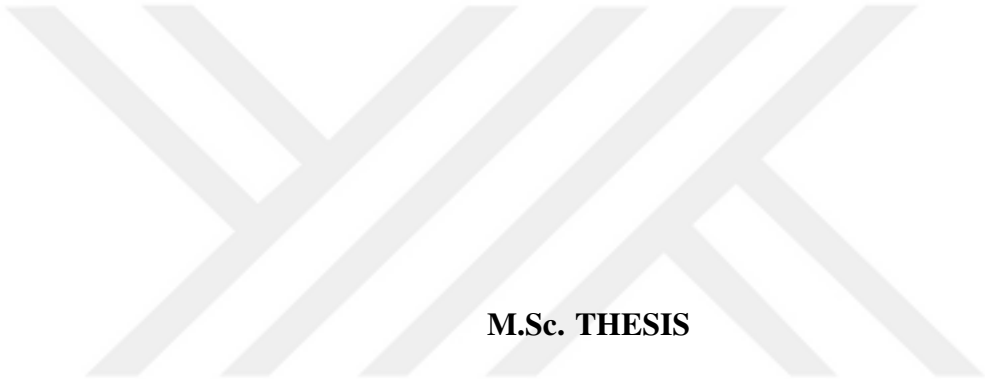


ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

**APPLICATION OF MACHINE LEARNING ALGORITHMS
IN SEISMIC FRAGILITY ANALYSIS OF STRUCTURES**



M.Sc. THESIS

Mohamed Tareq Mohamed SOLIMAN

Department of Civil Engineering

Structural Engineering Programme

MAY 2023

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

**APPLICATION OF MACHINE LEARNING ALGORITHMS
IN SEISMIC FRAGILITY ANALYSIS OF STRUCTURES**



M.Sc. THESIS

**Mohamed Tareq Mohamed SOLIMAN
(501201019)**

Department of Civil Engineering

Structural Engineering Programme

Thesis Advisor: Prof. Dr. Ufuk YAZGAN

MAY 2023

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

**YAPILARIN SİSMİK KIRILGANLIK ANALİZİNDE
MAKİNE ÖĞRENME ALGORİTMALARININ UYGULANMASI**

YÜKSEK LİSANS TEZİ

**Mohamed Tareq Mohamed SOLIMAN
(501201019)**

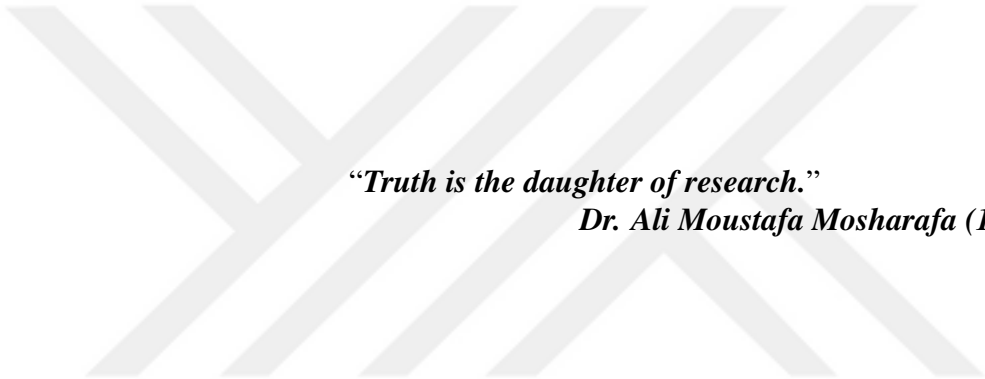
İnşaat Mühendisliği Anabilim Dalı

Yapı Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Ufuk YAZGAN

MAYIS 2023





“Truth is the daughter of research.”

Dr. Ali Moustafa Mosharafa (1898 -1950)



FOREWORD

Our study aims to explore advantages and limitations of using machine learning algorithms in developing seismic fragility models for reinforced concrete structures. Additionally, it identifies potential avenues for future research to overcome the limitations of the current approaches. The findings provided by our study can be beneficial for structural engineers in selecting retrofitting options based on resilience versus cost for structures prone to seismic events.

Words can not express my deepest gratitude to my thesis supervisor, Prof. Ufuk Yazgan, for his valuable patience and feedback. Without his guidance, this work would not have been completed.

Special thanks to my colleagues and friends Feras Abla, Haya Aldaragmeh, Bilal Aloubaidi, Zaied Alhaj, Merve Şahin, Sevde Nur Güldiken, and Ebru Toy for their support during my study.

Medrese Davutpaşa Lisansüstü Araştırma Merkezi'ni kuran Fatih Belediyesi'ne şükranlarımı sunuyorum. Bu tesis, lisansüstü öğrencilerine özel olarak ders çalışmak için rahat ve samimi bir yer sağlamıştır. Ergün Turan Bey'e, İbrahim Cihangir Bey'e ve orada tanıştığımız tüm arkadaşlara da teşekkürler.

Last but not least, I am deeply indebted to my family for their belief in me. Their financial and psychological support through thick and thin has kept my spirits high during my whole journey.

May 2023

Mohamed Tareq M. Soliman

TABLE OF CONTENTS

	Page
FOREWORD	ix
TABLE OF CONTENTS	xii
ABBREVIATIONS	xiii
SYMBOLS	xv
LIST OF TABLES	xvii
LIST OF FIGURES	xx
SUMMARY	xxi
ÖZET	xxiii
1. INTRODUCTION	1
1.1 Background	1
1.2 Literature Review	3
1.2.1 Response surface method	6
1.2.2 Artificial neural networks	8
1.3 Objective and Scope of Study	9
1.4 Overview	10
2. CASE STUDY RC FRAME BUILDING	11
2.1 Case Study Building and the Analysis Model	11
2.2 Material Properties	14
2.2.1 Concrete	14
2.2.2 Steel	16
2.3 Member Sections	17
2.4 Member Elements	18
2.5 Analysis	19
2.5.1 Static analysis	19
2.5.2 Modal analysis	19
2.5.3 Pushover analysis	21
2.5.4 Moment-Curvature analysis	22
2.6 Development of the Meta-Model	23
2.7 Ground Motion Selection	24
2.8 Intensity Measures	25
2.9 Incremental Dynamic Analysis	25
2.10 Fragility Model Formulation	26
2.11 Benchmark Model	27
2.12 Case Study II: High Variance Case Study	27
3. RESPONSE SURFACE METHOD	31
3.1 General	31
3.2 Procedure	32
3.2.1 Specifying the inputs and outputs	32
3.2.2 Determination of the levels and range of the inputs	32
3.2.3 Training the response surface	34
3.2.4 Developing the final fragility model	35
4. ARTIFICIAL NEURAL NETWORKS	37
4.1 General	37
4.2 Procedure	38
4.2.1 Determination of inputs and outputs	38
4.2.2 Adjusting the networks architecture	39
4.2.3 Training the neural networks	39
4.2.4 Developing the final fragility model	41
5. DISCUSSION OF RESULTS	43
5.1 Results of Monte Carlo Simulations	43
5.2 Results of Response Surface Method	43
5.3 Results of Artificial Neural Networks Method	54

5.4 Results of Case Study II	60
5.4.1 High variance case: Monte Carlo simulations	60
5.4.2 High variance case: RSM algorithm	60
5.4.3 High variance case: ANN algorithm	62
6. CONCLUSIONS AND RECOMMENDATIONS.....	69
6.1 Conclusions	69
6.2 Recommendations	71
APPENDICES	81
APPENDIX A : Dataset tables	83
A.1 Strong motion records list	84
A.2 Incremental Dynamic Analysis: Monte Carlo Simulation dataset	85
A.3 Incremental Dynamic Analysis: RSM training dataset	90
A.4 Incremental Dynamic Analysis: ANN training dataset	91
APPENDIX B : Source code	93
B.1 Incremental Dynamic Analysis python code	93
B.2 RSM fragility model python code	99
B.3 ANN fragility model python code	104
CURRICULUM VITAE	109



ABBREVIATIONS

ANN	: Artificial neural networks
CCM	: Central composite method
CDF	: Cumulative distribution function
EDP	: Engineering demand parameter
IDA	: Incremental dynamic analysis
IM	: Intensity measure
MDOF	: Multi degree of freedom
MLP	: Multi-layer perceptron
PGA	: Peak ground acceleration
PGV	: Peak ground velocity
PSRA	: Probabilistic seismic risk assessment
RC	: Reinforced concrete
RSM	: Response surface method
RMSE	: Root mean square error
SA	: Pseudo-spectral acceleration
SDOF	: Single degree of freedom



SYMBOLS

A_{st}	: Reinforcing bars area at top section
A_{sb}	: Reinforcing bars area at bottom section
b	: Width of reinforced concrete section
D	: Current damage level
d	: Damage criterion
E_c	: Young's modulus of concrete
E_s	: Young's modulus of steel
f_y	: Yield strength of concrete
L_{cf}	: Length of contraflexural point
L_p	: Plastic hinge length
M	: Actual Modified Mercalli intensity measure
m	: Modified Mercalli intensity measure criterion
t	: Thickness of reinforced concrete section
n	: Modular ratio of concrete
V_{ult}	: Ultimate shear strength
y_{pred}	: Predicted value of collapse likelihood
y_{act}	: Actual value of collapse likelihood
α_{ecu}	: Ultimate concrete strain metavariabale
α_{V_u}	: Ultimate shear strength metavariabale
ϵ	: Strain
ϵ_c	: Strain of concrete
ϵ_{c0}	: Strain at maximum strength of concrete
ϵ_{cu}	: Ultimate strain of concrete
ϵ_s	: Strain of steel
ζ	: Logarithmic standard deviation of cumulative density function
λ	: Logarithmic mean of cumulative density function
μ	: Ductility
σ	: Stress
σ_c	: Compressive strength of concrete
ϕ	: Diameter of reinforcing bar



LIST OF TABLES

	<u>Page</u>
Table 2.1 : Distribution parameters of the random variables that define the failure limit state.	14
Table 2.2 : Material properties used for defining the concrete model.....	15
Table 2.3 : Material properties used for modeling reinforcement behavior.....	16
Table 2.4 : Section properties: width, b , height, h , reinforcement at top, A_{st} , and reinforcement at the bottom A_{st}	17
Table 2.5 : High variance case: Parameters that define the limit state failure criteria random variables.	28
Table 3.1 : Experimental design points of the response surface method.....	34
Table 4.1 : The distribution of training, validation, and test data points.....	40
Table A.1 : The strong motion records list used in dynamic analyses	84
Table A.2 : Incremental Dynamic Analysis results for the benchmark dataset	85
Table A.3 : Incremental Dynamic Analysis results for the RSM training dataset.	90
Table A.4 : Incremental Dynamic Analysis results for the ANN training dataset .	91



LIST OF FIGURES

	<u>Page</u>
Figure 1.1 : Deaths from earthquakes since 2000	2
Figure 1.2 : General representation of fragility model	3
Figure 2.1 : Geometry and the section properties of the case study RC frame building model	12
Figure 2.2 : Case study frame model with load distributions	12
Figure 2.3 : Histograms of randomly generated log-normally distributed random variables	13
Figure 2.4 : Cyclic response of Concrete04 material model	15
Figure 2.5 : Cyclic response of reinforcement fibers simulated using Steel02 material model.....	17
Figure 2.6 : Axial force diagram of the frame under static loading	19
Figure 2.7 : Shear force diagram of the frame under static loading	20
Figure 2.8 : Bending moment diagram of the frame understatic loading	20
Figure 2.9 : Fundamental mode shape of the frame.	21
Figure 2.10 : Static pushover curve of the frame.	22
Figure 2.11 : Moment-Curvature relationship for the column section.	23
Figure 2.12 : Moment-Curvature relationship for beam section.	24
Figure 2.13 : Histograms of random log-normal distribution of dominant limit-state criteria for high variance model	28
Figure 2.14 : Box plot of random limit state variables for standard case and high variance case.....	29
Figure 3.1 : Central composite design configuration.....	33
Figure 4.1 : Neurons computational operations in Artificial neural networks approach	37
Figure 4.2 : Multilayer perceptron neural networks configuration	40
Figure 5.1 : Size of simulations set vs. estimated fragility parameters	44
Figure 5.2 : Estimated median fragility of the benchmark points	45
Figure 5.3 : Histograms of benchmark fragility model developed from Monte Carlo Simulations	46
Figure 5.4 : Benchmark final fragility model developed from Monte Carlo Simulations	47
Figure 5.5 : Estimated fragility models for the experimental design points of RSM technique	48
Figure 5.6 : 2 nd degree polynomial response surface	49
Figure 5.7 : 2 nd degree polynomial response surface: λ residuals of experimental design points	50
Figure 5.8 : Residual errors produced at different meta-variables variations for Response surface method	51

Figure 5.9 : Histograms of benchmark fragility model developed from Response Surface Method	52
Figure 5.10 : Final fragility model produced by response surface method	52
Figure 5.11 : 3 rd degree polynomial response surface	53
Figure 5.12 : Root mean square error for the response surface method versus degrees of equation.....	54
Figure 5.13 : Variation of neural networks parameters	56
Figure 5.14 : Artificial neural networks: Variations of predicted CDF parameters	57
Figure 5.15 : Violin plots of residual errors produced at each intensity measure level for the machine learning algorithms.....	58
Figure 5.16 : Histograms of benchmark fragility model developed from Artificial Neural Networks.....	59
Figure 5.17 : Final fragility model yielded by ANN.....	60
Figure 5.18 : Consumed time analysis for each number of simulations	61
Figure 5.19 : Benchmark final fragility model for the high variance case.....	61
Figure 5.20 : Benchmark final fragility model for standard case and high variance case.....	62
Figure 5.21 : High variance case: Response surfaces of fragility parameters	63
Figure 5.22 : RSM final fragility model for the high variance case	64
Figure 5.23 : RSM final fragility model for standard case and high variance case	64
Figure 5.24 : High variance case: Response surfaces of ANN algorithm for fragility parameters.....	65
Figure 5.25 : High variance case: Violin plot of residuals for the machine learning approaches	66
Figure 5.26 : High variance case: Final fragility model developed from machine learning algorithms.....	67
Figure 5.27 : ANN final fragility model for standard case and high variance case	67

APPLICATION OF MACHINE LEARNING ALGORITHMS IN SEISMIC FRAGILITY ANALYSIS OF STRUCTURES

SUMMARY

Probabilistic performance-based seismic assessment has become a popular research area in recent years, particularly the fragility analysis of structures. This type of analysis provides the probability of a structure exceeding a damage limit state when subjected to shaking with a certain ground motion intensity. However, it can be a cumbersome process, especially when using the Monte Carlo Simulation approach to account for uncertainties. To overcome this issue, machine learning algorithms have been introduced in the literature as an alternative to the computationally expensive crude Monte Carlo simulation approach. The primary goal of machine learning techniques is to estimate seismic performance with a reliable degree of accuracy while minimizing the computational effort to a manageable level.

The structure examined in our research is designed to have multiple potential failure modes, which are associated with resistance capacities that are a function of random variables. With the aim of developing the benchmark fragility model for the structure that has a random variation of the limit capacities; Monte Carlo simulation approach is utilized. The structure is subjected to a set of incrementally scaled ground motion records that are dynamically applied until one of the limit-state criteria is triggered. This set of ground motion records is selected to capture record-to-record variability. Based on the statistical evaluation of the results, fragility models are developed for each limit state. As the index parameter for comparison, the medians of the final fragility models are considered.

First machine learning technique examined in our study is known as the Response Surface Method (RSM). RSM aims to provide attributes that define the final fragility curve by fitting a polynomial equation based on a limited number of simulations that correspond to alternative combinations of random variable values. The value combinations are based on a design scheme that covers a specific zone within the domain of random variables. The second machine learning technique that is considered is known as artificial neural networks (ANN). ANN is inspired by how human brains work. In this study, these networks are trained on data that are obtained from an evenly-spaced grid of data points. This grid enables the ANN to capture the variation of fragility model parameters over a specific zone within in the domain of random variables.

The findings of the study reveal that the Response Surface Method (RSM) is a simpler strategy for developing fragility models as compared to the Monte Carlo method but it may also provide final fragility models that are less reliable. Due to the fact that the structure in our study is prone to multiple failure modes, the relationship

between the meta-variables and the fragility model parameters is highly nonlinear. This nonlinearity poses additional challenges in the use of the RSM as an alternative to the conventional Monte Carlo-based Incremental Dynamic Analysis (IDA). This is because the RSM fails to capture the nonlinear interference of the controlling limit state at the edges of the random variable domain.

On the other hand, the artificial neural network (ANN) approach is able to yield a more reliable final fragility model compared to the RSM. Moreover, the ANN approach is able to capture the highly nonlinear relationship that exists between the fragility model parameters and the random variables, which is a limitation of the RSM. One reason for this privilege is ANN does not rely on an approximate polynomial equation but instead utilizes the estimates obtained directly by training the network.

The computational challenges of the approaches are subsequently discussed. While the conventional Monte-Carlo simulation yields more reliable results, it also requires excessive computational power, making it a cumbersome option. The first machine learning algorithm, RSM, is found to be the most computationally efficient among the techniques studied. However, the accuracy of the RSM approach raises doubts about its suitability as an alternative to the conventional method. The ANN, on the other hand, provides higher accuracy estimates of the benchmark's fragility model with less computational cost than the conventional Monte Carlo approach. However, when compared to RSM, the cost of computational resources required for ANN is found to be six times more expensive.

YAPILARIN SİSMİK KIRILGANLIK ANALİZİNDE MAKİNE ÖĞRENME ALGORİTMALARININ UYGULANMASI

ÖZET

Son yıllarda, olasılıksal deprem performans değerlendirmesi yöntemleri özellikle yapıların kırılma analizi konusu yoğun şekilde çalışılan bir alana dönüşmüştür. Kırılma analizi kapsamında, bir yapının belirli bir sarsıntı seviyesinde deprem etkisine maruz kalması halinde alacağı hasarın belirli bir hasar seviyesini aşması ihtimali değerlendirilir. Bu değerlendirmede, yapıya etkiyecek deprem etkisi ve yapının kritik taşıyıcı sistem özellikleriyle ilgili belirsizliklerin doğru şekilde göz önüne alınması önem taşır. Bu kapsamda sıklıkla kullanılan yöntemlerden biri Monte Carlo simülasyonudur. Ancak deprem davranışının hesabı yoğun sayısal hesaplama gerektiren karmaşık yapılarda bu yöntem kullanıldığında bu yöntem önemli sayısal hesap zorluklarını da beraberinde getirir. Bu sorunu çözmek için mevcut literatürde çeşitli makine öğrenmesi algoritmaları, hesaplama gücü yoğun olan Monte Carlo simülasyon yaklaşımına alternatif olarak geliştirilmiştir. Makine öğrenimi tekniklerinin temel amacı, yapının deprem kırılma analizi için yeterli sayısal doğruluk seviyesinde tahmin etmek ve hesaplama zorluklarını yönetilebilir seviyelere indirmektir.

Bu tez çalışmasının amacı, kapasite tasarımı ilkelerine uygun şekilde inşa edilmemiş ve deprem dayanımı yetersiz betonarme çerçeve yapılar için deprem kırılma modelleri oluşturulmasında makine öğrenimi algoritmalarının kullanımının avantajlarını ve kusurlarını keşfetmektir. Ek olarak, mevcut yöntemlerin sınırlamalarını aşmak için gelecekte araştırılması faydalı olabilecek konular da tespit çalışma kapsamında tespit edilerek tezde sunulmuştur. Çalışmadan elde edilen temel sonuç, deprem ve yapı mühendislerinin deprem etkilerine maruz kalan yapıların sebep olduğu risklerin daha doğru tespit edilmesinde makine öğrenimi yöntemlerinin uygun ve etkili kullanımının önemli faydalar sağlayabileceğidir. Bununla birlikte, göz önüne alınan yapının özelliklerine uygun olmayan yöntemlerin kullanılması durumunda hatalı sonuçlar elde edilmesi de olasıdır.

Araştırmada incelenen yapı, birden fazla potansiyel hasar moduna sahip olan, deprem etkisi altında gevrek hasar modlarında göçme ihtimali bulunan betonarme bir çerçevedir. Yapının hasar modunu etkileyen kritik yapısal özellikleri rassal değişkenler olarak tanımlanmıştır. Referans kırılma modelinin geliştirilmesi amacıyla hasar sınır durum kriterlerinin rassal değişkenler olarak hesaba katıldığı durum için Monte Carlo simülasyon yaklaşımı kullanılarak kırılma hesabı yapılmıştır. Bu kapsamda yapı, sınır durum kriterlerinden biri tetiklenene kadar dinamik olarak uygulanan artımlı ölçeklendirilmiş bir dizi yer hareketi kaydına tabi tutulmuştur. Bu yer hareketi kayıtları, bir kayıttan diğer kayıda var olan değişkenliği yakalamak için seçilmiştir. Sonuçların istatistiksel değerlendirmesiyle her sınır durumu kombinasyonu

için kırılgenlik modeli oluřturulmuřtur. Sonuç kırılgenlik modeli ise, hasar sınırı ařılma olasılıęı medyan deęerlerinden elde edilmiřtir. Yoęun hesap gerektiren Monte Carlo yaklařımıyla elde edilen kırılgenlik modeli referans model olarak göz önüne alınmıřtır.

Bu çalıřmada ilk incelenen makine öğrenimi yöntemi Cevap Yüzeyi Yöntemi'dir (CYY). Bu yöntemin literatürdeki bir dięer adı da Tepki Yüzeyi Yöntemi'dir. CYY, belirli bir sistemin ürettięi cevabın girdi parametreleriyle ne řekilde deęiřtięini arařtırmak için oluřturulmuř analiz noktaları kümesini tanımlayan bir tasarım řemasına dayanmaktadır. Bu řema, az sayıda bir nokta kümesinde analizler yapıp bu analiz sonuçlarına polinom bir yüzey denklemi uyarlayı hedeflemektedir. Bu uyarlanmıř denklemler, nihai kırılgenlik modelini tanımlayan parametreler saęlar. Analizlerin yürütüldüęü az sayıda noktayı içeren küme, sınır durum kriterlerinin belirli bir bölgesini arařtıran bir tasarım řemasına dayalı olarak oluřturulur.

İkinci yöntem ise, insan beyninin çalıřma prensiplerinden ilham alarak geliřtirilen Yapay Sınır Ağları (YSA)'dir. Bu ağlar, eřit aralıklı bir veri noktası ızgarasından elde edilen verilerle eęitilir. Bu eřit aralıklı veriler, YSA yönteminin, CYY yöntemiyle aynı sınır durum kriterleri bölgesini kapsamasını hedefler. CYY yönteminde kullanılan polinom yüzey denklemi idealizasyonu YSA yönteminde kullanılmamaktadır. Bu nedenle, girdi parametreleri ve kırılgenlik parametreleri arasındaki iliřkideki sert deęiřkenlik YSA yönteminde daha iyi yakalanabilmektedir. Bu tez kapsamında göz önüne alınmıř yapı için bu durumun kırılgenlik analizinde önemli avantaj saęladığı tespit edilmiřtir. Bunun nedeni, yapının kırılgenlik özelliklerinin taşıyıcı sistem parametrelerine göre önemli seviyede hassasiyet göstermesi ve CYY yönteminde kullanılan basit yüzey polinom denkleminin bu iliřkiyi modellemede yetersiz kalmasıdır.

Çalıřmanın bulguları, Cevap Yüzey Yönteminin (CYY) düşük hesaplama süresi nedeniyle avantajlı olduęunu ortaya koymaktadır. Ancak, CYY, sınır durumu deęiřkenleri arasındaki doğrusal olmayan iliřkiyi yakalayamamaktadır. Örneęin, bir sınır durumunun çok düşük bir deęerinde dięer sınır durumunu artırmak, dięer sınır durumu zaten tetiklendięi için kırılgenlikte önemsiz bir deęiřikliğe neden vermeliyken, CYY bunu yansıtamamaktadır. Doğrusallıktan sapmanın yakalanamaması, CYY'nin araya giren sınır durumları bölgesini orta varyanslı çan řeklinde bir eęri yerine düz bir bölge olarak temsil etmesiyle açıkça görölmektedir. Bunun yanı sıra, medyan kapasiteye karřılık gelen řiddet ölçüsü, referans modeline göre biraz kaydırılmıřtır.

CYY yönteminin sınırlarını daha ileri seviyede test etmek için, polinom denkleminin derecesinin arttırılması yaklařımı test edilmiřtir. Bu yöntem, eęitim veri noktalarında daha yüksek doğrulukta sonuç hesaplanmasını saęlamıřtır, ancak test veri noktalarındaki hataları büyötmüřtür. Ek olarak, nihai kırılgenlik modeli önemli ölçüde iyileřtirilmemiřtir. Ayrıca, polinom denkleminin derecesini arttırmak için daha fazla analiz noktasına ihtiyaç duyulmaktadır ve bu da daha fazla hesaplama gücü gerektirmektedir.

Bu dezavantajlar, konvensiyonel Monte-Carlo Simölasyon yöntemi yerine bir alternatif olarak CYY'nin kullanımında ek zorluklar ortaya çıkarır. Bununla birlikte, bu dezavantajlara raęmen, CYY belirli baęlamlarda meta-deęiřkenler ve çıktı deęiřkenleri arasındaki iliřkileri modellemek için yararlı bir yöntem olmaya devam

etmektedir. Örneğin, tek potansiyel çökme modu olan, değişkenler arasındaki ilişki nispeten basit ve doğrusal olduğunda veya hesaplama gücü sınırlı olduğunda uygun olabilir.

Yapıların kırılma analizinde için yapay sinir ağları (YSA) yöntemin kullanılmasının avantajı, yaklaşık polinom denklemlerine dayanmamasıdır. Bunun yerine, eğitim verileriyle eşleşmesi için değişebilen ağırlıklara sahip nöronları kullanır. YSA yönteminin önemli bir başarısı, CYY'nin yakalayamadığı kırılma modeli parametreleri ile rastgele değişkenler arasında var olan üst mertebeden doğrusal olmayan ilişkiyi yakalayabilmesidir. Sonuç olarak, ortaya çıkan yüzey, plato bölgesi olarak bilinen çok düşük meta-değişken değerlerde istendiği gibi düzleşir.

Ek olarak, YSA ve CYY modelleri tarafından üretilen artıkların tüm şiddet ölçüsün seviyeleri üzerinde tartışılmaktadır. YSA yöntemi, tüm şiddet ölçümleri seviyelerinde daha az çökme olasılığı artık ürettiği bulunmuştur. Bundan dolayı, YSA yöntemi, yeni bir kombinasyondan çökme olasılığını oldukça iyi bir doğruluk düzeyiyle tahmin edebilmektedir. Diğer yandan, CYY modeli daha büyük artıklar sağlamıştır. Bunun nedeni, eğrisinin pürüzsüzlüğü nedeniyle sistemin doğrusal olmayanlıklarını yakalama yeteneğinin sınırlanmasıdır. Ayrıca, yapay sinir ağı (YSA) yöntemi, CYY'ye kıyasla daha güvenilir bir nihai kırılma modeli üretebilmektedir.

Makine öğrenimi yöntemlerinin her biri kendisine özgü hesaplama zorlukları tez kapsamında kıyaslamalı olarak tartışılmıştır. Konvansiyonel Monte-Carlo simülasyonu daha güvenilir sonuçlar sağlasa da yüksek hesaplama gücü gerektirdiğinden külfetli bir seçenektir. Çalışma kapsamında incelenmiş CYY, hesaplama açısından en verimli yöntem olarak bulunmuştur. Öte yandan, YSA, Monte-Carlo simülasyonuna göre daha az hesaplama gücüyle referans kırılma modelinin daha yüksek doğruluk tahminlerini sağlamaktadır. Ancak, YSA kullanmak, daha az gelişmiş makine öğrenimi algoritmalarından daha fazla hesaplama gücü gerektirmektedir. Bu sorunu hafifletmek için bir duyarlılık analizi yapılmıştır ve bu analiz, tutarlı bir son kırılma modeli oluşturmak için en az 49 sınır durumu kombinasyonuna ihtiyaç olduğunu ortaya koymuştur.

Bu durumda, YSA için 3 saatlik bir hesaplama süresine ihtiyaç duyulmuştur. Bu yöntem, hala Monte-Carlo simülasyonuna göre zaman tasarrufu sağlamaktadır, ancak uygulanması 32 dakika süren bir uygulama CYY yöntemiyle karşılaştırıldığında, CYY yönteminden daha külfetlidir. Bu bulgu, YSA'nın çoklu serbestlik derecesine ve karmaşık yapılar için daha da geliştirilmesinin gerekli olduğunu göstermektedir.

Makine öğrenimi algoritmalarının kırılma modellerinin geliştirilmesinde uygulanabilirliği konusundaki araştırmayı genişletmek için ek bir vaka çalışması oluşturulmuştur. Bu vaka çalışmasında, malzeme özellikleri belirsizlik seviyesi yükseltilmiştir. Bu durum sınır durumlarıyla tepki arasındaki ilişkide daha yüksek seviyede bir doğrusallıktan sapma ortaya çıkarmaktadır. YSA yöntemi, referans modelini bire bir yakalayamamış olsa da, CYY'e göre daha az kalan ve daha güvenilir bir nihai kırılma modeli sağlamaktadır. YSA, CYY yöntemine göre, daha iyi bir Monte Carlo simülasyonu alternatifi olarak değerlendirilebilir.



1. INTRODUCTION

1.1 Background

Throughout history, earthquakes have posed a persistent threat to urban areas. Some earthquakes cause shaking that is weak and barely noticeable, while others can be extremely violent and cause devastating losses of life and property. In 2023, two major earthquakes hit Turkey and Syria within a short span of time: the Kahramanmaraş and Gaziantep earthquakes, 7.8 and 7.5 magnitudes, followed by more than 7,500 aftershocks. These earthquakes, caused extensive damage to buildings, with approximately 67,000 structures between heavily damaged or completely destroyed (Hancılar et al., 2023). Figure 1.1 compares the casualties caused by the aforementioned earthquake with earlier strong motions. As a result, the country experienced a 4 percent economic loss in terms of Turkey's 2021 GDP (World Bank, 2023). Also there are several other active faults that continue to pose threats to various megacities and urban regions. According to a report by Parsons (2004), there is a 38 percent chance of a 7-magnitude earthquake occurring in Istanbul within the next 30 years.

The devastating consequences of earthquakes and their negative impact on communities have motivated researchers to develop theoretical methods for assessing seismic effects on structures. A widely used framework for this purpose is the Probabilistic Seismic Risk Assessment (PSRA), which evaluates the potential risks associated with seismic activity that the structure may face. The probabilistic seismic risk assessment framework is capable of estimating the probability that potential consequences occur within a given time frame. These outcomes can be in any form of damage to the structure, financial loss, or serious harm to individuals. PSRA provides crucial information to decision makers and stakeholders to help them prepare and mitigate the potential risks associated with seismic activity. By assessing the seismic

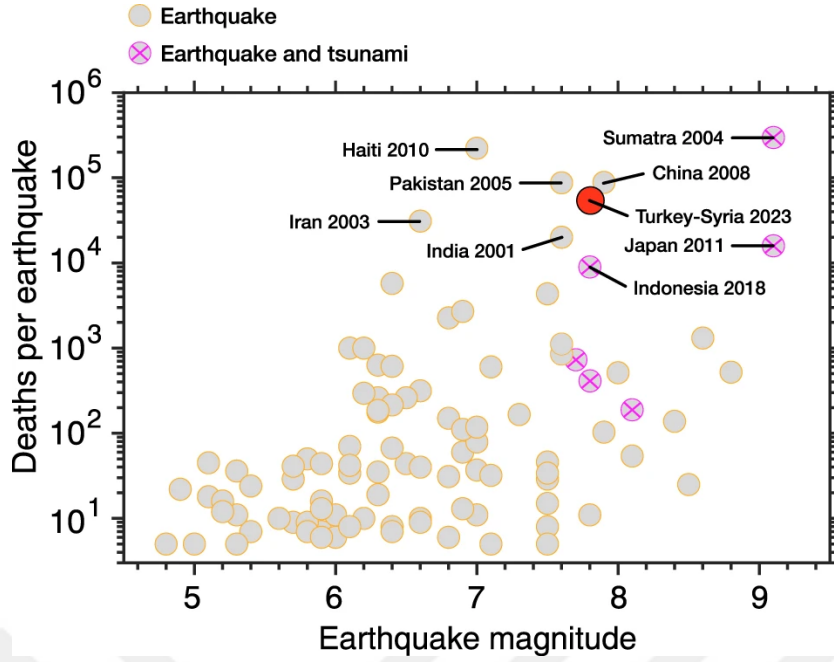


Figure 1.1 : The recent Turkey-Syria earthquakes were among the deadliest and most severe earthquakes of magnitude 7.8 or higher (Dal Zilio, 2023).

risk of structures and their surroundings, measures can be taken to reduce the potential for damage or harm and to improve the overall safety and resilience of the structure. Put differently, PSRA is an assessment framework that aims to predict the likelihood of potential damage before a seismic event occurs; therefore its a pre-event assessment. This framework can be applied to a variety of structures, including buildings, bridges, dams, and other critical infrastructure. A common approach for evaluating the seismic risk of structures is the Cornell-McGuire equation given in Equation (1.1) (McGuire, 2008). It is a probabilistic approach that estimates the likelihood that a given measure of ground motion intensity is exceeded for a specific level of structural performance.

$$P(f) = \int P[IM > IM_{ref}^* \times F_{M,R}(m,r)|M,R] \times P[D > d|\mu, IM] dm dr d\mu \quad (1.1)$$

Probabilistic Seismic Risk Assessment (PSRA) involves two constituent parts. The first part is the Seismic Hazard Model, which is used to analyze the strong-motion characteristics of a specific location. In the equation, it is presented as the probability of seismic intensity measure level (IM) exceeding a particular intensity measure (IM_{ref}^*) multiplied by a factor that is a function of both measures (m and M) and epicentre distance (r and R). This model provides information about the potential seismic

activity that may occur to the structure in the area under consideration. Various seismic hazard assessment tools, such as OpenQuake, OpenSHA, etc, are available to model the seismic hazard in a region and to compute the probabilities of various intensity measure levels being exceeded at the sites of interest.

The second constituent part of the PSRA process is the Fragility Model. This model estimates the likelihood of structural damage (D) exceeds a certain damage grade (d) in response to a given intensity measure. Given its ability to provide information on the potential damage to a specific structure given the severity of the seismic event; the fragility model is an important tool for assessing the vulnerability of structures to seismic activity and taking appropriate measures to reduce the risk of damage or harm. Our research specifically focuses on the development of the fragility model. Therefore, the hazard model is left outside the scope of the present study. A general description of the fragility model is shown in Figure 1.2.

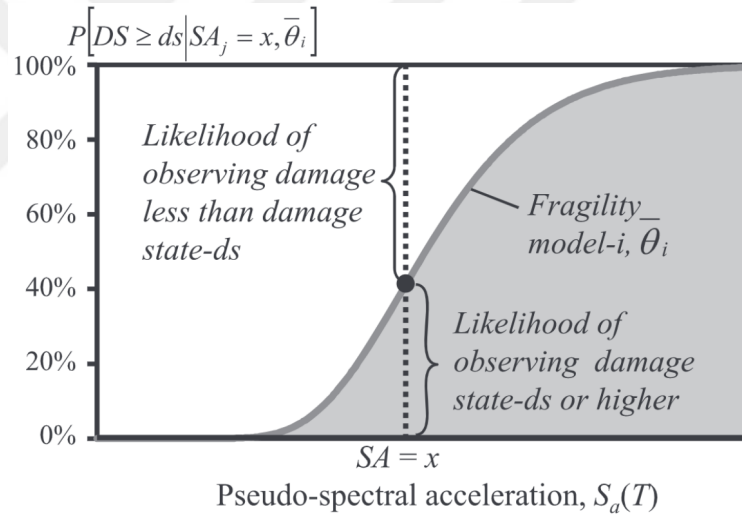


Figure 1.2 : General representation of fragility model (Yazgan, 2015).

