UNIVERSITY OF
THESSALY

# Load Identification & Response Reconstruction Using Vibrational Measurements

Ioannis Tetepoulidis

October 2025

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Submitted in partial fulfillment of the requirements for the degree of
Diploma in Mechanical Engineering at the University of Thessaly

Volos, 3rd of October 2025

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

UNIVERSITY OF
THESSALY

# Load Identification & Response Reconstruction Using Vibrational Measurements

Ioannis Tetepoulidis

October 2025

Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

2025 Ioannis Tetepoulidis

All rights reserved. The approval of the present Diploma Thesis by the department of Mechanical Engineering, School of Engineering, University of Thessaly, does not imply acceptance of the views of the author (Law 5343/32 art.202)



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## Approved by the Committee on Final Examination:

**Advisor:** Dr. Costas Papadimitriou,
Professor, Department of Mechanical Engineering, University of Thessaly

**Member:** Dr. Spyros Karamanos,
Professor, Department of Mechanical Engineering, University of Thessaly

**Member:** Dr. Filippos Filippitzis,
Postdoc Researcher, Department of Mechanical Engineering, University of Thessaly



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## Acknowledgments

First and foremost, I would like to express my deepest gratitude to my Professor, Dr. Costas Papadimitriou, for his invaluable guidance, profound ideas, and continuous scientific support throughout the preparation of my Diploma Thesis.

Thereafter, I am also sincerely thankful to my family, my parents, whose guidance and support made me the person I am today. I am also deeply thankful to my little sister, Mary-Helen, for the time he consumed to create the figures that exploited in the 5th Chapter.

Last, but certainly not least, I would like to thank my partner, Mary, for the psychological support all over this demanding period.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

**Load Identification & Response Reconstruction Using Vibrational Measurements**

**Ioannis Tetepoulidis**
Department of Mechanical Engineering, University of Thessaly

**Supervisor:** Dr. Costas Papadimitriou
Professor of Structural Dynamics

## Abstract

Structural Health Monitoring stands as a major challenge nowadays. Monitoring and identifying loads and responses could prevent structural degradation while reducing maintenance costs. Several frameworks have been developed to estimate inputs and responses, ranging from generic methods such as Ground Vibration Testing and substructure creation, to Data-Driven Methodologies exploiting Machine-Learning advancements and also Model-Driven Schemes such as Augmented and Dual Kalman Filters. Within this Diploma Thesis, two innovative formulations are presented, mainly aiming at minimizing an Objective Function. There is also proposed an algorithm derived from the second suggested Formulation aiming at estimating unknown input-parameters such as springs. These two formulations were be applied to a Spring-Mass Chain Model. Ten different cases were examined, simulating sinusoidal and double-impact excitations and varying sensor configurations. Results were compared against Augmented Kalman Filter Schemes. Testing indicated an advantage of the proposed formulations over Augmented Kalman Filter for acceleration only sensor installation. Collocated sensors configuration was also advantageous, mainly at degrees of freedom of the model far from the installed sensors. Modal Analysis showed remarkable accuracy at all cases, especially compared to all-mode analysis. Furthermore, regarding input-parameter estimation methodology, sufficient number of strain sensors proved vital for valid predictions. Augmented Kalman Filter methodology was less affected than proposed formulations by ineffective input-parameter estimation. Finally, testing of the formulations to military Aircraft is suggested in order to verify their robustness.

**Keywords:** Load Identification, Response Reconstruction, Modal Analysis, Input-Parameter Estimation, Augmented Kalman Filter, Spring-Mass Chain Model



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Contents

**List of Figures** **VIII**

**List of Tables** **X**

**1 Introduction** **1**

**2 Literature Review** **3**

2.1 Generic Schemes . . . . . 3

2.2 Substructure Approaches . . . . . 3

2.3 Data-Driven Methodologies . . . . . 4

2.4 Model-Driven Frameworks . . . . . 4

**3 Mathematical Background** **11**

3.1 Preliminaries . . . . . 11

3.2 Modal Analysis . . . . . 13

3.3 Model & Measurement Error . . . . . 15

3.3.1 Measurement Error . . . . . 15

3.3.2 Model Error . . . . . 15

3.4 Augmented Kalman Filters . . . . . 16

3.5 Proposed Formulation 1 . . . . . 19

3.6 Proposed Formulation 2 . . . . . 20

3.6.1 Linear System . . . . . 20

3.6.2 Augmented System . . . . . 21

3.6.3 Input-State Estimation . . . . . 22

3.7 Input-State-Parameter Estimation . . . . . 23



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

**4 Application in Spring-Mass Chain Model** **24**

4.1 Spring-Mass Model Presentation . . . 25

4.2 $\sigma_p = 10^{\gamma}$ Automated Estimation Technique From L-curve . . . 26

4.3 Software used . . . 26

4.4 Hardware Used . . . 26

4.5 Application Schematic Illustration . . . 27

**5 Results** **28**

5.1 Case 1 . . . 29

5.2 Case 2 . . . 32

5.3 Case 3 . . . 35

5.4 Case 4 . . . 38

5.5 Case 5 . . . 41

5.6 Case 6 . . . 45

5.7 Case 7 . . . 49

5.8 Case 8 . . . 52

5.9 Case 9 . . . 55

5.10 Case 10 . . . 59

**6 Conclusion** **63**

6.1 Deductions . . . 63

6.2 Suggestions for Future Research . . . 64

**References** **65**

**A Automated Estimation of $\sigma_p = 10^{\gamma}$ via L-curve Curvature** **69**

**B Matlab Scripts** **70**

B.1 Main Program . . . 70

B.2 Functions . . . 80

B.2.1 Term Initialization . . . 81

B.2.2 State-Space & Observation Matrices . . . 86

B.2.3 Error Calculations . . . 88

B.2.4 Input Parameter Estimation . . . 89

B.2.5 Response Reconstruction & Load Identification Formulations . . . 96

B.2.6 Figures . . . 99



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# List of Figures


4.1 Spring-Mass Chain Model . . . . . 25

4.2 Experimental Procedure Flowchart. . . . . 27

5.1 Spring-Mass Model - Case 1 . . . . . 29

5.2 L-curve for error 0.10 - Case 1 . . . . . 30

5.3 Inputs for variable errors - Case 1 . . . . . 30

5.4 Displacement Plots at 2nd DOF - Case 1 . . . . . 31

5.5 Velocity Plots at 7th DOF - Case 1 . . . . . 31

5.6 Displacement Plots at 10th DOF - Case 1 . . . . . 31

5.7 Spring-Mass Model - Case 2 . . . . . 32

5.8 Inputs for variable errors - Case 2 . . . . . 33

5.9 Displacement Plots - Case 2 . . . . . 33

5.10 Strain Plots - Case 2 . . . . . 34

5.11 Velocity Plots - Case 2 . . . . . 34

5.12 Spring-Mass Model - Case 3 . . . . . 35

5.13 Inputs for variable errors - Case 3 . . . . . 36

5.14 Displacement Plots at 1st DOF - Case 3 . . . . . 37

5.15 Strain Plots at 1st DOF - Case 3 . . . . . 37

5.16 Velocity Plots at 1st DOF - Case 3 . . . . . 37

5.17 Spring-Mass Model - Case 4 . . . . . 38

5.18 L-curve for error 0.05 - Case 4 . . . . . 39

5.19 Input Plots for variable error - Case 4 . . . . . 39

5.20 Displacement Plots at 8th and 10th DOF for error 0.05 - Case 4 . . . . . 40

5.21 Strain Plots at 8th and 10th DOF for error 0.05 - Case 4 . . . . . 40

5.22 Spring-Mass Model - Case 5 . . . . . 41

5.23 $\underline{\theta}$-Convergence Plots for varying error - Case 5 . . . . . 42




Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

5.24 Input Plots for variable error - Case 5 . . . . . 43
5.25 Displacement Plots for error 0.10 - Case 5 . . . . . 43
5.26 Strain Plots for error error 0.10 - Case 5 . . . . . 43
5.27 Velocity Plots for error 0.10 - Case 5 . . . . . 44
5.28 Spring-Mass Model - Case 6 . . . . . 45
5.29 L-curve for error 0.10 - Case 6 . . . . . 46
5.30 Input Plots - Case 6 . . . . . 46
5.31 Displacement Plots for error 0.10 - Case 6 . . . . . 47
5.32 Strain Plots for error 0.10 - Case 6 . . . . . 47
5.33 Velocity Plots for error 0.10 - Case 6 . . . . . 48
5.34 Spring-Mass Model - Case 7 . . . . . 49
5.35 L-curve for error 0.10 - Case 7 . . . . . 50
5.36 Input Plots - Case 7 . . . . . 50
5.37 Displacement Plots - Case 7 . . . . . 51
5.38 Strain Plots - Case 7 . . . . . 51
5.39 Velocity Plots - Case 7 . . . . . 51
5.40 Spring-Mass Model - Case 8 . . . . . 52
5.41 Inputs Plots for variable errors - Case 8 . . . . . 53
5.42 Displacement Plots for error 0.10 - Case 8 . . . . . 54
5.43 Strain Plots for error 0.10 - Case 8 . . . . . 54
5.44 Velocity Plots for error 0.10 - Case 3 . . . . . 54
5.45 Spring-Mass Model - Case 9 . . . . . 55
5.46 L-curve for error 0.10 - Case 9 . . . . . 56
5.47 Input Plot for error 0.10 . . . . . 56
5.48 Displacement Plots for error 0.10 - Case 9 . . . . . 57
5.49 Strain Plots for error 0.10 - Case 9 . . . . . 57
5.50 Velocity Plots for error 0.10 - Case 3 . . . . . 58
5.51 Spring-Mass Model - Case 10 . . . . . 59
5.52 $\underline{\theta}$-Convergence Plots for variable error - Case 10 . . . . . 60
5.53 Input Plots for varying error - Case 10 . . . . . 61
5.54 Displacement Plots at 7th DOF - Case 10 . . . . . 61
5.55 Strain Plots at 7th DOF - Case 10 . . . . . 61
5.56 Velocity Plots for error 0.10 - Case 3 . . . . . 62



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# List of Tables


2.1 Literature Review on Structural Health Monitoring. . . . . 9

2.2 Chronological Distribution of References . . . . . 9

2.3 Type Reference Distribution . . . . . 9

2.4 Data Type of References . . . . . 9

4.1 Virtual Experiment Specifications . . . . . 24

5.1 $\Delta\theta\%$ Relative Error - Case 3 . . . . . 36

5.2 $\Delta\theta\%$ Relative Error - Case 5 . . . . . 42

5.3 $\Delta\theta\%$ Relative Error - Case 8 . . . . . 53

5.4 $\Delta\underline{\theta}\%$ Relative Error - Case 10 . . . . . 60




Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Chapter 1: Introduction

In our epoch, the idiomorphic issue of Load Identification and Response Reconstruction has attracted substantial interest in the scientific community. Aircraft, bridges, skyscrapers, racing cars, outer-space vehicles. In all of them, stresses, forces and strains are applied. Need for safe infrastructures lies through decades. For instance, there is vast literature about the assurance of safe structures in aviation, as it is referenced in (Wells & King, 1970), an almost sixty year Technical Order that fully demonstrates the issue of Aircraft Structure Integrity Program in United States Air Force military Aircraft. The danger that lies upon the aging of structures, civil or military cannot be ignored. One notable example which should not be neglected is the Hasselt Bridge collapse at March, 1938 (Hayes, 1996) due to high stresses. There are also several incidents in military and civil Aviation, like the Macchi MB 326 wing failure at 1990 (Clark & Jackson, 2010) and Aloha Airlines accident that occurred in April, 1990 (Pidaparti, 2006) (Wanhill, Molent, Barter, & Amsterdam, 2015). Herein, these incidents caused several casualties, excessive costs for maintenance purposes, reduced aircraft availability and traumatized public opinion about safety in infrastructures (Clark & Jackson, 2010).

It is an undeniable fact that the majority of these catastrophic events lacked a computational diagnostic tool for Load Identification & Resonse Reconstruction (Clark & Jackson, 2010). Nowadays, thanks to the increase in computer's calculation strength, there is a vast amount of literature, equipment and software to provide high-fidelity prognosis about Structural Health Monitoring. One recent and illustrative example of Structural Health Monitoring consists of the use of Ground Vibration Testing machines in the Hellenic Air Force airplanes in order to assure that pylons, missiles and launchers outside the available Configuration list are structurally safe during flight. (Ketseas, 2025). Ground Vibration Testing is a method also used in Civilian Aviation at countries such as Belgium & Spain apart from USA (Peeters, Hendricx, & Debille, 2009) with remarkable results. Their application method is mainly focused on specific parts of a structure that their loads need to be estimated and are being tested iteratively. However, taking into consideration the maintenance costs and the time pressure by the Aviation Industry, there arises the need for a quicker and equally valid method. In the pursuit of quick computational calculations there are several articles such as (Yang et al., 2023) & (Wan, Li, Huang, & Wang, 2014) which aimed to reduce computational requirements through the formation of substructures (Yang et al., 2023) or with Empirical methods such as Empirical Mode Decomposition (Wan et al., 2014). Apart from these methodologies, there are more sophisticated suggestions such as Data-driven methods, namely Support Vector Machine, Deep Learning & Artificial Neural Networks (ANN) (Fu et al., 2023) or approaches originated towards Control Theory such as Dual Kalman Filters



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

(Azam, Chatzi, & Papadimitriou, 2015; Lourens, Reynders, Roeck, Degrande, & Lombaert, 2011). There will be a more thorough demonstration of these techniques in the Literature Review Section.

The motivation behind this Diploma Thesis is the author's intent to contribute to the ongoing research in load identification and response reconstruction with a quick, solid and valid technique that minimizes the computational claim. More specifically, firstly an extensive research in the literature, foreign or domestic is conducted in order to demonstrate the current direction of scientific community towards the approaches of Load Identification & Response Reconstruction. Secondly, the Theoretical Background is analyzed, firstly through mathematical analysis of the proposed formulation and the already known Kalman filters as the basis for a comparison. Modal Analysis revision will also be conducted for clarification purposes. Thereafter, through the chapter of application in Spring-Mass Chain Model (fourth), the experiment presentation will be addressed, performed with simulated data. Afterwards, the Results will be displayed, divided into 10 different cases. Finally, in the last chapter consists of two sections, the first one referring to the deductions made from the experiments that granted as useful to the scientific field. There is also a section about future propositions, applied and theoretical, for future research in order to extent knowledge and experience about Load Identification and Response Reconstruction.


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Chapter 2: Literature Review

Within this Chapter, there will be a thorough presentation of the amount of methodologies that contribute to Response Reconstruction & load identification of either simple structures or large scale ones. Herein, Literature Review originates through some generic methods, frameworks regarding substructure synthesis, Data-Driven Techniques based mainly on Machine Learning and on Model-Driven propositions in whom Bayesian Statistics are involved.

## 2.1 Generic Schemes

To begin with, a brief illustration of some generic methodologies cannot omit the Ground Vibration Testing methodologies. More explicitly, (Ketseas, 2025) & (Peeters et al., 2009) articles enlighten step by step the procedure on regarding Aviation Structural Health Monitoring. Ground Vibration Testing framework can be definitely combined with methods proposed later such as Machine Learning or Model-Driven ones. Additionally, a practical challenge that arises from this scheme lies upon the fact that re-installing of the sensors need to be performed in order to achieve optimal monitoring.

Furthermore, remaining within the generic framework part of the Literature Review, (Zhu, Du, & Tang, 2023) exploits the inverse Finite Element Method with decent results, although reconstructing strains and displacements. (Zhu et al., 2023). Through numerical data, (Jayalakshmi, Lakshmi, & Rao, 2020) & (Karlsson, 1996) attempted through sophisticated algorithms to accomplish impact load prognosis even with noise contaminated data (Jayalakshmi et al., 2020). Simultaneously, (Kantouris, 2024) and (Giagopoulos et al., 2019) performed experimental researches in miscellaneous structures (Airfoil, Power Plant) via optimization Algorithms (Covariance Matrix Adaptation Evolution Strategy) and Finite Element Models in order to estimate fatigue damage (Giagopoulos et al., 2019) and impact loads (Kantouris, 2024).

## 2.2 Substructure Approaches

Indisputably, substructure creation frameworks deserve a fair share in this review. Mainly, these methodologies aim at simplifying the structure in order to achieve less computational costs (Yang et al., 2023). (Wan et al., 2014) & (Yang et al., 2023) launched the Empirical Mode Decomposition scheme gathering simulated/numerical data to accomplish response reconstruction. Similar structures followed (Dollon, 2023) amalgamated with Bayesian Statistics proving that model updating is solid even with fewer DOF used. (Daitong, Yugang,



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Hongkun, & Xiaowen, 2023) also used the substructure methodology in rectangular coordinates to fulfill full-field response reconstruction. (Papadimitriou & Papadioti, 2013) performed familiar schemes in a Hellenic Bridge to verify the computational save without compromising the accuracy of his Finite Element Model. Finally, the book of (Jensen & Papadimitriou, 2019) is cited to provide the opportunity for the reader to deepen his/her knowledge.

## 2.3 Data-Driven Methodologies

With the computer advancements and extensive data acquisition opportunities nowadays, data-driven approaches have emerged. (Seventekidis & Giagopoulos, 2021, 2022, 2023; Seventekidis, Giagopoulos, Arailopoulos, & Markogiannaki, 2020), published a series of articles on Structural Health Monitoring and damage identification. They fused simulated and real data via a Finite Element Model along with Convolutional Neural Network technique. (Seventekidis et al., 2020) and (Seventekidis & Giagopoulos, 2023) had some challenges to experimentally validate their models. (Seventekidis & Giagopoulos, 2021, 2022) concluded their articles with the necessity for solid dataset in order for their models to succeed in damage classification. Neural Networks are extremely popular among the scientists as pointed out in the articles of (Jiang, Han, Du, & Ni, 2021; Koutsoupakis, Seventekidis, & Giagopoulos, 2023; Sarego, Zaccariotto, & Galvanetto, 2017) regarding an airfoil structure, a transmission structure and a footbridge in China, respectively. Additionally, a Deep Learning approach for load identification was proposed from (Rosafalco, Manzoni, Mariani, & Corigliano, 2021), with the Nearest Neighbor technique applied in order to reduce the computer cost. Last but not least, the articles of (Li et al., 2025) & (Fu et al., 2023) are excellent Literature Reviews that extensively illustrate not only data-related methodologies, but also perform mathematical analysis to provide an insight to the interested reader (Li et al., 2025).

## 2.4 Model-Driven Frameworks

Besides the aforementioned frameworks, model related schemes posses the biggest share of scientific interest and rely more to physical and mathematical models. More specifically, (Papadimitriou, Fritzen, Kraemer, & Ntotsios, 2011), (Zhi, Li, & Fang, 2016) & (C. D. Zhang & Xu, 2016, 2017; X. H. Zhang & Wu, 2019; X. H. Zhang, Zhu, Yuan, & Zhu, 2021), consist a series of journal articles that originate towards response reconstruction and Damage identification via Kalman Filter approach. (C. D. Zhang & Xu, 2016) primary researched optimal sensor placement in order to obtain the highest estimation accuracy. Thereafter, he researched the cases of unknown measurements (C. D. Zhang & Xu, 2017) and added measurement noise covariance with also solid results. (X. H. Zhang & Wu, 2019). Later on, (X. H. Zhang et al., 2021) experimented with real and simulated dataset on the optimal mode selection procedure claiming that response reconstruction with less modes can even obtain more accurate results. (Zhi et al., 2016) is a more of an independent venture, aiming at Wind Load consequences at skyscrapers. (Papadimitriou et al., 2011) implemented Kalman Filter in order to monitor the remaining life span of metallic trusses. Moreover, generalized Bayesian Statistics


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

methodologies are referenced in (Z. Chen et al., 2023) & (Hu, Zhou, Chen, & Weng, 2023) for damage identification associated with the uncertainties specter. (Wang, Peng, Xie, Chen, & Du, 2021) and (Liu & Wang, 2023) are based on regularization propositions performing both numerical and experimental instances to strengthen their claims on load identification. Additionally, remaining to the Response Reconstruction domain, (Z. Chen et al., 2023), (Ni, Li, Han, Xu, & Du, 2023) & (Li, Wang, Xia, & Sun, 2023) performed several numerical and experimental researches in conjunction with Bayesian updating approaches. (Li et al., 2023) accomplished decent accuracy even with high noise values. (Z. Chen et al., 2023) proved his framework was robust even with several practical challenges such as varying acting forces in a complex structure.

Last but not least, the articles of (Azam et al., 2015; Azam, Chatzi, Papadimitriou, & Smyth, 2017; Lourens et al., 2011), are worth mentioning not only for their scientific value, but also for their connection to this Diploma Thesis. More precisely, in all of these articles input and output estimation is performed. (Azam et al., 2015) applied dual Kalman Filter scheme in simulated examples with unknown inputs for load identification and response reconstruction, setting it in opposition with Dual Kalman Filter, Augmented Kalman Filter and Gillijns and De Moor Filter (GDF). Better scores in accuracy in displacement assessment recorded albeit with process covariance prediction. In (Azam et al., 2017) the authors claimed effective avoidance of the drift occurrence for Dual & Augmented Kalman Filter. What is more, Dual Kalman Filter dominated the rest approaches when data quality is assured. According to (Lourens et al., 2011), Dynamic Programming Algorithm was proposed, Kalman Filter and mainly Augmented Kalman Filter surpasses Dynamic Programming in case of collocated sensors installation. On the other hand, simulated and experimental designs referred on (Azam et al., 2015, 2017; Lourens et al., 2011) concluded prediction issues for Augmented Kalman Filter, Dual Kalman Filter and GDF models when sensors are installed far from the applied load and also when low-frequency affects the data, compromising the flexibility of these models in practical terms.

To sum up, all the scientific research referenced above such as journal articles, papers, Diploma Thesis, articles in the Internet are also illustrated in the Table 2.1 beneath for clarification purposes in case any of the readers wish to expand their knowledge on response reconstruction & load identification.

| **Literature Review on Structural Health Monitoring:** | | | | |
|---|---|---|---|---|
| **Numb.** | **Reference** | **Year** | **Ref. Type** | **Comments/Remarks** |
| 1 | (Azam et al., 2015) | 2015 | Journal Article | Dual Kalman Filter, Load Identification & Response Reconstruction, Unspecified Loads |
| 2 | (Azam et al., 2017) | 2017 | Journal Article | Augmented Kalman Filter, DKalman Filter, GDF, Load Identification, Bayesian Schemes |



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

| Numb. | Reference | Year | Ref. Type | Comments/Remarks |
|---|---|---|---|---|
| 3 | (Z. Chen et al., 2023) | 2023 | Journal Article | Damage and Force Estimation, Laplace Approach, Chebyshev Polynomial, Second-Order Sensitivity, Bayesian Approaches |
| 4 | (X. Chen et al., 2024) | 2024 | Journal Article | Model-based Response Reconstruction, Matrix Regularization Technique, FEM, Equivalent Load Identification |
| 5 | (Daitong et al., 2023) | 2023 | Journal Article | Modal Reduction Technique, Blade Tip Timing, FE |
| 6 | (Dollon, 2023) | 2023 | Journal Article | Modal Reduction Technique, ROM, FE, Bayesian Updating, Gibbs Sampler |
| 7 | (Fu et al., 2023) | 2023 | Journal Article | Literature Review,Types of Sensors, Dynamic Load Localization Techniques, Data Driven Techniques for Dynamic RE |
| 8 | (Giagopoulos et al., 2019) | 2019 | Journal Article | FE, Fatigue Damage & Remaining Lifetime Estimation, Optimization Algorithm |
| 9 | (Hu et al., 2023) | 2023 | Journal Article | Bayesian model, AI, Improved Particle Swarm Optimization Algorithm |
| 10 | (Jayalakshmi et al., 2020) | 2020 | Journal Article | Constraint Optimization Problem, Impact Load RE, Multiple Impact Loads |
| 11 | (Jensen & Papadimitriou, 2019) | 2019 | Book | Uncertainties, Modal Reduction Techniques Theory, Reduced-Order Models, Reliability Analysis, Bayesian FE Model Updating |
| 12 | (Jiang et al., 2021) | 2021 | Journal Article | Deep Learning, RNN, Continuous Signal, Short-term Dependencies, |
| 13 | (Kantouris, 2024) | 2024 | Diploma Thesis | Optimization algorithm, Simulated Data, Real Data, FE |


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

| Numb. | Reference | Year | Ref. Type | Comments/Remarks |
|---|---|---|---|---|
| 14 | (Karlsson, 1996) | 1996 | Journal Article | Optimization Algorithm, Sensitivity Analysis, Force Identification |
| 15 | (Ketseas, 2025) | 2025 | Web Page | GVT Illustration, FRF |
| 16 | (Koutsoupakis et al., 2023) | 2023 | Journal Article | Machine Learning, Convolutional NN, Damage Identification, Multi-body Dynamics System |
| 17 | (Li et al., 2025) | 2025 | Journal Article | Literature Review, on Load Identification & Response Reconstruction, Applications of DT, Novel Sensing Technologies, |
| 18 | (Li et al., 2023) | 2023 | Journal Article | Load Identification & Response Reconstruction, Bayesian Statistics |
| 19 | (Liu & Wang, 2023) | 2023 | Journal Article | Stochastic Excitation Identification, Chebyshev Model, 2 Step Regularization Method |
| 20 | (Lourens et al., 2011) | 2011 | Journal Article | Augmented Kalman Filter, Dynamic Programming Algorithm, Force Identification, Experimental Modal Analysis, FEM |
| 21 | (Ni et al., 2023) | 2023 | Journal Article | Response Reconstruction, Bayesian Inference, Sub-structural Technique, Stochastic Gradient Descent, FEM, Update |
| 22 | (Papadimitriou et al., 2011) | 2011 | Journal Article | Damage Identification, Power Spectral Density, Kalman Filter, Fatigue Predictions |
| 23 | (Papadimitriou & Papadioti, 2013) | 2013 | Journal Article | Craig-Bampton CMS, SSA, Dynamic Reduction Techniques |
| 24 | (Peeters et al., 2009) | 2009 | Report | GVT, Software Advancements, Aviation |
| 25 | (Rosafalco et al., 2021) | 2021 | Journal Article | Load Identification, Deep Learning, False Nearest Neighbor, Auto Encoder Framework |


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

| Numb. | Reference | Year | Ref. Type | Comments/Remarks |
|---|---|---|---|---|
| 26 | (Sarego et al., 2017) | 2017 | Conference Proceedings | Artificial NN, Force Reconstruction, FEM |
| 27 | (Seventekidis & Giagopoulos, 2022) | 2022 | Journal Article | FEM, Deep Learning, Damage Identification Classification |
| 28 | (Seventekidis et al., 2020) | 2020 | Journal Article | Deep Learning, FEM, Convolutional Neural Network, Damage Identification, Base Excitation |
| 29 | (Seventekidis & Giagopoulos, 2023) | 2023 | Journal Article | Supervised Damage Identification, FEM, Convolutional Neural Network, Deep Learning |
| 30 | (Seventekidis & Giagopoulos, 2021) | 2021 | Journal Article | FEM, Deep Learning, Convolutional Neural Network, Damage Identification Classification |
| 31 | (Wan et al., 2014) | 2014 | Journal Article | EMD, Inverse Optimization Algorithm, FE |
| 32 | (Wang et al., 2021) | 2021 | Journal Article | Load Identification, Iterative Regularization Algorithm |
| 33 | (Yang et al., 2023) | 2023 | Journal Article | EMD, Craig- Bampton CMS, Modal Synthesis, FE, |
| 34 | (C. D. Zhang & Xu, 2016) | 2016 | Journal Article | Damage Identification, Response Reconstruction, FEM, Modal Analysis, Kalman Filter, Optimal Sensor Placement, Kalman Filter |
| 35 | (C. D. Zhang & Xu, 2017) | 2017 | Journal Article | Damage Identification, Response Reconstruction, Kalman Filter, Modal Analysis, Response Sensitivity Updating, Radio Basis Function |
| 36 | (X. H. Zhang et al., 2021) | 2021 | Journal Article | Dynamic Response Reconstruction, Kalman Filter, FEM, Mode Selection |
| 37 | (X. H. Zhang & Wu, 2019) | 2019 | Journal Article | Response Reconstruction, Moving Window Kalman Filter, Unknown Measurement Noise Covariance |
| 38 | (Zhi et al., 2016) | 2016 | Journal Article | Wind Load Identification, Kalman Filter, Modal Analysis |


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

| Numb. | Reference | Year | Ref. Type | Comments/Remarks |
|---|---|---|---|---|
| 39 | (Zhu et al., 2023) | 2023 | Journal Article | Inverse FEM, Strain & Displacement Reconstruction, |

Table 2.1: Literature Review on Structural Health Monitoring.

| Chronological Distribution of References | |
|---|---|
| **Period** | **Number of References** |
| Before 2000 | 1 |
| 2000-2009 | 1 |
| 2010-2019 | 13 |
| 2020-Present | 24 |

Table 2.2: Chronological Distribution of References

| Type Reference Distribution | |
|---|---|
| **Type Reference** | **Number of References** |
| Journal Article | 34 |
| Conference Proceedings | 1 |
| Diploma Thesis | 1 |
| Book | 1 |
| Report | 1 |
| Web Page | 1 |

Table 2.3: Type Reference Distribution

| Data Type of References | |
|---|---|
| **Data Type** | **Number of References** |
| Simulated Data | 11 |
| Real/Experimental Data | 5 |
| Both | 18 |
| Other (Literature Review, Book etc.) | 5 |

Table 2.4: Data Type of References

Herein, from the Tables upwards, there are several qualitative conclusions to be extracted regarding this Chapter. More precisely, according to the Table 2.2, it is apparent that SHM receives growing attention, especially in the last two decades, mainly due to the advancement of computer speed plus the emergence of new mathematical approaches based on Machine



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Learning. Furthermore, as shown in 2.3 there is a tendency of the author towards the selection of Journal Articles (34 out of 39). Books, Reports Web Pages, Diploma Thesiss and Conference Proceedings pose a supportive role in the Literature Review. Finally, referring to 2.4, it is indisputable that the type of data exploited for the presented researches are normally distributed between experimental, simulated/numerical, both of them and other types such as Literature Review, Books etc.


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Chapter 3: Mathematical Background

Currently, the theoretical background of the formulations applied for (Load Identification) & Response Reconstructions is presented. In the first part, basic principles albeit with Modal Analysis & Model and Measurement Errors will be briefly presented. Secondly, AKF (Augmented Kalman Filter) and the proposed formulations will be analyzed. Finally, theoretical background for input parameter estimation will be examined. The articles of (Azam et al., 2015, 2017; Li et al., 2025; Lourens et al., 2011) & the book of (He & Fu, 2001) are some references that were studied for this Chapter.

## 3.1 Preliminaries

Firstly, equation of motion is as follows:

$$M\underline{\ddot{u}} + C\underline{\dot{u}} + K\underline{u} = R \cdot \underline{F}(t) \tag{3.1}$$

where:

- $M$ stands as the mass matrix.
- $C$ stands as the damping matrix.
- $K$ stands as the stiffness matrix.
- $\underline{u}$ stands as the displacement variable.
- $R$ stands as the excitation position matrix.
- $F(t)$ stands as the excitation function in continuous form.

The main scope is to use measurements from the structure to infer/estimate F(t). Equation (3.1) will be the starting point of every mathematical analysis in this Diploma Thesis.

Thereafter, the state-space in continuous form equation is written as follows:

$$\underline{\dot{x}}(t) = A_c \cdot \underline{x}(t) + B_c \cdot \underline{F}(t) \tag{3.2}$$

where:

- $\underline{x} = \begin{Bmatrix} \underline{u} \\ \underline{\dot{u}} \end{Bmatrix}, \quad \underline{u} \in R^n$



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

- n denotes the number of DOF.
- $A_c \quad \& \quad B_c$, $A_c \in R^{2n}$ & $B_c \in R^{2nxn_p}$ stand as state-space matrices in continuous form.
- $A_c = \begin{bmatrix} 0 & I \\ -M^{-1}K & -M^{-1}C \end{bmatrix}$
- $B_c = \begin{bmatrix} 0 \\ M^{-1}R \end{bmatrix}$
- $n_p$, stands as the number of excitations in the system.
- $I$ stands as the Identity Matrix.

Equation (3.2) can be discretized into:

$$\underline{x}_{k+1} = A \cdot \underline{x}_k + B \cdot \underline{F}_k \tag{3.3}$$

where

- k stands as the time instance of the system.
- $\underline{x}_k = \underline{x}(k\Delta t)$, is the state space variable for every time instance t.
- $\Delta t$, stands as the time step.
- $A = e^{A_c \Delta t}$ stands as the State Matrix in discrete form.
- $B = [A - I]A_c^{-1}B_c$ stands as the Input Matrix in discrete form.
- $\underline{F}_k = \underline{F}(k\Delta t)$ are the **unknown** values of the loads.

In general, the Observation equation in discrete form is written as:

$$\underline{\hat{y}}_k = C \cdot \underline{x}_k + D \cdot \underline{F}_k \tag{3.4}$$

where

- $\underline{\hat{y}}_k$ denotes the measured data at time instances $t = k\Delta t$
- C denotes the output state matrix, with $C = \begin{bmatrix} L & 0_{n_s \times n} \end{bmatrix} \in R^{n_s \times 2n}$ for displacement observations, where $L \in R^{n_s \times n}$ denotes the Boolean input position matrix. Each cell takes the value 1 in case of a sensor installation in the respective DOF, and the value 0 otherwise.
- D denotes the direct transmission matrix. For instance, $D = [0_{n_s \times n_p}]$ in case of displacement observations.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Herein, the $\underline{F}_k$ values are the loads which need to be estimated from measurements $\underline{\hat{y}}_k$.

The estimation will be accomplished through the error (measurement loss) between model predictions & measurements:

$$J = \sum_{k=1}^{N} \|C \cdot \underline{x}_k + D \cdot \underline{F}_k - \underline{\hat{y}}_k\|^2 \tag{3.5}$$

Consequently, equation (3.5) is transformed into:

$$J = \sum_{k=1}^{N} \|C \cdot \sum_{l=1}^{k-1} H_l \underline{F}_l + D \cdot \underline{F}_k - \underline{\hat{y}}_k\|^2 \tag{3.6}$$

Note that Equation (3.5) is a quadratic objective function.

## 3.2 Modal Analysis

It will cover the basic principles connected with the purpose of the Diploma Thesis. Mainly, theoretical analysis is based on (Azam et al., 2015; He & Fu, 2001; Li et al., 2025). More specifically, using Modal Analysis, equation (3.1) is defined as:

$$\ddot{\underline{\xi}} + C^* \cdot \dot{\underline{\xi}} + \Lambda \cdot \underline{\xi} = \Phi^T \cdot R \cdot \underline{F}(t) \tag{3.7}$$

I introduce: $\underline{x} = \begin{Bmatrix} \underline{\xi} \\ \dot{\underline{\xi}} \end{Bmatrix}$ which entails to:

$$\dot{\underline{x}} = A_c \cdot \underline{x} + B_c \cdot \underline{F}(t) \tag{3.8}$$

where $A_c = \begin{bmatrix} 0 & I \\ -\Lambda & -C^* \end{bmatrix}$, $B_c = \begin{Bmatrix} 0 \\ \Phi^T \cdot R \end{Bmatrix}$

Also, the displacements define as:

$$\underline{u} = \sum_{r=1}^{n_p} \Phi_r \underline{\xi}_r = \Phi \underline{\xi} \tag{3.9}$$

where:

- $\Phi = \begin{bmatrix} \underline{\Phi}_1 & \underline{\Phi}_2 & \dots & \underline{\Phi}_n \end{bmatrix}$ are the modal shapes.
- $\underline{\xi} = \begin{Bmatrix} \underline{\xi}_1 \\ \underline{\xi}_2 \\ \vdots \\ \underline{\xi}_n \end{Bmatrix}$, are coefficients in the main coordinates.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Once $\underline{\Phi}$ and $\underline{\xi}$ are available, then computation of $\underline{u}$ is possible:

$$\underline{u} = \sum_{r=1}^{n_p} \Phi_r \underline{\xi}_r = \Phi \underline{\xi} = \cdots = \begin{bmatrix} \underline{\Phi} & \underline{0} \end{bmatrix} \underline{x}$$

Then, from (3.7), the acceleration vector takes the form:

$$\underline{\ddot{u}} = \Phi \cdot \underline{\ddot{\xi}} = \Phi \begin{bmatrix} -\Lambda \underline{\xi} & -C^* \underline{\dot{\xi}} + \Phi^T \cdot R \cdot \underline{F}(t) \end{bmatrix} \Rightarrow$$

$$\Rightarrow \underline{\ddot{u}} = \begin{bmatrix} -\Phi \cdot \Lambda & -\Phi \cdot C^* \end{bmatrix} \begin{Bmatrix} \underline{\xi} \\ \underline{\dot{\xi}} \end{Bmatrix} + \Phi \Phi^T R \cdot \underline{F}(t) \Rightarrow$$

$$\Rightarrow \underline{\ddot{u}} = -\Phi \begin{bmatrix} \Lambda & C^* \end{bmatrix} \underline{x} + \Phi \Phi^T R \cdot \underline{F}(t) \tag{3.10}$$

From equation (3.2), we obtain for MA:

$$\underline{\hat{y}}_k = \underline{\ddot{u}} = C \cdot \underline{x} + D \cdot \underline{F_k} \tag{3.11}$$

where:

- $C = -\Phi \begin{bmatrix} \Lambda & C^* \end{bmatrix}$, as the modal output state matrix.
- $C^* = \begin{bmatrix} 2\zeta_1\omega_1 & 0 & \dots & 0 \\ 0 & 2\zeta_2\omega_2 & \dots & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & \dots & \dots & 2\zeta_n\omega_n \end{bmatrix}$, with $\zeta$ standing as the modal damping ratio.
- $\Lambda = \begin{bmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ \vdots & 0 & \ddots & 0 \\ 0 & \dots & \dots & \lambda_{n_p} \end{bmatrix}$, with $\lambda_i = \omega_i^2$ being the modal frequencies.
- $D = \Phi \Phi^T R$, being the modal direct transmission matrix.

