

UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Development of an Optimal Sensor Placement Framework for Structural Health Monitoring of an Aircraft Wing's Spar

Permalink

<https://escholarship.org/uc/item/01f1x276>

Author

Hokama Razzini, Adrielly

Publication Date

2024

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Development of an Optimal Sensor Placement Framework for Structural Health
Monitoring of an Aircraft Wing's Spar

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Structural Engineering

by

Adrielly Hokama Razzini

Committee in charge:

Professor Michael D. Todd, Chair
Professor Charles Farrar
Professor William Hodgkiss
Professor Hyonny Kim
Professor Francesco Lanza di Scalea

2024

Copyright

Adrielly Hokama Razzini, 2024

All rights reserved.

The Dissertation of Adrielly Hokama Razzini is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2024

DEDICATION

To my dear friend Allan Demétrio Sales Dias, *in memoriam*.

To my mother, Márcia Helena Hokama.

To my dog, Lilly.

TABLE OF CONTENTS

Dissertation Approval Page	iii
Dedication	iv
Table of Contents	v
List of Figures	viii
List of Tables	xvii
Acknowledgements	xviii
Vita	xxi
Abstract of the Dissertation	xxiii
Chapter 1 Introduction	1
1.1 Motivation	1
1.2 Structural Health Monitoring	3
1.2.1 Optimal sensor placement	5
1.3 Proposed optimal sensor placement framework	6
1.3.1 Define the structure to be monitored and damage of interest	7
1.3.2 Define the type of sensor(s) to be used	9
1.3.3 Data acquisition and/or synthetic data generation	11
1.3.4 Develop the damage diagnosis model	12
1.4 Define the cost function	13
1.5 Define the optimization function and perform optimization	14
1.6 Summary and dissertation organization	15
1.7 Contributions of the dissertation work	16
Chapter 2 Data generation - Development and validation of the FE model	18
2.1 The finite element model	18
2.1.1 Geometry and material properties	18
2.1.2 The mesh	23
2.1.3 Constraints	24
2.1.4 Loads and boundary conditions	26
2.1.5 Results from pristine beam	29
2.1.6 Debonding damage and buckling	32
2.1.7 Collecting strain data	36
2.2 Model validation	41
2.3 Mechanics of Composite Laminates	42
2.3.1 Engineering the Caps, Shear Clips, and Web	46
2.4 Manufacturing	47

2.4.1	Top and Bottom Caps	47
2.4.2	Shear Clips	48
2.4.3	Web	48
2.4.4	Assembly	51
2.5	Experimental Setup	55
2.6	Summary	68
Chapter 3	Data generation - Surrogate models	69
3.1	Statistical Modeling	69
3.2	Running Abaqus with Python	73
3.2.1	1 meter sub-model	75
3.2.2	Creating sampling distribution	75
3.2.3	Data assembly	82
3.2.4	Two-step outlier detector	82
3.2.5	Data fusion	84
3.3	Machine learning surrogate models	86
3.4	Surrogate model - eigenvalues	87
3.4.1	Gaussian process regressor	88
3.5	Surrogate model - Strain measurements	92
3.5.1	Code to find the coordinates of points p1, p2, p3, and p4	93
3.5.2	Strain reconstruction of each of the 5 segments	96
3.5.3	Artificial Neural Network to predict coordinates of p1, p2, p3, and p4.	98
3.5.4	Defining the neural network's architecture	101
3.5.5	Results from the ANN model	105
3.5.6	Strain reconstruction using coordinates obtained with the forward model	109
3.6	Expansion to the 5m model	111
3.6.1	Adjustments to Python Scripts	111
3.6.2	Long run of 5m model	112
3.6.3	Eigenvalue results	114
3.6.4	Strain results	114
3.6.5	Updated forward models for the 5m beam	119
3.6.6	Strain reconstruction of the 5m beam	126
3.7	Improvement in computational performance	131
3.8	Summary	131
Chapter 4	Damage diagnosis	133
4.1	Bayesian inference	134
4.2	Likelihood computation	136
4.2.1	Maximum likelihood estimation	138
4.2.2	Maximum a posteriori estimation	140
4.2.3	Linking damage diagnosis to criticality	141

4.2.4	Performance of Bayesian inference model considering all sensors are present	142
4.3	Observability limitations	145
4.4	Bayesian model updating	153
4.4.1	Sequential Monte Carlo	154
4.5	Summary	159
Chapter 5	Optimal Sensor Placement	160
5.1	Defining the cost function	161
5.2	Bayesian Optimization	164
5.2.1	Results from first optimization	169
5.2.2	Results from second optimization	171
5.2.3	Results from the third optimization	173
5.2.4	Results and Summary	175
Chapter 6	Conclusions and future work	184
6.1	Future work	187
Bibliography		188

LIST OF FIGURES

Figure 1.1.	Increase of the weight content of composites in aircraft structures over a 30-year time span [1]. “A” represents Airbus aircraft, while “7” represents Boeing’s 700 series.	2
Figure 1.2.	Main components of an aircraft’s wing. Image obtained from [2]. ...	7
Figure 1.3.	Cross sectional views of possible buckling mode shapes for delaminated plate: (a) global buckling; (b) mixed global and local deformation; (c) local buckling only of sublaminates; (d) local buckling of single sublamine. Image obtained from [3].	8
Figure 1.4.	How fiber Bragg grating sensors work. Figure obtained from [4].	10
Figure 2.1.	Composite wing’s spar.	19
Figure 2.2.	Nomenclature of the spar’s components referenced in the dissertation.	19
Figure 2.3.	Drawing of the wing’s spar (I beam) including composites ply orientation. This represents a 1m subsection of the 5m beam. All units in [mm].	20
Figure 2.4.	Section A-A from Figure 2.3. All units in [mm].	20
Figure 2.5.	Dimensions of the shear clips. All units in [m].	21
Figure 2.6.	Full assembly of the spar created on Abaqus/CAE. The local coordinate systems are shown in yellow, where the x-direction is along the length of the beam, the z-direction is perpendicular to the surface and the y-direction completes the right-hand coordinate system.	22
Figure 2.7.	Comparison of initial mesh design (a) and updated mesh design (b). Solid elements were replaced by continuum shell elements (SC8R) on the caps and shear clips.	24
Figure 2.8.	Tie constraints applied to the model to emulate the adhesive layer of structural epoxy that attaches all parts together.	25
Figure 2.9.	Loading scheme provided by the project’s sponsors, describing the maximum service load distribution on the beam. Units of the x-axis are in [mm].	26

Figure 2.10.	Definition of coupling constraint to apply loads to the structure. (a) shows the reference point, (b) shows the selection of the nodes that correspond to master structures from the tie constraints, and (c) shows the coupling constraint.	28
Figure 2.11.	Loads applied to the spar.	28
Figure 2.12.	Magnitude of the displacements, units in m. The fixed end is on the left side of the spar.	29
Figure 2.13.	Strain in the 1 direction, following the spar cap's coordinate system (direction 1 is the axial direction, along the beam's length).	30
Figure 2.14.	Strain in the 2 direction, following the spar cap's coordinate system. .	31
Figure 2.15.	Strain measurement in the 1 direction at a path below the top-right shear clip.	31
Figure 2.16.	Buckling analysis of two plates of random geometry bonded together, used for proof of concept. On (a), global buckling occurs. On (b), as a small portion of the bonding is removed, local buckling occurs. ...	33
Figure 2.17.	Partition created to emulate debonding between different parts.	34
Figure 2.18.	Buckling simulation of pristine beam. As there is no damage, global buckling occurs.	35
Figure 2.19.	Buckling simulation of a damaged beam. Local buckling occurs at the debonding region.	35
Figure 2.20.	Visual representation of how damage size and damage location are defined. Regions in purple indicate that tie constraints were applied between them and the top cap (removed from the image for a better view).	36
Figure 2.21.	Coordinate system specified to guarantee that ε_{11} is being measured in the longitudinal direction of the beam, and that ε_{22} is orthogonal to it.	37
Figure 2.22.	Paths defined for strain collection.	38
Figure 2.23.	Strain ε_{11} collected on all paths defined in Figure 2.22. Notice that this is a 1m submodel of the 5m beam.	39
Figure 2.24.	Strain ε_{22} collected on all paths defined in Figure 2.22. Notice that this is a 1m submodel of the 5m beam.	40

Figure 2.25.	Global $x - y$ and local $x_1 - x_2$ coordinate system on a single ply composite under plane-stress, and load only on x direction.	42
Figure 2.26.	Top and bottom caps vacuum-bag curing schematics.	48
Figure 2.27.	Shear clips vacuum-bag curing schematics.	49
Figure 2.28.	Manufacturing of the shear clips: (a) wet layup and (b) vacuum bag cure.	50
Figure 2.29.	Web carbon fiber - foam sandwich vacuum-bag curing schematics. ...	51
Figure 2.30.	I-beam components after they were trimmed to the correct size.	52
Figure 2.31.	Two stages of the I-beam assembly: (a) the bottom cap, two shear clips and web were assembled, (b) remaining cap and two shear chips where assembled to the “T” beam formed in stage 1.	53
Figure 2.32.	Assembly of the I beam components forming first “T” beam (a) and the final I-beam (b).	54
Figure 2.33.	Debonding damage insertion by adding a release film layer (a) to some of the bonding area between the top cap and the web, (b) to the same bonding area of the top cap (redundancy). (c) Shows the debonding defect on the completely assembled composite I-beam.	55
Figure 2.34.	Cantilever beam at (a) no deflection and (b) 30 mm deflection due to a tip load.	56
Figure 2.35.	Experimental Setup.	57
Figure 2.36.	Baseline voltage at the 15 strain gauges. In blue are the operational strain gauges and in red the defective strain gauges.	58
Figure 2.37.	Voltage signal at the 15 strain gauges when the beam’s tip is deflected 30 mm. In blue are the operational strain gauges and in red the defective strain gauges.	59
Figure 2.38.	Amplitude V_r at the 15 strain gauges when the beam’s tip is deflected 30 mm. In blue are the operational strain gauges and in red the defective strain gauges.	60
Figure 2.39.	Strain at the 15 strain gauges when the beam’s tip is deflected 30 mm. In blue are the operational strain gauges and in red the defective strain gauges.	62

Figure 2.40.	FE Model's strain versus strain from experimental data for a prescribed tip displacement of 10 mm: (a) average experimental strain and (b) average experimental strain and it's $\pm\sigma_\epsilon$ confidence intervals.	63
Figure 2.41.	FE Model's strain versus strain from experimental data for a prescribed tip displacement of 20 mm: (a) average experimental strain and (b) average experimental strain and it's $\pm\sigma_\epsilon$ confidence intervals.	64
Figure 2.42.	FE Model's strain versus strain from experimental data for a prescribed tip displacement of 30 mm: (a) average experimental strain and (b) average experimental strain and it's $\pm\sigma_\epsilon$ confidence intervals.	65
Figure 2.43.	FE Model's strain versus strain from experimental data for a prescribed tip displacement of 40 mm: (a) average experimental strain and (b) average experimental strain and it's $\pm\sigma_\epsilon$ confidence intervals.	66
Figure 3.1.	Prior distributions of damage size (a) and damage location (b).	70
Figure 3.2.	Flowchart of how Abaqus is executed using Python scripts.	75
Figure 3.3.	Histograms of damage size and damage location generated using Latin Hypercube sampling for 800 samples.	76
Figure 3.4.	Sampling space for the first long run of the 1m beam.	77
Figure 3.5.	Results of the first long run (800 cases) - Distribution of eigenvalues (left) and eigenvalues per damage location for a 0.05m damage size (right).	78
Figure 3.6.	New samples for the second long run, with a new damage size distribution.	79
Figure 3.7.	Eigenvalue results from the buckling analysis of the second long run (part 1 of 2).	80
Figure 3.8.	Eigenvalue results from the buckling analysis of the second long run (part 2 of 2).	81
Figure 3.9.	Data assembly process. Left: files extracted from scripts. Right: files after assembly.	82
Figure 3.10.	Left: example of an atypical strain behavior found by the outlier detector. Right: same sample after another simulation, with the correct result (baseline signal removed, so there is no slope on the strain measurement).	83

Figure 3.11.	Outlier eigenvalues found by outlier detector	84
Figure 3.12.	Flowchart of data fusion process.	85
Figure 3.13.	Measurements extracted from path 6, for damage size 0.05m and damage location 0.7040m. Left: original measurement. Right: Baseline subtracted from results.	86
Figure 3.14.	Representation of forward and inverse models created to replace Abaqus.	86
Figure 3.15.	Actual (blue) and predicted (orange) eigenvalue from the polynomial fit for 15cm damage.	88
Figure 3.16.	Eigenvalues per damage location, for damage size of 1cm (left) and 2cm (right).	90
Figure 3.17.	Eigenvalue predictions of the Gaussian Process Regression model, for damage sizes between 3 and 15cm for all damage locations.	91
Figure 3.18.	Principal component analysis of strain data for path 6.	93
Figure 3.19.	Strain measurement minus baseline broken down in 5 segments, with peaks and valleys highlighted by points p1 to p4.	94
Figure 3.20.	Samples of strain measurements for different damage sizes and locations.	95
Figure 3.21.	Example of function to find peaks being used under noise.	96
Figure 3.22.	Strain reconstruction based on equations from Table 3.2. Left: reconstruction zoomed on damage location. Right: global strain reconstruction.	98
Figure 3.23.	Reconstruction of strain measurements shown in Figure 3.20, based only on x-y coordinates of points p1, p2, p3, and p4. Plot (2,2) is zoomed in as the damage is too small.	99
Figure 3.24.	Root-mean-squared error for each output of the neural network, with different number of layers and neurons on each layer.	102
Figure 3.25.	Chosen artificial neural network (ANN) architecture, with a single hidden layer.	103
Figure 3.26.	Flow chart of the procedure to develop the artificial neural network model.	106

Figure 3.27.	Results from artificial neural network predicting coordinates of points p1, p2, p3, and p4. The plots show the results for all samples, overlaying a comparison of the predicted and true value (from Abaqus) for all samples.	107
Figure 3.28.	Results from artificial neural network predicting coordinates of points p1, p2, p3, and p4. The plots show the results of the predicted values against the true value, and a line in red.	108
Figure 3.29.	Flow chart of the procedure to reconstruct strain ε_{11} from path 6. ..	109
Figure 3.30.	Strain reconstruction of different damage sizes and locations, generated with the forward model.	110
Figure 3.31.	Results of strain measurements along path 6 for the 5m beam without damage (pristine). Left: full beam, right: zoomed in around 1m load region.	112
Figure 3.32.	(a) Nodal points of loading region highlighted in red, after being automatically selected by the Python scripts. (b) Creation of coupling constraints.	113
Figure 3.33.	Damage size and damage location histograms, and sampling space, for 800 samples of the 5m beam.	113
Figure 3.34.	Eigenvalues obtained from 5m beam simulation, with damage around loading region (1m cross-section).	114
Figure 3.35.	Strain ε_{11} obtained from path 6 on the 5m-long beam. Results shown focus on damage cases that are around the loading region (shown in gray). The small red dots will be used for strain reconstruction later in this section.	116
Figure 3.36.	Principal component analysis result of 5m beam, focusing on 1m loading region only.	117
Figure 3.37.	Strain reconstruction of several damage cases, using the first 35 principal components.	118
Figure 3.38.	Results of the (x,y) coordinates of the 4 points that correspond to the damage effect on the strain measurements. Results are shown for each coordinate, plotted against damage size and damage location.	120
Figure 3.39.	GPR results of the predictions of the testing data for (x, y) coordinates of points 1 and 2 (models 1 to 4).	124