1.2 Literature Review

Various techniques have been explored in the literature to derive fragility models, with different approaches being used to model various types of structures. There are three fundamental approaches for developing fragility models. These approaches are: (1) Expert opinion based approaches, (2) Empirical approaches and (3) Analytical approaches. The specifics of each approach are summarized by Porter et al. (2012). In this thesis study, an analytical approach is considered to develop fragility functions.

One of the well-known studies in analytical fragility model development based nonlinear static procedure is the study by Shinozuka et al. (2000) that recommends models for typical bridges in California. These fragility models were compared to those developed by nonlinear time history analysis approach. It was found that the static analysis approach conformed well with the time history analysis approach up to PGA of $0.25g$. However, higher PGA lead the static analysis approach to underestimate the fragility of the structure. In a study by Hwang et al. (2001), fragility models were developed for multiple highway bridges under various damage models, using spectral acceleration S_A as the seismic intensity measure. In another study, Liel et al. (2009) examined the impact of incorporating uncertainties related to structural modeling parameters into fragility models and found that disregarding such uncertainties led to unconservative results in almost all cases. The effect of the number of dynamic analyses on the reliability of fragility models obtained by linear regression and maximum likelihood estimates was studied by Gehl et al. (2015). In addition, Yazgan (2015) proposed a framework for establishing fragility models that suits any damage state of interest, as long as post-earthquake damage observations are given for that particular state. The author developed analytical fragility models for 4 storey RC buildings with data extracted from 516 residential buildings affected by 1999 Düzce and 2003 Bingöl earthquakes. Later, Silva et al. (2016) continued in the research domain of fragility analysis by investigating the fragility models collected as a part of the European project Syner-G and proposing new design maps for Europe. Furthermore, Akkar (2021a) developed fragility models for residential buildings with moment-resisting mid-rise RC frames in Turkey. These models are based on the building's damage class. To develop the fragility models, the author used the Modified Mercalli Intensity scale as a function since some of the fragility models were based on post-earthquake on-site damage observations.

There are several reasons why it is important to derive fragility functions for structures (Di-Sarno and Elnashai, 2021), including:

- a) The ability to quickly determine the expected seismic integrity of a structure for each failure mode and limit state subjected to a given level of shaking.

- b) providing governments with prioritized-based retrofitting and recovery plans (Elnashai and Di Sarno, 2015).
- c) the formulation of mitigation and emergency plans (Elnashai and Di Sarno, 2015). Seismic fragility models for different mitigation strategies were proposed for bridge piers by Salkhordeh et al. (2021). The various EDPs used to derive fragility models offer designers a broader range of options when selecting a suitable pier system that meets the desired performance level.
- d) develop vulnerability functions that help insurance and reinsurance companies to develop insurance schemes (Pitilakis et al., 2014). In a study conducted by Akkar (2021b), a method was proposed to derive fragility functions that take into account the damage to buildings caused by earthquakes. These fragility models can be used in loss and risk modeling to compute relevant metrics for insurance and reinsurance purposes. Furthermore, the proposed analytical method incorporates uncertainties associated with ground motions and the various probabilities of content damage under different types of damage. This method can be used systematically to develop vulnerability and risk models that are specific to each country.
- e) the improvement of guidelines using a life-cycle cost assessment framework (Calvi et al., 2006). A study done by Xu et al. (2021) outlines a method for performing life-cycle seismic fragility analysis on reinforced concrete (RC) bridges that have aged and may be impacted by corrosion. This was achieved by creating a numerical model of the column that can capture the effects of corrosion on the shear capacity and flexure-shear behaviors of the bridge columns. This developed model can efficiently predict the seismic response of corroded bridge columns.
- f) provides a robust tool to meet the increasing demands of seismic-prone mega-cities (Di-Sarno and Elnashai, 2021).

While random sampling is considered relatively simple and favored by practitioners (Monteiro, 2016), the LHS method offers improved convergence rates and reduces the number of required analyses by optimizing the exploration of the entire limit state parameter space (Zentner et al., 2017). However, when simulating independent

variables with multiple dominant variables in the output, the random sampling method provides a more suitable approach (Huntington and Lyrantzis, 1998).

Incorporating uncertainties into fragility models requires the generation of samples, and two widely used methods for this purpose are: (a) random sampling method and (b) Latin Hypercube Sampling (LHS) method. While random sampling is considered relatively simple and favored by practitioners (Monteiro, 2016), the LHS method offers improved convergence rates and reduces the number of required analyses by optimizing the exploration of the entire limit state parameter space (Zentner et al., 2017). However, when simulating independent variables with multiple dominant variables in the output, the random sampling method provides a more suitable approach (Huntington and Lyrantzis, 1998). Moreover, employing random sampling schemes can be a "major asset" in the seismic assessment of any type of RC structure when a large sampling size is required; this is the case when record-to-record uncertainty is required to fully consider record-to-record uncertainty in the seismic assessment (Monteiro, 2016). Based on these considerations, the random sampling method is used to develop the benchmark model for our case study.

The use of machine learning techniques in seismic fragility analysis has been gaining popularity over the last two decades. A review of the available methods are provided by Xie et al. (2020). In this investigation, two of such methods are considered: (1) Response Surface Method, and (2) Artificial Neural Networks Method. The first one is relatively more established in the earthquake engineering literature while the second one is a relatively novel strategy explored in this thesis study. The utilization of both Response Surface Method and Artificial Neural Networks provides a contrast comparison between basic and advanced machine learning tools. In addition, it provides insights into machine learning's applicability in tackling nonlinear numerical problems.

1.2.1 Response surface method

Response surface method (RSM) is a machine learning algorithm that attempts to derive a polynomial relationship between the input variables and the output response. It was developed by Box and Wilson (1951) with the purpose of designing experiments

for identifying the optimal response. It involves fitting a polynomial to the data that best approximates the relationship between the input and output variables. In the literature, RSM has been frequently suggested as the preferred method for constructing fragility models. The selection of RSM as a method is primarily driven by its simplicity as a basic machine learning technique, along with its ability to present results in a clear and transparent manner. Additionally, RSM offers a polynomial equation that facilitates the determination of the dominant limit states, as their significance factors can be interpreted by examining the coefficients assigned to them. Moreover, in seismic risk analysis, RSM helps reduce the computational time and resources required to derive fragility models Liel et al. (2009).

In one study, Iervolino et al. (2004) simulated response surfaces for median and standard deviation parameters of fragility models for a range of structural components of a steel tank. The study found that the proposed method is able to assess the seismic vulnerability of structures belonging to the same structural type accurately. Similarly, Lupoi et al. (2006) utilized a response surface method to derive fragility functions for an RC frame structure. The probabilistic approach yielded results that realistically captured the impact of input ground-motion uncertainty and mechanical parameter variability on the structure's fragility. Again using RSM, the aim of the research conducted by Segura et al. (2019) was to create Polynomial Response Surface (PRS) meta-models of different orders for assessing the seismic fragility of gravity dams. The research demonstrated that machine learning methods are crucial for evaluating the vulnerability of structures that involve computationally expensive FEM. In another study, Xiao et al. (2020) suggested an enhanced RSM to evaluate the seismic fragility analysis of base-isolated structures. The enhanced RSM directly takes into account the correlations between the random variables that are considered in the analysis. This approach proved to be a more computationally affordable alternative to the Monte Carlo simulation method with significant performance in both efficiency and accuracy. This was achieved by taking into account the correlation between the most significant input variables and the response outputs and by considering the component correlation in base-isolated structures.

1.2.2 Artificial neural networks

On the other side, Artificial Neural Networks (ANN) method represents the cutting-edge side of the machine learning toolkit. While there are more advanced tools available (such as boosting algorithms), ANN is considered less cumbersome in comparison. Artificial neural networks use a web of interconnected neurons to learn and generalize complex relationships between input and output data. These networks are trained using a set of input-output data combinations and are able to produce output responses that are close to the original model with fewer training data. Neural networks offer three significant benefits. First, they can capture nonlinear relationships between variables, which makes them particularly helpful when working with large, complex datasets. Second, they do not rely on any presumptions about how the data is to be distributed. Third, they are highly adaptable in the presence of missing or noisy data, making them a reliable alternative for such issues (Vellido et al., 1999).

The Probabilistic Seismic Risk Analysis (PSRA) field is not new to the Artificial Neural Networks (ANN) algorithm. In fact, neural network models have been extensively explored in the literature. For instance, Neural networks have been developed to forecast some characteristics of anticipated ground motions. Mahmoudi et al. (2016) examined putting Multi-Layer Perceptron (MLP) neural network to use in predicting earthquake magnitude through identifying a pattern from previous records. Their proposed strategy was thorough and did not require any prior assumption. They discovered that MLP network models had an average accuracy of about 72%, demonstrating the effectiveness of MLP network as a thorough tool that does not rely on any prior assumptions to estimate an earthquake's magnitude in a specific region at a predetermined time period.

In addition, ANNs were also employed in a post-earthquake assessment of structures by detecting damage to buildings. Zang and Imregun (2001) used frequency response function from sets of healthy and damaged specimens as training data for neural networks. According to the findings of their study, ANN performs best in damage detection when the input data is reduced using principal component analysis. While,

Vazirizade et al. (2017) developed ANN models that quantify and locate damage in the columns of a 3-story steel frame.

Another field that ANN does a great job in is yielding a probabilistic seismic fragility model. In other words, ANN can be utilized to predict the likelihood of collapse for a specific structure by training the networks on data obtained from Incremental Dynamic Analysis (IDA) method. Giovanis et al. (2016) studied the usage of neural networks in the reduction of the computing effort of developing a reliable fragility model for a nine-story steel frame with six random variables that govern the backbone behavior of the beam connections. The findings of his research showed that the fragility model developed by neural networks proved to be reliable by exhibiting a level of consistency for the whole range of limit states. While Wang et al. (2017) compared the results of the fragility model developed through random sampling with one developed by training an adaptive ANN with a nonlinear Takeda oscillator, the findings show that the adaptive ANN fragility model showed a high level of accuracy. Furthermore, Jia and Wu (2021) developed a fragility model for an RC frame-shear wall structure through ensemble ANN, which considers both record-to-record variability and structural variability without assuming the distribution type of the structural variability. Meanwhile, Jia and Wu (2022a) utilized neural networks to derive vulnerability curves for RC frame-shear wall structures, while taking the uncertainties of the structural model and seismic excitation into account.

1.3 Objective and Scope of Study

The main objective of this study is to explore the benefits and drawbacks of using machine learning algorithms in deriving reliable fragility models for structures that are prone to multiple failure modes with affordable computational resources. The study also aims to integrate two types of uncertainty that significantly affect the structural response during earthquakes. The first type of uncertainty is known as aleatoric uncertainty, which originates from the inherent randomness in the strong motion characteristics of the earthquake. This type of uncertainty is challenging to control and predict because of its stochastic nature. The second type of uncertainty is epistemic uncertainty, which arises from the lack of knowledge about the material properties

of the structural elements and their capacity to withstand earthquake-induced forces. Combination multiple failure limit states along with incorporating both types of uncertainties create a complex scenario that requires advanced techniques to estimate the structural response accurately.

The study investigates two machine learning tools that are capable of developing fragility models in the most efficient and less computationally expensive manner: the Response Surface Method (RSM) and Artificial Neural Networks (ANN).

1.4 Overview

This thesis comprises six chapters. The first chapter introduces a literature review of significant studies that have explored probabilistic seismic assessment of structural systems, including the application of machine learning algorithms such as Response Surface Methodology and Artificial Neural Networks, followed by the study's scope and objectives. The second chapter provides details on the case study building and numerical benchmark model that produces the reference fragility model. In the third chapter, the response surface method is introduced as a machine learning algorithm alternative to the Monte Carlo simulation method. In the fourth chapter, the Artificial Neural Networks approach is presented as the second machine learning algorithm. The fifth chapter reports the study's results, with a comprehensive discussion of all the findings. Finally, the concluding chapter summarizes the conclusions and recommendations based on the discussion of the results. the thesis provides a reference list and appendices at the end.

2. CASE STUDY RC FRAME BUILDING

2.1 Case Study Building and the Analysis Model

To explore the advantages and limitations of machine learning algorithms, an RC frame with multiple failure mechanism types is investigated in this thesis. The presence of multiple failure mechanism types is a common issue in seismically deficient buildings. In RC buildings that are properly designed and constructed according to capacity design principles, critical structural members are designed to fail in ductile failure mechanisms such as flexural yielding. Accordingly, each member is detailed to have sufficient shear and axial resistance capacities to ensure that flexural failure will be the expected failure mechanism. In the case of seismically deficient substandard buildings, this is not the case. As a result, the structural members are often prone to brittle failure mechanisms (e.g. shear or axial failures). Such a multitude of potential failure mechanisms results in additional complexity in fragility analysis. Such a seismically deficient building model is considered the case study building in this investigation. The structure is specifically selected to exhibit two failure modes: (1) flexural and (2) shear failure.

The case study structure consists of a 2D single-story, single-bay frame, with one column relatively shorter than the other one. The structural configuration of the frame is illustrated in Figure 2.1 and the load distributions are shown in Figure 2.2. This frame was originally investigated by Pinto et al. (2004, p.220) as a part of a fragility analysis research using an alternative method not utilized in this thesis. The reinforcement bars of the columns and the beam sections are evenly distributed at the top and bottom of the sections with a) 0.857% reinforcement geometric percentage in the beam, subdivided in the ratio 2 to 1 between top and bottom in the end sections ($A - A$) and 1 to 2 at midspan ($B - B$), and b) a corresponding reinforcement ratio in the two columns equals 1.2%, evenly distributed at the top and bottom of the section

(C – C). The point loads applied on the columns are of magnitude $P = 500[kN]$, while the distributed load is $36[kN/m]$, applied along the beam. The case study model does not provide any specific details regarding the transverse reinforcement. In addition, The median shear capacity is directly reported without any specific information related to transverse reinforcement. As a result, the finite element model used in the study does not consider the confinement effects, assuming the concrete to be unconfined.

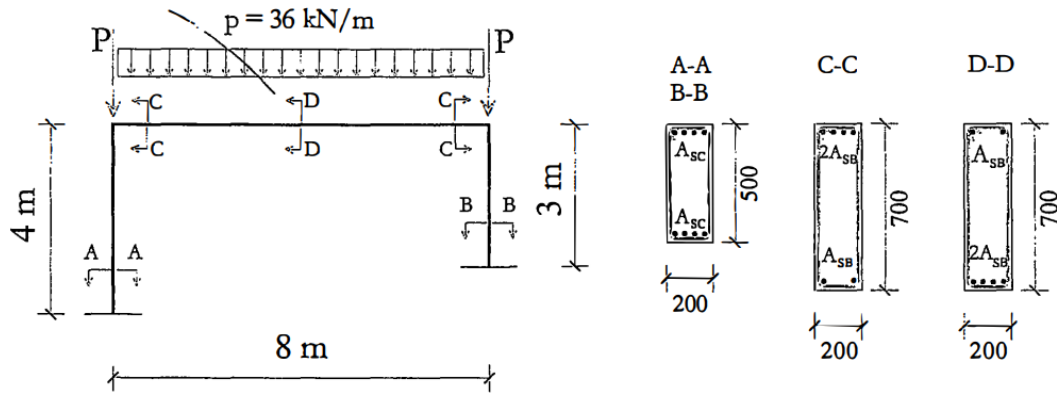


Figure 2.1 : Geometry and the section properties of the case study RC frame building model (Pinto et al., 2004, p.220).

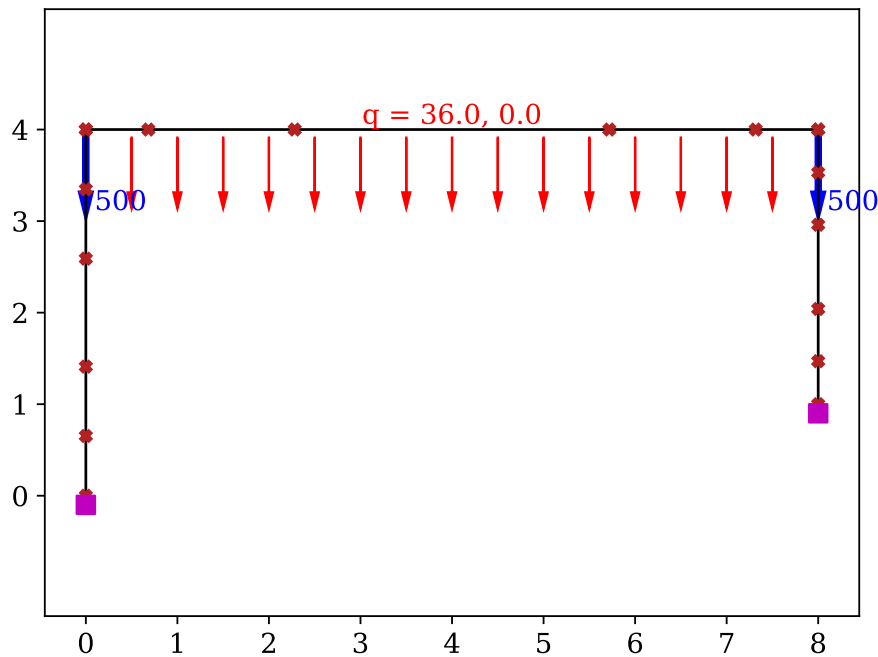


Figure 2.2 : Case study frame model with load distributions (Units: [kN, kN/m]).

The material properties and behavior for both the concrete and steel are explained in Section 2.2.1 and Section 2.2.2, respectively. This frame is designed to be vulnerable to three primary failure modes, all of which occur in the shorter column: shear failure due

to insufficient shear strength (V_u), as well as flexural failure due to concrete crushing (ϵ_{cu}) at the top and bottom sections of the column. These limit state failure criteria are considered as one of the main sources of uncertainty that impact the fragility (i.e. in addition to the record-to-record variability). Therefore, they are considered random as the variables in this study Figure 2.3. They are modelled using lognormal random variables with specific median and logarithmic standard deviations values listed in Table 2.1.

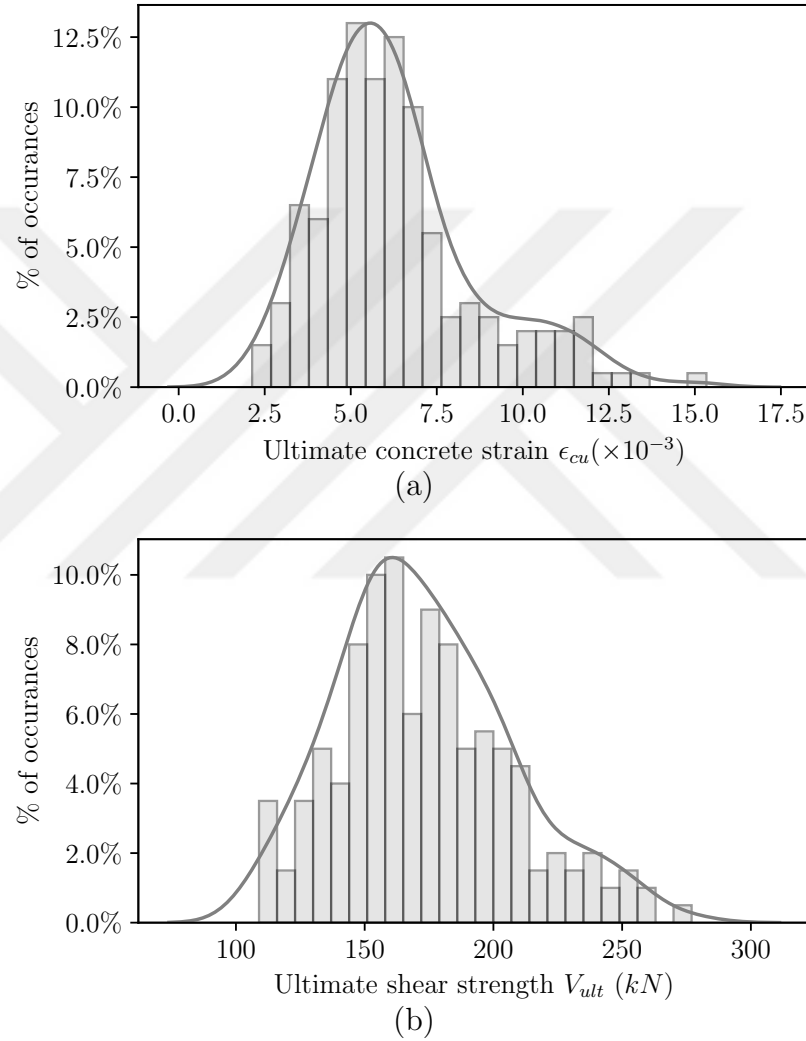


Figure 2.3 : Histograms of randomly generated log-normally distributed random variables: (a) ultimate concrete strain capacity and (b) ultimate shear capacity.

The open-source package OpenSees is an efficient tool for the finite element modeling and analysis of structures (Mazzoni et al., 2006). It is particularly capable of efficiently handling repetitive analysis tasks such as the Monte Carlo simulations performed in this thesis.

Table 2.1 : Distribution parameters of the random variables that define the failure limit state.

Random variable	V_u	ε_{cu}
<i>median</i> ($\hat{C}_x$)	170 [kN]	0.006
<i>dispersion</i> (β_{C_x})	0.2	0.35

In order to construct the OpenSees model, the following procedure is adopted:

- i) The structure's geometry is depicted using nodes.
- ii) Boundary conditions are defined for each node.
- iii) Materials, sections, and elements are defined, and dead loads are then applied.
- iv) The system of equations is assembled. Moreover, the constraint handler, solution algorithm, and integration scheme are adjusted.
- v) The system of equations is handled to obtain the response of the structure.
- vi) The results are post-processed and analyzed.

2.2 Material Properties

2.2.1 Concrete

The uniaxial constitutive material Concrete04 is compliant with TEC Turkish Earthquake Code (2018) (the regulations to be followed in modeling the structure in the analyses of our conducted study). This constitutive model is utilized to simulate concrete's behavior in compression, tension, and cyclic loading since it integrates multiple concrete models that are based on: a) the Popovics curve (Popovics, 1973), b) Mander's model (Mander et al., 1988), and c) Karsan-Jira model (Karsan and Jirsa, 1969). Uniaxial cyclic behavior of concrete that is obtained using this procedure can be seen in Figure 2.4. The material properties used to model concrete are listed in Table 2.2.

The concrete is considered as unconfined due to insufficient information provided about transverse reinforcement. At low strains up to strain at maximum strength of

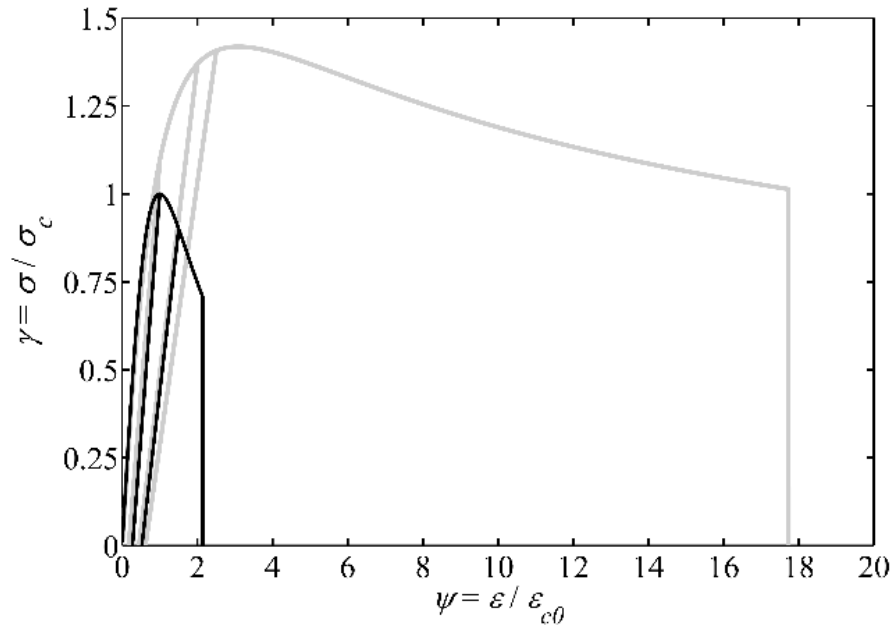


Figure 2.4 : Cyclic response of Concrete04 material model with both confined and unconfined model (Kashani et al., 2015).

Table 2.2 : Material properties used for defining the concrete model.

Concrete	
f_{cu}	30 MPa
E_c	30×10^3 MPa
ϵ_{c0}	0.2%
f_{tc}	3 MPa

concrete ϵ_{c0} , the concrete behaves as if in an elastic stress-strain relationship. The constitutive model uses a linear relationship to approximate the behavior of concrete during this stage. The concrete strain at maximum strength is estimated to be 0.22% in Eurocode CEN (2004) and 0.2% in TEC Turkish Earthquake Code (2018).

With increasing stress beyond this point, the Concrete04 constitutive model transitions to Popovics (1973) model, where concrete's compressive strength and stiffness gradually degrade due to cracking and damage until it reaches ultimate concrete strain, at which point it suffers a sudden failure. This ultimate concrete strain cannot be predicted accurately most of the time and has a critical impact on the ultimate deformation capacity of a structure. Therefore in this investigation, this strain capacity is considered as a random variable.

In addition, the constitutive model includes a cyclic degradation rule that simulates the material's behavior under cyclic loading. This degradation is simulated by Karsan and Jirsa (1969) model part that defines the slope of the loading-unloading curve. On the other hand, concrete's tension behavior is demonstrated as a linear-exponential decay curve.

2.2.2 Steel

The Uniaxial Steel02 model, which is based on the Giuffré-Menegotto-Pinto model (Filippou et al., 1983), is utilized to simulate the behavior of reinforcing steel in the concrete sections.

A key advantage of this constitutive material model is that it allows steel to undergo elastic deformation along with plastic deformation in the analysis. In this model, the Steel02 model is defined by a set of parameters that include the yield strength of steel (f_y), initial elastic modulus (E), and the strain hardening ratio (b), which represents the ratio between post-yield tangent and initial elastic tangent; all given in Table 2.3. The behavior of the constitutive material under cyclic loading is shown in Figure 2.5.

This model also considers the Bauschinger's effect, which is the tendency of steel to exhibit early yielding in tension following multiple reverse cyclic loadings. As a result, the Steel02 model is favored in our study as the constitutive material for simulating the behavior of steel reinforcing bars under seismic loading.

Table 2.3 : Material properties used for modeling reinforcement behavior.

Steel	
Model type	Steel02
f_y	400 [MPa]
E_s	210×10^3 [MPa]
b	0.03

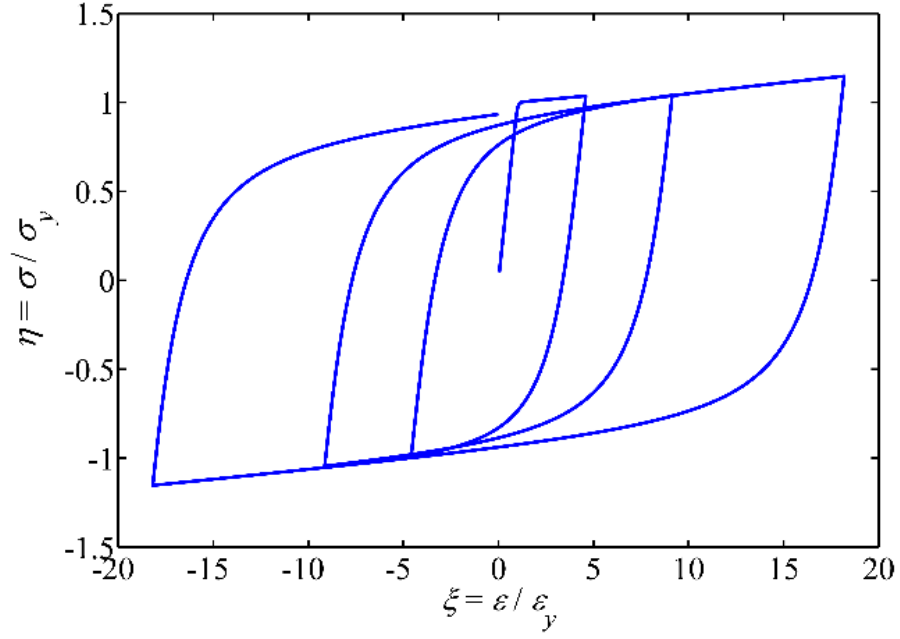


Figure 2.5 : Cyclic response of reinforcement fibers simulated using Steel02 material model (Kashani et al., 2015).

2.3 Member Sections

The structural properties of the element sections are presented in Table 2.4, with width of the section as b , depth as t , top reinforcement as A_{st} , and bottom reinforcement as A_{sb} . Sections A-A and B-B are for the long column and short column, respectively, while sections C-C and D-D represent the beam end sections and midspan sections.

Table 2.4 : Section properties: width, b , height, h , reinforcement at top, A_{st} , and reinforcement at the bottom A_{sb} .

Section	A-A	B-B	C-C	D-D
b	0.2	0.2	0.2	0.2
t	0.5	0.5	0.7	0.7
A_{st}	4 ϕ 14	4 ϕ 14	4 ϕ 16	2 ϕ 16
A_{sb}	4 ϕ 14	4 ϕ 14	2 ϕ 16	4 ϕ 16

In order to integrate the multiple stress-strain behaviors of the different constitutive materials that make up the reinforced concrete section of the elements of the frame; a fiber section is defined by dividing the cross section of the element into discrete

fibers, each of which has its own constitutive material properties. The concrete (Concrete04) is simulated using a mesh of rectangular-shaped fibers separated into 2 and 12 subdivisions in the local axes y and z , respectively. On the other hand, the steel is represented by a layer of evenly spaced points based on given data, each of which resembles a steel bar with Steel02 attributes.

When the two ends of a nonlinear beam-column finite element is constrained along the axial direction, erroneous axial forces may be exhibited in the element. These axial forces may cause unanticipated axial forces in beam members, and thus produce unexpected results. To ensure that no axial forces exhibits in the beam element, an elastic section with insignificant stiffness should be the beam section in the axial direction. This section is integrated into the beam section using the *SectionAggregator* object in OpenSees, which allows multiple sections to be combined into a single section.

2.4 Member Elements

The structural members of the frame are modeled using the *forceBeamColumn* element. This approach is based on force-based finite element formulations, rather than displacement-based formulations. Flexibility of the beam-column joint region was not considered in the model. The plasticity of the elements is defined by plastic hinges spread over the element at integration points (Scott and Fenves, 2006). This is done by applying two-point Gauss integration at the midspan of the element in addition to two-point Gauss-Radau integration applied to sections at each end of the element that are four times the length of the plastic hinge lengths, L_p at each end ($4L_{pI}$ and $4L_{pJ}$); a total of six integration points are used to capture the spread of plasticity along the element. The plastic hinge length is calculated using Equation 2.1 (Paulay and Priestley, 1992):

$$L_p = 0.08L_{cf} + 0.022\phi f_y \quad (2.1)$$

where L_{cf} is the distance of point of contraflexure from the critical section, ϕ is the diameter of the reinforcing bar, and f_y is the yield strength of the steel.

The location of the point of contraflexure is calculated from the bending moment diagram presented in the Section 2.5.1.

2.5 Analysis

2.5.1 Static analysis

The first step after setting the model is to analyze the model under the given set of dead loads by calculating the structure's internal forces without taking into account the effects of dynamic behavior. The outcomes of the static analysis procedure in OpenSees, as shown in Figure 2.6, Figure 2.7, and Figure 2.8, reveal that the dead loads applied result in normal forces carried solely by the columns, thanks to the effect of canceling out the axial stiffness of the beam. The beam, meanwhile, experienced a combination of shear forces and bending moments.

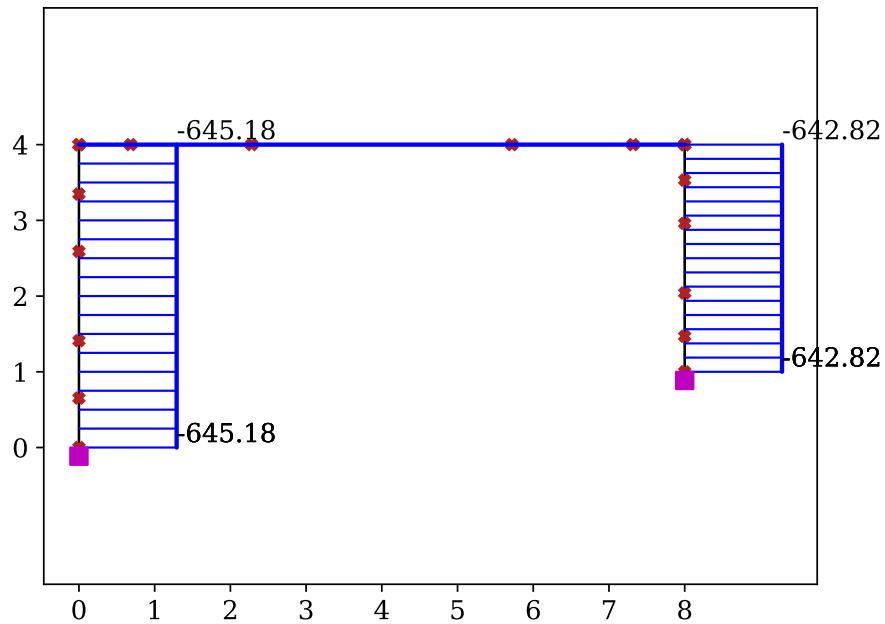


Figure 2.6 : Axial force diagram of the frame under static loading (Units: $[kN]$).

2.5.2 Modal analysis

In order to perform modal analysis, it is essential to construct a mathematical model of the frame with the frame's mass integrated. Pinto et al. (2006) states that fundamental vibration period of the frame is 0.4s. Using this target fundamental period, $T_1 = 0.48[s]$

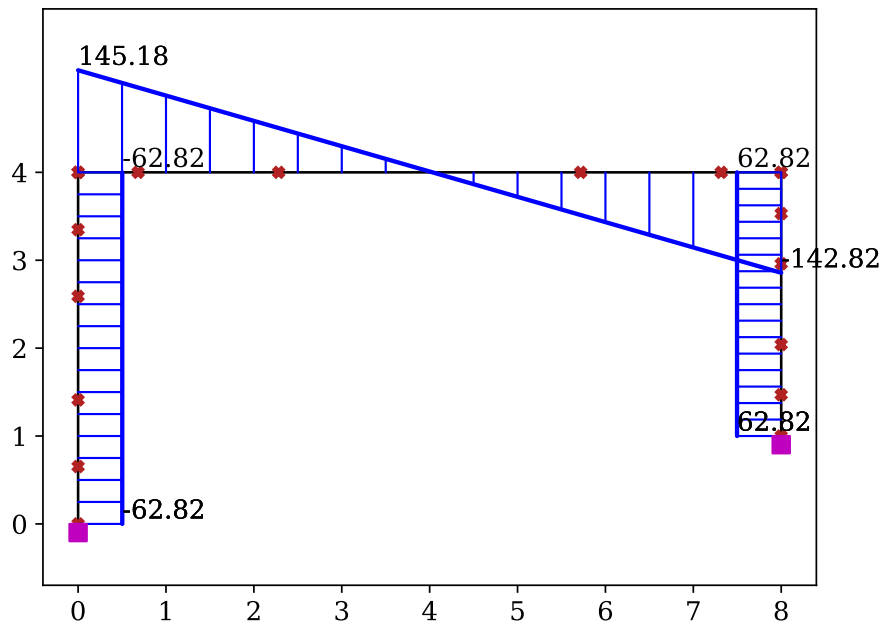


Figure 2.7 : Shear force diagram of the frame under static loading (Units: $[kN]$).