If one knows the system, then:

$\Rightarrow$ **K, M, C, R** are known in (3.1).

$\Rightarrow$ $\boldsymbol{\Lambda}, \boldsymbol{C^*}, \boldsymbol{\Phi}, \boldsymbol{R}$ are known in (3.7).

$\Rightarrow$ $\boldsymbol{\Lambda}$ & $\boldsymbol{\Phi}$ are obtained by solving the eigenvalue problem, namely:

$$(K - \lambda_i M)\Phi_i = 0$$

.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 3.3 Model & Measurement Error

Model and Measurement Errors are considered in the articles of (Ni et al., 2023; Zhi et al., 2016). More specifically:

### 3.3.1 Measurement Error

Measurement Error is added to the output time histories simulated by the model (e.g. spring mass chain model). Let $y_i(k) = y_i(k\Delta t)$, where $t_k = k\Delta t$ is the time instant for which the measurement is available, k = 1,....,N, is the number of instances and i is the sensor number. Then, the measurement contaminated by a percentage of Gaussian error is obtained as:

$$\hat{y_i}(k) = y_i(k) + \epsilon y_{i,int} n_i(k) \tag{3.12}$$

where:

- $y_{i,int}$ is the intensity of the signal given by its root mean square value: $y_{i,int} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} y_i^2(k)}$
- $n_i(k)$ is a sample drawn from a Gaussian distribution with mean zero and standard deviation equal to 1, that is to say: $n_i(k) \sim N(0,1)$
- $\epsilon$ is the amount of error (e.g. $\epsilon = 0.02$ means that the error is 2% of the intensity of the signal).

In this Diploma Thesis, it will be used $n_i(k) = normrnd(0,1)$ in Matlab to sample $n_i(k) \sim N(0,1)$ and $u = normrnd(0,1,N,M)$ to generate a $NxM$ matrix of n Gaussian noise samples. The final y matrix is given by:

$$y = \begin{pmatrix} y_1(1) & y_1(2) & ... & y_1(N) \\ y_2(1) & y_2(2) & ... & y_2(N) \\ \vdots & \vdots & \ddots & \vdots \\ y_{N_0}(1) & y_{N_0}(2) & ... & y_{N_0}(N) \end{pmatrix}$$

where $N_0$ is the number of sensors and N is the number of time history values.

### 3.3.2 Model Error

The Model Error is used to account for the difference between the **nominal model** and the **actual system**. Any model is an approximation of the system. Data are collected from the system and the model is used to make analyses. To sum model error is mandatory to generate measurements from a perturbed model and not the nominal one used to make analyses. The perturbation is assigned to the system properties such as mass and stiffness characteristics. For example, if $k_j$ & $m_j$ are the stiffness and mass properties of a nominal spring-mass chain system, the perturbed model that will represent the actual system will have



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

stiffness and mass properties given by $\hat{k_j}$ & $\hat{m_j}$, where $\hat{k_j}$ and $\hat{m_j}$ are obtained by Gaussian perturbation as follows:

$$\hat{k_j} = k_j \cdot (1 + \epsilon_{stiff}\xi_j)$$
$$\hat{m_j} = m_j \cdot (1 + \epsilon_{mass}\eta_j)$$

where:

- $\xi_j$ $\eta_j$ are samples drawn from a Gaussian distribution with mean zero and standard deviation equal to 1:

$$\xi_j \sim N(0, 1)$$
$$\eta_j \sim N(0, 1)$$

- and $\epsilon_{stiff}$ & $\epsilon_{mass}$ are the amount of perturbation assigned to the stiffness and mass characteristics, respectively (e.g. $\epsilon_{stiff} = 0.02$ means that the error 2% is assigned to the stiffness properties and $\epsilon_{mass} = 0.01$ means that the error is 1%, assigned to the mass properties).

The artificial measurements are now generated by the actual system (the perturbed model) corresponding to the stiffness and mass properties $\hat{k_j}$ & $\hat{m_j}$, respectively. Consequently, model & measurement error can be implemented by generating simulated responses from the perturbed model and adding measurement error (noise) to the simulated response to generate the artificial measurements.

## 3.4 Augmented Kalman Filters

This technique is not within the scope of this Diploma Thesis but for correlation purposes only. The analysis is based on (Azam et al., 2017; Lourens et al., 2011). In general, AKF is extensively exploited to predict the characteristics, inputs and responses of structures (Azam et al., 2017). More specifically, beginning with the augmented state-space model equation (Lourens et al., 2011):

$$\underline{x}_{k+1} = A\underline{x}_k + B\underline{p}_k + \underline{w}_k \tag{3.13}$$

where:

- A & B stand as discrete state and input matrices, respectively.
- $\underline{w}_k$ denotes the noise vector as a component of the stochastic noise process $(\underline{w}_k \in R^n)_{k=0}^{\infty}$ where $n$ refers to the number of states (DOF).

Equation (3.13) is combined with:

$$\underline{p}_{k+1} = \underline{p}_k + \underline{\eta}_k \tag{3.14}$$

where:

- $\underline{p} \in R^{n_p}$ relates to the force vector at times k and k+1.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

- $\eta_k$ relates to a component of the stochastic process $(\underline{\eta}_k \in R^{n_p})_{k=0}^{\infty}$ where $n_p$ is the number of loads/inputs in the system.

Equation (3.14) permits the calculation of the force time history through a valid selection of the covariance matrix of the process $\eta_k$. By combining equations (3.13) , (3.14) and redefining the sate vector $\underline{x}^a$, where the superscript 'a' denotes the augmented. Consequently:

$$\underline{x}_k^a = \begin{bmatrix} \underline{x}_k \\ \underline{p}_k \end{bmatrix} \tag{3.15}$$

an augmented state equation is obtained:

$$\underline{x}_{k+1}^a = A_a \underline{x}_{k+1}^a + \underline{\zeta}_k \tag{3.16}$$

where:

- $A_a \in R^{(n+n_p)x(n+n_p)}$ is defined as:

$$A_a = \begin{bmatrix} A & B \\ 0 & I \end{bmatrix} \tag{3.17}$$

- The noise vector $\underline{\zeta} \in R^{(n+n_p)}$ is assumed unknown, accounts for the modeling errors $\underline{w}_k \in R^n$ and stands as:

$$\underline{\zeta}_k = \begin{Bmatrix} \underline{w}_k \\ \underline{\eta}_k \end{Bmatrix} \tag{3.18}$$

The classical discrete time observation equation with unknown noise vector $n_k$ becomes, in the augmented model:

$$\underline{d}_k = C_a \underline{x}_k^a + \underline{n}_k \tag{3.19}$$

where $\underline{d} \in R^{n_d}$ represents the measured data vector and the matrix $D_a \in R^{n_d \times (n+n_p)}$ is assembled from the output influence and direct transmission matrices C & D as below:

$$D_a = \begin{bmatrix} C & D \end{bmatrix} \tag{3.20}$$

To conclude, the state and observation equation defining the augmented state-space model are formulated as follows (Lourens et al., 2011):

$$\underline{x}_{k+1}^a = A_a \underline{x}_k^a + \underline{\zeta}_k \tag{3.21}$$

$$\underline{d}_k = C_a \underline{x}_k^a + \underline{v}_k \tag{3.22}$$

Error Covariance Matrix is defined as:

$$P_{k|l} = E[(\underline{x}_k^a - \underline{\hat{x}}_{k|l}^a)(\underline{x}_k^a - \underline{\hat{x}}_{k|l}^a)^T] \tag{3.23}$$

where:



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

- $\underline{\hat{x}}^a_{k|l}$ stands as an estimate of $\underline{x}^a_k$ given $\{\underline{d}_n\}^l_{n=0}$.
- $P_{k|l} \in R^{(n+n_p)\times(n+n_p)}$
- It is hypothesized that the initial conditions are known (Lourens et al., 2011).

Noise processes $\{\underline{w}_k \in R^n\}^\infty_{k=0}]$, $\{\underline{v}_k \in R^{n_d}\}^\infty_{k=0}]$ & $\{\underline{\eta}_k \in R^{n_p}\}^\infty_{k=0}$

> "are assumed to be stationary, mutually uncorrelated stochastic processes with zero mean. Their covariances are hypothesized as known and are represented by matrices $Q \in R^{n_s\times n_s}$, $R \in R^{n_d\times n_d}$ and $Q \in S^{n_p\times n_p}$, *respectively*" (Lourens et al., 2011) [1]

$$E\{\underline{w}_k\underline{w}^T_l\} = Q\delta_{k-l} \tag{3.24}$$

$$E\{\underline{v}_k\underline{v}^T_l\} = R\delta_{k-l} \tag{3.25}$$

$$E\{\underline{\eta}_k\underline{\eta}^T_l\} = S\delta_{k-l} \tag{3.26}$$

where $\delta_{k-l}$ is the Kronecker delta, $E\{\underline{w}_k\underline{w}^T_l\}$ stands as the independent Gaussian sequence & $Q$ is the covariance matrix.

$\underline{\hat{x_k}}$ is estimated using the following two steps:

1. Measurement update equations:

$$L_k = P_{k|k-1}C^T_a(C_aP_{k|k-1}C^T_a + R)^{-1} \tag{3.27}$$

$$\underline{\hat{x}}^a_{k|k} = \underline{\hat{x}}^a_{k|k-1} + L_k(\underline{d}_k - C_a\underline{\hat{x}}^a_{k|k-1}) \tag{3.28}$$

$$P_{k|k} = P_{k|k-1} - L_kC_aP_{k|k-1} \tag{3.29}$$

2. Time Update:

$$\underline{\hat{x}}^a_{k+1|k} = A_a\underline{\hat{x}}^a_{k|k} \tag{3.30}$$

$$P_{k+1|k} = A_aP_{k|k}A^T_a + Q_a \tag{3.31}$$

Note that taking into consideration the augmented noise vector $\zeta_k$ of equation (3.21), the below mentioned augmented covariance matrix $Q_a \in R^{(n_s+n_p)\times(n_s+n_p)}$ stands as:

$$Q_a = \begin{bmatrix} Q & 0 \\ 0 & S \end{bmatrix} \tag{3.32}$$

[1] Page 448, E. Lourens et al. / Mechanical Systems and Signal Processing 27 (2012) 446-460.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 3.5 Proposed Formulation 1

This subsection consists of the first of the two proposed formulations. More specifically, due to modal error, the equation of motion is not satisfied exactly. This means that, (3.3) is written as:

$$\underline{x}_{k+1} - A\underline{x}_k - B\underline{F}_k = \underline{e}_k \tag{3.33}$$

where $\underline{e}_k$ defines as the error in satisfying (3.3).

Instead of selecting $\underline{F}_k$ to minimize the residuals given from equations (3.3), (3.5), we solve a different optimization problem of selecting both $\underline{\boldsymbol{F}}_{\boldsymbol{k}}$ and $\underline{\boldsymbol{x}}_{\boldsymbol{k}}$ so as to minimize a combination of residuals in (3.5) and square of the errors in (3.33). That is to select $\underline{\boldsymbol{F}}_{\boldsymbol{k}}$ and $\underline{\boldsymbol{x}}_{\boldsymbol{k}}$ to minimize:

$$I(\underline{F}_k, \underline{x}_k) = \beta \cdot \sum_{k=1}^{N} \|e_k\|^2 + J(\underline{F}_k, \underline{x}_k)$$

which is equivalent written as:

$$I(\underline{F}_k, \underline{x}_k) = \beta \cdot \sum_{k=0}^{N-1} \|\underline{x}_{k+1} - A\underline{x}_k - B\underline{F}_k\|^2 + \sum_{k=1}^{N} \|C\underline{x}_K + D\underline{F}_k - \hat{y}_k\|^2 \tag{3.34}$$

where $\beta$ is a weighting factor that weighs the contribution of the second term in $I$, $J(\underline{F}_k, \underline{x}_k)$ is the objective function and $\|\dots\|^2$ is the Euclidean Norm.

Note that, $I(\underline{F}_k, \underline{x}_k)$ is a quadratic function of $\underline{\boldsymbol{F}}_{\boldsymbol{k}}$ and $\underline{\boldsymbol{x}}_{\boldsymbol{k}}$. Hence, the optimal values of $\underline{\boldsymbol{F}}_{\boldsymbol{k}}$ and $\underline{\boldsymbol{x}}_{\boldsymbol{k}}$ can be derived by analytical expressions to be developed as follows.

At the minimum, the following stationary equations define as:

$$\nabla_{\underline{z}} I = \underline{0}$$
$$\nabla_{\underline{F}_k} I = \underline{0}$$

which mean that gradient of I with respect to $\underline{\mathrm{F}}$ & gradient of $I$ with respect to $\underline{\mathrm{z}}$ are zero, respectively.

Continuing from (3.34), we obtain:

$$I(\underline{x}, \underline{F}) = \beta \cdot \sum_{k=0}^{N-1} (\underline{x}_{k+1} - A\underline{x}_k - B\underline{F}_k)^T (\underline{x}_{k+1} - A\underline{x}_k - B\underline{F}_k) + \sum_{k=1}^{N} (C\underline{x}_k + D\underline{F}_k - \hat{y}_k)^T (C\underline{x}_k + D\underline{F}_k - \hat{y}_k)$$

Partial derivatives with respect to $\underline{x}_k$:

$$\nabla_{\underline{x}_k} I = 2 \cdot \beta \cdot \sum_{k=0}^{N-1} \{-A^T (\underline{x}_{k+1} - A\underline{x}_k - B\underline{F}_k) + I^T (\underline{x}_k - A\underline{x}_{k-1} - B\underline{F}_{k-1})\} + 2 \cdot \sum_{k=1}^{N} C^T (C\underline{x}_k + D\underline{F}_k - \hat{y}_k) \Rightarrow$$



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

$$\Rightarrow \nabla_{\underline{x}_k} I = 0 \Rightarrow$$

$$-\beta \cdot A^T \underline{x}_{k+1} + \underline{x}_k[(C^T C + \beta \cdot A^T A) + \beta \cdot I] - A\underline{x}_{k-1} + (CD^T + \beta \cdot A^T B)\underline{F}_k - \beta \cdot B\underline{F}_{k-1} = C^T \underline{\hat{y}}_k \tag{3.35}$$

Partial derivatives with respect to $\underline{F}_k$:

$$\begin{aligned} \nabla_{\underline{F}_k} I = 0 \Rightarrow D^T(C\underline{x}_k + D\underline{F}_k - \hat{y}_k) - \beta \cdot B^T(\underline{x}_{k+1} - A\underline{x}_k - B\underline{F}_k) = 0 \Rightarrow \\ \Rightarrow -\beta B^T \underline{x}_{k+1} + (D^T C + \beta B^T A)\underline{x}_k + (D^T D + \beta B^T B)\underline{F}_k = D^T \hat{y}_k \end{aligned} \tag{3.36}$$

Setting $\underline{z}_{k+1} = \begin{Bmatrix} \underline{x}_{k+1} \\ \underline{F}_k \end{Bmatrix}$:

$$\begin{bmatrix} -A^T\beta & C^T D + \beta A^T B \\ -B^T B & D^T C + \beta B^T B \end{bmatrix} \underline{z}_{k+1} + \begin{bmatrix} C^T C + \beta A^T A + \beta I & -B \\ D^T C + \beta B^T A & \underline{0} \end{bmatrix} \underline{z}_k + \begin{bmatrix} -A & \underline{0} \\ \underline{0} & \underline{0} \end{bmatrix} = \begin{bmatrix} C^T \underline{\hat{y}}_k \\ D^T \underline{\hat{y}}_k \end{bmatrix} \tag{3.37}$$

- For $k = -1 \Rightarrow \underline{z}_0 = \begin{Bmatrix} \underline{x}_0 \\ \underline{F}_{-1} \end{Bmatrix}$, set $\underline{z}_0 = 0 \Rightarrow \begin{Bmatrix} \underline{x}_0 = 0 \\ \underline{F}_{-1} = 0 \end{Bmatrix}$
- For $k = 0 \Rightarrow \underline{z}_1 = \begin{Bmatrix} \underline{x}_1 \\ \underline{F}_0 \end{Bmatrix}$, set $\underline{F}_0 = 0$, use $\underline{x}_1 = A\underline{x}_0 + B\underline{F}_0$ which is the equivalent to equation (3.2) for $k = 0$.
- For the rest k values, (3.37) is used iteratively (from $k = 1$ to $k = N - 1$).

-

## 3.6 Proposed Formulation 2

### 3.6.1 Linear System

Consider the linear discrete state space equation

$$\underline{x}_{k+1} = A_x \underline{x}_k + B\underline{p}_k + \underline{e}_k^x \tag{3.38}$$

where $\underline{x} \in R^n$ is the state vector, $\underline{p}_k \in R^{n_p}$ is the load vector, $A_k \in R^{n \times n}$ is the state matrix, $B_k \in R^{n \times n_p}$ is the load to state matrix, and $\underline{e}_k^x \sim N(0, \Sigma_x)$ is the zero-mean model error with noise covariance $\Sigma_x = Q_x^{-1}$.

Assume that the load is unknown and consider a load model :

$$\underline{p}_{k+1} = A_p \underline{p}_k + \underline{e}_k^p \tag{3.39}$$

where $\underline{e}_k^p \sim N(0, \Sigma_p)$ with $\Sigma_p = Q_p^{-1}$ is the noise covariance for the load. Selecting $A_p = I$ then the load model is a random walk model.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

The observation equation is:

$$\underline{\hat{y}}_k = C_x \underline{x}_k + C_p \underline{p}_k + \underline{\eta}_k \tag{3.40}$$

where $C_x \in R^{n_s \times 2n}$ is the state to observation matrix, $C_p \in R^{n_s \times n_p}$ is the load to observation matrix, and $\underline{\eta}_k \sim N(0, \Sigma_\eta)$ is the measurement (and model) error term with covariance matrix $\Sigma_\eta = Q_\eta^{-1}$. The observed data $\underline{\hat{y}}_k \in R^{n_y}$ may consist of displacements, velocities, accelerations and/or strains, depending on the selection of $C_x$ & $C_p$ matrices.

### 3.6.2 Augmented System

Using the augmented state vector $\underline{z}_k = (\underline{x}_k^T \quad \underline{p}_k^T)^T$, and introducing the augmented noise term $\underline{e}_k = (\underline{e}_k^{xT} \quad \underline{e}_k^{pT})^T$, the augmented state space equation takes the form:

$$\underline{z}_{k+1} = A\underline{z}_k + \underline{e}_k \tag{3.41}$$

where the augmented state matrix $A$ takes the form:

$$A = \begin{bmatrix} A_x & B \\ 0 & A_p \end{bmatrix} \tag{3.42}$$

and the observation equation takes the form

$$\underline{\hat{y}}_k = C\underline{z}_k + \underline{\eta}_k \tag{3.43}$$

where $C = [C_x \quad C_p]$.

State to observation ($C_x$) & load to observation ($C_p$) matrices define as follows:

1. For acceleration Observations, given **force** excitation at the mass:
   - $C_x = \left[-L_{acc}M^{-1}K \quad -L_{acc}M^{-1}\tilde{C}\right] \in R^{n_s \times 2n}$
   - $C_p = \left[L_{acc}M^{-1}R\right] \in R^{n_s \times n_p}$
2. For acceleration Observations, given **base** excitation:
   - $C_x = \left[-L_{acc}M^{-1}K \quad -L_{acc}M^{-1}\tilde{C}\right] \in R^{n_s \times 2n}$
   - $C_p = \left[L_{acc}(-M + I)_{n_s \times n_s}\underline{1}_{n_s \times n_p}\right] \in R^{n_s \times n_p}$
3. For strain Observations, either **force** (at the mass) or **base** excitation:
   - $C_x = \left[L_{strain}L_e \quad 0_{n_s \times n}\right] \in R^{n_s \times 2n}$
   - $C_p = \left[0\right] \in R^{n_s \times n_p}$,

where $L_{acc} \in R^{n_s \times n}$ and $L_{str} \in R^{n_s \times n}$ are the Boolean Matrices that denote the number of installed acceleration & strain sensors. In case of an installed sensor, the respective cell takes the value 1. Moreover,

$$L_e = \begin{bmatrix} 1 & 0 & \dots & \dots & 0 \\ -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & \dots & \dots & -1 & 1 \end{bmatrix} \in R^{n \times n}$$

stands as the strain calculation matrix.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

### 3.6.3 Input-State Estimation

The augmented vector $\underline{z} = [\underline{z}_1^T, \underline{z}_2^T, \ldots, \underline{z}_N^T]^T$, consisted of the load and state vectors, are estimated by minimizing the sum of the measure of fit which is the error between the model-based predictions and the measurements, and the equation error which guarantees that the physical equations are satisfied. This results to the following objective function

$$J(\underline{z}) = J(\underline{x}, \underline{p}) = \sum_{k=1}^{N} (\underline{\hat{y}}_k - C\underline{z}_k)^T \Sigma_\eta^{-1} (\underline{y}_k - C\underline{z}_k) + \sum_{k=0}^{N-1} (\underline{z}_{k+1} - A\underline{z}_k)^T \Sigma_e^{-1} (\underline{z}_{k+1} - A\underline{z}_k) \quad (3.44)$$

where

$$\Sigma_e = \begin{bmatrix} \Sigma_x & 0 \\ 0 & \Sigma_p \end{bmatrix} \quad (3.45)$$

Setting $\Sigma_e^x = \sigma_x^2 I_x$, $\Sigma_p^x = \sigma_p^2 I_p$ and $\Sigma_\eta = \sigma_\eta^2 I_\eta$, then $Q_e^x = (1/\sigma_x^2) I_x$, $Q_p^x = (1/\sigma_p^2) I_p$ and $Q_\eta = (1/\sigma_\eta^2) I_\eta$. Also, the objective function (3.44) simplifies further.

Minimizing the objective function with respect to the augmented state vector, one obtains the following recursive equation for solving for the augmented state vector $\underline{z} = (\underline{z}_1^T, \underline{z}_2^T, \ldots, \underline{z}_N^T)$ given that the system is known and the covariance matrices $\Sigma_x$, $\Sigma_p$ and $\Sigma_\eta$ are also known. More specifically, setting the second sum term as $\sum_{k=1}^{N} (\underline{z}_k - A\underline{z}_{k-1})^T \Sigma_e^{-1} (\underline{z}_k - A\underline{z}_{k-1})$, $\Sigma_\eta^{-1} = Q_\eta$ and $\Sigma_e^{-1} = Q_e$ we have:

$$J(\underline{z}) = J(\underline{x}, \underline{p}) = \sum_{k=1}^{N} (\underline{\hat{y}}_k - C\underline{z}_k)^T Q_\eta (\hat{uy}_k - C\underline{z}_k) + \sum_{k=0}^{N-1} (\underline{z}_{k+1} - A\underline{z}_k)^T Q_e (\underline{z}_{k+1} - A\underline{z}_k) \Rightarrow$$

$$\frac{\partial J(z)}{\partial \underline{z}_k} = -C^T Q_\eta (\underline{\hat{y}}_k - C\underline{z}_k) - (\underline{\hat{y}}_k - C\underline{z}_k)^T Q_\eta C + Q_e (\underline{z}_k - A\underline{z}_{k-1}) + (\underline{z}_k - A\underline{z}_{k-1})^T Q_e = 0 \Rightarrow$$

Simplifying the terms:

$$\Rightarrow C^T Q_\eta C\underline{z}_k - C^T Q_\eta \underline{\hat{y}}_k + Q_e \underline{z}_k - Q_e A\underline{z}_{k-1} = 0 \Rightarrow$$

$$\Rightarrow (C^T Q_\eta C + Q_e)\underline{z}_k = C^T Q_\eta \underline{\hat{y}}_k + Q_e A\underline{z}_k \Rightarrow$$

Finally, developing the terms $Q_e, Q_\eta, A$ & $C^T$, emerges:

$$\begin{bmatrix} Q_x + C_x^T Q_\eta C_x & C_x^T Q_\eta C_p \\ C_p^T Q_\eta C_x & Q_x + C_x^T Q_\eta C_x \end{bmatrix} \begin{bmatrix} \underline{x}_k \\ \underline{p}_k \end{bmatrix} = \begin{bmatrix} C_x^T Q_\eta \\ C_p^T Q_\eta \end{bmatrix} \underline{\hat{y}}_k + \begin{bmatrix} Q_x A_x & Q_x B \\ 0 & Q_p A_p \end{bmatrix} \begin{bmatrix} \underline{x}_{k-1} \\ \underline{p}_{k-1} \end{bmatrix} \quad (3.46)$$

It should be noted that $Q_\eta$ and $Q_x$ have to be assigned values based on the expected measurement and model error, respectively. However, the $\sigma_p$ in $Q_p$ has to be selected using the L-curve concept (Azam et al., 2015). Specifically, writing the $\sigma_p$ in the form $\sigma_p = 10^\gamma$, one varies the exponent $\gamma$ from $\gamma_{low}$ to $\gamma_{high}$ values and plots the L-curve which is the values of $J(\underline{z})$ versus $\sigma_p = 10^\gamma$ and selects the optimal value $\sigma_{p_{opt}} = 10^{\gamma_{opt}}$ of $\sigma_p = 10^\gamma$ at the corner of the L-curve.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 3.7 Input-State-Parameter Estimation

Let $\underline{\theta}$ be a set of model parameters associated with the spring stiffness and mass values. Then the matrices $A \equiv A(\underline{\theta})$ and $C \equiv C(\underline{\theta})$ in the formulation are a function of the parameter set $\underline{\theta}$. In this case the Eq. (3.46) takes the form:

$$\begin{bmatrix} Q_x + C_x(\underline{\theta})^T Q_\eta C_x(\underline{\theta}) & C_x^T(\underline{\theta}) Q_\eta C_p(\underline{\theta}) \\ C_p^T(\underline{\theta}) Q_\eta C_x(\underline{\theta}) & Q_x + C_x^T(\underline{\theta}) Q_\eta C_x(\underline{\theta}) \end{bmatrix} \begin{bmatrix} \underline{x}_k(\underline{\theta}) \\ \underline{p}_k(\underline{\theta}) \end{bmatrix} = \begin{bmatrix} C_x^T(\underline{\theta}) Q_\eta \\ C_p^T(\underline{\theta}) Q_\eta \end{bmatrix} \hat{\underline{y}}_k + \begin{bmatrix} Q_x A_x(\underline{\theta}) & Q_x B(\underline{\theta}) \\ 0 & Q_p(\underline{\theta}) A_p(\underline{\theta}) \end{bmatrix} \begin{bmatrix} \underline{x}_{k-1}(\underline{\theta}) \\ \underline{p}_{k-1}(\underline{\theta}) \end{bmatrix} \tag{3.47}$$

Another objective is to estimate the state vector $\underline{z}$ and the parameter set $\underline{\theta}$ altogether. This can be accomplished by augmenting the parameter set $\underline{z}$ to include the model parameter set $\underline{\theta}$ and minimizing the objective function (3.44) with respect to both $\underline{z}$ and $\underline{\theta}$. Hence, the objective function (3.44) is written as:

$$J(\underline{z}, \underline{\theta}) = \sum_{k=1}^{N} (\hat{\underline{y}}_k - C(\underline{\theta}) \underline{z}_k(\underline{\theta}))^T \Sigma_\eta^{-1} (\hat{\underline{y}}_k - C(\underline{\theta}) \underline{z}_k(\underline{\theta})) + \sum_{k=0}^{N-1} (\underline{z}_{k+1}(\underline{\theta}) - A(\underline{\theta}) \underline{z}_k(\underline{\theta}))^T \Sigma_e^{-1} (\underline{z}_{k+1}(\underline{\theta}) - A(\underline{\theta}) \underline{z}_k(\underline{\theta})) \tag{3.48}$$

The minimization can be performed by computing $\underline{z}_k = (\underline{x}_k^T \quad \underline{p}_k^T)^T$ from the stationary conditions:

$$\nabla_{\underline{z}} J(\underline{z}, \underline{\theta}) = \underline{0} \tag{3.49}$$

to obtain the solution $\underline{z}(\underline{\theta})$ from the equations (3.47) with respect to the the model parameters $\underline{\theta}$, and then replacing the solution $\underline{z}(\underline{\theta})$ to the objective function $J(\underline{z}, \underline{\theta})$ to obtain the new objective function:

$$\tilde{J}(\underline{\theta}) = J(\underline{z}(\underline{\theta}), \underline{\theta}) = \sum_{k=1}^{N} (\hat{\underline{y}}_k - C \underline{z}_k(\underline{\theta}))^T \Sigma_\eta^{-1} (\hat{\underline{y}}_k - C \underline{z}_k(\underline{\theta})) + \sum_{k=0}^{N-1} (\underline{z}_{k+1}(\underline{\theta}) - A \underline{z}_k(\underline{\theta}))^T \Sigma_e^{-1} (\underline{z}_{k+1}(\underline{\theta}) - A \underline{z}_k(\underline{\theta})) \tag{3.50}$$

with respect to $\underline{\theta}$ only. For each $\underline{\theta}$ value, vector $\underline{z}$ is computed via (3.47). Function "fminsearch" from MATLAB is exploited to estimate the minimum value of the objective function and the corresponding optimal $\underline{\theta}$ values.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Chapter 4: Application in Spring-Mass Chain Model

For this Diploma Thesis, several (10) cases were researched. All of them consist of virtual-driven data. Further information is provided as follows.

As can be easily seen from Figure 4.1, the virtual system consists of 10 DOF (Degrees of Freedom), 10 point masses, that are located in a vertical row. There are strings and damping mechanisms between them. The first mass, $m_1$, has spring $k_1$ & damping mechanism $c_1$, the second one $m_2$, has $k_2$ & $c_2$, correspondingly, et cetera. X stands as the horizontal axis and Y as the vertical. The simulated data regarding displacement refer to the Y axis.

Analytic information about the Model's specifications are presented at Table 4.1:

| **Virtual Experiment Specifications** | | |
|---|---|---|
| **Notation** | **Name** | **Value - Unit** |
| m | Mass | 1 - g |
| k | Stiffness | 18.2 - N/mm |
| c | Damping Coefficient | 0.02 - Ns/mm |
| u | Displacement | N/A - mm |

Table 4.1: Virtual Experiment Specifications

Regarding the sensor placement, each case will comprise several configurations, namely:

1. Acceleration sensor(s) only.
2. Both sensors in different DOFs.
3. Collocated Sensors.

The ultimate purpose of this set of configurations consists to evaluate the sensor type and placement affect the model accuracy. Furthermore, the $\underline{\theta}$ values of stiffness are set for parameter estimation in some of the cases. Functions B.17 & B.18 assist in the input parameter identification procedure.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 4.1 Spring-Mass Model Presentation

Firstly, the created software is tested in a virtual-driven experiment, as seen in the figure 4.1:

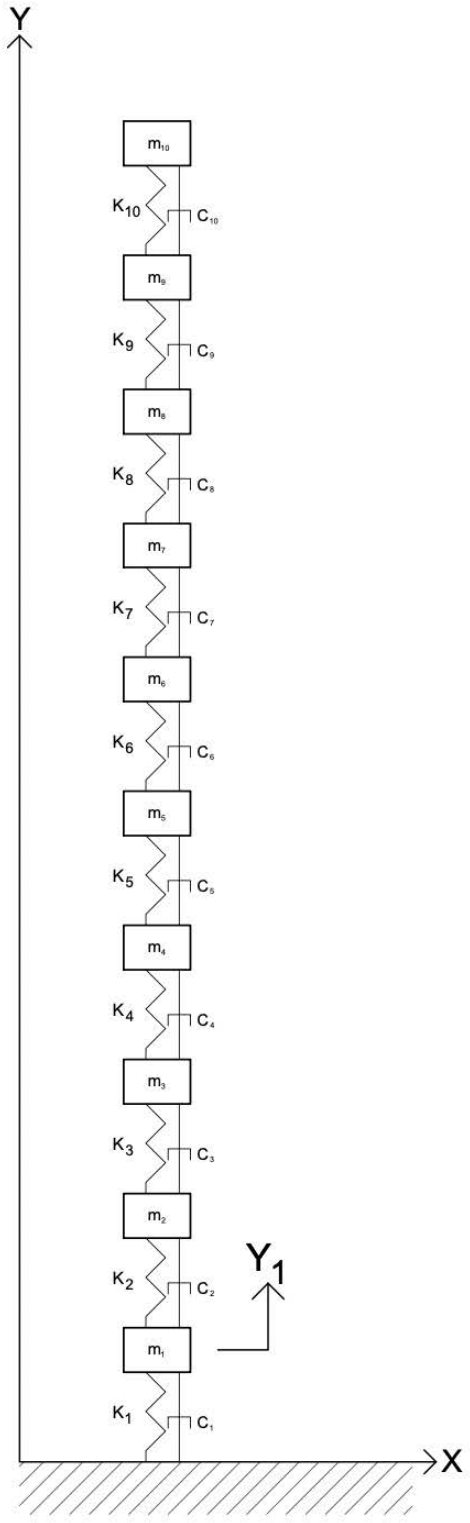



Figure 4.1: Spring-Mass Chain Model



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 4.2 $\sigma_p = 10^{\gamma}$ Automated Estimation Technique From L-curve

Regarding the $\sigma_p = 10^{\gamma}$ values that are strongly associated with $\underline{\theta}$ predictions, in case of input parameter estimation necessity, through the main software B.1, is proposed a technique to statistically define the best value of Covariance Matrices for Formulation 2, by selecting the most probable $\sigma_p = 10^{\gamma}$ for variant $\underline{\theta}$ values. More specifically:

1. A specific structure and individual set of force excitations are set for experiment (There is a concession that each experiment will have different optimal Covariance Matrix Values).
2. In the beginning of the software, there will be multiple L-curve creations in order to produce adequate amount of data. They will be generated by altering $\underline{\theta}$ value by a step coefficient.
3. For each and every spring value, that creates a L-curve figure, optimal value will be estimated according to the (Hansen, 2001) book[1]. More information about this analysis is presented in the Appendix Part of the Diploma Thesis.
4. Assuming that $s_p$ & $Q_p$ values are logarithmic, every iteration stores the optimal value in a new vector with its index.
5. After several tests, the most probable value is selected. In case of a tie, the higher value prevails. This method is considered in the B.1 script.
6. Due to the fact that this framework is untested in the long run, manual comparison with the L-curve is performed in order to validate the selection of $\underline{\theta}$.

## 4.3 Software used

The creation of Diploma Thesis requires the use of software to compute and identify the loads and responses. More precisely, in this Diploma Thesis, MATLAB (Matrix Laboratory), academic version, R2024b was used.

## 4.4 Hardware Used

For the writing of this Diploma Thesis plus the software running, personal notebook used, with technical characteristics:

- Operating System: Windows 10 Pro 64-bit
- CPU: AMD A10-9600P (4 cores, 4 threads, 2400 MHz Stock Core Speed
- RAM: 6 Giga Bytes
- GPU: AMD Radeon R5

[1]Chapter: "The L-curve and its use in the numerical treatment of inverse problems"



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 4.5 Application Schematic Illustration

Finally, before moving to the Results Chapter, a flow-chart of the total procedure of the suggested framework is illustrated at 4.2 for clarification purposes.

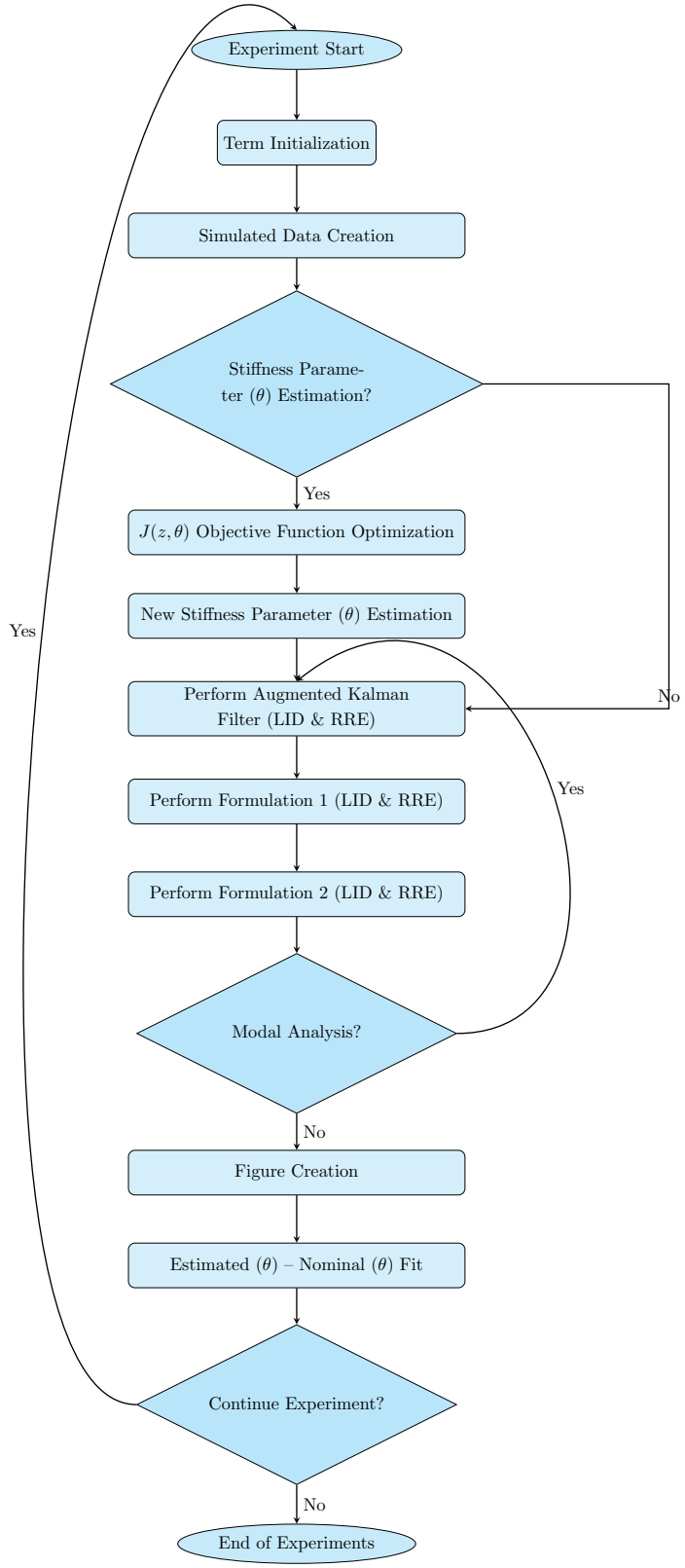



Figure 4.2: Experimental Procedure Flowchart.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Chapter 5: Results

In this Chapter, the results of ten simulated cases will be presented with figures and tables. There will be also extensive discussion of the results.

