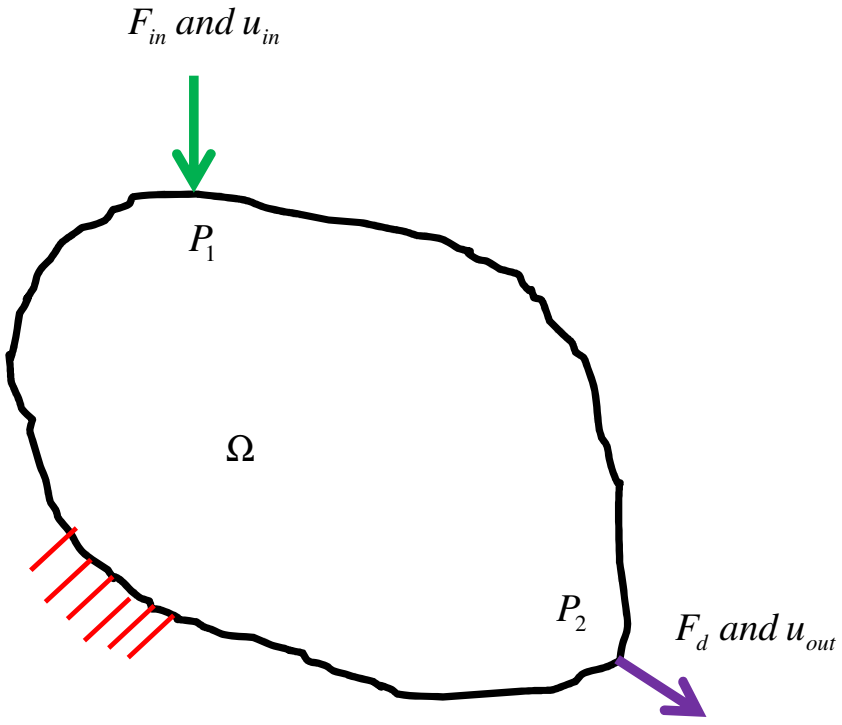


Figure 5. 10 Design domain and problem specification for a compliant mechanism with input at P_1 and output at P_2

The ground structure on which the optimization is performed can be made of frame elements or continuum finite elements. The continuum finite elements in 2D usually yields undesirable checker board patterns [19] and hinged regions unless special measures are taken. Furthermore, image processing is necessary to get the final topology that can be fabricated. The discrete ground structure based topology optimization method yields topologies with distributed compliance but has limited design space because of the fixed orientation of the beams in the ground structure.

In this dissertation, we will focus on the discrete topology optimization method for the design of compliant mechanisms. This method operates on a fixed mesh of finite elements and defines a design variable, which is associated with each element in the mesh. The optimization algorithm determines the value of the design variables. The values of the design variables define the optimal topology of the mechanism. The design obtained from topology optimization is specific to the design domain, loading, and boundary conditions. The design variables are driven towards the optimal topology by the objective function and the constraints, which are specific to the problem.

For a discretized finite element model of the continuum, the following equations can substitute Eqs. (5.18-5.19):

$$MSE = V^T KU \tag{5.20}$$

and

$$SE = 0.5U^T KU \tag{5.21}$$

where

$$KU = F_{in} \tag{5.22}$$

and

$$KV = F_d \tag{5.23}$$

Here U and V represent the displacements for the actual load F_{in} applied at the input degree of freedom and unit dummy load F_d applied at the output degree of freedom. Note that by definition of our dummy unit load F_d at the output point and the corresponding displacement V at the output point, Eq. (5.17), which is given below, can be rewritten as in Eq. (5.24) after neglecting the constant factor of 0.5 in the expression for Strain Energy in Eq. (5.21).

Minimize MSE / SE (5.17)

Minimize u_{out} / u_{in} (5.24)

where u_{out} and u_{in} are the displacements at the output and the input nodes respectively for the actual input load F_{in}. This objective function can be used in order to design mechanisms which have inputs and outputs along specific directions depending on the particular application. In the next section this formulation will be used for designing a compliant gripper mechanism that can be used for biological application.

5.4.2 Design of Complaint Gripper Mechanism for Biological Application

A biological cell is a complex machine that constitutes the basic building block of all organisms. It is self-contained with provisions for input and output between itself and its environment. Conventionally, microscopes have been used to study a cells size, shape, morphology and bio-chemical expressions [119]. Although much has been learnt about cells in this manner, it is not completely understood what the cells actually sense: is it stress, strain, strain energy or something else [120]. These mechanical tests increase our

general understanding of cells and how they operate when they are under the influence of externalities. Therefore there is a need for mechanical testing of cells which requires mechanical manipulation at the single cell level as well as the organelle.

Of all the techniques available now for grasping and manipulating single cells, aspiration using pipettes is the most widely used [121]. It involves using fine hollow needles and applying a negative pressure to partially suck the cell into the hollow tube or hold it in place if the cell is much bigger than the exit diameter of the pipette and the cell membrane is sufficiently stiff. Even though it is simple to hold the cell and apply a mechanical load, the implications are not known. Other researchers have used magnetic or dielectric particles that get attached to the cells and then magnetic, electric or optical fields are varied to hold, move and manipulate cells [122-125]. All these methods described are intrusive. Atomic Force Microscope (AFM) and other probes are also used to test cells mechanically but only cell membranes can be studied with these techniques [126]. Another method that has fewer disadvantages than the above mentioned ones is using flow channels of appropriate shapes to squeeze the cells.

None of the methods explained thus far have the versatility and manipulation capability that mechanical grasping fingers offer [127]. Since micro and meso fabrication technologies and micro-actuation are fairly developed today it is worth considering micro and meso-grippers in order to manipulate cells and biological entities. Furthermore, with grippers it is possible not only to just grasp the cell but also to manipulate it to make it undergo gross motions such as rigid-body translation and rotation and deformations, both elastic and plastic.

Although it is possible to design miniature grippers with joints and assemble the parts together, it is more economical to design a gripper through the compliant mechanism route since the absence of joints alleviates the gripper from friction and wear. Another advantage of designing miniature grippers using compliant mechanisms is that with compliant mechanisms it is possible to measure forces with its visually captured deformation data [128].

For the design of the miniature gripper mechanism we want to minimize the number of actuation points so that controlling the gripper becomes easy. Hence only one input is considered in the following example. Each of the elements of the groundtruss is assumed to behave as a frame element and we assume only small deformations so that linear Finite Element Analysis is valid. The initial design space and the groundtruss considered for this design are shown in Figure 5. 11.

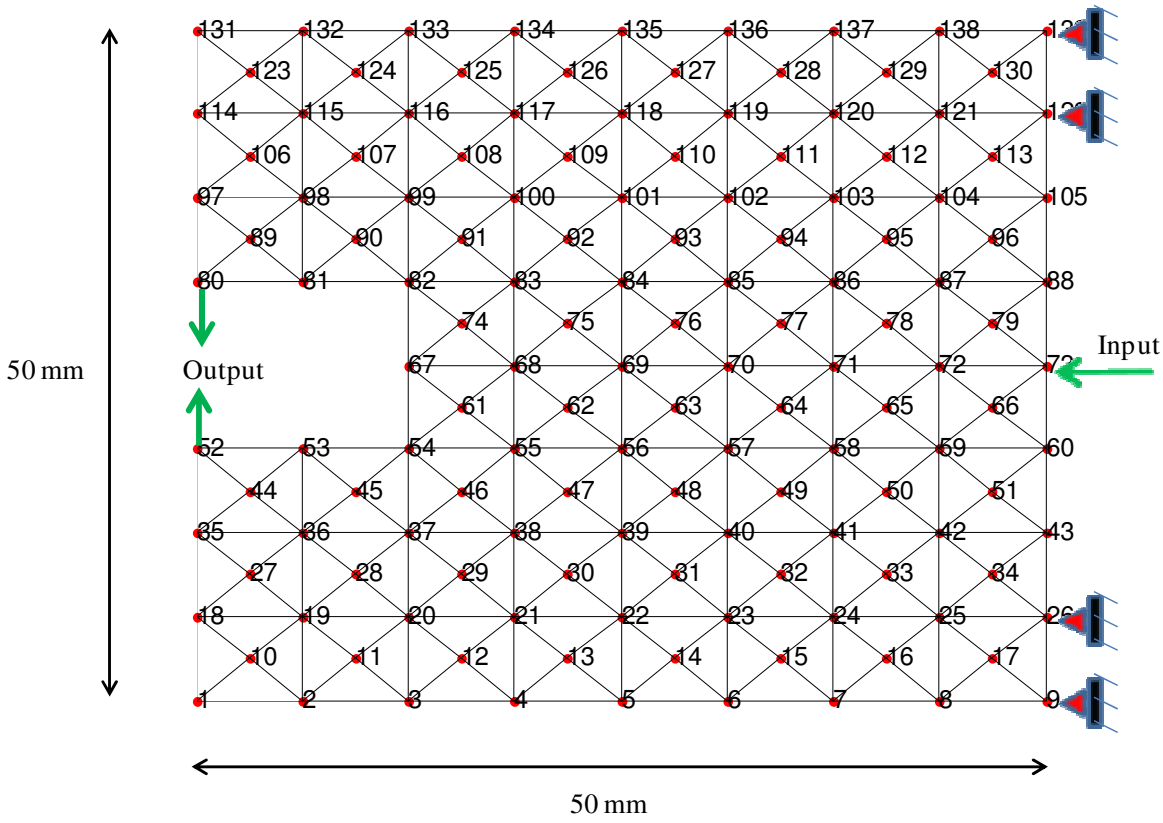


Figure 5. 11 Full Groundtruss with input and desired output

The overall dimension of the design space is a 50 mm X 50 mm square. There are 139 nodes and 378 elements in the initial groundtruss as shown in Figure 5. 11. The input force is applied at Node no. 73 in the direction shown in the figure and it is expected that the gripper nodes located at Node nos. 52 and 80 move towards each other so that an object can be gripped. Also Nodes 9, 26, 122 and 139 and fixed—suppressing all the degrees of freedom at these nodes. For this design example we will assume that the groundtruss consists of 2D frame elements, with three degrees of freedom per node. These three degrees of freedom are the directions of axial displacement, the off-axis perpendicular displacement and the end rotation. All of the elements have the same out-of-plane thickness of 4.0 mm. The widths of each frame element are allowed to be

adjusted by the optimization algorithm. Hence, the widths of the elements are the design variables. We decided on a cut-off length of 0.5 mm for the widths based on our manufacturing capabilities available at this time. Hence all the elements that have widths lower than 0.5 mm will be assumed to be not present in the final topology determined by the optimization algorithm. All elements with widths larger than 0.5 mm will stay in the final topological solution. The Young's Modulus of material used in this analysis is 9.65 GPa corresponding to Accura Bluestone Plastic which is widely used in Stereolithography applications.