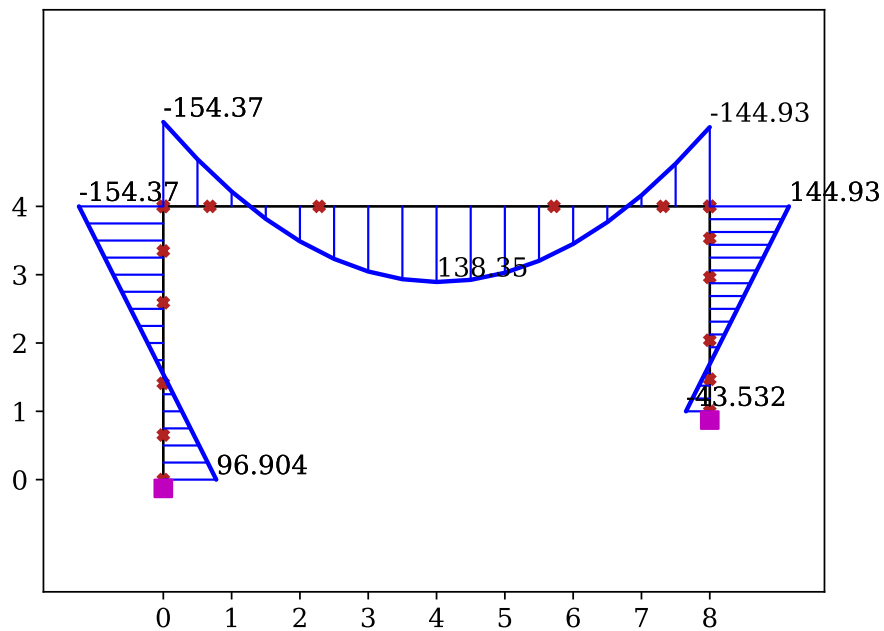


Figure 2.8 : Bending moment diagram of the frame under static loading (Units: $[kN.m]$).

as well as the structure's lateral stiffness, it is straightforward to compute the mass and afterwards apply it as nodal mass distributed across the frame nodes.

Based on this assessment the masses of $m = 234[t]$ were identified to be suitable and this mass value was defined for the two nodes located at the two ends of the beam. To make sure the structure is properly modeled, the fundamental vibration period is recalculated and confirmed to be approximately equal to $\hat{T}_1 = 0.51[s]$. The outcomes of the modal analysis is shown in Figure 2.9.

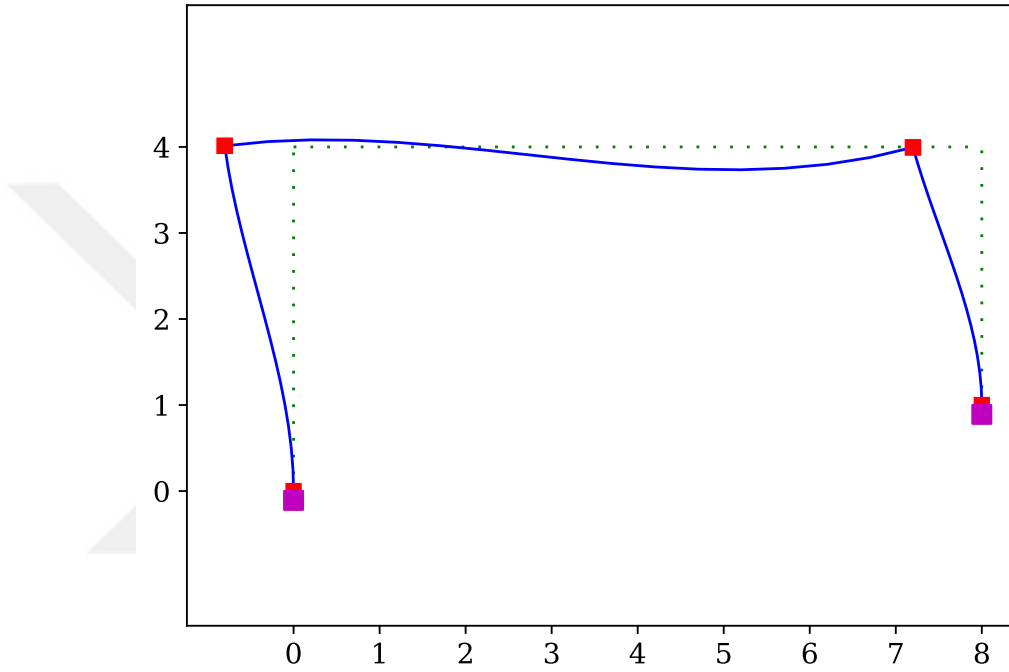


Figure 2.9 : Fundamental mode shape of the frame.

2.5.3 Pushover analysis

The process of static pushover analysis involves the application of a lateral quasi statically-increasing load until one of the limit state criteria is triggered.

The result of the pushover analysis for the median case of our study frame is presented in Figure 2.10, which reveals that the frame exhibits a relatively ductile behavior up to its failure point, which is observed at a base shear of about 200 [kN].

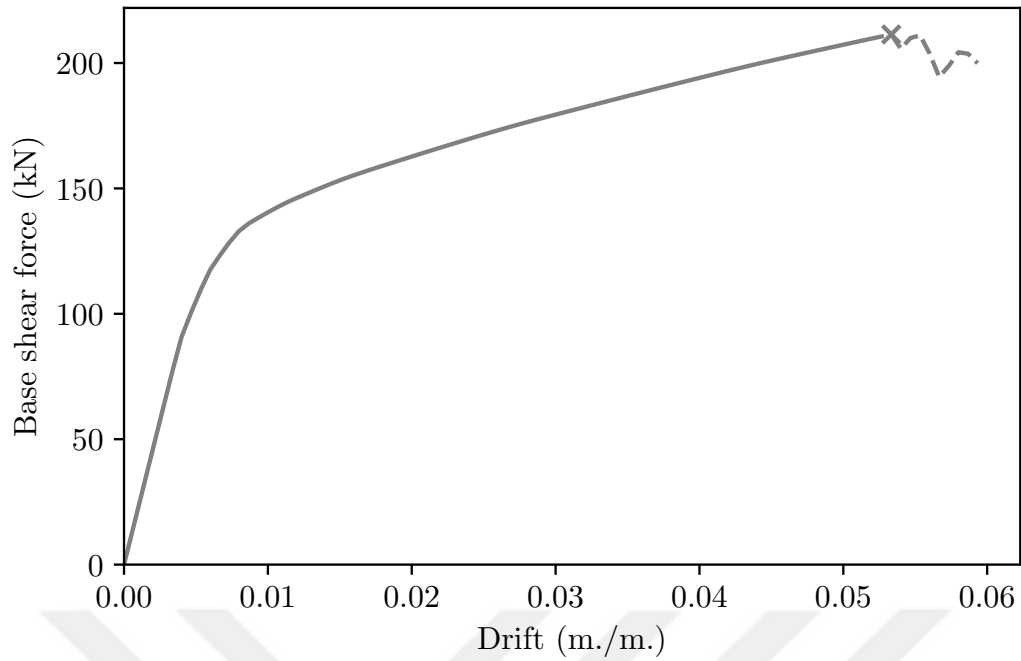


Figure 2.10 : Static pushover curve of the frame.

2.5.4 Moment-Curvature analysis

The Moment-Curvature Analysis of each section is performed using OpenSees as follows:

- i) Two nodes at the origin point are defined, one of which is fixed completely and the other of which is restrained only in the axial and bending directions.
- ii) A *zeroLengthSection* element is defined that will make use of the section and geometric transformation objects.
- iii) A recorder is configured to capture node displacement as well as the stress-strain deformation of fiber sections.
- iv) The integrator technique is set to *LoadControl* and a single analysis is performed under a 'constant' time series that controls the application of the axial load.
- v) The section is thereafter to be subjected to a monotonic increasing moment; the time series is *Linear*; and the integrator is *DisplacementControl*.

vi) The analysis is repeated on the section several times up to the ultimate strain point. The results are subsequently retrieved from the recorders, post-processed, and analyzed.

Figure 2.11 shows that the reinforced concrete column section—which is subjected to axial loads—exhibited ductile behavior with significant plastic deformation as concrete reached its maximum point. The main controlling factor in the failure of the column section is the ultimate strain of concrete.

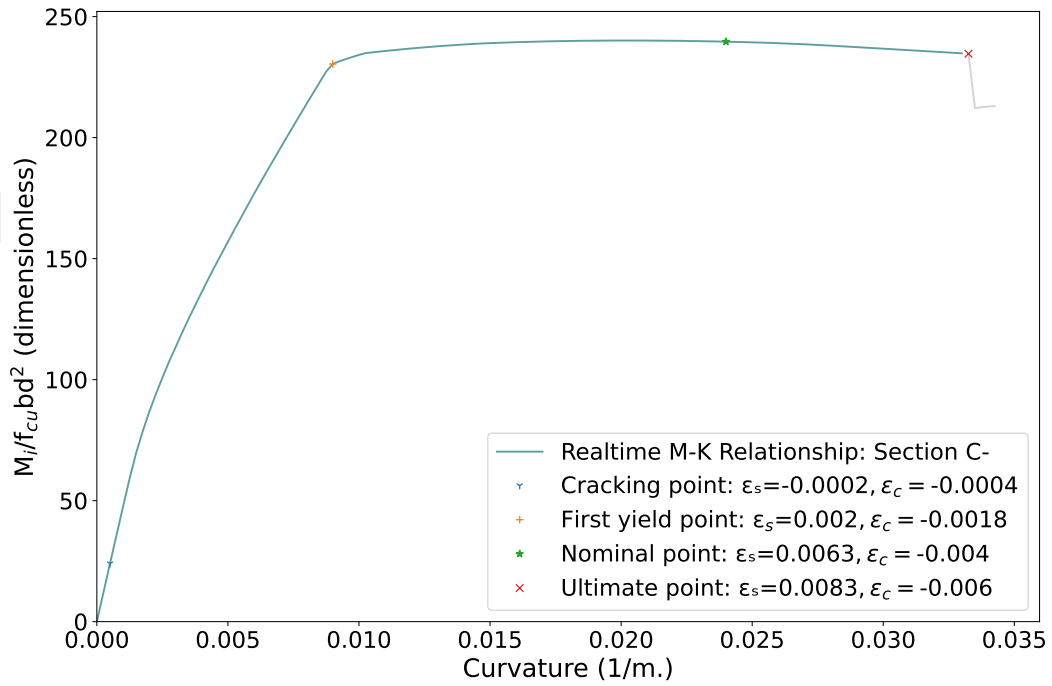


Figure 2.11 : Moment-Curvature relationship for the column section.

On the other hand, it is observed from Figure 2.12 that in the beam section, when steel reaches its ultimate strain point and can no longer undergo further plastic deformation, the moment-curvature counter-intuitively still goes further upwards beyond this point; this is due to the fact that concrete is still undergoing further deformation, namely cracking and crushing. This fact ensured the ductility behavior required from the beam.

2.6 Development of the Meta-Model

A simplified version of the numerical model called a meta-model is utilized in machine learning algorithms. The meta-model enables the use of Monte Carlo simulations and optimization calculations (Ait-Amir et al., 2015). The meta-model is developed by

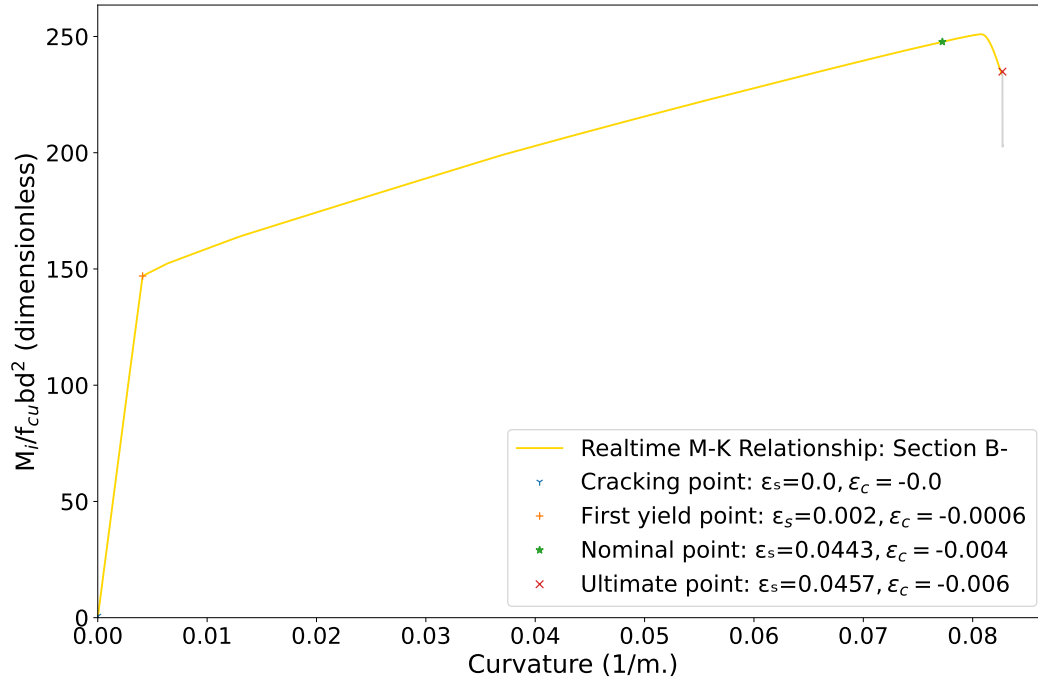


Figure 2.12 : Moment-Curvature relationship for beam section.

having the input parameters undergo a normalization process. This process introduces newly scaled meta-variables having same mean and standard deviation. This ensures the consistency of both input parameters so that no meta-variable has an undue influence on the results due to differences between ranges. According to Anysz et al. (2016), the standardization of input and output data can significantly affect the accuracy of predictions made by machine learning algorithms, e.g., artificial neural networks. In our case study, the limit-state input variables are scaled into standard meta-variables with a mean value (μ) of zero and a standard deviation (σ^2) of 1.

2.7 Ground Motion Selection

Some researchers have warned about potential effects of the limited ground motion dataset on the evaluation results. In fact, A study conducted by Buratti et al. (2010) suggested that a minimum of 9 ground-motion time histories should be used to accurately capture ground-motion variability in the model. Another case study by Jia and Wu (2022b) conducted with 10 ground motion records claimed that the traditional lognormal assumption may lead to inaccurate evaluation results. Accordingly, 12 ground motion records are used in our analysis; which, in addition, comply with the recommendations of TEC Turkish Earthquake Code (2018). The selected ground

motion records, displayed in Table A.1, are selected from Akkar et al. (2005) ground motion list. Their magnitude range between 6.0 and 7.6 with small epicentral distances up to 16 km, and soil profiles correspond to NEHRP C and D sites.

2.8 Intensity Measures

In the literature, peak ground acceleration (PGA) and pseudo-spectral acceleration (PSA) are typical choices as intensity measure indicators (Choun and Elnashai, 2010; Liel et al., 2009; Shinozuka et al., 2000). One of the main advantages of PGA is its availability in the ground motion records data, which is convenient for empirical derivation of fragility models from earthquake data (Hwang et al., 2001). However, some studies criticized the usage of these intensity measure indicators in literature, indicating that they may not be suitable seismic intensity measures in the case of developing fragility models for nonlinear systems (Kafali and Grigoriu, 2007; Radu and Grigoriu, 2018). Moreover, a research conducted by Elenas (2000) showed that the Peak Ground Acceleration (PGA) intensity measure has a weak correlation with overall structural damage indices.

The study of Lu et al. (2013) found that PGV correlated well with story drift ratio as the intensity measure indicator rather than others such as peak ground acceleration (PGA) and spectral acceleration of the fundamental period ($S_A(T_1)$). The study also found that PGV reduces dispersion of structural response when the structure exhibits nonlinear deformations, stating that it is able to provide efficiency in collapse analysis.

2.9 Incremental Dynamic Analysis

The IDA methodology involves conducting nonlinear dynamic structural analysis using a ground motion record that is incrementally scaled until a specific damage limit is reached. The IDA approach was first introduced by Vamvatsikos and Cornell (2002, 2004). Later, Baker (2015) proposed new methods to post-process the outcomes obtained through the IDA. A significant benefit of the IDA approach is its ability to obtain a set of structural capacities while estimating their distribution and parameters using maximum likelihood estimation method. However, this comes with the limitation of conducting a greater number of time-consuming nonlinear analyses

up to the collapse point (Zentner et al., 2017). In our case study, the response of the structure is simulated to different levels of seismic activity, ranging from a minor shake having $PGV = 5\text{cm/s}$ to a destructive strong motion equivalent to the Izmit 1999 earthquake with $PGV = 40\text{cm/s}$.

2.10 Fragility Model Formulation

After performing IDA using the limit state meta-variables, a fragility model is developed for each limit state combination based on the probability of exceeding one of the limit states in the set of ground motions. In previous studies, the log-normal cumulative density function was confirmed as a good fit for the fragility curve; this confirmation comes from tests of goodness-of-fit (Akkar et al., 2005; Kircher et al., 1997; Shinozuka et al., 2000). The logarithmic mean λ and logarithmic standard deviation ζ are used to define the cumulative distribution function (CDF) of the equivalent log-normal fragility model in Equation (2.2). Fragility model is defined using the following expression:

$$P(D > d | IM = x) = \Phi\left(\frac{\ln(x) - \lambda}{\zeta}\right) \quad (2.2)$$

In the equation above, λ and ζ are the fragility function parameters and Maximum Likelihood Estimation (MLE) approach is used to determine their values.

In order to capture the impact of uncertainty associated with key random features, a huge set of realizations of random value combinations is generated. In order to measure the level and tendency of the probability of failure is computed at each intensity measure. Typically, the most common measures of central tendency for continuous variables are the mean and median. The mean is simple and easy to comprehend; however, it is easily affected by outliers (Thorne and Giesen, 2000). The median is considered advantageous in our analysis since it can provide an accurate estimate of the central tendency as well as λ parameter of the fragility function. One more advantage of using the median instead of the mean is that the median can be more robust to outliers or extreme values in the dataset.

The median at each intensity measure is selected, and then the CDF's parameters are estimated again. Ultimately, a final fragility curve is then developed from the newly estimated CDF's parameters.

2.11 Benchmark Model

Monte Carlo Simulation approach is used to develop the benchmark model. Monte Carlo approach is the approach that demands greatest computation resources. The steps for establishing a benchmark model are as follows:

- i) A set of random parameter value combinations is generated from the predefined median and dispersion. This set is large enough so as to provide an unalterable final fragility model. On the other hand, among all alternative options considered here, establishing fragility models using crude Monte Carlo is the most computationally intensive one.
- ii) Incremental Dynamic Analysis (IDA) is conducted using a suite of ground motion records applied to the structure one at a time.
- iii) The probability of triggering each limit-state at each intensity measure (IM) is computed from the results of IDA.
- iv) Fragility function parameters are estimated using maximum likelihood estimation method from the median probability at each IM level to develop a fragility model.

2.12 Case Study II: High Variance Case Study

For further exploration of the applicability of machine learning techniques in producing fragility models, a second case study is proposed. In this case study, a high variance model is studied. This model is generated by doubling the dispersion of both material uncertainties, which in turn perplexing the nonlinearity of the relationship between the limit states and the response. The random variable distribution of the high variance limit states is shown in Table 2.5 and Figure 2.13. A box plot comparison between the standard case and high variance case is displayed in Figure 2.14 with the standard case colored in yellow and the high variance case in red. The lower and upper

quartile whiskers represent 16% and 84%, respectively, whereas the mid-quartile in the box represents the median of the dataset.

Table 2.5 : High variance case: Parameters that define the limit state failure criteria random variables.

Random variable	Vu	ε_{cu}
<i>median</i> ($\hat{C}_x$)	170 kN	0.006
<i>dispersion</i> (β_{C_x})	0.4	0.7

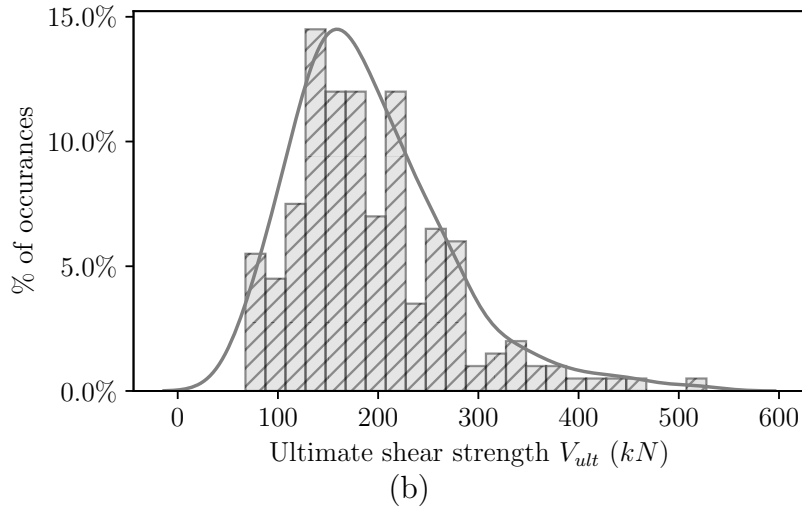
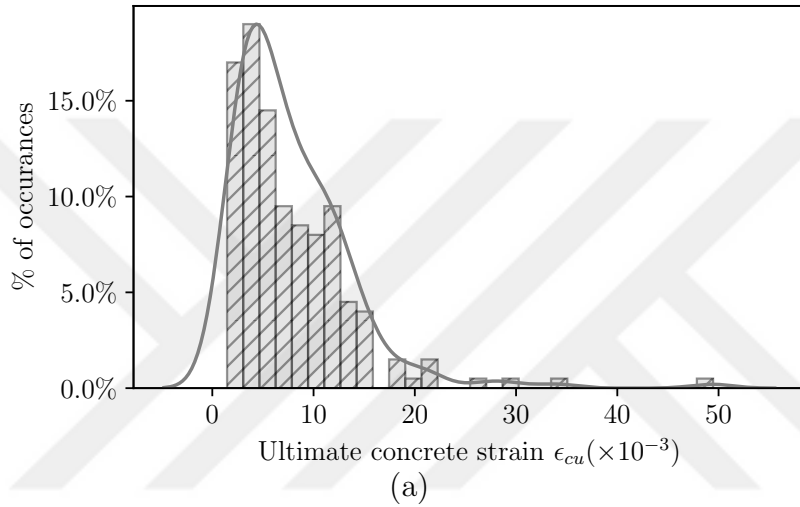
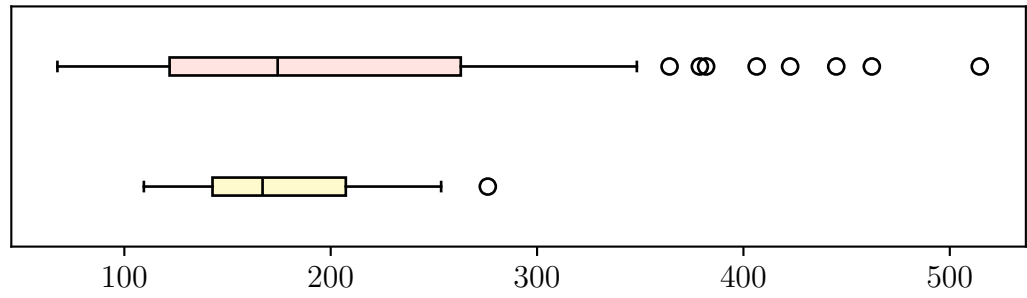
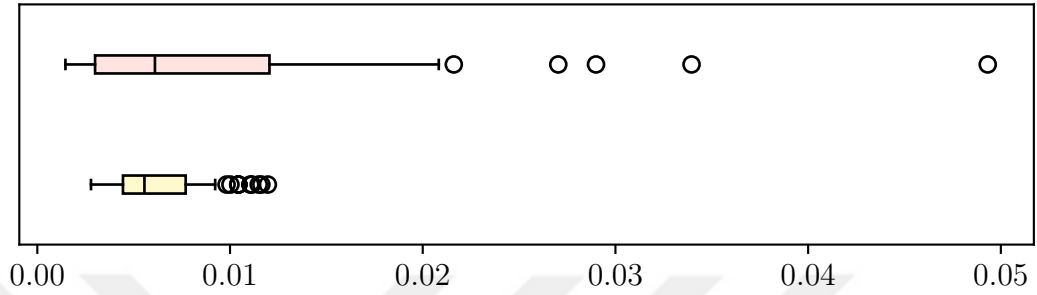


Figure 2.13 : Histograms of random log-normal distribution of dominant limit-state criteria for high variance model (a) ultimate concrete strain capacity (b) ultimate shear capacity.

The same analysis steps are repeated to develop the fragility model. The efficiency of machine learning techniques is then to be discussed. Increasing the variance of



(a) V_{bu} distribution set [kN]



(b) ε_{cu} distribution set [$m./m.$]

Figure 2.14 : Box plot of a) ultimate concrete strain and b) ultimate shear random log-normal limit state variables for: standard case (in yellow) and high variance case (in red).

the limit state random variables causes them to spread out around the mean, which in turns is expected to shift the fragility curve indicating a different level of structural vulnerability.



3. MACHINE LEARNING ALGORITHMS: RESPONSE SURFACE METHOD

3.1 General

The study investigates the benefits and limitations of the Response Surface Method (RSM) approach as an alternative machine learning algorithm to the Monte Carlo simulations based benchmark model in seismic fragility analysis. The primary goal of RSM approach is to optimize the output while balancing computational power and accuracy. To achieve this goal, RSM algorithm approaches the final response through developing an approximate mathematical polynomial equation. This equation explores the relationships between independent meta-variables and how their variation may impact on the final response. RSM approach is extensively applied in developing models which their several input variables potentially influence the expected response (Myers et al., 2009). The input parameters in our case study are the shear strength and concrete strain capacity meta-variables and the outputs are the fragility distribution parameters.

In order to develop the polynomial equation, RSM relies on a training dataset of experimental design points that are collected to cover a particular region of input meta-variables of interest. These points are selected based on one of experimental design methods to be discussed in Section 3.2. With this dataset, the model is fitted using the least square method to find the polynomial coefficients that best fit the relationship between the input meta-variables and output response. This polynomial equation is capable of predicting final fragility models for different combinations of random capacity meta-variables within the specified range. In addition, it provides insights into effect of uncertainties on the system's final fragility model when applied on a mega data-set of randomly generated input meta-variables.

The procedure for utilizing the RSM approach in the simulation of seismic fragility model can be outlined in the following steps:

3.2 Procedure

To apply the response surface method to develop a fragility model of a structure while incorporating uncertainties in both the limit state criteria and seismicity properties, the following steps are followed:

3.2.1 Specifying the inputs and outputs

Typically, before applying any machine learning approach, researcher first needs to identify the factors that are likely to affect the outcome response in the aim of eliminating nonessential variables. These influencing variables are identified either by experience or by screening experiments (Myers et al., 2009). The input parameters in our case study are related to the independent variables of the limit-state criteria of the short column, specifically the flexural failure of the upper and lower sections of the column due to reaching the ultimate concrete strain capacity, as well as the ultimate shear capacity. These limit state criteria are subjected to normalization process as mentioned before in Section 2.6 and is termed in later sections as meta-variables. The analysis aims to derive two desired outcomes: the log-mean (λ) and log standard deviation (ζ) of the log-normal cumulative density function that is used in constructing the fragility model.

In our study, it is essential to incorporate the uncertainties related to the failure criteria in the fragility model, along with the uncertainties associated with ground motion records. The limit state criteria uncertainties are categorized as epistemic uncertainties, which are defined as independent variables in the mathematical equation. On the other hand, the uncertainties related to the ground motion records are aleatoric uncertainties that are included in the system by performing multiple analyses and estimating the median out of the analyses results.

3.2.2 Determination of the levels and range of the inputs

One of the essential stages in implementing RSM is to determine the values of the input variables at which data used to train the RSM formula is to be gathered; these values are known as experimental points. The choice of experimental points plays a crucial role as

it can have a significant effect on the accuracy and efficiency of the model (Ait-Amir et al., 2015). One approach commonly employed to pick experimental points is the central composite design (CCD) approach.

Central composite design, first introduced by Box and Wilson (1951), is a structured technique that generates a series of experimental runs through the combination of factorial and axial points, in addition to center points for the estimation of pure error and curvature. These points cover the input variable space and are selected based on statistical formulas to guarantee adequate sampling of the response surface. The configuration of the CCD approach is shown in Figure 3.1. The range of experimental data to be investigated can be determined by a regression analysis using a linear model to find an approximate with a view towards finding the location of the best fit. Once the optimal region is identified, a more sophisticated model can be used around this region (Montgomery, 2012). In our case study, the range is nonetheless selected based on the parameters of the random variables considered.

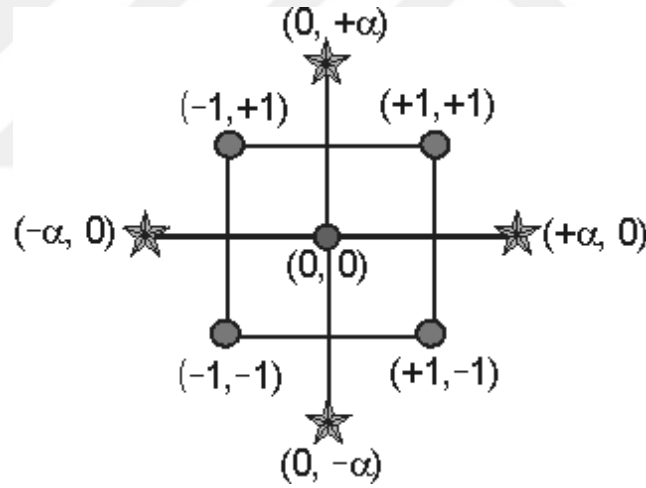


Figure 3.1 : Central composite design configuration with axial points represented as circles and star points as stars.

In the limit-state real value domain, the experimental design for the first limit-state variable, ultimate concrete strain, consists of three factorial points, 4.356×10^{-3} , 6×10^{-3} , and 8.243×10^{-3} . The experimental design also incorporates two star-points ($\pm\alpha$) for this variable, each having a value of 3.667×10^{-3} and 9.164×10^{-3} , respectively. These star points are included to account for the anticipated highly nonlinear response. Regarding the second variable, ultimate shear capacity, the factorial points are 137.1

[kN], 170 [kN], and 205.5 [kN], with two star-points at 121.6 [kN] and 218.4 [kN], respectively. The experimental design steps for each combination of design points in both meta-variable domain and actual values domain are listed in Table 3.1.

Table 3.1 : Experimental design points of the response surface method.

# of exp.	$\alpha_{\epsilon_{cu}}$	α_{V_u}	ϵ_{cu}	V_u
1	$-\sqrt{2}$	0	3.667×10^{-3}	170.0 [kN]
2	-1	-1	4.356×10^{-3}	137.1 [kN]
3	-1	1	4.356×10^{-3}	205.5 [kN]
4	0	$-\sqrt{2}$	6.000×10^{-3}	121.6 [kN]
5	0	0	6.000×10^{-3}	170.0 [kN]
6	0	$\sqrt{2}$	6.000×10^{-3}	205.5 [kN]
7	1	-1	8.243×10^{-3}	137.1 [kN]
8	1	1	8.243×10^{-3}	205.5 [kN]
9	$\sqrt{2}$	0	9.164×10^{-3}	170.0 [kN]

3.2.3 Training the response surface

The training of the model in response surface methods involves fitting a mathematical polynomial function to the data collected from the experimental design. In this study, Polynomial bivariable equations are selected for the analysis to imitate the relationship between the input parameters and the outcome. Our study focuses on optimizing a response surface based on a quadratic equation as in Equation (3.1) for estimating the logarithmic mean and a linear one as in Equation (3.2) for the logarithmic standard deviation. The majority of RSM related researches rely on either one or both of these equations (Montgomery, 2012). Nevertheless, higher degrees are examined to assess of the impact of increasing the degree of the polynomial equation on the accuracy of capturing the response surface model.

$$y = \beta_0 + \sum_{i=1}^k \beta_i x_i + \sum_{i=1}^k \beta_{ii} x_i^2 + \sum_{i=1}^k \sum_{j=1}^k \beta_{ij} x_i x_j \quad (3.1)$$

$$y = \beta_0 + \sum_{i=1}^k \sum_{j=1}^k \beta_{ij} x_i x_j + \sum_{i=1}^k \beta_i x_i \quad (3.2)$$

where y is the output variable, x_i and x_j are the input meta-variables, and β are the regression coefficients.

Fitting the polynomial function to the collected data required regression analysis to be employed throughout the training phase. For each experiment combination of the input parameters, incremental dynamic analyses are performed using the pre-specified suite of ground motion records. This regression analysis aims to find the best-fit equation that approximate the relationship between the input variables and the output responses, so that this equation can then be used to predict the output responses for any given set of input variables. The coefficients of the polynomial equation are estimated by least square method using *scikit – learn* library and fragility models for each experimental design point are developed using the approximated log-normal parameters. Following this, a 3-D mesh surface is plotted by *matplotlib* library in python.

3.2.4 Developing the final fragility model

Once the RSM equation is established, deriving the final fragility model becomes a swift and straightforward process. The first step is to input a mega set (1000) of randomly generated meta-variables combinations into the RSM equations to determine their outcomes. Subsequently, the fragility models are derived for all 1000 meta-variable combinations. Ultimately, the final fragility model of the RSM method is now ready to be fitted through a log-normal cumulative density curve using the median points of the megaset at each intensity measure level. The results are to be compared to the benchmark model with a view towards assessing the effectiveness and accuracy of the RSM approach.



4. MACHINE LEARNING ALGORITHMS: ARTIFICIAL NEURAL NETWORKS

4.1 General

Artificial Neural Networks (ANN) is the second machine learning algorithm to be investigated in the study. ANNs are essentially made up of networks of layered, interconnected nodes called neurons that are inspired by the structure the human brain and how it functions (Haykin, 1999). These neurons have the ability to learn from training data provided by the user, through adjusting each weight of their inputs. Each neuron receives input variables multiplied by their estimated weights from a previous layer. The neuron then sums them up and passes the resultant through an activation function before sending the output to the next layer. This computational operation is described in Figure 4.1. The variety of inputs and weights that are passed to many neurons along with the activation function before passing to the next layers bring nonlinearity into the computation (He and Fu, 2001).

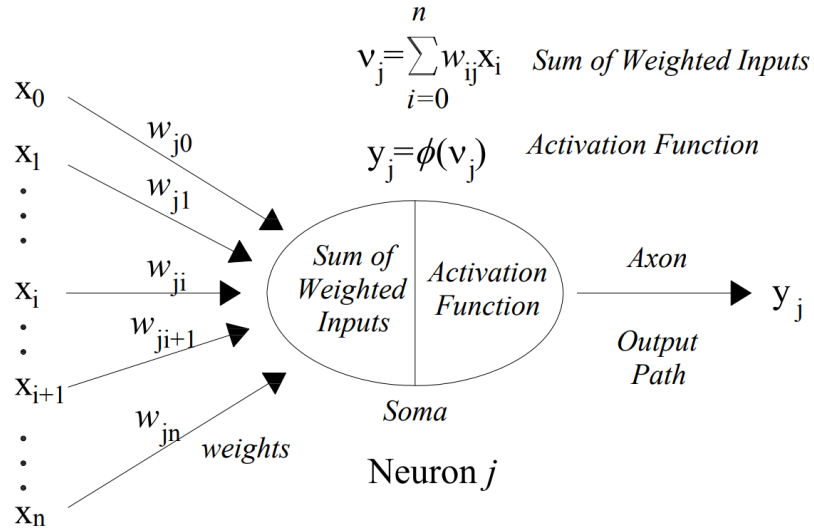


Figure 4.1 : Neurons computational operations in Artificial neural networks approach (Şahin, 2010).

Artificial neural networks approach has emerged as an alternative method to conventional regression in recent years, since it is capable of detecting and modeling complex and nonlinear relationships between input and output variables (Abiodun et al., 2018). However, some researchers, including Benítez et al. (1997), have criticized ANNs for being "black boxes", meaning that it can be difficult to interpret how they arrive at their predictions.