Each Test will run in multiple levels of errors ( from 0% to 10% model and measurement error) in order to observe the evolution of each Formulation. Considering the optimal value of $\theta$ is set as 18.2. Additionally, the first guess for parameter estimation equals to 25 for each DOF (cases that require input-parameter estimation). The number of modes in Modal Analysis is preset to $m = 3$ modes.

For the rest of the chapter, each figure case is explained as follows:

1. Red arrows denote each force that excites the model.
2. Blue arrows denote accelerator sensors that monitor the model.
3. Green arrows denote strain sensors that monitor the model.
4. Orange arrows denote the unknown input-parameters of the model that require estimation.


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.1 Case 1

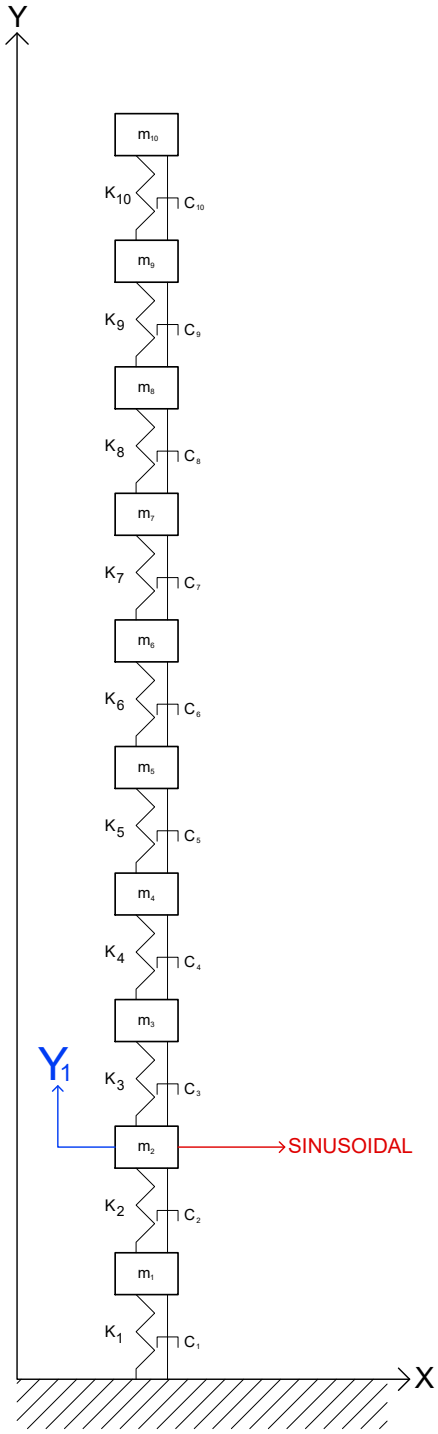


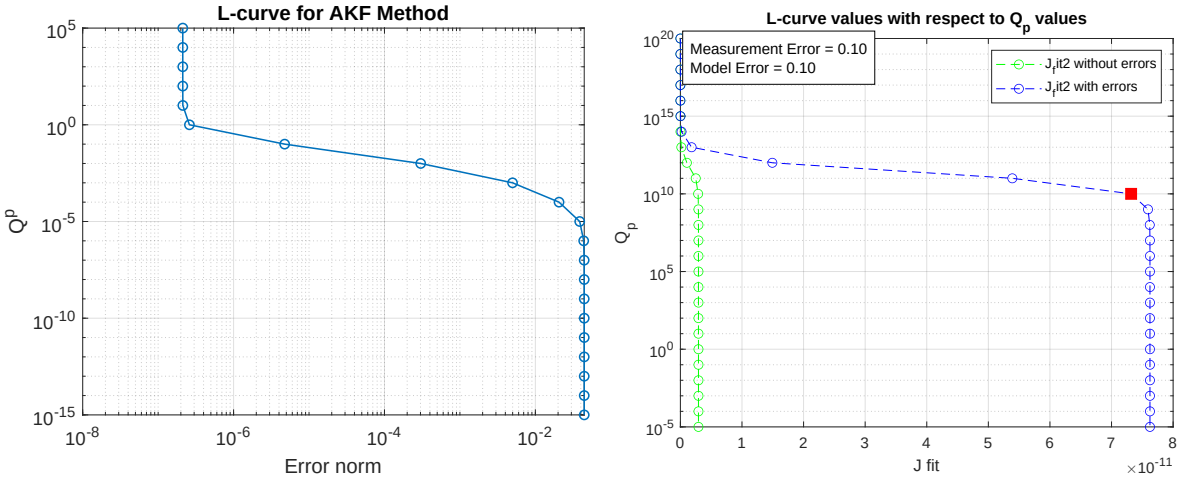

Figure 5.1: Spring-Mass Model - Case 1

Derived from figure 5.1, an accelerator is the only sensor to monitor the structure. There are no unknown $\theta$ parameters to estimate. Sinusoidal is the excitation acting on the second mass. L-curve curves for AKF & Formulation 2 are illustrated at figures 5.2a & 5.2b for



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

error value $e = 0.10$. $\sigma_{p1}$ and $\sigma_{p3}$ values are selected as 10 and $10^{14}$ correspondingly. Manual selection needed, meaning that automated selection proposed in Section 4.2 was ineffective in this case.

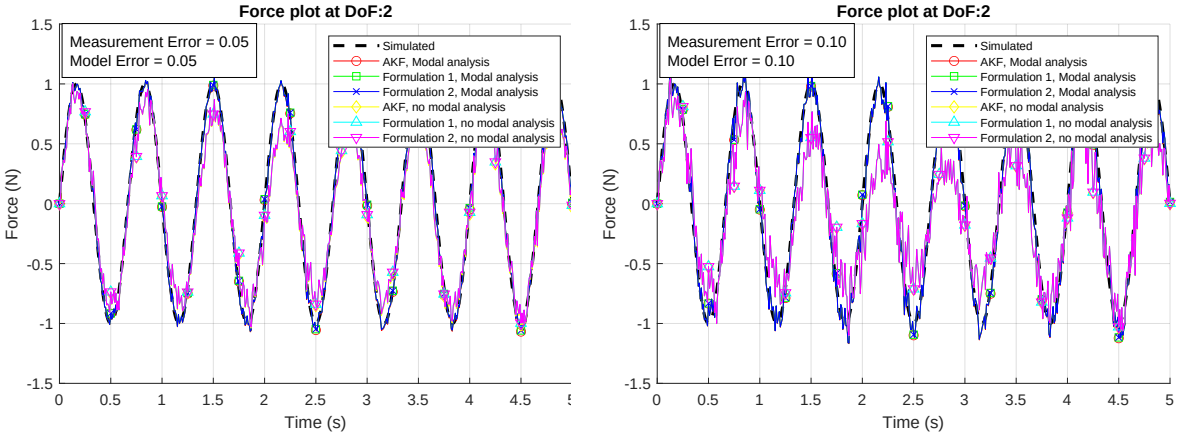



(a) L-curve for AKF Method

(b) L-curve for Formulation 2

Figure 5.2: L-curve for error 0.10 - Case 1

Considering the monitoring of the loaded DOF by an acceleration sensor, figures at 5.3 illustrate the results of the input estimation frameworks. Indisputably, Modal Analysis approaches demonstrate excellent precision even for large error values (figure 5.3b) and hold a significant edge compared to the all-modes analysis respective methods.




(a) Inputs for error 0.05

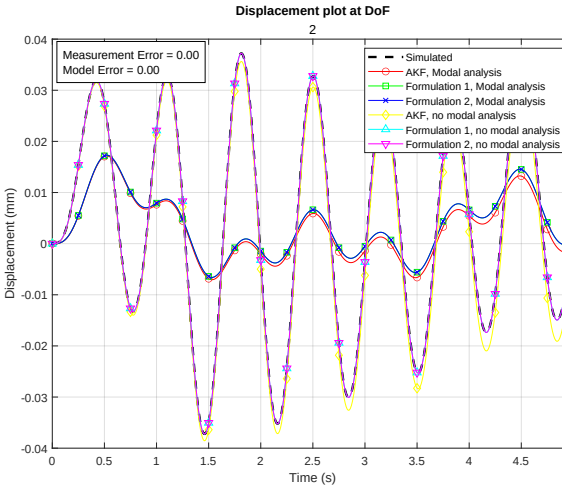
(b) Input for error 0.10

Figure 5.3: Inputs for variable errors - Case 1

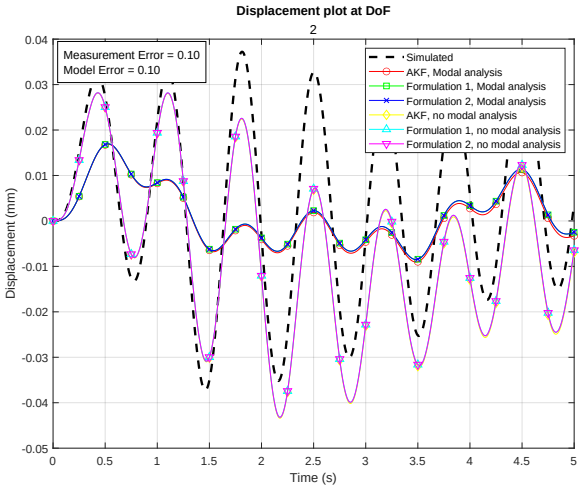
The reconstructed displacement, velocity and strain responses are shown in figures 5.4, 5.5 & 5.6 regarding various DOF of the Spring-Mass Model (2nd, 7th & 10th DOF, accordingly). For zero error values it is observed for all types of figures (5.4a, 5.5a & 5.6a high-fidelity results. On the contrary, according to figures 5.4b, 5.5b & 5.6b lower accuracy observations are apparent. Moreover, Modal Analysis seems unable to reconstruct the responses, scoring poor precision, fact that is anticipated considering the lower number of modes. Lack of displacement or strain sensor installation explains these observations.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

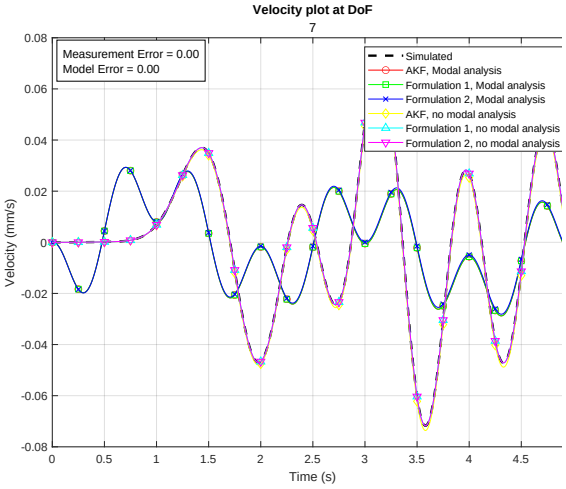



(a) Displacement Plot for error 0.00

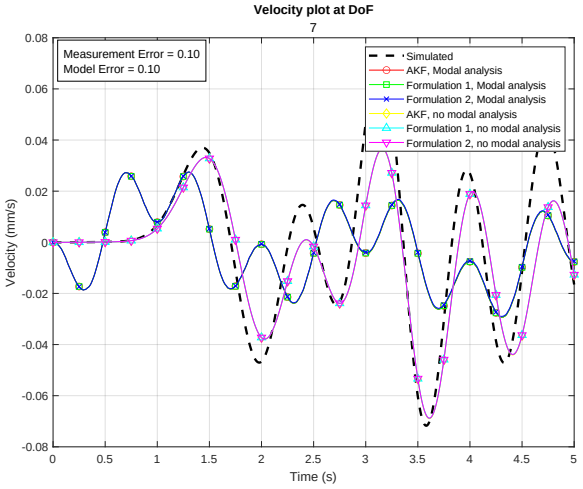



(b) Displacement Plot for error 0.10

Figure 5.4: Displacement Plots at 2nd DOF - Case 1

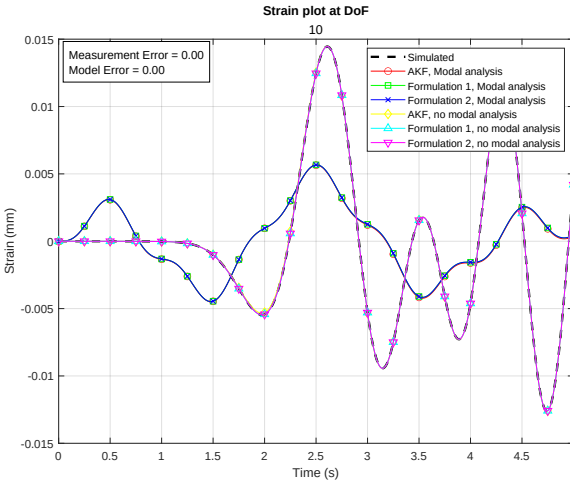



(a) Velocity Plot for error 0.05

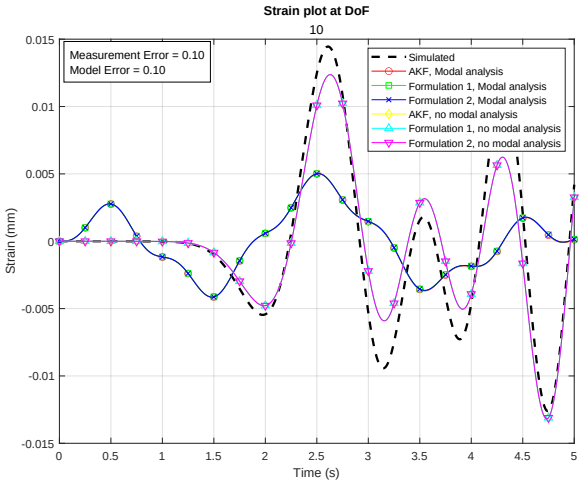



(b) Velocity Plot for error 0.10

Figure 5.5: Velocity Plots at 7th DOF - Case 1




(a) Strain Plot for error 0.05




(b) Strain Plot for error 0.10

Figure 5.6: Displacement Plots at 10th DOF - Case 1



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.2 Case 2

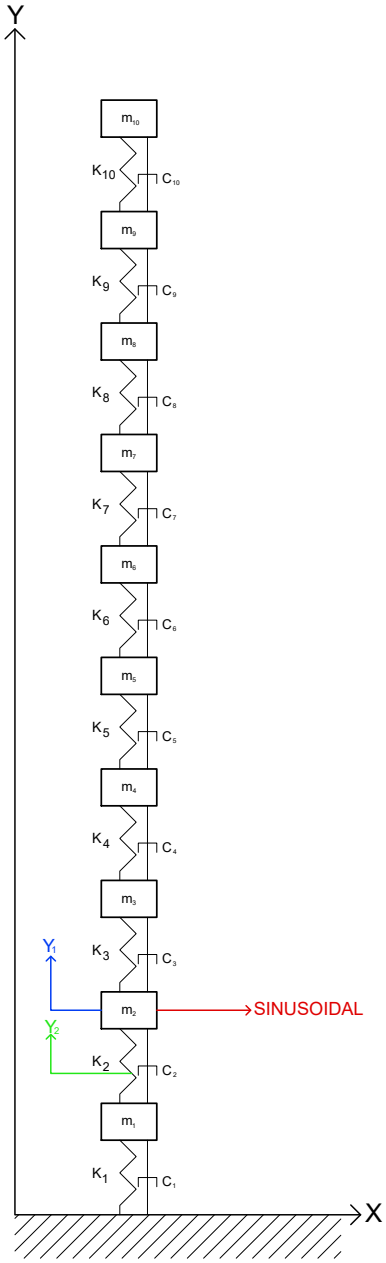



Figure 5.7: Spring-Mass Model - Case 2

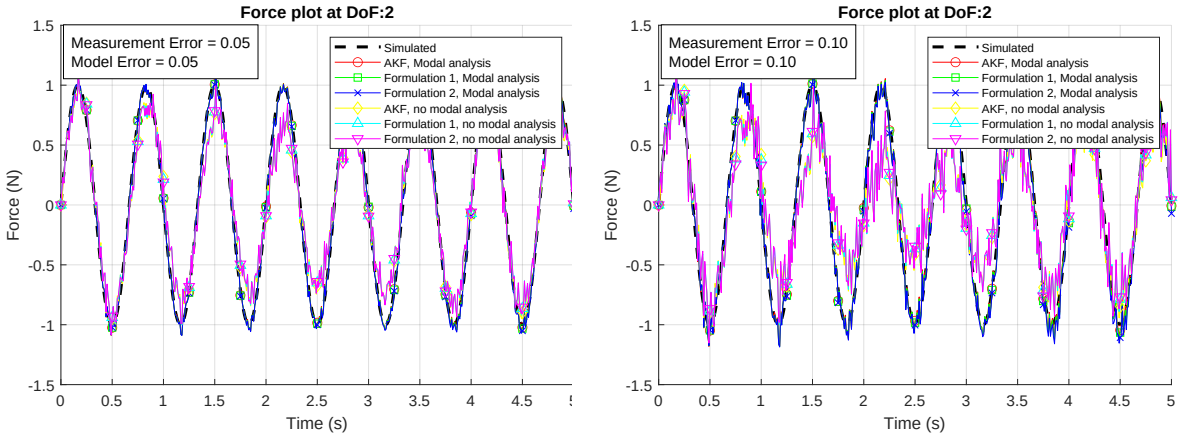
Derived from figure 5.7, apart from the accelerator to monitor the input, there is also a strain sensor installed in the second spring, creating a set of collocated sensors configuration. All spring parameters are known. Sinusoidal remains the excitation acting on the second



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

mass. This case was tested once again for varying errors, starting from zero error values to 0.10 error level.

The results derived from input figures at 5.8 indicate once more high-fidelity approach for all the methods (AKF, Formulation 1 & Formulation 2) in case of Modal Analysis, not only for normal perturbation scale, as shown at 5.8a, but also for highly contaminated data as illustrated at figure 5.8b. On the contrary, all-modes analysis seem to struggle to identify the sinusoidal load with precision. It can be deducted that the installation of strain sensor doesn't affect input estimation capability.

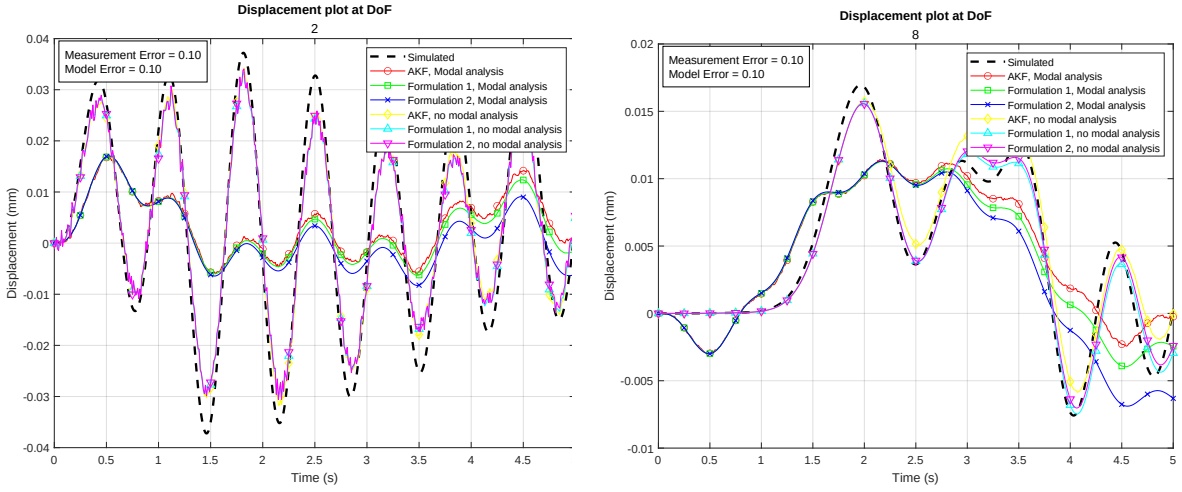



(a) Inputs for error 0.05

(b) Input for error 0.10

Figure 5.8: Inputs for variable errors - Case 2

Estimated responses, for error equal to 0.10, at second and eighth mass are illustrated in figures 5.9, 5.10 & 5.11 for displacements, strains & velocities respectively. Modal Analysis once again fail to provide trustworthy predictions, irrespective of the formulation used. Regarding all-modes analysis, from displacement and strain figures, it is observed equal estimation capability near the strain sensor (figurex 5.9a and 5.10a).



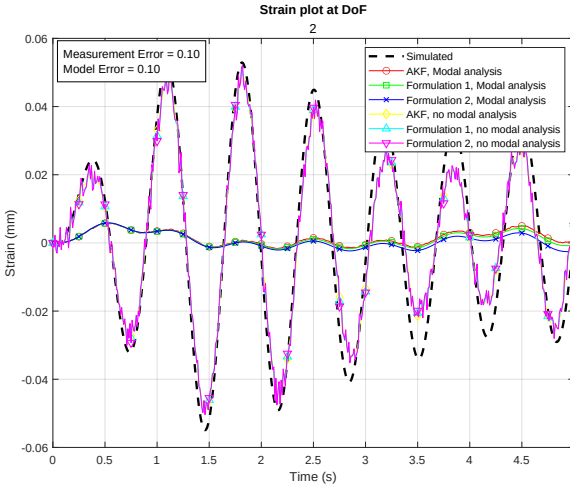

(a) Displacement Plot at 2nd DOF

(b) Displacement Plot at 8th DOF

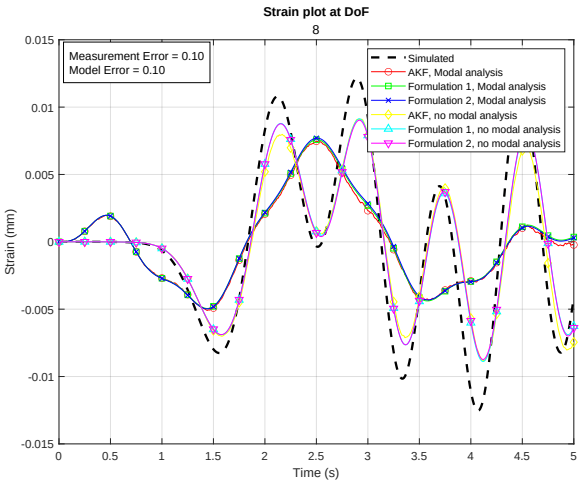
Figure 5.9: Displacement Plots - Case 2



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250



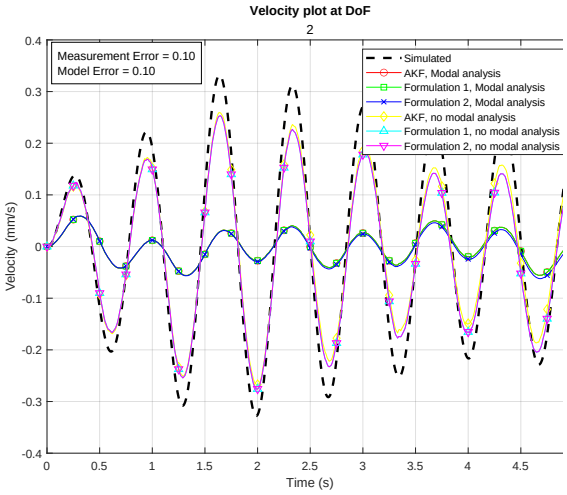

(a) Strain Plot at 2nd DOF



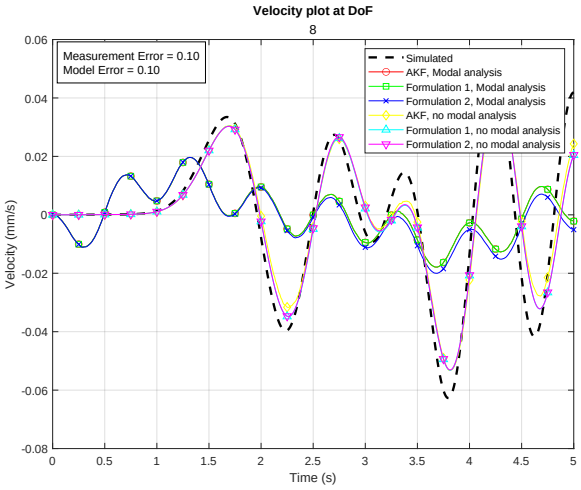

(b) Strain Plot at 8th DOF

Figure 5.10: Strain Plots - Case 2

On the contrary, **proposed Formulations 1 and 2 score higher-fidelity far from the sensor installation** (figures 5.9b & 5.10b). For velocity responses, according to figure 5.11a, AKF method is advantageous over proposed techniques near sensor placement, while Formulation 1 and 2 predict relatively better the responses far from the sensor placement (figure 5.11b). Overall, adding one more sensor at the input undeniably improve system's monitoring accuracy.




(a) Velocity Plot at 2nd DOF




(b) Velocity Plot at 8th DOF

Figure 5.11: Velocity Plots - Case 2



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.3 Case 3

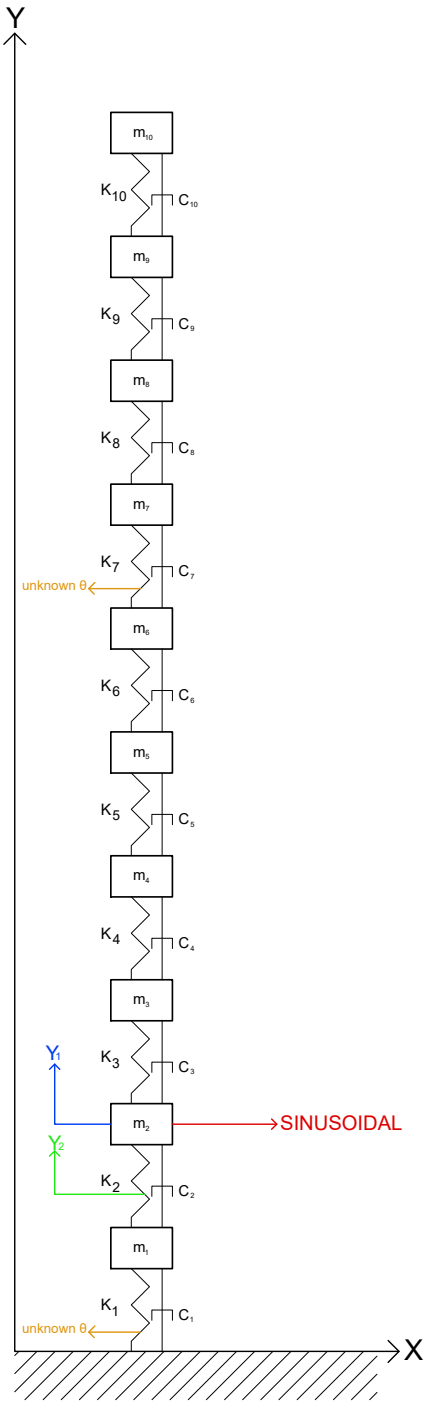



Figure 5.12: Spring-Mass Model - Case 3

Derived from figure 5.12, and also considering 5.7, the only difference on the third case lies upon the lack of information about spring parameters ($k_1$ and $k_7$). Except for load identification and response reconstruction, within this case input-parameter estimation will



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

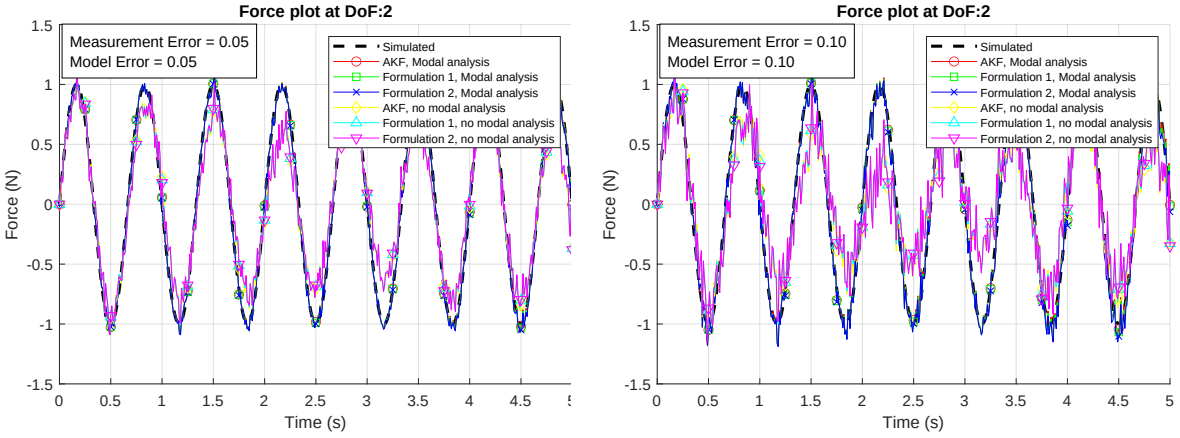
be performed.

Beginning with the input-parameter estimation, by observing table 5.1, is concluded that spring estimation near the strain sensor (1st DOF) was successful, while, on the other hand, spring estimation at 7th DOF failed considerably. This is expected due to the number of installed strain sensors. It also indicates **the necessity of adding another strain sensor near the other unknown spring parameter**.

| $\Delta\theta\%$ **Input Parameter Relative Error** | | | | |
|---|---|---|---|---|
| **DOF** | **e = 0.00** | **e = 0.02** | **e = 0.05** | **e = 0.10** |
| 1 | 0.09 | -3.14 | 3.25 | 3.94 |
| 7 | N/A | N/A | -70.11 | -75.35 |

Table 5.1: $\Delta\theta\%$ Relative Error - Case 3

Input identification figures are illustrated at 5.13. Modal Analysis shows remarkably high-fidelity even for extreme error scenarios, as shown in figure 5.13b. On the other hand, all-modes analysis doesn't prove high accuracy once again.




(a) Input for error 0.05

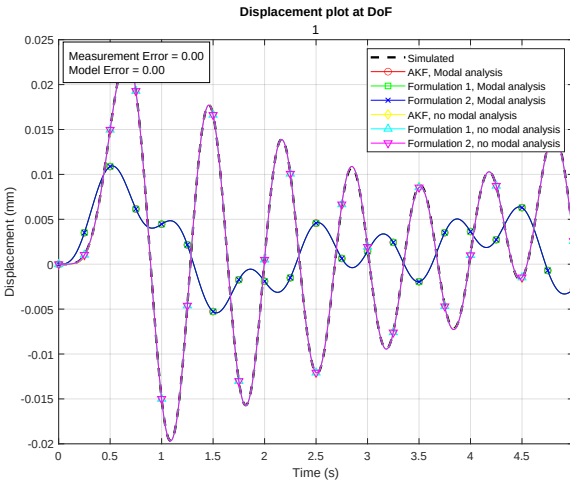
(b) Input for error 0.10

Figure 5.13: Inputs for variable errors - Case 3

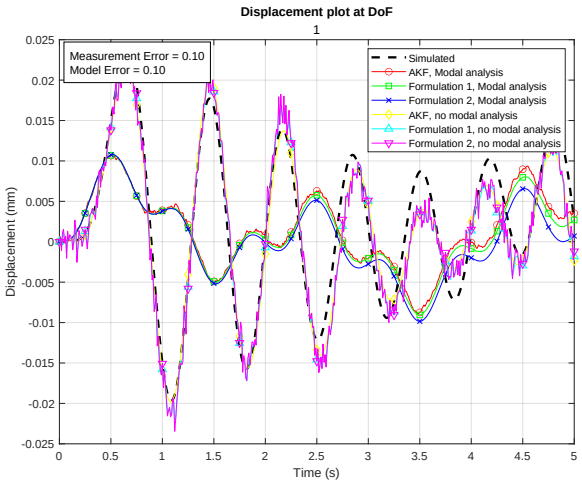
The results for the displacement, strain and velocity estimation, considering 1st DOF where the input-parameter prognosis was successful, is shown at figures 5.14, 5.15 and 5.16 respectively. All-mode analysis is for choice, as Modal Analysis implementation records poor accuracy. In case of displacements, there isn't considerable difference in fidelity between Augmented Kalman Filter, Formulation 1 & 2 (figure 5.14b). However, increasing the applied error to the model results in precision deterioration, with Augmented Kalman Filter scheme recording better results for strain (5.15b) & velocity responses (5.16b). Consequently, Augmented Kalman Filter is less affected by contaminated spring-parameter estimation than the proposed frameworks.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

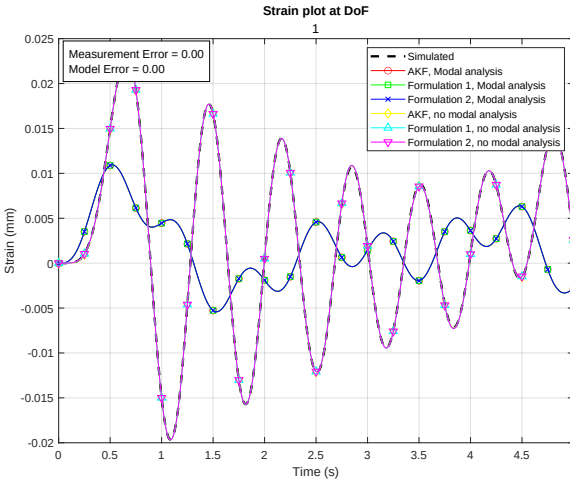



(a) Displacement Plot for no error

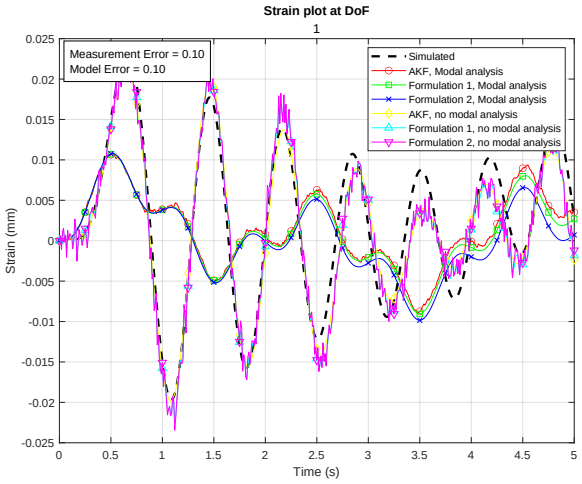



(b) Displacement Plot for error 0.10

Figure 5.14: Displacement Plots at 1st DOF - Case 3

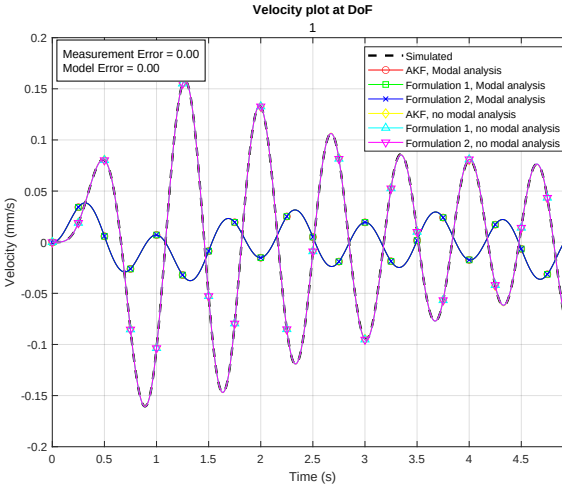



(a) Strain Plot for no error

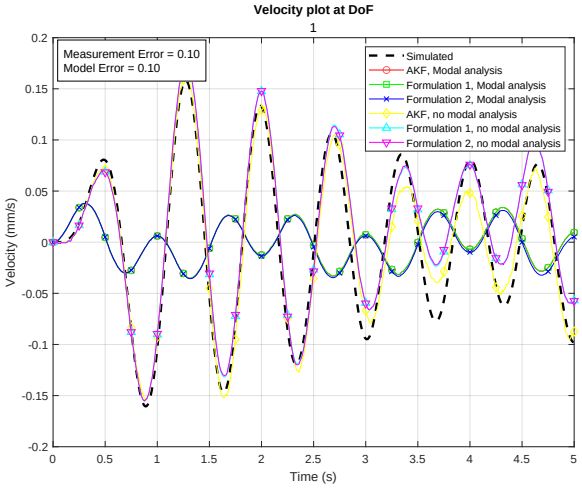



(b) Strain Plot for error 0.10

Figure 5.15: Strain Plots at 1st DOF - Case 3




(a) Strain Plot for no error




(b) Velocity Plot for error 0.10

Figure 5.16: Velocity Plots at 1st DOF - Case 3



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.4 Case 4

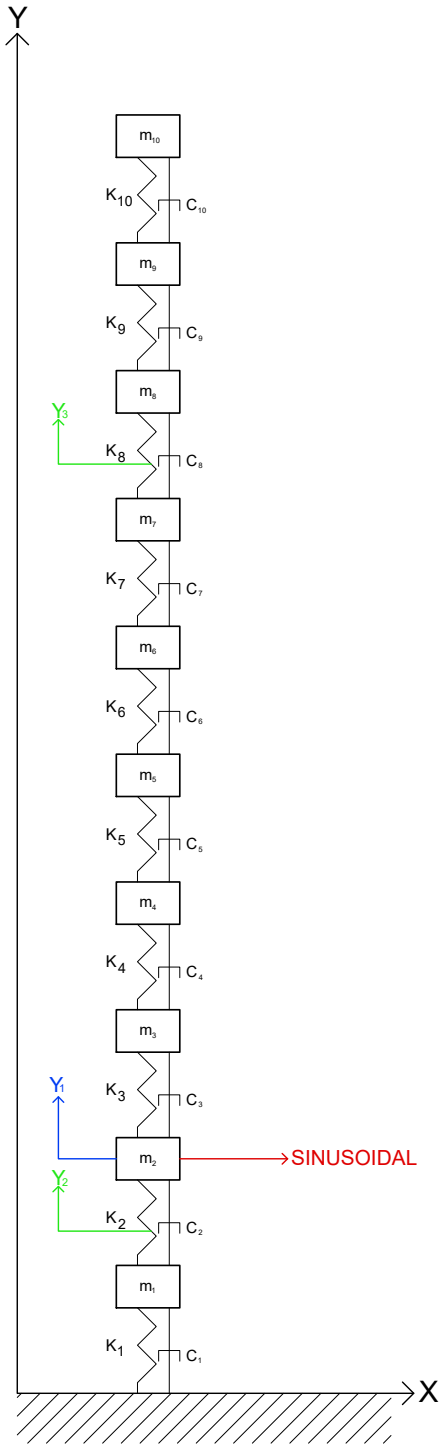



Figure 5.17: Spring-Mass Model - Case 4

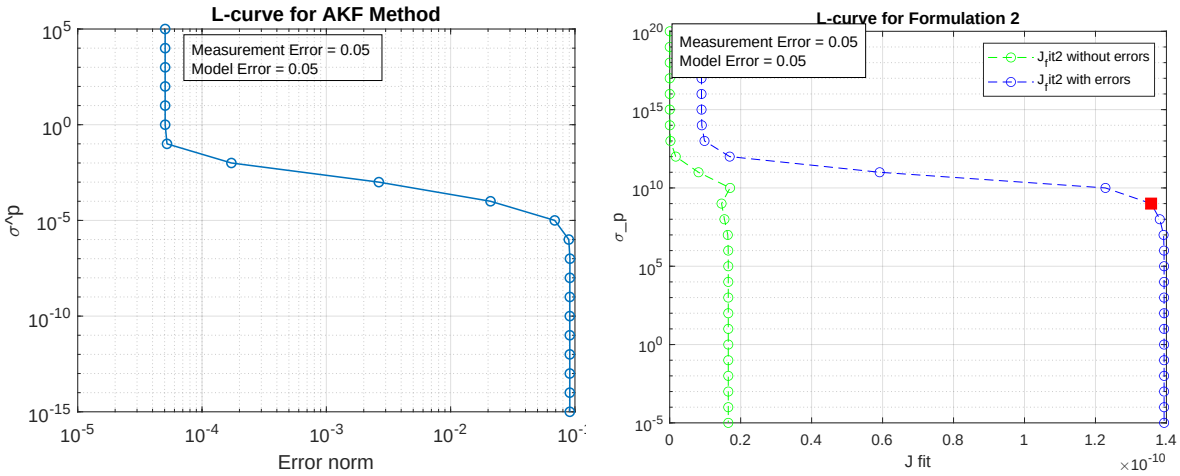
Derived from figure 5.17, apart from the collocated sensors configuration of Case 3, another strain sensor at the 8th spring, $k_8$, is applied. All spring parameters are known, meaning that there isn't input-parameter necessity in this case. Force excitation remains sinusoidal,



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

implemented at $m_2$.