Figure 3.40.	GPR results of the predictions of the testing data for (x, y) coordinates of points 3 and 4 (models 5 to 8).	125
Figure 3.41.	Strain reconstruction for path 6 along the entire 5m beam, for damage size= 15cm and damage location = 0.990m.	127
Figure 3.42.	Reconstruction of a damage case that ended right before 1m.	127
Figure 3.43.	Reconstruction of a damage case that occurs through the 1m loading region.	128
Figure 3.44.	Reconstruction of a damage case that occurs right before the second loading region (2 meters).	128
Figure 3.45.	Reconstruction of a damage case far from loading region.	129
Figure 3.46.	Eigenvalues predicted using GP regressor model for 5m beam.	129
Figure 3.47.	Flowchart of the surrogate model for strain reconstruction.	130
Figure 4.1.	Prior distributions of damage size (a) and damage location (b).	140
Figure 4.2.	Flow chart of how a noisy strain measurement processed to be converted into damage criticality level.	141
Figure 4.3.	Confusion charts showing the performance of 833 sensors for criticality assessment using the maximum likelihood estimation (left) and maximum a posteriori estimation (right).	143
Figure 4.4.	MSE from 833 sensors used for damage inference using the maximum likelihood estimation (left) and maximum a posteriori estimation (right).	144
Figure 4.5.	Strain signature of damage cases far from the loading region, after subtracting the baseline.	145
Figure 4.6.	Left axis: Coordinates of point 1 compared to the noise level (black dashed line). Right axis: eigenvalues per x coordinate.	146
Figure 4.7.	Left axis: Coordinates of point 2 compared to the noise level (black dashed line). Right axis: eigenvalues per x coordinate.	147
Figure 4.8.	Left axis: Coordinates of point 3 compared to the noise level (black dashed line). Right axis: eigenvalues per x coordinate.	147
Figure 4.9.	Left axis: Coordinates of point 4 compared to the noise level (black dashed line). Right axis: eigenvalues per x coordinate.	148

Figure 4.10.	Example of how the Bayesian inference works. Blue lines correspond to strain measurements obtained every 5mm and orange dots are where sensors were placed by the optimization.	149
Figure 4.11.	Zoomed-in view of Figure 4.10, bottom, of the subtracted strains. ...	150
Figure 4.12.	Incorrect prediction of damage location using a sensor array of 65 sensors.	151
Figure 4.13.	Another example of how the Bayesian inference works. Blue lines correspond to strain measurements obtained every 5mm and orange dots are where sensors were placed by the optimization.	152
Figure 4.14.	Correct prediction of damage size and location using a sensor array of 65 sensors optimized sequentially.	153
Figure 4.15.	Bayes model updating for $\sigma = 0.03m$ and $x = 2.072m$. The first plot shows the maximum likelihood estimation of the damage state, and the remaining plots show the evolution of the maximum a posteriori estimation. Green triangle: true value, red circle: prediction.	157
Figure 4.16.	Bayes model updating for $\sigma = 0.04m$ and $x = 2.384m$. The first plot shows the 1st MAP estimate and the second plot shows the 50th MAP estimate. Green triangle: true value, red circle: prediction.	158
Figure 4.17.	Bayes model updating for $\sigma = 0.11m$ and $x = 1.356m$. The first plot shows the 1st MAP estimate and the second plot shows the 50th MAP estimate. Green triangle: true value, red circle: prediction.	158
Figure 5.1.	Confusion chart and costs related to each prediction.	162
Figure 5.2.	Bayesian optimization looking for the best location for the first sensor in a sequential optimization.	166
Figure 5.3.	Costs for binary classification of 1st optimization.	169
Figure 5.4.	ROC curves for sequential optimization for different load levels. Left: 20 sensors, right: 40 sensors.	169
Figure 5.5.	Results from the Bayesian optimization for different number of sensors. Notice that the axes do not have the same limits.	170
Figure 5.6.	Costs for binary classification of 1st optimization.	171
Figure 5.7.	ROC curves for sequential optimization for different load levels, 2nd run.	172

Figure 5.8.	Results from the 2nd Bayesian optimization for different number of sensors. Notice that the axes do not have the same limits.	173
Figure 5.9.	Costs for binary classification of 3rd optimization.	174
Figure 5.10.	Cost per number of sensors.	177
Figure 5.11.	Histogram of costs per random sensor designs, versus optimized designs.	177
Figure 5.12.	Confusion chart for the evenly spaced sensor array.	178
Figure 5.13.	Confusion chart for the sensor array from the combinatory optimization.	178
Figure 5.14.	Confusion chart for the sensor array from the sequential optimization.	179
Figure 5.15.	Receiver Operating Characteristic (ROC) curves for a variety of sensor sets, obtained at the ultimate load level.	179
Figure 5.16.	ROC curves for 40 combinatory sensors, for different load levels. ...	180
Figure 5.17.	ROC curves for 40 sensors obtained sequentially, for different load levels.	180
Figure 5.18.	Final sensor layout after third run.	182

LIST OF TABLES

Table 2.1.	Material properties of the low density foam used in the core.....	21
Table 2.2.	Material properties of the CFRP components of the spar.	22
Table 2.3.	Properties of the different materials used for construction.	46
Table 2.4.	Effective moduli of the laminates analyzed.	47
Table 2.5.	Position of strain gauges along the I beam (units in [mm]).	57
Table 2.6.	Strain gauge properties.	61
Table 3.1.	Dynamic load description.	71
Table 3.2.	Description of each of the 5 segments used for strain reconstruction. .	97
Table 3.3.	Description of the output of each GPR model and the kernel function used.	122
Table 3.4.	MSE results using 85% of samples for training the GPR models.	123
Table 3.5.	Governing equations of each strain segment, as a function of the x coordinate of points p1-p4, for each case described in 3.6.4.	126
Table 4.1.	Seven classes of damage criticality, defined by the buckling eigenvalue, from not critical (Class 0) to most critical (Class 6).	141
Table 5.1.	SHM system design costs.	164
Table 5.2.	Final layout of each optimization run. Sensor location displayed in [mm].	181
Table 5.3.	Summary of results from optimization runs.	182

ACKNOWLEDGEMENTS

I want to start by expressing my sincerest gratitude to Professor Todd, who gave me the opportunity to join his prestigious lab and come to the US to pursue my Ph.D. Thank you for all the guidance, mentorship, trust, and encouragement throughout these years. It is comforting to know that your advisor is someone who will support you amidst all the chaos that a Ph.D. entails.

To the members of my committee: Professor Farrar, one of the most brilliant people in the SHM field and probably the most humble one, thank you for having me as your TA for the past five years and for the invaluable guidance and continuous support you have provided me; I genuinely appreciate it. Professor Lanza, thank you for your incessant kindness and all the knowledge you shared with me. Professor Kim, thank you for all the guidance you provided and your willingness to help. Finally, Professor Hodgkiss has given me an immense understanding of the digital signal-processing universe. All of you have my gratitude.

To our collaborators in this project: Moshe Tur, Iddo Kressel, and Yoav Ofir. Thank you for all the knowledge shared and productive discussions.

To my mother, Márcia Helena Hokama, who has always supported me and encouraged me to pursue my dreams, I'd like to express my uttermost gratitude and love. To my father, José Razzini, my brother Eddie, and my niece Clara, thank you for all your support.

To Izabela Bastista, without whom this journey would have been much harder, for always being by my side and never stopping believing in me. You have my deepest gratitude, support, and admiration.

To my fur baby, Lilly, who has made my world a happier place!

There are many people that contributed to this journey and made San Diego a better place. At the risk of forgetting a few names, I would like to thank Ricardo Bustamante & Valentina Vasquez, Sumit Gupta, Leticia Nocko, Chaiane Wiggers, Douglas

Bruninho Aquino, Carolina Jauregui, Geovanna Callasans, Daniela Sodré, Luis Vilani, Diptojit Datta, Andrea & Ivan, Leslie & Caroline, Mari & David. I will always cherish our moments together.

To the current and past members of the SHM group, thank you for all the moments and knowledge we shared.

Portions of this dissertation have been published or are currently being prepared for publication.

Chapter 1, in part, has been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., & Tur, M. , 2023. The title of this paper is “The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application”. The dissertation author was the primary investigator and author of this paper.

Chapter 2, in part, has been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., Tur, M. , & Yehoshua, T.,2023. The title of this paper is “The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application”. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in part, has been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., Tur, M. , & Yehoshua, T.,2023. The title of this paper is “The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application”. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in part, has been published in *Data Science in Engineering, Volume 9. Conference Proceedings of the Society for Experimental Mechanics Series.*, Razzini, A.H., Kressel, I., Ofir, Y., Tur, M., Yehoshua, T., & Todd, M.D., 2023. The title of this paper is “Development of a Surrogate Model for Structural Health Monitoring of a UAV Wing Spar”. The dissertation author was the primary investigator and author of this paper.

Chapters 4 and 5, in part, have been published in *Data Science in Engineering, Volume 10. SEM 2023. Conference Proceedings of the Society for Experimental Mechanics Series*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., & Tur, M. , 2023. The title of this paper is “Optimal Fiber-Optic Sensor Placement Framework for Structural Health Monitoring of an Aircraft’s Wing Spar”. The dissertation author was the primary investigator and author of this paper.

Chapters 4 and 5, in part, have been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., Tur, M. , & Yehoshua, T., 2023. The title of this paper is “The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application”. The dissertation author was the primary investigator and author of this paper.

The author acknowledges financial support from the Israel Ministry of Defense under contract award #4440850252. Funding for this project is greatly appreciated.

VITA

- 2015 Bachelor of Science in Mechanical Engineering, Universidade Estadual Paulista (UNESP), Ilha Solteira, Brazil
- 2017 Master of Science in Mechanical Engineering, Universidade Estadual Paulista (UNESP), Ilha Solteira, Brazil
- 2021 Master of Science in Structural Engineering with Specialization in Structural Health Monitoring & Non-Destructive Evaluation, University of California San Diego
- 2024 Doctor of Philosophy in Structural Engineering, University of California San Diego

PUBLICATIONS

Razzini, A.H., Todd, M.D., Kressel, I., Ofir, Y., Tur, M. (2023). The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application. *Proceedings of the International Workshop on Structural Health Monitoring 2023*. 10.12783/shm2023/36728

Razzini, A.H., Todd, M.D., Kressel, I., Ofir, Y., Tur, M. (2023). Optimal Fiber-Optic Sensor Placement Framework for Structural Health Monitoring of an Aircraft's Wing Spar. In: Madarshahian, R., Hemez, F. (eds) *Data Science in Engineering, Volume 10. SEM 2023. Conference Proceedings of the Society for Experimental Mechanics Series*. Springer, Cham. https://doi.org/10.1007/978-3-031-34946-1_16

Razzini, A.H., Todd, M.D., Kressel, I., Ofir, Y., Tur, M., Yehoshua, T. (2023). Damage Assessment of an Aircraft's Wing Spar Using Gaussian Process Regressors. In: Rizzo, P., Milazzo, A. (eds) *European Workshop on Structural Health Monitoring. EW-SHM 2022. Lecture Notes in Civil Engineering, vol 270*. Springer, Cham. https://doi.org/10.1007/978-3-031-07322-9_40

Razzini, A.H., Kressel, I., Ofir, Y., Tur, M., Yehoshua, T., Todd, M.D. (2022). Development of a Surrogate Model for Structural Health Monitoring of a UAV Wing Spar. In: Madarshahian, R., Hemez, F. (eds) *Data Science in Engineering, Volume 9. Conference Proceedings of the Society for Experimental Mechanics Series*. Springer, Cham. https://doi.org/10.1007/978-3-031-04122-8_12

FIELDS OF STUDY

Major Field: Structural Engineering

Studies in Structural Dynamics
Professor Michael Todd

Studies in Structural Health Monitoring
Professors Charles Farrar and Francesco Lanza di Scalea

Studies in Signal Processing
Professor William Hodgkiss

Studies in Finite Elements Modeling
Professors Hyonny Kim, Petr Krysl, and Yuri Bazilevs

ABSTRACT OF THE DISSERTATION

Development of an Optimal Sensor Placement Framework for Structural Health
Monitoring of an Aircraft Wing's Spar

by

Adrielly Hokama Razzini

Doctor of Philosophy in Structural Engineering

University of California San Diego, 2024

Professor Michael D. Todd, Chair

The performance gains achieved by increasingly wide adoption of composite materials in the aerospace sector remains challenged by failure modes that are not always well understood. Implementing a structural health monitoring (SHM) system enables real-time monitoring of a structure, allowing for continuous assessment of its current condition. This work proposes an comprehensive optimal fiber optic sensor placement framework for structural health monitoring applications. The framework is applied to an aircraft's wing spar entirely made of composite materials. The damage of interest is debonding between laminates, which may cause local buckling that leads to reduced structural load carrying

capabilities. The development and validation of a high-fidelity finite element (FE) model that is used as a synthetic data generator are presented in detail. The inputs to the model are loads and debonding damage parameters (size and location), and the outputs are uniaxial strain measurements and buckling eigenvalues. Then, this work describes “run time” surrogate models created using different machine learning methods to overcome the high computational costs of each run of the physics-based model.

Bayesian inference is used to estimate the damage parameters given strain measured at candidate sensor locations. These estimations are used to assess damage criticality, which is linked to buckling eigenvalues, and transformed into decisions. Bayesian optimization is used to select the candidates that minimize a utility function that considers the costs associated with making a certain decision plus the costs of acquiring and installing the SHM hardware (sensors, data acquisition system, etc.). The goal of the optimization is to find the sensor set that provides the lowest cost. The resulting optimal sensor configuration is presented, consisting of the number of sensors to be deployed and their respective locations. Finally, this work analyzes the performance achieved by several different cost functions, highlighting the importance of defining an objective function that reflects the goal of the SHM system, demonstrated in an aerospace application.

Chapter 1

Introduction

1.1 Motivation

To enhance the performance of commercial and military aircraft, the aerospace industry is constantly driving the development of improved high-performance structural materials [5]. Among these materials are the structural composites - a material system consisting of two or more phases on a macroscopic scale, whose mechanical performance and properties are designed to be superior to those of the constituent materials acting independently [6].

Carbon fiber-reinforced polymers (CFRP) are known for their strength-to-density and stiffness-to-density ratios, allowing for weight savings in the range of 20 to 50% [1]. Other benefits of CFRP composites include their resistance to corrosion, temperature stability - meaning that they do not expand or contract excessively with rapid change in environmental temperatures, ability to build into complex shapes, and the tailoring of mechanical properties along preferential stress directions [1, 7, 8].

Figure 1.1 shows the continuous growth in usage of composite materials in commercial aviation over several decades. Two classic examples are the Boeing 787 and the Airbus A350 XWB. The Boeing 787 aircraft is 80% composite by volume. By weight, the material contents are 50% composite, 20% aluminum, 15% titanium, 10% steel, and 5% other materials [9]. The Airbus A350 contains 54% composite materials by weight [10].