MLP with backpropagation has gained credibility in the research field since it is a quick and straightforward algorithm with simple inputs that do not involve any black boxes. For instance, in a study conducted by Rasheed et al. (2022), it was found that MLP performed better than the radial base function approach in predicting the optimum design parameters of the slab. Furthermore, another study found that MLP produced more reliable fragility models compared to the genetic programming algorithm (Rafiq et al., 2001). However, training data should be preprocessed carefully as the algorithm is highly sensitive to noise in the data, which may lead to errors (Braspenning et al., 1995).

In our research, the Multi-Layer Perceptron (MLP) with backpropagation is considered as the technique used in training the neural networks. Typically, the neural networks used to be trained in the MLP technique consist of an input layer, one or more hidden layers, and an output layer. MLP trains the neural networks by adjusting the weights using gradient descent optimization. Each weight of neuron are fine-tuned using backward pass in aim of finding a model that fits the training data the best.

4.2 Procedure

The process of utilizing the artificial neural networks for developing fragility analysis in a structure with multiple failure criteria involves the following steps:

4.2.1 Determination of inputs and outputs

The input parameters and the acquired response are set same as in response surface method. The inputs for the neural networks are the failure criteria, and the outputs are the logarithmic mean (λ) and logarithmic standard deviation (ζ) of the approximate log-normal cumulative density function.

The nonlinearity and complexity of the relationship between the input parameters and the response should be thoroughly examined in order for neural networks to be capable of predicting fragilities with high reliability. For example, a very low limit state might govern the failure state of the structure, and thus increasing the other limit state is expected to have no effect. If this issue is resolved, the efficiency of the neural networks model will improve. However, more training efforts may be required, and using a larger number of data points for the learning stage can be computationally intensive. Therefore, the training data should allow for a sufficient number of training points objective enough to capture the full range of uncertainties in the system, without overwhelming the computational resources required to train the model. In this regard, a grid of evenly spaced random parameter value combinations is generated, consisting of 7 parameter values from both random variables, each limit-state value ranges from -2 to 2, which corresponds to the 3rd and 97th percentiles in the Z-score table. Each two of the meta-variables are paired together, summing up to a final training dataset of 49 combinations.

4.2.2 Adjusting the networks architecture

For each parameter of the fragility function, a neural networks model is created using multilayer perceptron architecture. Each of these models consists of: (1) an input layer that takes a combination of limit states, (2) three hidden layers of the networks contain 5 neurons each; and (3) the predicted response of the parameter. Figure 4.2 depicts the multilayer perceptron networks to be used in our study. To address any high bias that may occur, a small L2 regularization term of 0.1% is applied. Moreover, the "logistic" function is used as the activation function for the network layers.

4.2.3 Training the neural networks

The training dataset is arbitrary divided into 7 folds. One fold is reserved as a validation set, while the remaining folds are used for training the neural networks. During the training process, the neural networks learn the patterns between the input data and their corresponding output data by adjusting the weights of the neurons using an optimization algorithm, specifically a quasi-Newton method.

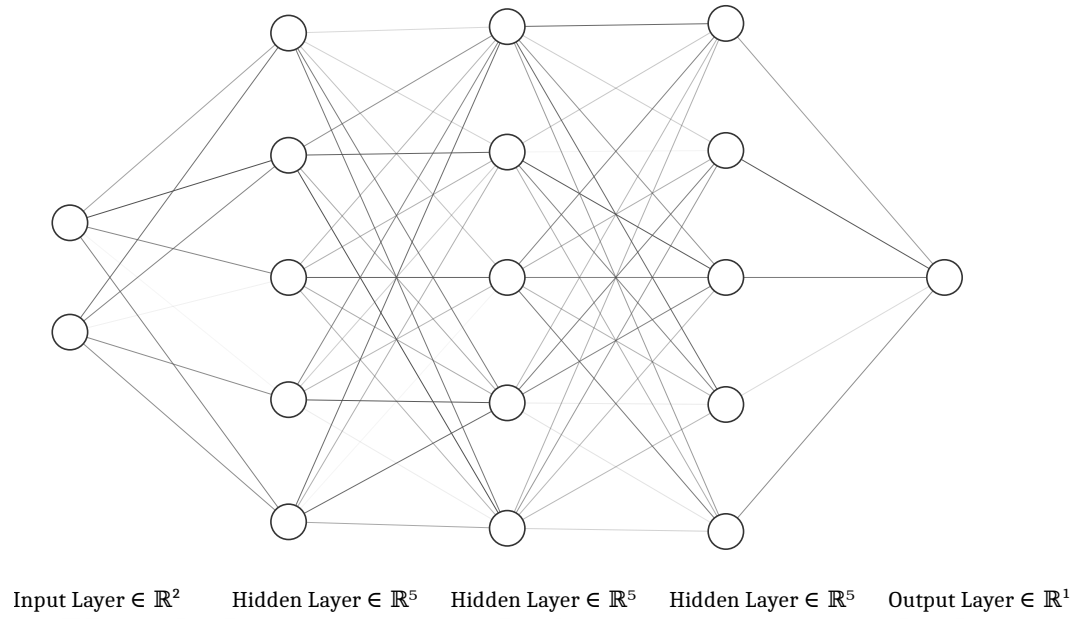


Figure 4.2 : Multilayer perceptron neural networks configuration (LeNail, 2019).

The validation dataset plays a crucial role in conducting sensitivity analysis to determine the optimal constitutive properties of the neural networks. Its significance lies in preventing overfitting and ensuring accurate predictions of fragility function parameters for new limit-state combinations beyond the training data range.

To assess the performance of the neural networks, a separate test dataset comprising 200 benchmark points, utilized in developing the benchmark model, is used. The Table 4.1 shows the distribution of datasets used in growing our neural networks.

Table 4.1 : The distribution of training, validation, and test data points.

Training dataset	Validation dataset	Test dataset
6 folds out of 7 folds of the training dataset	1 fold out of 7 folds of the training dataset	Benchmark points dataset
42 observations (17%)	7 observations (3%)	200 observations (80%)

4.2.4 Developing the final fragility model

Once neural networks are successfully optimized with the training data, it can be used to develop a final fragility model that takes into account the uncertainties and nonlinearities of the system. First, a set of 1000 realization of limit state combination is inputted into the network. The networks then develop a fragility model for each combination. These realizations are large enough to represent the range of possible values for the various metavariables involved in the analysis. Following the developing of the fragility models for all 1000 meta-variable combinations, the median value at each intensity measure is subsequently calculated. Finally, a final fragility model is estimated with maximum likelihood estimation using these medians, and that final fragility model is to be compared with the benchmark model and the Response Surface Method model.



5. DISCUSSION OF RESULTS

5.1 Results of Monte Carlo Simulations

A sensitivity analysis is conducted to find the size of the Monte Carlo random simulations set. As a result, its findings, displayed in Figure 5.1, show that 200 random realizations of limit state meta-variable combinations have sufficient unbiasedness to develop an invariant final fragility model. After conducting incremental dynamic analysis for this set of limit state combinations, parameters defining the fragility function for each combination are estimated using MLE: the log-mean and log standard deviation of the CDF. The results of log-mean of each combination shown in Figure 5.2 reveal an upward trend when increasing the values of the meta-variables. Consequently, the probabilities of failure at each intensity measure is calculated from the fragility parameters estimates as shown in Figure 5.3. Ultimately, the final fragility model for the benchmark is fitted to a cumulative distribution curve with logarithmic mean (λ) and logarithmic standard deviation (ζ) estimated through maximum likelihood estimation from the medians of probability of failure at each intensity measure level, as shown in Figure 5.4.

5.2 Results of Response Surface Method

At each experimental design point, an incremental dynamic analysis is conducted for the suite of ground motions, and the number of times the system reaches failure state is calculated at each intensity measure step, shown in Table A.3. Fragility models are then developed for each design point, as shown in Figure 5.5. The results demonstrate a steady shift in the fragility model with increasing the meta-variables. However, at very low meta-variables, such as $-\alpha$ and -1 , any increase in the other meta-variable has nearly a negligible effect on shifting the fragility model. This phenomenon suggests that the structure is already due failure for these low values, and increasing the other meta-variables may not have any impact on improving the fragility model.

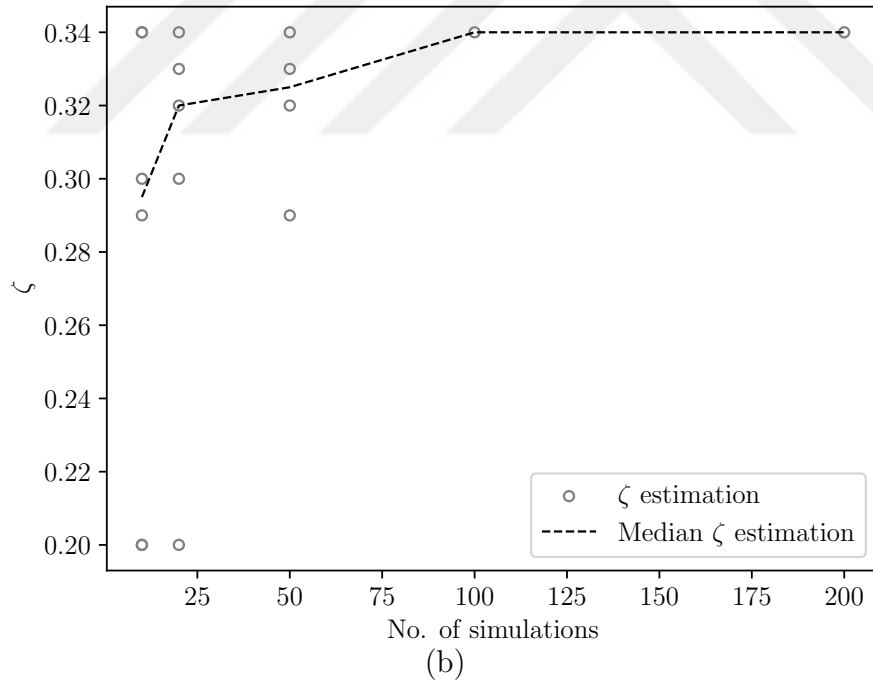
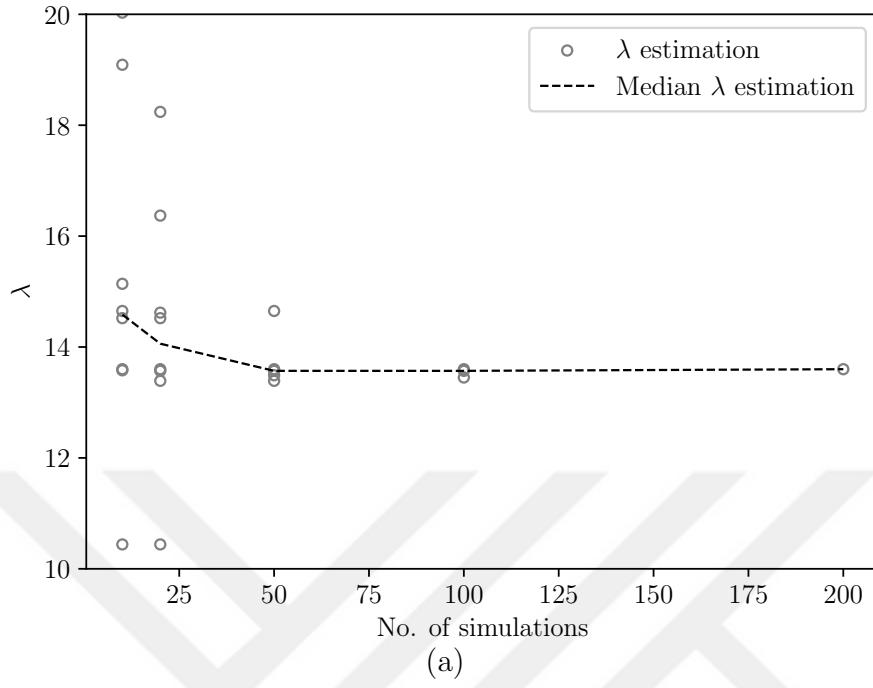


Figure 5.1 : Size of simulations set vs. (a) Estimated median fragility (λ) in [cm/s] and, (b) Estimated ζ parameter of fragility of the benchmark points using Maximum Likelihood Estimation (MLE) approach.

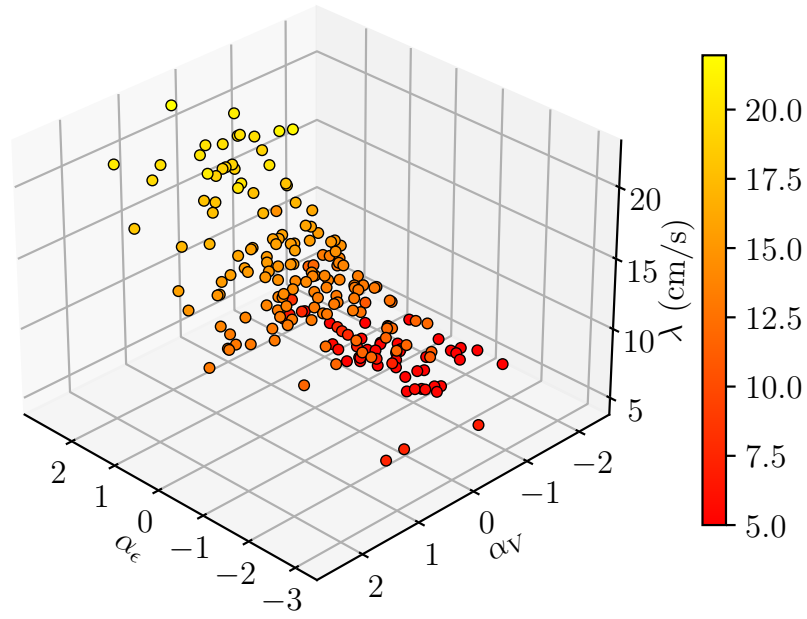


Figure 5.2 : Estimated median fragility (λ) of the benchmark points using Maximum Likelihood Estimation (MLE) approach.

Two response surfaces, of linear and quadratic polynomial equations, are established for the two fragility parameters (λ and ζ) based on data obtained from the 9 experimental design points. The surfaces are shown in Figure 5.6. Polynomial equations are used to establish these response surfaces, and the equations for a second degree surface are given in Equations (5.1) and (5.2). The results demonstrate that as adjusting one of the meta-variables to higher values while the other meta-variable is held constant, there is a consistent rise in the logarithmic mean value of fragility (z-axis). This increase indicates a more ductile system, which is better able to withstand strong motions.

$$\lambda(cm/s) = \exp(0 + 0.264\alpha_\epsilon + 0.457\alpha_V - 0.24\alpha_\epsilon^2 + 0.091\alpha_\epsilon\alpha_V - 0.276\alpha_V^2) \quad (5.1)$$

$$\zeta = 0 + 0.016\alpha_\epsilon + 0.088\alpha_V \quad (5.2)$$

However, there comes a point where further increase in the same meta-variable no longer has any effect on the logarithmic mean value of fragility. This can be explained by the interference of controlling meta-variables. In other words, another meta-variable is already triggered. Interestingly, the second-degree response surface

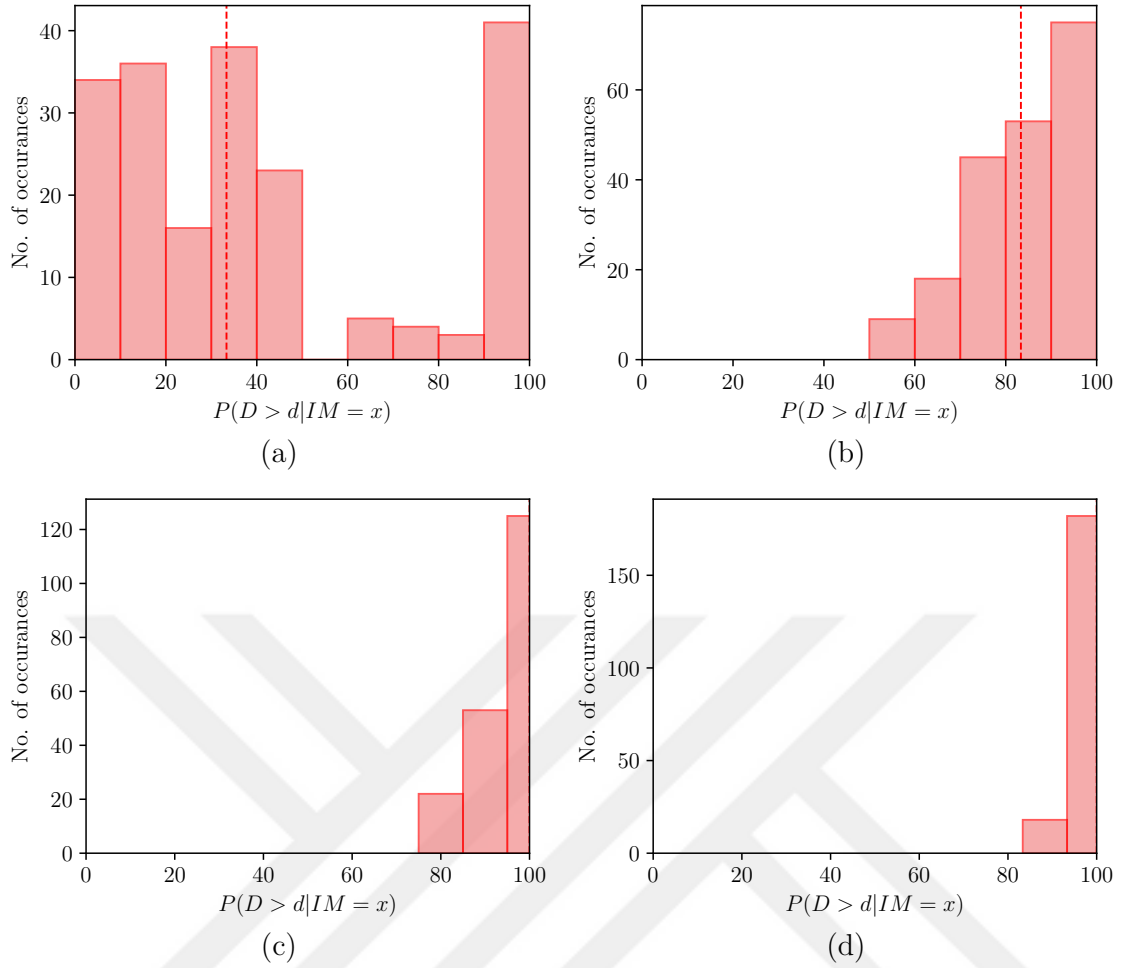


Figure 5.3 : Benchmark fragility model with histogram of probabilities of failure from Monte Carlo Simulation of (a) IM= 10 [cm/s], (b) IM= 20 [cm/s] (c) IM= 30 [cm/s], and (d) 40 [cm/s].

fails to detect this phenomenon. The reason behind its failure is the smoothness of the surface. Instead of displaying this zone as a plateau, which is expected for the mean fragility value, the second-degree response surface depicts it as a high-variance bell-shaped zone, which is not logical for the mean fragility value to decrease while increasing a meta-variable value. The experimental points residuals and benchmark points residuals induced by RSM approach is depicted in Figure 5.7 and Figure 5.8. In addition,

Furthermore, the probability of collapse is calculated at each intensity measure for a megaset of random limit-states combinations, as shown in Figure 5.9. A final fragility model developed from fitting a log-normal cumulative distribution function to the medians of probability damages of the random megaset, and is compared to the

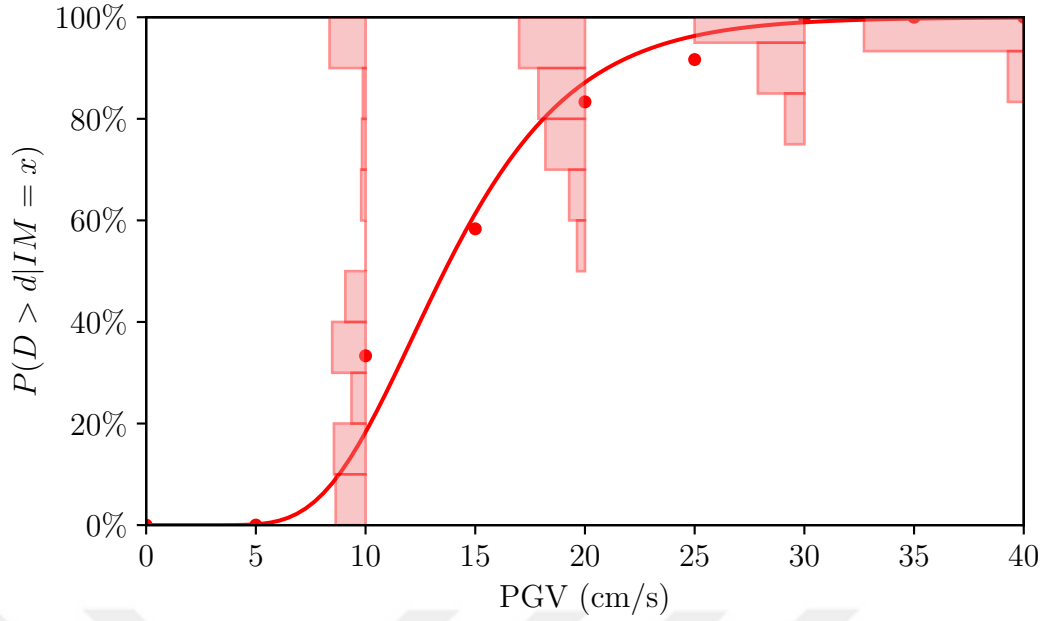


Figure 5.4 : Benchmark final fragility model developed from Monte Carlo Simulations.

benchmark model in Figure 5.10. The results of the comparison reveal that the RSM approach falls short in providing a reliable alternative, as the produced fragility model depicts significantly lower values than anticipated. Additionally, the intensity measure of median probability value falls below the expected range, indicating the model's inability to provide reliable predictions.

To study the effect of increasing polynomial degree on the results. Another surface of response is developed with third degree polynomial equation. In contrast to the quadratic surface, the third-degree response surface, displayed in Figure 5.11, does not suffer from the aforementioned limitation, inasmuch as it accurately depicts the phenomenon as a low variance bell-shaped curve. This difference in accuracy can be attributed to the more nonlinearity of the third-degree response surface.

However, this comes with a downside. The higher degree response surface might overfit the data and therefore might not be a good fit for predicting results outside of the range of experimental data. With an aim to thoroughly inspect this issue, a comparison is made between the accuracy of the results at each degree of the polynomial equation in Figure 5.12. While a high-degree polynomial may seem like the optimal choice for developing a response surface that accurately captures the results, it is not necessarily

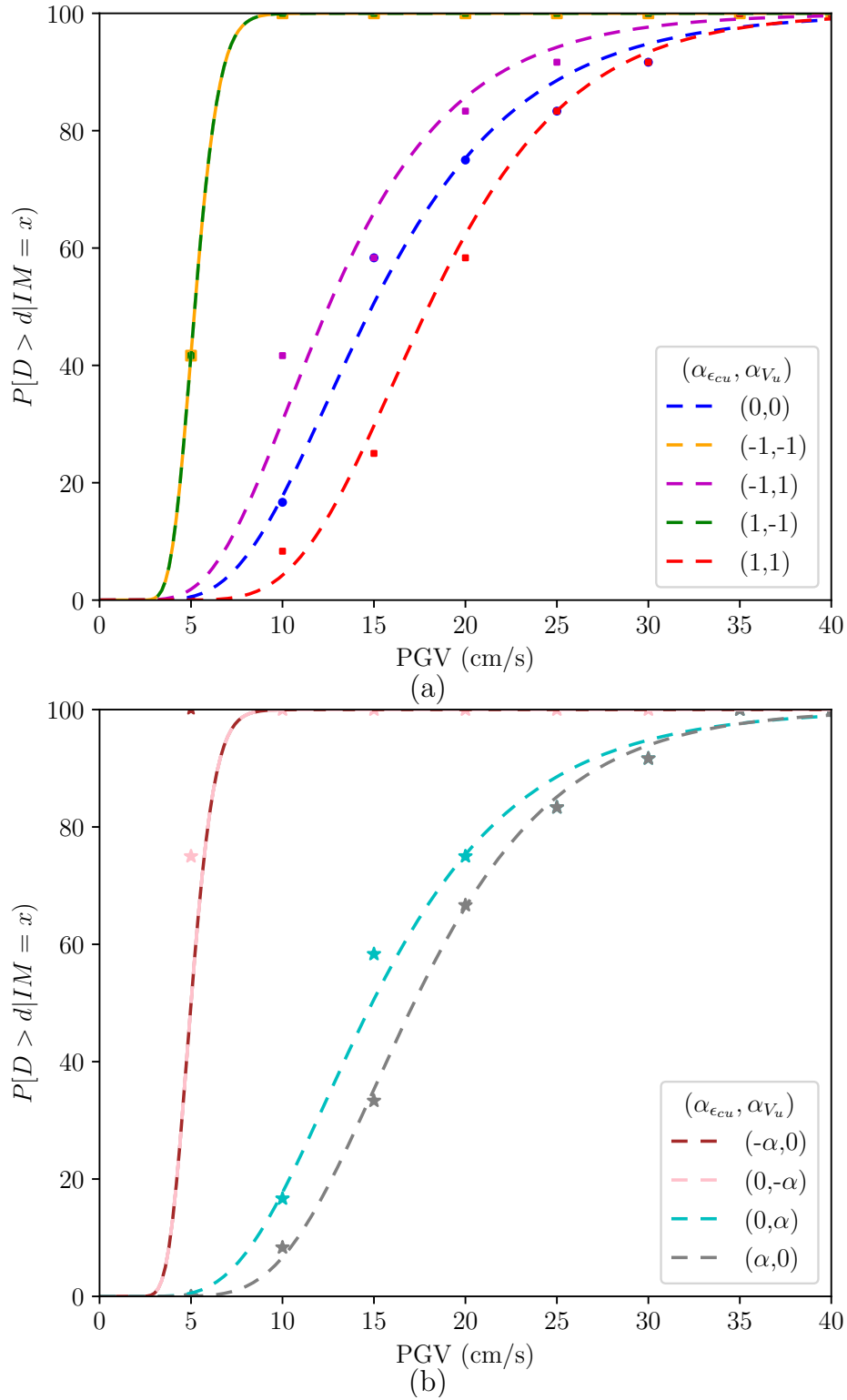
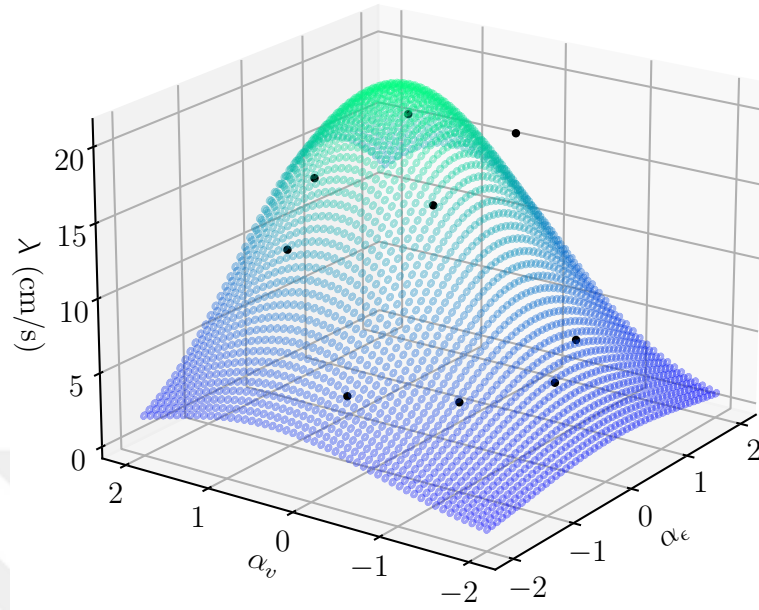
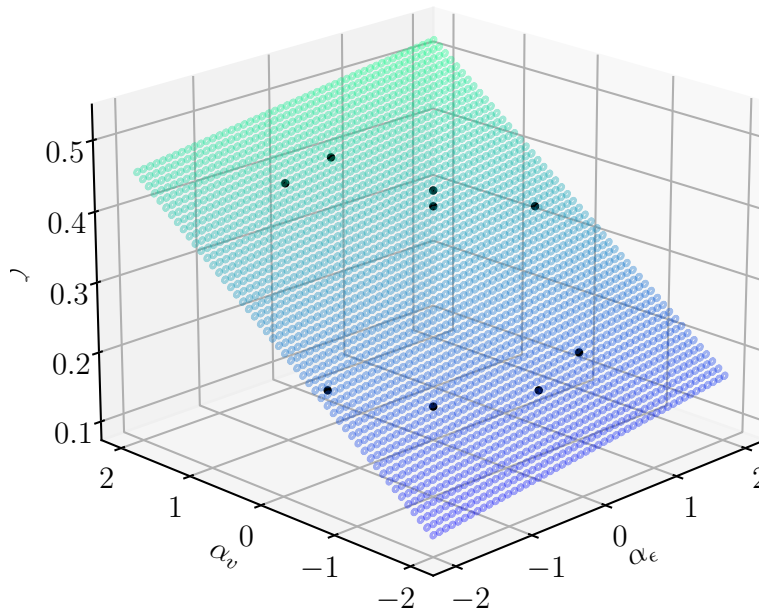


Figure 5.5 : Estimated fragility models for the (a) center point, axial points, (b) and the star points of the experimental design points of RSM technique.



(a)



(b)

Figure 5.6 : 2^{nd} degree polynomial response surface: Variations of predicted (a) logarithmic mean (b) logarithmic standard deviation of the fragility function.

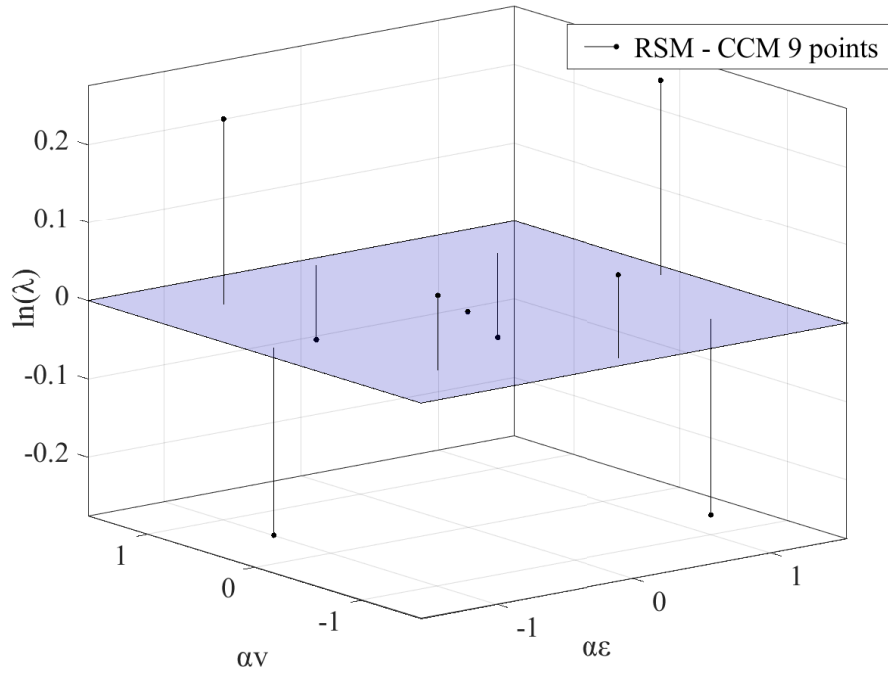


Figure 5.7 : 2^{nd} degree polynomial response surface: λ residuals of experimental design points.

the case. This is because although the root mean square error (RMSE) for the 9 points may be almost zero, the actual data points obtained from benchmark simulations do not align well with the surface. As the degree of the polynomial surface increases, the disparity between the surface and actual data points becomes quite noticeable.

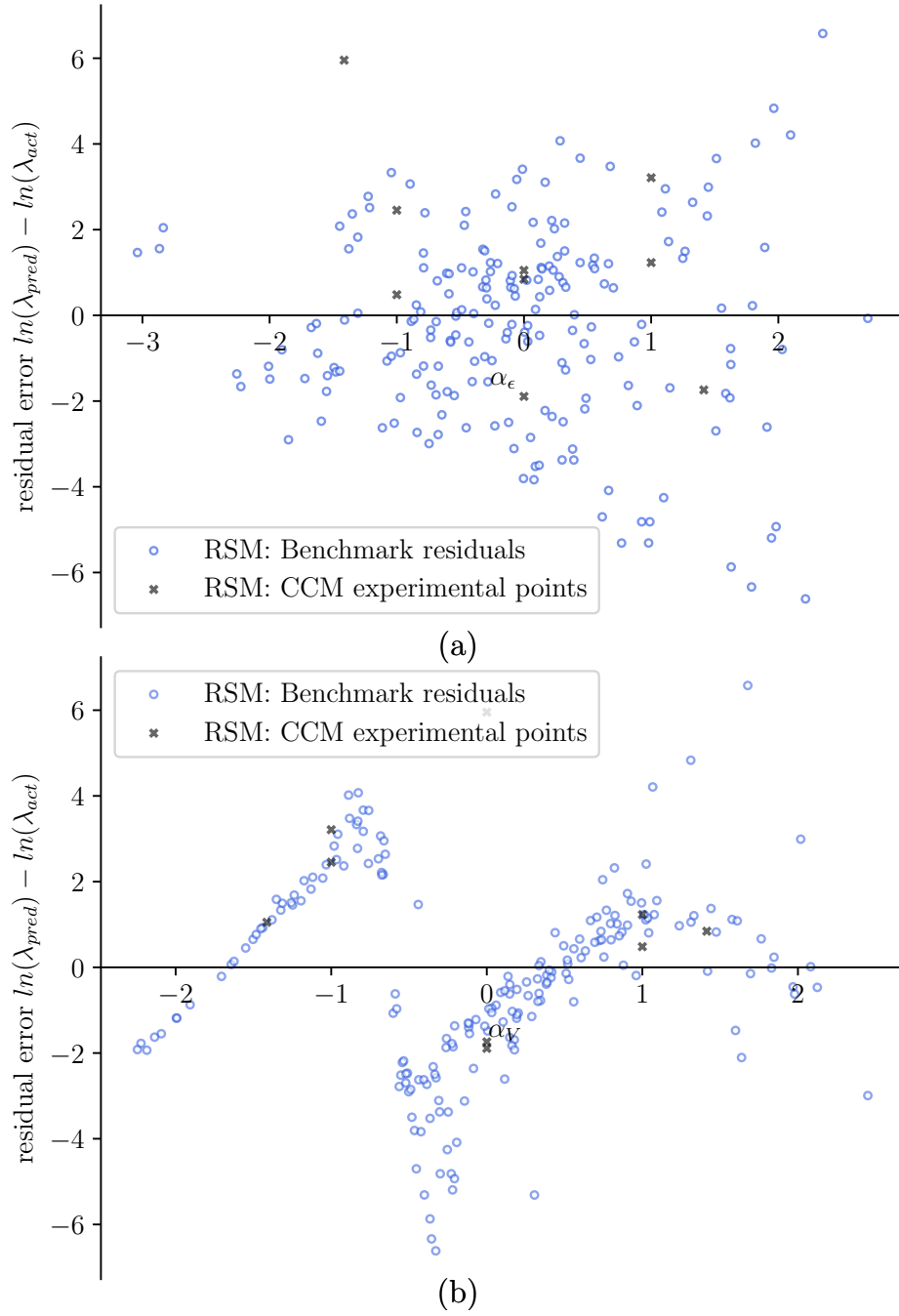


Figure 5.8 : Residual errors of response surface method produced at variations of (a) ultimate concrete strain capacity, and (b) ultimate shear capacity meta-variables.