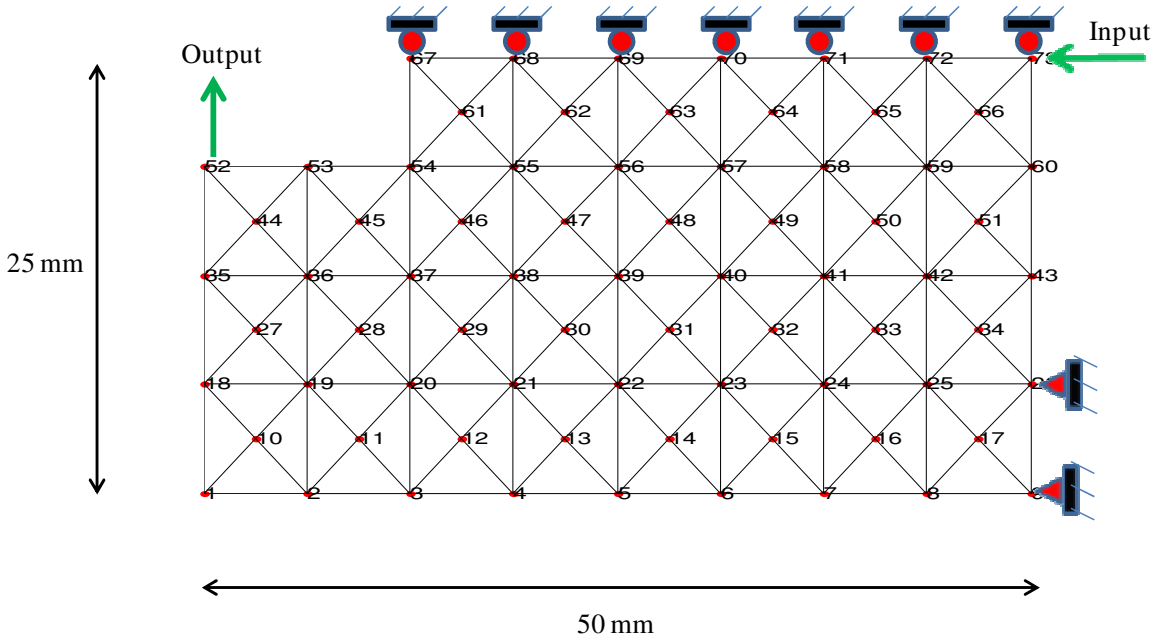


Figure 5. 12 Reduced Groundtruss that is considered for Topology Optimization

It is clear from Figure 5. 11 that there is a plane of symmetry for this design analysis problem and after considering the symmetry plane the groundtruss from Figure 5. 11 can be reduced to the one shown in Figure 5. 12.

The groundtruss in Figure 5. 12 consists of 73 nodes and 192 frame elements. Because of the symmetry boundary conditions the degrees of freedom corresponding to the vertical motion at Nodes 67, 68, 69, 70, 71, 72 and 73 are suppressed.

5.4.3 Deterministic and Reliability-based Topology Optimization Design Results

The analysis was conducted in two stages. In the first stage the deterministic topology optimization problem was solved and in the second stage the Reliability-based Topology Optimization problem was solved. The problem formulations used for both these stages and the solutions obtained are given below:

Deterministic Topology Optimization Problem Formulation:

$$\textbf{\textit{Minimize:}} \quad \frac{MSE}{SE} = \frac{V^T KU}{0.5U^T KU} \tag{5.25}$$

$$\textbf{\textit{Subject to:}} \quad \sum_{i=1}^{N} t\, wid_i\, L_i - V^* \le 0 \tag{5.26}$$

$$10^{-4} \le wid_i \le 4 \tag{5.27}$$

$$KU = F \tag{5.28}$$

$$KV = F_d \tag{5.29}$$

where t denotes the thickness of all frame elements used in this structure and V^* denotes the amount of material that can be used to design the final structure. The input force F applied at the designated point is equal to 20 Newtons. The solution obtained from the deterministic topology optimization is shown in Figure 5. 13, after taking off all elements for which the widths were less than 0.5 mm in the final solution.

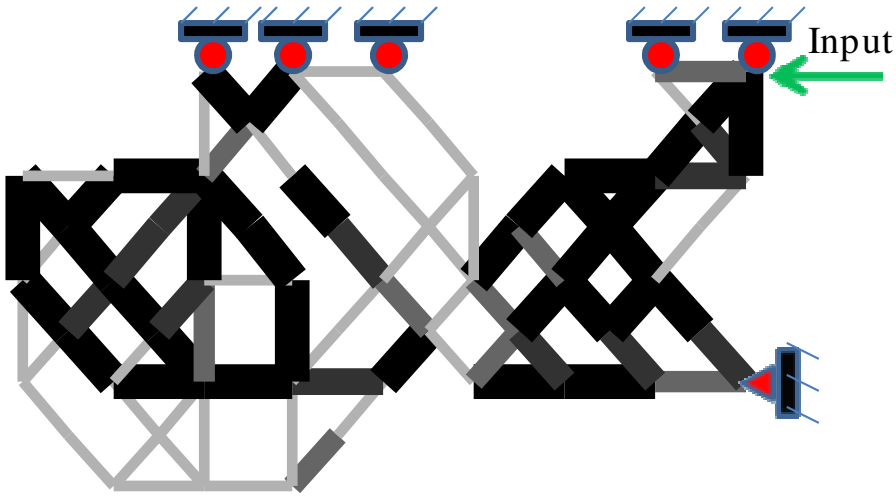


Figure 5. 13 Reduced groundtruss solution for gripper mechanism

In this solution the width of the resulting final solutions is shown as the darkest if the frame elements have converged to the upper bound, i.e., 4 mm. If the frame elements converged to a value between 4 mm and 3 mm, they are represented as thinner and lesser dark members. The members that converged to values between 3 mm and 2 mm are represented with lighter color in the figure. Finally, all elements whose widths are between 2 mm and 0.5 mm are shown with the thinnest thickness and the faintest color in Figure 5. 13. All the members with width less than 0.5 mm are removed from the final design, thus modifying the topology of the structural system. The wireframe structure showing this solution is also shown in Figure 5. 14 which shows the new Node numbers and element numbers for the obtained solution. The final solution has 56 nodes and 88 frame elements after implementing the cut-off length post-processing step. The resulting solution was analyzed in ABAQUS with the same input conditions in order to validate that the motion of the mechanism is indeed in the specified direction.

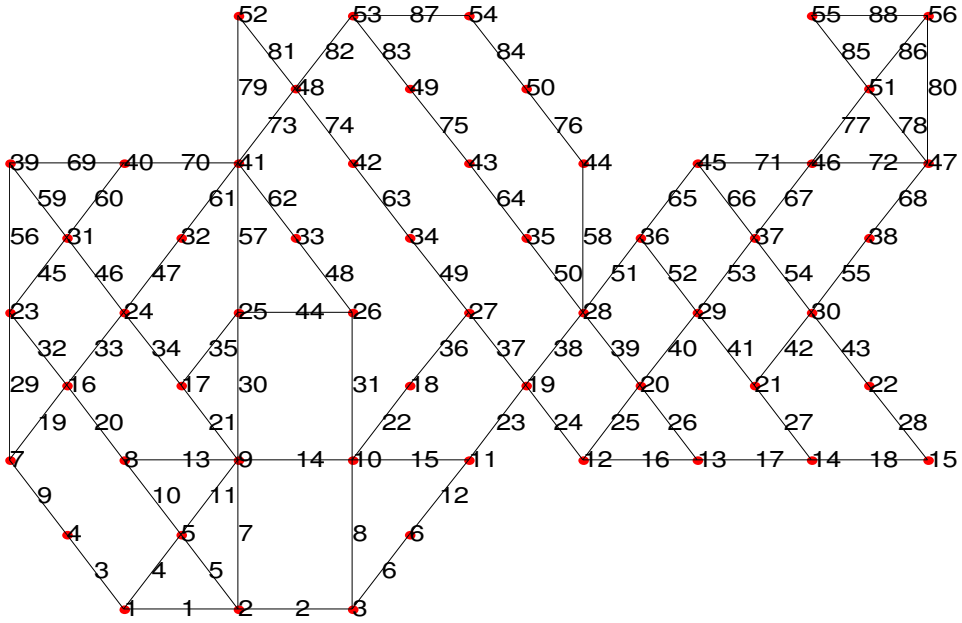


Figure 5. 14 Wireframe representation of the deterministic solution

The undeformed and deformed plots from ABAQUS are shown in Figure 5. 15. As expected, the nodes on the symmetry line are still on that line after deformation and the total displacement magnitude of Node 39 is 2.21 X 10^{-2} mm for a 20 N force applied at Node 56. In particular, the X-component of displacement at Node 39 is 1.0944 X 10^{-2} mm and the Y-component of displacement at Node 39 is 1.9301 X 10^{-2} mm. This analysis was repeated while considering geometric nonlinearity in ABAQUS and the X-component of displacement was found to be 1.0985 X 10^{-2} mm and the Y-component of displacement was found to be 1.92 X 10^{-2} mm. Since these values are very similar, it can be concluded that linear FEM analysis is a good engineering assumption in this case.