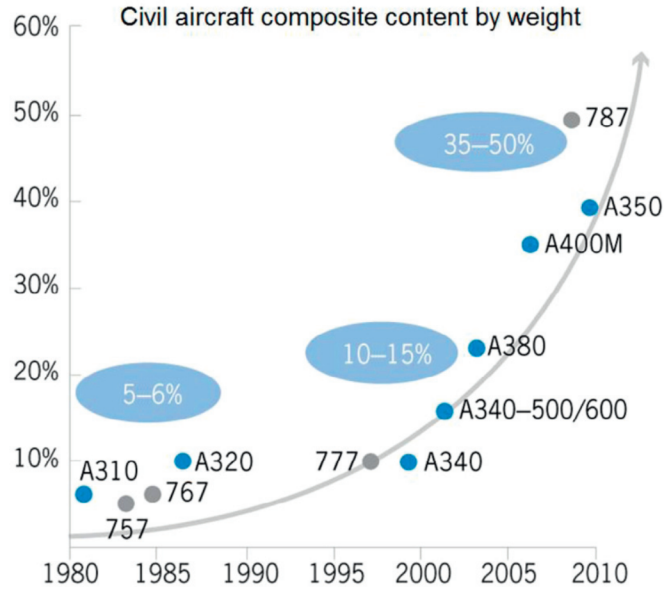


Figure 1.1. Increase of the weight content of composites in aircraft structures over a 30-year time span [1]. “A” represents Airbus aircraft, while “7” represents Boeing’s 700 series.

Despite the previously mentioned benefits and performance gains, the wide adoption of composite materials in the aerospace sector remains challenged by failure modes that are not always well understood [1]. An in-depth review of common failure modes in composite materials can be found in [11]. Such materials may suffer internal damage without any obvious changes on the surface, negating the value of visual inspections [12, 13].

One of the main concerns in composite structures composed of different parts is the debonding damage, caused by the loss of adhesion between two laminate parts. This debonding may be introduced during service or caused by manufacturing imperfections, such as improper curing, overloading, or moisture intrusion, among other factors [14, 15, 16].

Such damage can severely degrade the mechanical properties of the composite structures, and to meet aviation regulation requirements [17], any damage that would prevent the aircraft from carrying its ultimate service load must be detected [17]. In conventional periodic inspections, inspections are limited by accessibility to critical components and there is always a chance of missing growing damage in between the scheduled

maintenance [18]. Implementing a structural health monitoring (SHM) system may help optimal decisions and a maintenance schedule that depends on the current state of the structure [19].

Due to their performance requirements (high strength – low weight), Unmanned Aerial Vehicles (UAVs) have largely incorporated composite materials in their structures [20]. These aircraft have high maneuverability and are often subjected to extreme loading and environmental conditions, thus the desire to continuously monitor their structural integrity to ensure safe and reliable operations is clear. This work focuses on the implementation of an SHM system to monitor debonding damage on an all-composite wing’s spar.

1.2 Structural Health Monitoring

Structural health monitoring is the process of implementing a damage identification strategy for aerospace, civil, and mechanical engineering infrastructure. It involves observing a structure or mechanical system over time using periodically spaced measurements, extracting damage-sensitive features from these measurements, and statistically analyzing these features to determine the current state of the system’s health [21, 22]. Typically, SHM is associated with online, real-time damage identification in structural systems.

Structural health monitoring and non-destructive evaluation (NDE) [23] are related techniques used to assess the condition of structures, but they differ in a few key aspects. SHM is primarily used for continuous or periodic monitoring of structures to assess their health and detect damage over time. NDE, on the other hand, is typically used for a one-time inspection to detect specific defects or damage. SHM involves the use of sensors installed on or within the structure to continuously or periodically collect data on its condition. NDE techniques, such as infrared thermography [24, 25], radiography [26], and eddy current testing [27], require that operations are put on pause for inspection of

the structure. Finally, SHM is often more sensitive to small, incremental changes in the structure's condition over time, making it suitable for detecting early-stage damage. NDE techniques are more focused on detecting specific defects or damage that may be present at the time of inspection.

It has become standard to define the structural health monitoring process in terms of a four-step statistical pattern recognition paradigm [21, 22]. This following four-step process includes:

1. Operational evaluation. It is the initial step in implementing a damage identification capability through SHM. It involves assessing the economic and life-safety justifications for SHM, defining the concept of damage for the system, understanding the operational and environmental conditions, and determining data acquisition limitations. This evaluation helps tailor the SHM process to the specific features and unique aspects of the system being monitored.
2. Data acquisition, normalization, and cleansing. This step involves selecting the sensor types, locations, and data acquisition methods, considering economic factors and the interval at which data should be collected. Data normalization is essential to separate changes in sensor readings caused by damage from those caused by operational and environmental variations. Data cleansing involves selectively choosing or rejecting data based on the knowledge gained during data acquisition, using techniques such as filtering and resampling.
3. Feature extraction and information condensation. This step focuses on identifying data features that distinguish between undamaged and damaged structures. Methods include correlating system response quantities with observations, applying engineered flaws to understand parameters sensitive to damage, and using computational tools like finite element models. This process involves condensing the data to retain

sensitivity to structural changes while minimizing environmental and operational variability.

4. Statistical model development. This is the final step, involving the development of algorithms that quantify the damage state based on the extracted features. These algorithms fall into supervised or unsupervised learning categories, analyzing statistical distributions to enhance the damage identification process. The models address questions such as the existence, location, type, extent, and prognosis of damage, aiming to minimize false indications of damage and enhance safety and reliability.

Data obtained from an SHM system facilitate informed decision-making regarding structural maintenance, repair, and retrofitting activities [19, 28]. By providing real-time or periodic updates on the structural condition, SHM systems empower asset managers and engineers to prioritize and plan interventions effectively, minimizing downtime and maximizing operational efficiency. Therefore, the SHM problem can be treated as one of decision-making rather than one of damage identification, and the goal of the SHM system cannot be limited to making the most accurate decision, but also to make the safest and most economical ones.

This dissertation proposes looking at the four-step statistical pattern recognition paradigm listed above in an optimal sensor placement context.

1.2.1 Optimal sensor placement

Optimal sensor placement (OSP) in the context of structural health monitoring (SHM) refers to the strategic positioning of sensors on or within a structure to maximize the effectiveness of damage detection, localization, and characterization. The goal of the OSP reflects the goal of the SHM system, which usually is to ensure that the selected sensor locations provide the most informative data regarding the structural health and

integrity while minimizing the number of sensors needed and associated costs.

Lengthy reviews of existing literature in this area have been produced [29, 30]. In summary, there are three main advantages to performing an optimal sensor placement study before implementing an SHM system:

- **Enhanced probability of detection:** Properly placed sensors can increase probability of detection of the damage of interest, including cracks [31, 32], fatigue induced damage [33], impacts that may lead to barely visible damage in composites [34], and composites delamination [35], at an early stage. This allows for timely intervention and maintenance, preventing potential structural failures and ensuring the safety and reliability of the structure.
- **Accurate Localization of Damage:** Strategic sensor placement enables accurate localization of damage within the structure. By capturing data from critical locations, such as areas prone to high stress [36] or vulnerable to specific damage mechanisms [37], the monitoring system can pinpoint the location of damage, facilitating targeted repair or reinforcement efforts.
- **Optimization of Resources:** Optimal sensor placement helps optimize the allocation of resources by minimizing the number of sensors required while maximizing their effectiveness [38, 39]. This reduces the overall cost of the monitoring system, including installation, maintenance, and data processing, making SHM more feasible for large-scale or budget-constrained projects.

1.3 Proposed optimal sensor placement framework

The optimal sensor placement framework proposed in this dissertation consists of six steps. An introduction to each of them will be provided on the upcoming subsections, demonstrated on an aircraft SHM problem.

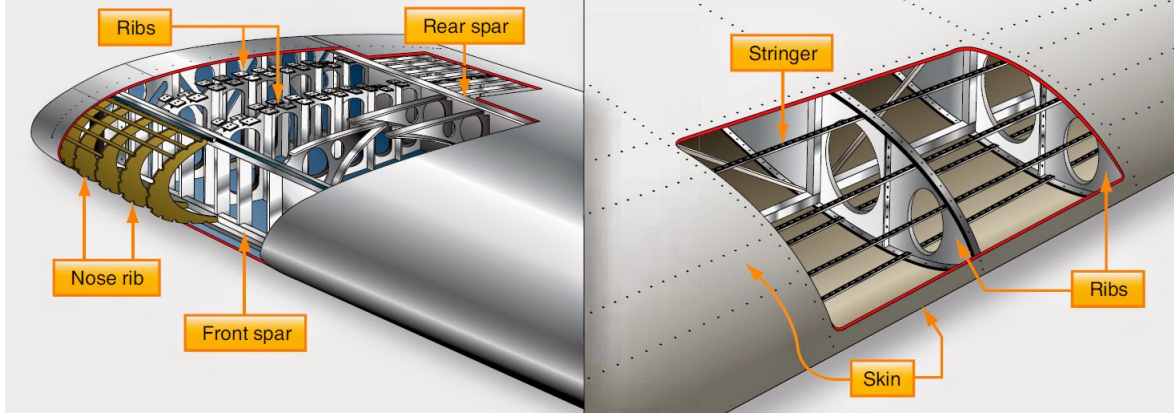


Figure 1.2. Main components of an aircraft's wing. Image obtained from [2].

1.3.1 Define the structure to be monitored and damage of interest

This work focuses on the monitoring of an unmanned aerial vehicle wing's spar. A wing spar is a structural component of an aircraft wing that runs from the fuselage to the wingtip and carries the majority of the wing's load during flight[40]. Spars are typically made of high-strength materials such as aluminum or carbon fiber and provide the necessary strength and stiffness to support the weight of the aircraft and withstand aerodynamic forces during flight. The spar also serves as a attachment point for other wing components such as ribs, skins, and fuel tanks [40]. Figure 1.2 shows the main components of an aircraft's wing. Although there are two spars in the figure, the aircraft from this study contains a single spar, also known as mono-spar.

Recently, to guarantee the safe operation of UAVs, particularly over populated regions, the industry has shifted towards setting airworthiness standards, such as the STANAG 4671 [17], departing from regulations designed for airworthiness of commercial manned aircraft. According to these standards, any damage that would prevent the aircraft from carrying its ultimate service load must be detected [17], and this is the case with debonding damage that may lead to local buckling.

In this work, the spar is 5m long, shaped as an I-beam entirely made of carbon

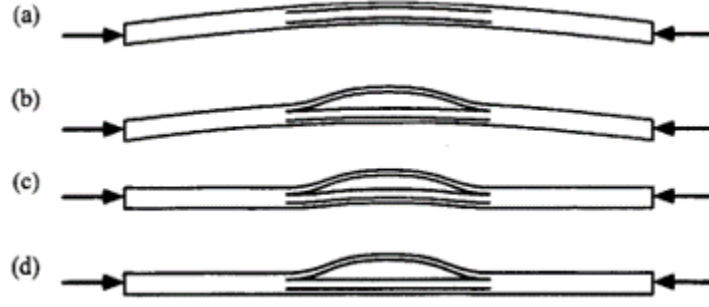


Figure 1.3. Cross sectional views of possible buckling mode shapes for delaminated plate: (a) global buckling; (b) mixed global and local deformation; (c) local buckling only of sublaminates; (d) local buckling of single sublaminates. Image obtained from [3].

fiber reinforced polymer composites, with a sandwich panel web. An in-depth discussion of the structure is provided in Chapter 2. The damage of interest is debonding between the top cap and the web, which may be caused by manufacturing imperfections or introduced during service. When the top cap is under compression, this debonding damage may cause local buckling, preventing the aircraft from carrying its ultimate service load. If this occurs, the damage is considered critical to the structure. This behavior was simulated in a finite element analysis, and it is governed by the buckling eigenvalue, λ . Given that

$$P_{critical} = \lambda \times P_{applied} \quad (1.1)$$

where F are the loads, any damage that yields $\lambda < 1$ for the applied loads are considered critical.

Figure 1.3, obtained from [3] provides a visual explanation of the difference between global (Figure 1.3 (a)) and local (Figure 1.3 (c,d)) buckling. A delamination is simply a debonding or separation that occurs between individual plies of the laminate during manufacture or service [41], this image also serves to illustrate the effects of debonding between two laminates, that may lead to local buckling.

1.3.2 Define the type of sensor(s) to be used

In aerospace applications where weight is a critical factor, the number of sensors, data acquisition systems and power supplies available is limited. A wide range of embedded and attached sensors have been studied for SHM applications, including strain gauges, accelerometers, fiber optic sensors, active ultrasonic sensors, passive acoustic sensors, among others.

This work considers fiber optic sensors based on fiber Bragg gratings (FBG) being used. These sensors are lightweight and compact, can be embedded within composite materials, are immune to electromagnetic interference, and have high resolution and sensitivity to strain measurements, making them suitable for aerospace applications [42, 43, 44]. The use of these fibers embedded in composite materials been tested during in-flight applications [45] and in laboratory simulated in-flight conditions [46] in other studies.

A brief overview of FBG optical sensors

Optical fibers can be used as sensors to measure strain, temperature, pressure and other quantities by modifying a fiber so that the quantity to be measured modulates the intensity, phase, polarization, wavelength or transit time of light in the fiber. Fiber Bragg Grating sensors are a specific type of fiber optic sensor, typically created by exposing a section of an optical fiber to a pattern of intense ultraviolet light [47]. This creates a periodic variation in the refractive index of the core of the fiber, forming a grating structure.

When light is transmitted through the fiber, some of it is reflected back at the grating. The wavelength of light that is reflected depends on the spacing of the grating. This wavelength is called the Bragg wavelength and is given by the Bragg condition:

$$\lambda_B = 2n_{eff}\Lambda_0 \tag{1.2}$$

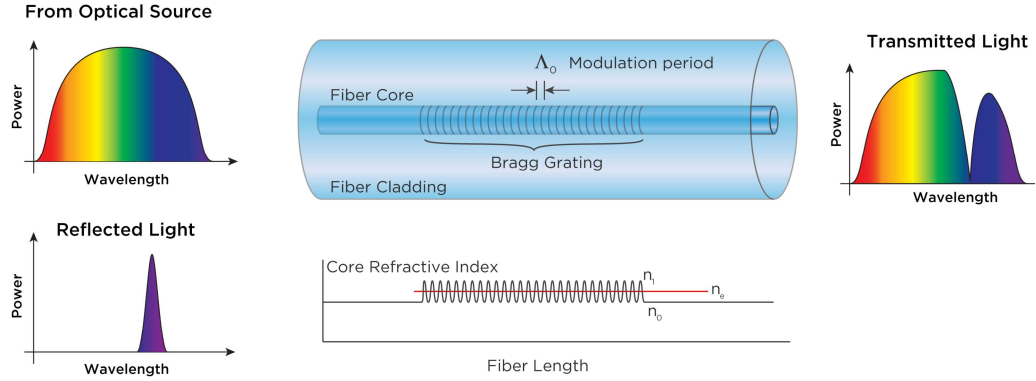


Figure 1.4. How fiber Bragg grating sensors work. Figure obtained from [4].

where λ_B is the Bragg wavelength, n_{eff} is the effective index of refraction of the FBG, and Λ_0 is the spatial period of the grating. Figure 1.4 presents the internal structure, and transmission and reflection spectra of an FBG.

When the FBG is subjected to an external stimulus (e.g., strain or temperature change), the grating experiences a change in its periodicity or effective refractive index, leading to a shift in the reflected wavelength. By measuring this wavelength shift, the applied stimulus can be determined. The relative shift in the Bragg wavelength, $\Delta\lambda_B/\lambda_B$, due to an applied strain, ε , is approximately given by

$$\frac{\Delta\lambda_B}{\lambda_B} = (1 - \rho_e)\varepsilon + (\alpha_\Lambda + \alpha_n)\Delta T \quad (1.3)$$

where ρ_e is the effective photo-elastic constant of the core material in the fiber, α_Λ is the thermal expansion coefficient of the optical fiber, and α_n is the thermo-optic coefficient [43].

To measure the reflected wavelength, an optical interrogator is used. This device sends a broadband light source through the fiber and analyzes the spectrum of the reflected light. The shift in the reflected wavelength is proportional to the applied stimulus, allowing

for accurate measurement.