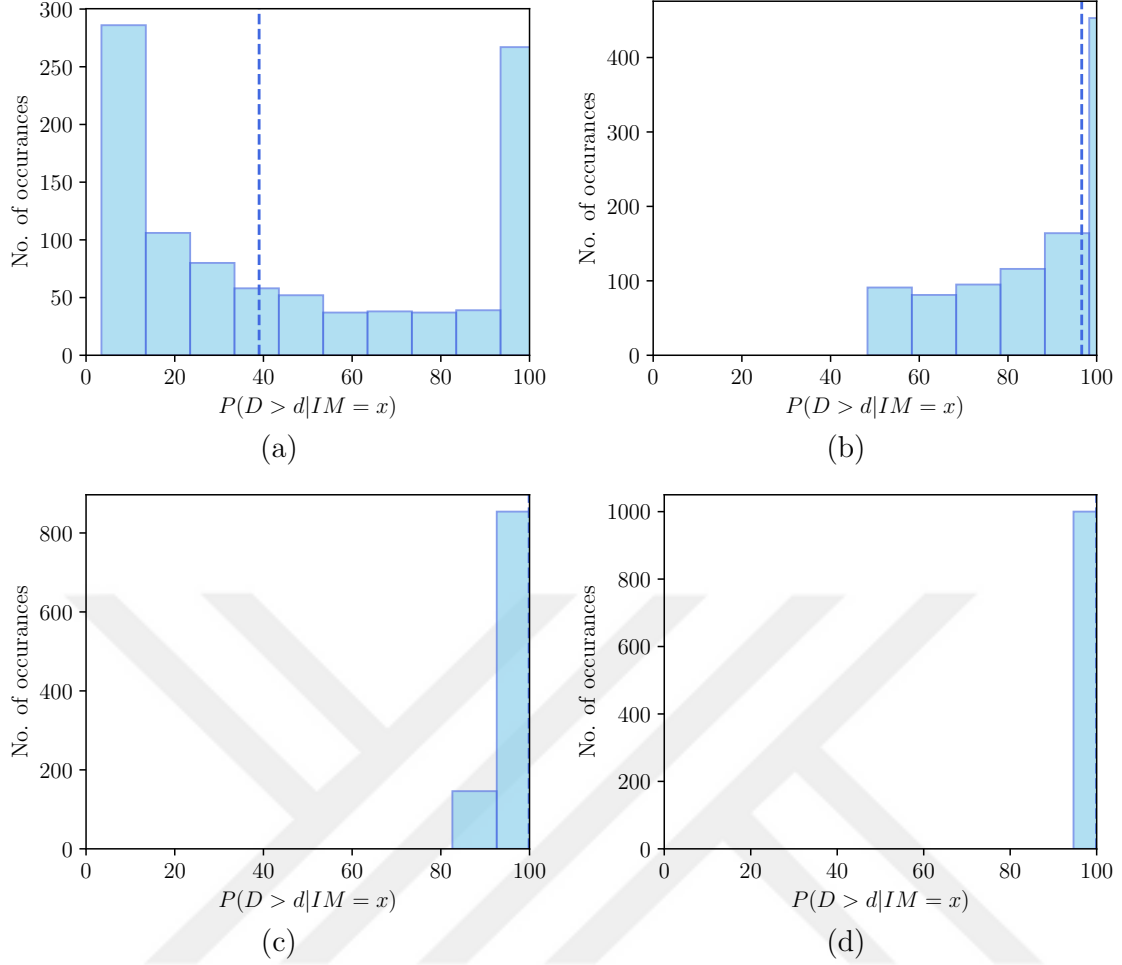


Figure 5.9 : Benchmark fragility model with histogram of probabilities of failure from RSM of (a) IM= 10 [cm/s], (b) IM= 20 [cm/s] (c) IM= 30 [cm/s], and (d) 40 [cm/s].

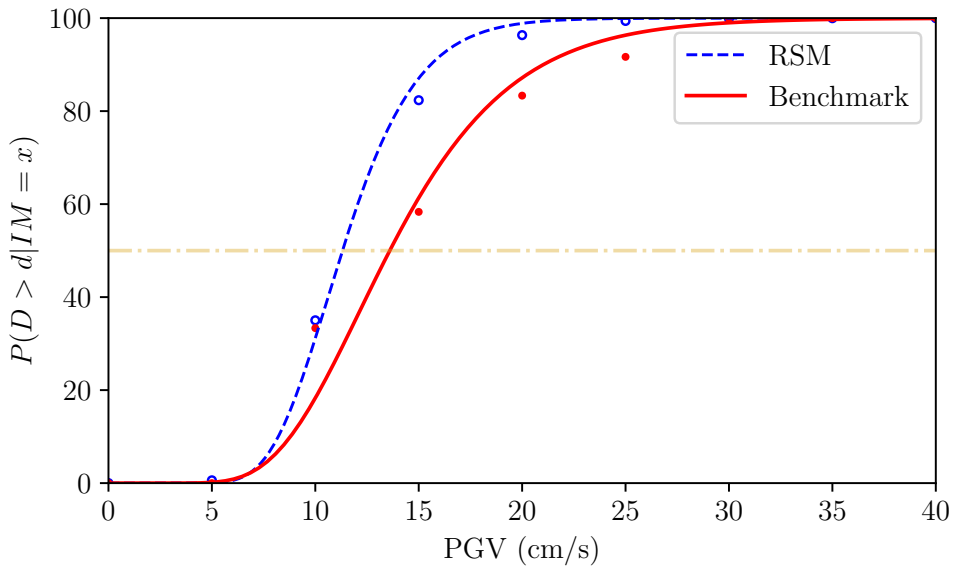


Figure 5.10 : Final fragility model produced by response surface method.

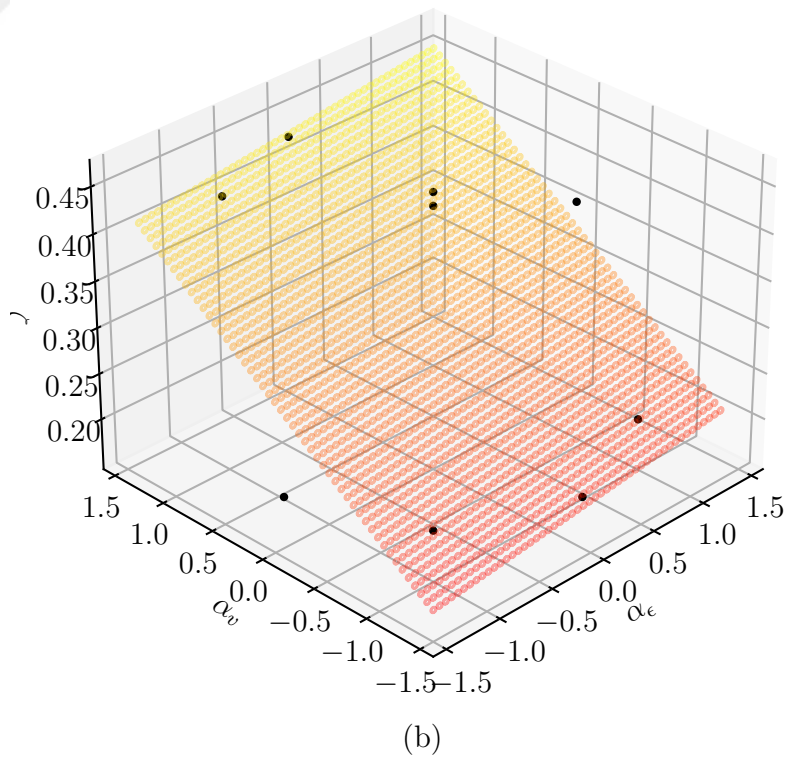
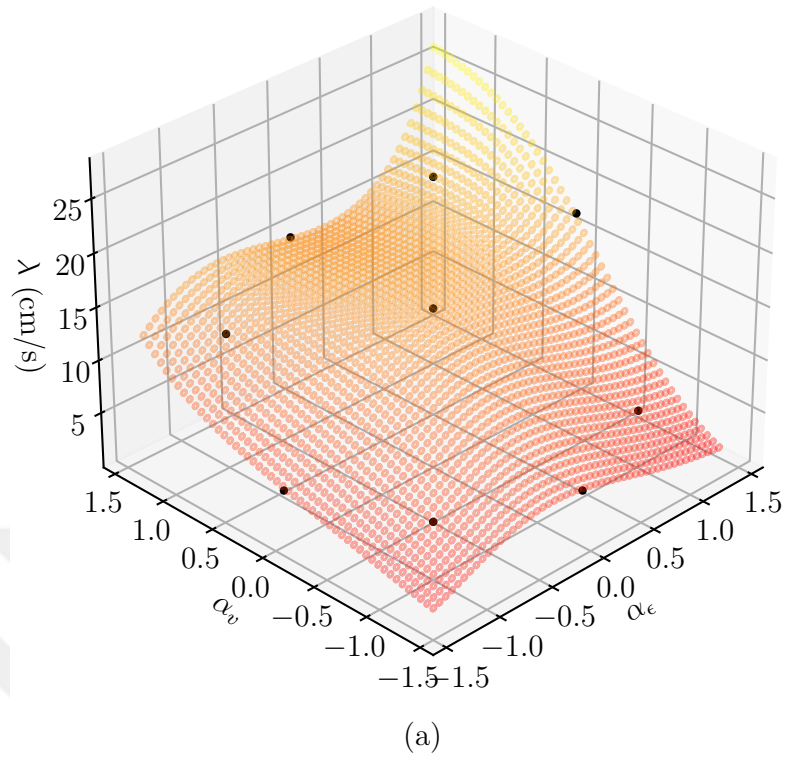


Figure 5.11 : 3^{rd} degree polynomial response surface: Variations of predicted (a) logarithmic mean (b) logarithmic standard deviation of the fragility function.

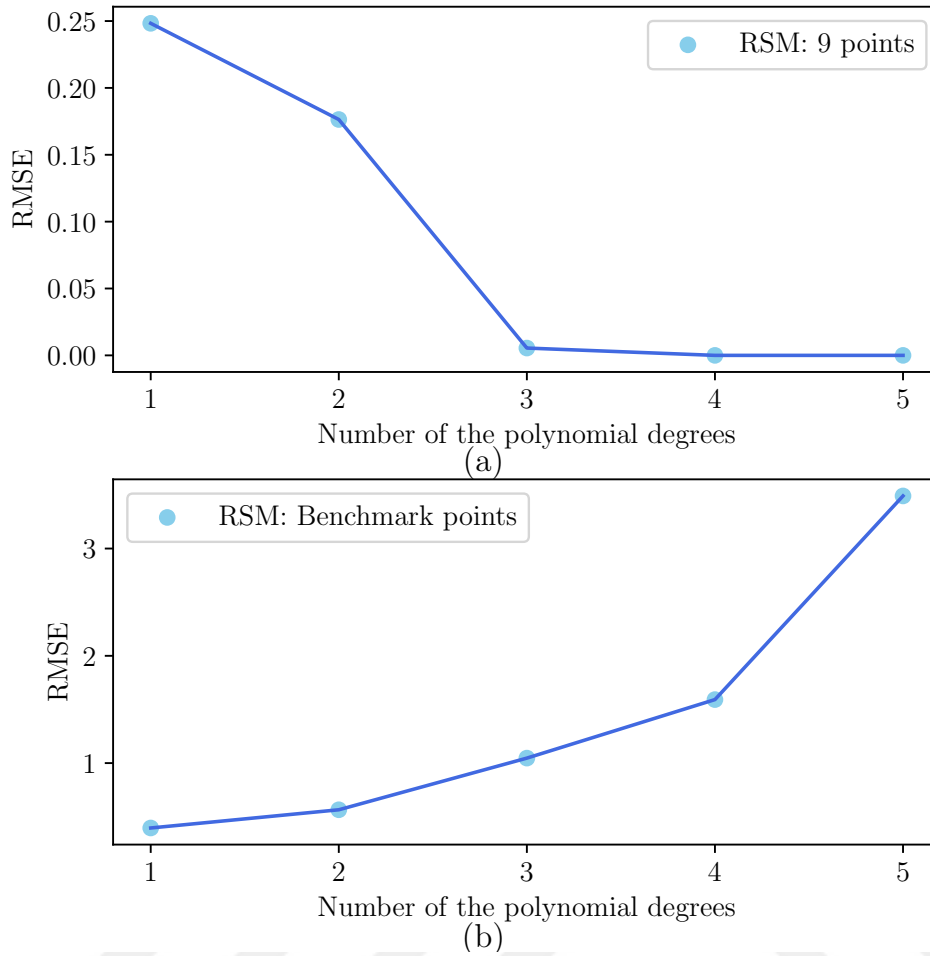


Figure 5.12 : Root mean square error for the response surface method versus degrees of equation at (a) the training points (b) the test points.

5.3 Results of Artificial Neural Networks Method

Initially, parametric analysis is performed to find the suitable configuration of neural networks that result in best performance. The neural networks undergo training using a 7-fold cross-validation approach, where one of the folds is reserved for validation purposes. This process is repeated iteratively, with the test fold changing to a different fold in each iteration until all folds have been used for validation. Subsequently, the average of the accuracy results is computed for each combination of the number of neurons and hidden layers. The findings of this analysis, depicted in Figure 5.13, reveals that the ideal configuration is 5 neurons in 3 hidden layers each with L2

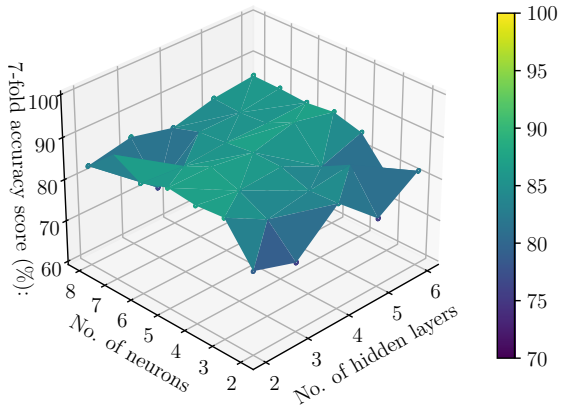
regularization parameter of 0.1%. This small regularization term encourages larger weights of the neurons; thus prevents underfitting.

The findings of fragility analysis presented in Figure 5.14 suggest that the ANN model is proficient in capturing the sophisticated nonlinear relationship between the input variables and the output response. This is likely attributed to the capability of ANNs in modelling highly nonlinear relationships between controlling meta-variables, which is challenging to handle using polynomial equations-based machine learning algorithms, such as RSM. It is observed that the log-mean surface shows a sudden transition at low shear capacity metavariable. This edge becomes sharper when increasing the ultimate concrete strain capacity metavariable.

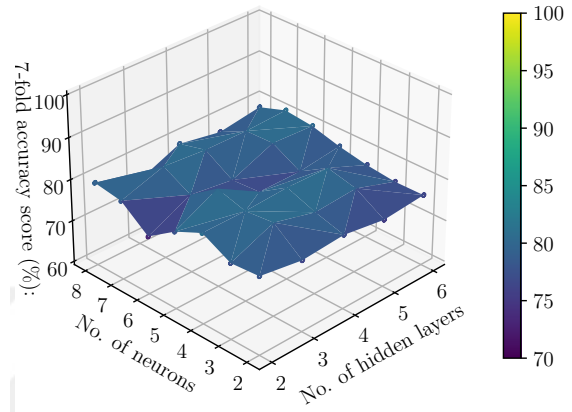
Moreover, the plateau zone detected by the ANN model is a crucial benefit in producing fragility function, since it represents a substantial range of limit state meta-variables where the likelihood of failure remains almost constant. In contrast, RSM techniques are unable to detect this plateau zone, since they are limited by the assumptions regarding the linear relationship between input and output variables. Therefore, the ANN approach outperforms RSM by providing a more accurate and comprehensive estimation of the fragility function parameters for any given combination of limit-state values.

The residuals produced by both the ANN and RSM models across all levels of intensity measures are displayed in Figure 5.15. The residuals are plotted as violin plots, where the width of the violin at each intensity measure represents the distribution of density of the residuals at that level. From the figure, it is clear that the ANN model produces much less residuals than the RSM model across all levels of intensity measures. This indicates that the ANN model is able to better capture the nonlinearities and uncertainties of the system, resulting in more accurate predictions of the fragility models. On the other hand, the RSM model produces larger residuals, due to smoothness of its curve that hindered its ability to capture the nonlinearities of the system.

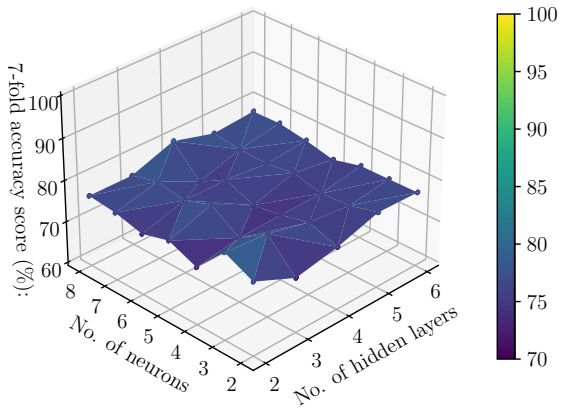
The final fragility model for the ANN approach is estimated using maximum likelihood estimation approach, and results are shown in Figure 5.16. In view of comparing



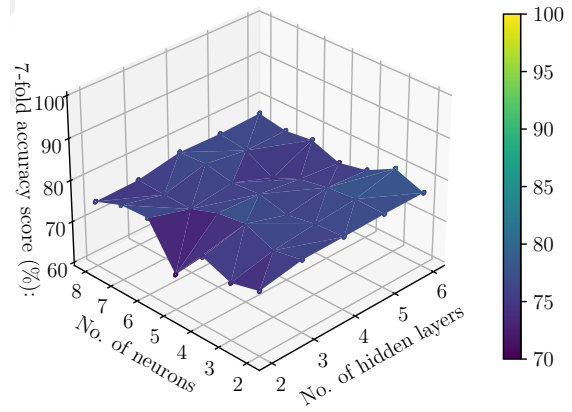
(a) L2 regularization $\alpha = 0.1 \%$



(b) L2 regularization $\alpha = 0.5 \%$



(c) L2 regularization $\alpha = 1.0 \%$



(d) L2 regularization $\alpha = 1.5 \%$

Figure 5.13 : Variation of neural networks parameters: Number of layers, number of neurons, and L2 regularization parameter α .

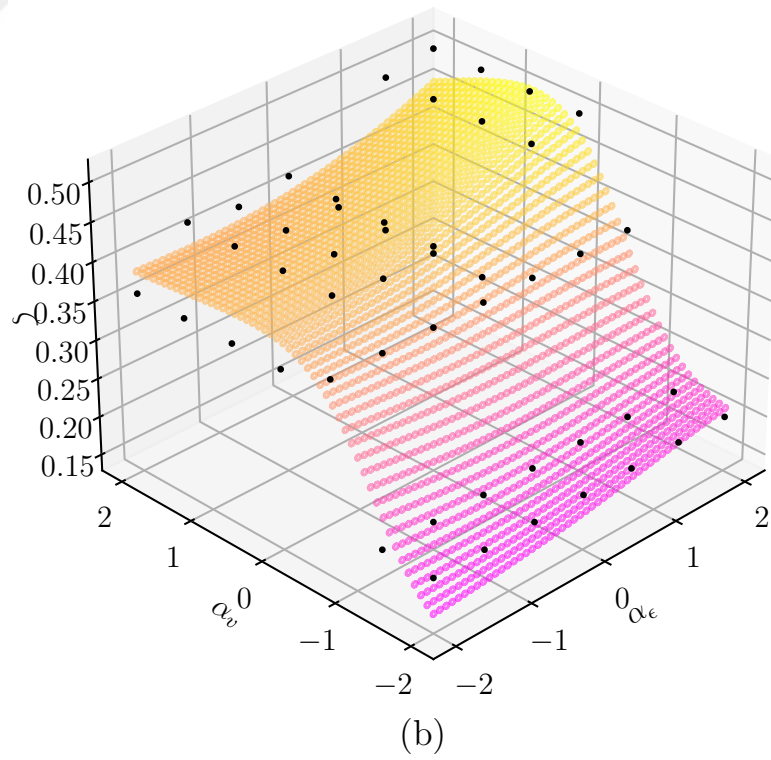
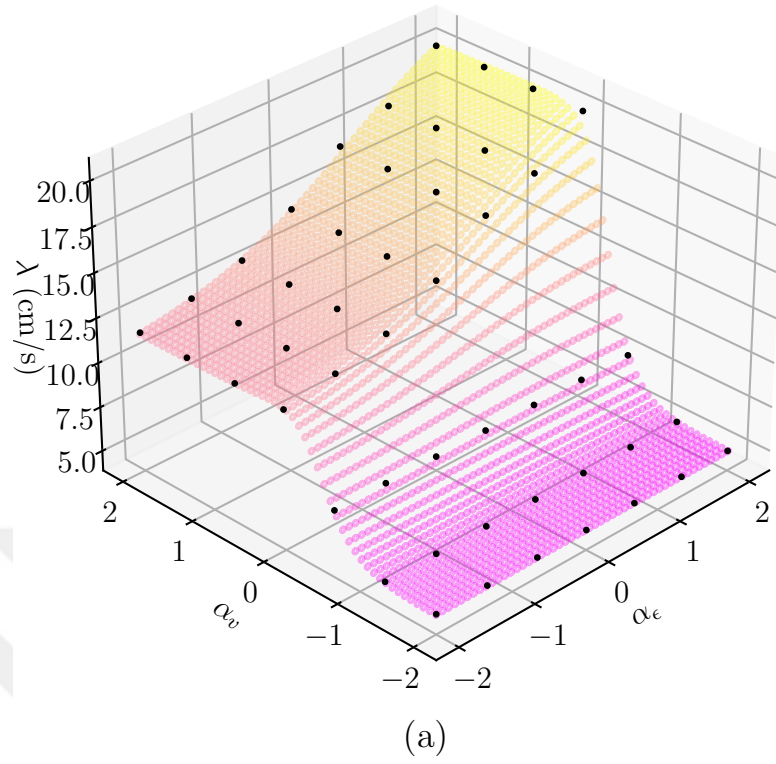


Figure 5.14 : Artificial neural networks: Variations of predicted (a) logarithmic mean (b) log standard deviation of the fragility function.

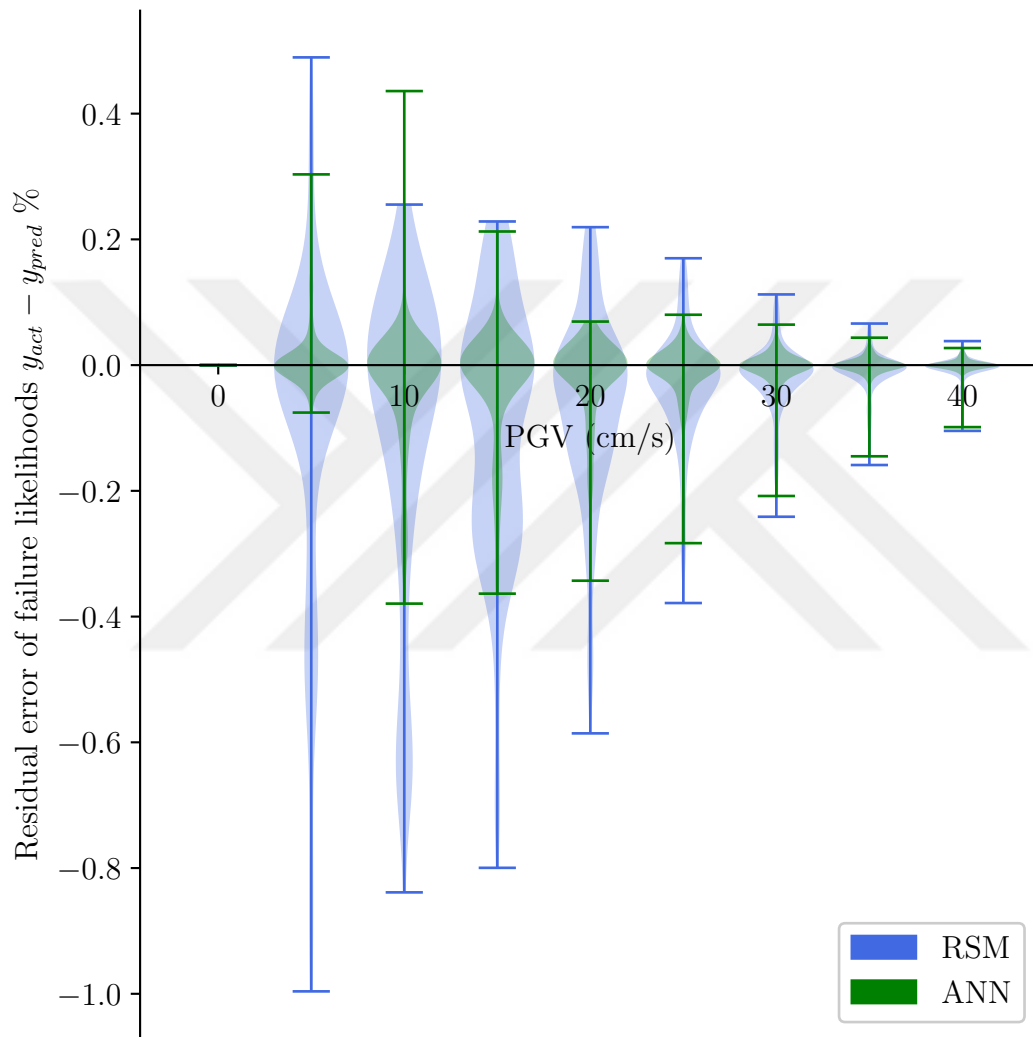


Figure 5.15 : Violin plots of residual errors produced at each intensity measure level for the machine learning algorithms.

between final fragility models produced by both ANN and RSM to the benchmark model generated by Monte Carlo simulations, which is visualized in Figure 5.17; it is clear that the ANN approach is capable of yielding a final fragility model that is nearly coincident to the benchmark model, which is an added advantage of using ANN over other machine learning techniques. The RSM approach, on the other hand, yields a model that is not too far from the benchmark model with likelihoods exceeding the benchmark. At the median probability of failure, the ANN model predicts intensity measure of value of 13.18 [cm/s], a quite close value to that of benchmark's which is equal to 13.60 [cm/s]. In comparison, the RSM model predicts a less value of 10.85 [cm/s]. It is evident that ANN outperforms RSM in this regard, which further supports the effectiveness of using ANN for fragility analysis of structures.

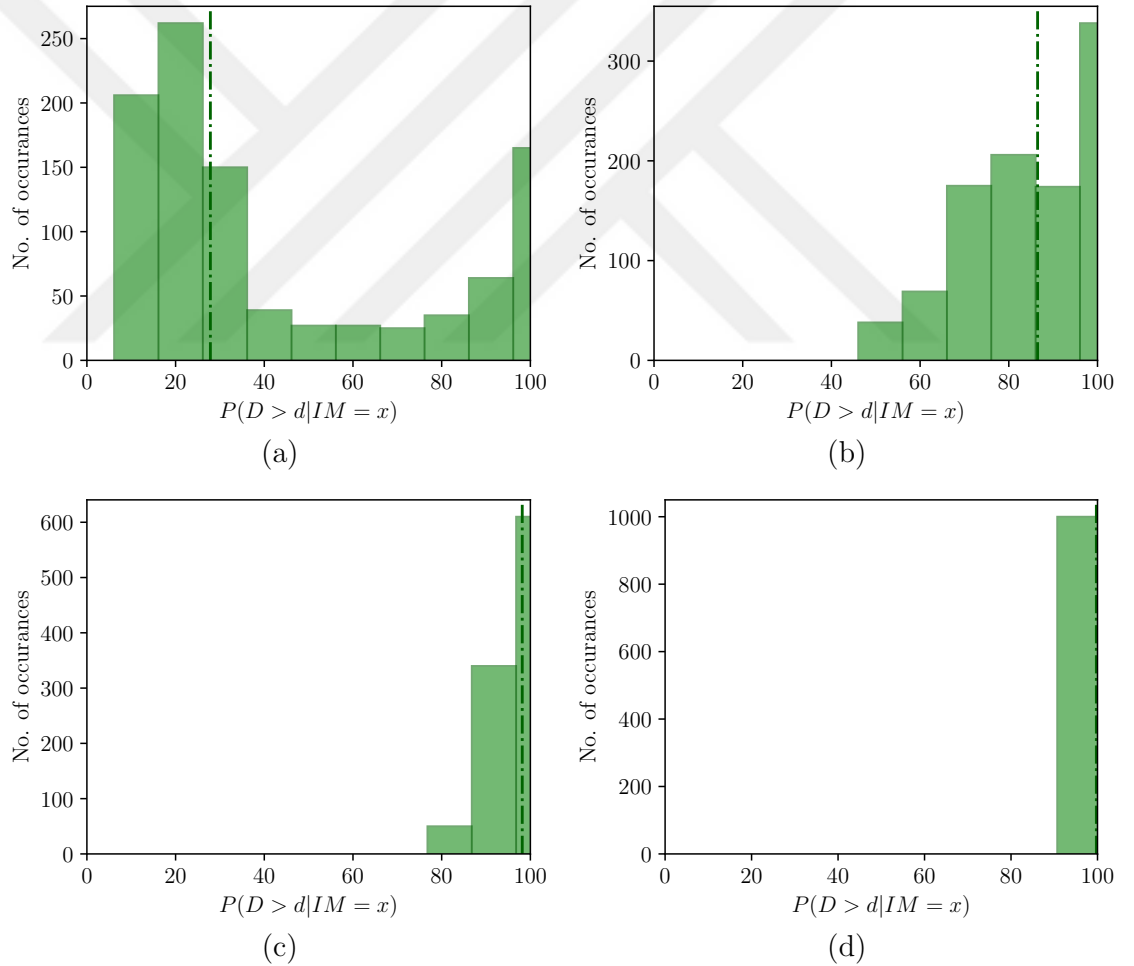


Figure 5.16 : Benchmark fragility model with histogram of probabilities of failure from ANN of (a) IM= 10 [cm/s], (b) IM= 20 [cm/s] (c) IM= 30 [cm/s], and (d) 40 [cm/s].

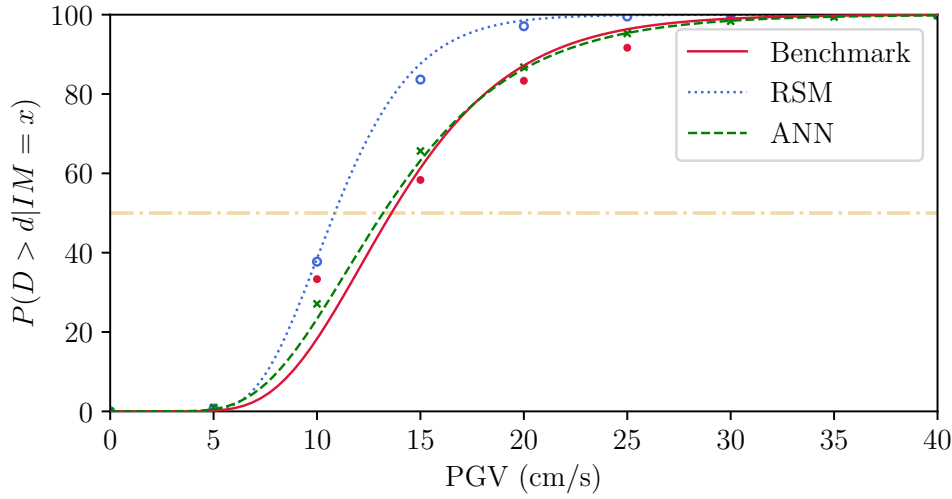


Figure 5.17 : Final fragility model yielded by ANN.

The time required to perform fragility analysis using a certain number of limit state combinations is compared to the simulation time, and the results are displayed in Figure 5.18. The development of the benchmark model lasts approximately 21 hours, while the RSM and ANN machine learning algorithms lasts around 32 minutes and 184 minutes, respectively. It is found that RSM is a more efficient alternative to the benchmark model, with a computational cost 5 times less than the ANN approach. However, the ANN approach is still proves to be a significantly time-saver when compared to the Monte Carlo simulation method.

5.4 Results of Case Study II

The residuals induced by maximum likelihood increases due to this high variance. In addition, there is a horizontal shift of the fragility models.

5.4.1 High variance case: Monte Carlo simulations

In the case study of high variance random variables model, the benchmark final fragility model is developed by the Monte Carlo simulations and shown in Figure 5.19. In the light of comparing between both cases shown in Figure 5.20, the higher variance fragility model features a lower log-mean fragility, which caused a left shift to the fragility model. However, the change in dispersion is considered insignificant.

5.4.2 High variance case: RSM algorithm

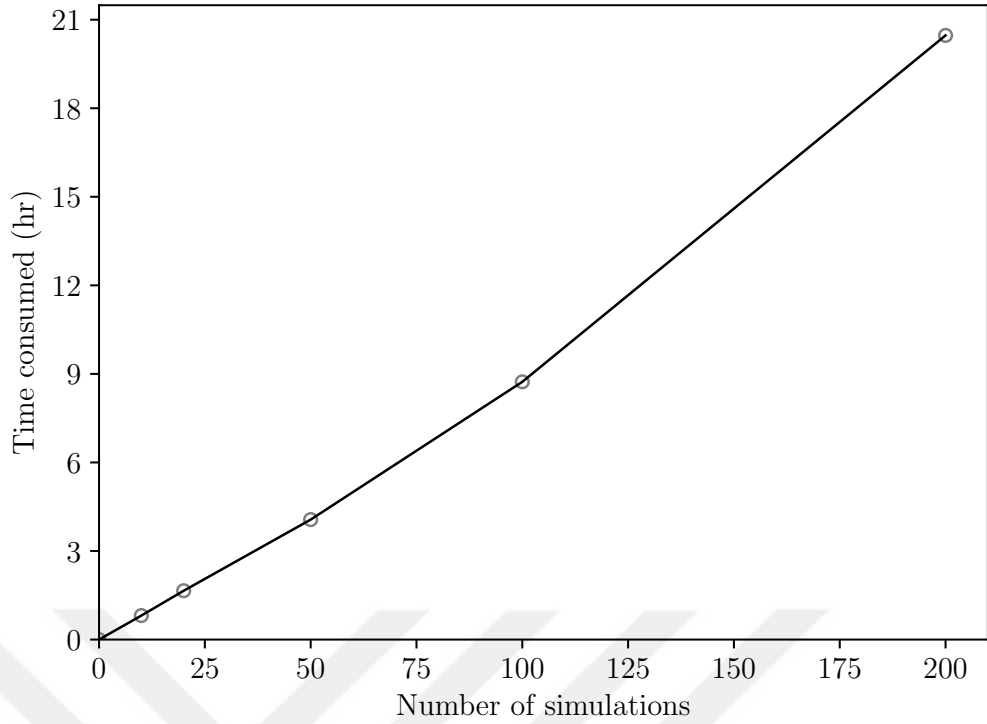


Figure 5.18 : Consumed time analysis for each number of simulations.

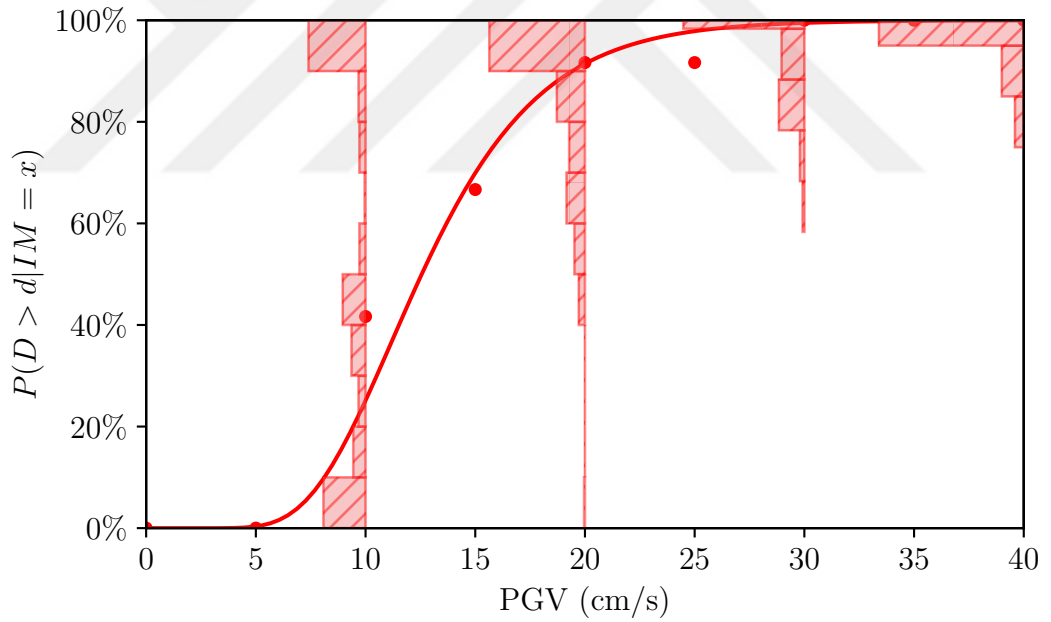


Figure 5.19 : Benchmark final fragility model for the high variance case.