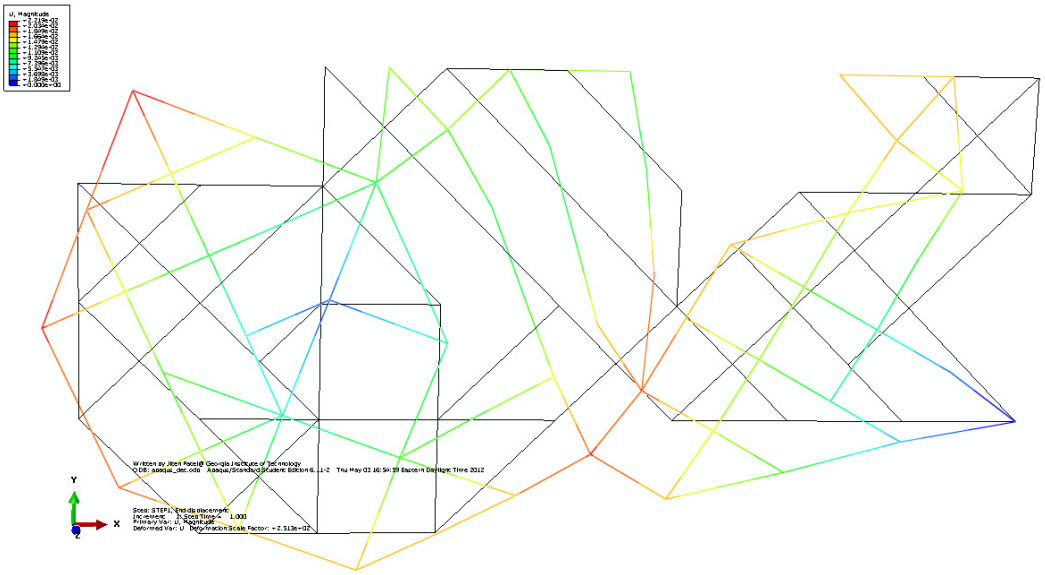


Figure 5. 15 Undeformed and deformed plot with displacement contours from ABAQUS using linear FEM

Also, the contours for the axial forces along the frame axis are plotted in Figure 5. 16. Note that there are five members which have high stress concentrations and are plotted in red in the figure. Four of these five members have an axial force value of 19.99 Newtons.

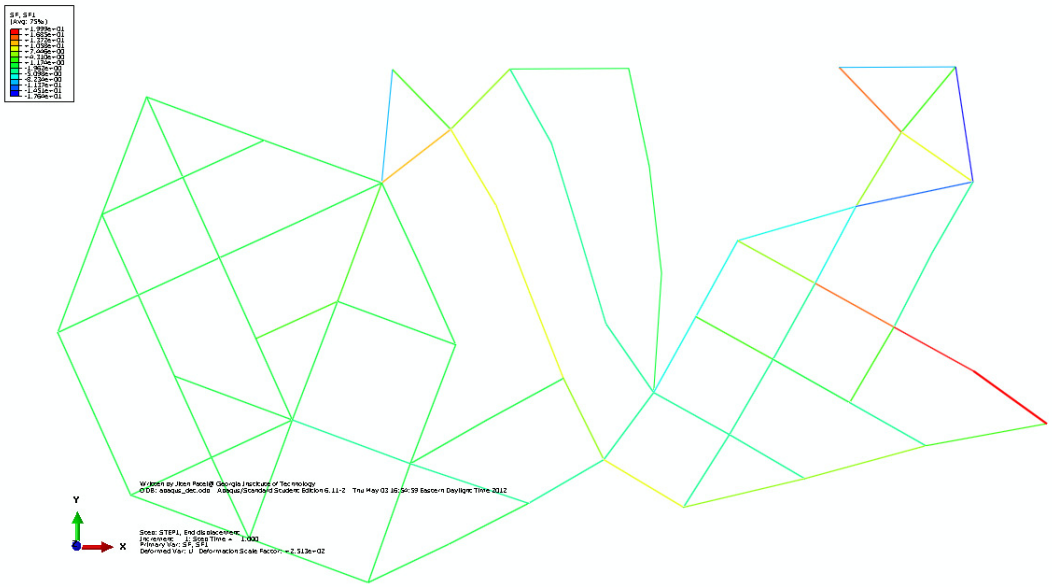


Figure 5. 16 Contours for the axial forces on deformed plot of gripper mechanism

Next, we will study the mechanism that resulted in the Reliability-based Topology Optimization procedure where variability in the input force was considered as the only source of potential uncertainty.

Reliability-based Topology Optimization Problem Formulation:

Minimize: $$\frac{MSE}{SE} = \frac{V^T KU}{0.5U^T KU} \tag{5.30}$$

Subject to: $$\sum_{i=1}^{NELEM} t\, wid_i\, L_i - V^* \le 0 \tag{5.31}$$

$$P_j[g_j(b, \underline{x}) < 0] \le 0.01 \tag{5.32}$$

$$10^{-4} \le wid_i \le 4 \tag{5.33}$$

$$KU = F \tag{5.34}$$

$$KV = F_d \tag{5.35}$$

The limit state function in this case is expressed as buckling failure criteria where the structure will be marked as failed if the stress due to compression in any truss member exceeds the Euler buckling stress. Euler buckling stress for members with cross-sectional areas *A* can be expressed as:

$$s_{icr} = -\frac{\pi E A_i}{4L_i^2} \tag{5.36}$$

The limit state function can then be expressed as

$$g(P_i) = s_{icr} - \frac{P_i}{A_i} \tag{5.37}$$

where P_i represents the axial force in each truss member. The axial forces are calculated using the FEA during every iteration of the optimization algorithm. The limit state function is only considered for the members in compression since buckling is a compression stress phenomenon. The limit state function in Eq. (5.37) gives a negative value in case any of the frame elements in the structure buckles. Note that the idea behind including a buckling constraint as a limit state function is just to illustrate the efficacy of the method when we include a stress constraint as a limit state function. For the reliability estimation case, a force, with a mean value of 3000 Newtons was assumed to act on the input point. The coefficient of variation was assumed to be 0.4. Recall that as was explained earlier, local stress constraints in Deterministic Topology Optimization (DTO) result in disjoint topology optimization problems, which are still a challenge to solve.

The gripper mechanism that was obtained as a result of solving the Reliability-based Topology Optimization is represented in Figure 5. 17. The darkest and thickest frame elements in this figure have a width of 4 mm whereas the thinnest and lightest have widths between 2 mm and 0.5 mm. The intermediate elements come from the ranges of 4 mm – 3 mm and 3 mm – 2 mm for intermediate darkness and thickness of the constituting elements.

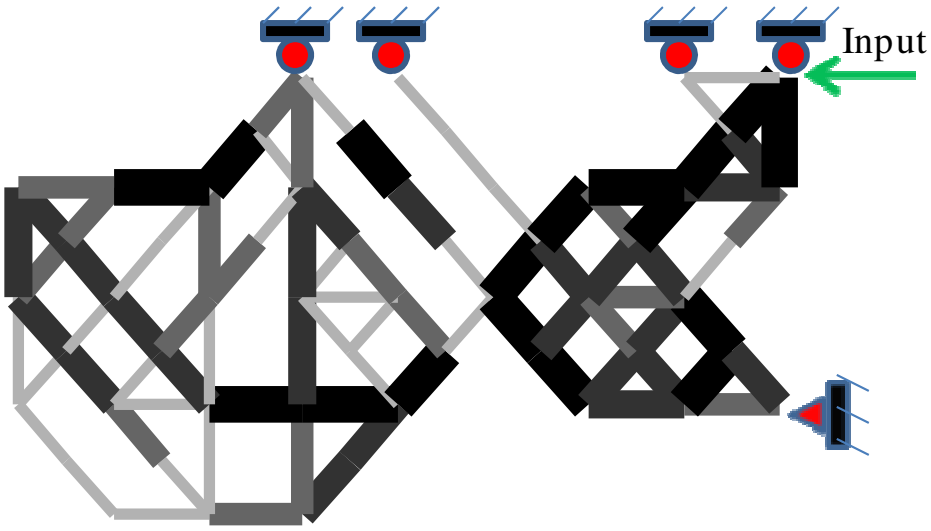


Figure 5. 17 Reduced groundtruss solution for gripper mechanism for RBTO case

The wireframe model of the obtained solution from Figure 5. 17 is shown in Figure 5. 18 where it can be seen that this structure consists of 54 nodes and 88 frame elements in contrast to the DTO solution which had 56 nodes and 88 frame elements. Figure 5. 19 shows the displacement contours on the superimposed un-deformed and deformed plots of this mechanism. Note that the 38^{th} Node, which is the output node in this case, has a X-displacement of 4.33×10^{-3} and a Y-displacement of 2.45×10^{-2}. Note the increase in displacement at the output node in this case from the DTO case where the X-displacement and Y-displacement were 1.0944×10^{-2} and 1.93×10^{-2} respectively. Since we are only concerned with the motion along the positive Y-axis, the RBTO solution shows better results. We also confirmed that the results obtained using the RBTO solution are close to the linear FEM solutions where geometric nonlinearity is considered.