L-curve curves for AKF & Formulation 2 are illustrated at figures 5.18a & 5.18b for error values $e = 0.10$. $\sigma_{p1}$ and $\sigma_{p3}$ values are selected as 0.1  $10^{14}$ correspondingly. Manual selection needed, meaning that automated selection proposed in Section 4.2 was ineffective in this case.

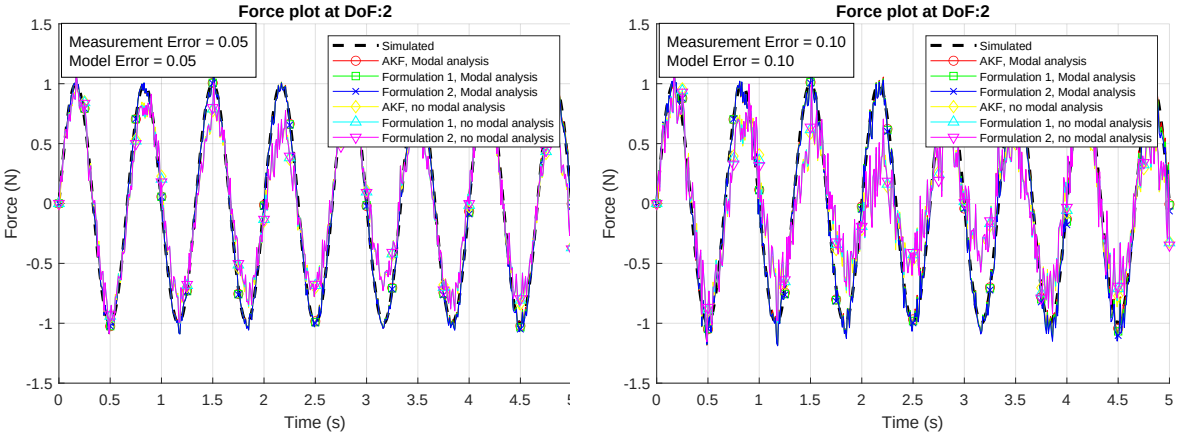



(a) L-curve for AKF Method

(b) L-curve for Formulation 2

Figure 5.18: L-curve for error 0.05 - Case 4

Load identification figure at 5.19a, shows once more high-fidelity for Modal Analysis implementation compared to the all-modes one. Augmented Kalman Filter, Formulation 1 & Formulation 2 show similar prediction capability. Modal Analysis implementation for higher error values ( up to 0.10), remains fully accurate, as shown at 5.19b, indicating that both proposed formulations are robust.




(a) Input Plot for error 0.05

(b) Input Plot for error 0.10

Figure 5.19: Input Plots for variable error - Case 4

Reconstructed responses of displacements and strains and velocities are shown at 5.20 and 5.21, respectively. Regarding displacement figure 5.20, for error equal to 0.05, high-precision results are produced, with similar potential for Augmented Kalman Filter, Formulation 1 and Formulation 2.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Equivalent results are also generated from strain (5.21) and velocity figures, proving the robustness of the proposed formulations, not only against Augmented Kalman Filter but also against simulated data. Finally, Modal Analysis implementation seems to struggle predicting precisely the responses.

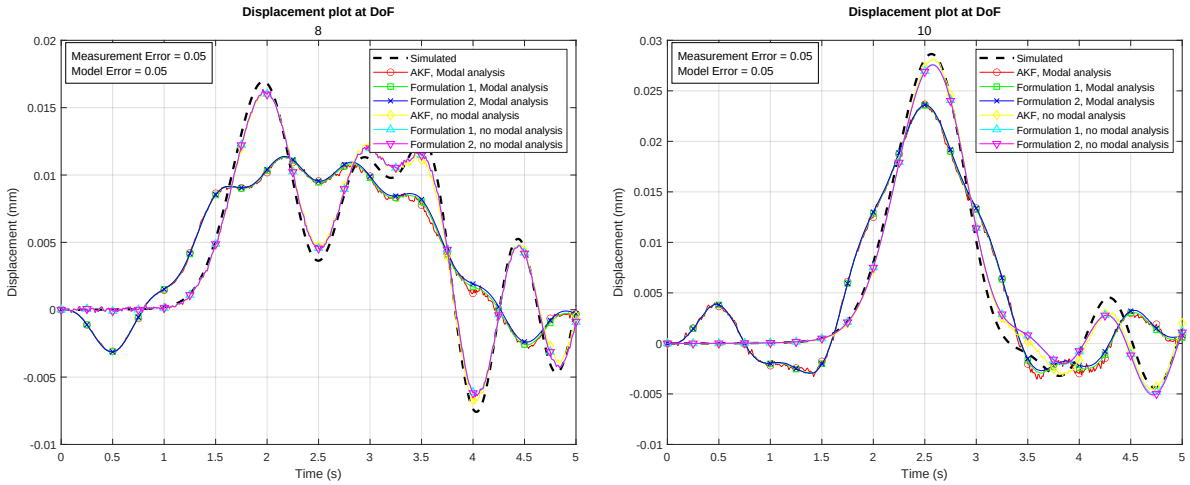



Figure 5.20: Displacement Plots at 8th and 10th DOF for error 0.05 - Case 4

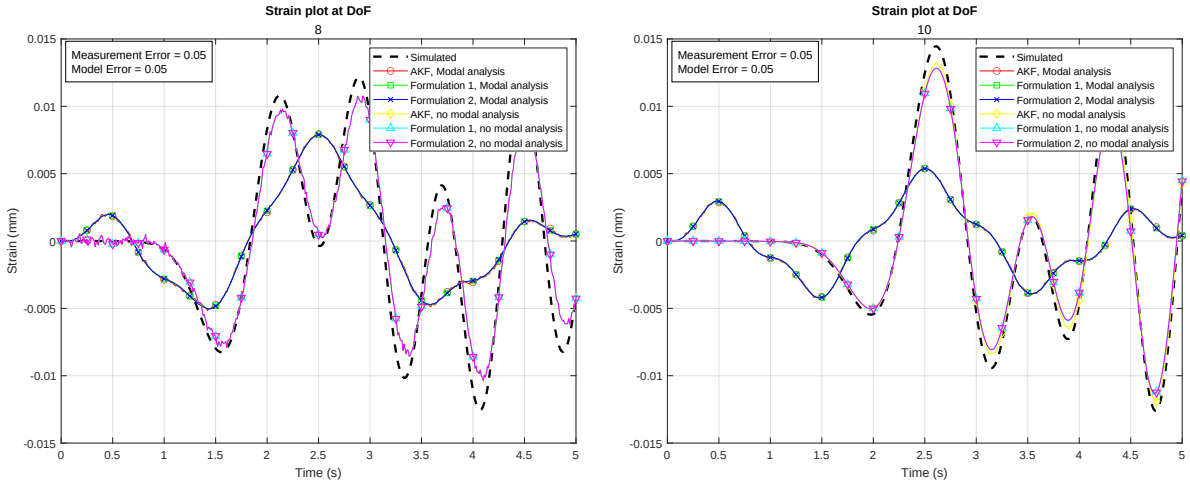



Figure 5.21: Strain Plots at 8th and 10th DOF for error 0.05 - Case 4



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.5 Case 5

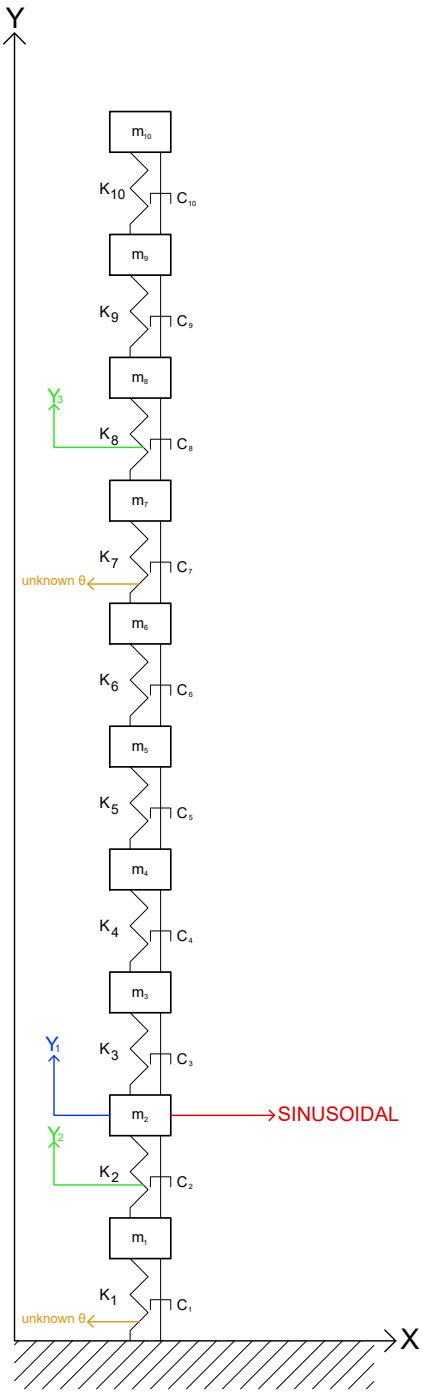



Figure 5.22: Spring-Mass Model - Case 5

Derived from figure 5.22, and also regarding 5.17, this case differentiates only at the unknown input-parameters, namely $\underline{\theta_1}$ and , $\underline{\theta_7}$, which need estimation. Compared to third Case, adding another strain sensor adjacent to $\underline{\theta_7}$ parameter makes this configuration critical.



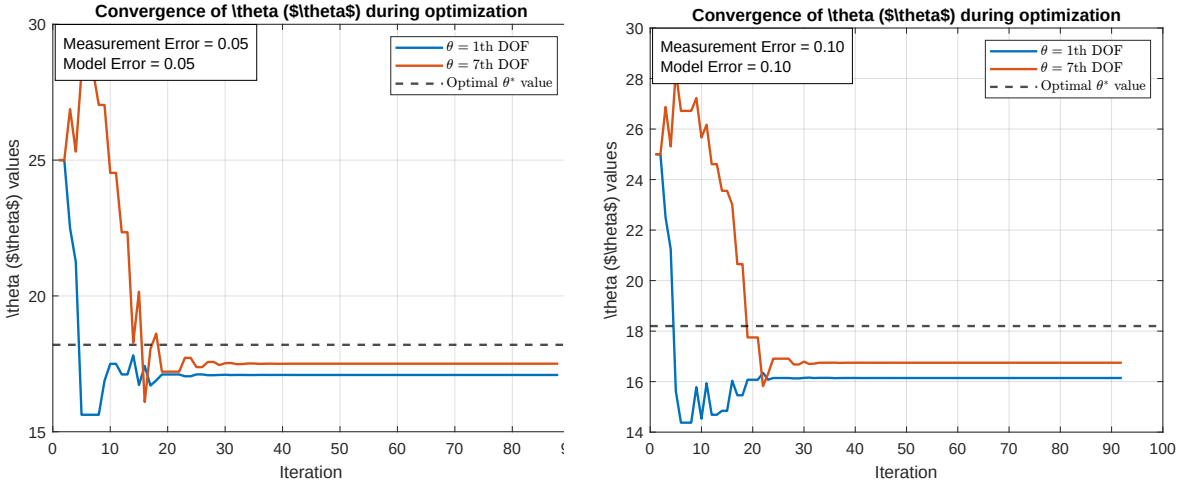
Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Table 5.2 shows the input-parameter relative error results. High-fidelity input-parameter estimation is achieved for varying error values.

| $\Delta\theta\%$ **Input Parameter Relative Error** | | | | |
|---|---|---|---|---|
| **DOF** | **e = 0.00** | **e = 0.02** | **e = 0.05** | **e = 0.10** |
| 1 | 0.00 | -2.53 | -6.1 | -11.3 |
| 7 | -0.07 | -1.57 | -3.82 | -7.98 |

Table 5.2: $\Delta\theta\%$ Relative Error - Case 5

Moreover, figure 5.23 illustrates the convergence of the unknown parameters with respect to the number of iterations. It can be seen from figures that software converges quite fast, approximately 30 iterations for error equal to 0.05 (5.23a) and 35 iterations for highest error value (5.23b). Optimal values of covariances are set $\sigma_{p1} = 0.1$ and $\sigma_{p3} = 10^{14}$ like the previous case.




(a) $\underline{\theta}$-Convergence Plot for error 0.05

(b) $\underline{\theta}$-Convergence Plot for error 0.05

Figure 5.23: $\underline{\theta}$-Convergence Plots for varying error - Case 5

Load identification results are presented at figure 5.24. As can be seen from figure 5.24b, even for high error values, Modal Analysis implementation estimates the excitation accurately for Augmented Kalman Filter, Formulation 1 and 2. On the contrary, all-modes analysis fidelity seems to deteriorate from figure 5.24a to 5.24b.

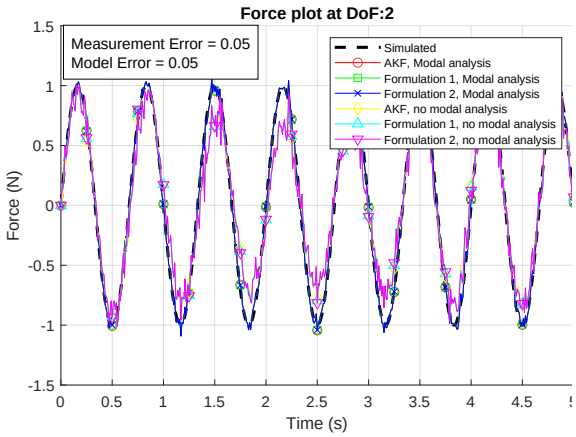
Reconstructed responses are presented at figures 5.25, 5.26 and 5.27 for displacements, strains and velocities. Error is set equal to 0.10. 6th and 10th DOF selected for illustration. It is observed that Augmented Kalman Filter method estimates slightly better the responses than the proposed formulations, as can be seen from figures 5.25b, 5.26b and 5.27b.

This remark concurs with findings from Case 3 (figure 5.14b, 5.16b) which also needed input-parameter estimation (5.12). Once more, modal analysis implementation of the methods doesn't give trustworthy results, as shown at 5.25, 5.26 and 5.27. On the other hand, taking into consideration second case, findings in this case opposites to figures 5.9b, 5.10b & 5.11b, where Formulations 1 and 2 showed better accuracy than Augmented Kalman Filter.

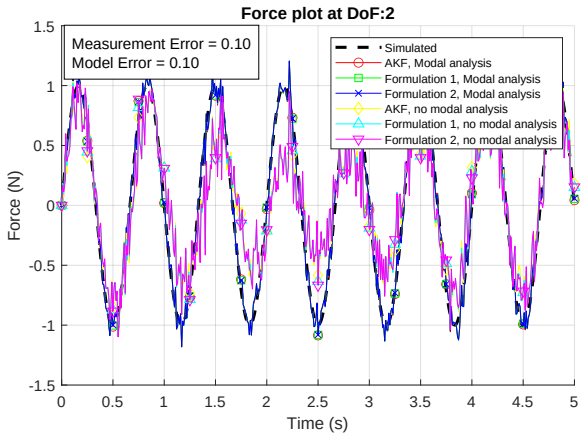
Concluding, compared to Case 3, adding an extra strain sensor near the unmonitored



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

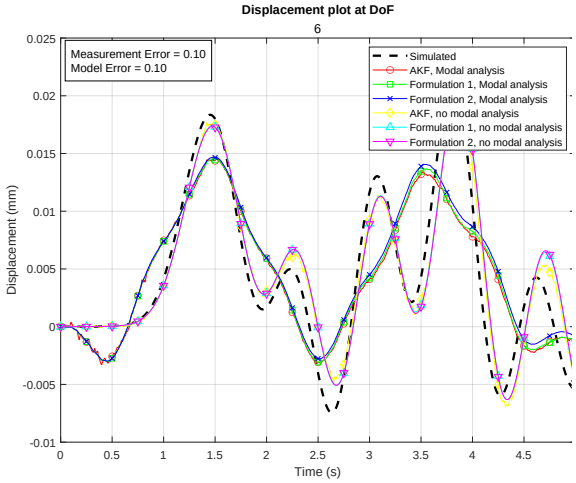



(a) Input Plot for error 0.05

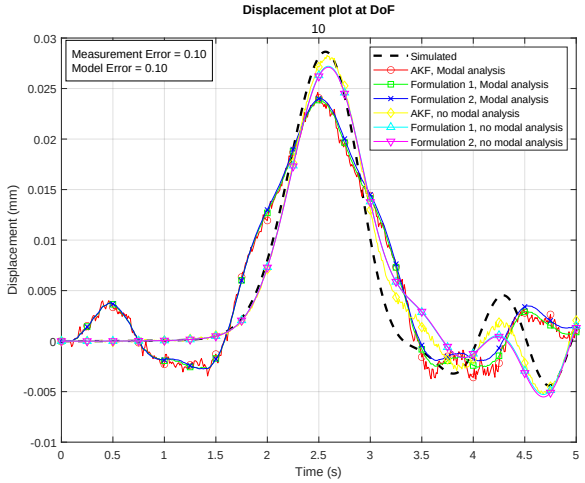



(b) Input Plot for error 0.05

Figure 5.24: Input Plots for variable error - Case 5

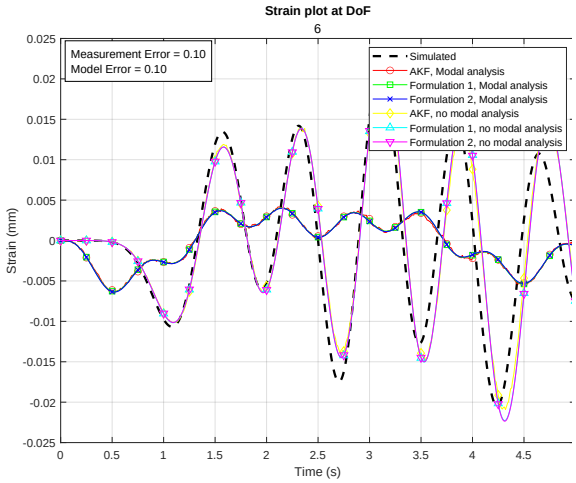



(a) Displacement Plot at 6th DOF

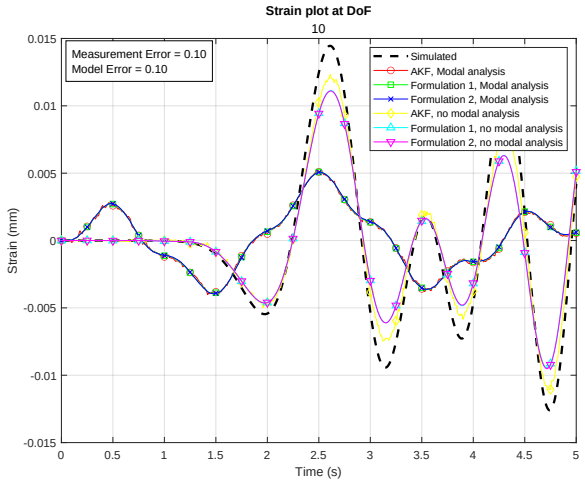



(b) Displacement Plot at 10th DOF

Figure 5.25: Displacement Plots for error 0.10 - Case 5




(a) Strain Plot at 6th DOF

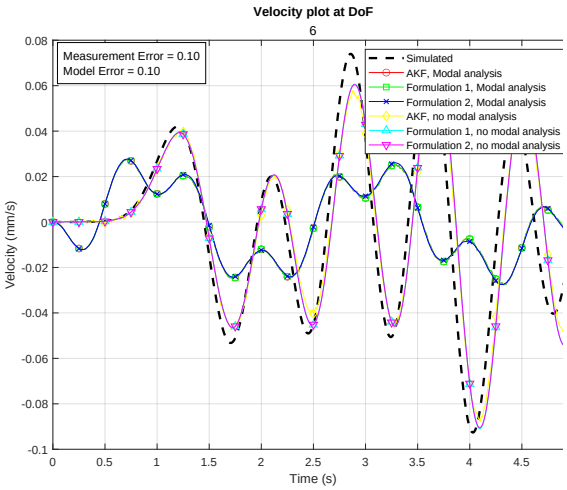



(b) Strain Plot at 10th DOF

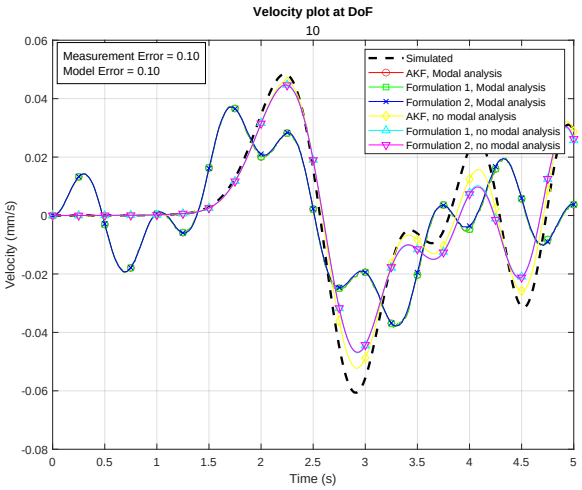
Figure 5.26: Strain Plots for error error 0.10 - Case 5



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250




(a) Velocity Plot at 6th DOF




(b) Input Plot for error 0.05

Figure 5.27: Velocity Plots for error 0.10 - Case 5

unknown $\underline{\theta_7}$ parameter decreased the input-parameter estimation error of the model and improved the software's precision at reconstructing responses.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.6 Case 6

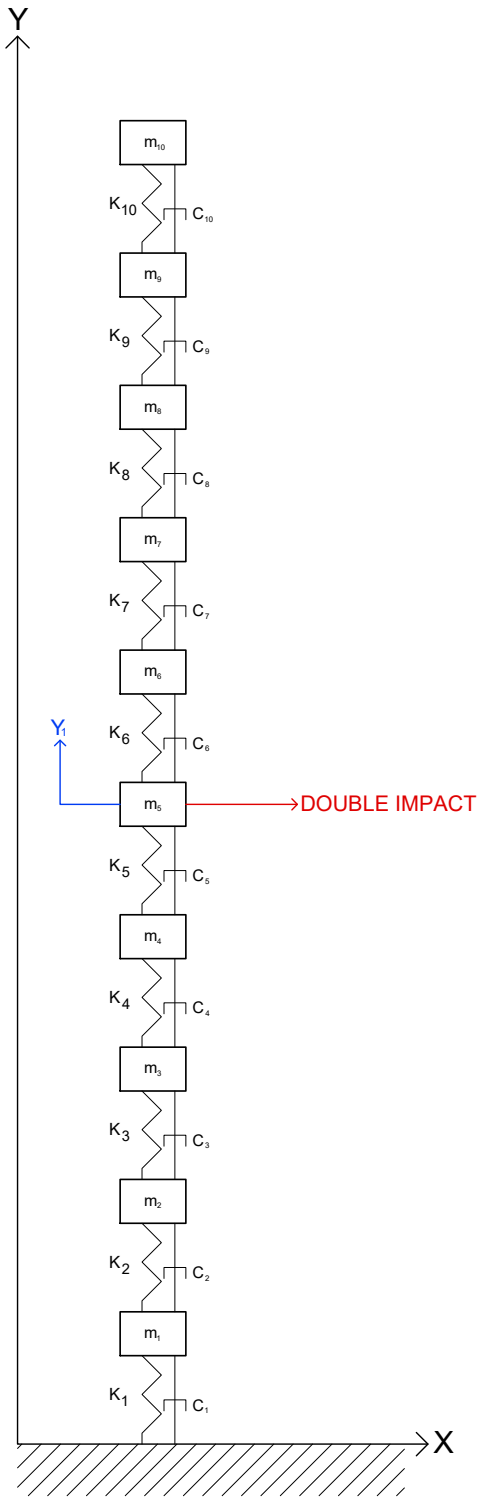


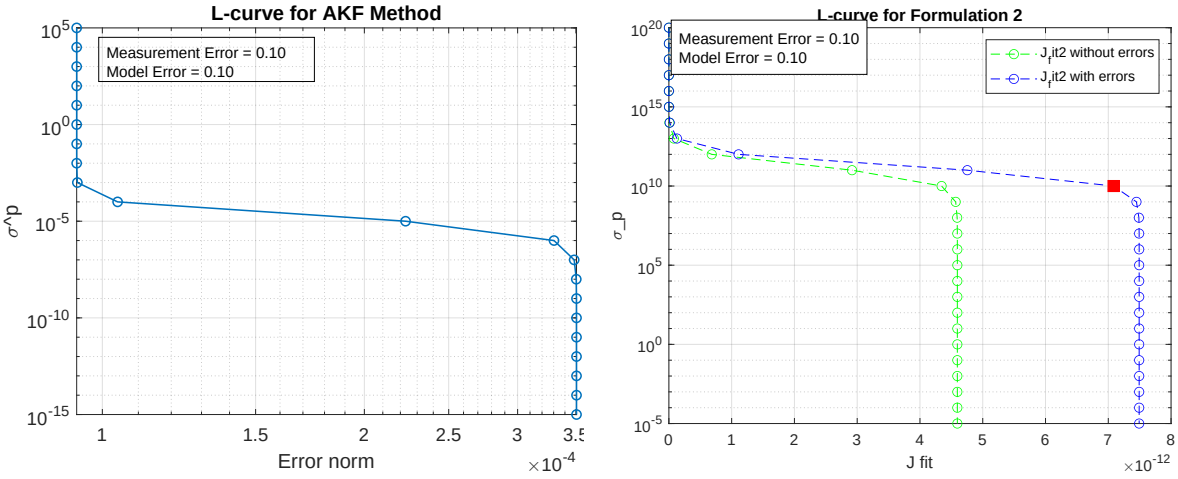

Figure 5.28: Spring-Mass Model - Case 6

Moving to the sixth Case, according to figure 5.28, input of the model changes from sinusoidal to double impact excitation, implemented at 5th mass ($m_5$) instead of ($m_2$), as in the previous cases. Only one acceleration sensor monitors the system, also at the fifth mass.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

There aren't any unknown input-parameters to estimate.




(a) L-curve for AKF Method

(b) L-curve for Formulation 2

Figure 5.29: L-curve for error 0.10 - Case 6

L-curves for AKF & Formulation 2 are illustrated at figures 5.29a & 5.29b for error value $e = 0.10$. Optimal $\sigma_{p1}$ and $\sigma_{p3}$ values are selected as $10^{-3}$ and $10^{14}$ correspondingly. Manual selection needed, meaning that automated selection proposed in Section 4.2 was ineffective in this case.

Input plots for error values equal to 0.05 and 0.10 are presented at figure 5.30. All the methods, with or without analysis, identify the loads with decent precision. However, modal analysis is slightly more precise between the two impacts and after the second impact time intervals, when there is no force acting on the fifth mass. All-modes implementation maintains some perturbation in the corresponding time margins, compromising load identification fidelity.

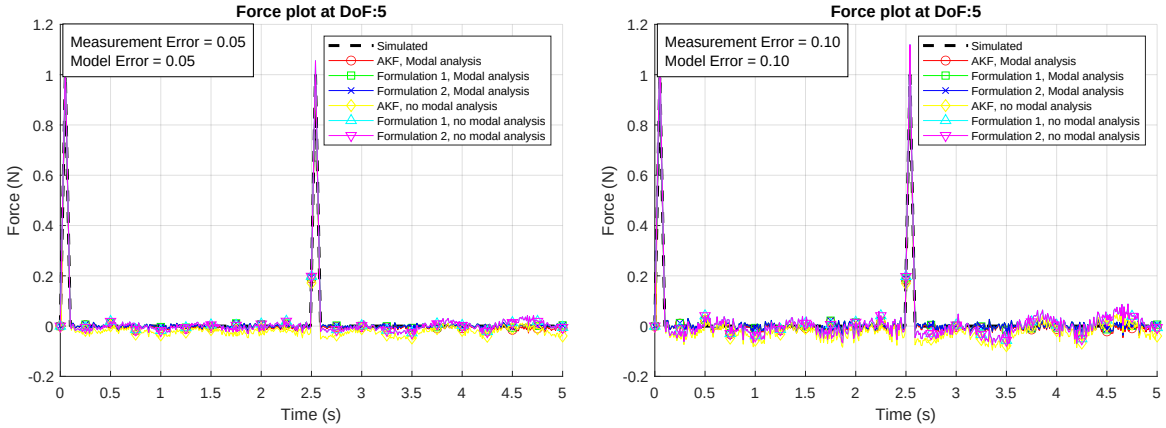


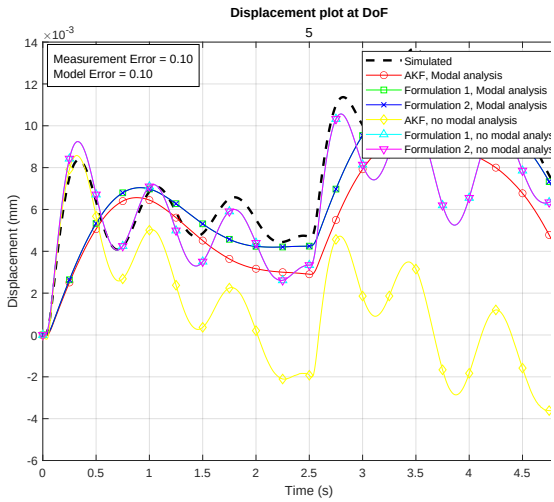

Figure 5.30: Input Plots - Case 6

The responses of the model are reconstructed as illustrated at figures 5.31, 5.32 and 5.33. In general, all-modes analysis is for preference, as less mode usage cannot estimate the responses precisely. Regarding displacement plots, 5.31a and 5.31b figures show that proposed formulations 1 and 2 stand clearly better than Augmented Kalman Filter, not only

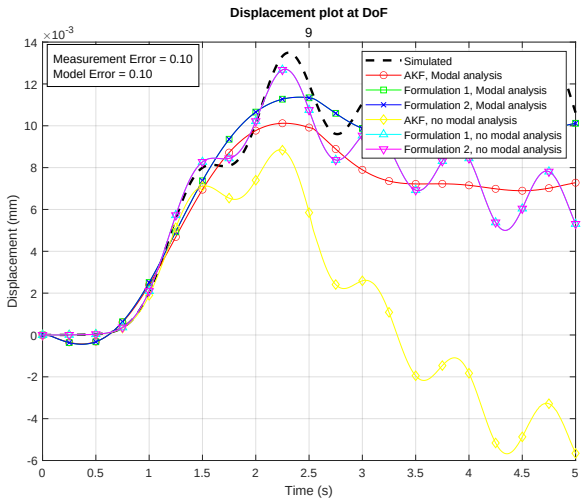


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

far from sensor ( figure 5.31b) but also at the monitored mass (figure 5.31a).




(a) Displacement Plot at 5th DOF

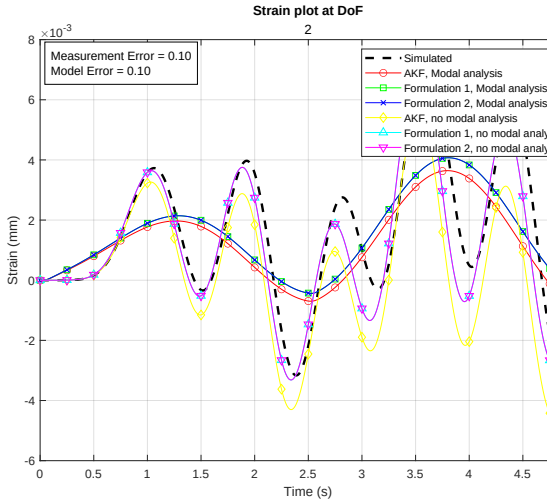



(b) Displacement Plot at 9th DOF

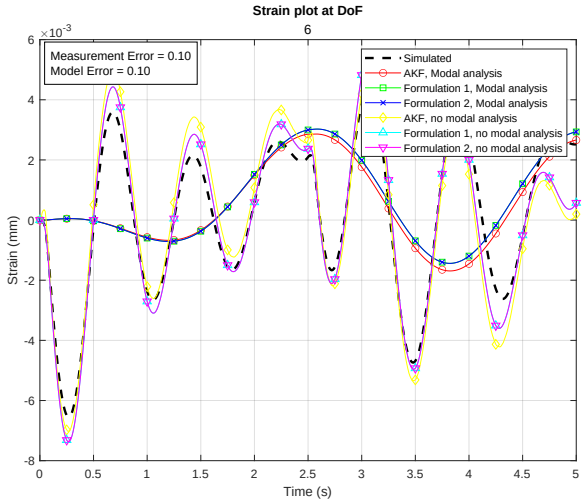
Figure 5.31: Displacement Plots for error 0.10 - Case 6

Considering strain responses,as shown at figure 5.32b, Augmented Kalman Filter is only slightly inferior to the suggested Formulations 1 and 2 in terms of precision. On the contrary, Formulation 1 and Formulation 2 record once again greater accuracy far from the installed sensor (2nd DOF), as shown at 5.32a.

Finally, velocity figures are illustrated for an error equal to 0.10 at the input $m_5$ and at the eighth mass $m_8$. Once again, Formulations 1 and 2 estimate with higher-fidelity the velocities far from the sensor (figure 5.33b). Also, at the monitored sensor (figure 5.33a), all methods show similar capability. Indisputably, the proposed methods seem more suitable in case of one acceleration sensor at input.