The polynomial equations estimated in the RSM approach for the high variance case are presented in Equations (5.3) and (5.4). In the light of response surface produced by RSM approach in the high variance case study and depicted in Figure 5.21, the high variance increases the nonlinearity of the relationship between the meta-variables, which in turns amplifies the residuals in predicting the fragility parameters for points.

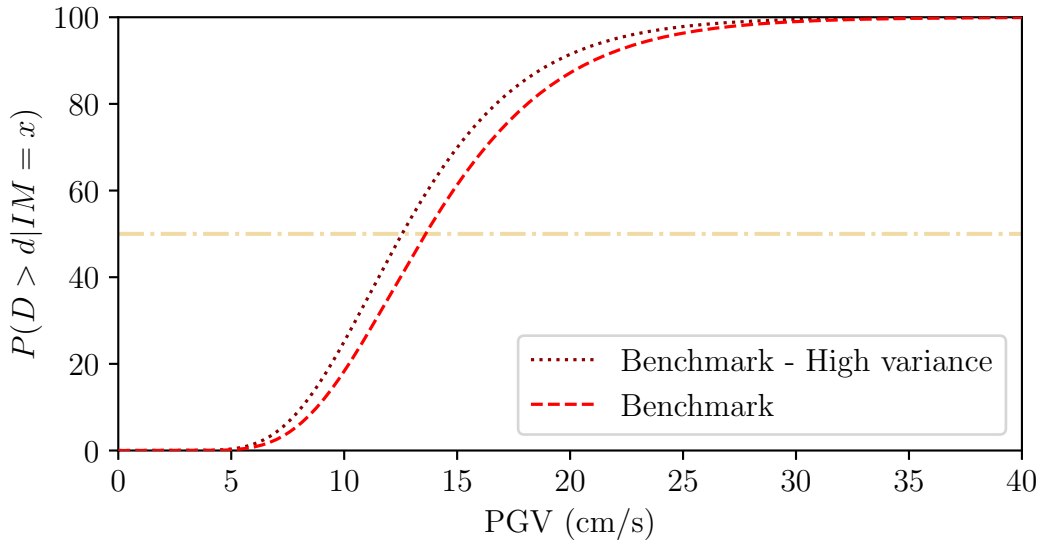


Figure 5.20 : Benchmark final fragility model for standard case and high variance case.

In a comparison between fragility model produced by RSM approach in standard case and high variance case; RSM approach for high variance case fails again to provide a better model to estimate the fragility model than the standard case as shown in Figure 5.22 and Figure 5.23.

Increasing degree of the polynomial equation is tested if it is capable of enhancing the performance of RSM approach, yet the results shows that the 3rd degree polynomial fails to provide valid predictions of responses in certain regions. Consequently, it is deemed unsuitable and discarded from the study.

$$\lambda^*([cm/s]) = \exp(0 + 0.235\alpha_\epsilon^* + 0.465\alpha_V^* - 0.098\alpha_\epsilon^{*2} + 0.127\alpha_\epsilon^*\alpha_V^* - 0.319\alpha_V^{*2}) \quad (5.3)$$

$$\zeta^* = 0 + 0.026\alpha_\epsilon^* + 0.088\alpha_V^* \quad (5.4)$$

5.4.3 High variance case: ANN algorithm

It is observed from the predictions of the grid training data for ANN approach, shown in Figure 5.24, that the high variance caused a sharper transition to a steep edge at low shear capacity meta-variable compared to that of standard case.

When considering the errors, Figure 5.25 proves that the fragility parameters prediction accuracy rate of ANN approach is still considered much better when compared to that of the RSM approach, even in high variance cases. Another benefit of using

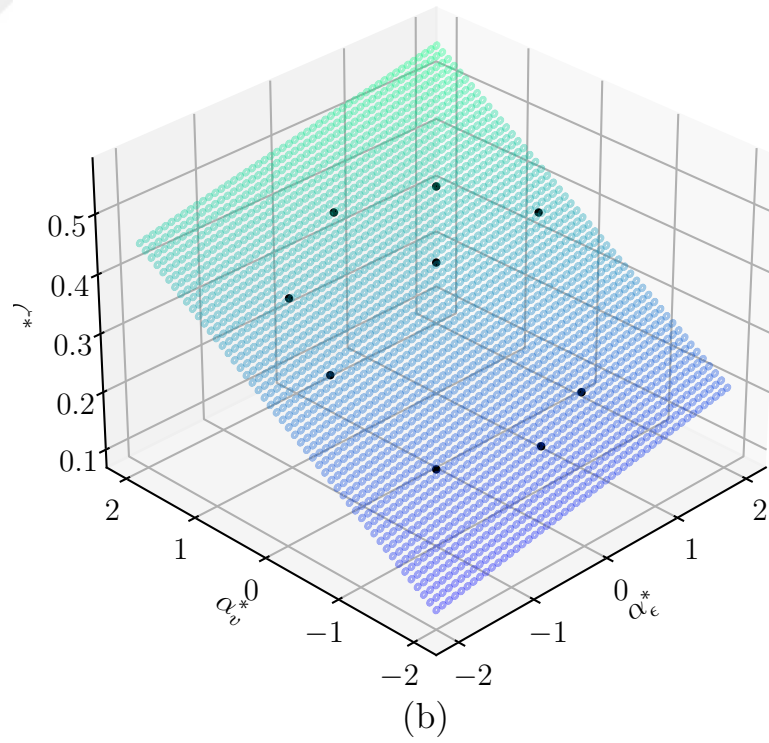
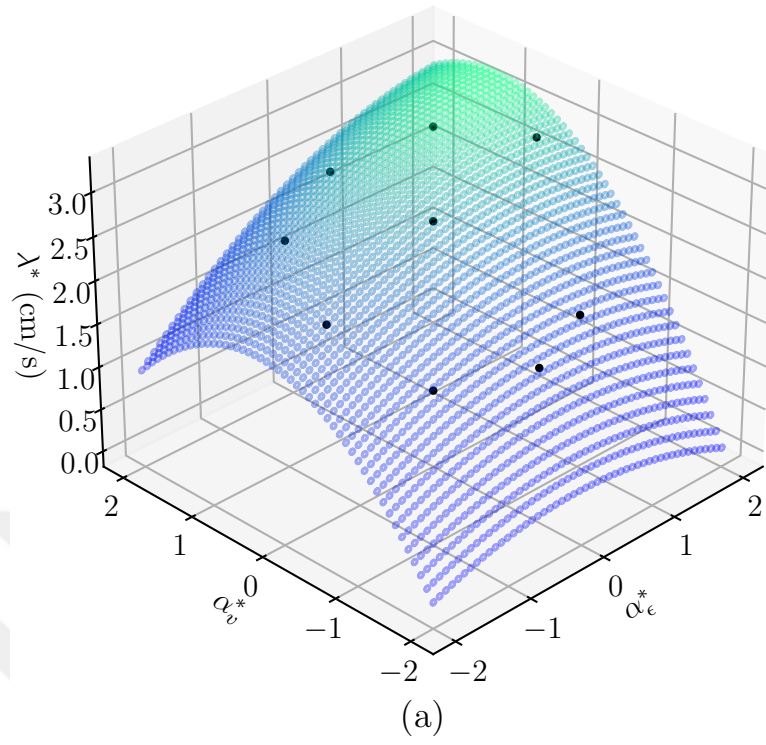


Figure 5.21 : High variance case: Response surfaces of fragility parameters: (a) logarithmic mean (λ), and (b) logarithmic standard deviation (ζ).

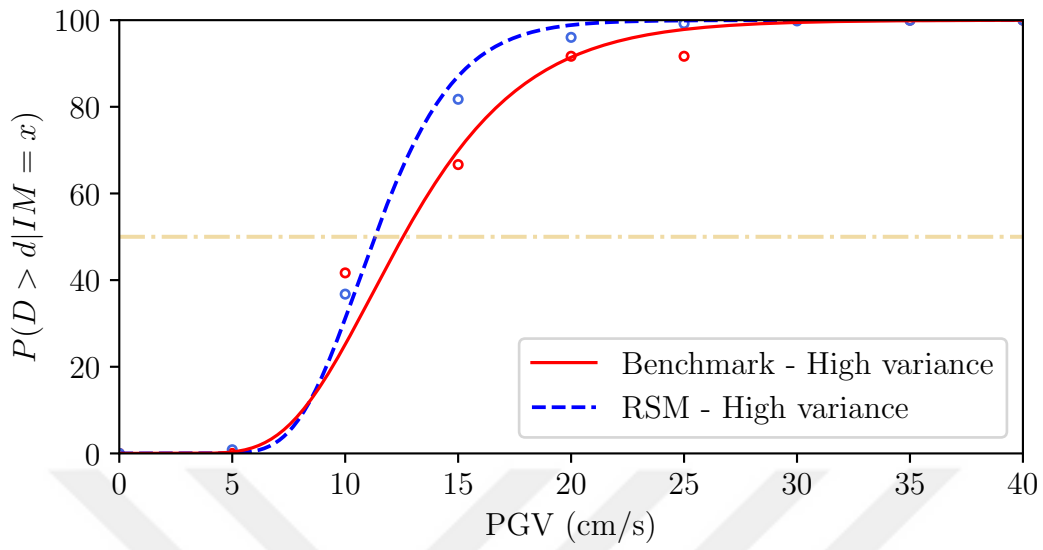


Figure 5.22 : RSM final fragility model for the high variance case.

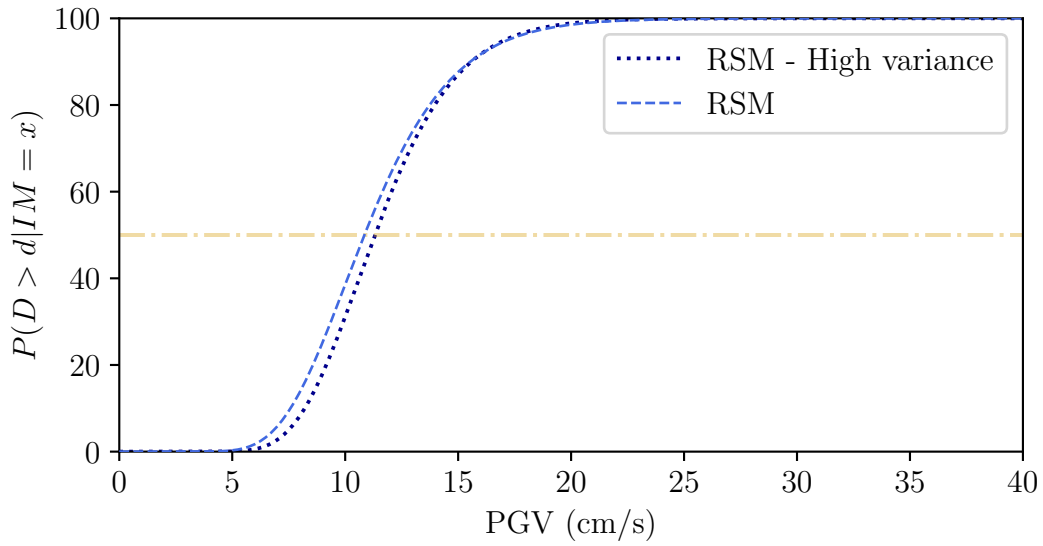


Figure 5.23 : RSM final fragility model for standard case and high variance case.

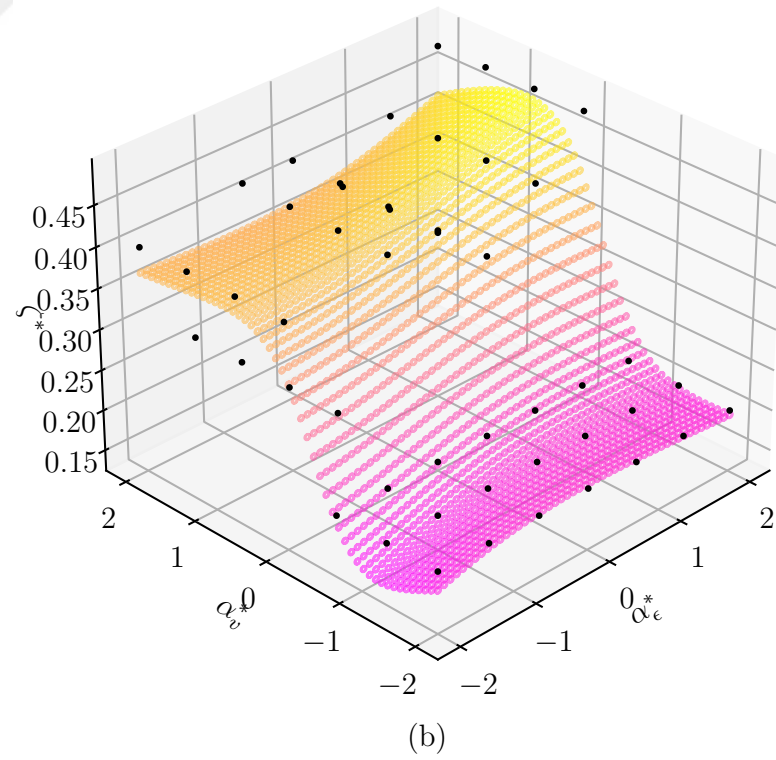
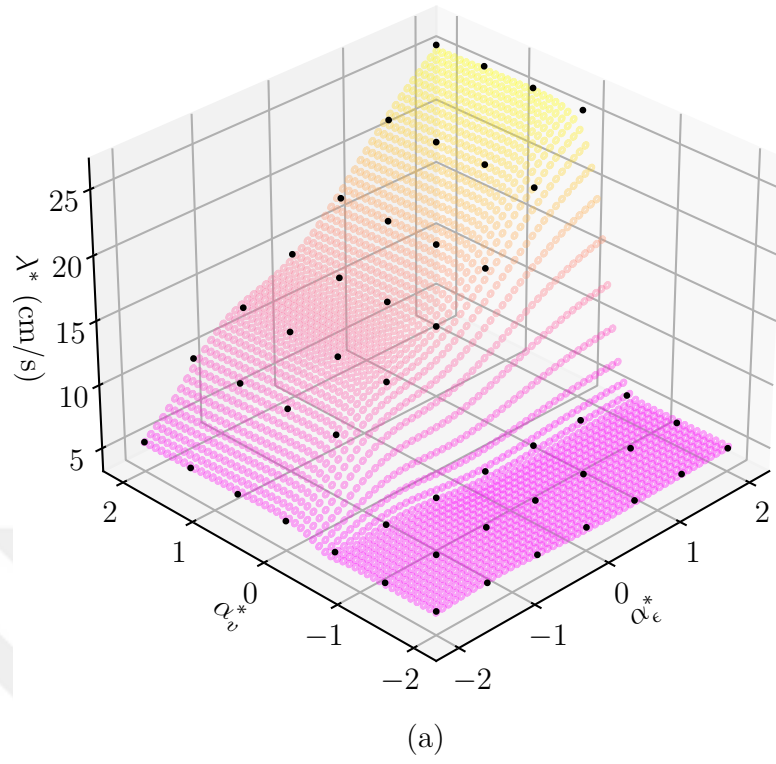


Figure 5.24 : High variance case: Response surfaces of ANN algorithm for fragility parameters: (a) logarithmic mean (λ), and (b) logarithmic standard deviation (ζ).

ANN approach is that it estimates a final fragility model that is more closely aligned with the benchmark model compared to the other machine learning algorithm, as shown in Figure 5.26. On the other side, The high variance of the random limit state variables causes a left shift and less spreading of the final fragility model, as shown in Figure 5.27.

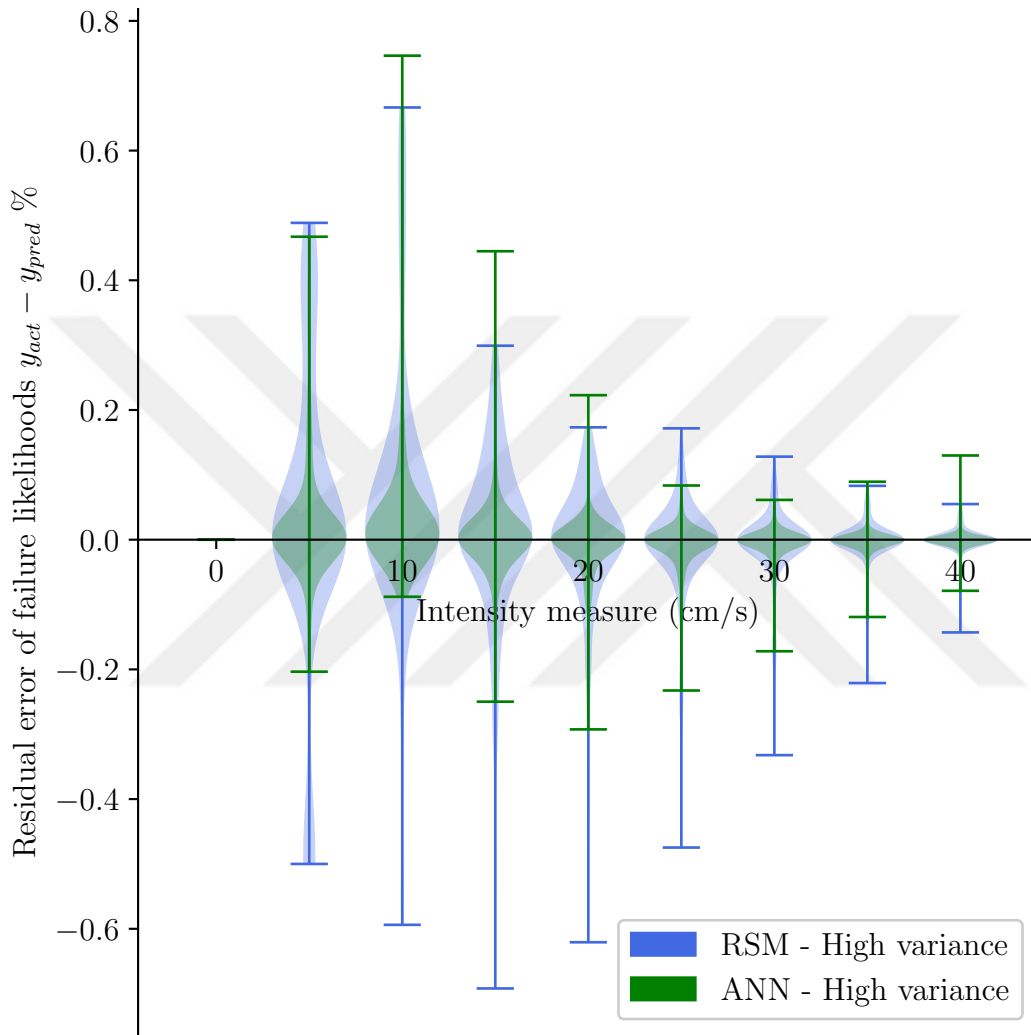


Figure 5.25 : High variance case: Violin plot of residuals for the machine learning algorithms.

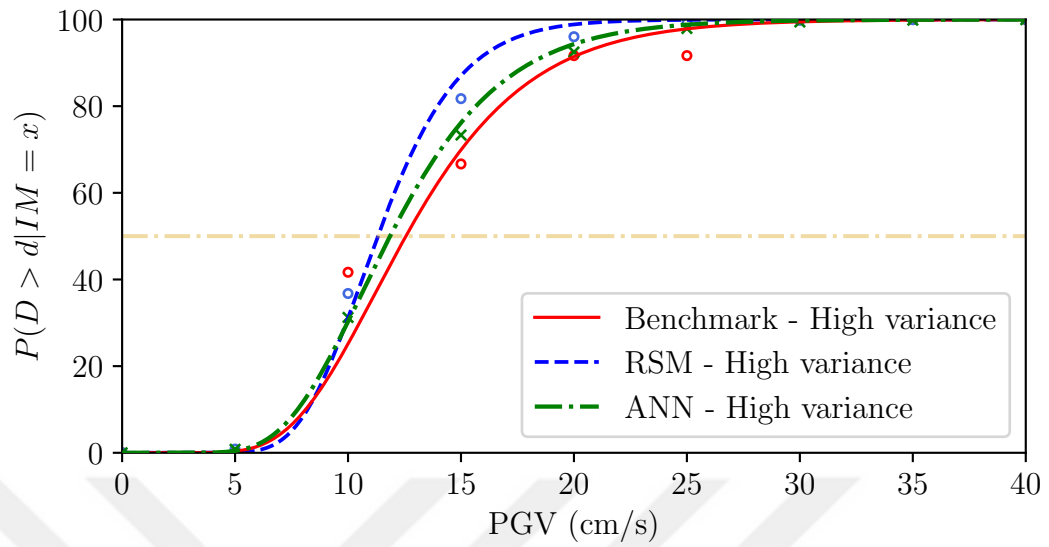


Figure 5.26 : High variance case: Final fragility models developed from machine learning algorithms.

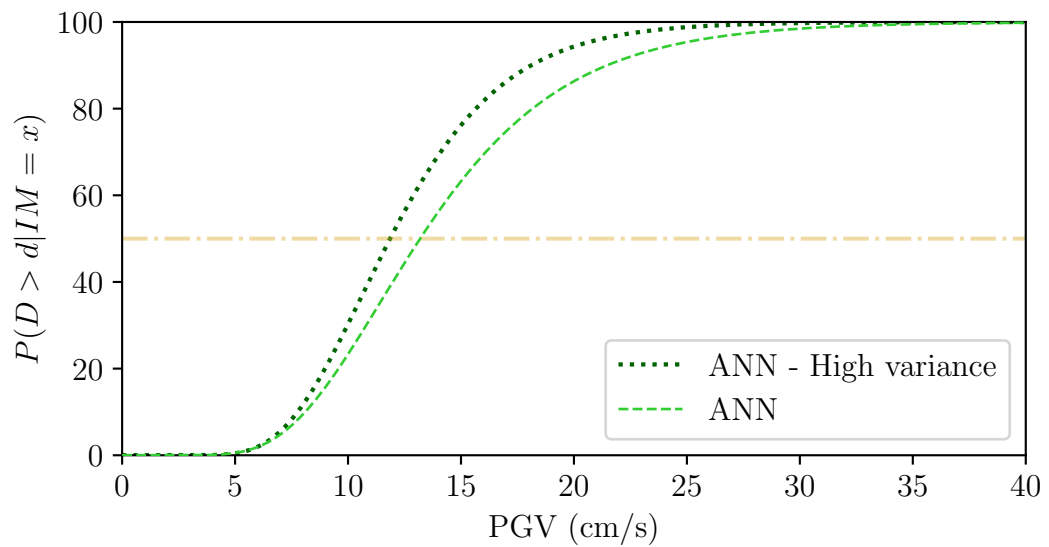


Figure 5.27 : ANN final fragility model for standard case and high variance case.



6. CONCLUSIONS AND RECOMMENDATIONS

6.1 Conclusions

One of the main advantages of using RSM is its simplicity and relatively shorter computation time compared to other techniques. However, this advantage comes at the expense of accuracy since RSM depends on fitted polynomial equations with a limited number of quadratic terms. This causes RSM to fail to capture the nonlinearity of the relationship between limit states. Specifically, RSM has difficulty detecting the transition in between the regions where different limit states trigger the structural failure. Depending on which limit state triggers failure, the effects of changing one of the meta-variables is canceled out. This issue becomes particularly visible when the RSM approach displays the interfering limit state zone as a smooth bell-shaped curve when it should have been a plateau zone with a sharp transition to a steep edge. In the end, the fragility model produced using RSM was observed to deviate from the expected benchmark, and the intensity measure corresponding to median capacity is found to be lower than that obtained for the benchmark model (i.e. using crude Monte Carlo simulation).

To address the limitations of RSM, the strategy of increasing the degree of the polynomial equation is tested to see if it can improve the accuracy of the fragility model. While this technique produces fewer residuals at the training data points, it is found that the residuals of test data points are magnified. In addition, the final fragility model does not show significant improvement. Furthermore, increasing the degree of the polynomial equation requires more training points, which in turn requires more computational power.

On the other hand, the use of artificial neural networks (ANN) in predicting the response of a structure is advantageous since it does not rely on any assumptions of polynomial equations. Instead, the fundamental idea is based on neurons, whose

weights change based on the training data. This leads to responses that approximate the expected response for the studied range of meta-variables. One of the most notable advantages of ANN is its ability to model the interface between regions of different dominant limit-state criteria. As a result, the resulting surface flattens -as desired- at very low meta-variable values, known as the plateau zone. Additionally, the ANN approach produces fewer residuals for most combinations of meta variable realizations, highlighting its ability to predict the likelihood of failure with a fairly good level of accuracy. Furthermore, the ANN approach has outperformed the RSM approach in producing a final fragility model that almost coincides with the benchmark model, which is a clear advantage of ANN over other machine learning techniques.

However, using ANN requires a larger dataset to train the neural networks than less advanced machine learning algorithms. Consequently, this leads to longer computational time. To investigate this issue, a sensitivity analysis is conducted. It is found that at least 49 random parameter value combinations (i.e. 7 meta-variable values each paired together) are needed to produce a robust estimate of the final fragility model. The computing time for the ANN approach was found to be equal to only 15% of that required for the benchmark model. This indicates the efficiency of the ANN approach as compared to the crude Monte Carlo. On the other hand, RSM requires only 3% of the computing time required for the benchmark model which indicates that it is the quickest approach. It should also be noted that for the case of more complex finite-element models this difference may be greater and may lead to ANN approach being infeasible.

To further investigate the benefits and limitations of machine learning algorithms in developing fragility models, an additional case study was conducted. In this case study, the standard deviation of each of the random variables representing material property uncertainty was doubled. This results in a higher level of nonlinearity in the relationship between the limit states and the response. Even though the ANN prediction accuracy decreases with respect to the benchmark model, ANN approach proved one more time that it can be considered as a better alternative to Monte Carlo simulation method by providing less residuals and a more reliable final fragility model compared to the model developed by RSM approach.

As a summary it can be concluded that optimal strategy for utilizing machine learning techniques is rather problem dependent. In the case when the structure of interest has a only a single controlling limit state, variation of fragility model parameters with the changing random variable values becomes smooth. In such cases, fragility model can be estimated efficiently even if only a simple second order quadratic surface is utilized. On the other hand, in the case of complex structures with multiple potential failure modes (e.g. flexural, shear, axial, torsion) response surface becomes highly nonlinear. Such nonlinearity cannot be captured effectively using the RSM. In such cases, ANN becomes a more reliable alternative to RSM even though it comes with additional computation effort. The gain in terms of the accuracy and robustness of the resulting fragility outweighs the increase in the corresponding computation cost.

6.2 Recommendations

Based on the limitations and findings discussed in this study, several recommendations for further research can be suggested. Initially, researchers can investigate the use of hybrid models that incorporate multiple algorithms to enhance accuracy and reduce computational cost. For example, including an algorithm with RSM approach to detect if a limit state is controlling the limit state function may lead to better performance. In addition, exploring the applicability of these techniques on more complex structures with greater degrees-of-freedom can provide insight into their efficiency and accuracy by considering real life problems. Moreover, it may be worthwhile to improve the effectiveness of the Response Surface Method approach by integrating an algorithm that improves the performance of the incremental dynamic analysis which is the most computationally intensive part of the entire process. Using a smart algorithm, the suitable range of ground intensities can be estimated prior to performing the dynamic analysis so as not to waste computational time for extreme intensity levels that are unnecessary. These research questions can contribute to overcoming the limitations of this study and enhancing our understanding of this topic.



REFERENCES

- Abiodun, O. I., Jantan, A., Omolara, A. E., Dada, K. V., Mohamed, N. A., and Arshad, H.** (2018). State-of-the-art in artificial neural network applications: A survey. *Heliyon*, 4(11):e00938.
- Ait-Amir, B., Pougnet, P., and El Hami, A.** (2015). Meta-Model Development. In *Embedded Mechatronic Systems 2*, pages 151–179. Elsevier.
- Akkar, S.** (2021a). Earthquake physical risk/loss assessment models and applications: A case study on content loss modeling conditioned on building damage. In *Advances in assessment and modeling of earthquake loss*, pages 223–237. Springer International Publishing Cham.
- Akkar, S.** (2021b). *Earthquake Physical Risk/Loss Assessment Models and Applications: A Case Study on Content Loss Modeling Conditioned on Building Damage*, pages 223–237. Springer International Publishing, Cham.
- Akkar, S., Sucuoğlu, H., and Yakut, A.** (2005). Displacement-based fragility functions for low- and mid-rise ordinary concrete buildings. *Earthquake Spectra*, 21(4):901–927.
- Anysz, H., Zbiciak, A., and Ibadov, N.** (2016). The influence of input data standardization method on prediction accuracy of artificial neural networks. *Procedia Engineering*, 153:66–70. XXV Polish – Russian – Slovak Seminar “Theoretical Foundation of Civil Engineering”.
- Baker, J. W.** (2015). Efficient analytical fragility function fitting using dynamic structural analysis. *Earthquake Spectra*, 31(1):579–599.
- Benítez, J. M., Castro, J. L., and Requena, I.** (1997). Are artificial neural networks black boxes? *IEEE Transactions on Neural Networks*, 8(5):1156–1164.

- Box, G. and Wilson, K.** (1951). On the experimental designs for exploring response surfaces. *Ann Math Stat*, 13:1–45.
- Braspenning, P. J., Thuijsman, F., and Weijters, A. J. M. M.** (1995). *Artificial neural networks: an introduction to ANN theory and practice*, volume 931. Springer Science & Business Media.
- Buratti, N., Ferracuti, B., and Savoia, M.** (2010). Response surface with random factors for seismic fragility of reinforced concrete frames. *Structural safety*, 32(1):42–51.
- Calvi, G. M., Pinho, R., Magenes, G., Bommer, J. J., Restrepo-Vélez, L. F., and Crowley, H.** (2006). Development of seismic vulnerability assessment methodologies over the past 30 years. *ISET journal of Earthquake Technology*, 43(3):75–104.
- CEN** (2004). *Eurocode 2: Design of concrete structures – Part 1-1: General rules and rules for buildings*. European Committee for Standardization.
- Choun, Y.-S. and Elnashai, A. S.** (2010). A simplified framework for probabilistic earthquake loss estimation. *Probabilistic Engineering Mechanics*, 25(4):355–364.
- Dal Zilio, Luca; Ampuero, J.-P.** (2023). Earthquake doublet in Turkey and Syria. *Communications Earth Environment*.
- Di-Sarno, L. and Elnashai, A. S.** (2021). *Seismic Fragility Relationships for Structures*, pages 189–222. Springer International Publishing.
- Elenas, A.** (2000). Correlation between seismic acceleration parameters and overall structural damage indices of buildings. *Soil Dynamics and Earthquake Engineering*, 20(1-4):93–100.
- Elnashai, A. and Di Sarno, L.** (2015). *Fundamentals of Earthquake Engineering: From Source to Fragility*. Wiley.
- Filippou, F., Popov, E., and Bertero, V.** (1983). *Effects of Bond Deterioration on Hysteretic Behavior of Reinforced Concrete Joints*. Earthquake Engineering Research Center, University of California.

- Gehl, P., Douglas, J., and Seyedi, D. M.** (2015). Influence of the number of dynamic analyses on the accuracy of structural response estimates. *Earthquake Spectra*, 31(1):97–113.
- Giovanis, D. G., Fragiadakis, M., and Papadopoulos, V.** (2016). Epistemic uncertainty assessment using incremental dynamic analysis and neural networks. *Bulletin of Earthquake Engineering*, 14:529–547.
- Hancılar, U., Şeşetyan, K., Çaktı, E., Şafak, E., Yenihayat, N., Malcioğlu, F. S., Dönmez, K., Tetik, T., and Süleyman, H.** (2023). Kahramanmaraş-Gaziantep Türkiye M7.7 Earthquake, 6 February 2023 Strong Ground Motion and Building Damage Estimations Preliminary Report (v6).
- Haykin, S.** (1999). *Neural Networks: A Comprehensive Foundation*. International edition. Prentice Hall.
- He, J. and Fu, Z.-F.** (2001). 12 - System identification using neural network. In *Modal Analysis*, pages 241–256. Elsevier.
- Huntington, D. and Lyrantzis, C.** (1998). Improvements to and limitations of latin hypercube sampling. *Probabilistic Engineering Mechanics*, 13(4):245–253.
- Hwang, H., Liu, J., and Chiu, Y.-H.** (2001). Seismic fragility analysis of highway bridges. *Mid-America Earthquake Center CD Release 01-06*.
- Iervolino, I., Fabbrocino, G., and Manfredi, G.** (2004). Fragility of standard industrial structures by a response surface based method. *Journal of Earthquake Engineering*, 8:927–945.
- Jia, D.-W. and Wu, Z.-Y.** (2021). Seismic fragility analysis of rc frame-shear wall structure under multidimensional performance limit state based on ensemble neural network. *Engineering Structures*, 246:112975.
- Jia, D.-W. and Wu, Z.-Y.** (2022a). Structural probabilistic seismic risk analysis and damage prediction based on artificial neural network. *Structures*, 41:982–996.

- Jia, D.-W. and Wu, Z.-Y.** (2022b). Structural probabilistic seismic risk analysis and damage prediction based on artificial neural network. In *Structures*, volume 41, pages 982–996. Elsevier.
- Kafali, C. and Grigoriu, M.** (2007). Seismic fragility analysis: Application to simple linear and nonlinear systems. *Earthquake Engineering & Structural Dynamics*, 36(13):1885–1900.
- Karsan, I. D. and Jirsa, J. O.** (1969). Behavior of concrete under compressive loadings. *Journal of the Structural Division*, 95(12):2543–2564.
- Kashani, M., Crewe, A., Alexander, N., and Lowes, L. N.** (2015). Reliably predicting the nonlinear cyclic response of rc bridge piers up to complete collapse. In *SECED 2015 Conference-Earthquake Risk and Engineering towards a Resilient World*.
- Kircher, C. A., Nassar, A. A., Kustu, O., and Holmes, W. T.** (1997). Development of building damage functions for earthquake loss estimation. *Earthquake spectra*, 13(4):663–682.
- LeNail, A.** (2019). Nn-svg: Publication-ready neural network architecture schematics. *Journal of Open Source Software*, 4(33):747.
- Liel, A., Haselton, C., Deierlein, G., and Baker, J.** (2009). Incorporating modeling uncertainties in the assessment of seismic collapse risk of buildings. *Structural Safety*, 31(2):197–211.
- Lu, X., Lu, X., Guan, H., and Ye, L.** (2013). Comparison and selection of ground motion intensity measures for seismic design of super high-rise buildings. *Advances in Structural Engineering*, 16(7):1249–1262.
- Lupoi, G., Franchin, P., Lupoi, A., and Pinto, P. E.** (2006). Seismic fragility analysis of structural systems. *Journal of Engineering Mechanics*, 132(4):385–395.
- Mahmoudi, J., Arjomand, M. A., Rezaei, M., and Mohammadi, H.** (2016). Predicting the earthquake magnitude using the multilayer perceptron neural network with two hidden layers. *Civil Engineering Journal*, 2.