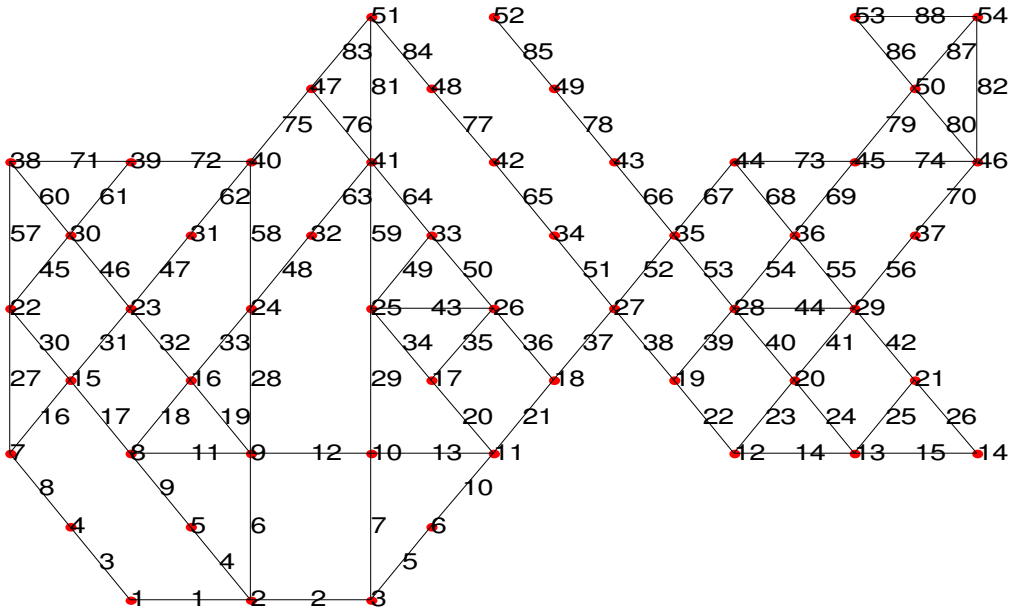


Figure 5. 18 Wireframe solution obtained from RBTO

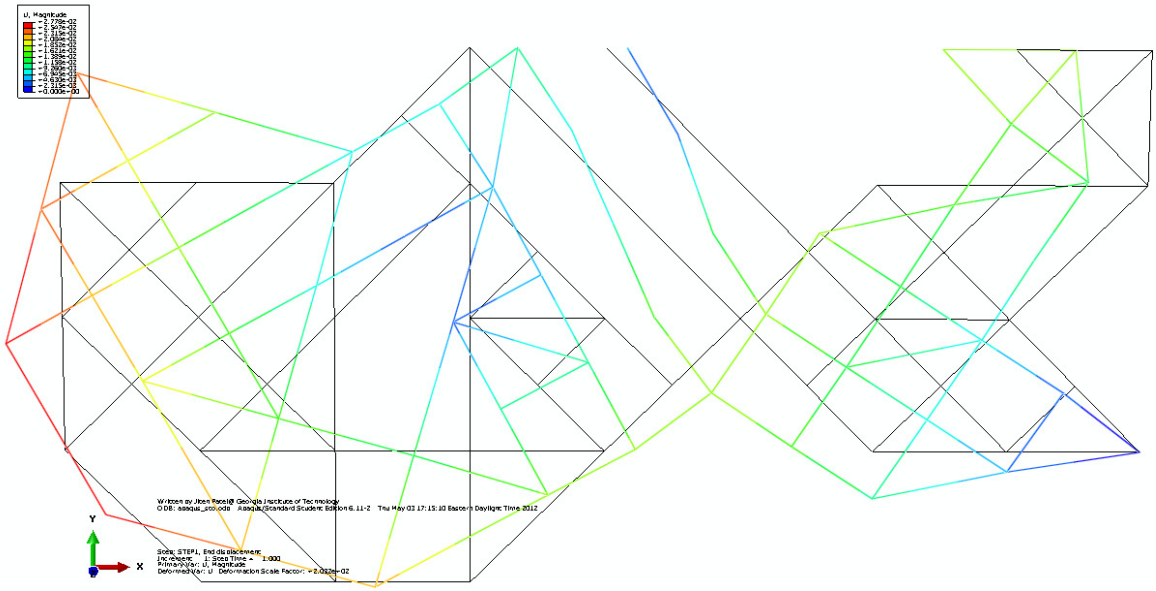


Figure 5. 19 Undeformed and deformed plot with displacement contours from ABAQUS using linear FEM for RBTO

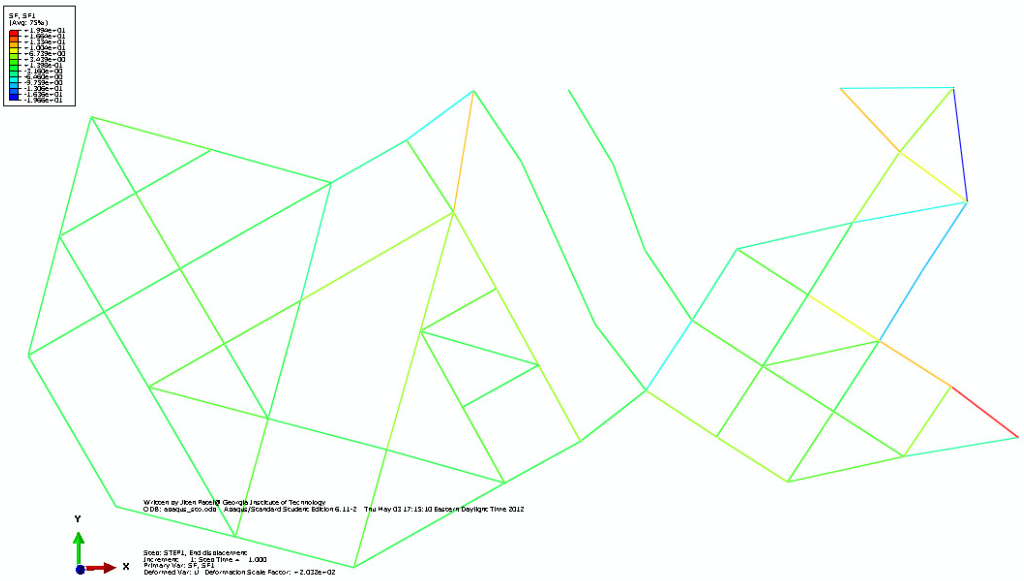


Figure 5. 20 Contours for the axial forces on deformed plot of gripper mechanism found using RBTO

Figure 5. 20 shows the axial forces on the frame elements. Note that there are only four elements indicated by red color and high stress concentration areas in contrast to five elements with high stress concentrations in the DTO case. Also, the maximum axial force that is present in any element in the RBTO solution is 19.33 Newtons in contrast to 19.99 Newtons in the DTO case. Even though the decrease in maximum axial force is not significant, it can be concluded that RBTO solution gives a better arrangement of material through the topology optimization procedure so that stresses are uniformly distributed though-out the structure.

The reliability levels of these two structures were estimated using the Euler buckling criteria limit state function by finding the response of the structures with 10,000,000 samples of MCS. The P_f value for the DTO mechanism was found to be 0.13 and the P_f value for the RBTO solution was 0.0089. SSL-2 algorithm was also used with 10 labeled points and 400 unlabeled points in both cases, which resulted in close results

to the ones obtained by using 10,000,000 samples of MCS. Hence, we can draw two conclusions from the reliability estimation process. First, RBTO process results in a structure, which has a lower probability of failure. Furthermore, for this example we also saw that the RBTO mechanism has a higher mechanical advantage. Secondly, the SSL-2 algorithm gives results that are comparable to that obtained by using MCS. This validates our hypothesis to the Secondary Research Question 2 that the proposed algorithm can be included in the conceptual design process of complex reliable engineering systems.

5.5 Summary

In this Chapter we validated our hypotheses that we proposed for the research questions identified in Chapter 1. The numerical examples in this chapter validate that SSL-1 and SSL-2 can be used for both linear as well as nonlinear problems. Furthermore, they are immune to disjoint failure domain problems, because of their underlying classification based architecture. The efficacy of the proposed algorithms for the linear, non-linear and the disjoint problems validates our hypothesis that these algorithms can be used with reliability estimation problems while ensuring enhanced accuracy when the number of available labeled data is constant. Then, we showed two different examples, the ten-bar truss problem and the gripper mechanism design problems in order to prove the efficacy of the algorithms in reliability estimation component of engineering applications.

CHAPTER 6

CONCLUSION AND FUTURE WORK

In this chapter we conclude this dissertation while summarizing the main points in this dissertation. All the contributions in this dissertation will be summarized in Section 6.1 and the limitations will be discussed in Section 6.2. In section 6.3, we suggest possible future work, which will remove the limitations and open many more doors for research in the future.

6.1 Contributions

An extended summary of the main points presented in this dissertation is given in Section 6.1.1 followed by summary of major contributions in Section 6.1.2.

6.1.1 Extended Summary

The unifying theme in this dissertation is the goal of improving the accuracy of the reliability estimation process while requiring less number of experiments or simulations (Experiments/simulations are required to obtain labeled data). In order to investigate the reduction in computational requirement of the reliability estimation process we posed the following Primary Research Question in Chapter 1.

Primary Research Question: *How can we design reliable engineering systems efficiently while minimizing the computational/experimental cost?*

The reliability estimation process is important because it enables the designer as well as the management to estimate the current safety of a system and gives them a parameter that justifies the added money and time spent on overdesigning a system. In contrast to the traditional Factor of Safety based methods where the designer doesn't know the reliability level of a structure, Reliability-based Design Optimization (RBDO) methods over-design a system intelligently by distributing material where it is most required—the high stress concentration regions in a design.

Furthermore, once the optimized design is obtained from the RBDO procedures, it is important to verify that the final design has the required safety level. This is traditionally done with a sampling based method such as Monte-Carlo Sampling, where a large number of random data (order of 10^5 or more data samples) is sampled and the response of the system is found for those random samples using an analysis tool such as FEM. The responses are then evaluated based on certain failure criteria. By counting the total number of times the system has failed, the Probability of Failure (P_f) of the structure can be estimated. The Probability of Failure (P_f) of a System is typically taken as a parameter that quantifies the safety level of a system.

Typically, a surrogate model can be used in order to approximate the behavior of the analysis tools such as FEM, which will map the uncertain design variables with the corresponding responses from FEM models. Once the model is trained the responses from any new samples can be found easily with the surrogate model. But as explained in Chapter 1, as the values of the uncertain design variables give responses which are closer to the failure region, many structures show a snap-through behavior which will lead to wrong estimations by almost all surrogate models. This snap through behavior is also

common with bistable compliant mechanisms where there is a discontinuity in the displacement and stresses of the structure. Furthermore, under particular external influences the structure can come under the influence of various material and geometrical nonlinearities where the behavior of a typical response such as displacement will have different behavior than when it was not influenced by non-linearity.