1.3.3 Data acquisition and/or synthetic data generation

The third step of the framework is acquiring data from the structure, under different environmental, operational, and damage conditions. In many cases, these data can be almost impossible to acquire, especially if damage is irreversible. In this work, the lack of directly available field data was mitigated using a high-fidelity finite element model, that will be described in Section 2.1. This model takes in the damage parameters and loads as inputs, and provides buckling eigenvalues and strain measurements as outputs.

To make a probabilistic assessment of the problem, the model needs to be executed hundreds or even thousands of times over all the uncertain parameters. However, each execution of the FE model, for a single case of damage size and location, takes around 50 minutes to run.

Long run times inhibit the use of models in applications which require many model runs, such as integrated modeling , uncertainty analysis, sensitivity analysis, and inverse modeling. Slow run times also prevent models being used in real time, necessary for applications such as decision support [48]. In these situations, surrogate models or metamodels are used to replace expensive simulation models of engineering problems, and they are usually machine learning based [49, 50, 51, 19].

To overcome this challenge and to achieve enough Monte Carlo simulations with reasonable computational cost/time, two surrogate models were built for this dissertation work: a Gaussian process regressor (GPR) to predict eigenvalues, and an artificial neural network (ANN) to reconstruct strain measurements, and they are explained in detail in Chapter 3.

1.3.4 Develop the damage diagnosis model

Once enough data is obtained or generated, the next step in the framework is developing a diagnosis model that is able to distinguish changes in sensor readings caused by damage from those caused by operational and environmental variations. In this study, the damage diagnosis step is performed via Bayesian inference.

A brief introduction to Bayesian inference

Bayesian statistical methods use Bayes' theorem to compute and update probabilities after obtaining new data. Bayes' theorem describes the conditional probability of an event based on data as well as prior information or beliefs about the event or conditions related to the event.

In structural health monitoring, Bayesian inference can be used to update the probability distribution of the health condition (usually defined by damage parameters, θ) of a structure based on sensor measurements. The damage parameters can be anything related to the problem of interest, such as the size and location of a debonding damage, the size of a crack, and so on. This approach allows for more accurate and reliable predictions of the structure's health compared to traditional methods.

A simplified example of how Bayesian inference can be applied to estimate the damage parameters of a structure is detailed below:

1. Define the Prior Distribution: Start with a prior distribution for θ , based on prior knowledge or historical data. This distribution represents the initial belief about θ before observing any sensor data.
2. Update with Sensor Data: As sensor data becomes available, use Bayes' theorem to update the prior distribution with the likelihood of observing the sensor data given

θ . This is done using Bayes' rule:

$$P(\theta|sensordata) = \frac{P(sensordata|\theta) \times P(\theta)}{P(sensordata)} \quad (1.4)$$

where $P(\theta|sensordata)$ is the posterior distribution of the RUL given the sensor data, $P(sensordata|\theta)$ is the likelihood of observing the sensor data given the RUL, $P(\theta)$ is the prior distribution of the RUL, and $P(sensordata)$ is the marginal likelihood of the sensor data, which acts as a normalizing constant.

3. Update Continuously: As new sensor data becomes available, continuously update the posterior distribution to refine the estimate of the damage parameters.

Bayesian inference provides a principled way to incorporate uncertainty into the damage parameters estimation process and allows for a more robust assessment of the structure's health condition.

1.4 Define the cost function

Once the damage detector is defined, the next step in the OSP framework is defining how the performance of a given sensor set will be evaluated. This is defined as the cost function, $\hat{C}(d)$. The first part of this cost is comprised of the cost of decision, C_{decision} . Once the damage state is inferred, the SHM system must decide what to do with that information. These decisions could simply be “damage is present” versus “damage is not present”, which carry implied actions, such as “stop operations for repair” or “continue operation”.

Let C_{ij} be the cost of deciding that the structural state is M_i when the true state is M_j . In SHM, however, not all damage states are necessarily treated equally. Certain damage states may be more probable or more costly if missed, requiring weighted consideration in the cost function. The goal of an SHM system is not necessarily to

make the most accurate decisions, but rather to make the least costly ones in terms of economics and safety. In an aerospace applications, false negatives usually have a higher cost associated with them [32]. However, a high probability of false positives causes unnecessary grounding of the aircraft, and the costs of unscheduled downtime can also be high. For a binary classification problem, the cost of decision can be defined as:

$$C_{\text{decision}} = \sum_{i,j=0}^1 C_{ij} P(D_i|M_j) P(M_j) \quad (1.5)$$

In addition, every SHM design incurs a cost that depends on its design parameters that drives the necessity of optimization. This cost can be the cost of the SHM hardware, the cost of installing and maintaining the SHM system, and even potential increase in operation costs of the structure (such as the extra weight in an aerospace system). In order for an SHM system to be feasible, its ability to decrease risk must be enough to at least offset the cost of the system. To construct a total cost function, the cost of action and the cost of design need to be in terms of the same units (usually monetary).

The design cost, C_{design} , can be defined for general applications as:

$$C_{\text{design}} = C_{\text{sensors}} + C_{\text{DAQ}} + C_{\text{ground station}} + C_{\text{installation}} \quad (1.6)$$

The total cost, or Bayes risk, is then given by:

$$\hat{C}(d) = C_{\text{decision}} + C_{\text{design}} \quad (1.7)$$

1.5 Define the optimization function and perform optimization

The last step in the proposed OSP framework is defining which optimization function to be used and actually running the optimization. Many algorithms have been

used for OSP, such as genetic algorithms [32, 52], gravitational search algorithms [53], particle swarm [54], Bayesian optimization [55, 56, 57], among others. An in depth review of these algorithms used for OSP for structural health monitoring applications can be found in [29].

This study uses Bayesian optimization to define the optimal sensor placement. Bayesian optimization is a sequential model-based optimization technique that is particularly useful for optimizing black-box functions, where the function’s analytical form is unknown or computationally expensive to evaluate. It builds a probabilistic model of the objective function and uses this model to decide where to sample next, aiming to find the global optimum with as few evaluations as possible.

The main idea behind Bayesian optimization is to use a probabilistic surrogate model (often a Gaussian process) to model the unknown objective function. This surrogate model is updated sequentially as new data points are observed, providing a probabilistic estimate of the objective function and its uncertainty. Based on this model, an acquisition function is used to determine the next point to evaluate. This point is chosen to balance exploration (sampling in regions where the objective function is uncertain) and exploitation (sampling in regions where the objective function is likely to be optimal).

1.6 Summary and dissertation organization

By framing the challenge of designing a structural health monitoring system in the context of a Bayesian experimental design, all elements of the SHM process can be integrated. The proposed optimal sensor placement framework allows the enhancement of an SHM system’s functionalities and communication of the system’s performance in a manner that resonates with the owner or operator of the monitored structural system. This approach is essential for widespread adoption of SHM technology in the industry.

Particular derivations, experiments, and findings with respect to this process will

be presented in the remaining chapters. Chapter 2 will detail the creation and validation of the finite element model. Chapter 3 will discuss the Python scripts created to accelerate the changes in the FE model, and the two forward surrogate models developed to serve as accurate "run-time" models. Chapter 4 discusses the Bayesian inference used for damage diagnosis and observability issues. Chapter 5 will describe the Bayesian optimization and will discuss the results given different costs functions and optimization strategies. Finally, Chapter 6 will provide the conclusions and future work.

This section, in part, has been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., & Tur, M. , 2023. The title of this paper is "The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application". The dissertation author was the primary investigator and author of this paper.

1.7 Contributions of the dissertation work

The significant contributions of this dissertation are summarized as follows:

1. The use of a multi-loading buckling approach to analyze the occurrence of local buckling in a composite wing's spar.
2. Development of an statistical framework that links strain measurements to damage parameters, buckling eigenvalues, and damage criticality.
3. Development of a high-fidelity finite element model of an all-composite wing's spar.
4. In-house construction and testing of a composite spar.
5. Development of two machine learning-based surrogate models with high prediction accuracy to overcome the high computational cost of the FE model.
6. Development and demonstration of an optimal sensor placement framework that uses Bayesian optimization as the algorithm of choice.

7. Analysis of different cost functions for the optimal sensor placement algorithm and their impact on the overall performance of the SHM system.

Chapter 2

Data generation - Development and validation of the FE model

2.1 The finite element model

In this work, a high-fidelity physics-based finite element model of a 5m long wing's spar (Figure 2.1) was developed in ABAQUS. Given the I-shaped geometry of the spar, the term I-beam will be used along the text interchangeably for the spar. The following sections will describe the model development in detail.

2.1.1 Geometry and material properties

The spar is composed of different elements that are manufactured separately and are assembled together using a structural epoxy adhesive layer. The nomenclature of each component of the spar is described in Figure 2.2.

The I-beam's dimensions were provided by the project's sponsors, and a sketch of a 1 meter subsection of the lateral view is shown in Figure 2.3, with all dimensions in mm. A sketch of the cross-section is shown in Figure 2.4. The I-beam has a total length of 5 m, so all the parts were extruded to meet this dimension.

A detailed sketch of the shear clips is provided in Figure 2.5 with units in m. Notice that the dimensions are slightly different from 2.4, as they were updated a few months later to provide a more realistic representation of the layup - the plies were extended



Figure 2.1. Composite wing's spar.

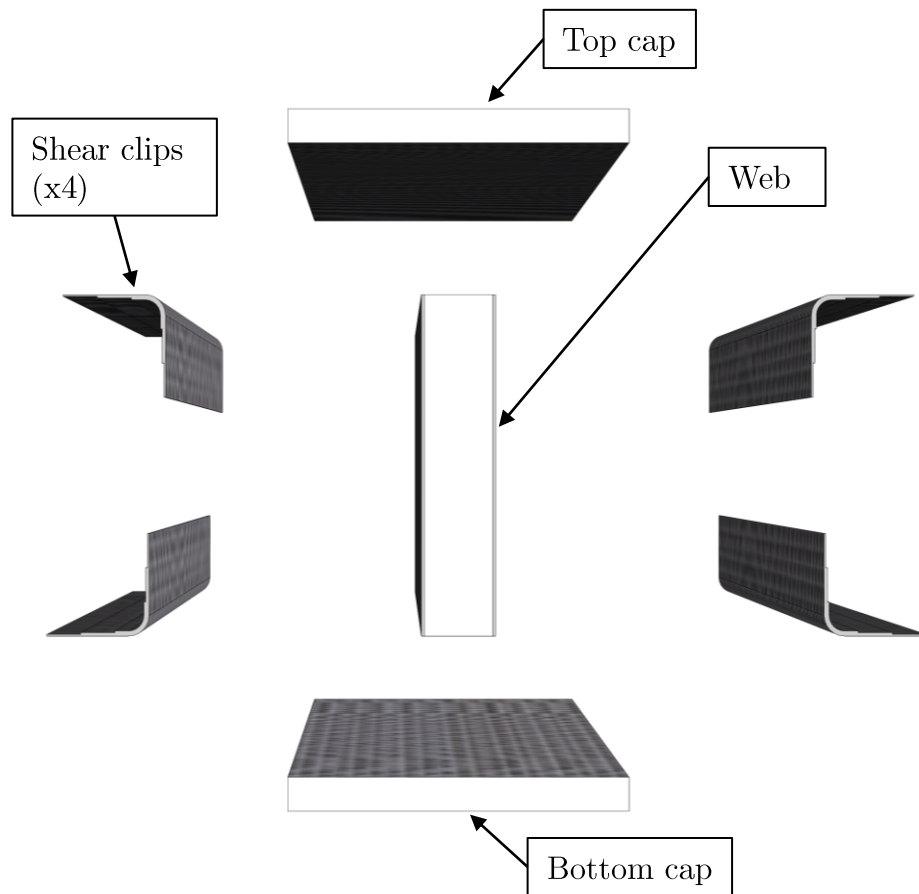


Figure 2.2. Nomenclature of the spar's components referenced in the dissertation.

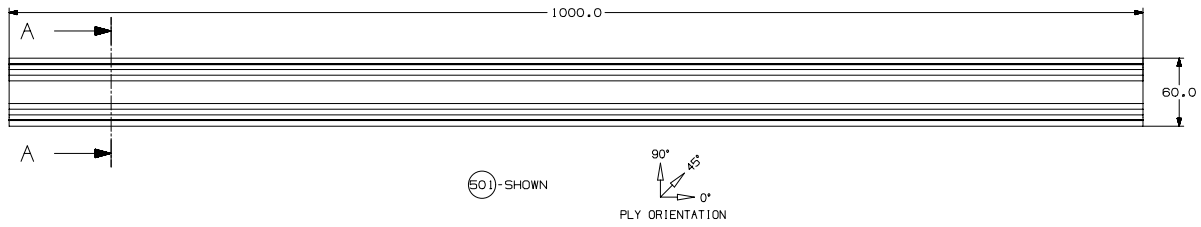


Figure 2.3. Drawing of the wing's spar (I beam) including composites ply orientation. This represents a 1m subsection of the 5m beam. All units in [mm].

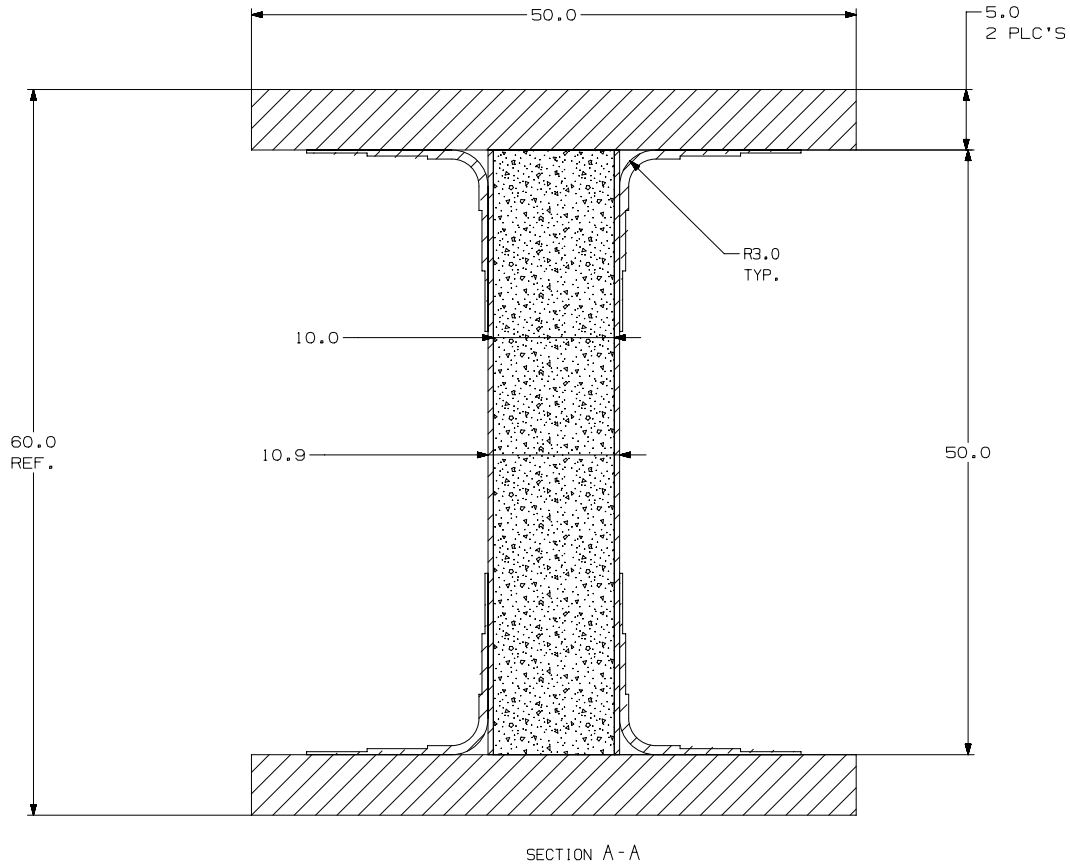


Figure 2.4. Section A-A from Figure 2.3. All units in [mm].

until the edges of the caps, with a constant thickness of 0.75 mm. For consistency, all dimensions were specified in Abaqus/CAE in m, and all other units followed the SI units, with the material properties being in Kg and Pa, the forces in N, moments in N.m, and pressure in Pa.