(a) Strain Plot at 2nd DOF

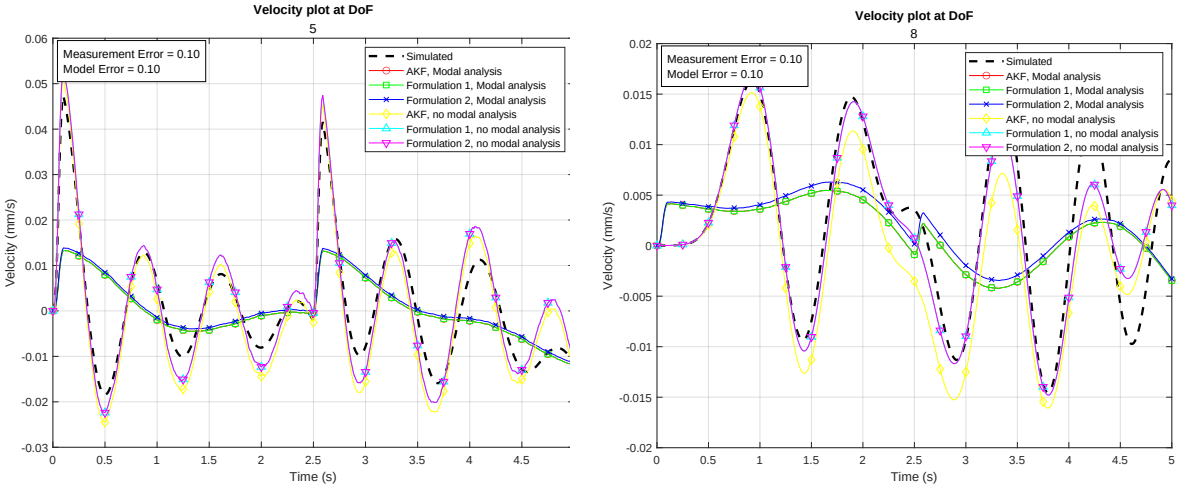



(b) Strain Plot at 6th DOF

Figure 5.32: Strain Plots for error 0.10 - Case 6



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250




(a) Velocity Plot at 5th DOF

(b) Velocity Plot at 8th DOF

Figure 5.33: Velocity Plots for error 0.10 - Case 6



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.7 Case 7

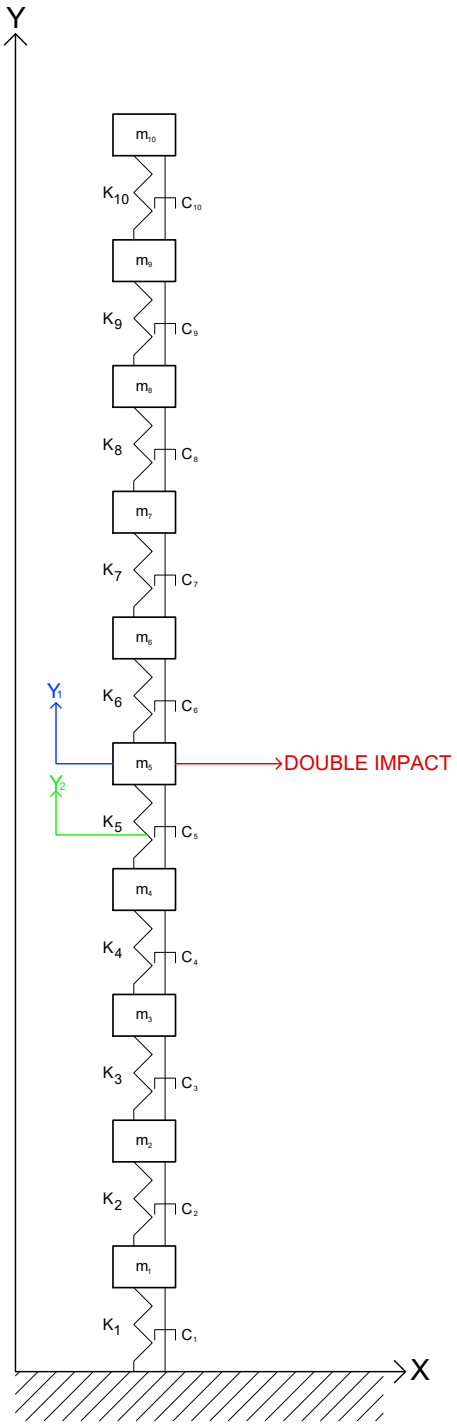


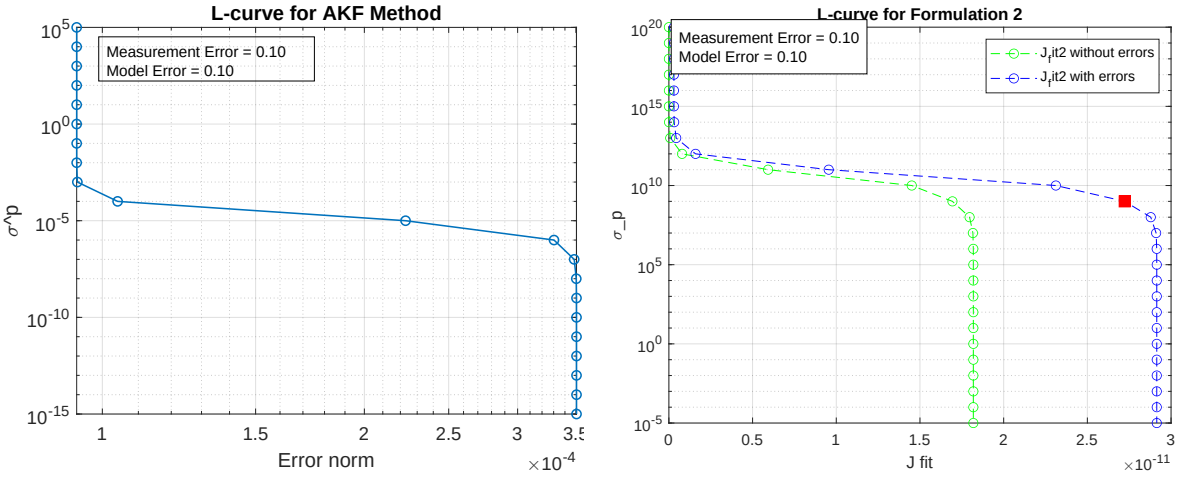

Figure 5.34: Spring-Mass Model - Case 7

Derived from figure 5.34, apart from the accelerator to monitor the input, there is also a strain sensor installed in the fifth spring, $k_5$, creating a set of collocated sensors configuration. All spring parameters are known. Double impact remains the excitation acting on the fifth



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

mass, $m_5$. This case was tested once again for varying errors, starting from zero error values to 0.10 ones.




(a) L-curve for AKF Method

(b) L-curve for Formulation 2

Figure 5.35: L-curve for error 0.10 - Case 7

L-curves for Augmented Kalman Filter and Formulation 2 are presented at 5.35. According to figure 5.35a, optimal Covariance Matrix for Augmented Kalman Filter stands equal to $\sigma_{p1} = 10^{-3}$, while, based on 5.35b, optimal value is equal to $\sigma_{p3} = 10^{14}$, respectively. Additionally, considering the red square at figure 5.35b, automated methodology proposed at Section 4.2 was unsuccessful. Moreover,input plots for error values equal to 0.05 and 0.10 are presented at figure 5.36. All the methods, with or without analysis, identify the loads with decent precision. However, once again modal analysis is slightly more precise regarding the unexcited time-intervals.

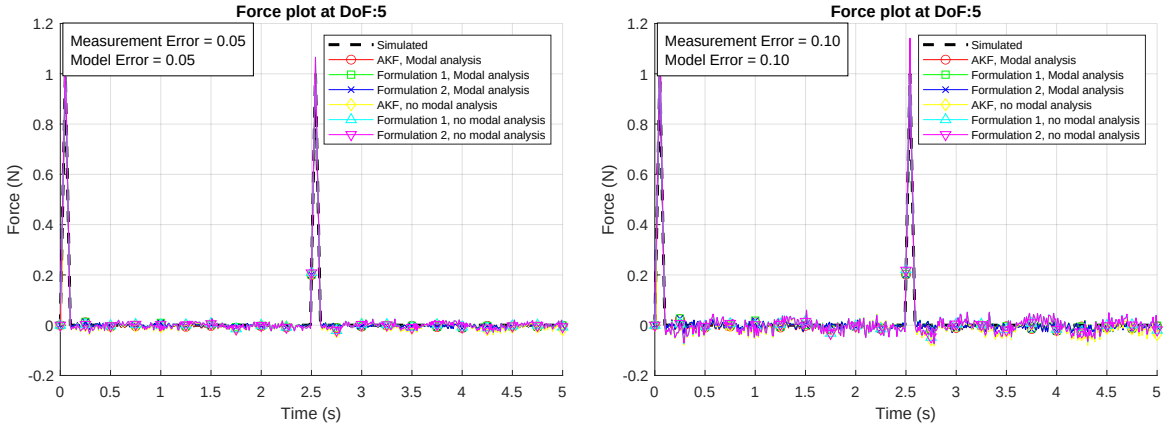



Figure 5.36: Input Plots - Case 7

The results for the displacements, strains and velocities are illustrated at figures 5.37, 5.38 and 5.39 respectively. These figures consider various DOF across the model. Compared to the sixth Case, it can be easily observed that estimation accuracy for all the responses improved considerably, regardless of the DOF examined. Once more, modal analysis couldn't manage to gather trustworthy response reconstruction data.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

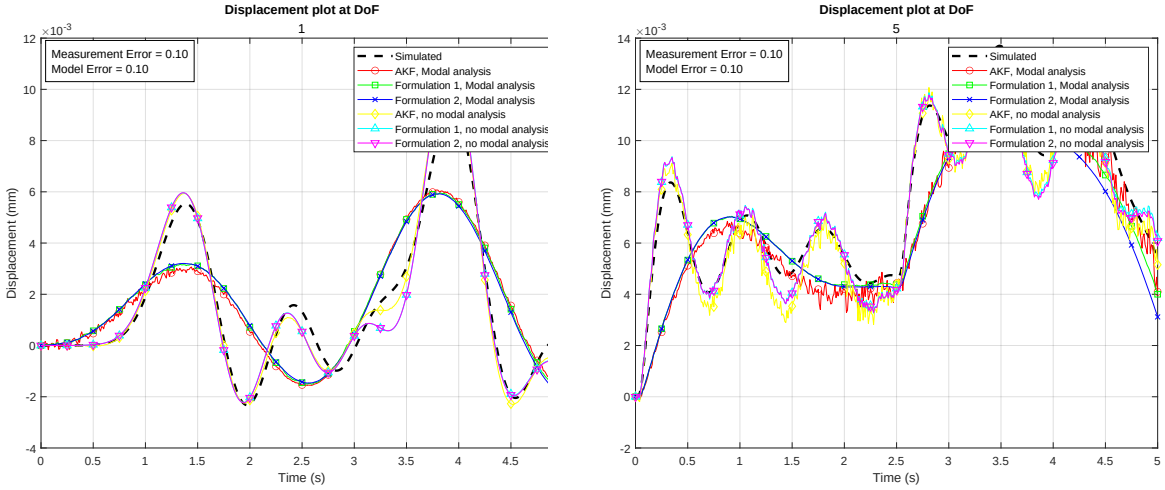



Figure 5.37: Displacement Plots - Case 7

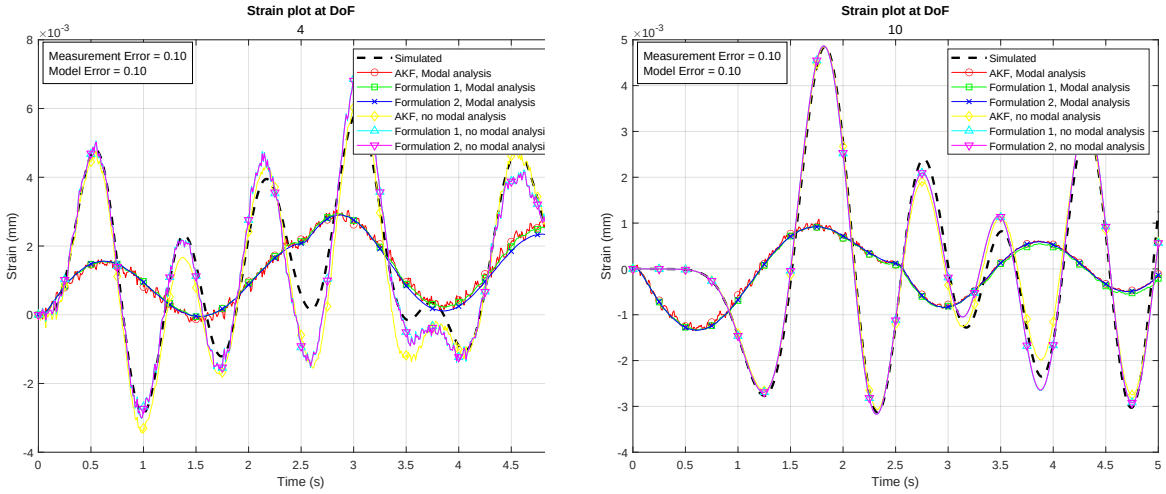



Figure 5.38: Strain Plots - Case 7

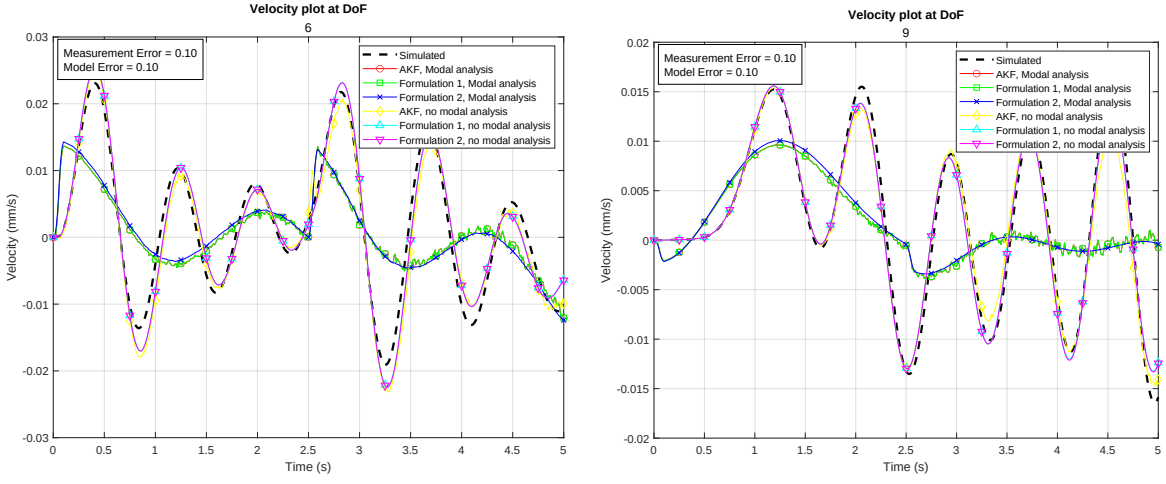



Figure 5.39: Velocity Plots - Case 7



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.8 Case 8

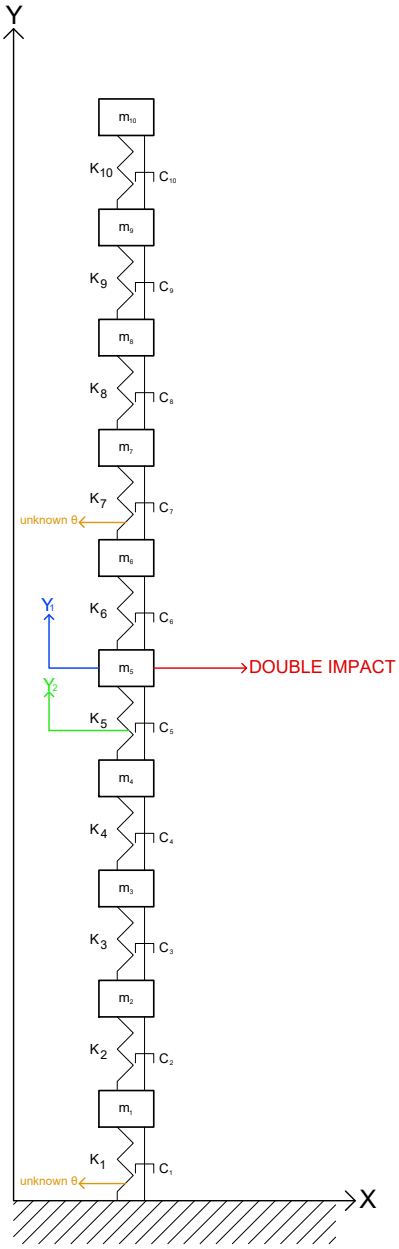



Figure 5.40: Spring-Mass Model - Case 8

Derived from figure 5.40, and also considering 5.34, the only difference lies upon the lack of information about spring parameters ($k_1$ and $k_7$). Except for load identification and response reconstruction, within this case input-parameter estimation will be performed.



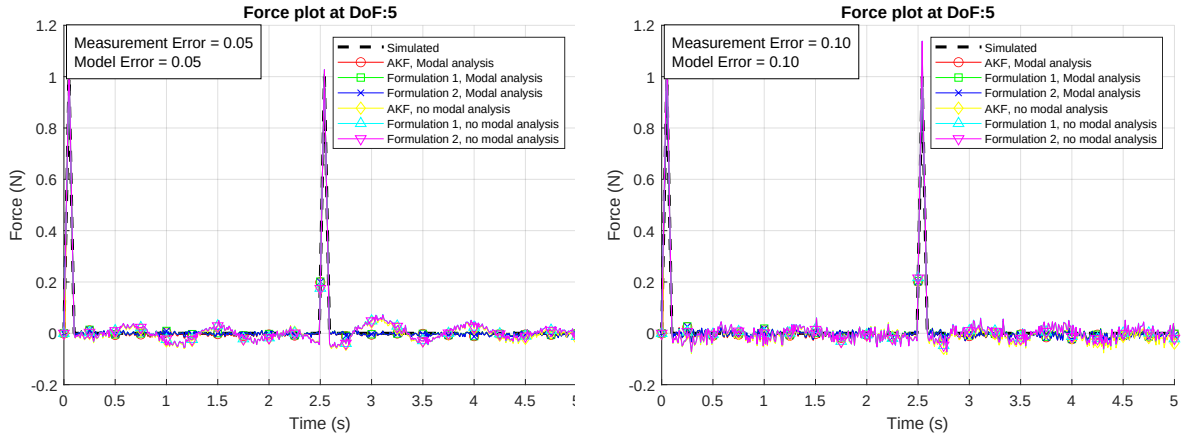
Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Beginning with the input-parameter estimation, by observing table 5.3, is concluded that spring estimation near the strain sensor (1st DOF) was successful, with input parameter relative error recording low values while, on the other hand, spring estimation at 7th DOF failed considerably. This is expected due to the number of installed strain sensors. It also indicates **the necessity of adding another strain sensor near the other unknown spring parameter**. Moreover, compared to the third case (table 5.1), clearly better results are observed, meaning that **Formulation 2 estimates unknown input-parameters more effectively for impact loads**. Optimal Covariance Matrices $\sigma_{p1}$ and $\sigma_{p3}$ are considered unaltered.

| $\Delta\theta\%$ **Input Parameter Relative Error** | | | | |
|---|---|---|---|---|
| **DOF** | **e = 0.00** | **e = 0.02** | **e = 0.05** | **e = 0.10** |
| 1 | 0.03 | 0.65 | 1.68 | 3.88 |
| 7 | 72.27 | 67.19 | 58.27 | -10.47 |

Table 5.3: $\Delta\theta\%$ Relative Error - Case 8

Input plots are presented at figure 5.41. It is observed that applying modal analysis achieves high-fidelity predictions **irrespective of the error values or the input-parameter estimation precision**. Additionally, it is remarkable that all-modes analysis records better precision for error equal to 0.10 than for error equal to 0.05, as illustrated at 5.41b and 5.41a. Preciser $\underline{\theta_7}$, as derived from table 5.3 stands the reason behind this observation.




(a) Input Plot for error 0.05

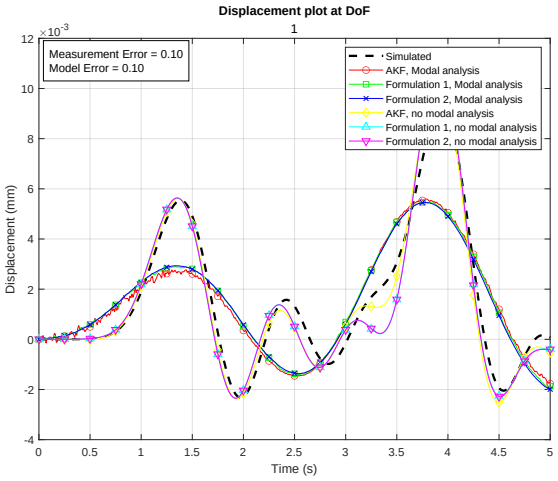
(b) Input Plot for error 0.10

Figure 5.41: Inputs Plots for variable errors - Case 8

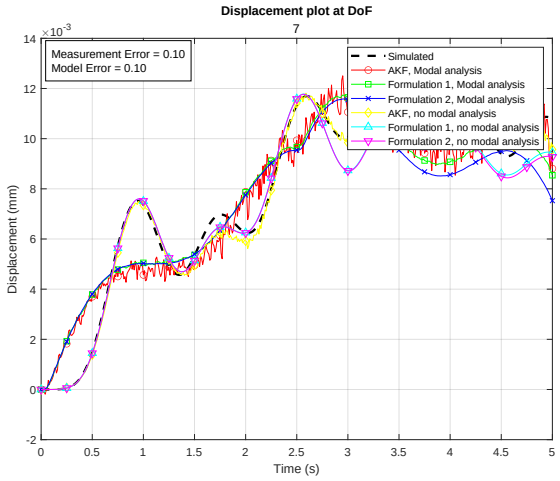
Responses of the model are reconstructed for the predicted springs $\underline{\theta_1}$ and $\underline{\theta_7}$ are presented at figures 5.42, 5.43 and 5.44 for displacements, strains and velocities correspondingly. Considering findings from table 5.3, where $\%\Delta\underline{\theta_1}$ prediction was more accurate than $\%\Delta\underline{\theta_7}$ one, is observed for displacement responses slightly better accuracy for Augmented Kalman Filter (figure 5.42b against proposed Formulations at the 7th DOF, while all methods claim similar fidelity at 1st DOF (figure 5.42a). Analogous deductions can be made also from velocity and strain responses, as shown at figures 5.43b, 5.44b and 5.43a 5.44a respectively.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

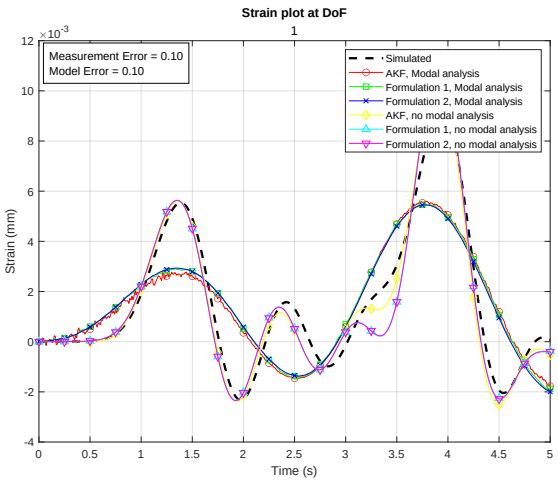



(a) Displacement Plot at 1st DOF

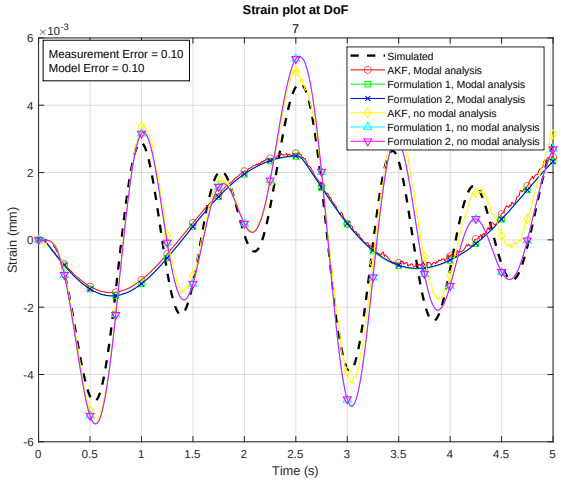



(b) Displacement Plot at 7th DOF

Figure 5.42: Displacement Plots for error 0.10 - Case 8

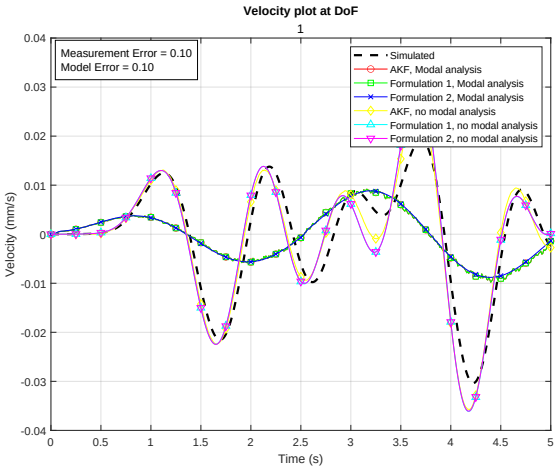



(a) Strain Plot at 1st DOF

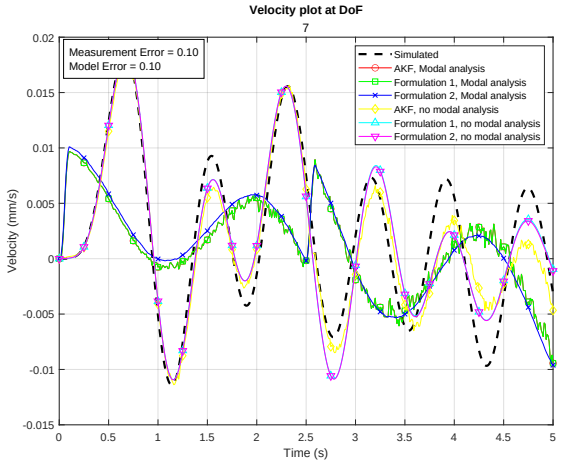



(b) Strain Plot at 7th DOF

Figure 5.43: Strain Plots for error 0.10 - Case 8




(a) Velocity Plot at 1st DOF




(b) Velocity Plot at 7th DOF

Figure 5.44: Velocity Plots for error 0.10 - Case 3



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.9 Case 9

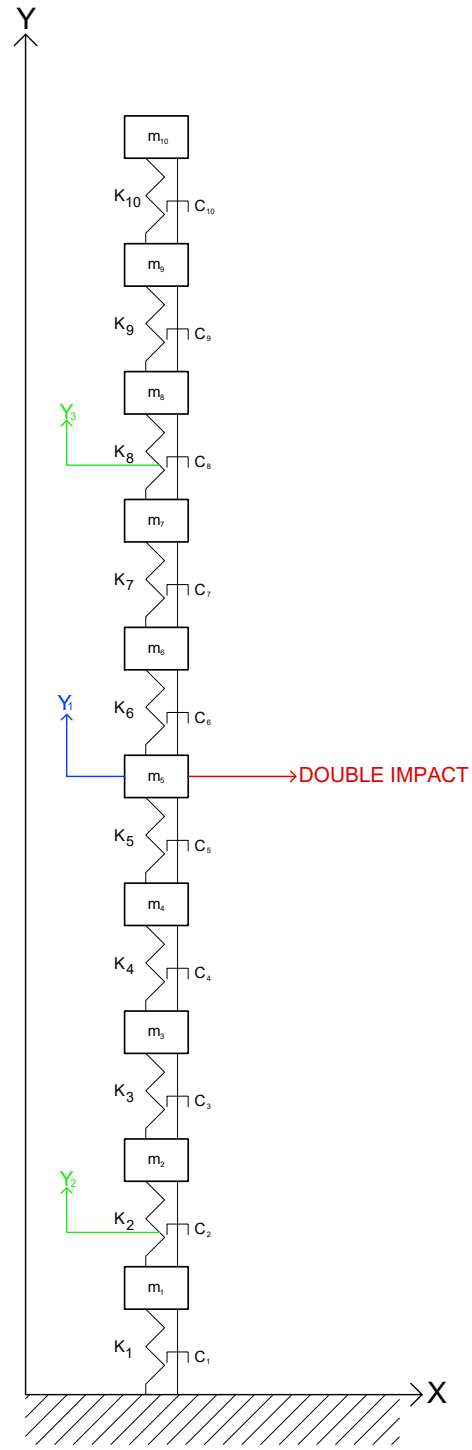



Figure 5.45: Spring-Mass Model - Case 9

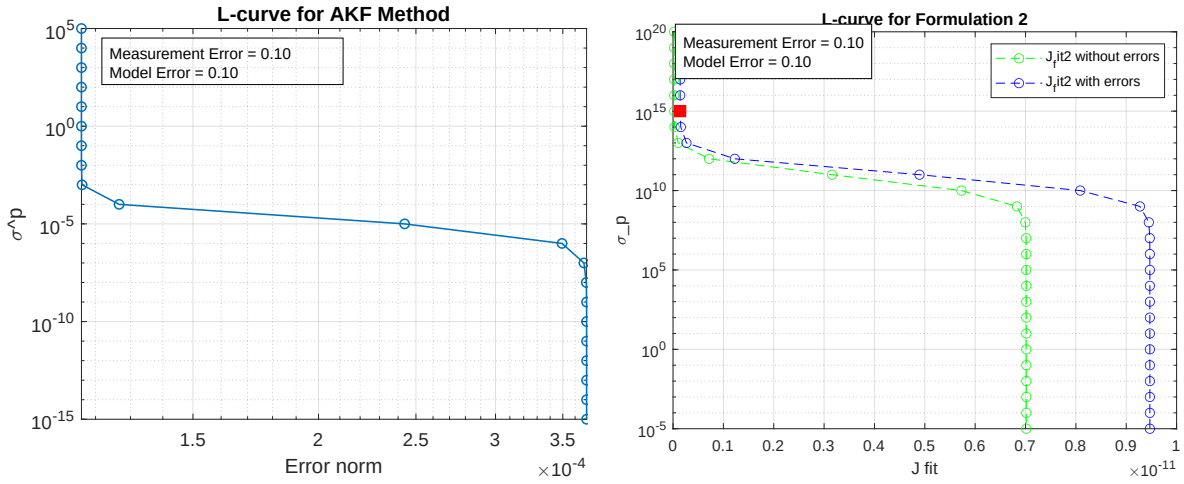
Derived from figure 5.45, strain sensor at 5th spring is removed and two strain sensors are installed at 2nd & 8th spring, $k_2$ & $k_8$, respectively. All spring parameters are known, meaning that there isn't input-parameter necessity in this case. Force excitation remains



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

double-impact load, implemented at $m_5$.

L-curve curves for AKF & Formulation 2 are illustrated at figures 5.46a & 5.46b for error values $e = 0.10$. $\sigma_{p1}$ and $\sigma_{p3}$ values are selected as $10^{-3}$ and $10^{14}$ correspondingly. Manual selection needed, meaning that automated selection proposed in Section 4.2 was ineffective in this case.




(a) L-curve for AKF Method

(b) L-curve for Formulation 2

Figure 5.46: L-curve for error 0.10 - Case 9

Input plot for error equal to 0.10 is presented at figure 5.47. Regarding all-modes analysis, placement of an extra sensor enhances the accuracy of load identification predictions (yellow, light-blue and magenta colored lines). Nevertheless, modal analysis remains preciser (red, green and blue colored lines) because it generates less noise during unexcited time-intervals.

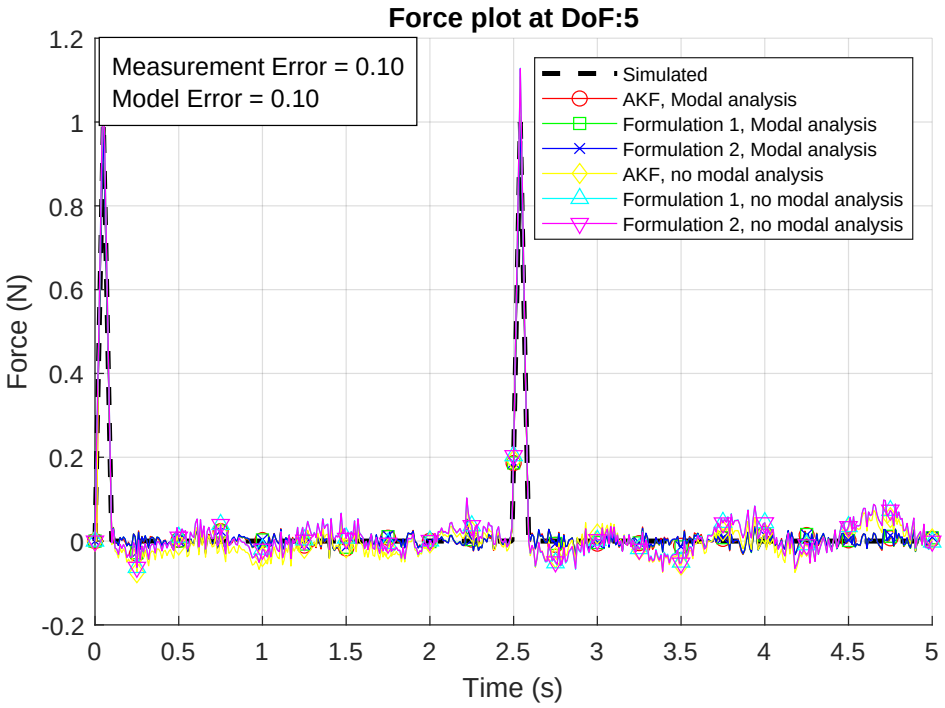
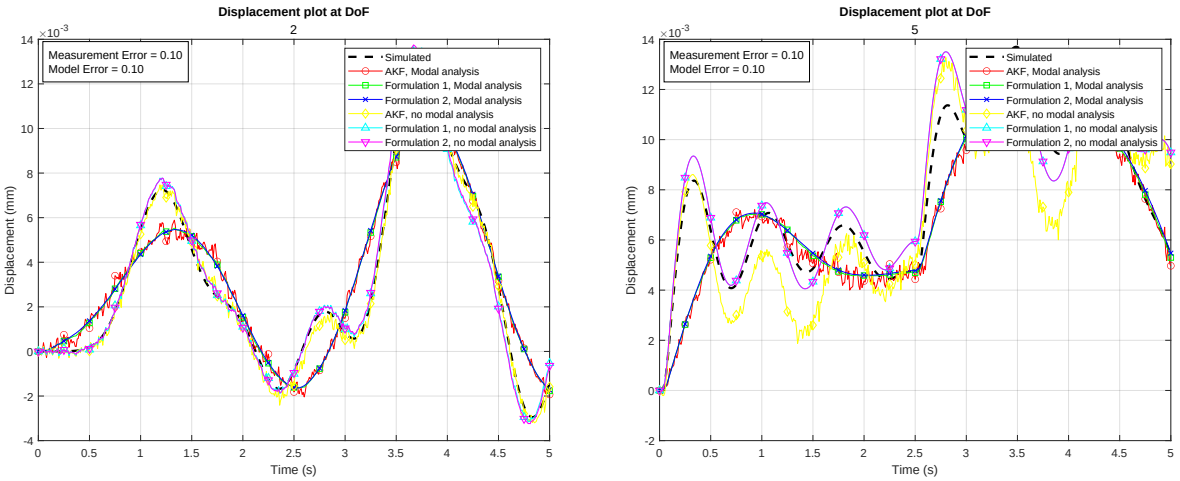



Figure 5.47: Input Plot for error 0.10



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

Results of displacements, strains and velocities for second and fifth DOF are presented at figures 5.48, 5.49 and 5.50. Regarding displacement responses, near strain sensor position (figure 5.48a), equal prediction capability between Augmented Kalman Filter, Formulation 1 and Formulation 2 is observed. On the contrary, at 5th DOF where the distance of the strain sensors is larger, Formulations 1 & 2 seem to have an advantage over Augmented Kalman Filter (figure 5.48b). Modal analysis records once more poor estimation fidelity against all-modes analysis.

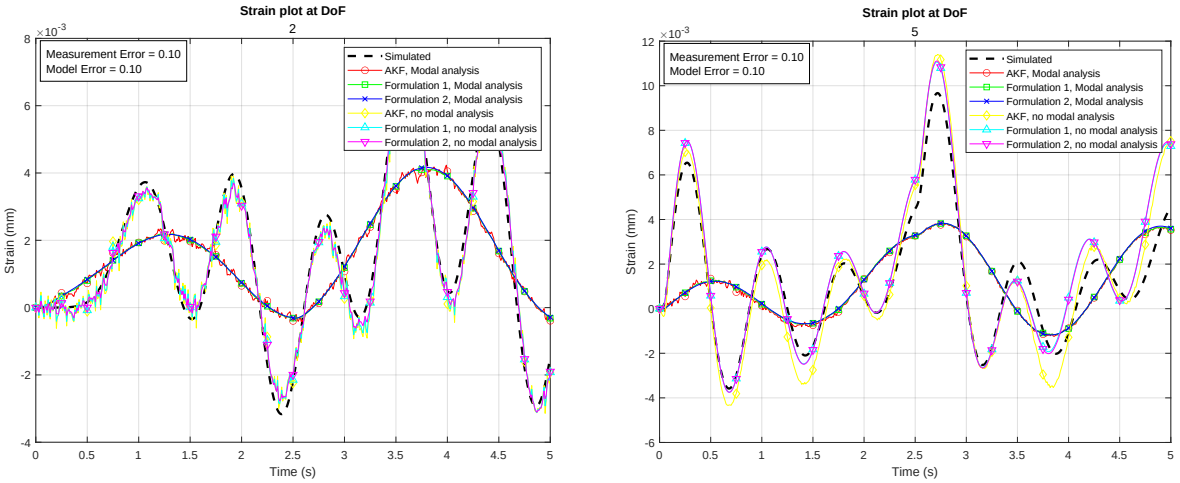



(a) Displacement Plot at 2nd DOF

(b) Displacement Plot at 5th DOF

Figure 5.48: Displacement Plots for error 0.10 - Case 9

Considering reconstructed responses of strains and velocities, is observed similar fidelity for Augmented Kalman Filter, Formulation 1 and Formulation regardless of the distance from the installed strain sensor, as shown at 5.49a, 5.49b figures for strains and at 5.50a, 5.50b figures for velocities, respectively. Finally, once again Mode analysis implementation of these methods consistently yield imprecise predictions, concluding that this approach is unreliable for response reconstruction.