Ideally, we would want a surrogate modeling technique, which can deal with linear zones, non-linear zones and discontinuous zones. However, there can be large computational costs for training a complex surrogate model which can deal with all these requirements. If more training data is required that is close to the failure region, more test prototypes have to be tested until they incur failure, which increases the overall cost of product development. We described the cost of acquiring additional labeled data as the additional experimental/prototype testing cost. Hence, we want a surrogate modeling technique that can be trained with less data and still produce accurate estimates close to the failure region. This led us to pose the following Secondary Research Question 1:

Secondary Research Question 1: How can we accurately predict the quintessential responses obtained from engineering analysis for reliability estimation without requiring additional experimental cost?

Given these constraints, a classification based surrogate modeling technique was hypothesized to serve our purpose well since classification based methods only estimate the class label, and the nonlinearity of the response or the discontinuity of the responses doesn't affect the process of estimating the labels. PNN was chosen as the classification process of choice since it assigns labels probabilistically before using the Bayes Decision

rule to assign class labels. Probabilistic assignment of class labels also allows integration of PNN with a Semi-Supervised Learning (SSL) based architecture, where unlabeled data is added to the already available pool of labeled data so that a more accurate classifier can be obtained for a given number of labeled data. Note that SSL is applicable and more efficient for reliability estimation process since obtaining labeled data is expensive but a large amount of unlabeled data can be obtained relatively inexpensively in reliability estimation processes since the PDF functions for the uncertain design variables is assumed to be known in most cases. We proposed two SSL algorithms SSL-1 and SSL-2 where SSL-1 uses both labeled and unlabeled data in order to estimate the parameters of Gaussian PDFs that are assumed to have 'spherical' covariances. On the other hand SSL-2 relaxes this assumption of 'spherical' covariances and assumes 'full' covariances which gives the algorithm added flexibility and more accuracy than SSL-1. These two algorithms were validated and their efficacy was validated for continuous as well as discontinuous failure domain problems in Chapter 5.

Once the reliability estimation procedures were validated, we wanted to show that these methods could be integrated in the design process so that reliable complex systems could be designed. This led us to pose the following Secondary Research Question 2.

Secondary Research Question 2: How can the proposed reliability estimation procedure used for the design of an engineering system?

We demonstrated that the proposed algorithms can be used for reliability estimation of the designed 10-bar truss problem and the design of compliant mechanisms under uncertainty in Chapter 5. These methods demonstrated that the proposed methods could

be used in design of complex systems. For our specific purpose, we chose design of structures through topology optimization as an example of complex system.

6.1.2 List of Contributions and Explanation

Many important challenges have been addressed in this dissertation. The specific contributions and their respective explanations are listed below:

- **Estimation of reliability of a system, which is under the influence of discontinuous failure domain, through a classification framework:** Typically, an experienced designer might know what kind of failure modes a system would come under. However, an inexperienced designer might have less knowledge about the different failure modes that the design might come under. In either case, we would recommend usage of a classification based method for reliability estimation process since they are robust methods of reliability estimation in case of both continuous as well as discontinuous failure domains.
- **Classification using Probabilistic Neural Networks (PNN) for reliability estimation within an optimization framework viz. Reliability-based Design Optimization (RBDO):** Probabilistic Neural Networks are feed-forward Radial Basis Function (RBF) based Artificial Neural Networks (ANN). They do not have a feedback loop for training unlike their more popular peers—Backpropagation Artificial Neural Networks. Hence, they are fast and are more suited for integration inside a RBDO framework where a typical complex systems design problem would need to estimate the reliability of the system many times during

optimization[4]. The basic PNN was used during the Reliability-based Topology Optimization (RBTO) example problems shown in this dissertation.

- **Classification based Reliability Estimation and integration with a Reliability-based Topology Optimization framework:** Until now few researchers have studied the impact of including a reliability constraint to topology optimization problems [43, 72].
- **Integration of labeled and unlabeled data via an Expectation-Maximization (EM)-like Algorithm for SSL-1 Algorithm:** The SSL-1 algorithm was adopted from the EM algorithm and inspired by the work done by Nigam et. al.[102] in the field of web-based Text Classification. This framework allows for the automatic variation of the smoothing parameter or σ^2 value in PNN, which ensures that the resulting PNN has the best parameter for the given set of labeled data. This framework relieves the designer of the tedious work of trying out various values for smoothing parameter for the underlying data and the uncertainty that it produces hence reducing the overall product development time requirement.
- **Liberating the 'spherical' covariance assumption for Gaussian kernels used in PNN and using 'full' covariance kernels in PNN for SSL-2 Algorithm:** Similar to SSL-1, SSL-2 also uses labeled and unlabeled data in order to train a better PNN classifier. However, the inherent training mechanisms for SSL-1 and SSL-2 are different. SSL-1 assumes that the Gaussian Kernels that are used for

[4] The current reliability level of the system is evaluated once every iteration of the optimization algorithm. In all the design problems illustrated in this document, there were a minimum of 50 iterations during optimization.

training are symmetrical, which constrains the corresponding covariances to be 'spherical'. Furthermore, all the kernels used in SSL-1 at all the labeled points are assumed the same. These constraints are relieved in SSL-2 where the EM algorithm is used on the combination of labeled (after ignoring the labels) and the unlabeled data for estimating the best set of Gaussian kernels. We allow these kernels to assume any shape, which is the same as allowing for 'full' covariances. Our tests have shown that SSL-2 outperforms all the major classification algorithms when less number of labeled data[5] is available and plenty of unlabeled data is available—which is generally the case.

- **Topology Optimization with Stress Constraints:** It is well known by researchers in the field of topology optimization that stress constraints on individual members during topology optimization can lead to the classic '*disjoint topology optimization problem*'. This is caused because, in discrete topology optimization, when a groundtruss is used as the starting design space discretization, the optimization procedure tries to minimize the value of the design variables. In cases when cross-sectional areas are used as design variables, as the values of the cross-sectional areas are minimized, the stresses increase. Hence, if there is an upper bound on the stresses at each member, the cross-sectional areas cannot be minimized, which leads to a failure of the optimization procedure.

  In this dissertation we demonstrated two RBTO examples where an upper bound on stress was used as a failure criteria or the limit state function. Hence, we proposed the alternate formulation for topology optimization problem with stress constraint or the '*disjoint topology optimization problem*' as a RBTO problem

[5] For the examples given in this dissertation, we validated this statement for 4-50 labeled points.

with stress limit state functions. In our formulations, where we are using a classification based reliability estimation procedure, even if there is buckling or a snap behavior inside the structure, the PNN based RBTO procedure can alleviate the disjoint failure domain problems [100].

In the example where the design of a compliant gripper was demonstrated, it was also noted that the maximum stresses in the compliant mechanism that was designed using the probabilistic framework was lower than the compliant mechanism that was designed using a deterministic optimization framework. This also validates that a probabilistic or reliability based method of structural design will lead to less failure over the system's lifetime.

In the next section, we will highlight the limitations of current work and in Section 6.3 we will suggest steps that can help alleviate these limitations.

6.2 Limitations

The limitations of the work presented in this dissertation are pointed out below with explanation.

- **Types of uncertainty considered:** In this dissertation, we considered only *aleatory* uncertainty, where it is assumed that a designer knows the probability density functions (PDFs) of the uncertain parameters beforehand.
- **Levels of uncertainty considered:** In all the examples considered in this dissertation, it was assumed that the designer wants a Probability of Failure (P_f) value in the range of 10^{-2}-10^{-5}. However, in most of the industries a P_f value in this range would be considered inadequate. For example, in the aerospace

industry, typical values of P_f are less than 10^{-9}. But in order to test the P_f value of a system which is close to 10^{-9}, 10^{11} Monte-Carlo samples should be used in order to validate the results. We did not have the computational resources to conduct such a validation exercise, which limited our scope to design examples where the P_f value required was relatively large.

- **Limitations of PNN:** As with every machine learning algorithms, PNNs have their limitations also. PNNs are sensitive to the initial training data and perform their best if there are equal numbers of labeled data points from each of the classes. Estimations tend to be biased towards the class, which happens to have the maximum representation in the labeled data set. SSL-1 also suffers from this same effect. However, this problem is less prominent with SSL-2.

  Our tests suggest that PNNs are most accurate in two dimensions when the number of training data points used is more than 30, where for a two-class problem there are 15 points from each class. For a problem with less than 30 points we do not recommend using PNN alone. However, as has been shown in this dissertation, SSL-2 will outperform almost all classifiers when the number of labeled data is less than 50.

- **Limitations of Semi-Supervised Learning:** We had mentioned earlier that SSL perform well when the *smoothness assumption* is valid. The smoothness assumption states that a point in the neighborhood of a point has been sampled from the same PDF that the original point was sampled from. In other words, the only information unlabeled data provides, that leads to the improvement in classifier performance, is the PDF information. Hence, if the unlabeled data are

sampled from a different distribution than what the labeled data was sampled from then the classifier performance may suffer. This condition has been noticed by many authors recently [129].

Providing unlabeled data, that are sampled from the same distribution as the labeled data, is not a problem in case of reliability estimation since it is assumed that the PDFs for random variables are known. However, when SSL is used on data that is collected from the 'field', it might be difficult to guess the correct PDF assumptions for the labeled data and the unlabeled data. In those cases, the SSL-1 as well as SSL-2 classification algorithms may result in poor classifier accuracy.