The material properties used for the finite element model are provided in Tables 2.1

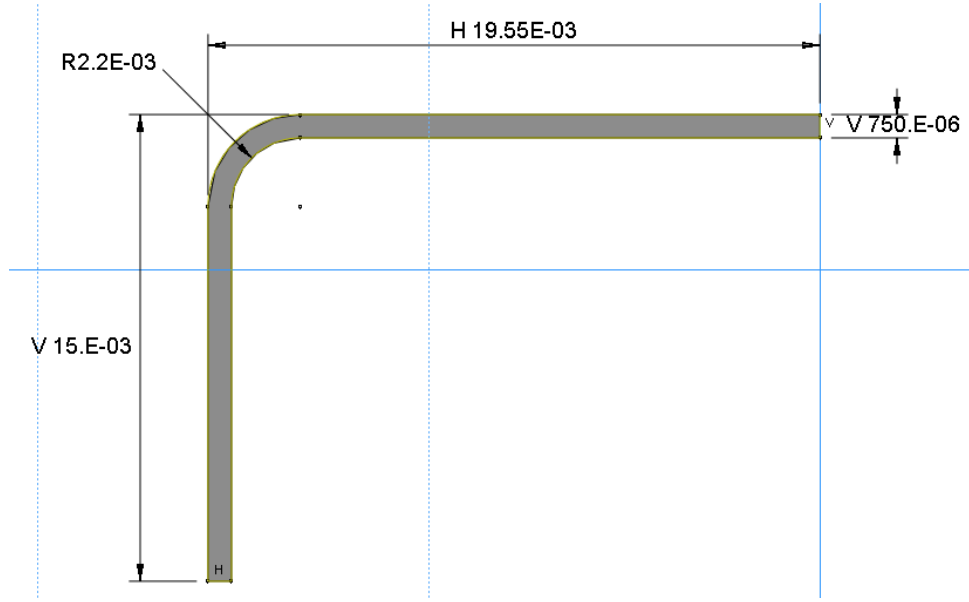


Figure 2.5. Dimensions of the shear clips. All units in [m].

and 2.2, where E , G , ν , and ρ represent the elastic modulus, shear modulus, Poisson's ratio, and density, respectively. The equivalent mechanical properties of the CFRP components are provided per each laminate, not per ply (lamina).

The caps and the web's skins were manufactured using prepreg carbon fiber, while the shear clips were manufactured using wet lay-up. The number of plies and stacking directions are proprietary of the sponsors of this project and will not be disclosed in this dissertation.

Table 2.1. Material properties of the low density foam used in the core.

Properties	Foam (core)
E	101 MPa
ν	0.262
G	40 MPa
ρ	75 kg/m ³

All the CFRP parts were assigned material orientations to be consistent with the notation provided in the drawings. For this purpose, local coordinate systems were created and for each part by itself the x-direction is along the length of the beam, the z-direction is

Table 2.2. Material properties of the CFRP components of the spar.

Properties	Caps	Web's skins	Shear Clips
E_{11}	85.5 <i>GPa</i>	13.7 <i>GPa</i>	33.35 <i>GPa</i>
E_{22}	29.5 <i>GPa</i>	13.7 <i>GPa</i>	33.35 <i>GPa</i>
E_{33}	9.9 <i>GPa</i>	9.9 <i>GPa</i>	9.9 <i>GPa</i>
ν_{12}	0.350	0.770	0.47
ν_{13}	0.302	0.062	0.060
ν_{23}	0.300	0.090	0.090
G_{12}	12.3 <i>GPa</i>	27.5 <i>GPa</i>	19.62 <i>GPa</i>
G_{13}	5 <i>GPa</i>	3.3 <i>GPa</i>	2.8 <i>GPa</i>
G_{23}	5 <i>GPa</i>	3.3 <i>GPa</i>	2.8 <i>GPa</i>
ρ	1600 <i>kg/m</i> ³	1490 <i>kg/m</i> ³	1530 <i>kg/m</i> ³

perpendicular to the surface (plies) and the y-direction completes the right-hand coordinate system. The full beam assembly and the local coordinate systems are shown in Figure 2.6.

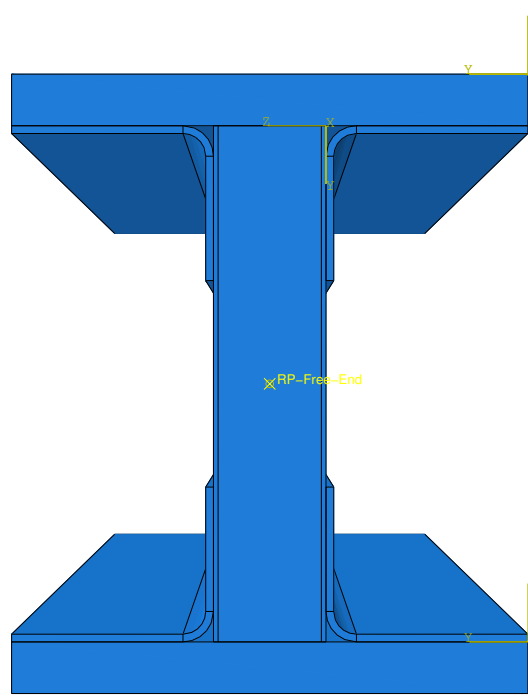


Figure 2.6. Full assembly of the spar created on Abaqus/CAE. The local coordinate systems are shown in yellow, where the x-direction is along the length of the beam, the z-direction is perpendicular to the surface and the y-direction completes the right-hand coordinate system.

2.1.2 The mesh

Initially, the spar caps, web's skins (outer part of the web), and foam (inner part of the web) were modeled with elements of the type C3D8R, which is a hexagon of the 3D stress family, 8-node linear brick, reduced integration. For these parts, the global seeds for the mesh were of $h=2\text{mm}$. The shear clips were modeled with 3D wedge elements (6-node linear triangular prism, called C3D6) and the global seeds for the mesh were of $h=1\text{mm}$. The previous meshed structure is shown in 2.7(a).

Then, a convergence study was performed to define the element types and sizes to be used on each part of the I-beam, to guarantee not only accurate results from the simulations, but also a good computational performance, therefore minimizing the duration of each simulation. After this study, it was decided to substitute the solid elements from the caps and shear clips by continuum shell elements. The sandwich core remained solid because of the thickness of the foam part.

In Figure 2.7(b) it is visible that the shear clips are no longer discretized by wedge elements, as the new meshes are composed of 8-nodes quadrilateral continuum shell (SC8R). These parts were partitioned in three sections to guarantee a structured mesh and to allow the refinement on the corners, to guarantee that no elements were warped. These partitions are artifacts to manipulate the meshes, and do not change the physical behavior of the structure.

Because of the change from solid to shell elements, it was possible to decrease the number of elements of the through-thickness direction of the caps, that are also SC8R elements. The discretization of the longitudinal direction was also analyzed. In that direction, the caps and the web have local seeds of $h = 2 \text{ mm}$, while the shear clips have $h = 1 \text{ mm}$.

Furthermore, a discrete material orientation was applied to guarantee that the plies of the shear clips would have the correct material orientation throughout the entire part,

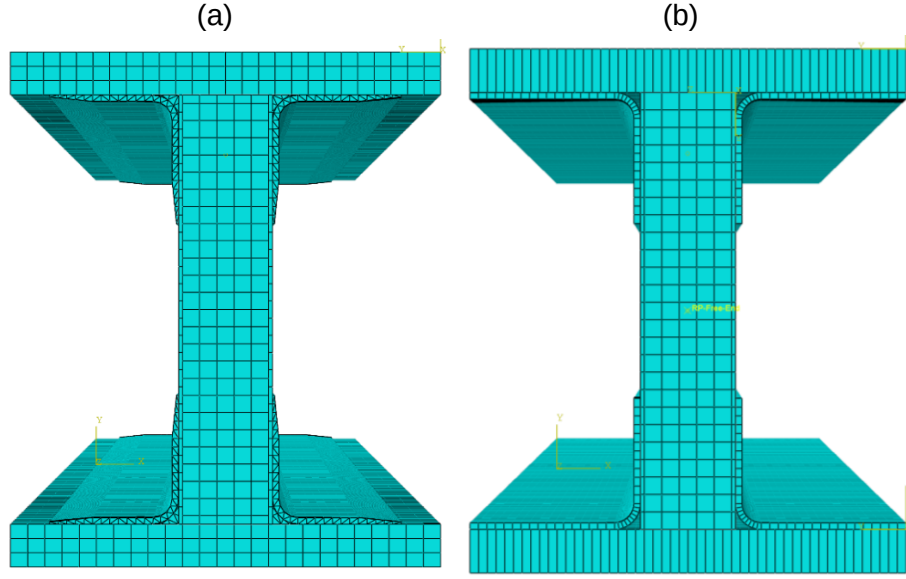


Figure 2.7. Comparison of initial mesh design (a) and updated mesh design (b). Solid elements were replaced by continuum shell elements (SC8R) on the caps and shear clips.

including the 90 degrees bend. All the composite parts have a stacking direction assigned to the meshes, to guarantee that they always follow the material orientation's z-direction.

2.1.3 Constraints

To emulate the effect of the structural epoxy, all the parts of the model were joined using tie constraints. These constraints connect the two selected surfaces in such a way that (as opposed to how contact works) they cannot separate or slide on each other throughout the analysis. A surface-based tie constraint can be used to make the translational and rotational motion as well as all other active degrees of freedom equal for a pair of surfaces. By default, nodes are tied only where the surfaces are close to one another. One surface in the constraint is designated to be the slave surface; the other surface is the master surface. [58]. In this model, the thicker parts (caps and core) were defined as the masters, and the shear clips were defined as slaves. The surfaces to which tie constraints are applied are highlighted in Figure 2.8.

The web of the beam is composed of foam in the middle and CFRP on the sides

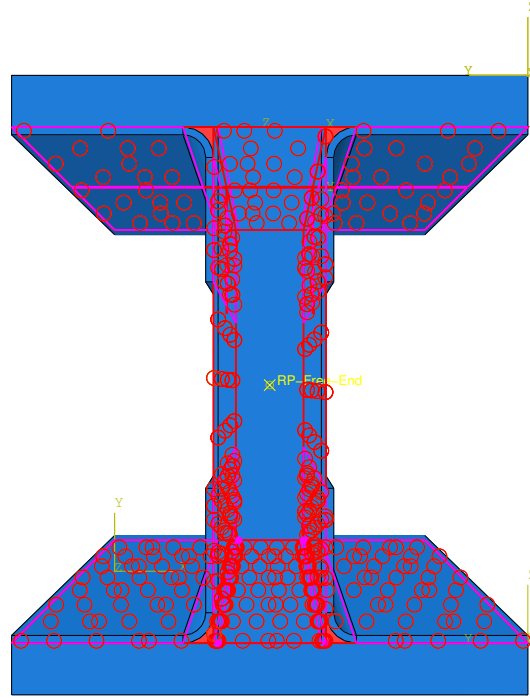


Figure 2.8. Tie constraints applied to the model to emulate the adhesive layer of structural epoxy that attaches all parts together.

(skins, or walls), as previously mentioned. To reduce the number of tie constraints necessary to put all the parts together, the core was created as a single part and divided in 3 partitions, with different sections assigned to them. The middle piece was assigned a solid, homogeneous section, with the “Foam” material properties defined in Table 2.1, while the side pieces were assigned a solid, homogeneous section, with the “Core walls” material properties defined in 2.2. The skins are anisotropic, and the material orientation was defined following the local coordinate system previously detailed (Figure 2.6). The foam is isotropic and does not require material orientations. Using this procedure to create the core, it is assumed that no debonding will occur between the CFRP skins and the foam.

2.1.4 Loads and boundary conditions

The boundary conditions were applied on the initial step. To apply the loads, a “static, linear perturbation” step was created, and the boundary conditions were propagated to this step. The model was defined as a cantilever beam. To create this boundary condition, all the master surfaces of the fixed end were selected, and a mechanical boundary condition of the type “symmetry/ anti-symmetry / encastré” was created, with all the degrees of freedom being set to zero (encastré condition). In addition, to emulate the effects of other wing components (ribs, stringers, and skin), it was defined that out-of-plane displacement should not occur. Thus, additional boundary conditions were applied to the side of the caps and core to restrain their out-of-plane displacement.

The loading distribution applied to the model was provided by the sponsors as the maximum service load, or limit load, and it is represented as follows (Figure 2.9):

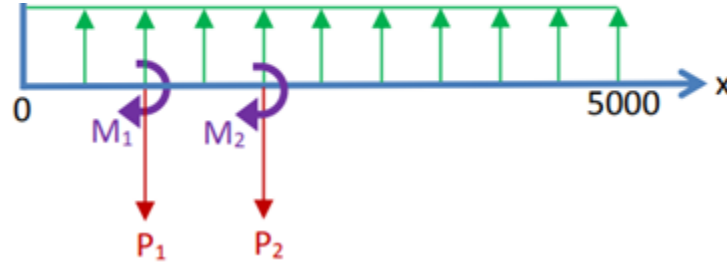


Figure 2.9. Loading scheme provided by the project’s sponsors, describing the maximum service load distribution on the beam. Units of the x-axis are in [mm].

- $q = +13729 \text{ Pa}$ is a uniformly distributed load, applied on the surface of the beam’s bottom cap, representing the sum of its self-weight and an outside pressure.
- $P1 = P2 = -490.3 \text{ N}$ are concentrated forces applied to the beam at points $x_1 = 1000 \text{ mm}$ and $x_2 = 2000 \text{ mm}$, respectively.
- $M1 = M2 = -1471 \text{ N.m}$ are two pure moments applied to the model, also at points x_1 and x_2 .

The concentrated forces and moments at the 1m and 2m cross-sections were implemented them using kinematic coupling constraints – a useful method to apply loads or boundary conditions to a model. Kinematic coupling constraints limit the motion of a group of nodes to the rigid body motion defined by a reference node [58].

In this model, the reference node was defined as the center of the cross-section, at 1m length of the beam (Figure 2.10(a)). After meshing the structure, all the master nodes of that cross-section region were selected (Figure 2.10(b)) and coupled to the reference node (Figure 2.10(c)). The concentrated force and pure moment were then applied to the reference node, which is the control point of the constraint. Then, the same procedure was repeated at the 2m cross-section. For both cases, all 6 degrees of freedom of each node were constrained. This procedure is shown in Figure 2.10. The implementation of the loads from Figure 2.9 applied to the model are shown in Figure 2.11.