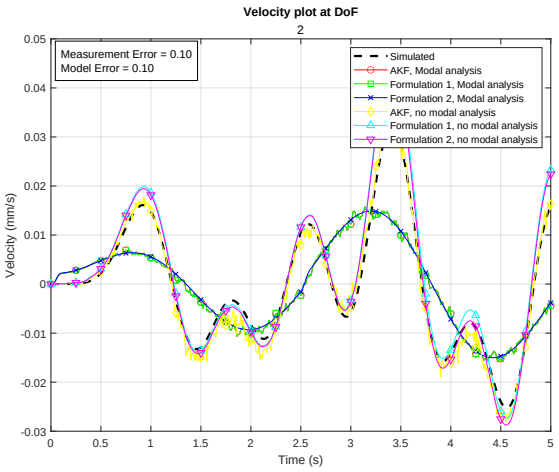

(a) Strain Plot at 2nd DOF

(b) Strain Plot at 5th DOF

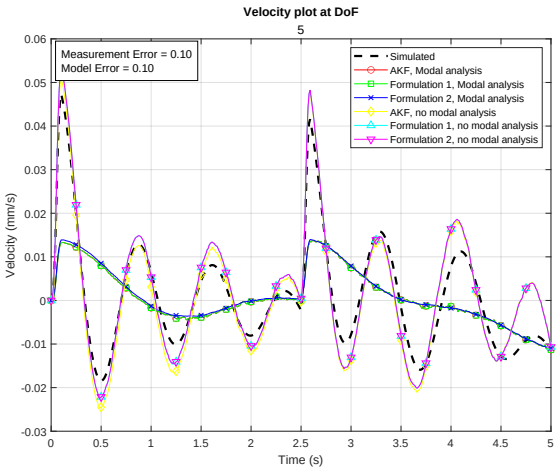
Figure 5.49: Strain Plots for error 0.10 - Case 9



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250




(a) Velocity Plot at 2nd DOF




(b) Velocity Plot at 5th DOF

Figure 5.50: Velocity Plots for error 0.10 - Case 3



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 5.10 Case 10

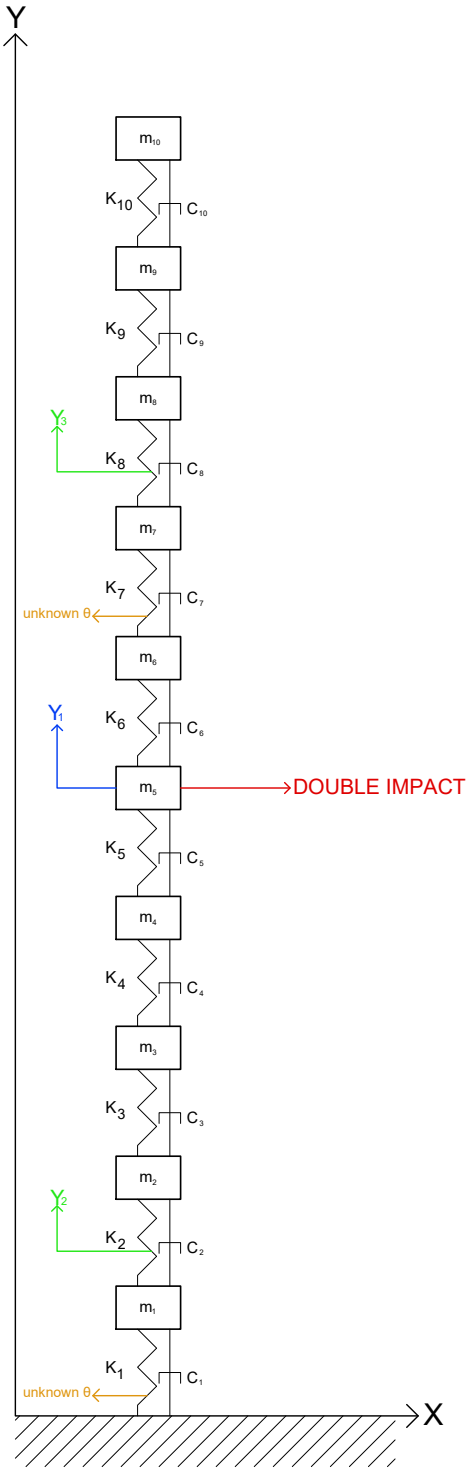



Figure 5.51: Spring-Mass Model - Case 10

Derived from figure 5.51,and also regarding 5.45, this case differentiates only at the unknown input-parameters, namely $\underline{\theta_1}$ and , $\underline{\theta_7}$, which need estimation. Table 5.4 shows the input-parameter relative error results. Decent-fidelity input-parameter estimation is achieved

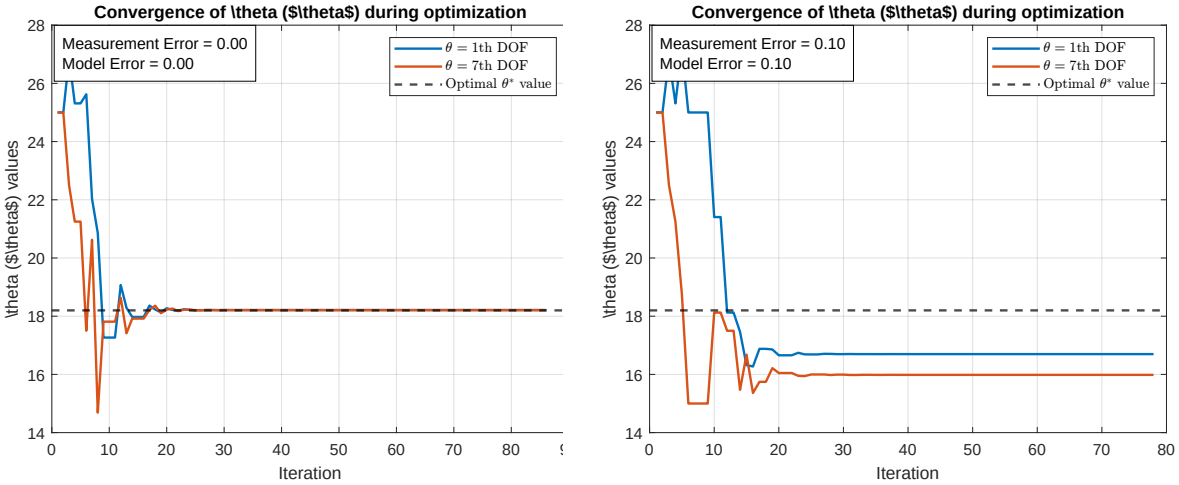


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

for varying error values. According to figure 5.52, $\underline{\theta}$-convergence is performed quite fast, approximately 25 iterations, with no difference between no error (5.52a) and an error value of 0.10 (5.52b). Optimal values of covariances are set $\sigma_{p1} = 10^{-3}$ and $\sigma_{p3} = 10^{14}$, as in Case 9.

| $\Delta\theta\%$ **Input Parameter Relative Error** | | | | |
|---|---|---|---|---|
| **DOF** | **e = 0.00** | **e = 0.02** | **e = 0.05** | **e = 0.10** |
| 1 | 0.06 | -1.7 | -4.18 | -8.26 |
| 7 | 0.05 | -2.4 | -6.07 | -12.18 |

Table 5.4: $\Delta\underline{\theta}\%$ Relative Error - Case 10




(a) $\underline{\theta}$-Convergence Plot for no error

(b) $\underline{\theta}$-Convergence Plot for error 0.10

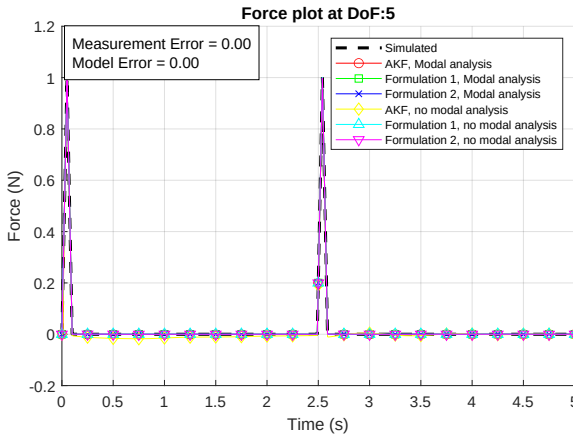
Figure 5.52: $\underline{\theta}$-Convergence Plots for variable error - Case 10

Input plots for varying error are shown at figure 5.53. Increasing the error from zero (figure 5.53a) to 0.10 (figure 5.53b) was the aim, in order to research the prediction capability of the formulations with the evolution of error during simulation. They can be observed consistent, high-fidelity predictions for modal analysis despite the increase in the model perturbation. Augmented Kalman Filter, Formulation 1 and Formulation 2 show similar capability. On the other hand, load estimates of an all-modes analysis are deteriorating while the perturbation of the model is increased.

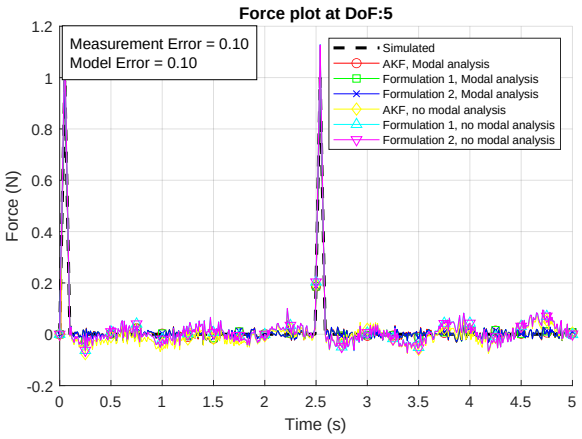
The results for the displacement, strain and velocity at 7th DOF, with unknown spring parameter, are presented at figures 5.54, 5.55 and 5.56. It can observed once more that modal analysis (red, green and blue lines in the figures), opposite to inputs, doesn't achieve decent accuracy for the responses, even for zero error condition (figures 5.54a, 5.55a & 5.56a). Regarding all-mode analysis, (yellow light-blue and magenta lines), it can be observed from figures 5.54b, 5.55b & 5.56b for displacement, strain and velocity responses that Augmented Kalman Filter stand slightly more precise than the proposed formulations.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

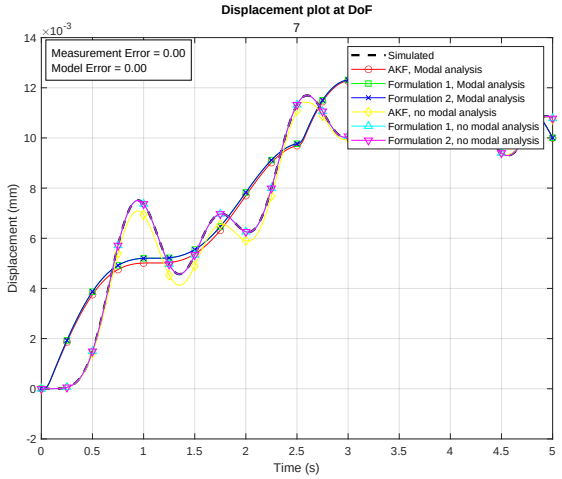



(a) Input Plot for no error

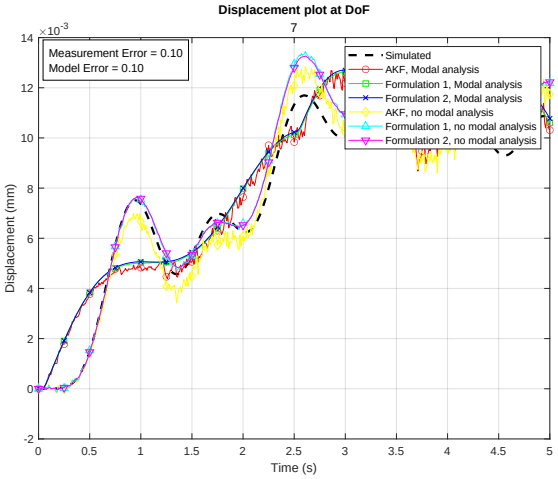



(b) Input Plot for error 0.10

Figure 5.53: Input Plots for varying error - Case 10

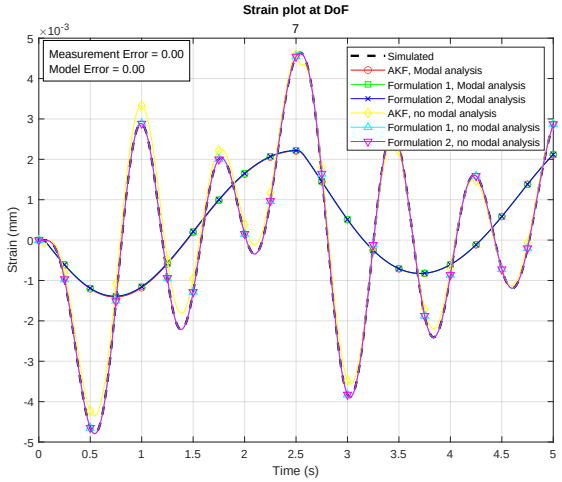



(a) Displacement Plot for no error

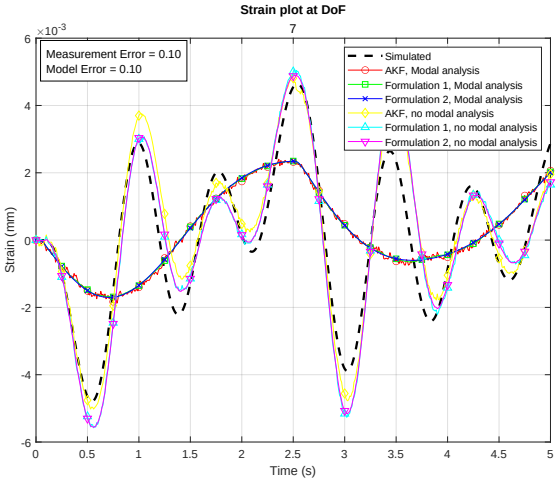



(b) Displacement Plot for error 0.10

Figure 5.54: Displacement Plots at 7th DOF - Case 10




(a) Strain Plot for no error

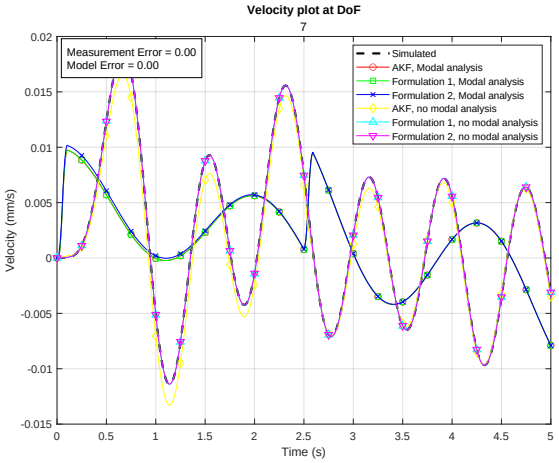



(b) Strain Plot for error 0.10

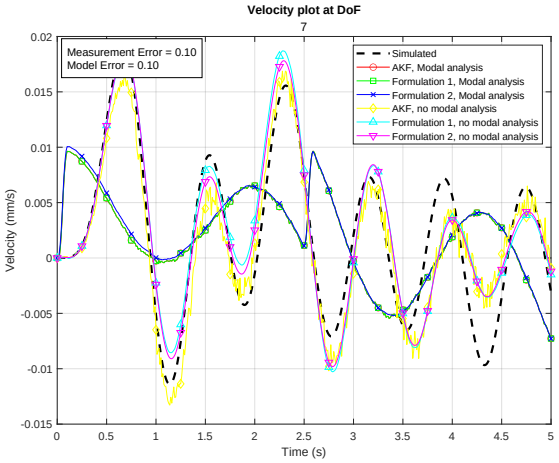
Figure 5.55: Strain Plots at 7th DOF - Case 10



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250




(a) Velocity for no error




(b) Velocity Plot for error 0.10

Figure 5.56: Velocity Plots for error 0.10 - Case 3



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Chapter 6: Conclusion

## 6.1 Deductions

Undoubtedly, a lot of useful deductions can be made from the simulated examples. These are recorded as follows:

1. Regarding sinusoidal excitation, acceleration sensor installation contributes to input estimation, although it doesn't reconstruct displacements, velocities and strains accurately.
2. Regarding double impact excitation, proposed Formulations 1 & 2 achieved reliable and higher-fidelity estimates than Augmented Kalman Filter for acceleration sensor installed at input.
3. Collocated sensors configuration provides high-fidelity input and state estimations irrespective of excitation.
4. Regardless of the excitation, proposed Formulations 1 & 2 are slightly more accurate at reconstructing responses far from installed sensors compared to Augmented Kalman Filter for collocated sensors configuration.
5. Regarding Load Identification, modal analysis implementation showed consistent accuracy in all cases even for high error values, especially against all-modes analysis, for the excitations applied to the model apiece.
6. Concerning input-parameter estimation, the number of strain sensors in conjunction with their placement to the model affect the procedure precision.
7. Considering reconstruction of responses, all-mode analysis is for preference. Modal analysis struggles to estimate the responses.
8. Augmented Kalman Filter reconstructs the responses slightly more accurately for poorly-estimated input-parameters compared to the proposed Formulations.
9. In case of precise input-parameter prediction, Augmented Kalman Filter aforementioned advantage against Formulations 1 & 2 minimizes.
10. In case of adequate number of sensors, $\underline{\theta}$ convergence is performed accurately and quickly (approximately 25-35 iterations) for limited number of unknown input-parameters.
11. In the majority of cases, automated Covariance Matrix identification proved unsuccessful, necessitating manual selection of $\sigma_p$ values.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

## 6.2 Suggestions for Future Research

In this section, some though-provoking suggestions for future research are proposed, lying into both theoretical and experimental grounds.

Firstly, it would be very interesting to propose a Data-Driven Method for optimal $\sigma_{p1}$ and $\sigma_{p3}$ estimation. As it was shown from the results section, automated method for load identification is not a panacea, meaning that manual check on every L-curve diagram is more than necessary. Hence, gathering a solid & uncontaminated amount of data, training a Machine Learning Model and test it with realistic values in order to identify with powerful level of confidence $\sigma_{p1}$ and $\sigma_{p3}$ values of a model will save cost and time from the manual observation of the corners at L-curve diagrams.

The second suggestion is more of a theoretical value. More specifically, is considered interesting to apply Fractal Calculus in order to come up with an innovative formulation for load identification & response reconstruction. The Mathematical Background required is considered much more complicated because it needs deep understanding of Gamma functions, as pointed out in (Aleroev & Elsayed, 2022). Several works are cited that implemented Fractal Calculus, mainly researches of (Leyden & Goodwine, 2016) and (Leyden & Goodwine, 2018) which refer to complex mechanical systems.

Last, but not least, very interesting proposition directed towards realistic conditions, would be to test these formulations into the Aircraft Structural Integrity Program for the Hellenic Air Force, mainly for the F-16 Fighting Aircraft. As mentioned earlier in the Introduction Section, there is both scientific interest not only to assess the possible structural degradation of F-16 during their operational life for economic reasons, but also in a matter of accomplishing safe flights (Pidaparti, 2006), by diagnosing future failures (Wells & King, 1970) & (Clark & Jackson, 2010). Herein, this kind of collaboration poses as a critical test for the formulations whatsoever given the complex nature of F-16 forces and stresses during flights.

Indisputably, Hellenic Air Force is an organization which can provide well-posed data for objective analyses. Plus, 111 Combat Wing in Nea Agxialos, Magnesia, being a military Airport with plenty of airplanes susceptible to flights with fighting origins, could provide the opportunity for collaboration. It is useful to mention that there is already a platform working on response reconstruction & load identification in depot level, exploiting the Ground Vibration Testing framework. Hence, testing and applying these proposed formulations will achieve a lot of benefits not only for the organization but also for the University's prestige.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# References


Aleroev, T. S., & Elsayed, A. M. (2022). Analytical and numerical methods to solve the fractional model of the vibration equation. In H. Singh, H. M. Srivastara, & J. J. Nieto (Eds.), *Handbook of fractional calculus for engineering and science advances in applied mathematics series editors* (p. 1-27). Taylor & Francis. Retrieved from `https://www.routledge.com/Advances-in-Applied-Mathematics/book-`

Azam, S. E., Chatzi, E., & Papadimitriou, C. (2015, 2). A dual kalman filter approach for state estimation via output-only acceleration measurements. *Mechanical Systems and Signal Processing*, *60-61*, 866-886.

Azam, S. E., Chatzi, E., Papadimitriou, C., & Smyth, A. (2017, 8). Experimental validation of the kalman-type filters for online and real-time state and input estimation. *JVC/Journal of Vibration and Control*, *23*, 2494-2519. doi: 10.1177/1077546315617672

Chen, X., Pan, C., Peng, S., Zeng, H., Wang, X., & Liu, X. (2024, 9). Structural response reconstruction of beam-like bridge based on equivalent loads under moving forces. *Structures*, *67*. doi: 10.1016/j.istruc.2024.107026

Chen, Z., Weng, S., Yu, H., Li, J., Zhu, H., Yan, Y., & Wu, L. (2023, 2). Bayesian-based method for the simultaneous identification of structural damage and moving force. *Mechanical Systems and Signal Processing*, *185*. doi: 10.1016/j.ymssp.2022.109742

Clark, G., & Jackson, P. (2010, 11). Structural integrity and damage type in military aircraft. *Fatigue and Fracture of Engineering Materials and Structures*, *33*, 752-764. doi: 10.1111/j.1460-2695.2010.01486.x

Daitong, W., Yugang, C., Hongkun, L., & Xiaowen, Z. (2023, 4). Real-time reconstruction method of full-field dynamic response of rotating bladed disks. *Mechanical Systems and Signal Processing*, *188*. doi: 10.1016/j.ymssp.2022.109953

Dollon, Q. (2023, 7). Efficient structural model updating with spatially sparse modal data: A bayesian perspective. *Mechanical Systems and Signal Processing*, *195*. doi: 10.1016/j.ymssp.2023.110324

Fu, D., Wang, L., Lv, G., Shen, Z., Zhu, H., & Zhu, W. D. (2023, 11). *Advances in dynamic load identification based on data-driven techniques* (Vol. 126). Elsevier Ltd. doi: 10.1016/j.engappai.2023.106871

Giagopoulos, D., Arailopoulos, A., Dertimanis, V., Papadimitriou, C., Chatzi, E., & Grompanopoulos, K. (2019, 7). Structural health monitoring and fatigue damage estimation using vibration measurements and finite element model updating. *Structural Health Monitoring*, *18*, 1189-1206. doi: 10.1177/1475921718790188

Hansen, P. (2001). *Computational inverse problems in electrocardiology.* WIT Press.

Hayes, B. (1996). *Classic brittle failures in large welded structures* (Vol. 3; Tech. Rep.).




Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

He, J., & Fu, Z.-F. (2001). *Modal analysis*. Butterworth - Heinemann.

Hu, Q., Zhou, N., Chen, H., & Weng, S. (2023, 11). Bayesian damage identification of an unsymmetrical frame structure with an improved pso algorithm. *Structures*, *57*. doi: 10.1016/j.istruc.2023.105119

Jayalakshmi, V., Lakshmi, K., & Rao, A. R. M. (2020, 3). Reconstruction of impact load on structures with limited measurements using dynamic hybrid adaptive differential search algorithm. *Inverse Problems in Science and Engineering*, *28*, 340-373. doi: 10.1080/17415977.2019.1567725

Jensen, H., & Papadimitriou, C. (2019). *Coupling for dynamic analysis application to complex simulation-based problems involving uncertainty* (Vol. 89; P. Wriggers & P. Eberhad, Eds.). Springer. Retrieved from `http://www.springer.com/series/4623`

Jiang, K., Han, Q., Du, X., & Ni, P. (2021, 11). Structural dynamic response reconstruction and virtual sensing using a sequence to sequence modeling with attention mechanism. *Automation in Construction*, *131*. doi: 10.1016/j.autcon.2021.103895

Kantouris, P. (2024). *Αναγνωριση κρυστικης δυναμης σε πτερυγι απ συνθετα υλικα και ανακατασκευη της απκρισης τυ μντελυ* (Unpublished doctoral dissertation). Aristotle University of Thessaloniki.

Karlsson, S. (1996, 2). Identification of external structural loads from measured harmonic responses. *Journal of Sound and Vibration*, *196*, 59-74.

Ketseas, D. (2025, 7). *Ground vibration testing (gvt) – Πιστπιηση Αερσκαφων και Εξωτερικων Φρτιων.*

Koutsoupakis, J., Seventekidis, P., & Giagopoulos, D. (2023, 5). Machine learning based condition monitoring for gear transmission systems using data generated by optimal multibody dynamics models. *Mechanical Systems and Signal Processing*, *190*. doi: 10.1016/j.ymssp.2023.110130

Leyden, K., & Goodwine, B. (2016, 5). Using fractional-order differential equations for health monitoring of a system of cooperating robots. In *International conference on robotics and automation (icra)* (p. 366-371). IEEE.

Leyden, K., & Goodwine, B. (2018, 5). Fractional-order system identification for health monitoring. *Nonlinear Dynamics*, *92*, 1317-1334. doi: 10.1007/s11071-018-4128-y

Li, Y., Luo, L., Wang, X., Zhou, X., Sun, L., & Xia, Y. (2025). *State-of-art review on load identification and response reconstruction to realize digital twin of infrastructures.* SAGE Publications Inc. doi: 10.1177/13694332251321196

Li, Y., Wang, X., Xia, Y., & Sun, L. (2023, 6). Sparse bayesian technique for load identification and full response reconstruction. *Journal of Sound and Vibration*, *553*. doi: 10.1016/j.jsv.2023.117669

Liu, Y., & Wang, L. (2023, 1). A two-step weighting regularization method for stochastic excitation identification under multi-source uncertainties based on response superposition-decomposition principle. *Mechanical Systems and Signal Processing*, *182*. doi: 10.1016/j.ymssp.2022.109565

Lourens, E., Reynders, E., Roeck, G. D., Degrande, G., & Lombaert, G. (2011, 10). An augmented kalman filter for force identification in structural dynamics. *Mechanical Systems and Signal Processing*, *27*, 446-460.

Ni, P., Li, Q., Han, Q., Xu, K., & Du, X. (2023, 1). Substructure approach for bayesian probabilistic model updating using response reconstruction technique. *Mechanical Systems*



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

*and Signal Processing*, *183*. doi: 10.1016/j.ymssp.2022.109624

Papadimitriou, C., Fritzen, C. P., Kraemer, P., & Ntotsios, E. (2011, 8). Fatigue predictions in entire body of metallic structures from a limited number of vibration sensors using kalman filtering. *Structural Control and Health Monitoring*, *18*, 554-573. doi: 10.1002/stc.395

Papadimitriou, C., & Papadioti, D. C. (2013). Component mode synthesis techniques for finite element model updating. *Computers and Structures*, *126*, 15-28. doi: 10.1016/j.compstruc.2012.10.018

Peeters, B., Hendricx, W., & Debille, J. (2009). *Modern solutions for ground vibration testing of large aircraft* (Tech. Rep.). Retrieved from `www.SandV.com`

Pidaparti, R. M. (2006). *Aircraft structural integrity assessment through computational intelligence techniques* (Vol. 2; Tech. Rep.). Retrieved from `www.airdisaster.com/photos`

Rosafalco, L., Manzoni, A., Mariani, S., & Corigliano, A. (2021, 6). An autoencoder-based deep learning approach for load identification in structural dynamics. *Sensors*, *21*. doi: 10.3390/s21124207

Sarego, G., Zaccariotto, M., & Galvanetto, U. (2017). Artificial neural networks for impact force reconstruction on composite plates. In *4th ieee international workshop on metrology for aerospace : proceedings : Padua, italy, june 21-23, 2017*. IEEE.

Seventekidis, P., & Giagopoulos, D. (2021, 8). A combined finite element and hierarchical deep learning approach for structural health monitoring: Test on a pin-joint composite truss structure. *Mechanical Systems and Signal Processing*, *157*. doi: 10.1016/j.ymssp.2021.107735

Seventekidis, P., & Giagopoulos, D. (2022, 9). Model-based damage identification with simulated transmittance deviations and deep learning classification. *Structural Health Monitoring*, *21*, 2206-2230. doi: 10.1177/14759217211054348

Seventekidis, P., & Giagopoulos, D. (2023, 2). Model error effects in supervised damage identification of structures with numerically trained classifiers. *Mechanical Systems and Signal Processing*, *184*. doi: 10.1016/j.ymssp.2022.109741

Seventekidis, P., Giagopoulos, D., Arailopoulos, A., & Markogiannaki, O. (2020, 11). Structural health monitoring using deep learning with optimal finite element model generated data. *Mechanical Systems and Signal Processing*, *145*. doi: 10.1016/j.ymssp.2020.106972

Wan, Z., Li, S., Huang, Q., & Wang, T. (2014, 1). Structural response reconstruction based on the modal superposition method in the presence of closely spaced modes. *Mechanical Systems and Signal Processing*, *42*, 14-30. doi: 10.1016/j.ymssp.2013.07.007

Wang, L., Peng, Y., Xie, Y., Chen, B., & Du, Y. (2021, 7). A new iteration regularization method for dynamic load identification of stochastic structures. *Mechanical Systems and Signal Processing*, *156*. doi: 10.1016/j.ymssp.2020.107586

Wanhill, R. J. H., Molent, L., Barter, S. A., & Amsterdam, E. (2015). *Nationaal lucht-en ruimtevaartlaboratorium milestone case histories in aircraft structural integrity* (Tech. Rep.). Retrieved from `www.nlr.nl`

Wells, H. J., & King, T. T. (1970, 5). *Air force aircraft structure integrity program: Airplane requirements* (Tech. Rep.). USAF.

Yang, J., Fu, Z., Zou, Y., He, X., Wei, X., & Wang, T. (2023, 2). A response reconstruction



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

method based on empirical mode decomposition and modal synthesis method. *Mechanical Systems and Signal Processing*, *184*. doi: 10.1016/j.ymssp.2022.109716

Zhang, C. D., & Xu, Y. L. (2016, 11). Structural damage identification via multi-type sensors and response reconstruction. *Structural Health Monitoring*, *15*, 715-729. doi: 10.1177/1475921716659787

Zhang, C. D., & Xu, Y. L. (2017, 8). Structural damage identification via response reconstruction under unknown excitation. *Structural Control and Health Monitoring*, *24*. doi: 10.1002/stc.1953

Zhang, X. H., & Wu, Z. B. (2019, 7). Dual-type structural response reconstruction based on moving-window kalman filter with unknown measurement noise. *Journal of Aerospace Engineering*, *32*. doi: 10.1061/(asce)as.1943-5525.0001016

Zhang, X. H., Zhu, Z., Yuan, G. K., & Zhu, S. (2021, 12). Adaptive mode selection integrating kalman filter for dynamic response reconstruction. *Journal of Sound and Vibration*, *515*. doi: 10.1016/j.jsv.2021.116497

Zhi, L., Li, Q. S., & Fang, M. (2016, 12). Identification of wind loads and estimation of structural responses of super-tall buildings by an inverse method. *Computer-Aided Civil and Infrastructure Engineering*, *31*, 966-982. doi: 10.1111/mice.12241

Zhu, H., Du, Z., & Tang, Y. (2023, 10). Full-field displacement and strain reconstruction for beam structures based on the extended inverse finite element method. *Advances in Structural Engineering*, *26*, 2429-2446. doi: 10.1177/13694332231187437



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Appendix A

# Automated Estimation of $\sigma_p = 10^{\gamma}$ via L-curve Curvature

We consider the trade-off between the fitting error $J_{\text{fit error}}$ and the parameter $\sigma_p$. To construct the L-curve, both axes are scaled logarithmically and normalized to $[0, 1]$. This analysis is based on (Hansen, 2001) book:

$$x = \frac{\log_{10}(J_{\text{fit error}}) - \min \log_{10}(J_{\text{fit error}})}{\max \log_{10}(J_{\text{fit error}}) - \min \log_{10}(J_{\text{fit error}})},$$

$$y = \frac{\log_{10}(\sigma_p) - \min \log_{10}(\sigma_p)}{\max \log_{10}(\sigma_p) - \min \log_{10}(\sigma_p)}.$$

The local derivatives are approximated numerically:

$$x' = \nabla x, \quad y' = \nabla y, \quad x'' = \nabla^2 x, \quad y'' = \nabla^2 y,$$

and the curvature of the parametric curve $(x, y)$ is then given by

$$\kappa = \frac{|x'y'' - y'x''|}{(x'^2 + y'^2)^{3/2}}.$$

Finally, the solution is identified at the point of maximum curvature, often referred to as the corner of the L-curve. $\kappa$ attains its maximum, corresponding to the "corner" of the L-curve.

$$\kappa_{max} \Rightarrow OptimalPoint(J_{fiterror,opt}, \sigma_{p,opt})$$



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

# Appendix B

# Matlab Scripts

In this Appendix, my personal work about scripting the software is recorded. Due to the high number of sub-routines, firstly the main program is presented and thereafter every function.