- **Limitations of Expectation Maximization (EM):** The Expectation-Maximization (EM) algorithm has been used in order to estimate the soft-labels of the unlabeled data for SSL-1 and for clustering, during Step-1, in case of SSL-2. Although it has been proved that EM increases the likelihood function with each iteration, it does not guarantee a global maximum. Furthermore, since the likelihood function does not have an upper bound there are possibilities of divergence. As the number of clusters is increased in case of EM, the required number of unlabeled data increases and the required computational cost of conducting EM increases because of more required duration for convergence. These problems could be expected to occur during SSL-2 but we found that these problems are not seen frequently even when the inbuilt MATLAB EM function is used.

Since in reliability estimation problems there is not a limitation on the number of unlabeled samples that can be availed, it is recommended that if there are n clusters being considered (when there are n labeled points), then at least $2n^2$ number of unlabeled samples be used.

In the next section, we offer some ways through which these limitations can be ameliorated and the current research presented in this dissertation can be improved. We will also suggest some areas which could be investigated in the future.

6.3 Future Work

Much research has been done in the field of surrogate modeling for uncertainty quantification by using function approximation based methods but the research in the field of surrogate modeling using classification based methods is still in its infancy. This dissertation bridges that gap so that systems that are under the influence of disjoint failure domain influences can still be modeled and their uncertainty can be quantified. We suggest the following avenues for future research so that the whole knowledge base for classification-based surrogate modeling is advanced which will enable the efficient design of complex systems.

- **Modeling of epistemic uncertainty:** As mentioned in the previous sections, we assumed that the PDF information for all the uncertain variables is available to the designer. This may not be the case in many cases. Hence, research should be conducted in order to model epistemic uncertainty and the methods should be developed so that these uncertainties can be quantified and modeled in the cases where the systems are under the influence of disjoint failure domains.

- **Integration of higher reliability levels in the design process:** Aerospace systems require probabilities of failure that are less than 10^{-9}. Logically we can guess that in order to say that the system has a probability of failure in the order of 10^{-9} we need to find the responses of the system using a FEM/CFD model 10^{11} times, at the least. This requires a large computational cost that was not available to us; as computing gets cheaper, it will be possible to validate those high reliability levels.
- **Experimentation with other classification algorithms:** We chose to use PNN because of its superior prediction power owing to its Artificial Neural Network (ANN) architecture. Furthermore, since it is a feed-forward network it doesn't require high training time unlike the Backpropagation ANNs. Another reason to choose PNN was that they classify patterns probabilistically before assigning the labels using the Bayes decision rule. This fact enables them to be integrated in SSL architecture seamlessly.

  In the case of reliability estimation problems, we can choose the training samples and make sure that there are equal numbers of points in either side of the classification boundary, but other fields of research do not allow such luxuries. In such cases PNN might not give optimum results. By inventing the novel SSL-2 algorithm we have alleviated this problem and SSL-2 gives superior results when compared to most classification algorithms under the cases when only few labeled data are available. Nevertheless, our tests show that SSL-2 would be much more accurate when it is combined with other *generative* classifiers, which do not suffer from classification bias, when the number of labeled data in each class is

not balanced. One of our recommended classification algorithms that can be used for SSL-2 would be the Naïve-Bayes Algorithm which is also a *generative* classifier and classifies the points probabilistically before assigning class labels to test data.

- **Semi-Supervised Learning:** As mentioned earlier, SSL gives superior results when the *smoothness assumption* is obeyed by the labeled and unlabeled data. For the *aleatory* uncertainty cases that we considered in this research, it is possible to sample points from the same distribution, which leads to superior classification accuracy. In cases where test data from the field are available, it is difficult to satisfy the smoothness assumption, which brings the classification accuracy of all SSL based algorithms into question. We do not have an answer for this questions right now and it will be useful to investigate the performance of SSL when the available data can't be assumed to be sampled from the same distribution. However, it might be possible to strategically sample unlabeled data from each of the regions corresponding to specific classes so that there are an equal number of unlabeled data from each class. Note that in case of reliability estimation it is possible to sample points from a region where the samples will definitely correspond to safe or failure regions. For example, if the uncertain variables are the cross-sectional areas of truss members and we sample unlabeled data from a region where all the cross-sectional areas are large, the structure will be in a safe region. Moreover, if the unlabeled data is sampled from a region where the cross-sectional areas are small, or even close to zero, the structure will invariably fail.

Hence, it is possible to sample unlabeled data, in case of reliability estimation problems, from both the classes.

- **The Expectation-Maximization (EM) Algorithm:** The EM algorithm is not immune to divergence, which might lead to non-convergence of the SSL-2 algorithm. This problem is again exacerbated for SSL-2 because more parameters of the individual 'full' covariance matrices are being estimated by EM. We have seen that this problem can be alleviated by using the '$2n^2$' rule for choosing the number of unlabeled data when n is the number of required clusters. However, this remains a rule of thumb and more efficient methods for EM and estimating the right number of unlabeled data are required. This will ensure that the SSL-1 and SSL-2 algorithms converge, every time they are engaged.
- **Active Learning:** Since PDFs are available during the reliability estimation process, it is possible to choose training samples that are very close to the classification boundary by using Active Learning, which will enable training a classifier that has higher classification accuracy than classifiers trained with just the original training data. Active learning chooses unlabeled data that would increase the accuracy of the classifier the most if the labels for those unlabeled data were known. The labels for those unlabeled data can be obtained by using the original FEM/CFD model and the classifier can be retrained to obtain the new classifier with improved accuracy.
- **Consideration of different failure mechanisms:** In this research, only stress constraints and displacement constraints were assumed as relevant failure criteria. Our idea here was to illustrate the general procedure of evaluating the reliability

of a system given any number of types of limit state functions. Future work should be focused on incorporating different failure phenomenon into the reliability-based design methods such as fatigue, creep, crack propagation etc.

- **Non-Linear Finite Element Analysis Methods:** In all the design and analysis exercises conducted in this dissertation we have assumed a linear behavior, because of which we have only used Linear-FEM. In cases when the individual truss or beam members come close to buckling, simple linear FEM modeling of the members is not accurate and geometrical nonlinearity should be considered. More investigation in this area should be conducted.

REFERENCES

1. Choi, S.-K., Fathianathan, M., and Schaefer, D., 2007. Optimization of Complex Engineered Systems under Risk and Uncertainty. *ASME International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*. Las Vegas, NV.

2. Maluf, N., 2004. *An Introduction to Microelectromechanical Systems Engineering*, Boston, MA, Artech House.

3. Sharpe, W.N., Turner, K.T., and Edwards, R.L., 1999. Tensile Testing of Polysilicon. *Experimental Mechanics*, **39**(3), pp. 162-170.

4. Neter, J., Kutner, M.H., Wasserman, W., and Nachtsheim, C.J., 1996. *Applied Linear Regression Models*. 3 ed, McGraw-Hill/Irwin. 720.

5. Martin, J. and Simpson, T., 2005. Use of Kriging Models to Approximate Deterministic Computer Models. *AIAA Journal*, **43**(4), pp. 853-863.

6. Simpson, T., Korte, J., Mauery, T., and Mistree, F., 1998. Comparison of response surface and kriging models for multidisciplinary design optimization. *7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. St. Louis, MO.

7. Gibson, L.J. and Ashby, M.F., 1997. *Cellular Solids:Structure and Properties*, Cambridge, UK, Cambridge University Press.

8. Rosen, D.W., 2007. Computer-Aided Design for Additive Manufacturing of Cellular Structures. *Computer-Aided Design & Applications*, **4**(5), pp. 585-594.

9. Hayes, A.M. and al., e., 2001. Mechanics of Linear Cellular Alloys. *International Mechanical Engineering Congress and Exposition*. New York, NY.

10. Seepersad, C.C., Dempsey, B.M., Allen, J.K., Mistree, F., and McDowell, D.L., 2004. Design of Multifunctional Honeycomb Materials. *AIAA Journal*, **42**(5), pp. 1025-1033.

11. Howell, L.L., 2001. *Compliant Mechanisms*, New York, NY, Wiley-Interscience.

12. Ananthasuresh, G.K., 2003. *Optimal Synthesis Methods for MEMS*, Norwell, MA, Kluwer Academic Publishers.

13. Howell, L.L., Midha, A., and Norton, T.W., 1996. Evaluation of Equivalent Spring Stiffness for Use in a Pseudo-Rigid-Body Model of Large-Deflection Compliant Mechanisms. *Journal of Mechanical Design*, **117**(1), pp. 156-165.

14. www.hoberman.com.

15. Patel, J. and Ananthasuresh, G.K., 2007. A Kinematic theory for radially foldable planar linkages. *International Journal of Solids and Structures*, **44**(18-19), pp. 6279-6298.

16. Hirosawa, H., Hirabayashi, H., Kobayashi, H., Murata, Y., Kii, T., Edwards, P., Natori, M., Takano, T., Yamamoto, Z.-I., Hashimoto, T., Inoue, K., Ohnishi, A., Ohshima, T., Ichikawa, T., Fujisawa, K., Wajima, K., Okayasu, R., Inoue, M., Kawaguchi, N., Kameno, S., Shibata, K., and Asaki, Y., 2002. Space VLBI satellite Halca and its engineering accomplishments. *Acta Astronautica*, **50**(5), pp. 301-309.

17. Evans, A.G., Hutchinson, J.W., Fleck, N.A., Ashby, M.F., and Wadley, H.N.G., 2001. The Topological Design of Multifunctional Cellular Materials. *Progress in Material Science*, **46**(3-4), pp. 309-327.

18. Bendsoe, M. and Kikuchi, N., 1988. Generating Optimal Topologies in Structural Design Using a Homogenization Method. *Computer Methods in Applied Mechanics and Engineering*, **71**, pp. 197-224.

19. Bendsoe, M. and Sigmund, O., 2003. *Topology Optimization: Theory, Methods and Applications*, Springer.

20. Frecker, M., Ananthasuresh, G.K., Nishiwaki, S., Kikuchi, N., and Kota, S., 1997. Topological Synthesis of Compliant Mechanisms Using Multi-Criteria Optimization. *Journal of Mechanical Design*, **119**(2), pp. 238-245.