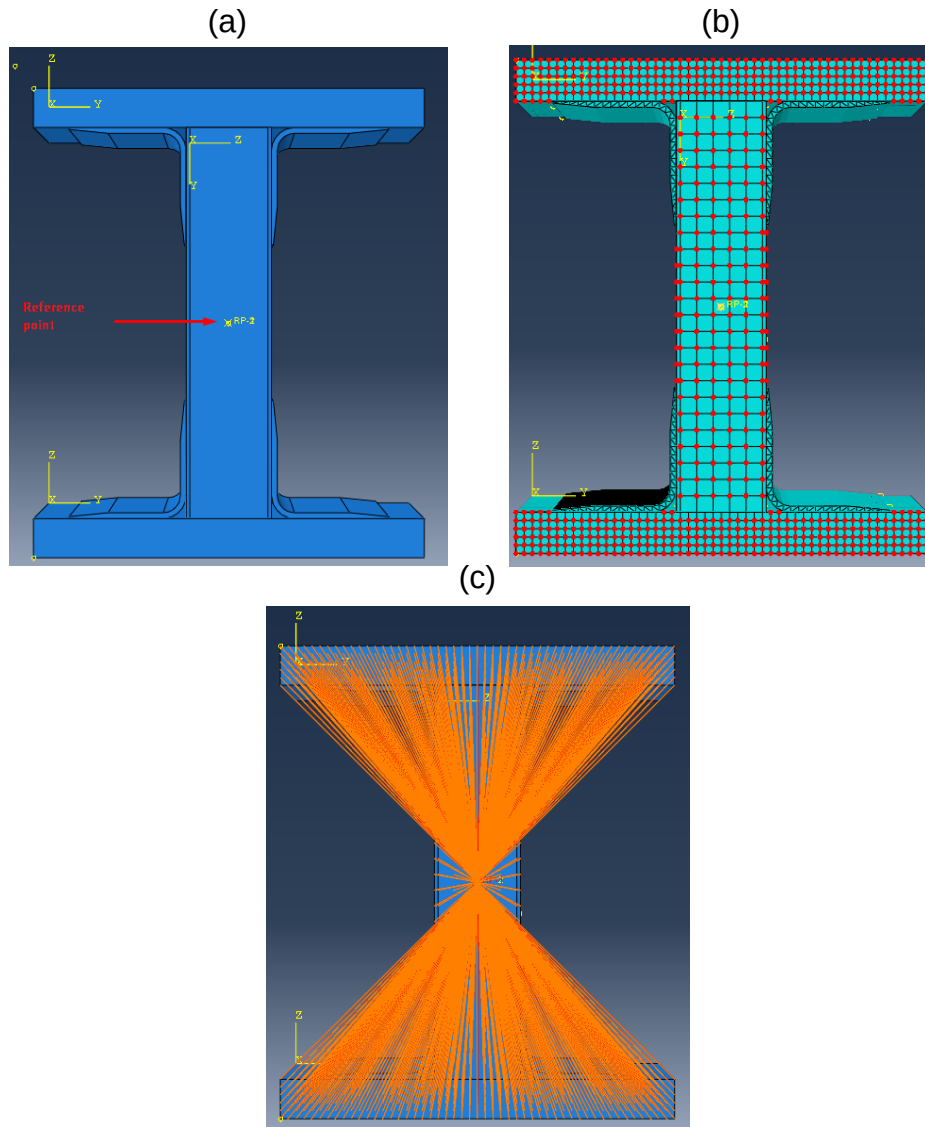


Figure 2.10. Definition of coupling constraint to apply loads to the structure. (a) shows the reference point, (b) shows the selection of the nodes that correspond to master structures from the tie constraints, and (c) shows the coupling constraint.

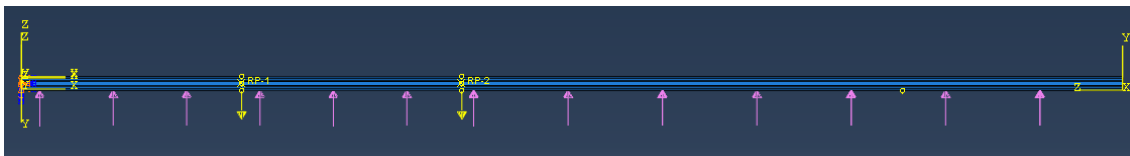


Figure 2.11. Loads applied to the spar.

2.1.5 Results from pristine beam

Initial results from the simulation of the pristine beam (damage-free) are shown below. To display the results, the averaging was turned off, and the results was transformed using a user-defined coordinate system, which is the same one defined for the spar cap's material orientation (1 direction: x-axis along the length of the beam, 2 direction: y-axis along the horizontal direction of the cross-section, and 3 direction: z-axis along the vertical direction of the cross-section).

The magnitude of the displacements is displayed in Figure 2.12, with units in m. The shape of the deflected beam shown in Figure 2.12 follows the expected behavior of a wing during flight. The maximum deflection at the end of the beam was estimated as 0.907m. Figures 2.13, 2.15, and 2.14 show the strain measurements in the different directions. The discontinuities on the strain measurements are clear at the sections where the forces and moments were applied.

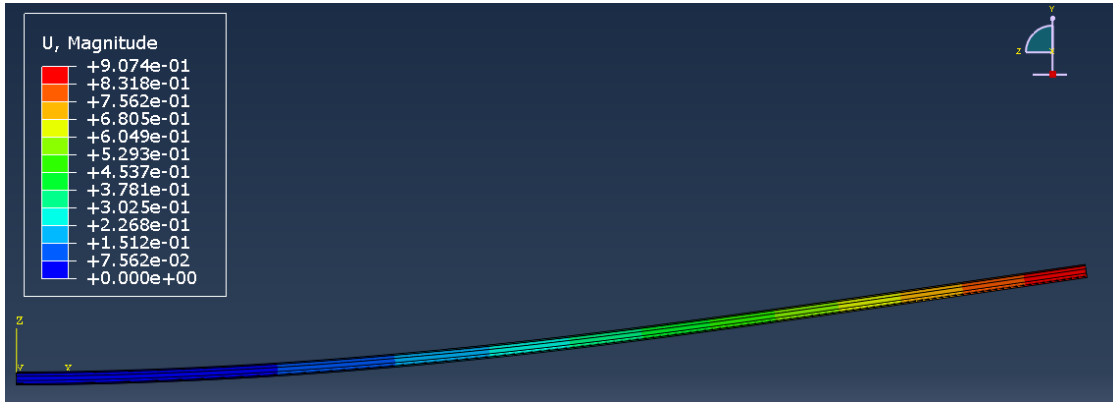


Figure 2.12. Magnitude of the displacements, units in m. The fixed end is on the left side of the spar.

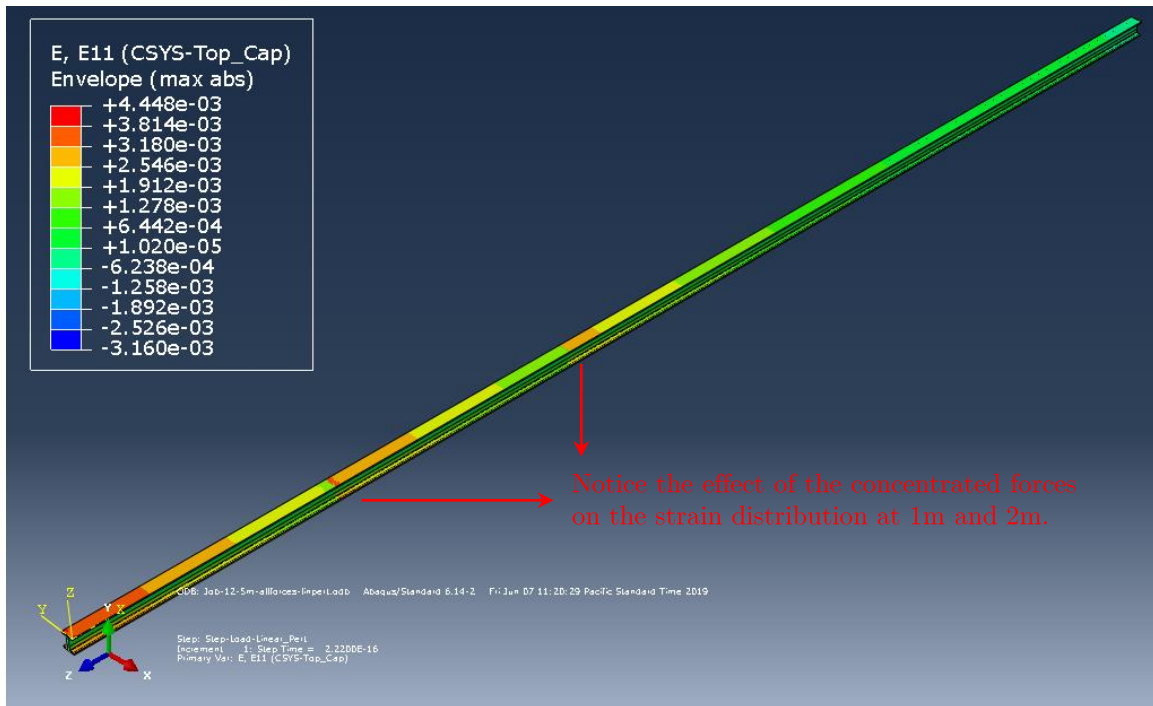


Figure 2.13. Strain in the 1 direction, following the spar cap's coordinate system (direction 1 is the axial direction, along the beam's length).

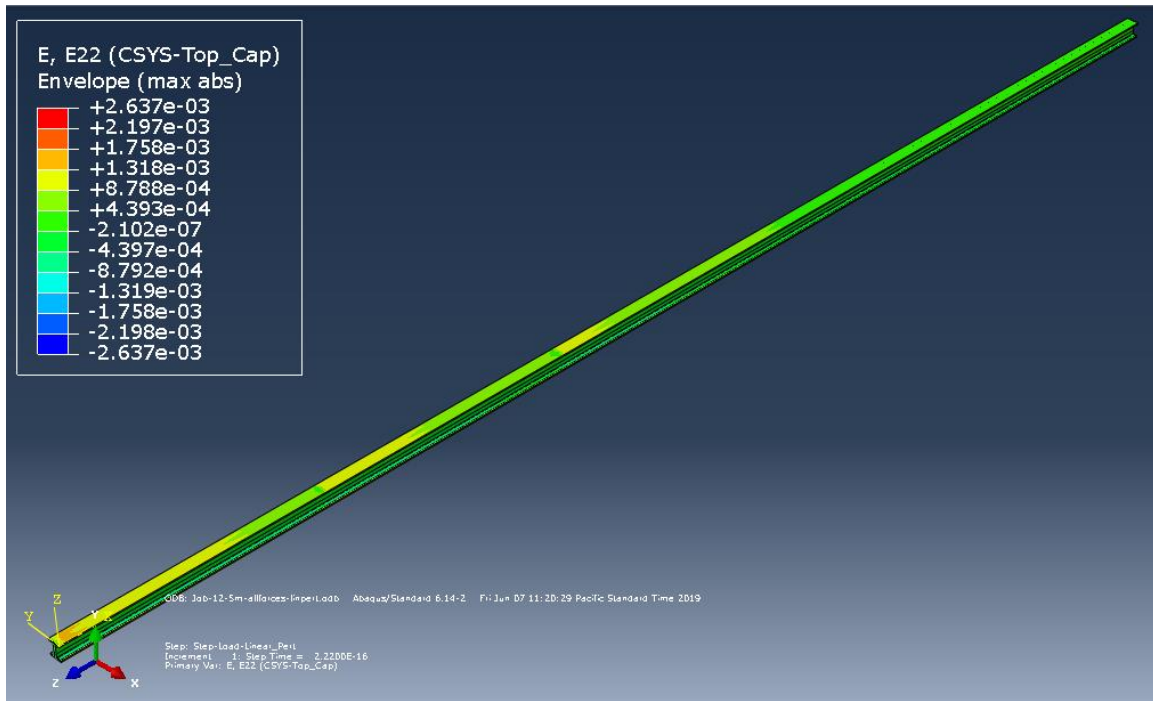


Figure 2.14. Strain in the 2 direction, following the spar cap's coordinate system.

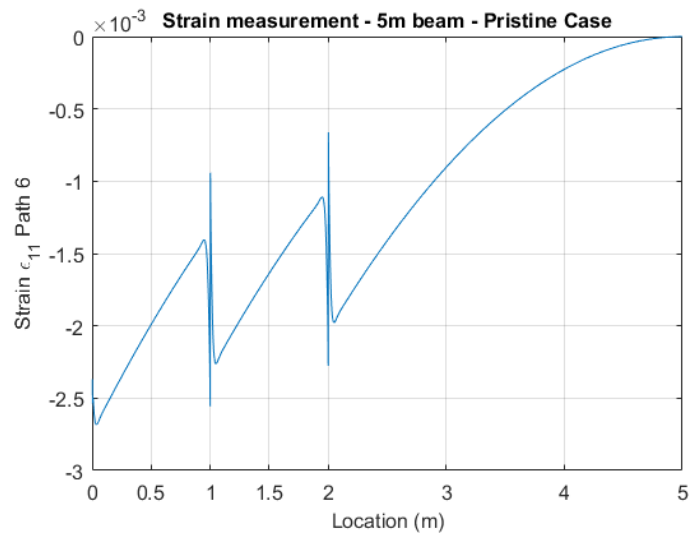


Figure 2.15. Strain measurement in the 1 direction at a path below the top-right shear clip.

2.1.6 Debonding damage and buckling

As mentioned in Chapter 1, the damage of interest in this study is the debonding of different parts (e.g., debonding between the shear clips and web and/or caps). This is a local buckling problem, not global buckling, given that the static bending loads cause compression on the upper half of the beam, and local buckling may occur depending on the size and location of the damage. Therefore, the focus should be on debonding between the top cap and the parts it is joint to (upper shear clips and web), not on the lower half of the beam.

The traditional Euler buckling theory does not represent the problem of interest, as it is wanted to investigate buckling under a complex loading scheme, so the multi-loading buckling approach must be used. In this case, the eigenvalues (λ) are interpreted as a multiplying factor to determine how the applied loads translate into a critical load:

$$P_{critical} = \lambda \times P_{applied} \quad (2.1)$$

Therefore, if $0 < \lambda < 1$, the structure will fail with a lower load distribution than the one applied. It should be noticed that if $\lambda < 0$, the structure will buckle with $-\lambda \times P_{applied}$.

The buckling analysis to collect eigenvalues and the static analysis to collect strain data are two different problems on Abaqus, so the job had to be divided into different, independent steps:

- Initial step: the boundary conditions are applied. They are propagated to all steps created afterwards. All analyses made on Abaqus must have an initial step.
- Static step, linear perturbation: all the forces are applied to the structure. Strain data is collected (stress and displacement could also be collected, if desired).
- Static step, buckle: all forces are applied to the structure again, as this step is

independent from the previous one, so the loads are not propagated to this step. Eigenvalues are collected.

One thing to notice is that the buckling step has a high computational cost, as it increased the total simulation time in about 30%. This, by itself, is a good reason to develop machine learning surrogate models that will be discussed later.

On Abaqus, a debonding damage can be simulated by removing the tie constraints between the desired parts. To confirm that this procedure works, two aluminum plates of random geometry were tied and a buckling analysis was performed - global buckling occurred. Next, the tie constraints were removed from a small portion in the center of the plates, the simulation was run again, and local buckling occurred, as shown in Figure 2.16.