## B.1 Main Program

The main software is presented here. There are comments across the listed script that elaborate on the critical points.

```
%% ---- Response Reconstruction/Load Identification Software with stiffness optimization ----
% This software computes stiffness ($\theta$) in case that it is unknown.
% Afterwards, I will calculate the figures for the methods as before.
% Seting the position of the springs that need optimization first.
% 1st Method ~ Augmented Kalman Filter
% 2nd Method ~ Formulation 1
% 3rd Method ~ Formulation 2
clear; close all; clc
%% 1. Initialization of terms
% Sensor Installation ~ Manually in this stage of the Diploma Thesis
modalanalysis = true; usualanalysis = true; all_modes = false;
base = false;
parameter_estimation = true;
rre_method = [1 2 3]; % 1 AKF, 2 Form1 & 3Form2
min_method = 2; % 1 for fminunc & 2 for fminsearch
force_applied = [1]; % 1 for harmonic, 2 for triangular, 1 2 for hybrid
forces =2;
if length(min_method) > 1
    error("The size of min_method mustn't be more than 1!")
end
mustBeInRange(min_method,1,2);
ndof = 10;
b=1; % Weighting factor for Formulation_1
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
m = ndof - 7 ; % Modes for modal analysis
if m > ndof
   error('m value must be less or equal to n!');
elseif m < 0
    error("m value must be more than zero!")
end
theta_position = []; % The DOF of springs that need optimization
mustBeInRange(theta_position,1,ndof); % A function from Matlab to
   ensure that we want to optimize a DOF that exists.
optimal_theta = 18.2;
guess = 25;
step = 0.5:0.1:2; % In order to calculate to find multiple values of
   J_fit
s_p_tank_3 = zeros(length(step),1);
theta_old = ones(ndof,1)*optimal_theta;
term_theta = guess * ones(length(theta_position), 1);
J_fit_theta = zeros(length(step), 1);
J_fit_error_theta = zeros(length(step), 1);
s_p_tank_3 = zeros(length(step), 1);
displacement = []; % Sensors
velocity = []; % Sensors
% I want all the DOF for plots:
displ_forplots = 1:(ndof);
veloc_forplots = 1:(ndof);
strain = []; % Sensors
strain_forplots = 1:(ndof);
acceleration = [2]; % Sensors
R_position = [];
R = [];
% Empty matrices creation ~ in case of fewer methods
[p_est1, p_est2,p_est3,p_est1m,p_est2m,p_est3m, x_est1, x_est2, x_est3
   , x_est1m, x_est2m, x_est3m,displ_1,displ_2,displ_3,veloc_1,
   veloc_2,veloc_3] = empty_outputs();
if base ~= true
    R_position = [2]; % Where the forces are applied
    n_p = size(R_position,2); % Number of excitations in this system
    R = zeros(ndof,size(R_position,2)); % Where the excitations apply,
        every collumn for a DoF
    % Check that R_position has values
    if size(R_position,2) > 0
        R(sub2ind(size(R), R_position, 1:size(R_position,2))) = 1;
    else
        error("Adjust the variable R_position. We don't have
           excitation applied!")
    end
else
    n_p = 1; % applied only at the base
end
% Qp_exp lower (lb) and upper (ub) bounds in order to run L-curve
Qpexp.lb=-15;
Qpexp.ub=5;
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
% Select from L-curve
run_only_Lcurve = 1; % = 1 to run L-curve and select Qp_value from
    figure;
gamma = 20:-1:-5; % The value of $\gamma$ in the Qp^$\gamma$ term.
s_p = 10.^gamma; % The coefficient of Qp matrix
s_n = 0.02; % The coefficient of Qn matrix
a_x = 1e-8; % The coefficient of Qx matrix
Ap = 1; % The coefficient of the Ap matrix
initial_conditions = [0.0; 0.0];
% L matrices calculation ~ Number of sensors calculation
[n_sensors, L_disp, L_veloc, L_acc, L_str] = L_matrices(displacement,
    velocity, strain, acceleration,ndof);
rng(42); % Set the seed for reproducibility
mu = 0;  % Mean
sigma = 1; % Standard deviation
ksi = normrnd(0, 1, ndof, 1); % Stiffness coefficient
itta = normrnd(0, 1, ndof, 1); % Mass coefficient
% Amount error preallocation
e_1 = 0.0; % Measurement error coeffecient
e_2 = 0.0; e_3 = e_2; % Model error coeffecient
A_p = Ap*eye(n_p); % I must choose this value so that abs(eigenvalues)
    < 1!
Qp_1 = 1e-1; % 1st method guess
Qp_3 = 1e-14; % 3rd method guess
manual_sp3 = 1e15;
L_e = eye(ndof) + diag(-ones(ndof-1, 1), -1);
%% 2. Excitation/Load function creation
t_end = 5; % Total time of the experiment
dt = 0.01; % Time Step
Ndata = t_end./dt; % Number of time instances
t = (0:Ndata)' *dt; % Time vector
fk = zeros(forces,1+Ndata); % Force vector initialization
% 1st Excitation: simple free oscillation
k = (0:Ndata)'; % Column vector of k values
T = 1/1.5; % Harmonic Excitation Period
fk(1,:) = sin((2*pi/T) * k * dt);
% 2nd Excitation: Triangular Function Excitation
t_vector = 0:0.01:0.1; % Define time vector with step dt
fk(2,1:length(t_vector)) = fk(2,1:length(t_vector)) + max(1 - abs(
    t_vector - 0.05) / 0.05, 0); % Triangle with peak at 1
fk(2,Ndata/2:Ndata/2 +length(t_vector)-1) = fk(2,Ndata/2:Ndata/2 +
    length(t_vector)-1) + max(1 - abs(t_vector - 0.05) / 0.05, 0); %
    Triangle with peak at 1
%% 3. External Disturbance Calculation
if base % acceleration = disturbance
        excitation = fk(2,:);
else % force = disturbance
        universal_excitation = R*fk(force_applied,:);
% Filter the rows that are non-zero:
        excitation = universal_excitation(R_position, :);  % Filtering
            the rows with values other than 0.
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
end
%% 4. K & M creation ~ !!! The perturbed model will be used only to
   generate perturbed values!!! Afterwards, only the nominal!!
% In this software strings are unknown, so I will iteratively compute
   the optimum values of $\theta$.
stiffness = theta_old; % Stiffness values (N/m) % stand also as
   theta_optimum
mass = ones(ndof,1); % Mass values (in Kg)
[stiffness_error, mass_error] = model_error_calc_0(mass, stiffness,
   e_2, e_3, ksi, itta); % Model error Calculation
%% 5. Simulated Data Generation Section ~ Once for optimal values of $
   \theta$.
[K,M] = buildMatrices(mass, stiffness, ndof);
[K_error, M_error] = buildMatrices(mass_error, stiffness_error, ndof);
    % With model error
[V, Lambda] = eig(K,M);
[V_error, Lambda_error] = eig(K_error,M_error); % With model error
if base
    [A, B, C_2, D_2] = A_B_C_D_base(ndof, n_p, L_str, L_veloc, L_acc,
       L_disp, L_e, n_sensors, K, M, V, Lambda, dt);
    [A_error, B_error, C_2_error, D_2_error] = A_B_C_D_base(ndof, n_p,
        L_str, L_veloc, L_acc, L_disp, L_e, n_sensors, K_error,
       M_error, V_error, Lambda_error, dt);
else
    [A, B, C_2, D_2] = A_B_C_D_Matrices(ndof, n_p, L_str, L_veloc,
       L_acc, L_disp, L_e, n_sensors, K, M, R, V, Lambda,dt);
    [A_error, B_error, C_2_error, D_2_error] = A_B_C_D_Matrices(ndof,
       n_p, L_str, L_veloc, L_acc, L_disp, L_e, n_sensors, K_error,
       M_error, R, V_error, Lambda_error,dt);
end
sys2 = ss(A, B, C_2, D_2, dt);
yk2 = lsim(sys2, excitation, t); % simulated without model error
yk2 = transpose(yk2(2:end, :)); % simulated for 18.2 k value.
% Generation data with calculation & model errors:
sys2_error = ss(A_error,B_error,C_2_error,D_2_error,dt);
yk2_error = lsim(sys2_error, excitation, t); % simulated with model
   error applied
[y_error_2] = error_calc_0(yk2_error, n_sensors, Ndata, e_1);
yk2_error = y_error_2;
% Universal Data for comparisons in the figure section:
[yk_forplots] = universal_data(ndof,n_p,L_e,K,M,R,V, Lambda, dt,t,
   excitation,base);
%% 6. Input-Parameter Estimation (if needed)
if ~isempty(theta_position)
    %% 6.1 Manually create a plot for varying J_fit
    for i = 1:length(step)
        theta_sub = step(i)*theta_old(theta_position);
        stiffness(theta_position) = theta_sub;
        [K, M] = buildMatrices(mass, stiffness, ndof);
        [V, Lambda] = eig(K,M);
        F = sqrt(diag(Lambda))/2*pi;
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
150         if base
151             [A, B, C_2, D_2] = A_B_C_D_base(ndof, n_p, L_str, L_veloc,
                    L_acc, L_disp, L_e, n_sensors, K, M, V, Lambda, dt);
152         else
153             [A, B, C_2, D_2] = A_B_C_D_Matrices(ndof, n_p, L_str,
                   L_veloc, L_acc, L_disp, L_e, n_sensors, K, M, R, V,
                   Lambda,dt);
154         end
155     % Initial Condition matrices: Ax0 + BF0
156         Fx_0 = initial_conditions(2)*ones(n_p,1);
157         [Q_n, Q_x, x_0, Px_0] = modal_nomodal(ndof, a_x,
               initial_conditions, n_p, s_n, yk2, Ndata);
158         [Q_n_error] = modal_nomodal(ndof, a_x, initial_conditions, n_p
               , s_n, yk2_error, Ndata);
159
160     % Define Qe and augmented matrices:
161         Qp_3 = Qp_3 * eye(n_p);
162         Q_e_local = [Q_x(1:2*ndof, 1:2*ndof), zeros(2*ndof, n_p);
               zeros(n_p, 2*ndof), Qp_3];
163
164     % Run Formulation 2 (Third method):
165         [x_est_local, p_est_local] = third_method(A, C_2, B, D_2, Q_x,
                Q_n, Qp_3, A_p, ndof, Ndata, n_p, yk2, x_0, Fx_0);
166         z_3_local = [x_est_local; p_est_local];
167         [x_est_local_error, p_est_local_error] = third_method(A, C_2,
               B, D_2, Q_x, Q_n, Qp_3, A_p, ndof, Ndata, n_p, yk2_error,
               x_0, Fx_0);
168         z_3_local_error = [x_est_local_error; p_est_local_error];
169
170     % Combine C and D
171         C_3 = [C_2 D_2];
172         A_3 = [A B; zeros(n_p, 2*ndof), A_p];
173
174     % J_fit diagrams illustration to find the optimal value of Q_p
175         [J_fit] = J_fit_calc(s_p, n_p, A, C_2, B, D_2, Q_x, Q_n, A_p,
               ndof, Ndata, yk2, x_0, Fx_0);
176         [J_fit_error] = J_fit_calc(s_p, n_p, A, C_2, B, D_2, Q_x,
               Q_n_error, A_p, ndof, Ndata, yk2_error, x_0, Fx_0);
177     % Find optimal s_p from L-curve:
178         [sp_opt3, Jfit_opt] = L_curve(J_fit_error, J_fit, s_p,step(i),
                theta_old(1),e_1,e_2,theta_position);
179         s_p_tank_3(i,1) = sp_opt3;
180
181     % Compute cost for every step value:
182         J_fit_theta(i) = 0;
183         J_fit_error_theta(i) = 0;
184         e_y1 = yk2(:,1) - C_3 * z_3_local(:,1);
185         e_z1 = z_3_local(:,1) - A_3 * [x_0; Fx_0];
186         e_y1_error = yk2_error(:,1) - C_3 * z_3_local_error(:,1);
187         e_z1_error = z_3_local_error(:,1) - A_3 * [x_0; Fx_0];
188
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
189         J_fit_theta(i) = J_fit_theta(i) + e_y1' * Q_n * e_y1 + e_z1' *
                Q_e_local * e_z1;
190         J_fit_error_theta(i) = J_fit_error_theta(i) + e_y1_error' *
               Q_n_error * e_y1_error + e_z1_error' * Q_e_local *
               e_z1_error;
191
192         for j = 2:Ndata
193             e_y = yk2(:,j) - C_3 * z_3_local(:,j);
194             e_y_error = yk2_error(:,j) - C_3 * z_3_local_error(:,j);
195             e_z = z_3_local(:,j) - A_3 * z_3_local(:,j-1);
196             e_z_error = z_3_local_error(:,j) - A_3 * z_3_local_error
                   (:,j-1);
197             J_fit_theta(i) = J_fit_theta(i) + e_y' * Q_n * e_y + e_z'
                   * Q_e_local * e_z;
198             J_fit_error_theta(i) = J_fit_error_theta(i) + e_y_error' *
                    Q_n_error * e_y_error + e_z_error' * Q_e_local *
                   e_z_error;
199         end
200     end
201     figure_counter = i +1;
202 % Plot for J_fit:
203     figure(figure_counter) % Without Modal Analysis
204     plot(step*optimal_theta,J_fit_theta,"b-",step*optimal_theta,
           J_fit_error_theta,"r-");
205     hold on;
206     xline(optimal_theta, '--k', 'DisplayName', sprintf('$\\theta^* = %.1f$', optimal_theta), ...
207       'Interpreter','latex', 'LineWidth', 1.5);
208     title("Jfit values with respect to $\theta$ values", 'Interpreter', 'latex' )
209     legend("Jfit without errors","Jfit with errors", 'optimal $\theta$ value', 'Interpreter', 'latex')
210     ylabel('J fit');
211     xlabel('($\theta$)/Spring Values');
212     grid on
213     addErrorBox(e_1,e_2);
214     hold off;
215 % Theta Matrices Initialization:
216     theta_sub = term_theta;
217     stiffness = 18.2.* ones(ndof,1);
218     disp("Theta sub values to begin with:");
219     disp(theta_sub);
220 %% 6.2. Iteration to find optimum value for springs
221 % Step 1: Cost function
222     cost_fun_error = @(theta_sub) compute_cost_J_fit2(theta_position,
           theta_sub, yk2_error,initial_conditions,a_x,s_n,Ndata,Qp_3,
           ...
223     mass_error,stiffness_error,ndof,base,n_p,L_str,L_veloc,L_acc,
           L_disp,L_e,n_sensors,R,A_p,dt);
224 % Step 2: Use fminunc or/and fminsearch to minimize with respect to
    theta
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
225     if min_method == 1
226         options = optimoptions('fminunc','Display', 'iter', ...
227         'Algorithm', 'quasi-newton','OptimalityTolerance', 1e-10, ...
228         'StepTolerance', 1e-10,'FunctionTolerance', 1e-10,'OutputFcn',
               @outfun);
229         [theta_new_sub_error, fval_error] = fminunc(cost_fun_error,
               term_theta, options);
230     else
231         options = optimset('Display', 'iter','TolX', 1e-10,'TolFun', 1
               e-10, 'OutputFcn',@outfun);
232         [theta_new_sub_error, fval_error] = fminsearch(cost_fun_error,
                term_theta, options);
233     end
234 % After parameter estimation, plot iteration history
235     figure_counter = figure_counter +1;
236     figure(figure_counter);
237     plot(theta_history,'LineWidth',1.5)
238     hold on;
239 % yline first so it can appear in the legend
240     yline(optimal_theta, '--k', 'DisplayName', sprintf('$\\theta^* = 
           %.1f$', optimal_theta), ...
241       'Interpreter','latex', 'LineWidth', 1.5);
242 % Labels, title, grid
243     xlabel('Iteration');
244     ylabel('\theta ($\theta$) values','Interpreter','latex');
245     grid on;
246     title('Convergence of \theta ($\theta$) during optimization','
           Interpreter','latex');
247 % Add error boxes
248     addErrorBox(e_1,e_2);
249 % Legend (automatically includes yline via DisplayName)
250     legend_entries = arrayfun(@(val) sprintf('$\\theta = %d$th DOF',
           val), theta_position,'UniformOutput', false);
251     legend_entries{end+1} = 'Optimal $\theta^*$ value';  % optional
           extra label for clarity
252     legend(legend_entries, 'Interpreter','latex');
253     hold off;
254     figure_counter = figure_counter +1;
255 %% 6.3 Display the optimum stiffness Values:
256     stiffness_error(theta_position) = theta_new_sub_error;
257     disp("The estimated optimal spring values with errors are:")
258     disp(stiffness_error);
259     theta_rel_error = 100*(theta_new_sub_error - stiffness(
           theta_position))./optimal_theta;
260     disp("The relative error between optimal and estimated theta
           parameters:")
261     disp(theta_rel_error)
262     pause;
263 else
264     % Initial Condition matrices: Ax0 + BF0
265     Fx_0 = initial_conditions(2)*ones(n_p,1);
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
266     [Q_n, Q_x, x_0, Px_0] = modal_nomodal(ndof, a_x,
            initial_conditions, n_p, s_n, yk2, Ndata);
267     [Q_n_error] = modal_nomodal(ndof, a_x, initial_conditions, n_p,
            s_n, yk2_error, Ndata);
268 end
269     %% 7. Testing the fit between optimal theta and computational
            theta.
270 [K_error, M] = buildMatrices(mass, stiffness_error, ndof);
271 [V_error, Lambda_error] = eig(K_error,M);
272 F_error = sqrt(diag(Lambda_error))/2*pi;
273 F = sqrt(diag(Lambda))/2*pi;
274 % Percentage Modal Frequency relative error:
275 F_rel_error = 100*((F - F_error)./F_error);
276 disp("The relative error between optimal and estimated modal
        frequencies:")
277 disp(F_rel_error);
278 if base
279     [A, B, C_2, D_2] = A_B_C_D_base(ndof, n_p, L_str, L_veloc, L_acc,
            L_disp, L_e, n_sensors, K_error, M, V_error, Lambda_error, dt)
            ;
280 else
281     [A, B, C_2, D_2] = A_B_C_D_Matrices(ndof, n_p, L_str, L_veloc,
            L_acc, L_disp, L_e, n_sensors, K_error, M, R, V_error,
            Lambda_error,dt);
282 end
283 %% 8. Methods without Modal Analysis
284 if ismember(1,rre_method)
285     if run_only_Lcurve == 1
286         AKF_L_curve_Tetepoulidis;
287     end
288     disp('Paused␣after␣L-curve.␣Adjust␣Qp_1␣in␣the␣beginning,␣if␣
            needed.');
289     input('Press␣Enter␣to␣continue...', 's');  % Waits for Enter key
290     Qp_1 = Qp_1*eye(n_p);
291     [p_est1,x_est1,Px_0_1, norm_Pxs_0_1, norm_Kxs_0_1] = first_method(
            A,B,C_2,D_2,A_p,Q_x,Qp_1,Q_n_error,x_0, Fx_0, Px_0,yk2_error,
            Ndata);
292
293 end
294 if ismember(2,rre_method)
295     [x_est2, p_est2] = second_method(A, C_2, B, D_2, b, ndof, Ndata,
            n_p, yk2_error, x_0, Fx_0); % both errors
296 end
297 if ismember(3, rre_method)
298     [J_fit] = J_fit_calc(s_p, n_p, A, C_2, B, D_2, Q_x, Q_n, A_p, ndof
            , Ndata, yk2, x_0, Fx_0);
299     [J_fit_error] = J_fit_calc(s_p, n_p, A, C_2, B, D_2, Q_x,
            Q_n_error, A_p, ndof, Ndata, yk2_error, x_0, Fx_0);
300     % Find optimal s_p from L-curve:
301     [sp_opt3, Jfit_opt] = L_curve(J_fit_error, J_fit, s_p, stiffness(
            theta_position)/theta_old(1), theta_old(1),e_1,e_2,
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
        theta_position);
302     if ~isempty(theta_position)
303         %Statistical Optimal Value
304         % Find the mode (most common value) in s_p_tank
305         % Unique values
306         [uvals, ~, idx] = unique(s_p_tank_3);
307         % Count occurrences
308         counts = histcounts(idx, 1:numel(uvals)+1);
309         % Find maximum count
310         maxCount = max(counts);
311         % Values that appear maxCount times
312         candidates = uvals(counts == maxCount);
313         % Choose highest
314         sp_stat_opt3 = max(candidates);
315         if sp_opt3 == manual_sp3
316             sp_stat_opt3 = sp_opt3;
317             disp("Mathematical Automatization was successful!")
318             disp(sp_stat_opt3);
319         else
320             disp("Mathematical Automatization failed!")
321             if sp_stat_opt3 == manual_sp3
322                 disp('Statistical␣Aytomatization␣was␣successful!␣Most␣
                        common␣optimal␣s_p␣value␣across␣steps␣(highest␣if␣
                        tie):');
323                 disp(sp_stat_opt3);
324             else
325                 sp_stat_opt3 = manual_sp3;
326                 disp('Needed␣manual␣optimal␣s_p␣value␣selection:');
327                 disp(sp_stat_opt3);
328             end
329         end
330     else
331        if sp_opt3 == manual_sp3
332             sp_stat_opt3 = sp_opt3;
333             disp("Mathematical Automatization was successful!")
334             disp(sp_stat_opt3);
335        else
336             sp_stat_opt3 = manual_sp3;
337             disp('Needed␣manual␣optimal␣s_p␣value␣selection:');
338             disp(sp_stat_opt3);
339        end
340     end
341     Qp_3 = (1/sp_stat_opt3)*eye(n_p);
342     [x_est3, p_est3] = third_method(A, C_2, B, D_2, Q_x, Q_n_error,
           Qp_3, A_p, ndof, Ndata, n_p, yk2_error, x_0, Fx_0); % both
           errors
343 end
344 %% 9. Methods with modal analysis
345 if modalanalysis
346   if base
347     [Am,Bm,Cm_2,Dm_2] = A_B_C_D_Modal_base(ndof, m, n_p, L_str,
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
        L_veloc, L_acc, L_disp, L_e, n_sensors, M, V_error,
        Lambda_error, dt);
348   else
349     [Am,Bm,Cm_2,Dm_2] = A_B_C_D_Modal(m, n_p, L_str, L_veloc, L_acc,
        L_disp, L_e, n_sensors, R, V_error, Lambda_error, dt);
350   end
351   sys2m_error = ss(Am,Bm,Cm_2,Dm_2,dt);
352   yk2m_error = lsim(sys2m_error, excitation, t); % simulated with
        model error applied
353   % Calculation of measurement errors:
354   [y_error_2] = error_calc_0(yk2m_error, n_sensors, Ndata, e_1);
355   yk2m_error = y_error_2;
356   [Q_nm, Q_xm, x_0m, Px_0m] = modal_nomodal(m, a_x, initial_conditions
        , n_p, s_n, yk2m_error, Ndata);
357   [Q_nm_error] = modal_nomodal(m, a_x, initial_conditions, n_p, s_n,
        yk2m_error, Ndata);
358   % 1st method - Augmented Kalman Filters
359   [p_est1m,x_est1m,Px_1, norm_Pxs_1, norm_Kxs_1] = first_method(Am,Bm,
        Cm_2,Dm_2,A_p,Q_xm,Qp_1,Q_nm,x_0m, Fx_0, Px_0m,yk2m_error,Ndata)
        ;
360   displ_1 = V_error(1:ndof,1:m)*x_est1m(1:m,:); % in order to
        calculate the displacements, not $\ksi$
361   veloc_1 = V_error(1:ndof,1:m)*x_est1m(1+m:2*m,:); % in order to
        calculate the accelerations, not $\ksi \ddot$
362   % 2nd Method - 1st Proposed Formulation
363   [x_est2m, p_est2m] = second_method(Am, Cm_2, Bm, Dm_2, b, m, Ndata,
        n_p, yk2m_error, x_0m, Fx_0); % both errors
364   displ_2 = V_error(:,1:m)*x_est2m(1:m,:); % in order to calculate the
         displacements, not $\ksi$
365   veloc_2 = V_error(1:ndof,1:m)*x_est2m(1+m:2*m,:); % in order to
        calculate the accelerations, not $\ksi \ddot$
366   % 3rd Method - 2nd Proposed Formulation
367   [x_est3m, p_est3m] = third_method(Am, Cm_2, Bm, Dm_2, Q_xm,
        Q_nm_error, Qp_3, A_p, m, Ndata, n_p, yk2m_error, x_0m, Fx_0); %
         both errors
368   displ_3 = V_error(1:ndof,1:m)*x_est3m(1:m,:); % in order to
        calculate the displacements, not $\ksi$
369   veloc_3 = V_error(1:ndof,1:m)*x_est3m(1+m:2*m,:); % in order to
        calculate the accelerations, not $\ksi \ddot$
370 end
371 pause
372 %% 12.Figure Section ~ Method Comparisons
373 % Excitation Figures
374 p_figures(p_est1,p_est2,p_est3,p_est1m,p_est2m,p_est3m,excitation,n_p,
    t,Fx_0,base,R_position,rre_method,modalanalysis,usualanalysis,e_1,
    e_2);
375 if ~isempty(displ_forplots)
376 % Displacement Figures
377     displacement_figures(x_est1,x_est2,x_est3,displ_1,displ_2,displ_3,
        yk_forplots,t, x_0, Fx_0, displ_forplots,ndof,rre_method,
        modalanalysis,usualanalysis,e_1,e_2);
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
end
% Velocity Figures
if ~isempty(veloc_forplots)
    velocity_figures(x_est1,x_est2,x_est3,veloc_1,veloc_2,veloc_3,
        yk_forplots,t, x_0, Fx_0,veloc_forplots,ndof,rre_method,
        modalanalysis,usualanalysis,e_1,e_2);
end
% Strain Figures
if ~isempty(strain_forplots)
    strain_figures(x_est1,x_est2,x_est3,displ_1,displ_2,displ_3,
        yk_forplots,t, x_0, Fx_0, strain_forplots,ndof,rre_method,
        modalanalysis,usualanalysis,e_1,e_2);
end
```

Listing B.1: **Main Program Script**

## B.2 Functions

This section contains all of the sub-routines created to support the main program. It is divided in six subsections, each one of them containing the respective functions.



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

### B.2.1 Term Initialization

This subsection includes all the preliminary functions needed for the main software running. Most of them doesn't pose any excelsior scientific value. They are considered to be very common, such Stiffness, Mass creation, eigenvalues calculation and Boolean Matrices creation needed for the next subsection. Moreover, there are several functions scripted mainly to overcome practical issues in the software and to settle it less time-consuming.

Firstly, function B.2 is scripted to just create empty matrices. B.3 is created just for annotated boxes to record the errors inside the figures:

```
function [p_est1, p_est2,p_est3,p_est1m,p_est2m,p_est3m, x_est1,
    x_est2, x_est3, x_est1m, x_est2m, x_est3m,displ_1,displ_2,displ_3,
    veloc_1,veloc_2,veloc_3] = empty_outputs()
    % Function to initialize the variables
    % x/p_est1(m), displ_1, veloc_1: First Method (Modal Analysis)
    % x/p_est2(m), displ_2, veloc_2: Second Method (Modal Analysis)
    % x/p_est3(m), displ_3, veloc_3: Third Method (Modal Analysis)
    p_est1 = []; p_est2 = []; p_est3 = [];
    p_est1m = []; p_est2m = []; p_est3m = [];
    x_est1 = []; x_est2 = []; x_est3 = [];
    x_est1m = []; x_est2m = []; x_est3m = [];
    displ_1 = []; displ_2 = []; displ_3 = [];
    veloc_1 = []; veloc_2 = []; veloc_3 = [];
end
```

Listing B.2: **Empty Matrices Initialization**

```
function h = addErrorBox(e1, e2)
    % addErrorBox  Add a box with error info in top-left corner of
        current axes
    % h = addErrorBox(e1, e2)
    % Inputs:
    % e1  - measurement error
    % e2  - model error
    % Output:
    % h   - handle to the text object

    % Build the string
    str = sprintf('Measurement␣Error␣=␣%.2f\nModel␣Error␣=␣%.2f', e1,
        e2);

    % Current axis limits
    xl = xlim;
    yl = ylim;

    % Add some padding relative to axis range
    dx = 0.02 * range(xl);   % 2% of width
    dy = 0.02 * range(yl);   % 2% of height

    h = text(xl(1)+dx, yl(2)-dy, str, ...
    'VerticalAlignment','top', ...
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
    'HorizontalAlignment','left', ...
    'BackgroundColor','w', ...
    'EdgeColor','k', ...
    'Margin',5, ...
    'FontSize',10);
end
```

Listing B.3: **Annotated Box Function**

Function B.4 contributes to creating simulated data for the figure section of the software:

```
function [yk] = universal_data(ndof,n_p,L_e,K,M,R,V, Lambda, dt,t,
    excitation,base)
    % This function creates simulated data for every DOF & each
       category
    % 1. Displacement
    % 2. Velocity
    % 3. Strain
    % 4. Acceleration
    displacement = 1:ndof;
    velocity = 1:ndof;
    strain = 1:ndof;
    acceleration = 1:ndof;
    [n_sensors, L_disp, L_veloc, L_acc, L_str] = L_matrices(
       displacement, velocity, strain, acceleration,ndof);
    % I expect that L_disp etc. matrices to be eye(ndof)
    if base == true
    % Base excitation matrices
        [A, B, C, D] = A_B_C_D_base(ndof, n_p, L_str, L_veloc, L_acc,
           L_disp, L_e, n_sensors, K, M, V, Lambda, dt);
    else
    % Normal excitation matrices
        [A, B, C, D] = A_B_C_D_Matrices(ndof, n_p, L_str, L_veloc,
           L_acc, L_disp, L_e, n_sensors, K, M, R, V, Lambda,dt);
    end
    sys = ss(A, B, C, D, dt);
    yk = lsim(sys, excitation, t);
    yk = transpose(yk);
end
```

Listing B.4: **Simulated Data Function**

Listings B.5, B.6, B.7 & B.8 are meant for initialization of necessary terms. More specifically:

System Eigenvalue Calculation:

```
function [abs_eigen] = abs_eigen_calc(method, ndof, n_p, A, B, C, D, b
    , s_p, Q_x, Q_n, A_p)
    if method == 2 % 1st formulation ~ 2nd Method
        gamma = [eye(2*ndof) + b*(C')*C b*(C')*D; b*(D')*C b*(D')*D];
        delta = zeros(n_p + 2*ndof);
        delta(1:2*ndof, 1:2*ndof + n_p) = [A B];
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
        [a, Lambda] = eig(delta, gamma);
        % Extract eigenvalues
        eigenvalues = diag(Lambda);
        % Check if any absolute eigenvalue exceeds 1
        if any(abs(eigenvalues) >= 1)
            warning('Eigenvalue(s) with abs >= 1 detected.');
        else
            disp('Everything is OK in the 2nd Method!');
        end
        abs_eigen = abs(eigenvalues);
    elseif method == 3 % 2nd formulation ~ 3rd Method
        abs_eigen = zeros(n_p + 2*ndof,length(s_p));
        for i=1:length(s_p) % I will check itiratively the absolute
            values.
            Q_p = (1/s_p(i))*eye(n_p); % load process covariance
            gamma =  [(Q_x + (C')*Q_n*C) (C')*Q_n*D; ((D')*Q_n*C) (Q_p
                + (D')*Q_n*D)];
            delta = [Q_x*A Q_x*B; zeros(n_p,2*ndof) Q_p*A_p];
            [a, Lambda] = eig(delta, gamma);
        % Extract eigenvalues
            eigenvalues = diag(Lambda);
            abs_eigen(:,i) = abs(eigenvalues);
        % Check if any absolute eigenvalue exceeds 1
            if any(abs(eigenvalues) >= 1)
                warning('Eigenvalue(s) with abs >= 1 detected at
                    iteration %d. Adjust the term of A_p.', i);
            else
                fprintf('Everything is OK in the 3rd Method at
                    iteration %d.', i);
            end
        end
    end
end
```

Listing B.5: **System Eigenvalue Calculation Function**

Stiffness and Mass matrices build-in:

```
function [K, M] = buildMatrices(mass, stiffness, n)
    % Initialize stiffness and mass matrices
    K = zeros(n); % Stiffness matrix
    M = zeros(n); % Mass matrix

    % Validate inputs
    if length(mass) ~= n || length(stiffness) ~= n
        error('Length of mass and stiffness arrays must match n.');
    end

    % Populate stiffness matrix K (tridiagonal structure)
    for i = 1:n
        if i > 1
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
            K(i, i-1) = -stiffness(i); % Off-diagonal (below main
                diagonal)
        end
        if i < n
            K(i, i) = stiffness(i) + stiffness(i+1); % Main diagonal
            K(i, i+1) = -stiffness(i+1); % Off-diagonal (above main
                diagonal)
        else
            K(i,i) = stiffness(i);
        end
    end

    % Populate mass matrix M (diagonal matrix with mass values)
    for i = 1:n
        M(i, i) = mass(i); % Mass values on the diagonal
    end
end
```

Listing B.6: **Stiffness and Mass Matrices Build-in Function**

Boolean Matrices for sensor placement creation:

```
function [n_sensors, L_disp, L_veloc, L_acc, L_str] = L_matrices(
    displacement, velocity, strain, acceleration,ndof)
    % Number of sensors of each category
    n_disp = size(displacement,2);
    n_acc = size(acceleration,2);
    n_str = size(strain,2);
    n_veloc =size(velocity,2);
    % Total sensors
    n_sensors = n_disp + n_acc + n_str + n_veloc; % The sum of
        respective sensor categories.
    % Initialize the matrices
    L_acc = zeros(n_acc,ndof);
    L_veloc = zeros(n_veloc,ndof);
    L_str = zeros(n_str,ndof);
    L_disp = zeros(n_disp,ndof);
    % Vectorized assignments
    % Acceleration sensors
    if n_acc > 0
        L_acc(sub2ind(size(L_acc), 1:n_acc, acceleration)) = 1;
    end
    % Displacement sensors
    if n_disp > 0
        L_disp(sub2ind(size(L_disp), 1:n_disp, displacement)) = 1;
    end
    % Velocity sensors
    if n_veloc > 0
        L_veloc(sub2ind(size(L_veloc), 1:n_veloc, velocity)) = 1;
    end
    % Strain sensors
    if n_str > 0
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
        L_str(sub2ind(size(L_str), 1:n_str, strain)) = 1;
    end
end
```

Listing B.7: **Boolean Matrices Creation Function**

Finally, Covariance Matrices initialization with and without Modal Analysis:

```
function [Qn, varargout] = modal_nomodal(ndof, a_x, initial_conditions
    , n_p, s_n, yk, Ndata)
    % Qn calculation
    Intensity = sqrt(sum(yk.^2, 2) / Ndata);  % Compute intensity for
       each sensor
    Qn = (s_n^2) * diag(Intensity.^2);       % Apply scalar scaling
       and square
    % Optional Qx
    if nargout > 1
        Qx = a_x .* eye(2*ndof);
        varargout{1} = Qx;
    end
    % Optional x_0 and Px_0
    if nargout > 2
        x_0 = initial_conditions(1) * 0.025 * ones(2*ndof, 1);
        varargout{2} = x_0;
    end
    if nargout > 3
        Px_0 = zeros(2*ndof + n_p, 2*ndof + n_p);
        varargout{3} = Px_0;
    end
end
```

Listing B.8: **Covariance Matrices Initialization Function**



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

### B.2.2 State-Space & Observation Matrices

In this subsection, 4 functions are illustrated. The two of them are associated with base excitation which is an extra option in the software albeit not included in any experiment. More specifically, A, B, C, and D matrices function:

```
function [A, B, C, D] = A_B_C_D_Matrices(ndof, n_p, L_str, L_veloc,
    L_acc, L_disp, L_e, n_sensors, K, M, R, V, Lambda, dt)
    Z = 0.02*eye(ndof); % Damping ratios
    C_star = zeros(ndof);
    C_star = C_star + 2.*sqrt(Lambda).*Z;
    C_tilda = M*V*C_star*V'*M;
% A, B Matrices ~ State Space ~ Without Modal Analysis
    Ac = [zeros(ndof) eye(ndof); -1.*(M\K) -1.*(M\C_tilda)];
    A = expm(Ac.*dt);
    Bc = [zeros(ndof,n_p); (M\R)];
    B = (A - eye(size(A)))*(Ac\Bc);
% C, D Matrices ~ Observation ~ Without Modal Analysis
    C = zeros(n_sensors,2*ndof);
    D = zeros(n_sensors, n_p);
    if size(L_disp,1) > 0 % displacement sensors
        C(1:size(L_disp,1),1:ndof) = L_disp;
        D(1:size(L_disp,1),:) = 0;
    end
    if size(L_veloc,1) > 0 % velocity sensors
        C(1 + size(L_disp,1):size(L_disp,1) + size(L_veloc,1), ndof
           +1:2*ndof) = L_veloc;
        D(1 + size(L_disp,1):size(L_disp,1) + size(L_veloc,1),:) = 0;
    end
    if size(L_str,1) > 0 % strain sensors
        C(1 + size(L_disp,1) + size(L_veloc,1):size(L_disp,1) + size(
           L_veloc,1) + size(L_str,1), 1:ndof) = L_str*L_e;
        D(1 + size(L_disp,1) + size(L_veloc,1):size(L_disp,1) + size(
           L_veloc,1) + size(L_str,1), :) = 0;
    end
    if size(L_acc,1) > 0 % acceleration sensors
        C(1 + size(L_disp,1) + size(L_veloc,1) + size(L_str,1):end, :)
            = L_acc*(-1)*[M\K M\C_tilda];
        D(1 + size(L_disp,1) + size(L_veloc,1) + size(L_str,1):end, :)
            = L_acc*(M\R);
    end
```

Listing B.9: **Function for A,B,C & D Matrices**

Modal Analysis:

```
function [A, B, C, D] = A_B_C_D_Modal(m, n_p, L_str, L_veloc, L_acc,
    L_disp, L_e, n_sensors, R, V, Lambda, dt)
    Z = 0.02*eye(m); % Damping ratios
% A, B Matrices ~ State Space ~ Modal Analysis
    Ac =[zeros(m) eye(m); -1.*Lambda(1:m,1:m) -2.*sqrt(Lambda(1:m,1:m)
       ).*Z];
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
    A = expm(Ac.*dt);
    Bc = [zeros(m,n_p); (V(:,1:m))'*R]; % Modeshapes nxm due to modal
       analysis
    B = (A - eye(size(A)))*(Ac\Bc);
% C, D Matrices ~ Observation ~ Modal Analysis
    C = zeros(n_sensors,2*m);
    D = zeros(n_sensors, n_p);
    if size(L_disp,1) > 0 % displacement sensors
        C(1:size(L_disp,1),1:m) = L_disp*V(:,1:m);
        D(1:size(L_disp,1),:) = 0;
    end
    if size(L_veloc,1) > 0 % velocity sensors
        C(1 + size(L_disp,1):size(L_disp,1) + size(L_veloc,1), m +1:2*
           m) = L_veloc*V(:,1:m);
        D(1 + size(L_disp,1):size(L_disp,1) + size(L_veloc,1),:) = 0;
    end
    if size(L_str,1) > 0 % strain sensors
        C(1 + size(L_disp,1) + size(L_veloc,1):size(L_disp,1) + size(
           L_veloc,1) + size(L_str,1), 1:m) = L_str*L_e*V(:,1:m);
        D(1 + size(L_disp,1) + size(L_veloc,1):size(L_disp,1) + size(
           L_veloc,1) + size(L_str,1), :) = 0;
    end
    if size(L_acc,1) > 0 % acceleration sensors
        C(1 + size(L_disp,1) + size(L_veloc,1) + size(L_str,1):end, :)
            = L_acc*(-1)*V(:,1:m)*[Lambda(1:m,1:m) 2.*sqrt(Lambda(1:m
           ,1:m)).*Z];
        D(1 + size(L_disp,1) + size(L_veloc,1) + size(L_str,1):end, :)
            = (L_acc*V(:,1:m)*transpose(V(:,1:m))*R);
    end
```

Listing B.10: **Function for A,B,C & D Matrices with Modal Analysis**



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

### B.2.3 Error Calculations

These subsection contains two functions, the first regarding Model Error calculation and the second one Measurement Error. Hence, for Model Error:

```
function [stiffness_tel, mass_tel] = model_error_calc_0(mass,
    stiffness, e_2, e_3, ksi, itta)
% Secondly, I will calculate the model error
stiffness_tel = stiffness.*(1 + e_2.*ksi);
mass_tel = mass.*(1 + e_3.*itta);
end
```

Listing B.11: **Model Error Calculation Function**

Measurement Error, respectively:

```
function [y_tel] = error_calc_0(yk, n_sensor, N, e_1)
rng(42);
mu = 0;  % Mean
sigma = 1; % Standard deviation
gaussian = normrnd(0, 1, n_sensor, N); % measurement coefficient
% Pre-allocate the y_intensity matrix
y_intensity = zeros(n_sensor, 1);
% I calculate the expression in advance in order not to get confused
    with the algebra.
expression = (transpose(yk)).^2;
% Calculations of y_intensity matrix
for i=1:n_sensor
    y_intensity(i,1) = y_intensity(i,1) + sqrt((1/N).*(sum(expression(
        i,2:N+1))));
end
% Measurement error calculation procedure:
y_tel = zeros(n_sensor,N);
for i=1:n_sensor
    y_tel(i,:) = yk(2:N+1, i)' + e_1 .* y_intensity(i,1) .* gaussian(i
        ,:);
end
```

Listing B.12: **Measurement Error Calculation Function**



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

### B.2.4 Input Parameter Estimation

In this subsection, the functions considering the optimized $\theta$ parameters along with Covariance Matrices estimation are included.