- Mander, J. B., Priestley, M. J. N., and Park, R.** (1988). Theoretical stress-strain model for confined concrete. *Journal of Structural Engineering*, 114(8):1804–1826.
- Mazzoni, S., McKenna, F., Scott, M. H., Fenves, G. L., and others** (2006). OpenSEES command language manual. *Pacific Earthquake Engineering Research (PEER) Center*, 264(1):137–158.
- McGuire, R. K.** (2008). Probabilistic seismic hazard analysis: Early history. *Earthquake Engineering & Structural Dynamics*, 37(3):329–338.
- Monteiro, R.** (2016). Sampling based numerical seismic assessment of continuous span rc bridges. *Engineering Structures*, 118:407–420.
- Montgomery, D.** (2012). *Design and Analysis of Experiments, 8th Edition*. John Wiley & Sons, Incorporated.
- Myers, R. H., Montgomery, D. C., and Anderson-Cook, C. M.** (2009). *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*. John Wiley & Sons.
- Parsons, T.** (2004). Recalculated probability of $M \geq 7$ earthquakes beneath the Sea of Marmara, Turkey. *Journal of Geophysical Research: Solid Earth*, 109(B5).
- Paulay, T. and Priestley, M. N.** (1992). *Seismic design of reinforced concrete and masonry buildings*, volume 768. Wiley New York.
- Pinto, P., Giannini, R., and Franchin, P.** (2004). *Seismic Reliability Analysis of Structures*. IUSS Press: Pavia, Italy.
- Pitilakis, K., Crowley, H., and Kaynia, A. M.** (2014). Syner-g: typology definition and fragility functions for physical elements at seismic risk. *Geotechnical, Geological and Earthquake Engineering*, 27:1–28.
- Popovics, S.** (1973). A numerical approach to the complete stress-strain curve of concrete. *Cement and concrete research*, 3(5):583–599.

- Porter, K., Farokhnia, K., Cho, I., Grant, D., Jaiswal, K., Wald, D., and Noh, H.** (2012). Global vulnerability estimation methods for the global earthquake model. In *15th World Conference on Earthquake Engineering, Lisbon, Portugal*, pages 24–28.
- Radu, A. and Grigoriu, M.** (2018). An earthquake-source-based metric for seismic fragility analysis. *Bulletin of Earthquake Engineering*, 16:3771–3789.
- Rafiq, M., Bugmann, G., and Easterbrook, D.** (2001). Neural network design for engineering applications. *Computers & Structures*, 79(17):1541–1552.
- Rasheed, A., Usman, M., Zain, M., and Iqbal, N.** (2022). Machine learning-based fragility assessment of reinforced concrete buildings. *Computational Intelligence and Neuroscience*, 2022.
- Salkhordeh, M., Govahi, E., and Mirtaheri, M.** (2021). Seismic fragility evaluation of various mitigation strategies proposed for bridge piers. *Structures*, 33:1892–1905.
- Scott, M. H. and Fenves, G. L.** (2006). Plastic hinge integration methods for force-based beam–column elements. *Journal of Structural Engineering*, 132(2):244–252.
- Segura, R. L., Padgett, J. E., and Paultre, P.** (2019). Polynomial response surface-based seismic fragility assessment of concrete gravity dams. In *12th Canadian Conference on Earthquake Engineering, June*, pages 1–8.
- Shinozuka, M., Feng, M. Q., Lee, J., and Naganuma, T.** (2000). Statistical analysis of fragility curves. *Journal of engineering mechanics*, 126(12):1224–1231.
- Silva, V., Crowley, H., and Bazzurro, P.** (2016). Exploring risk-targeted hazard maps for europe. *Earthquake Spectra*, 32(2):1165–1186.
- TEC Turkish Earthquake Code** (2018). *Specifications for Building Design under Earthquake Effects*. Number 30364.
- Thorne, B. M. and Giesen, J. M.** (2000). *Statistics for the behavioral sciences*. McGraw-Hill Humanities, Social Sciences & World Languages.

- Vamvatsikos, D.** and **Cornell, C. A.** (2002). Incremental dynamic analysis. *Earthquake Engineering & Structural Dynamics*, 31(3):491–514.
- Vamvatsikos, D.** and **Cornell, C. A.** (2004). Applied incremental dynamic analysis. *Earthquake spectra*, 20(2):523–553.
- Vazirizade, S. M., Nozhati, S., and Zadeh, M. A.** (2017). Seismic reliability assessment of structures using artificial neural network. *Journal of Building Engineering*, 11:230–235.
- Vellido, A., Lisboa, P. J., and Vaughan, J.** (1999). Neural networks in business: a survey of applications (1992–1998). *Expert Systems with applications*, 17(1):51–70.
- Wang, Z., Zentner, I., Pedroni, N., and Zio, E.** (2017). Adaptive artificial neural networks for seismic fragility analysis. In *2017 2nd international conference on system reliability and safety (ICSRS)*, pages 414–420. IEEE.
- World Bank** (2023). Global Rapid Post-Disaster Damage Estimation (GRADE) Report: February 6, 2023 Kahramanmaraş Earthquakes - Türkiye Report. *Washington DC: World Bank Publications*.
- Xiao, Y., Ye, K., and He, W.** (2020). An improved response surface method for fragility analysis of base-isolated structures considering the correlation of seismic demands on structural components. *Bulletin of Earthquake Engineering*, 18:4039–4059.
- Xie, Y., Sichani, M. E., Padgett, J. E., and DesRoches, R.** (2020). The promise of implementing machine learning in earthquake engineering: A state-of-the-art review. *Earthquake Spectra*, 36(4):1769–1801.
- Xu, J.-G., Cai, Z.-K., and Feng, D.-C.** (2021). Life-cycle seismic performance assessment of aging rc bridges considering multi-failure modes of bridge columns. *Engineering Structures*, 244:112818.
- Yazgan, U.** (2015). Empirical seismic fragility assessment with explicit modeling of spatial ground motion variability. *Engineering Structures*, 100:479–489.

- Zang, C. and Imregun, M.** (2001). Structural Damage Detection Using Artificial Neural Networks and Measured Frf Data Reduced via Principal Component Projection. *Journal of Sound and Vibration*, 242:813–827.
- Zentner, I., Gündel, M., and Bonfils, N.** (2017). Fragility analysis methods: Review of existing approaches and application. *Nuclear Engineering and Design*, 323:245–258.
- Şahin, U.** (2010). Artificial neural networks approach for the determination of aquifer parameters. Master's thesis, Boğaziçi University.



APPENDICES

APPENDIX A : Dataset tables

APPENDIX B : Python code





APPENDIX A : Dataset tables



A.1 Strong motion records list

Table A.1 : The strong motion records list used in dynamic analyses

Event	Station	M	d (km.)	Site	Fault	PGV (cm/s)
Duzce (1999)	Duzce Meteorology Station, 270	7.3	8.2	D	SS	83.5
Northridge (1994)	Newhall, 360	6.7	7.1	D	RN	97.3
Chi-Chi (1999)	CHY028, W	7.6	7.3	D	RN	72.8
Gazli (1976)	Karakyr, 000	7.3	3	D	RN	65.4
Cape Mendocino (1992)	Petrolia, 000	7.1	9.5	D	RN	48.4
Morgan Hill (1984)	Halls Valley, 240	6.1	3.4	D	SS	39.4
Landers (1992)	22170 Joshua Tree, 000	7.4	11.6	C	SS	43.2
Kocaeli (1999)	Arcelik, 000	7.4	17	C	SS	17.7
Loma Prieta (1989)	Gilroy-Array #2, 000	7.1	12.7	D	RO	32.9
Whittier (1987)	Brea-S. Flower Ave, 020	6	17.9	D	RN	7.1
Superstition Hills (1987)	El-Centro Imp Co Center, 090	6.6	13.9	D	SS	40.9
N. Palm Springs (1986)	Fun Valley, 045	6	15.8	C	RO	6.4

A.2 Incremental Dynamic Analysis: Monte Carlo Simulation dataset

Table A.2 : Incremental Dynamic Analysis results for the benchmark dataset

	IM (cm/s)	5	10	15	20	25	30	35	40
α_{ecu}	α_{Vu}								
-3.039239	-0.440682	3	9	12	12	12	12	12	12
-2.866021	1.092899	2	9	12	12	12	12	12	12
-2.836042	0.744984	2	9	12	12	12	12	12	12
-2.257714	-0.206802	0	5	10	12	12	12	12	12
-2.225966	-0.258874	0	5	10	11	12	12	12	12
-2.008153	0.191919	0	5	8	11	12	12	12	12
-1.998370	0.005546	0	5	8	11	12	12	12	12
-1.903142	0.558613	0	5	8	11	12	11	12	12
-1.851622	-0.501937	0	5	8	10	12	12	12	12
-1.723009	1.598771	0	5	8	11	12	12	12	12
-1.672764	0.526698	0	5	8	11	12	12	12	12
-1.632167	0.960155	0	5	8	11	12	12	12	12
-1.621709	0.059762	0	5	8	11	12	12	12	12
-1.593220	-0.509642	0	5	8	11	12	12	12	12
-1.552562	-2.222873	11	12	12	12	12	12	12	12
-1.545286	-0.116624	0	5	8	11	11	12	12	12
-1.493415	-0.069835	0	5	8	11	11	12	12	12
-1.479284	-0.113444	0	5	8	11	11	12	12	12
-1.449085	-1.054197	5	12	12	12	12	12	12	12
-1.448942	-0.119612	0	5	8	11	11	12	12	12
-1.410964	0.417503	0	5	8	11	11	12	12	12
-1.377384	-1.194345	5	12	12	12	12	12	12	12
-1.349729	-0.919277	5	11	12	12	12	12	12	12
-1.306664	-1.129969	5	12	12	12	12	12	12	12
-1.306007	0.877965	0	5	7	11	11	12	12	12
-1.225222	-0.830959	5	11	12	12	12	12	12	12
-1.214044	-0.966091	5	12	12	12	12	12	12	12
-1.113506	-0.437391	0	5	7	10	11	12	12	12
-1.077070	0.202276	0	4	7	10	11	12	12	12
-1.042817	-0.836914	5	11	12	12	12	11	12	12
-1.042352	0.032461	0	5	7	10	11	12	12	12
-1.020212	-0.552463	0	5	7	11	12	12	12	12
-0.971772	-1.907064	11	12	12	12	12	12	12	12
-0.971654	-2.245652	11	12	12	12	12	12	12	12
-0.895356	-0.682583	2	9	12	12	12	12	12	11
-0.886476	0.510357	0	4	7	10	11	12	12	11
-0.866548	1.419098	0	4	7	10	11	12	12	12

Table A.2 (continued): Incremental Dynamic Analysis results for the benchmark dataset

	IM (cm/s)	5	10	15	20	25	30	35	40
α_{cu}	α_{vu}								
-0.846238	0.752774	0	4	7	10	11	12	12	11
-0.844552	-0.013148	0	4	7	10	11	12	12	11
-0.840824	-0.385524	0	4	7	10	11	12	12	11
-0.811972	0.520809	0	4	7	10	11	12	12	12
-0.790693	-1.248147	5	12	12	12	12	12	12	12
-0.789730	-1.995619	11	12	12	12	12	12	12	12
-0.789095	-1.380593	7	12	12	12	12	12	12	12
-0.778807	-1.032127	5	12	12	12	12	12	12	12
-0.746317	2.449260	0	4	7	10	11	12	12	12
-0.733918	0.193531	0	4	7	10	11	12	12	12
-0.732890	0.255868	0	4	7	10	11	12	12	12
-0.729457	-2.135385	11	12	12	12	12	12	12	12
-0.691497	-0.216367	0	4	7	10	11	12	12	12
-0.690105	1.695219	0	4	7	10	11	12	12	12
-0.681448	1.040816	0	4	7	10	11	12	12	12
-0.674750	-0.562688	0	4	7	11	12	12	12	12
-0.673363	-1.992607	11	12	12	12	12	12	12	12
-0.645401	-0.344604	0	4	7	10	11	12	12	12
-0.607776	0.904152	0	4	7	10	11	12	12	12
-0.603346	-0.223885	0	4	7	10	11	12	12	12
-0.589912	0.494344	0	4	7	10	11	12	12	12
-0.589039	1.237352	0	4	7	10	11	12	12	12
-0.579621	0.089093	0	4	7	10	11	12	12	12
-0.576291	1.978517	0	4	7	10	11	12	12	12
-0.548308	-0.261744	0	4	7	10	11	12	12	12
-0.536050	1.968829	0	4	7	10	11	12	12	12
-0.535846	1.830324	0	4	7	10	11	12	12	12
-0.528734	-1.642813	9	12	12	12	12	12	12	12
-0.490592	1.021719	0	4	7	10	10	12	12	12
-0.489758	0.347873	0	4	7	10	10	12	12	12
-0.469207	-1.118583	5	12	12	12	12	12	12	12
-0.458061	-0.761114	3	10	12	12	12	12	12	12
-0.454302	-0.405353	0	4	7	10	10	12	12	12
-0.433434	0.146922	0	4	7	10	9	12	12	12
-0.406774	-0.111342	0	4	7	10	9	12	12	12
-0.400461	0.837063	0	4	7	10	9	12	12	12
-0.395340	0.335435	0	4	7	10	9	12	12	12
-0.388268	-0.601898	0	5	10	12	12	12	12	12
-0.324698	1.764167	0	4	7	10	10	12	12	12
-0.323078	0.927679	0	4	7	10	10	12	12	12
-0.309003	-1.257232	5	12	12	12	12	12	12	12

Table A.2 (continued): Incremental Dynamic Analysis results for the benchmark dataset

	IM (cm/s)	5	10	15	20	25	30	35	40
α_{ecu}	α_{Vu}								
-0.308588	0.013879	0	4	7	9	10	12	12	12
-0.299017	1.474939	0	3	7	10	10	12	12	12
-0.295012	0.798813	0	3	7	9	10	12	12	12
-0.291460	0.633552	0	3	7	9	10	12	12	12
-0.283424	-2.092399	11	12	12	12	12	12	12	12
-0.277002	0.385250	0	3	7	9	10	12	12	12
-0.263921	0.795645	0	3	7	10	10	12	12	12
-0.263392	1.075900	0	3	7	10	10	12	12	12
-0.251555	0.031718	0	3	7	10	10	12	12	12
-0.229094	-0.325953	0	3	7	10	10	12	12	12
-0.225907	1.846173	0	3	7	10	10	12	12	12
-0.223276	-0.981362	5	12	12	12	12	12	12	12
-0.207603	0.823801	0	3	7	10	10	12	12	12
-0.142229	0.115344	0	3	7	10	10	12	12	12
-0.131288	0.148121	0	3	7	10	10	12	12	12
-0.121008	-0.333841	0	3	7	10	10	12	12	12
-0.107304	0.439367	0	3	7	10	10	11	12	12
-0.103519	-1.502519	8	12	12	12	12	12	12	12
-0.095201	0.136546	0	3	7	10	10	11	12	12
-0.094590	-0.696736	2	10	12	12	12	12	12	12
-0.093541	-1.439423	7	12	12	12	12	12	12	12
-0.078770	-0.308496	0	2	7	10	10	11	12	12
-0.076526	0.723060	0	2	7	10	10	11	12	12
-0.068557	-1.549882	8	12	12	12	12	12	12	12
-0.056739	-0.793895	4	11	12	12	12	12	12	12
-0.011778	-0.828608	5	11	12	12	12	12	12	12
-0.003844	-0.464870	0	2	7	10	10	11	12	12
0.005395	0.382394	0	2	7	9	10	11	12	12
0.022005	0.868532	0	2	7	9	10	11	12	12
0.029056	0.412946	0	2	7	9	10	11	12	12
0.030211	0.335712	0	2	7	9	10	11	12	11
0.051794	-0.489162	0	4	7	9	10	12	12	12
0.071889	-0.665091	2	8	12	12	12	12	12	12
0.078604	-0.423357	0	2	7	9	10	11	12	12
0.088821	-0.364918	0	2	7	9	10	11	12	12
0.090657	-1.624979	9	12	12	12	12	12	12	12
0.112140	0.734129	0	2	7	9	10	11	12	12
0.119186	-0.481406	0	3	7	9	10	11	12	12
0.122514	2.124450	0	2	7	9	10	11	12	12
0.123702	0.559735	0	2	7	9	10	11	12	12
0.132020	-1.237832	5	12	12	12	12	12	12	12

Table A.2 (continued): Incremental Dynamic Analysis results for the benchmark dataset

	IM (cm/s)	5	10	15	20	25	30	35	40
α_{ecu}	α_{vu}								
0.135105	1.576737	0	2	7	9	10	11	12	12
0.141767	1.609928	0	2	7	9	10	11	12	12
0.165695	-0.542875	0	4	7	11	11	12	12	12
0.166541	-0.957976	5	12	12	12	12	12	12	12
0.198585	1.035026	0	2	6	9	10	11	12	12
0.201417	0.694418	0	2	6	9	10	11	12	12
0.218316	-0.676271	2	8	12	12	12	12	12	12
0.220993	-0.085440	0	2	6	9	10	11	12	12
0.229827	1.312882	0	2	6	9	9	11	12	12
0.239310	-1.174496	5	12	12	12	12	12	12	12
0.262716	1.441062	0	2	6	9	10	11	12	12
0.276414	-1.452163	7	12	12	12	12	12	12	12
0.284043	-0.825967	5	11	12	12	12	12	11	12
0.298947	0.193122	0	2	6	9	10	11	11	12
0.299611	-0.302874	0	2	6	9	10	11	12	12
0.306136	-1.482506	8	12	12	12	12	12	12	12
0.307791	-0.519835	0	4	7	10	10	12	12	12
0.319582	-0.674504	2	8	12	12	11	12	12	12
0.319803	0.994776	0	2	6	9	10	11	12	12
0.327282	0.104628	0	2	6	9	10	11	12	12
0.328494	0.596714	0	2	6	9	10	11	12	12
0.381918	0.387581	0	2	5	9	10	11	12	12
0.382209	-0.143422	0	2	5	9	9	11	12	12
0.391783	-0.247960	0	2	5	9	10	11	12	12
0.396060	2.082025	0	2	5	9	10	11	12	12
0.441380	1.011387	0	2	5	9	9	11	12	12
0.441612	-0.793970	4	11	12	12	11	12	12	12
0.472954	0.265778	0	2	5	9	10	11	12	12
0.478160	-0.534467	0	4	7	10	11	12	12	12
0.487568	-2.185317	11	12	12	12	12	12	12	12
0.525078	0.163775	0	2	5	9	10	11	12	12
0.530087	0.325780	0	2	5	9	10	11	12	12
0.540215	0.707383	0	2	5	9	10	11	12	12
0.552943	-1.324055	5	12	12	12	12	12	12	12
0.555290	0.666853	0	2	5	9	10	11	12	12
0.615557	-0.452936	0	2	4	9	10	11	12	12
0.632148	0.851056	0	1	4	9	10	11	12	12
0.664072	1.331177	0	1	4	9	10	10	12	12
0.665694	-0.193389	0	1	4	9	10	10	12	12
0.678192	-0.881990	5	11	12	12	12	12	12	12
0.702821	0.737857	0	1	4	9	10	11	12	12

Table A.2 (continued): Incremental Dynamic Analysis results for the benchmark dataset

	IM (cm/s)	5	10	15	20	25	30	35	40
α_{ecu}	α_{Vu}								
0.743673	-0.579267	0	5	7	12	12	12	12	12
0.767203	0.306267	0	1	4	9	9	11	12	12
0.820731	0.147512	0	1	4	9	10	11	12	12
0.868269	-0.588425	0	5	8	12	12	12	11	12
0.889780	1.637577	0	1	4	9	9	11	12	12
0.925196	-1.705285	9	12	12	12	12	12	12	12
0.925500	-0.228404	0	1	3	7	10	11	12	12
0.979753	-0.401244	0	1	4	8	10	10	12	12
0.989691	-0.299093	0	1	3	8	10	11	12	12
1.084017	1.024458	0	1	4	8	10	11	12	12
1.098455	-0.253974	0	1	4	8	9	11	12	12
1.111541	-0.661538	2	8	12	12	12	12	12	12
1.136671	0.904157	0	1	4	8	9	11	11	12
1.147204	0.176686	0	1	4	8	9	11	11	12
1.248968	0.768996	0	1	4	7	10	11	10	12
1.266531	-1.314441	5	12	12	12	12	12	12	12
1.326151	-0.652561	1	8	12	12	12	12	12	12
1.441140	0.820386	0	1	4	8	10	10	10	12
1.449560	2.017887	0	1	4	8	10	9	10	12
1.509006	-0.520590	0	2	7	10	10	11	12	12
1.513096	-0.759003	3	10	12	12	12	12	12	12
1.553604	0.517692	0	0	4	8	10	10	10	11
1.586488	0.161509	0	0	4	8	10	10	10	12
1.619530	0.177823	0	0	4	8	10	9	10	12
1.624858	0.345649	0	0	4	8	10	9	10	12
1.626402	0.288962	0	0	4	8	10	9	10	12
1.629068	-0.364909	0	0	4	8	10	9	10	12
1.789157	-0.355256	0	0	3	8	10	10	9	11
1.796346	0.606454	0	0	3	7	10	10	9	11
1.819249	-0.888401	5	11	12	12	12	12	12	12
1.893350	-1.352377	6	12	12	12	12	12	12	12
1.910625	0.114848	0	0	3	7	10	10	10	11
1.946163	-0.219992	0	0	3	7	10	10	10	11
1.964398	1.311166	0	0	3	8	10	10	10	11
1.982395	-0.207973	0	0	3	8	10	9	10	11
2.030083	0.328762	0	0	3	8	10	10	9	10
2.096336	1.066697	0	0	3	7	10	10	10	11
2.213658	-0.327922	0	0	3	7	10	10	8	10
2.349238	1.677437	0	0	3	6	9	10	10	11
2.703553	0.401639	0	0	2	6	8	10	9	11

A.3 Incremental Dynamic Analysis: RSM training dataset

Table A.3 : Incremental Dynamic Analysis results for the RSM training dataset.

	IM (cm/s)	5	10	15	20	25	30	35	40
α_{ecu}	α_{vu}								
$-\sqrt{2}$	0.000	12	12	12	12	12	12	12	12
-1.000	-1.000	5	12	12	12	12	12	12	12
-1.000	1.000	0	5	7	10	11	12	12	12
0.000	$-\sqrt{2}$	9	12	12	12	12	12	12	12
0.000	0.000	0	2	7	9	10	11	12	12
0.000	$-\sqrt{2}$	0	2	7	9	10	11	12	12
1.000	-1.000	5	12	12	12	12	12	12	12
1.000	1.000	0	1	3	7	10	11	12	12
$\sqrt{2}$	0.000	0	1	4	8	10	11	12	12

A.4 Incremental Dynamic Analysis: ANN training dataset

Table A.4 : Incremental Dynamic Analysis results for the ANN training dataset

	IM (cm/s)	5	10	15	20	25	30	35	40
α_{ecu}	α_{Vu}								
-2.000000	-2.000000	11	12	12	12	12	12	12	12
-2.000000	-1.333333	5	12	12	12	12	12	12	12
-2.000000	-0.666667	2	8	12	12	12	12	12	12
-2.000000	0.000000	0	5	8	11	12	12	12	12
-2.000000	0.666667	0	5	8	11	12	12	12	12
-2.000000	1.333333	0	5	8	11	12	12	12	12
-2.000000	2.000000	0	5	8	11	12	12	12	12
-1.333333	-2.000000	11	12	12	12	12	12	12	12
-1.333333	-1.333333	5	12	12	12	12	12	12	12
-1.333333	-0.666667	2	8	12	12	12	12	12	12
-1.333333	0.000000	0	5	7	11	11	12	12	12
-1.333333	0.666667	0	5	7	11	11	12	12	12
-1.333333	1.333333	0	5	7	11	11	12	12	12
-1.333333	2.000000	0	5	7	11	11	12	12	12
-0.666667	-2.000000	11	12	12	12	12	12	12	12
-0.666667	-1.333333	5	12	12	12	12	12	12	12
-0.666667	-0.666667	2	8	12	12	12	12	12	12
-0.666667	0.000000	0	4	7	10	11	12	12	12
-0.666667	0.666667	0	4	7	10	11	12	12	12
-0.666667	1.333333	0	4	7	10	11	12	12	12
-0.666667	2.000000	0	4	7	10	11	12	12	12
0.000000	-2.000000	11	12	12	12	12	12	12	12
0.000000	-1.333333	5	12	12	12	12	12	12	12
0.000000	-0.666667	2	8	12	12	12	12	12	12
0.000000	0.000000	0	2	7	10	10	11	12	12
0.000000	0.666667	0	2	7	10	10	11	12	12
0.000000	1.333333	0	2	7	10	10	11	12	12
0.000000	2.000000	0	2	7	10	10	11	12	12
0.666667	-2.000000	11	12	12	12	12	12	12	12
0.666667	-1.333333	5	12	12	12	12	12	12	12
0.666667	-0.666667	2	8	12	12	12	12	12	12
0.666667	0.000000	0	1	4	9	10	11	12	12
0.666667	0.666667	0	1	4	9	10	11	12	12
0.666667	1.333333	0	1	4	9	10	11	12	12
0.666667	2.000000	0	1	4	9	10	11	12	12
1.333333	-2.000000	11	12	12	12	12	12	12	12
1.333333	-1.333333	5	12	12	12	12	12	12	12
1.333333	-0.666667	2	8	12	12	12	12	12	12
1.333333	0.000000	0	1	4	8	10	11	10	11

Table A.4 (continued): Incremental Dynamic Analysis results for the ANN training dataset

	IM (cm/s)	5	10	15	20	25	30	35	40
α_{ecu}	α_{vu}								
1.333333	0.666667	0	1	4	8	10	11	10	11
1.333333	1.333333	0	1	4	8	10	11	10	11
1.333333	2.000000	0	1	4	8	10	11	10	11
2.000000	-2.000000	11	12	12	12	12	12	12	12
2.000000	-1.333333	5	12	12	12	12	12	12	12
2.000000	-0.666667	2	8	12	12	12	12	12	12
2.000000	0.000000	0	0	3	8	10	10	9	10
2.000000	0.666667	0	0	3	8	10	10	9	10
2.000000	1.333333	0	0	3	8	10	10	9	10
2.000000	2.000000	0	0	3	8	10	10	9	10

APPENDIX B : Source code

B.1 Incremental Dynamic Analysis python code

Monte Carlo simulation approach for constructing the benchmark model:

```
1  import openseespy.opensees as ops
2  import opsvis as opsv
3  import numpy as np
4  import pandas as pd
5  import matplotlib.pyplot as plt
6  import math
7  import xlswriter
8  import itertools as itools
9  import subprocess as sp
10 import time
11 from google.colab import files
12
13 start_time = time.time()
14 tmp = sp.call('cls', shell=True)
15
16
17 def ReadRecord (inFilename, outFilename):
18     dt = 0.0
19     npts = 0
20     inFileID = open(inFilename, 'r')
21     outFileID = open(outFilename, 'w')
22     flag = 0
23     for line in inFileID:
24         if line == '\n':
25             continue
26         elif flag == 1:
27             outFileID.write(line)
28         else:
29             words = line.split()
30             lengthLine = len(words)
31             if lengthLine >= 4:
32                 if words[0] == 'NPTS=':
33                     for word in words:
34                         if word != ':':
35                             if flag == 1:
36                                 dt = float(word)
37                                 break
38                             if flag == 2:
39                                 npts = int(word.strip(','))
40                                 flag = 0
41                             if word == 'DT=' or word == 'dt':
42                                 flag = 1
43                             if word == 'NPTS=':
44                                 flag = 2
45             elif words[-1] == 'DT':
46                 count = 0
47                 for word in words:
```

```

48         if word != '':
49             if count == 0:
50                 npts = int(word)
51             elif count == 1:
52                 dt = float(word)
53             elif word == 'DT':
54                 flag = 1
55                 break
56             count += 1
57
58     inFileID.close()
59     outFileID.close()
60     return dt, npts
61
62 g = 9.81
63 record_list = ['RSN126_GAZLI_GAZ000', 'RSN461_MORGAN_HVR240',
64               'RSN518_PALMSPR_FVR045', 'RSN599_WHITTIER.A_A-FLO020',
65               'RSN721_SUPER.B_B-ICC090', 'RSN766_LOMAP_G02000',
66               'RSN828_CAPEMEND_PET000', 'RSN864_LANDERS_JOS090',
67               'RSN1044_NORTHR_NWH360', 'RSN1148_KOCAELI_ARE000',
68               'RSN1197_CHICHI_CHY028-E', 'RSN1605_DUZCE_DZC270']
69
70 pgv_list = [66.19, 39.32, 6.28, 7.13, 41.77, 34.74,
71            49.30, 42.57, 96.54, 13.95, 61.36, 84.19]
72
73 factor_list = np.linspace(40, 5, 8)
74
75 ecu_list = -np.random.lognormal(np.log(0.006), np.sqrt(np.log(1+0.35**2)), 100)
76 Vult_list = np.random.lognormal(np.log(170), np.sqrt(np.log(1+0.20**2)), 100)
77
78 ultimate_result = pd.DataFrame(np.zeros((len(factor_list)*
79                                         len(record_list)*len(ecu_list), 10)),
80                               columns=['state', 'IM', 'record', 'ecu',
81                                       'Vult', 'Vbmax', 'ect', 'ecb',
82                                       'uindex', 'convergence'])
83
84 j1 = 0
85 nrows = len(factor_list)*len(record_list)*len(ecu_list)
86 z_cstrn=np.zeros((18002, 8, len(factor_list), len(record_list), len(ecu_list)))
87 Tcalc_0 = 0
88
89 for rl, record, pgv in zip(itertools.count(), record_list, pgv_list):
90     dt, nPts = ReadRecord('https://github.com/mtareqsoliman/mstthesis/blob/main/
91                          Data_Input/'+record+'.AT2',
92                          'Data_Output/'+record+'.dat')
93     for fl, factor in enumerate(factor_list):
94         for el, ecu, Vult in zip(itertools.count(), ecu_list, Vult_list):
95             print("=====")
96             print("Starting RCFrameGravity example")
97
98             ops.wipe()
99             ops.model('basic', '-ndm', 2, '-ndf', 3)
100
101             bay, height1, height2 = 8.0, 4.0, 3.0
102
103             ops.node(1, 0.0, 0.0)
104             ops.node(2, bay, height1-height2)
105             ops.node(3, 0.0, height1)
106             ops.node(4, bay, height1)

```

```

107 ops.fix(1, 1, 1, 1)
108 ops.fix(2, 1, 1, 1)
109
110
111 ops.equalDOF(3,4,1)
112
113 fcu, Ec, ftc, ec0 = [-30*10**3, 30*10**6, 3000, -0.002]
114 fy, Es, bs = [400*10**3, 200*10**6, 0.03]
115
116 ops.uniaxialMaterial('Concrete04', 1, fcu, ec0, ecu, Ec, ftc)
117 ops.uniaxialMaterial('Steel02', 3, fy, Es, bs)
118
119 seccol, bc, tc, cover, phic, ntopc, nbotc = [1, 0.2, 0.5, 0.025]
120 phic, ntopc, nbotc = [0.014, 4, 4]
121 zlc, ylc, Asc = [bc/2, tc/2, math.pi*phic**2/4]
122 fib_sec_col = [['section', 'Fiber', seccol, '-GJ', 0.0e6],
123               ['patch', 'rect', 1, 18, 1, -ylc, -zlc, ylc, zlc],
124               ['layer', 'straight', 3, ntopc, Asc,
125                ylc - cover-phic/2, zlc - cover-phic/2,
126                ylc - cover-phic/2, cover - zlc+phic/2],
127               ['layer', 'straight', 3, nbotc, Asc,
128                cover - ylc+phic/2, zlc - cover-phic/2,
129                cover - ylc+phic/2, cover - zlc+phic/2]]
130 opsv.fib_sec_list_to_cmds(fib_sec_col)
131
132 secbij, bb, tb, cover = [2, 0.2, 0.7, 0.025]
133 phib, ntopbij, nbotbij = [0.016, 4, 2]
134 zlb, ylb, Asb = [bb/2, tb/2, math.pi*phib**2/4]
135 fib_sec_bij = [['section', 'Fiber', secbij, '-GJ', 0.0],
136               ['patch', 'rect', 1, 18, 1, -ylb, -zlb, ylb, zlb],
137               ['layer', 'straight', 3, ntopbij, Asb,
138                ylb - cover-phib/2, zlb - cover-phib/2,
139                ylb - cover-phib/2, cover - zlb+phib/2],
140               ['layer', 'straight', 3, nbotbij, Asb,
141                cover - ylb+phib/2, zlb - cover-phib/2,
142                cover - ylb+phib/2, cover - zlb+phib/2]]
143 opsv.fib_sec_list_to_cmds(fib_sec_bij)
144
145 secbms, bb, tb, cover = [3, 0.2, 0.7, 0.025]
146 phib, ntopbms, nbotbms = [0.016, 2, 4]
147 zlb, ylb, Asb = [bb/2, tb/2, math.pi*phib**2/4]
148 fib_sec_bms = [['section', 'Fiber', secbms, '-GJ', 0.0],
149               ['patch', 'rect', 1, 18, 1, -ylb, -zlb, ylb, zlb],
150               ['layer', 'straight', 3, ntopbms, Asb,
151                ylb - cover-phib/2, zlb - cover-phib/2,
152                ylb - cover-phib/2, cover - zlb+phib/2],
153               ['layer', 'straight', 3, nbotbms, Asb,
154                cover - ylb+phib/2, zlb - cover-phib/2,
155                cover - ylb+phib/2, cover - zlb+phib/2]]
156 opsv.fib_sec_list_to_cmds(fib_sec_bms)
157
158
159 ops.uniaxialMaterial('Elastic', 4, 10**0)
160 ops.section('Aggregator', 20, 4, 'P', '-section', secbij)
161 ops.section('Aggregator', 30, 4, 'P', '-section', secbms)
162
163 ops.geomTransf('Linear', 1)
164 #plc = 0.08*L(m) + 0.022*phi(m)*fy(MPa)
165 plc1 = (0.08*0.38*height1) + (0.022*phic*fy*10**3)

```

```

166     plc2=(0.08*0.22*height2)+(0.022*phic*fy*10**-3)
167     ops.beamIntegration('HingeRadau', 1, 1, plc1,1,plc1,1)
168     ops.beamIntegration('HingeRadau', 2, 1, plc2,1,plc2,1)
169
170     #           e           tag ndI ndJ transfTag integrationTag
171     ops.element('forceBeamColumn', 1, 1, 3, 1, 1)
172     ops.element('forceBeamColumn', 2, 2, 4, 1, 2)
173
174     ops.geomTransf('Linear', 2)
175     plbm=(0.08*0.18*bay)+(0.022*phib*fy*10**-3)
176
177     ops.beamIntegration('HingeRadau', 3, 2, plbm,2,plbm,30)
178     ops.element('forceBeamColumn', 3, 3, 4, 2, 3)
179
180
181     ops.timeSeries('Linear', 1)
182     ops.pattern('Plain', 1, 1)
183
184
185     P, Wz = 500.0, 36.0
186     #           nd   FX, FY, MZ
187     ops.load(3, 0.0, -P, 0.0)
188     ops.load(4, 0.0, -P, 0.0)
189     ops.eleLoad('-ele', 3, '-type', '-beamUniform', -Wz)
190
191     n= Es/Ec
192
193     colIz = (0.2*0.5**3/12)
194             + (n-1)*(nbotc*Asc)*(z1c-cover-phic/2)**2
195             + (n-1)*(ntopc*Asc)*(z1c-cover-phic/2)**2
196     Kl= 12*Ec*colIz*(1/(height1**3)+1/(height2**3))
197
198     Tn = 0.48
199     mass1 = ((Tn**2)*Kl)/(4*(math.pi**2))
200
201     # node#,      mx   my   mz
202     ops.mass(3, mass1/4, 0, 0.0)
203     ops.mass(4, mass1/4, 0, 0.0)
204
205     ops.system('BandGeneral')
206     ops.constraints('Transformation')
207     ops.numberer('RCM')
208     ops.test('NormDispIncr', 1.0e-12, 10, 3)
209     ops.algorithm('Newton')
210     ops.integrator('LoadControl', 0.1)
211     ops.analysis('Static')
212
213     ops.analyze(10)
214
215     print("=====")
216
217     # =====
218     #
219     print("Start RCFRAMEEarthquake Example")
220     #
221     # =====
222
223     ops.wipeAnalysis()
224     ops.loadConst('-time', 0.0)