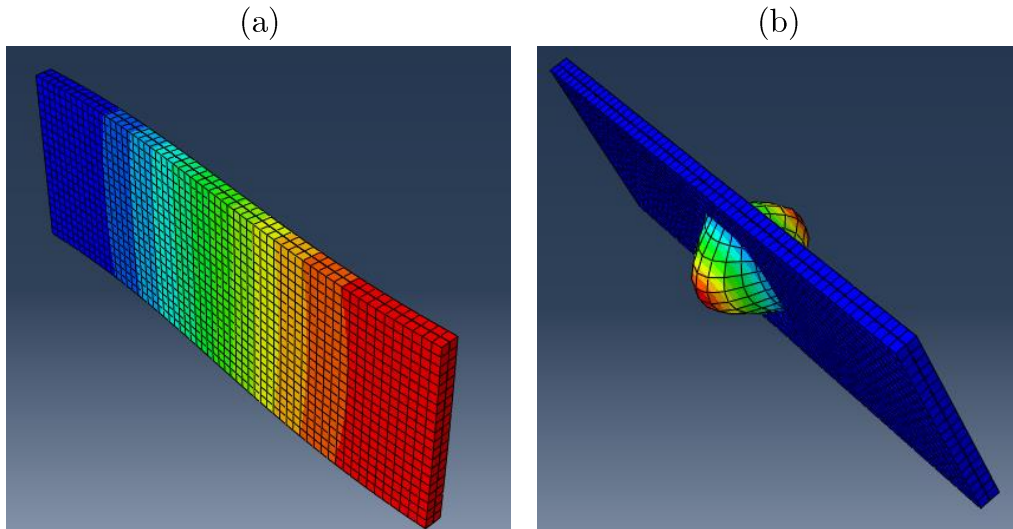


Figure 2.16. Buckling analysis of two plates of random geometry bonded together, used for proof of concept. On (a), global buckling occurs. On (b), as a small portion of the bonding is removed, local buckling occurs.

To apply that procedure to the spar, a partition of the defined damage size must be created at the defined location, on all faces of the shear clips that will be debonded from the web and from the cap. The same partition must be created on the surfaces of the web and the cap. An example of such partition is shown in Figure 2.17. Then, when

selecting the master and slave surfaces for the tie constraints, these regions where the partitions were created should not be selected – therefore, one part is not bonded to the other in those regions. This procedure can be visualized in Figure 2.20, where there is a debonding damage between the shear clips, web, and the top cap. In purple is the region where the tie constraints are applied, and the region in blue between them represents the damage. The top cap was suppressed from this image to make it possible to visualize the damage, as this damage occurs below the top cap.

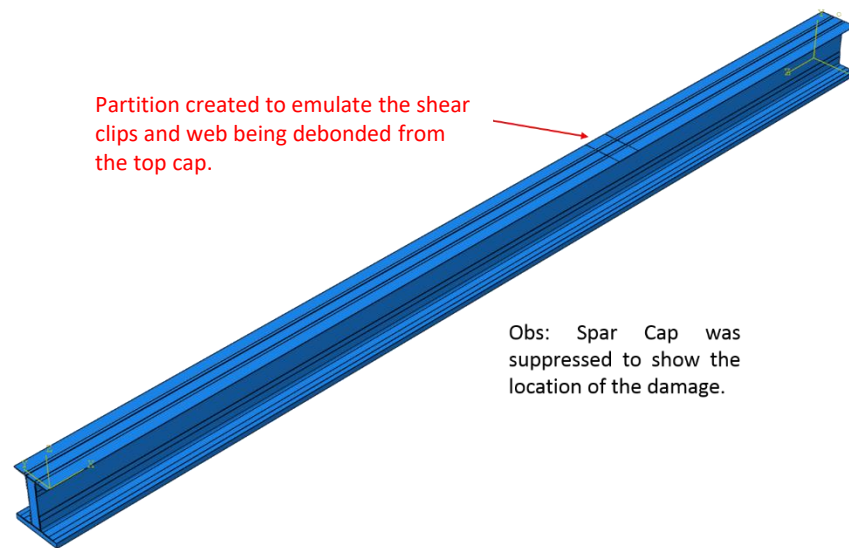


Figure 2.17. Partition created to emulate debonding between different parts.

To confirm that removing the tie constraints will emulate a debonding damage that leads to local buckling on the spar, the buckling step was simulated on Abaqus for the pristine beam. The result is shown in Figure 2.18, and it can be seen that global buckling occurred, as there is no damage on the beam. Next, a debonding damage was created following the procedure described above, and local buckling happened at the damage region, as shown in Figure 2.19.

As shown in Figure 2.20, the damage location is defined as the distance from where the debonding starts to the root of the beam. Then, damage size as defined as the length of the debonding. It is assumed that the entire width of the cap is debonded from the

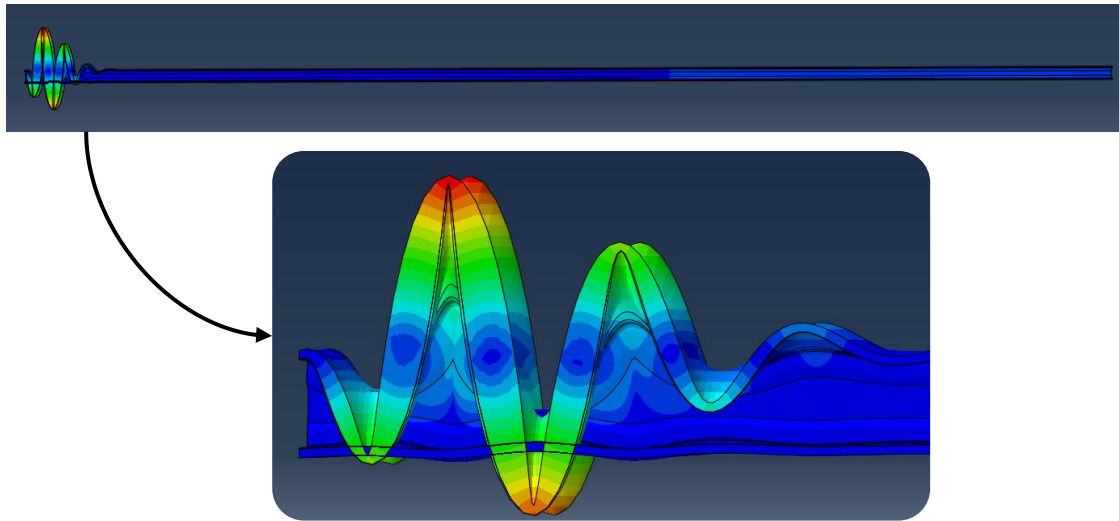


Figure 2.18. Buckling simulation of pristine beam. As there is no damage, global buckling occurs.

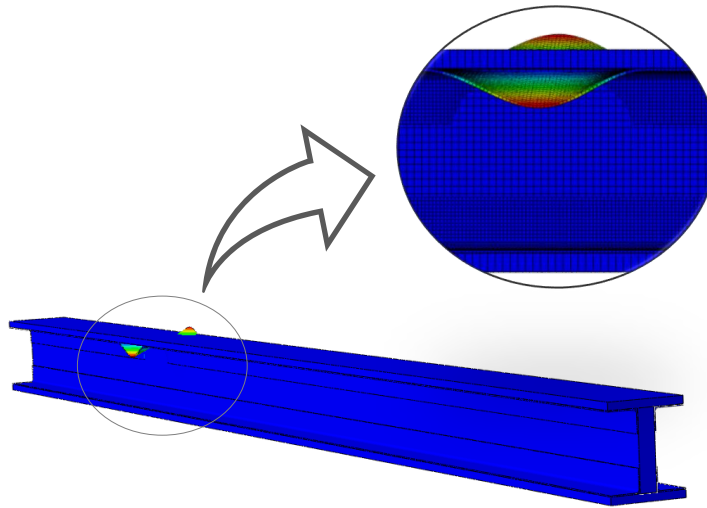


Figure 2.19. Buckling simulation of a damaged beam. Local buckling occurs at the debonding region.

other parts on the damage region.

To create a new damage, all the partitions created for the previous damage must be removed from the model and new partitions must be created at the location and size of

interest. By doing so, all the meshes of the altered parts are automatically removed and the parts must be meshed again. Also, all tie constraints must be re-defined. This is a very time-consuming process if done manually at the GUI, therefore the need to create Python scripts to control Abaqus and perform this process in an automated manner.

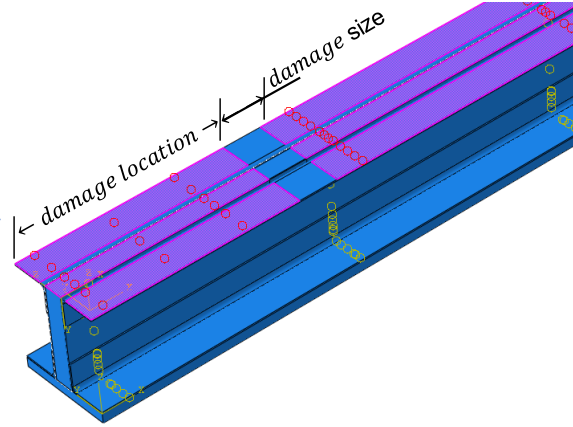


Figure 2.20. Visual representation of how damage size and damage location are defined. Regions in purple indicate that tie constraints were applied between them and the top cap (removed from the image for a better view).

2.1.7 Collecting strain data

Knowing that an optical fiber Bragg grating sensor system would be deployed in field, the target measurements from the FE model were uniaxial strain. On Abaqus, the default results respect element orientations but suppress nodal transformation. To generate results that would actually be compatible with strain measured by strain gauges or FBG sensors, a local coordinate system was created on the corner of the top spar cap (showed in yellow in Figure 2.21) where the x-direction is along the beam length, the z-direction is perpendicular to the surface and the y-direction completes the right-hand coordinate system. Then, a transformation was applied to all element and node-based results using this specific coordinate system. This transformation and the strain components measured are illustrated in Figure 2.21.

For this coordinate system, the sensors aligned in the x-direction would read the strain component ε_{11} , while the sensors orthogonal to them would read the strain component ε_{22} . If sensors were placed in the 45° direction on that same surface, it would be necessary to generate a second local coordinate system rotated by 45° around the z-axis and extract the strain component ε_{11} given by the results transformation using that second coordinate system.

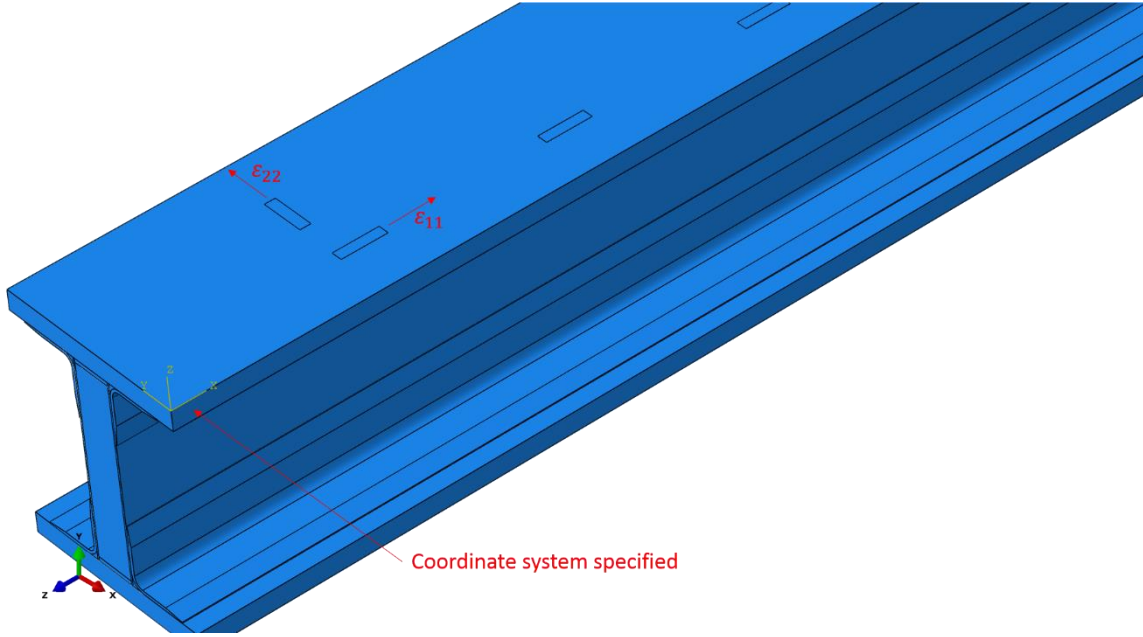


Figure 2.21. Coordinate system specified to guarantee that ε_{11} is being measured in the longitudinal direction of the beam, and that ε_{22} is orthogonal to it.

A preliminary study was made to observe the strain field along the entire spar, and it was determined that saving the strain data along a few paths on the beam would suffice to capture the local change in strain caused by the damage. This saves time and computational memory, as strain data did not need to be recorded for every single location at the beam. Therefore, nine paths were defined, as shown in Figure 2.22. Paths 1 to 5 are located at the top cap - paths 1 and 5 are 5mm from the edge of the cap, path 3 is right at the center, and paths 2 and 4 are in the middle of them. Paths 6 (right) and 8 (left) are symmetrically located at the bottom of the top shear clips, right at the center.

Paths 7 (right) and 9 (left) are also symmetrical from each other, at the lateral of the top shear clips.

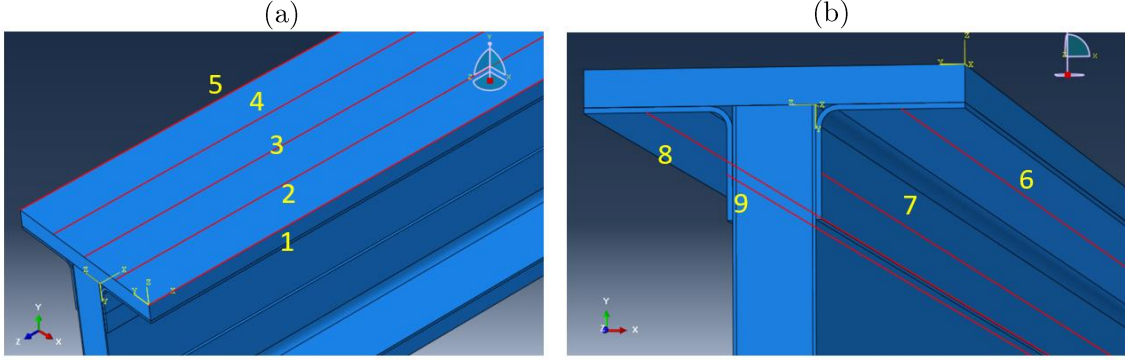


Figure 2.22. Paths defined for strain collection.

Strain measurements collected from these paths when the spar contains a debonding damage that is $0.05m$ long and located at $0.74m$ from the root are shown in Figures 2.23 and 2.24. Recall from Figures 2.4 and 2.5 that the spar's caps are $5mm$ thick, while the shear clips are $0.75mm$ thick. Therefore, the change in strain caused by the damage is much more noticeable on the shear clips, making them strong candidates for sensor placement.

Notice that the debonding damage only affects the strain field at a local level, around the damage region, and the overall strain behavior remains the same in locations further away from the debonding. This will be a key aspect in the upcoming Chapters and when defining the optimization problem.