More specifically, the first three scripts, B.13, B.14 & B.15 are referring to Objective function Jfit.

```
function [J_fit2] = J_fit_calc(s_p, n_p, A, C_2, B, D_2, Q_x, Q_n, A_p
    , ndof, Ndata, yk, x_0, Fx_0)
% This function contributes to the L_curve Figure Creation
% It is based on Chapter 3 Secntion 7
% Helps select the optimal Qp_value
J_fit1 = zeros(length(s_p),1);
J_fit2 = zeros(length(s_p),1);
C_3 = [C_2 D_2]; % C_2 stands for Cx & D_2 stands for Cp
A_3 = [A B; zeros(n_p,2*ndof) A_p];
for i=1:length(s_p) % I will store itiratively the values of J_fit in
    the respective Matrix
    Q_p = (1/s_p(i)).*eye(n_p);
    Q_e = [Q_x(1:2*ndof,1:2*ndof), zeros(2*ndof,n_p); zeros(n_p,2*ndof
        ) Q_p];
    [x_est3, p_est3] = third_method(A, C_2, B, D_2, Q_x, Q_n, Q_p, A_p
        , ndof, Ndata, n_p, yk, x_0, Fx_0);
    z_3 = [x_est3; p_est3];
    % J_fit calculation procedure
    for j=1:Ndata
        J_fit1(i) = J_fit1(i) + transpose(yk(:,j) - C_3*z_3(:,j))*Q_n
            *(yk(:,j) - C_3*z_3(:,j));
    end
    % J_fit2 Calculation ~ First term manually:
    J_fit2(i) = J_fit1(i) + transpose(z_3(:,1) - A_3*[x_0; Fx_0])*Q_e
        *(z_3(:,1) - A_3*[x_0; Fx_0]);
    % The rest itiratively:
    for j = 2:Ndata
        J_fit2(i) = J_fit2(i) + transpose(z_3(:,j) - A_3*z_3(:,j-1))*
            Q_e*(z_3(:,j) - A_3*z_3(:,j-1));
    end
end
```

Listing B.13: **Jfit Calculation Function**

```
function [J_fit2m] = J_fit_modal_calc(s_p, n_p, Am, C_2m, Bm, D_2m,
    Q_xm, Q_nm, A_p, m, Ndata, yk, x_0, Fx_0,V)
% This function contributes to the L_curve Figure Creation ~ Modal
    Analysis
% It is based on Chapter 3 Secntion 7
% Helps select the optimal Qp_value
    J_fit1m = zeros(length(s_p),1);
    J_fit2m = zeros(length(s_p),1);
    C_3 = [C_2m D_2m]; % C_2 stands for Cx & D_2 stands for Cp
    A_3 = [Am Bm; zeros(n_p,2*m) A_p];
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
    for i=1:length(s_p) % I will store itiratively the values of J_fit
        in the respective Matrix
        Q_p = (1/s_p(i)).*eye(n_p);
        Q_e = [Q_xm(1:2*m,1:2*m), zeros(2*m,n_p); zeros(n_p,2*m) Q_p];
        [x_est3, p_est3] = third_method(Am, C_2m, Bm, D_2m, Q_xm, Q_nm
            , Q_p, A_p, m, Ndata, n_p, yk, x_0, Fx_0);
        displ_3 = V(:,1:m)*x_est3(1:m,:); % in order to calculate the
            displacements, not $\ksi$
        veloc_3 = V(:,1:m)*x_est3(m+1:2*m,:);
        z_3 = [displ_3(1:m,:); veloc_3(1:m,:); p_est3];
    end
    % J_fit calculation procedure
    for j=1:Ndata
        J_fit1m(i) = J_fit1m(i) + transpose(yk(:,j) - C_3*z_3(:,j))*
            Q_nm*(yk(:,j) - C_3*z_3(:,j));
    end
    % J_fit2 2nd term calculation ~ First term manually:
    J_fit2m(i) = J_fit1m(i) + transpose(z_3(:,1) - A_3*[x_0; Fx_0])*
       Q_e*(z_3(:,1) - A_3*[x_0; Fx_0]);
    % The rest itiratively:
    for j = 2:Ndata
        J_fit2m(i) = J_fit2m(i) + transpose(z_3(:,j) - A_3*z_3(:,j-1))
            *Q_e*(z_3(:,j) - A_3*z_3(:,j-1));
    end
end
```

Listing B.14: **Jfit Calculation with Modal Analysis Function**

```
function J = compute_cost_J_fit2(theta_position,theta_sub, yk2,
   initial_conditions,a_x,s_n,Ndata,Qp_3, ...
    mass,stiffness,ndof,base,n_p,L_str,L_veloc,L_acc,L_disp,L_e,
       n_sensors,R,A_p,dt)
 % The same as J_fit_calc
 % Only for input parameter estimation procedure
    stiffness(theta_position) = theta_sub;
    [K, M] = buildMatrices(mass, stiffness, ndof);
    [V, Lambda] = eig(K,M);
    if base
        [A, B, C_2, D_2] = A_B_C_D_base(ndof, n_p, L_str, L_veloc,
           L_acc, L_disp, L_e, n_sensors, K, M, V, Lambda, dt);
    else
        [A, B, C_2, D_2] = A_B_C_D_Matrices(ndof, n_p, L_str, L_veloc,
            L_acc, L_disp, L_e, n_sensors, K, M, R, V, Lambda,dt);
    end
    Fx_0 = initial_conditions(2)*ones(n_p,1);
    [Q_n, Q_x, x_0, Px_0] = modal_nomodal(ndof, a_x,
       initial_conditions, n_p, s_n, yk2, Ndata);
    Q_p3 = Qp_3 * eye(n_p);
    Q_e_local = [Q_x(1:2*ndof, 1:2*ndof), zeros(2*ndof, n_p); zeros(
       n_p, 2*ndof), Q_p3];

```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
    % Run third method
    [x_est_local, p_est_local] = third_method(A, C_2, B, D_2, Q_x, Q_n
        , Q_p3, A_p, ndof, Ndata, n_p, yk2, x_0, Fx_0);
    z_3_local = [x_est_local; p_est_local];

    % Combine C and D
    C_3 = [C_2 D_2];
    A_3 = [A B; zeros(n_p, 2*ndof), A_p];

    % Compute cost
    J = 0;
    e_y1 = yk2(:,1) - C_3 * z_3_local(:,1);
    e_z1 = z_3_local(:,1) - A_3 * [x_0; Fx_0];
    J = J + e_y1' * Q_n * e_y1 + e_z1' * Q_e_local * e_z1;

        for j = 2:Ndata
            e_y = yk2(:,j) - C_3 * z_3_local(:,j);
            e_z = z_3_local(:,j) - A_3 * z_3_local(:,j-1);
            J = J + e_y' * Q_n * e_y + e_z' * Q_e_local * e_z;
        end
end
```

Listing B.15: **Jfit Calculation for Input Estimation Function**

Thereafter, functions B.16 & B.17 consider the L_curve creation and finally, B.18 is a function that helps storing the $\theta$ history along with the number of iterations to display how quickly the software converges in the optimized $\theta$.

```
% Run L-curve
clear qQ_p L1 L2 L3 smooth_n_1 smooth_n_2
p_real = excitation;
lb = Qpexp.lb;
ub = Qpexp.ub;
for i = lb:ub

% clear p_est;
% qQp = 10^i * eye(np);
    Qp = 10^i * eye(n_p);
% Q = [Qx Z; Z' qQp];  % cont time
    innovation = zeros(n_sensors,Ndata);
    x_est = zeros(2*ndof+n_p,Ndata);
    Px = zeros(2*ndof+n_p,2*ndof+n_p,Ndata);

    [p_est,x_est,Px] = first_method(A,B,C_2,D_2,A_p,Q_x,Qp,Q_n,x_0,
        Fx_0, Px_0,yk2_error,Ndata);

    innovation = yk2_error-[C_2  D_2]*x_est;

    qQ_p(i-lb+1) = 10^i;
%    p_est = x_est(2*ndof+1:end,:);

```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
    L1(i-lb+1) = norm(innovation,'fro')^2/Ndata;   %error norm (
        normalized)
    L2(i-lb+1) = norm(innovation,'fro')^2;      %error norm (Lourens)
    L3(i-lb+1) = norm(innovation,'fro');        %error norm (Cumbo)

    %smooth_n_1(i-lb+1) =  norm(p_est,2)^2;
    smooth_n_1(i-lb+1) =  norm(p_est,'fro')^2;
    %smooth_n_2(i-lb+1) =  norm(p_est,2)/ norm(p_real,2);
    smooth_n_2(i-lb+1) =  norm(p_est,'fro')^2/ norm(p_real,'fro')^2;
    %norm(p_real,'fro')

end

%Type 1 ---
%Original by Omid
% loglog(L,Q_p);
createfigure(L1, qQ_p);
if run_only_Lcurve ~= 1
    title_str = 'L-curve Type1';
    Script_save_title_figs;
end

%%
%Type 2 ---
%In Lourens et al.
createfigure(L2, smooth_n_2);
txts_v = sprintfc('%.0e',qQ_p);
%text(L2, smooth_n_2, txts_v);
text(L2(1:2:end), smooth_n_2(1:2:end), txts_v(1:2:end));
% ylm = ylim;
% xlm = xlim;
% text(L2(1:2:end), smooth_n_2(1:2:end)+0.01*(ylm(2) - ylm(1)), txts_v
    (1:2:end));
xlabel('Error Norm'); ylabel('Smothing Norm');
if run_only_Lcurve ~= 1
    title_str = 'L-curve Type2';
    Script_save_title_figs;
end

% %%
% i1 = -lb+1;
% i2 = length(qQ_p);
% %i1 = 11;
% %i2 = 16;
% figure;
% loglog(L2(i1:i2), smooth_n_2(i1:i2)); hold on; grid on;
% %plot(L2(i1:i2), smooth_n_2(i1:i2)); hold on; grid on;
% txts_v = sprintfc('%.0e',qQ_p);
% text(L2(i1:i2), smooth_n_2(i1:i2), txts_v(i1:i2));
% xlabel('Error Norm'); ylabel('Smothing Norm');

```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
%Type 3 ---
%In Cumbo et al.
% createfigure(L3, smooth_n_2);
% text(L3, smooth_n_2, txts_v);

%Type 4 ---
%In Cumbo et al. [NEEDS WORK]
%Using random walk covariance
%S_ct = (dt * 1/dt * mean(p_real))^2

function createfigure(X1, Y1)
%CREATEFIGURE(X1, Y1)
%  X1:  vector of x data
%  Y1:  vector of y data
%  no:  figure number

% Create figure
figure1 = figure;

% Create axes
axes1 = axes('Parent',figure1);
hold(axes1,'on');

% Create loglog
loglog(X1,Y1,'Marker','o','LineWidth',1);

% Create ylabel
ylabel('Q^p');

% Create xlabel
xlabel('Error norm');

box(axes1,'on');
% Set the remaining axes properties
set(axes1,'FontSize',12,'XGrid','on','XMinorTick','on','XScale','log'
    ,...
    'YGrid','on','YMinorTick','on','YScale','log');
end
```

Listing B.16: **AKF L-curve Function**

```
function [sp_opt, Jfit2_opt] = L_curve(J_fit_error2, J_fit2, s_p, step
    , theta_old, e_1, e_2,theta_position)
    %  Input: J_fiterror2  % (x-axis), s_p (y-axis)
    x = (log10(J_fit_error2(:)) - min(log10(J_fit_error2(:)))) ...
    / (max(log10(J_fit_error2(:))) - min(log10(J_fit_error2(:))));
    y = (log10(s_p(:)) - min(log10(s_p(:)))) ...
    / (max(log10(s_p(:))) - min(log10(s_p(:))));
    % Numerical derivatives
    dx = gradient(x);
    dy = gradient(y);
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
    d2x = gradient(dx);
    d2y = gradient(dy);

    % Curvature formula
    kappa = abs(dx .* d2y - dy .* d2x) ./ (dx.^2 + dy.^2).^(3/2);

    % Find index of maximum curvature (L-curve "corner")
    [~, idx_opt] = max(kappa);

    % Optimal values
    Jfit2_opt = J_fit_error2(idx_opt);
    sp_opt = s_p(idx_opt);

    % Display results
    disp('---- Optimal values from L-curve for method 3: ----')
    fprintf('Optimal J_fiterror2 = %.2e\n', Jfit2_opt);
    fprintf('Optimal s_p          = %.2e\n', sp_opt);

    figure;
    semilogy(J_fit2, s_p', "g--o", J_fit_error2, s_p', "b--o" );
    hold on;
    loglog(Jfit2_opt, sp_opt, 'rs', 'MarkerSize', 10, 'MarkerFaceColor
       ','r');
    % Use LaTeX interpreter for the title
    if ~isempty(theta_position)
        title(sprintf('L-curve values with respect to Q_p values, for
           spring (theta = %.2f)', step*theta_old), ...
        'Interpreter','tex');
    else
        title('L-curve values with respect to Q_p values','Interpreter
           ','tex')
    end
    legend('J_fit2 without errors', 'J_fit2 with errors'); %
       underscore needs escaping
    xlabel('J fit');
    ylabel(' _p ');  % escape underscore
    grid on
    hold off
    addErrorBox(e_1,e_2);
    if isempty(theta_position)
        filename = sprintf('Lcurve_error_%.2f.pdf', e_1);
        % Save as vector PDF (publication quality)
        exportgraphics(gcf, filename, 'ContentType','vector');
        %close(gcf); % prevent too many open figures
    end
end
```

Listing B.17: **L-curve for Formulation 2**

```
function stop = outfun(x,~,state)
% For figure Iteration/X Convergence of \theta/Y
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
    stop = false; % Do not stop
    persistent theta_store
    switch state
        case 'init'
            theta_store = [];
        case 'iter'
            theta_store = [theta_store; x(:)'];
        case 'done'
            assignin('base','theta_history',theta_store); % Save in
               base workspace
    end
end
```

Listing B.18: $\boldsymbol{\theta}$ **Convergence vs Iteration Figure Function**



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

### B.2.5 Response Reconstruction & Load Identification Formulations

The first method, AKA Augmented Kalman Filter was provided by my DIT supervisor, Dr. Costas Papadimitriou to whom I am gratefully thankful. The function is presented below:

```
function [p_est,x_est,Px, norm_Pxs, norm_Kxs] =first_method(Ax,B,C,D,
    Ap,Qx,Qp,R,x_0, Fx_0,Px_0,y_meas,Ndata)
% AKF algorithm
% Function AKF
% Transforms the state space equation into an augmented system and
    performs
% the estimation on the augmented system
%
% INPUT Arguments:
% Ax, Bm C, D = State and observation matrices for original discrete
    system
% Ap = state matrix for input model for load p
% Qx = covariance of state space equation error vector e
% Qp = covariance of load state space equation error ep
% R = covariance of observation error vector w
% x_0 = initial value of augmented state vector [x;p]
% Px_0 = initial value of the covariance of state vector
% y_meas = measuements; a matrix of size (ny x Ndata), where ny=size(C
    ,1)
% Ndata = number of data points
%
% State space system
% x_{k+1}=Ax*x_k + e_k,              e_k ~ N(0,Qx)
% Load model
% p_{k+1} = Ap*p_k + ep_k,           ep_k ~ N(0,Qp)
% Observation equation
% y_k=C*x_k + D * p_k + w_k,         w_k ~ N(0,R)
%
% Form the augmented state space system
% Introduce the augmented vector: x_k=[x_k;p_k]'
% x_{k+1}=A*x_k + n_k,                n_k ~ N(0,Q)
% A = augmented state matrix (discrete form)
% A=[Ax B; 0 Ap]
% Observation equation
% y_k=G*x_k + w_k,                    w_k ~ N(0,R)
% G = augmented matrix (discrete form)
% G=[C  D]
%
% OUTPUT Arguments
% x_est = matrix (2*ndof+np x Ndata) that contains the augmented state
%           vector (u,u_dot,p)'; u=displacement vector, u_dot=velocity
%           vector, p=load vector
% p_est = matrix (np x Ndata) that contains the loads p
% p_est = x_est(2*ndof+1:end,:);
% Px = Covariance matrix for the augmented state vector
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
42
43 ndof=size(Ax,1)/2;
44 np=size(Ap,1);
45 % Creating the augmented matrices
46 A = [Ax B; zeros(np,2*ndof) eye(np)];
47 G = [C D];
48 % Augmented state process covariance
49 Z = zeros(2*ndof,np);
50 Q = [Qx Z; Z' Qp];
51 % Preallocate
52 x_est = zeros(2*ndof+np,Ndata);
53 Px = zeros(2*ndof+np,2*ndof+np,Ndata);
54 norm_Pxs = zeros(Ndata,1);   % ??????????????
55 norm_Kxs = zeros(Ndata-1,1); % ??????????????
56 % Initial values
57 x_est(:,1)=[x_0; Fx_0];
58 Px(:,:,1)=Px_0;
59
60 for k = 1:Ndata-1
61     % 1. Time update for state ---
62     x_est(:,k+1) = A*x_est(:,k);
63     Px(:,:,k+1) = A*Px(:,:,k)*A'+Q;
64
65     % 2. Kalman gain for state estimation ---
66     Q_inverted = G*Px(:,:,k+1)*G'+R;
67     %Q_inverted = (Q_inverted+Q_inverted') / 2;
68     Kx = Px(:,:,k+1)*G'*inv(Q_inverted);
69     %eig_inverted(:,k+1) = eig(Q_inverted);
70
71     %eig_inverted(:,k+1) = eig(G*Px(:,:,k+1)*G'+R);
72     %Kx = Px(:,:,k+1)*G'*inv(G*Px(:,:,k+1)*G'+R);
73
74     % 3. Measurement update for state ---
75     x_est(:,k+1) = x_est(:,k+1)+Kx*(y_meas(:,k+1)-G*x_est(:,k+1));
76     Px(:,:,k+1) = Px(:,:,k+1) - Kx*G*Px(:,:,k+1);
77
78     %Added---
79     norm_Pxs(k+1) = norm(Px(:,:,k+1),'fro');
80     norm_Kxs(k) = norm(Kx,'fro');
81 end
82 p_est = x_est(2*ndof+1:end,:);
```

Listing B.19: **AKF Method Function**

Afterwards, the proposed formulations 1 & analyzed in Chapter 3 are presented from here, respectively. Formulation 1:

```
1 function [x_matrix, F_matrix] = second_method(A, C, B, D, b, ndof, N,
      n_p, yk, x_0, F_0)%, Gzz_size, n_sensors, n_excitations,yk)
2     % Second Method = Formulation 1
3     I=eye(2*ndof);
4
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
    G=[I+b*C'*C  b*C'*D; % for
        b*D'*C    b*D'*D];
    % The first iteration is executed manually due to the difference
       between the sums of the terms!!!
    dk=[b*C'*yk(:,1) + [A  B]*[x_0; F_0];
        b*D'*yk(:, 1)];

    z(:,1)=G\dk;
    % Now, iteration for the rest of the time instances. All the terms
        are taken into consideration!!
    for i=2:N
        dk=[b*C'*yk(:, i)+ [A  B]*z(:,i-1); % the calculations from
           the previous time checkpoint/instance x(i) = A*x(i-1) + B*
           F(i-1)
            b*D'*yk(:, i)];
        z(:,i)=G\dk;
    end
    % RRE & LID
    x_matrix = z(1:2*ndof,:);
    F_matrix = z(2*ndof+1:2*ndof+n_p,:);
end
```

Listing B.20: **Formulation 1 Function**

Finally, Formulation 2:

```
function [x_matrix3, F_matrix3] = third_method(A, C, B, D, Q_x, Q_n,
   Q_p, A_p, ndof, N, n_p, yk, x_0, F_0)%, Gzz_size, n_sensors,
   n_excitations,yk)
    % Third Method = Formulation 2
    G=[(Q_x+C'*Q_n*C)  (C'*Q_n*D); % for
        (D'*Q_n*C)   (Q_p + D'*Q_n*D)];
    % The first iteration is executed manually due to the difference
       between the sums of the terms!!!
    dk=[(C')*Q_n*yk(:,1); (D')*Q_n*yk(:,1)]+ [Q_x*A  Q_x*B; zeros(n_p
       ,2*ndof) Q_p*A_p]*[x_0; F_0];

    z(:,1)=G\dk;
    % Now, iteration for the rest of the time instances. All the terms
        are taken into consideration!!
    for i=2:N
        dk=[(C')*Q_n*yk(:, i); (D')*Q_n*yk(:, i)] +  [Q_x*A  Q_x*B;
           zeros(n_p,2*ndof) Q_p*A_p]*z(:,i-1);
        z(:,i)=G\dk;
    end
    % LID & RRE:
    x_matrix3 = z(1:2*ndof,:);
    F_matrix3 = z(2*ndof+1:2*ndof+n_p,:);
end
```

Listing B.21: **Formulation 2 Function**



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

### B.2.6 Figures

Last but not least, AKF, Formulation 1 & Formulation 2 estimations will be compared through the figure section. More precisely, firstly a function for the legends of the figures was created for time and cost saving:

```
function [legendNoModal, legendModal, legendCombined] = legends(method)
% I will automatically create legends in order to spare lines and time.
% Preallocation:
    legendNoModal = strings(1, 1 + size(method,2));
    legendModal = strings(1, 1 + size(method,2));
    legendCombined = strings(1, 1 + 2*size(method,2));
% The first legend is always the Simulated/Theoretical:
    legendNoModal(1) = "Simulated";
    legendModal(1) = "Simulated";
    legendCombined(1) = "Simulated";
    for i = 1:size(method,2)
        if method(i) == 1
            legendNoModal(i+1) = "AKF, no modal analysis";
            legendModal(i+1)   = "AKF, Modal analysis";
            legendCombined(i+1) = "AKF, Modal analysis";
            legendCombined(1+size(method,2)+i) = "AKF, no modal
                analysis";
        end
        if method(i) == 2
            legendNoModal(i+1) = "Formulation 1, no modal analysis";
            legendModal(i+1)   = "Formulation 1, Modal analysis";
            legendCombined(i+1) = "Formulation 1, Modal analysis";
            legendCombined(1+size(method,2)+i) = "Formulation 1, no
                modal analysis";
        end
        if method(i) == 3
            legendNoModal(i+1) = "Formulation 2, no modal analysis";
            legendModal(i+1)   = "Formulation 2, Modal analysis";
            legendCombined(i+1) = "Formulation 2, Modal analysis";
            legendCombined(1+size(method,2)+i) = "Formulation 2, no
                modal analysis";
        end
    end
end
```

Listing B.22: **Function for Legends Creation**

Secondly, LID figures function:

```
function [] = p_figures(p_est1, p_est2, p_est3, p_est1m, p_est2m,
    p_est3m, force, n_p, t, Fx_0, base, R_position, method,
    modalanalysis, nomodalanalysis, e_1, e_2)
% This functions illustrates the inputs of each excitation
% Compares the estimation of AKF, Form1 & Form2 altogether
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
% n_p: Number of loads

% Define markers for each method
markers = {'o','s','x','d','^','v','>','<','p','h'};

% Add initial condition Fx_0 if not empty
if ~isempty(p_est1);  p_est1  = [Fx_0 p_est1];  end
if ~isempty(p_est2);  p_est2  = [Fx_0 p_est2];  end
if ~isempty(p_est3);  p_est3  = [Fx_0 p_est3];  end
if ~isempty(p_est1m); p_est1m = [Fx_0 p_est1m]; end
if ~isempty(p_est2m); p_est2m = [Fx_0 p_est2m]; end
if ~isempty(p_est3m); p_est3m = [Fx_0 p_est3m]; end

% Force vs Accel label
if base
    yLabelText = 'Acceleration (mm/s^2)';
    titlePrefix = 'Acceleration plot with base excitation';
else
    yLabelText = 'Force (N)';
    titlePrefix = 'Force plot at DoF:';
end

% Legend Labels Creation
[~, ~, legendCombined] = legends(method);

% === Both Comparisons ===
for i = 1:n_p
    figure('Units','inches','Position',[1 1 6 4], 'Visible','on');
    hold on;
    plot(t, force(i,:), "k--",'LineWidth',2);

    if modalanalysis
        if any(method == 1), plot(t', p_est1m(i,:), 'r','Marker',
           markers{1}, 'MarkerIndices', 1:round(length(t)/20):length(
           t)); end
        if any(method == 2), plot(t', p_est2m(i,:), 'g','Marker',
           markers{2}, 'MarkerIndices', 1:round(length(t)/20):length(
           t)); end
        if any(method == 3), plot(t', p_est3m(i,:), 'b','Marker',
           markers{3}, 'MarkerIndices', 1:round(length(t)/20):length(
           t)); end
    end

    if nomodalanalysis
        if any(method == 1), plot(t', p_est1(i,:), 'y','Marker',
           markers{4}, 'MarkerIndices', 1:round(length(t)/20):length(
           t)); end
        if any(method == 2), plot(t', p_est2(i,:), 'c','Marker',
           markers{5}, 'MarkerIndices', 1:round(length(t)/20):length(
           t)); end
        if any(method == 3), plot(t', p_est3(i,:), 'm','Marker',
```


Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
            markers{6}, 'MarkerIndices', 1:round(length(t)/20):length(
            t)); end
    end

    if base
        title(titlePrefix);
        filename = sprintf('baseexcitation_%02d_error_%.2f.pdf', i,
            e_1);
    else
        title([titlePrefix, num2str(R_position(i))]);
        filename = sprintf('input_dof_%d_error_%.2f.pdf', R_position(i
            ), e_1);
    end

    hLeg = legend(legendCombined, 'Location','northeast');
    hLeg.FontSize = hLeg.FontSize * 0.85;  % scale down legend text
    xlabel('Time (s)');
    ylabel(yLabelText);
    grid on;
    addErrorBox(e_1, e_2);

    % Save as vector PDF (publication quality)
    exportgraphics(gcf, filename, 'ContentType','vector');
    close(gcf); % prevent too many open figures
end
end
```

Listing B.23: **Function for Input Estimation**

Thereafter, figures for Displacements, Strains and Velocities are presented downwards. Acceleration terms are not reconstructed so these kind of figures were omitted:

```
function [] = displacement_figures(x_est1,x_est2,x_est3,displ_1,
    displ_2,displ_3,yk,t, x_0_0, Fx_0,displacement,ndof,method,
    modalanalysis,nomodalanalysis,e_1,e_2)
% Compares the estimation of AKF, Form1 & Form2 altogether
% 2 Categories of Diagramms:
% 1st: Without Modal Analysis, n_p diagrams
% 2nd: With Modal Analysis, n_p diagrams
% Storing the Initial conditions
% Limited Number of DOF - All of them illustrated.
% Initial Condition Storing into one matrix
% Define markers for each method
markers = {'o','s','x','d','^','v','>','<','p','h'}; % circle, square,
    cross, diamond, etc.
    iteration = 1:size(displacement,2);
    if ~isempty(x_est1); x_est1 = [[x_0_0; Fx_0] x_est1]; end
    if ~isempty(displ_1); displ_1 = [x_0_0(1:ndof) displ_1]; end
    if ~isempty(x_est2); x_est2 = [x_0_0 x_est2]; end
    if ~isempty(displ_2); displ_2 = [x_0_0(1:ndof) displ_2]; end
    if ~isempty(x_est3); x_est3 = [x_0_0 x_est3]; end
    if ~isempty(displ_3); displ_3 = [x_0_0(1:ndof) displ_3]; end
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
18
19      % Legend Labels Creation
20      [~, ~, legendCombined] = legends(method);
21
22  % Both Comparisons
23          for i = iteration
24              figure('Units','inches','Position',[1 1 12 6], 'Visible','
                    on');
25              plot(t', yk(displacement(i),:),"k--", 'Linewidth', 2 );
26              hold on
27              if modalanalysis
28                  if any(method == 1), plot(t', displ_1(displacement(i)
                        ,:),'r','Marker', markers{1}, 'MarkerIndices', 1:
                        round(length(t)/20):length(t)); end
29                  if any(method == 2), plot(t', displ_2(displacement(i)
                        ,:),'g','Marker', markers{2}, 'MarkerIndices', 1:
                        round(length(t)/20):length(t)); end
30                  if any(method == 3), plot(t', displ_3(displacement(i)
                        ,:), 'b','Marker', markers{3}, 'MarkerIndices', 1:
                        round(length(t)/20):length(t)); end
31              end
32              if nomodalanalysis
33                  if any(method == 1), plot(t', x_est1(displacement(i)
                        ,:),'y','Marker', markers{4}, 'MarkerIndices', 1:
                        round(length(t)/20):length(t)); end
34                  if any(method == 2), plot(t', x_est2(displacement(i)
                        ,:),'c','Marker', markers{5}, 'MarkerIndices', 1:
                        round(length(t)/20):length(t)); end
35                  if any(method == 3), plot(t', x_est3(displacement(i)
                        ,:),'m','Marker', markers{6}, 'MarkerIndices', 1:
                        round(length(t)/20):length(t)); end
36              end
37              title('Displacement␣plot␣at␣DoF', num2str(displacement(i))
                    );
38              legend(legendCombined)
39              xlabel('Time␣(s)');
40              ylabel('Displacement␣(mm)');
41              hold off
42              grid on;
43              %fig = fig +1;
44              addErrorBox(e_1,e_2);
45              % Save as Vector PDF Save each as PDF with unique name
46              filename = sprintf('displacement_%02d_error_%.2f.pdf', i,
                    e_1);
47              exportgraphics(gcf, filename, 'ContentType', 'vector');
48              close(gcf); % optional: close to avoid too many open
                    figures
49          end
50  end
```

Listing B.24: **Function for Displacement Estimation**



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
function [] = strain_figures(x_est1,x_est2,x_est3,displ_1,displ_2,
    displ_3,yk,t, x_0_0, Fx_0,strain,ndof,method,modalanalysis,
    nomodalanalysis,e_1,e_2)
% Compares the estimation of AKF, Form1 & Form2 altogether
% 2 Categories of Diagramms in the same figure:
% 1st: Without Modal Analysis, n_p diagrams
% 2nd: With Modal Analysis, n_p diagrams
% Storing the Initial conditions
% Limited Number of DOF - All of them illustrated.
% Initial Condition Storing into one matrix
% Define markers for each method
markers = {'o','s','x','d','^','v','>','<','p','h'}; % circle, square,
     cross, diamond, etc.

    iteration = 1:size(strain,2);
    if ~isempty(x_est1); x_est1 = [[x_0_0; Fx_0] x_est1]; end
    if ~isempty(displ_1); displ_1 = [x_0_0(1:ndof) displ_1]; end
    if ~isempty(x_est2); x_est2 = [x_0_0 x_est2]; end
    if ~isempty(displ_2); displ_2 = [x_0_0(1:ndof) displ_2]; end
    if ~isempty(x_est3); x_est3 = [x_0_0 x_est3]; end
    if ~isempty(displ_3); displ_3 = [x_0_0(1:ndof) displ_3]; end

    % Legend Labels Creation
    [~, ~, legendCombined] = legends(method);

% Both Comparisons
        for i = iteration
            figure('Units','inches','Position',[1 1 12 6], 'Visible','
               on');
            plot(t', yk(2*ndof + strain(i),:),"k--", 'Linewidth', 2 );
            hold on
            if strain(i) == 1 % x1 - x0 where x0 doesn't exist!
                if modalanalysis
                    if any(method == 1), plot(t', displ_1(strain(i),:)
                       ,'r','Marker', markers{1}, 'MarkerIndices', 1:
                       round(length(t)/20):length(t)); end
                    if any(method == 2), plot(t', displ_2(strain(i),:)
                       ,'g','Marker', markers{2}, 'MarkerIndices', 1:
                       round(length(t)/20):length(t)); end
                    if any(method == 3), plot(t', displ_3(strain(i),:)
                       , 'b','Marker', markers{3}, 'MarkerIndices',
                       1:round(length(t)/20):length(t)); end
                end
                if nomodalanalysis
                    if any(method == 1), plot(t', x_est1(strain(i),:),
                       'y','Marker', markers{4}, 'MarkerIndices', 1:
                       round(length(t)/20):length(t)); end
                    if any(method == 2), plot(t', x_est2(strain(i),:),
                       'c','Marker', markers{5}, 'MarkerIndices', 1:
                       round(length(t)/20):length(t)); end
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
                    if any(method == 3), plot(t', x_est3(strain(i),:),
                        'm','Marker', markers{6}, 'MarkerIndices', 1:
                        round(length(t)/20):length(t)); end
                end
            else
                if modalanalysis
                    if any(method == 1), plot(t', displ_1(strain(i),:)
                         - displ_1(strain(i) -1,:) ,'r','Marker',
                        markers{1}, 'MarkerIndices', 1:round(length(t)
                        /20):length(t)); end
                    if any(method == 2), plot(t', displ_2(strain(i),:)
                         - displ_2(strain(i) - 1,:),'g','Marker',
                        markers{2}, 'MarkerIndices', 1:round(length(t)
                        /20):length(t)); end
                    if any(method == 3), plot(t', displ_3(strain(i),:)
                         - displ_3(strain(i) - 1,:), 'b','Marker',
                        markers{3}, 'MarkerIndices', 1:round(length(t)
                        /20):length(t)); end
                end
                if nomodalanalysis
                    if any(method == 1), plot(t', x_est1(strain(i),:)
                        - x_est1(strain(i) - 1,:) ,'y','Marker',
                        markers{4}, 'MarkerIndices', 1:round(length(t)
                        /20):length(t)); end
                    if any(method == 2), plot(t', x_est2(strain(i),:)
                         - x_est2(strain(i) - 1,:),'c','Marker',
                        markers{5}, 'MarkerIndices', 1:round(length(t)
                        /20):length(t)); end
                    if any(method == 3), plot(t', x_est3(strain(i),:)
                        - x_est3(strain(i) - 1,:), 'm','Marker',
                        markers{6}, 'MarkerIndices', 1:round(length(t)
                        /20):length(t)); end
                end
            end
            title('Strain plot at DoF', num2str(strain(i)));
            legend(legendCombined)
            xlabel('Time (s)');
            ylabel('Strain (mm)');
            hold off
            grid on;
            %fig = fig +1;
            addErrorBox(e_1,e_2);
            filename = sprintf('strain_%02d_error_%.2f.pdf', i,e_1);
            exportgraphics(gcf, filename, 'ContentType', 'vector');
            close(gcf); % optional: close to avoid too many open
                figures
        end
end
```

Listing B.25: **Function for Strain Estimation**



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
function [] = velocity_figures(x_est1,x_est2,x_est3,veloc_1,veloc_2,
    veloc_3,yk,t, x_0_0, Fx_0,velocity,ndof,method,modalanalysis,
    nomodalanalysis,e_1,e_2)
% Compares the estimation of AKF, Form1 & Form2 altogether
% 2 Categories of Diagramms:
% 1st: Without Modal Analysis, n_p diagrams
% 2nd: With Modal Analysis, n_p diagrams
% Storing the Initial conditions
% Limited Number of DOF - All of them illustrated.
% Initial Condition Storing into one matrix
    % Define markers for each method
    markers = {'o','s','x','d','^','v','>','<','p','h'}; % circle,
        square, cross, diamond, etc.

    % Legend Labels Creation:
    [~, ~, legendCombined] = legends(method);

    iteration = 1:size(velocity,2);
    if ~isempty(x_est1); x_est1 = [[x_0_0; Fx_0] x_est1]; end
    if ~isempty(veloc_1); veloc_1 = [x_0_0(ndof+1:2*ndof) veloc_1];
        end
    if ~isempty(x_est2); x_est2 = [x_0_0 x_est2]; end
    if ~isempty(veloc_2); veloc_2 = [x_0_0(ndof+1:2*ndof) veloc_2];
        end
    if ~isempty(x_est3); x_est3 = [x_0_0 x_est3]; end
    if ~isempty(veloc_3); veloc_3 = [x_0_0(ndof+1:2*ndof) veloc_3];
        end

    % Both Comparisons
        for i = iteration
            figure('Units','inches','Position',[1 1 12 6], 'Visible','
                on');
            %figure(fig);
            plot(t', yk(ndof + velocity(i),:),"k--", 'Linewidth',2);
            hold on
            if modalanalysis
                if any(method == 1), plot(t', veloc_1(velocity(i),:),'
                    r','Marker', markers{1}, 'MarkerIndices', 1:round(
                    length(t)/20):length(t)); end
                if any(method == 2), plot(t', veloc_1(velocity(i),:),'
                    g','Marker', markers{2}, 'MarkerIndices', 1:round(
                    length(t)/20):length(t)); end
                if any(method == 3), plot(t', veloc_3(velocity(i),:),
                    'b','Marker', markers{3}, 'MarkerIndices', 1:round
                    (length(t)/20):length(t)); end
            end
            if nomodalanalysis
                if any(method == 1), plot(t', x_est1(ndof + velocity(i
                    ),:),'y','Marker', markers{4}, 'MarkerIndices', 1:
                    round(length(t)/20):length(t)); end
                if any(method == 2), plot(t', x_est2(ndof + velocity(i
```



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250

```
                    ),:),'c','Marker', markers{5}, 'MarkerIndices', 1:
                    round(length(t)/20):length(t)); end
37                  if any(method == 3), plot(t', x_est3(ndof + velocity(i
                    ),:),'m','Marker', markers{6}, 'MarkerIndices', 1:
                    round(length(t)/20):length(t)); end
38              end
39              title('Velocity plot at DoF', num2str(velocity(i)));
40              legend(legendCombined);
41              xlabel('Time (s)');
42              ylabel('Velocity (mm/s)');
43              grid on;
44              hold off
45              %fig = fig +1;
46              addErrorBox(e_1,e_2);
47              filename = sprintf('velocity_%02d_error_%.2f.pdf', i,e_1);
48              exportgraphics(gcf, filename, 'ContentType', 'vector');
49              close(gcf); % optional: close to avoid too many open
                    figures
50          end
51  end
```

Listing B.26: **Function for Velocity Estimation**



Institutional Repository - Library & Information Centre - University of Thessaly
04/09/2026 10:32:05 EEST - 216.73.216.250