```

```

225
226 ops.timeSeries('Path', 2, '-filePath',
227               'Data_Output/'+record+'.dat',
228               '-dt', dt, '-factor', factor*9.81/(pgv))
229 ops.pattern('UniformExcitation', 2, 1, '-accel', 2)
230 #ops.rayleigh(0.0, 0.0, 0.0, 0.000625)
231 ops.system('BandGeneral')
232 ops.constraints('Plain')
233 ops.test('NormDispIncr', 1.0e-12, 10 )
234 ops.algorithm('Newton')
235 ops.numberer('RCM')
236 ops.integrator('Newmark', 0.5, 0.25 )
237 ops.analysis('Transient')
238 tFinal, tCurrent, ok = nPts*dt, ops.getTime(), 0
239 ops.record()
240
241 i = 0
242 z_v1 = []
243 z_v2 = []
244
245 while ok == 0 and tCurrent < tFinal:
246
247     tCurrent = ops.getTime()
248
249     z_v1.append(abs(ops.eleForce(1,1))),
250     z_v2.append(abs(ops.eleForce(2,1)))
251     # [BOT lowerstrain, BOT upperconc strain,
252     # TOP lowerstrain, TOP upperstrain]
253     z_cstrn[i,0,f1,r1,e1],z_cstrn[i,1,f1,r1,e1],
254     z_cstrn[i,2,f1,r1,e1],z_cstrn[i,3,f1,r1,e1]=
255     ops.eleResponse(1, 'section', 1, 'fiber', -y1c, 0, 1,
256                    'stressStrain')[1],
257     ops.eleResponse(1, 'section', 1, 'fiber', y1c, 0, 1,
258                    'stressStrain')[1],
259     ops.eleResponse(1, 'section', 6, 'fiber', -y1c, 0, 1,
260                    'stressStrain')[1],
261     ops.eleResponse(1, 'section', 6, 'fiber', y1c, 0, 1,
262                    'stressStrain')[1]
263
264     z_cstrn[i,4,f1,r1,e1],z_cstrn[i,5,f1,r1,e1],
265     z_cstrn[i,6,f1,r1,e1],z_cstrn[i,7,f1,r1,e1]=
266     ops.eleResponse(2, 'section', 1, 'fiber', -y1c, 0, 1,
267                    'stressStrain')[1],
268     ops.eleResponse(2, 'section', 1, 'fiber', y1c, 0, 1,
269                    'stressStrain')[1],
270     ops.eleResponse(2, 'section', 6, 'fiber', -y1c, 0, 1,
271                    'stressStrain')[1],
272     ops.eleResponse(2, 'section', 6, 'fiber', y1c, 0, 1,
273                    'stressStrain')[1]
274
275     if ultimate_result.loc[j1,'state'] == False and
276     min(z_cstrn[i,:,f1,r1,e1]) < ecu:
277         ultimate_result.loc[j1,'uindex'] = i
278         ultimate_result.loc[j1,'state'] = 1
279     if ultimate_result.loc[j1,'state'] == False and
280     max(max(z_v1),max(z_v2),abs(min(z_v1)),abs(min(z_v2)))>Vult:
281         ultimate_result.loc[j1,'uindex'] = i
282         ultimate_result.loc[j1,'state'] = 1
283

```

```

284         ok = ops.analyze(1, dt)
285
286         if ok != 0:
287             print("modified newton failed")
288             print("lets try an initail stiffness for this step")
289             ops.test('NormDispIncr', 1.0e-12, 100, 0)
290             ops.algorithm('ModifiedNewton', '-initial')
291             ok =ops.analyze( 1, dt)
292             if ok == 0:
293                 print('Succeeded, return back to Newton Algorithm')
294                 ops.algorithm('Newton')
295         if ok != 0:
296             print("Trying Newton with Initial Tangent")
297             ops.test("EnergyIncr",1.e-8,6,0)
298             ops.algorithm("Newton", '-initial')
299             ok =ops.analyze( 1, dt)
300             if ok == 0:
301                 print('Succeeded, return back to Newton Algorithm')
302                 ops.test("NormDispIncr", 1.e-8,2000,0)
303                 ops.algorithm('Newton')
304         if ok!=0:
305             print("Trying Broyden ..")
306             ops.algorithm("Broyden", 8)
307             ok =ops.analyze( 1, dt)
308             if ok == 0:
309                 print('Succeeded, return back to Newton Algorithm')
310                 ops.algorithm('Newton')
311         if ok!=0:
312             print("Trying NewtonWithLineSearch")
313             ops.algorithm("NewtonLineSearch",0.8)
314             ok =ops.analyze( 1, dt)
315             if ok == 0:
316                 print('Succeeded, return back to Newton Algorithm')
317                 ops.algorithm('Newton')
318         if ok!=0:
319             print('Failed to converge :( breaking the loop')
320             ultimate_result.loc[j1,'convergence'] = 'FAILED'
321             break
322
323         i = i+1
324
325     ops.wipe()
326
327     # =====
328     # Postprocessing Stage
329     # =====
330
331     print("Progress =" +str(round((j1/nrows),4)*100)+" %")
332
333     ultimate_result.loc[j1,'IM'] = factor #IM
334     ultimate_result.loc[j1,'record'] = record #record
335     ultimate_result.loc[j1,'ecu'] = ecu #ecu
336     ultimate_result.loc[j1,'Vult'] = Vult #V
337
338     ultimate_result.loc[j1,'ect'] = min(min(z_cstrn[:,6,f1,r1,e1]),
339                                         min(z_cstrn[:,7,f1,r1,e1]))
340     ultimate_result.loc[j1,'ecb'] = min(min(z_cstrn[:,4,f1,r1,e1]),
341                                         min(z_cstrn[:,5,f1,r1,e1]))
342     ultimate_result.loc[j1,'Vbmax'] = max(z_v1+z_v2)

```

```

343
344
345         # result - IM - record - ecu - Vult - Vb - ect - ecb - umax
346
347         j1= j1+1
348
349         def graphPlotter(x,y,lbl,t1,xlbl,ylbl):
350             fig, ax = plt.subplots(figsize=(11,7))
351             ax.plot(x, y, label=lbl)
352
353     print("--- %s seconds ---" % (time.time() - start_time))
354     ultimate_result.to_csv('MC_ultimate_result.csv',index=False)
355     files.download('MC_ultimate_result.csv')

```

B.2 RSM fragility model python code

Developing fragility model for RSM approach.

```

1  NLTHA_RSM = pd.read_csv('https://github.com/mtareqsoliman/mstthesis/blob/main/
2                          CaseStudy_1/sims_RSM_c1.csv?raw=true')
3
4  NLTHA_RSM_pivot = pd.pivot_table(NLTHA_RSM, values='state',
5                                   index=['ecu_alpha', 'Vult_alpha'],
6                                   columns=['IM'], aggfunc=np.sum)
7  NLTHA_RSM_pivot.insert(0,0,0)
8
9  from sklearn.preprocessing import PolynomialFeatures
10 from sklearn.linear_model import LinearRegression
11 from sklearn.metrics import mean_squared_error, r2_score
12
13 ecu_RSM_alpha_frag, Vult_RSM_alpha_frag = zip(*NLTHA_RSM_pivot.index.values)
14
15 lambda_RSM_list = []
16 zeta_RSM_list = []
17 z_RSM_list = []
18
19 IM = np.arange(0,45,5)
20 fragility_table_RSM = pd.DataFrame(columns = ['0', '5', '10', '15', '20',
21                                               '25', '30', '35', '40'])
22
23 def frag(lambdazeta):
24     estimated_fragility = sp.norm.cdf(np.log(factor_list),
25                                     np.log(lambdazeta[0]), lambdazeta[1])
26     log_likelihood = np.log(sp.binom.pmf(ncllps_RSM_i,
27                                     nrcrds, estimated_fragility))
28     fun = -np.sum(log_likelihood)
29
30     return fun
31
32 for i,ecu_i in enumerate(ecu_RSM_alpha_frag[0:9]):
33     factor_list = np.array([0,5,10,15,20,25,30,35,40])
34     ncllps_RSM_i = NLTHA_RSM_pivot.iloc[i]
35     nrcrds = 12
36
37     # starting guess
38     lambdazeta_start = [20,0.4]

```

```

39
40 # Bounds
41 bnds = ((5, 50), (0.2, 1))
42
43 result = spo.minimize(frag, lambdazeta_start,
44                      options={'disp': True}, bounds=bnds)
45
46 if result.success:
47     print('\u03BB='+str(lambda_mxlk)+' , \u03B6='+str(zeta_mxlk)+' ,
48           cost='+str(z_mxlk), end = '\n')
49     miny = result.x
50     lambda_mxlk = round(miny[0], 2)
51     zeta_mxlk = round(miny[1], 2)
52     lambda_RSM_list.append(lambda_mxlk)
53     zeta_RSM_list.append(zeta_mxlk)
54
55     z_mxlk = -round(result.fun, 2)
56     z_RSM_list.append(z_mxlk)
57
58 else:
59     print("Failed!", end = '\n')
60     miny = result.x
61     lambda_mxlk = round(miny[0], 2)
62     zeta_mxlk = round(miny[1], 2)
63     lambda_RSM_list.append(lambda_mxlk)
64     zeta_RSM_list.append(zeta_mxlk)
65
66     z_mxlk = -round(result.fun, 2)
67     z_RSM_list.append(z_mxlk)
68
69 from sklearn.preprocessing import StandardScaler
70
71 x1_RSM, x2_RSM = map(list, zip(*NLTHA_RSM_pivot.index.values))
72
73 x_RSM = np.column_stack((x1_RSM, x2_RSM))
74 y_RSM = np.column_stack((np.log(lambda_RSM_list), zeta_RSM_list))
75
76 deg2_polymodel = PolynomialFeatures(degree=2)
77 deg1_polymodel = PolynomialFeatures(degree=1)
78 x_poly_RSM_deg2 = deg2_polymodel.fit_transform(x_RSM)
79 x_poly_RSM_deg1 = deg1_polymodel.fit_transform(x_RSM)
80
81 deg2_RSM_regmodel = LinearRegression()
82 deg1_RSM_regmodel = LinearRegression()
83
84 deg2_RSM_regmodel.fit(x_poly_RSM_deg2, y_RSM[:, 0])
85 deg1_RSM_regmodel.fit(x_poly_RSM_deg1, y_RSM[:, 1])
86
87 y_pred_deg2 = deg2_RSM_regmodel.predict(x_poly_RSM_deg2)
88 y_pred_deg1 = deg1_RSM_regmodel.predict(x_poly_RSM_deg1)
89 y_pred = np.column_stack((y_pred_deg2, y_pred_deg1))
90
91 print(deg2_RSM_regmodel.coef_)
92 print(deg1_RSM_regmodel.coef_)
93
94 MSE = mean_squared_error(y_RSM[:, 0], y_pred_deg2, squared=False)
95
96 x1_mesh, x2_mesh = np.linspace(-2, 2, 50), np.linspace(-2, 2, 50)
97 x1_mesh, x2_mesh = np.meshgrid(x1_mesh, x2_mesh)

```

```

98 x_mesh = np.array([x1_mesh.flatten(), x2_mesh.flatten()]).T
99 poly_x_mesh_deg2 = deg2_polymodel.fit_transform(x_mesh)
100 poly_x_mesh_deg1 = deg1_polymodel.fit_transform(x_mesh)
101 y_mesh_deg2_lambda = deg2_RSM_regmodel.predict(poly_x_mesh_deg2)
102 y_mesh_deg1_zeta = deg1_RSM_regmodel.predict(poly_x_mesh_deg1)
103 y_mesh_deg2 = np.column_stack((y_mesh_deg2_lambda, y_mesh_deg1_zeta))
104
105 plt.rcParams['figure.figsize'] = (width*0.8/72.27,width*0.4/72.27)
106 plt.rcParams['figure.dpi'] = 100
107
108 fig, ax = plt.subplots(figsize=(width*1.4/72.27,width*0.8/72.27),
109                             subplot_kw={'projection': '3d'})
110 ax.autoscale(enable=True, axis='x', tight=True)
111 ax.set_zlabel(r"$\lambda$ (cm/s)",labelpad=-5)
112 ax.set_xlabel(r"$\alpha_\epsilon$",labelpad=-5)
113 ax.set_ylabel(r"$\alpha_v$",labelpad=-5)
114 ax.set_title("(a)")
115 ax.tick_params(axis='both', which='major', pad=-2.5)
116 ax.azim = 200
117 ax.set_box_aspect(aspect=None, zoom=0.94)
118 ax.view_init(20,225)
119
120
121
122 ax.scatter(x_RSM[:,0], x_RSM[:,1], np.exp(y_RSM[:,0]),color='black',
123            marker='.',alpha=1,s=20)
124 sc_2 = ax.scatter(x1_mesh, x2_mesh, np.exp(y_mesh_deg2[:,0]),
125                  c=np.exp(y_mesh_deg2[:,0]),
126                  marker="$\u25EF$", s=5, cmap='winter',alpha=0.35)
127 plt.savefig('RSM_surf_2nd_lambda.pdf', backend='pdf',
128            dpi=200, bbox_inches='tight')
129
130
131 fig, ax = plt.subplots(figsize=(width*1.4/72.27,width*0.8/72.27),
132                             subplot_kw={'projection': '3d'})
133 ax.set_zlabel(r"$\zeta$",labelpad=-2)
134 ax.set_xlabel(r"$\alpha_\epsilon$",labelpad=-5)
135 ax.set_ylabel(r"$\alpha_v$",labelpad=-5)
136 ax.set_title("(b)")
137 ax.tick_params(axis='both', which='major', pad=-1.5)
138 ax.azim = 225
139 ax.set_box_aspect(aspect=None, zoom=0.94)
140 ax.view_init(20,180+45)
141
142
143 ax.scatter(x_RSM[:,0], x_RSM[:,1], y_RSM[:,1],color='black',
144            marker='.',alpha=1,s=20)
145 sc_2 = ax.scatter(x1_mesh, x2_mesh, y_mesh_deg2[:,1],
146                  c=y_mesh_deg2[:,1],marker="$\u25EF$",
147                  s=5, cmap='winter',alpha=0.35)
148
149 plt.savefig('RSM_surf_2nd_zeta.pdf', backend='pdf',
150            dpi=200, bbox_inches='tight')
151
152 # should be in the form [1, a, b, a^2, ab, b^2]
153 from sklearn.preprocessing import PolynomialFeatures
154 from sklearn.linear_model import LinearRegression
155 from sklearn.metrics import mean_squared_error
156 from sklearn.metrics import r2_score

```

```

157
158 x1_BM,x2_BM = map(list,zip(*NLTHA_BM_pivot.index.values))
159 x_BM =np.column_stack((x1_BM,x2_BM))
160 y_BM = np.column_stack((np.log(lambda_BM_list),zeta_BM_list))
161
162
163 alpha_ecu_1000 = np.random.normal(0, 1, 1000)
164 alpha_Vult_1000 = np.random.normal(0, 1, 1000)
165
166 x_1000_RSM = np.array([alpha_ecu_1000, alpha_Vult_1000]).T
167 x_1000_poly_RSM_deg2 = deg2_polymodel.fit_transform(x_1000_RSM)
168 x_1000_poly_RSM_deg1 = deg1_polymodel.fit_transform(x_1000_RSM)
169
170 y_1000_RSM_deg2_lambda = deg2_RSM_regmodel.predict(x_1000_poly_RSM_deg2)
171 y_1000_RSM_deg2_zeta = deg1_RSM_regmodel.predict(x_1000_poly_RSM_deg1)
172 y_1000_RSM_deg2 = np.column_stack((y_1000_RSM_deg2_lambda,y_1000_RSM_deg2_zeta))
173
174 fragility_table_RSM_1000_deg2 = pd.DataFrame(columns = ['0','5','10','15','20',
175                                                         '25','30','35','40'])
176
177 for i in np.arange(len(y_1000_RSM_deg2)):
178     fragility_table_RSM_1000_deg2.loc[i] = sp.norm.cdf(np.log(IM),
179                                                         y_1000_RSM_deg2[i,0],
180                                                         y_1000_RSM_deg2[i,1])*12
181
182 import math
183
184 def frag(lambdazeta):
185     estimated_fragility = sp.norm.cdf(np.log(factor_list),
186                                     np.log(lambdazeta[0]),lambdazeta[1])
187     log_likelihood = np.log(sp.binom.pmf(ncllps_CCM1000_deg2,12,
188                                     estimated_fragility))
189     fun = -np.sum(log_likelihood)
190
191     return fun
192
193 factor_list = np.array([0,5,10,15,20,25,30,35,40])
194 nrcrds = 12
195 ncllps_CCM1000_deg2 = np.round(fragility_table_RSM_1000_deg2.median())
196 pcllps_CCM1000_deg2 = ncllps_CCM1000_deg2/12
197
198 # starting guess
199 lambdazeta_start = np.array([20,0.4])
200
201 # Bounds
202 bnds = ((5, 50), (0.2, 1))
203
204 result = spo.minimize(frag,lambdazeta_start,options={'disp': True},bounds=bnds)
205 RSM_1000_lgnrml = []
206 if result.success:
207     print("Success!", end = '\n')
208     miny = result.x
209     RSM_1000_lambda = round(miny[0],2)
210     RSM_1000_zeta = round(miny[1],2)
211     RSM_1000_cost = -round(result.fun,2)
212
213 RSM_1000_lz_deg2 = [RSM_1000_lambda,RSM_1000_zeta,RSM_1000_cost]
214 RSM_1000_lgnrml_deg2 = sp.norm.cdf(np.log(np.linspace(0,40,1000)),
215                                     np.log(RSM_1000_lz_deg2[0]),

```

```

216                                     RSM_1000_lz_deg2[1])
217
218 print(result)
219
220 import matplotlib.ticker as mtick
221
222 for y, sel_IM, title in zip([0, 1, 2, 3], ["10", "20", "30", "40"],
223                               ['(a)', '(b)', '(c)', '(d)']):
224     fig, ax = plt.subplots(figsize=(width*0.5/72.27, width*0.45/72.27))
225
226     ax.hist((fragility_table_RSM_1000_deg2[sel_IM]*100)/12, color='skyblue',
227             edgecolor='royalblue', alpha=0.65,
228             bins=np.arange(min((fragility_table_RSM_1000_deg2[sel_IM]*100)/12),
229                             max((fragility_table_RSM_1000_deg2[sel_IM]*100)/12)+10,
230                             10))
231
232     ax.set_xlim(0, 100)
233     ax.xaxis.set_major_formatter(pticks)
234     ax.set_ylim(0)
235     ax.set_xlabel(r"$P(D>d|IM=x)$")
236     ax.set_ylabel('No. of realizations')
237     ax.set_title(title)
238     ax.axvline(fragility_table_RSM_1000_deg2[sel_IM].median()*100/12,
239               color='royalblue', linestyle='--', linewidth=1.5)
240     plt.savefig('histogram1020_RSM_'+str(sel_IM)+'.pdf', backend='pdf',
241               dpi=600, bbox_inches="tight", transparent=True)
242
243 fig, ax = plt.subplots(figsize=set_size(width))
244
245 ax.plot(np.linspace(0, 40, 1000), RSM_1000_lgnrml_deg2*100,
246         linestyle='--', color='blue', label='RSM', linewidth=1.2)
247
248 ax.scatter(np.arange(0, 45, 5), fragility_table_RSM_1000_deg2.median()*100/12,
249           s=10, facecolors='none', edgecolors='blue', linewidth=1)
250
251 ax.plot(np.linspace(0, 40, 100), NLTHA_BM_frag*100, linestyle='-',
252         color='red', label='Benchmark', linewidth=1.5)
253 ax.scatter(np.arange(0, 45, 5), NLTHA_BM_pivot.median()*100, s=5, color='red')
254 ax.axhline(50, color='goldenrod', alpha=0.4, linestyle='-.')
255
256 ax.set_xlim(0, 40)
257 ax.set_ylim(0, 100)
258 ax.yaxis.set_major_formatter(pticks)
259 ax.set_xlabel('Intensity measure (cm/s)')
260 ax.set_ylabel(r"$P(d>D|IM=x)$")
261 ax.legend()
262
263 fmt = '%.0f%%' # Format you want the ticks, e.g. '40%'
264 yticks = mtick.FormatStrFormatter(fmt)
265 ax.yaxis.set_major_formatter(yticks)
266
267 # do not forget to make the straight line between 0 and 5
268 plt.savefig('finalfrag_RSM.pdf', backend='pdf', dpi=600,
269           bbox_inches="tight", transparent=True)
270
271 fragility_table_RSM_1000_deg2

```

B.3 ANN fragility model python code

Developing fragility model for ANN approach.

```
1 NLTHA_ANN = pd.read_csv('https://github.com/mtareqsoliman/mstthesis/blob/main/
2 CaseStudy_1/sims_ANN_c1.csv?raw=true')
3
4 NLTHA_ANN_pivot = pd.pivot_table(NLTHA_ANN, values='state',
5 index=['ecu_alpha', 'Vult_alpha'],
6 columns=['IM'], aggfunc=np.sum)
7 NLTHA_ANN_pivot.insert(0,0,0)
8
9 ecu_ANN_alpha, Vult_ANN_alpha = zip(*NLTHA_ANN_pivot.index.values)
10 ecu_ANN_alpha = np.asarray(ecu_ANN_alpha)
11 Vult_ANN_alpha = np.asarray(Vult_ANN_alpha)
12 result_ANN_list, lambda_ANN_list, zeta_ANN_list = [], [], []
13
14 IM = np.arange(0,45,5)
15 fragility_table_ANN = pd.DataFrame(columns = ['0', '5', '10', '15', '20',
16 '25', '30', '35', '40'])
17 z_list = []
18
19 def frag(lambdazeta):
20     estimated_fragility = sp.norm.cdf(np.log(factor_list),
21 np.log(lambdazeta[0]), lambdazeta[1])
22     log_likelihood = np.log(sp.binom.pmf(pcllps*nrcrds,
23 nrcrds, estimated_fragility))
24     fun = -np.sum(log_likelihood)
25
26     return fun
27
28 for i, ecu_i in enumerate(ecu_ANN_alpha):
29     factor_list = np.array([0, 5, 10, 15, 20, 25, 30, 35, 40])
30     ncllps_NLTHA_ANN = NLTHA_ANN_pivot.iloc[i]
31     nrcrds = 12
32     pcllps = ncllps_NLTHA_ANN/nrcrds
33     # starting guess
34     lambdazeta_start = [20, 0.4]
35     # Bounds
36     bnds = ((5, 50), (0.2, 1))
37
38     result = spo.minimize(frag, lambdazeta_start,
39 options={'disp': True}, bounds=bnds)
40
41     if result.success:
42         miny = result.x
43         lambda_mxlk = round(miny[0], 2)
44         zeta_mxlk = round(miny[1], 2)
45         lambda_ANN_list.append(lambda_mxlk)
46         zeta_ANN_list.append(zeta_mxlk)
47
48         z_mxlk = -round(result.fun, 2)
49         z_list.append(z_mxlk)
50         fragility_table_ANN.loc[i] = sp.norm.cdf(np.log(IM),
51 np.log(lambda_mxlk), zeta_mxlk)
52
53 lambda_ANN_list = np.array(lambda_ANN_list)
54 zeta_ANN_list = np.array(zeta_ANN_list)
```

```

56
57 print(pd.DataFrame({"eps":ecu_ANN_alpha, "v":Vult_ANN_alpha,
58                     "$lambda$":lambda_ANN_list, '$zeta$':zeta_ANN_list,
59                     'cost':z_list}, index=None))
60
61 from sklearn.neural_network import MLPRegressor
62
63
64 x_BM = np.column_stack((ecu_BM_alpha_frag, Vult_BM_alpha_frag))
65 y_BM = np.column_stack((lambda_BM_list, zeta_BM_list))
66
67 ecu_ANN_alpha, Vult_ANN_alpha = map(list, zip(*NLTHA_ANN_pivot.index.values))
68 x_ANN = np.column_stack((ecu_ANN_alpha, Vult_ANN_alpha))
69 y_ANN = np.column_stack((np.log(lambda_ANN_list), zeta_ANN_list))
70
71 nneurons = 7
72 nlyrs = 3
73
74 kfold = model_selection.KFold(n_splits=7, random_state=2023, shuffle=True)
75 regr_ANN = MLPRegressor(hidden_layer_sizes=(5, 3),
76                         solver='lbfgs', alpha=0.001, activation='logistic')
77 regr_ANN.fit(x_ANN, y_ANN[:, 0])
78 kfold_2 = model_selection.KFold(n_splits=7, random_state=2023, shuffle=True)
79 regr_ANN_zeta = MLPRegressor(hidden_layer_sizes=(5, 3),
80                             solver='lbfgs', alpha=0.001, activation='logistic')
81 regr_ANN_zeta.fit(x_ANN, y_ANN[:, 1])
82
83 y_pred_ANN_lambda = regr_ANN.predict(x_ANN)
84 y_pred_ANN_zeta = regr_ANN_zeta.predict(x_ANN)
85 y_pred_ANN = np.column_stack((y_pred_ANN_lambda, y_pred_ANN_zeta))
86
87 score_kfold = model_selection.cross_val_score(regr_ANN, x_train_ANN,
88                                              y_train_ANN, cv=kfold)
89
90 print(score_kfold)
91
92
93 x1_mesh_ANN, x2_mesh_ANN = np.linspace(-2, 2, 50), np.linspace(-2, 2, 50)
94 x1_mesh_ANN, x2_mesh_ANN = np.meshgrid(x1_mesh_ANN, x2_mesh_ANN)
95 X_mesh_ANN = np.array([x1_mesh_ANN.flatten(), x2_mesh_ANN.flatten()]).T
96 y_mesh_ANN_pred_lambda = regr_ANN.predict(X_mesh_ANN)
97 y_mesh_ANN_pred_zeta = regr_ANN_zeta.predict(X_mesh_ANN)
98 y_mesh_ANN_pred = np.column_stack((y_mesh_ANN_pred_lambda, y_mesh_ANN_pred_zeta))
99
100 fig, ax = plt.subplots(figsize=(width*0.8/72.27, width*0.8/72.27),
101                          subplot_kw={'projection': '3d'})
102 ax.autoscale(enable=True, axis='x', tight=True)
103 ax.set_zlabel(r"$\lambda$ (cm/s)", labelpad=-5)
104 ax.set_xlabel(r"$\alpha_\epsilon$", labelpad=-5)
105 ax.set_ylabel(r"$\alpha_v$", labelpad=-5)
106 ax.set_title("(a)")
107 ax.tick_params(axis='both', which='major', pad=-2)
108 ax.scatter(x_ANN[:, 0], x_ANN[:, 1], np.exp(y_ANN[:, 0]),
109           color='black', marker='.', alpha=1, s=10)
110 sc_1 = ax.scatter(x1_mesh_ANN, x2_mesh_ANN, np.exp(y_mesh_ANN_pred[:, 0]),
111                  c=np.exp(y_mesh_ANN_pred[:, 0]),
112                  marker=r"$\u25EF$", s=5, cmap='autumn', alpha=0.35)
113 ax.azim = 225
114 ax.set_box_aspect(aspect=None, zoom=0.95)

```

```

115
116
117 plt.savefig('ANN_surf_lambda.pdf',
118             backend='pdf', dpi=600, bbox_inches="tight")
119
120 fig, ax = plt.subplots(figsize=(width*0.8/72.27,width*0.8/72.27),
121                          subplot_kw={'projection': '3d'})
122
123 ax.set_zlabel(r"$\zeta$",labelpad=-5)
124 ax.set_xlabel(r"$\alpha_\epsilon$",labelpad=-5)
125 ax.set_ylabel(r"$\alpha_v$",labelpad=-5)
126 ax.set_title("(b)")
127 ax.tick_params(axis='both', which='major', pad=0)
128 ax.scatter(x_ANN[:,0], x_ANN[:,1], y_ANN[:,1],
129            color='black',marker='.',alpha=1,s=10)
130 sc_2 = ax.scatter(x1_mesh_ANN, x2_mesh_ANN, y_mesh_ANN_pred[:,1],
131                  c=y_mesh_ANN_pred[:,1],
132                  marker="$\u25EF$", s=5, cmap='autumn',alpha=0.55)
133 ax.azim = 225
134 ax.set_box_aspect(aspect=None, zoom=0.95)
135
136 plt.savefig('ANN_surf_zeta.pdf', backend='pdf',
137             dpi=600, bbox_inches="tight")
138
139
140
141 result_table = pd.DataFrame({'y_act':y_ANN.tolist(),
142                             'y_pred':y_pred_ANN.tolist()
143                             }).to_excel('ssss.xlsx')
144
145 IM = np.arange(0,45,5)
146 fragility_table_ANN = pd.DataFrame(columns = ['0','5','10','15','20',
147                                             '25','30','35','40'])
148 z_list = []
149
150 for i in np.arange(len(y_ANN)):
151     fragility_table_ANN.loc[i] = sp.norm.cdf(np.log(IM),y_ANN[i,0],y_ANN[i,1])
152
153
154 alpha_ecu_1000 = np.random.normal(0, 1, 1000)
155 alpha_Vult_1000 = np.random.normal(0, 1, 1000)
156
157 x_1000_ANN = np.array([alpha_ecu_1000, alpha_Vult_1000]).T
158 y_1000_ANN_lambda = regr_ANN.predict(x_1000_ANN)
159 y_1000_ANN_zeta = regr_ANN_zeta.predict(x_1000_ANN)
160 y_1000_ANN = np.column_stack((y_1000_ANN_lambda,y_1000_ANN_zeta))
161
162 fragility_table_ANN_1000_mesh = pd.DataFrame(columns = ['0','5','10','15','20',
163                                                         '25','30','35','40'])
164
165 for i in np.arange(len(y_1000_ANN)):
166     fragility_table_ANN_1000_mesh.loc[i] = sp.norm.cdf(np.log(IM),
167                                                         y_1000_ANN[i,0],
168                                                         y_1000_ANN[i,1])
169
170 fragility_table_ANN_1000_mesh
171
172 import math
173 def frag(lambdazeta):

```

```

174     estimated_fragility = sp.norm.cdf(np.log(factor_list),
175                                     np.log(lambdazeta[0]), lambdazeta[1])
176     log_likelihood = np.log(sp.binom.pmf(ncllps_ANN1000_mesh, nrcrds,
177                                     estimated_fragility))
178     fun = -np.sum(log_likelihood)
179
180     return fun
181
182 factor_list = np.array([0, 5, 10, 15, 20, 25, 30, 35, 40])
183 nrcrds = 12
184 ncclps_ANN1000_mesh = np.round(fragility_table_ANN_1000_mesh.median()*12)
185 pcllps_ANN1000 = ncclps_ANN1000_mesh/12
186
187 # starting guess
188 lambdazeta_start = np.array([20, 0.4])
189
190 # Bounds
191 bnds = ((5, 50), (0.2, 1))
192
193 result = spo.minimize(frag, lambdazeta_start, options={'disp': True}, bounds=bnds)
194 ANN1000_mesh_lgnrml = []
195 if result.success:
196     print("Success!", end = '\n')
197     miny = result.x
198     ANN1000_mesh_lambda = round(miny[0], 2)
199     ANN1000_mesh_zeta = round(miny[1], 2)
200     ANN1000_mesh_cost = -round(result.fun, 2)
201
202 ANN1000_mesh_lz = [ANN1000_mesh_lambda, ANN1000_mesh_zeta, ANN1000_mesh_cost]
203 ANN1000_mesh_lgnrml = sp.norm.cdf(np.log(np.linspace(0, 40, 1000)),
204                                   np.log(ANN1000_mesh_lz[0]),
205                                   ANN1000_mesh_lz[1])
206
207 print(result)
208
209 np.savetxt('ANN1000_lz.txt',
210           ANN1000_mesh_lz, delimiter=',')
211 np.savetxt('ANN1000_points.txt',
212           fragility_table_ANN_1000_mesh.median()*12, delimiter=',')
213
214 ANN1000_mesh_lz
215
216 from scipy.interpolate import interp1d
217
218 for y, sel_IM, title in zip([0, 1, 2, 3], ["10", "20", "30", "40"], ['(a)', '(b)',
219                                                                    '(c)', '(d)']):
220     fig, ax = plt.subplots(figsize=(width*0.5/72.27, width*0.45/72.27))
221     fig.tight_layout(pad=4.0)
222     ax.hist(fragility_table_ANN_1000_mesh[sel_IM]*100, color='green',
223            edgecolor='darkgreen', alpha=0.55,
224            bins=np.arange(min(fragility_table_ANN_1000_mesh[sel_IM]*100),
225                           max(fragility_table_ANN_1000_mesh[sel_IM]*100) + 10,
226                           10))
227     ax.set_xlim(0, 100)
228     ax.xaxis.set_major_formatter(pticks)
229     ax.set_ylim(0)
230     ax.set_xlabel(r'$P(D>d|IM=x)$')
231     ax.set_ylabel('No. of realizations')
232     ax.set_title(title)

```

```

233     ax.axvline(np.median(fragility_table_ANN_1000_mesh[sel_IM]*100,axis=0),
234                color='darkgreen', linestyle='-.', linewidth=1.5)
235     plt.savefig('hist_ANN_'+str(sel_IM)+'.pdf', backend='pdf', dpi=600,
236                bbox_inches="tight", transparent=True)
237
238     fig, ax = plt.subplots(figsize=(width*1/72.27,width*0.55/72.27))
239
240     ax.scatter(np.arange(0,45,5),fragility_table_ANN_1000_mesh.median()*100,
241               linewidth=1,s=8,marker='x',color='green')
242     ax.plot(np.linspace(0,40,1000),ANN1000_mesh_lgnrml*100,linestyle='--',
243            color='green',label='ANN approach',linewidth=1,zorder=3)
244
245     fmt = '%.0f%%' # Format you want the ticks, e.g. '40%'
246     yticks = mtick.FormatStrFormatter(fmt)
247     ax.yaxis.set_major_formatter(yticks)
248     ax.legend()
249     ax.set_xlim(0,40)
250     ax.set_ylim(0,100)
251     ax.set_title('(c)')
252     ax.set_xlabel('Intensity measure (cm/s)')
253     ax.set_ylabel(r'$P(D>d|IM=x)$')
254     ax.axhline(50,color='goldenrod',alpha=0.4,linestyle='-.')
255     ax.legend()
256
257     plt.savefig('finalfrag_ANN.pdf', backend='pdf',
258                dpi=600, bbox_inches="tight", transparent=True)
259
260     #ANN1000_lgnrml
261
262     print(np.linspace(0,40,1000)[np.where(ANN1000_mesh_lgnrml >= 0.5)[0][0]])
263
264     fragility_table_ANN_1000_mesh.median()

```

CURRICULUM VITAE

Mohamed Tareq M. SOLIMAN

EDUCATION:

- **M.Sc.:** 2023, Istanbul Technical University (Turkiye), Graduate School, Structural Engineering.
- **B.Sc.:** 2018, Helwan University Cairo (Egypt), Faculty of Engineering Matariya, Civil Engineering.

PROFESSIONAL EXPERIENCE:

- 11.2019–01.2020 Project Engineer Intern at PMO NUST University, NUST University Islamabad, Pakistan.
- 10.2018–04.2019 Civil Engineer at Badawy Group R.E.D, Waterway New Cairo, Egypt.
- 2016 Civil Engineer Intern at ECG S.A, Mall of Arabia 6th of October, Giza, Egypt.
- 2015 Civil Engineer Intern at Arab Contractors, Al-Azhar University, Cairo, Egypt.

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **SOLIMAN M.T.M., YAZGAN U. (2022).** “Exploring the benefits and limitations of machine learning techniques in seismic fragility assessment”. *Proceedings of Dokuz Eylul University International Symposium on Graduate Research DEUISGR22-GeoScience*, December 8-9, 2022. Izmir, Turkey.