21. Saxena, A. and Ananthasuresh, G.K., 2000. On an Optimal Property of Compliant Topologies. *Structural and Multidisciplinary Optimization*, **19**(1), pp. 36-49.

22. Eschenauer, H.A. and Olhoff, N., 2001. Topology Optimization of Continuum Structures: A Review. *Applied Mechanics Review*, **54**(44), pp. 331-389.

23. Bendsoe, M.P., Diaz, A., and Kikuchi, N., 1993, *Topology and generalized layout optimization of elastic structures*, in *Topology Design of Structures*, Kluwer Academic Publishers, Dordrecht, 159-205.

24. Yang, R.J. and Chuang, C.H., 1994. Optimal topology design using linear programming. *Computers and Structures*, **52**, pp. 265-275.

25. Kohn, R.V. and Strang, G., 1986. Optimal Design and Relaxation of Variational Problems. *Communications on Pure and Applied Mathematics*.

26. Du, L., Choi, K.K., and Youn, B.D., 2004. A new fuzzy analysis method for possibility-based design optimization. *10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*. Albany, NY.

27. Dempster, A.P., 1967. Upper and lower probabilities induced by a multi-valued mapping. *The annals of mathematical statistics*, **38**, pp. 325-339.

28. Shafer, G., 1976. *A Mathematical Theory of Evidence*, Princeton, Princeton University Press.

29. Kendall, D.G., 1974. *Foundations of a theory of random sets, in: Harding, E. and Kendall, D. (eds), Stochastic Geometry*, New York.

30. Berleant, D., 1993. Automatically verified reasoning with both intervals and probability density functions. *Interval Computations*, **2**, pp. 48-70.

31. Ferson, S. and Ginzburg, L.R., 1996. Different methods are needed to propagate ignorance and variability. *Reliability engineering and system safety*, **54**, pp. 133-144.

32. Ferson, S., Kreinovich, V., Ginzburg, L., Myers, D.S., and Sentz, K., *Constructing probability boxes and Dempster-Shafer structures*, in *Technical Report*. 2003, Sandia National Laboratories.

33. Walley, P., 1991. *Statistical Reasoning with Imprecise Probabilities*, London, Chapman and Hall.

34. Ben-Haim, Y. and Elishakoff, I., 1990. *Convex modelsof uncertainty in applied mechanics*. Elsevier Science, Amsterdam.

35. Moore, R.E., 1966. *Interval Analysis*, Englewood Cliffs, Prentice Hall.

36. Zadeh, L.A., 1965. Fuzzy sets. *Information and Control*, **8**, pp. 338-353.

37. Koyluoglu, U., Cakmak, S., Ahmet, N., and Soren, R.K., 1995. Interval algebra to deal with pattern loading and structural uncertainty. *Journal of Engineering Mechanics*, **11**, pp. 1149-1157.

38. Muhanna, R.L. and Mullen, R.L., 1995. Development of Interval Based Methods for Fuzziness in Continuum Mechanics *ISUMA-NAFIPS*.

39. Muhanna, R.L. and Mullen, R.L., 1999. Formulation of Fuzzy Finite Element Methods for Mechanics Problems. *Computer-Aided Civil and Infrastructure Engineering*, **14**, pp. 107-117.

40. Muhanna, R.L. and Mullen, R.L., 2001. Uncertainty in Mechanics Problems-Interval-Based Approach. *Journal of Engineering Mechanics*, **127**(6), pp. 557-566.

41. Hastie, T., Tibshirani, R., and Friedman, J., 2009. *The Elements of Statistical Learning: Data Mning, Inference and Prediction*. 2nd ed. Springer Series in Statistics, Springer- Verlag.

42. Choi, S., Grandhi, R.V., and Canfield, R.A., 2006. *Reliability- Based Design Optimization*, London, Springer.

43. Kharmanda, G., Olhoff, N., Mohamad, A., and Lemaire, M., 2004. Reliability-Based Topology Optimization. *Structural and Multidisciplinary Optimization*, **26**(5), pp. 295-307.

44. Michell, A., 1904. The Limits of Economy of Material in Frame Structures. *Philosophical Magazine*, **8**, pp. 589-597.

45. Tu, J. and Choi, K.K., *A Performance Measure Approach in Reliability- based Structural Optimization*, in *Technical Report R97-02*. 1997, University of Iowa.

46. Choi, S., Grandhi, R.V., Canfield, R.A., and Pettit, C.L., 2004. Polynomial Chaos Expansion with Latin Hypercube Sampling for Estimating Response Variability. *AIAA Journal*, **42**(6), pp. 1191-1198.

47. Choi, S., 2008. Reliability Assessment via Stochastic Expansion with Local Regression Method. *49th AIAA/ASME/ASCE/AHS/ASC SDM and 10th AIAA NDA Conference*. Schaumburg, IL.

48. Fortes, M.A. and Ashby, M.F., 1999. The effect of non-uniformity of the in-plane modulus of honeycombs. *Acta Materialia*, **47**, pp. 3469-3473.

49. Wang, A.J. and McDowell, D.L., 2003. Optimization of a metal honeycomb sandwich beam-bar subjected to torsion and bending. *International Journal of Solids and Structures*, **40**(9), pp. 2085-2099.

50. Wallach, J. and Gibson, L., 2001. Mechanical behaviour of a three-dimensional truss material. *International Journal of Solids and Structures*, **38**, pp. 7181-7196.

51. Chiras, S., Mumm, D.R., Evans, A.G., Wicks, N., Hutchinson, J.W., Dharmasena, K., Wadley, H.N.G., and Fichter, S., 2002. The strucutural performance of near-optimized truss core panels. *International Journal of Solids and Structures*, **39**(15), pp. 4093-4115.

52. Johnston, S.R., Reed, M., Wang, H., and Rosen, D.W., 2006. Analysis of mesostructure unit cells comprised of octet-truss structures. *Solid Freeform Fabrication Symposium*. Austin, Tx.

53. Wang, H., Chen, Y., and Rosen, D.W., 2005. A hybrid geometric modeling method for large scale conformal cellular structures. *ASME Computers and Information in Engineering Conference*. Long Beach, CA.

54. Wang, H. and Rosen, D.W., 2006. An automated design synthesis method for compliant mechanisms with application to morphing wings. *ASME IDETC*. Philadelphia, PA.

55. Smith, J., Hodgins, J., Oppenheim, I., and Witkin, A., 2002. Creating models of truss structures with optimization. *29th Annual conference on compter graphics and interactive techniques*. San Antonio, Tx.

56. Rozvany, G.I.N., 2011. A review of new fundamental principles in exact topology optimization. *Computer Methods in Mechanics*. Warshaw, poland.

57. Cox, H.L., 1965. *The design of structures of least weight*, Pergamon, Oxford.

58. Hegemier, G.A. and Prager, W., 1969. On Michell trusses. *International Journal of Mechanical Sciences*, **11**, pp. 209-215.

59. Svanberg, K., 1987. The method of moving asymptotes-A new method for structural optimization. *International Journal for Numerical Methods in Engineering*, **24**, pp. 359-373.

60. Svanberg, K., 2002. A class of globally convergent optimization methods based on conservative convex separable approximations. *SIAM Journal on Optimization*, **12**, pp. 555-573.

61. Fleury, C., 1989. CONLIN: An efficient dual optimizer based on convex approximation concepts. *Structural and Multidisciplinary Optimization*, **1**(2), pp. 81-89.

62. Goldberg, D.E., 1989. *Genetic Algorithms in Search Optimization and Machine Learning*, Boston, MA, Kluwer Academic Publishers.

63. Fourie, P.C. and Groenwold, A.A., 2002. The particle swarm optimization algorithm in size and shape optimization. *Structural and Multidisciplinary Optimization*, **23**, pp. 259-267.

64. Hajela, P. and Lee, E., 1995. Genetic algorithms in truss topological optimization. *International Journal of Solids and Structures*, **32**(22), pp. 3341-3357.

65. Chu, J., Graf, G., and Rosen, D.W., 2008. Design for Additive Manufacturing of Cellular Structures. *Computer-Aided Design & Applications*, **5**(5), pp. 686-696.

66. Chang, P., *An improved size, matching, and scaling synthesis method for the design of meso-scale truss structures*, in *George W. Woodruff School of Mechanical Engineering*. 2011, Georgia Institute of Technology: Atlanta.

67. Dorn, W., Gomory, R., and Greenberg, H., 1964. Automatic design of optimal structures. *Journal de Mecanique*, **3**, pp. 25-52.

68. Zowe, J., Kocvara, M., and Bendsoe, M.P., 1997. Free material optimization via mathematical programming. *Mathematical Programming*, **79**(1-3), pp. 445-466.

69. Tsui, K.L., 1992. An overview of Taguchi method and newly developed statistical methods for robust design. *IIE Transactions*, **24**(5), pp. 44-57.

70. Venter, G. and Haftka, R.T., 1999. Using response surface approximations in fuzzy set based design optimization. *Structural and Multidisciplinary Optimization*, **18**(4), pp. 218-227.

71. Basudhar, A., Missoum, S., and Sanchez, A.H., 2008. Limit state function identification using Support Vector Machines for discontinous responses and disjoint failure domains. *Probabilistic Engineering Mechanics*, **23**, pp. 1-11.

72. Maute, K. and Frangopol, D.M., 2003. Reliability-based design of MEMS mechanisms by topology optimization *Computers and Structures*, **81**((8-11)), pp. 813-824.

73. Wang, M.Y., Chen, S.K., Wang, X., and Mei, Y., 2005. Design of milti-material compliant mechanisms using level set method. *Journal of Mechanical Design*, **127**(5), pp. 941-956.

74. Conti, S., Held, H., Pach, M., Rumpf, M., and Schultz, R., 2008. Shape optimization under uncertainty- A stochastic programming perspective. *SIAM Journal on Optimization*, **19**(4), pp. 1610-1632.

75. Chen, S., Chen, W., and Lee, S., 2010. Level set based robust shape and topology optimization under random field uncertainties. *Structural and Multidisciplinary Optimization*, **41**(4), pp. 507-524.

76. Aceves, S.M. and Berry, G.D., 1998. Thermodynamics of Insulated Pressure Vessels for Vehicular Hydrogen Storage. *Journal of Energy Technology*, **120**(2), pp. 137-143.

77. Boggs, P.T. and Tolle, J.W., 2000. Sequential quadratic programming for large-scale nonlinear optimization. *Computational and Applied Mathematics*, **124**(1-2), pp. 123-137.

78. Hasofer, A.M. and Lind, N.C., 1974. Exact and Invariant Second-Moment Code Format. *Journal of Engineering Mechanics Division*, **ASCE 100**(EM1), pp. 111-121.

79. Hohenbichler, M., Gollwitzer, S., Kruse, W., and Rackwitz, R., 1987. New light on first- and second-order reliability methods. *Structural Safety*, **4**(4), pp. 267-284.

80. Shinozuka, M. and Deodatis, G., *Response variability of Stochastic Finite Element Systems*, in *Stochastic Mechanics*. 1986, Department of Civil Engineering and Engineering Mechanics, Columbia University: New York, NY.

81. Haldar, A. and Mahadevan, S., 2000. *Reliability Assessment Using Stochastic Finite Element Analysis*, New York, John Wiley & Sons.

82. Cameron, R.H. and Martin, W.T., 1947. The Orthogonal Development of Nonlinear Functionals in Series of Fourier-Hermite Functionals. *Annals of Mathematics*, **48**, pp. 385-392.

83. Sobol, I.M., 1994. *A Primer for the Monte Carlo Method*. CRC Press.

84. McKay, M.D., Beckman, R.J., and Conover, W.J., 1979. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output form a Computer Code. *Technometrics*, **21**(2), pp. 239-245.

85. Wasserman, L., 2004. *All of Statistics: A concise Course in Statistical Inference*, New York, Springer.

86. Thorburn, W.M., 1918. The myth of Occam's razor. *Mind*, **27**, pp. 345-353.

87. Jeffreys, H., 1939. *Theory of Probability*. Third ed, Oxford: Clarendon Press.

88. Good, I.J., 1968. Corroboration, explanation, evolving probability, simplicity, and a sharpened razor. *British Journal of Philosophy of Science*, **19**, pp. 123-143.

89. Good, I.J., 1977. Explicativity: a mathematical theory of explanation with statistical applications. *Proceedings of the Royal Society A*, **354**, pp. 303-330.

90. Jaynes, E.T., 1979. Inference, method, and decision: Towards a Bayesian philosophy of science. *Journal of the American Statistical Association*, **74**, pp. 740-741.

91. Lancaster, P. and Salkauskas, K., 1981. Surfaces Generated by Moving Least-Squares Methods. *Mathematics of Computation*, **37**, pp. 141-158.

92. Rosenblatt, F., 1958. The Perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, **65**(6), pp. 386-408.

93. Vapnik, V., 1996. *The Nature of Statistical Learning Theory*, New York, Springer-Verlag

94. Duda, R.O., Hart, P.E., and Stork, D.G., 2001. *Pattern Classification*, New York, Wiley-Interscience Publication.

95. Bouchard, G. and Triggs, B., 2004. The trade-off between generative and discriminative classifiers. *IASC 16th International Symposium on Computational Statistics*. Prague, Czech Republic, pp. 721-728.

96. Gray, A. Computational Data Analysis: Foundations of Machine Learning and Data Mining. 2008; Available from: http://www.cc.gatech.edu/~niche/cse6740/fall08.html.

97. Orr, G., Schraudolph, N., and Cummins, F. CS-449: Neural Networks. 1999; Available from: http://www.willamette.edu/~gorr/classes/cs449/intro.html.

98. Bishop, C.M., 1996. *Neural Networks for Pattern Recognition*, Oxford University Press, USA.

99. Papadrakakis, M., Lagaros, N.D., and Tsompanakis, Y., 1997. Structural optimizatioin using evolution strategies and neural networks. *Computer methods in applied mechanics and engineering*, **156**, pp. 309-333.

100. Patel, J. and Choi, S.-K., 2012. Classification Approach for Reliability-based Topology Optimization using Probabilistic Neural Networks. *Structural and Multidisciplinary Optimization*, **45**(4), pp. 529-543.

101. Bishop, C.M., 2006. *Pattern Recognition and Machine Learning*. Information Science and Statistics, ed. M. Jordan, J. Kleinberg, and B. Schölkopf, New York, NY, Springer Science+Business Media LLC.

102. Nigam, K., McCallum, A.K., Thrun, S., and Mitchell, T., 1999. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, **39**(2-3), pp. 103-134.

103. Chtioui, Y., Bertrand, D., Devaux, M.F., and Barba, D., 1997. Comparison of multilayer perceptron and probabilistic neural networks in artificial vision. Application to the discrimination of seeds. *Journal of chemometrics*, **11**(2), pp. 111-129.

104. Wang, Y., Adali, T., Kung, S.Y., and Szabo, Z., 1998. Quantification and segmentation of brain tissues for MR images: A probabilistic neural network approach. *IEEE Transactions on Image Processing*, **7**(8), pp. 1165-1181.

105. Parzen, E., 1962. On estimation of a probability density function and mode. *Annals of Mathematical Statistics*, **33**, pp. 1065-1076.

106. Specht, D.F., 1990. Probabilistic Neural Networks. *Neural Networks*, **3**, pp. 109-118.

107. Richard, M.D. and Lippmann, R., 1991. Neural network classifiers estimate Bayesian *a posteriori* probabilities. *Neural Computation*, **3**, pp. 461-483.

108. Cacoullos, T., 1966. Estimation of a multivariate density. *Annals of Mathematical Statistics*, **18**(2), pp. 179-189.

109. Specht, D.F., 1967. Vectorcardiographic Diagnosis Using the Polynomial Discriminant Method of Pattern Recognition. *IEEE Transactions on Biomedical Engineering*, **14**(2), pp. 90-95.

110. Witten, I.H. and Frank, E., 2005. *Data Mining:Practical Machine Learning Tools and Techniques*, San Francisco, CA, Morgan Kauffman Publications.

111. Dempster, A.P., Laird, N.M., and Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of Royal Statistical Society*, **Series B 39**, pp. 1-38.

112. Patel, J. and Choi, S., 2011. Classification-based Reliability Assessment with Semi-Supervised Learning. *52nd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Material Conference and 7th AIAA Multidisciplinary Design Optimization Specialist Conference*. Denver, CO.

113. Chapelle, O., Schölkopf, B., and Zien, A., 2006. *Semi-Supervised Learning*, Cambridge, MA, MIT Press.

114. Cacoullos, T., 1966. Estimation of multivariate density. *Annals of Mathematical Statistics*, **18**(2), pp. 179-189.

115. McLachlan, G.J. and Basford, K.E., 1988. *Mixture Models:Inference and Applications to Clustering*. Statistics: Textbooks and Monographs, ed. D.B. Owen, New York, NY, Marcel Dekker Inc.

116. Hurtado, J.E. and Alvarez, D.A., 2010. An optimization method for learning statistical classifiers in structural reliability. *Probabilistic Engineering Mechanics*, **25**, pp. 26-34.

117. Ananthasuresh, G.K., *A new design paradigm in microelectromechanical systems & investigations on compliant mechanisms*, in *Mechanical Engineering*. 1994, University of Michigan: Ann Arbor.

118. Bendsoe, M.P. and Sigmund, O., 1999. Material interpolation schemes in topology optimization. *Archives of Applied Mechanics* **69**((9-10)), pp. 635-654.

119. Reddy, A.N., Maheshwari, N., Sahu, D.K., and Ananthasuresh, G.K., 2010. Miniature compliant grippers with vision-based force sensing. *IEEE Transactions on Robotics*, **26**(5), pp. 867-877.

120. Humphrey, J.D., 2003. Continuum Biomechanics of Soft Biological Tissues. *Proceedings of the Royal Society: A Mathmatical Physics and Engineering Sciences*, **459**(2029), pp. 3-46.

121. Wang, W., Liu, X., Gelinas, D., Ciruna, B., and Sun, Y., 2007. A fully automated robotic system for microinjection of zebrafish embryos. *PLoS One*, **2**(9), pp. e862.

122. Dai, J. and Sheetz, M.P., 1995. Mechanical properties of neuronal growth cone membranes studied for tether formation with laser optical tweezers. *Journal of Biology and Physics*, **68**, pp. 988-996.

123. Butler, J.P. and Kelly, S.M., 1998. A model for cytoplasmic rheology consistent with magnetic twisting cytometry. *Journal of Biorheology*, **35**(3), pp. 193-209.

124. Crick, F. and Hughes, A.F.W., 1950. The physical properties of cytoplasm: A Study by Means of the Magnetic Particle Method, Part I. Experimental. *Experimental Cell Research*, pp. 37-80.

125. Guck, J., Ananthakrishnan, R., Mahmood, H., Moon, T.J., Cunningham, C.C., and Kas, J., 2001. The optical stretcher: A novel laser tool to micromanipulate cells. *Journal of Biophysics*, **81**(2), pp. 767-784.

126. Ethier, C.R. and Simmons, C.A., 2007. *Introductory Biomechanics: from Cells and Organisms*, Cambridge, Cambridge University Press.

127. Caille, O., Thoumine, O., Tardy, Y., and Meister, J.J., 2002. Contribution of the nucleus to the mechanical properties of endothelial cells. *Journal of Biomechanics*, **35**(2), pp. 177-187.

128. Wang, X., Ananthasuresh, G.K., and Ostrowski, J.P., 2001. Vision-based sensing of forces in elastic objects. *Sensors and Actuators A: Physical*, **94**(3), pp. 142-156.

129. Cozman, F.G. and Cohen, I., 2002. Unlabeled data can degrade classification performance of generative classifiers. *Florida Artificial Intelligence Research Society Conference*. Pensacola, FL.