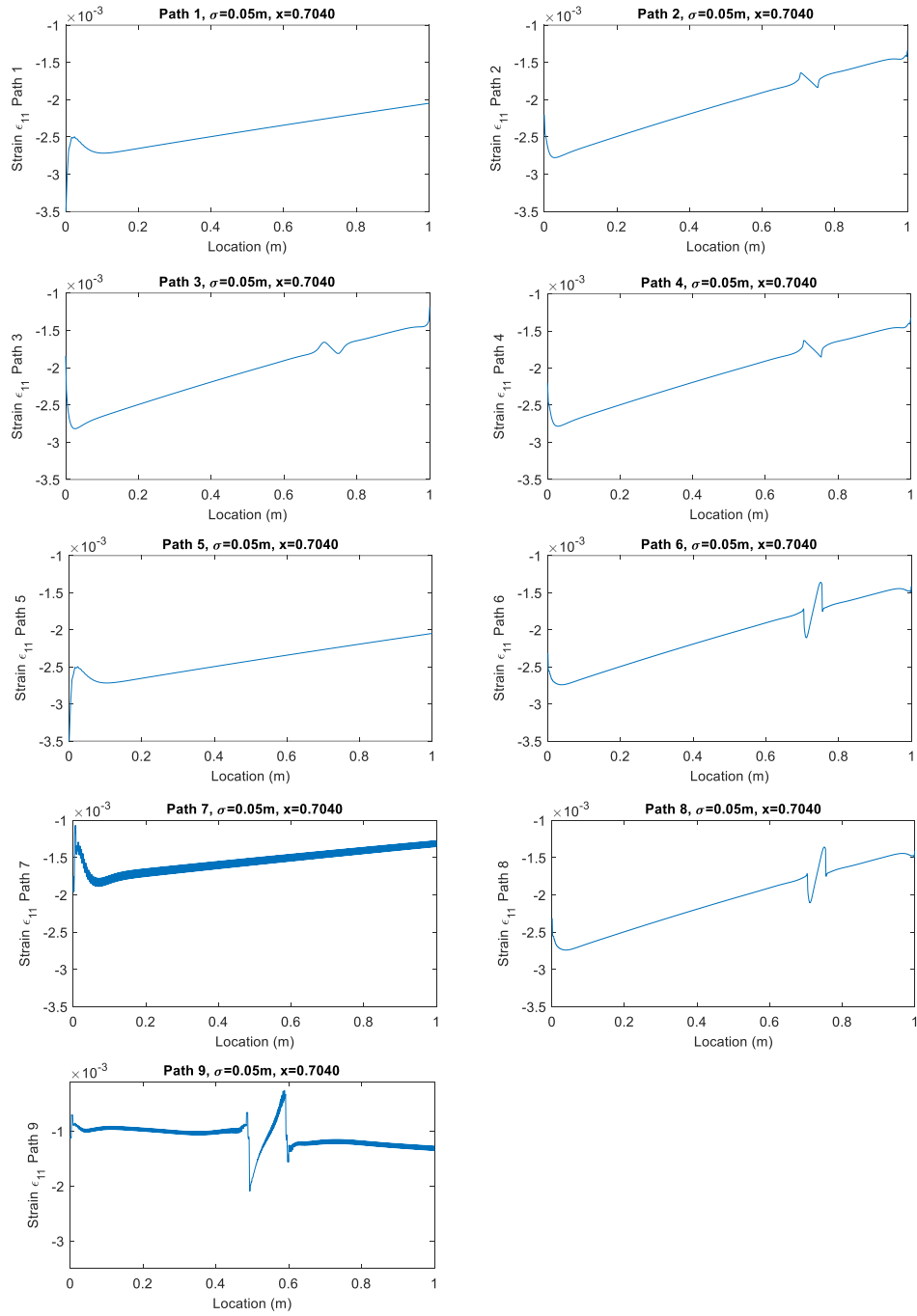


Figure 2.23. Strain ϵ_{11} collected on all paths defined in Figure 2.22. Notice that this is a 1m submodel of the 5m beam.

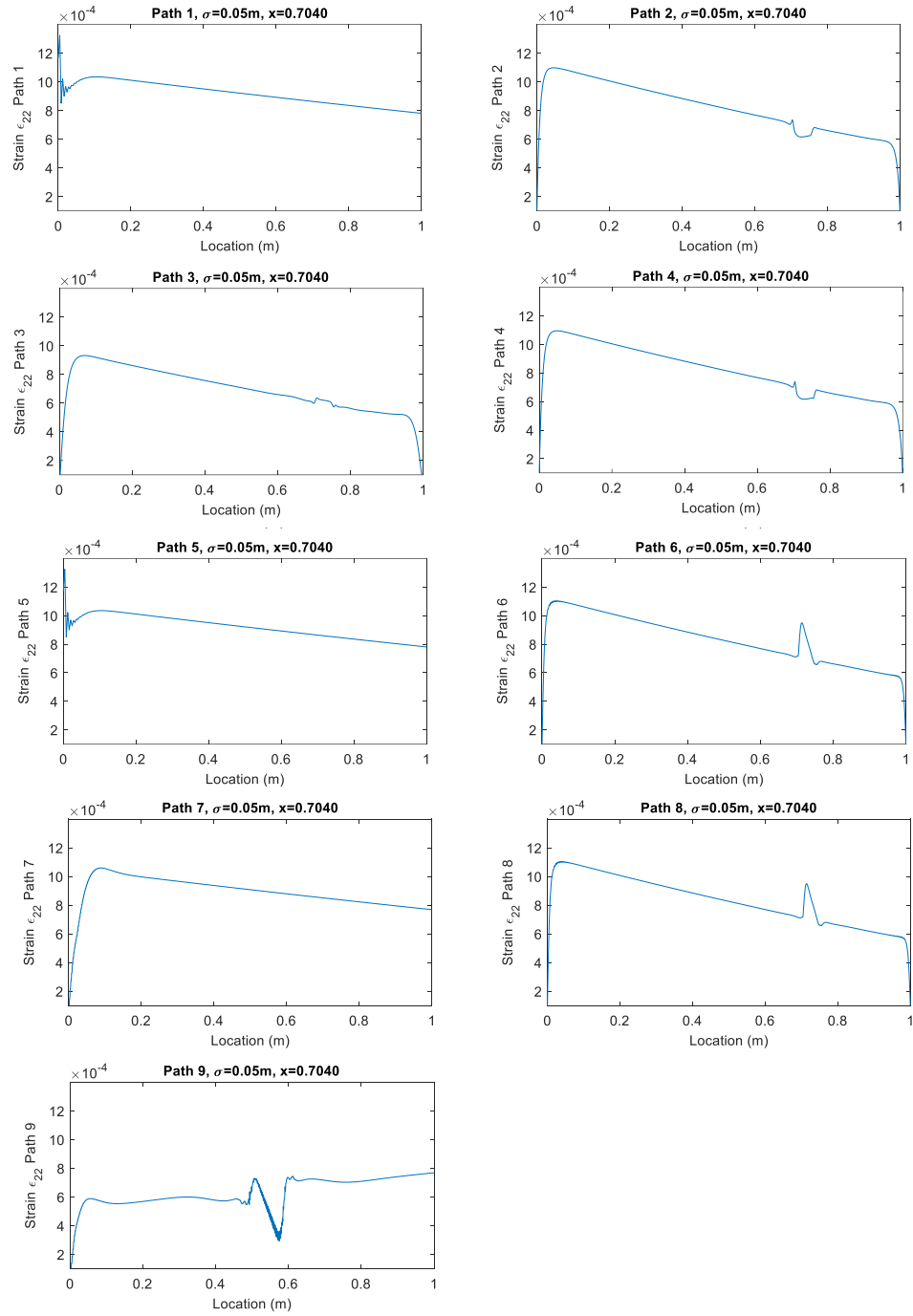


Figure 2.24. Strain ϵ_{22} collected on all paths defined in Figure 2.22. Notice that this is a 1m submodel of the 5m beam.

2.2 Model validation

To validate the strain behavior FE model previously discussed, a 1-meter replica of the I-beam with a prescribed debonding defect was constructed. The experimental setup consisted of mounting the beam in a fixed-free boundary condition and loading the free-end with a point load. The strain behavior of the beam was observed using 14 strain-gauges attached to strategic locations along the beam. The goal was to observe and validate the global deflection of the beam and, more importantly, the local buckling of the purposely made defect.

To achieve close dimensions and similar properties to the I-beam shown in the technical drawings from Figure 2.4, it was decided to build the top and bottom caps with the same thickness of 5 mm as the modeled I-beam. The web would consist of a 10mm foam core sandwiched by 3 layers of carbon fiber on each side (regardless of the 0.45 mm carbon fiber layers from the drawings), and 4-ply shear clips (regardless of the 0.75 mm thick from the actual I-beam). These decisions were made based on the properties of the carbon fiber material available for construction in the educational lab at the Structural Engineering Department, UCSD. Therefore, it was expected that the mechanical properties of the constructed parts would diverge significantly from the modeled beam's properties. Nonetheless, the overall strain behavior should remain the same.

The properties of the carbon fiber woven fabric that was readily available for the construction are: longitudinal modulus $E_1 \approx 77$ GPa, transverse modulus $E_2 \approx 75$ GPa, Poisson's ratio $\nu_{12} \approx 0.006$, and ply thickness $t \approx 0.00033$ m. The next section will describe the analytical estimate of the laminate properties.

2.3 Mechanics of Composite Laminates

For a state of plane stress parallel to $x_1 - x_2$ plane in an orthotropic solid ($\sigma_{33} = \sigma_{13} = \sigma_{23} = 0$), the stress-strain relationship is given by

$$\begin{Bmatrix} \varepsilon_{11} \\ \varepsilon_{22} \\ \gamma_{12} \end{Bmatrix} = \begin{bmatrix} S_{11} & S_{12} & S_{16} \\ S_{21} & S_{22} & S_{26} \\ S_{61} & S_{62} & S_{66} \end{bmatrix} \cdot \begin{Bmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{12} \end{Bmatrix} \quad (2.2)$$

where the compliance matrix' $\mathbf{S}$ components for all plies in a local coordinate system are $S_{11} = \frac{1}{E_1}$, $S_{12} = S_{21} = \frac{-\nu_{12}}{E_1}$, $S_{16} = S_{61} = S_{62} = S_{26} = 0$, $S_{22} = \frac{1}{E_2}$, $S_{66} = \frac{1}{G_{12}}$. To transform the stress and strain components from the local coordinate system $x_1 - x_2$ (aligned with the composite fiber direction) to the global coordinate system $x - y$, as shown in Figure 2.25, it is necessary to find the transformation matrices $\mathbf{T}_\sigma$ and $\mathbf{T}_\varepsilon$ that perform these projections.

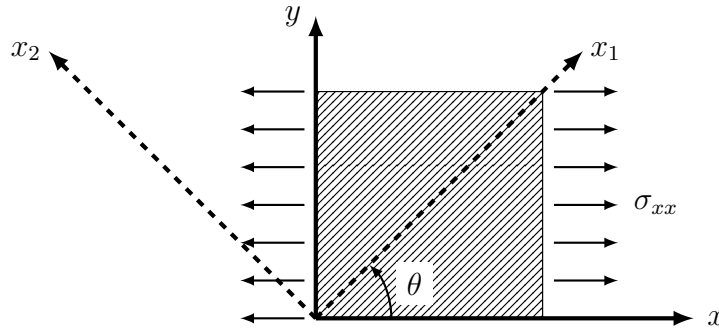


Figure 2.25. Global $x - y$ and local $x_1 - x_2$ coordinate system on a single ply composite under plane-stress, and load only on x direction.

The transformation matrix that projects the stresses $\boldsymbol{\sigma}$ from global to local coordinate systems such as

$$\boldsymbol{\sigma}_{x_1-x_2} = \mathbf{T}_\sigma \cdot \boldsymbol{\sigma}_{x-y} \quad (2.3)$$

is given by

$$\mathbf{T}_\sigma = \begin{bmatrix} \cos^2 \theta & \sin^2 \theta & 2 \sin \theta \cos \theta \\ \sin^2 \theta & \cos^2 \theta & -2 \sin \theta \cos \theta \\ -\sin \theta \cos \theta & \sin \theta \cos \theta & \cos^2 \theta - \sin^2 \theta \end{bmatrix} \quad (2.4)$$

Similarly, the strain $\boldsymbol{\varepsilon}$ transformation is given by

$$\boldsymbol{\varepsilon}_{x_1-x_2} = \mathbf{T}_\varepsilon \cdot \boldsymbol{\varepsilon}_{x-y} \quad (2.5)$$

in which

$$\mathbf{T}_\varepsilon = \begin{bmatrix} \cos^2 \theta & \sin^2 \theta & \sin \theta \cos \theta \\ \sin^2 \theta & \cos^2 \theta & -\sin \theta \cos \theta \\ -2 \sin \theta \cos \theta & 2 \sin \theta \cos \theta & \cos^2 \theta - \sin^2 \theta \end{bmatrix}. \quad (2.6)$$

Using these relations, the tensor transformation that projects the compliance matrix from the local coordinate ($\mathbf{S}$) system to the global coordinate system ($\bar{\mathbf{S}}$) can be derived by substituting Equations (2.4) and (2.6) into the strain transformation in Equation (2.5)

$$\boldsymbol{\varepsilon}_{x-y} = \mathbf{T}_\varepsilon^{-1} \cdot \boldsymbol{\varepsilon}_{x_1-x_2} = \mathbf{T}_\varepsilon^{-1} \cdot \mathbf{S} \cdot \boldsymbol{\sigma}_{x_1-x_2} = \mathbf{T}_\varepsilon^{-1} \cdot \mathbf{S} \cdot \mathbf{T}_\sigma \cdot \boldsymbol{\sigma}_{x-y} \quad (2.7)$$

thus,

$$\bar{\mathbf{S}} = \mathbf{T}_\varepsilon^{-1} \cdot \mathbf{S} \cdot \mathbf{T}_\sigma = \mathbf{T}_\sigma^T \cdot \mathbf{S} \cdot \mathbf{T}_\sigma = \begin{bmatrix} \bar{S}_{11} & \bar{S}_{12} & \bar{S}_{16} \\ \bar{S}_{21} & \bar{S}_{22} & \bar{S}_{26} \\ \bar{S}_{61} & \bar{S}_{62} & \bar{S}_{66} \end{bmatrix} \quad (2.8)$$

which is used to compute the global reduced stiffness for each ply via $\bar{\mathbf{Q}} = \bar{\mathbf{S}}^{-1}$.

When multiple composite laminae are stacked, a laminate is formed. The stacking sequence is represented by $[\theta_1/\theta_2/\theta_3/\dots/\theta_N]$ in which θ_k is the orientation of the k -th ply of a laminated made of N plies [59]. The classical laminate plate theory and the plate

constitutive equations together can be used to describe the plate (laminate) behavior as

$$\{N\} = [A] \{\varepsilon^0\} + [B] \{\kappa\} \quad (2.9a)$$

$$\{M\} = [B] \{\varepsilon^0\} + [D] \{\kappa\}. \quad (2.9b)$$

or, in a more detailed format,

$$\begin{pmatrix} N_x \\ N_y \\ N_{xy} \\ M_x \\ M_y \\ M_{xy} \end{pmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{16} & B_{11} & B_{12} & B_{16} \\ A_{12} & A_{22} & A_{26} & B_{12} & B_{22} & B_{26} \\ A_{16} & A_{26} & A_{66} & B_{16} & B_{26} & B_{66} \\ B_{11} & B_{12} & B_{16} & D_{11} & D_{12} & D_{16} \\ B_{12} & B_{22} & B_{26} & D_{12} & D_{22} & D_{26} \\ B_{16} & B_{26} & B_{66} & D_{16} & D_{26} & D_{66} \end{bmatrix} \begin{pmatrix} \varepsilon_x^0 \\ \varepsilon_y^0 \\ \gamma_{xy}^0 \\ \kappa_x \\ \kappa_y \\ \kappa_{xy} \end{pmatrix} \quad (2.10)$$

in which N_i are the in-plane forces applied to the laminate, M_i are the bending moment, ε_i^0 are the in-plane strains, and κ_i are the mid-surface curvature associated with out-of-plane deflections. The extensional stiffness matrix $\mathbf{A}$, the coupling stiffness matrix $\mathbf{B}$, and the bending stiffness matrix $\mathbf{D}$ are

$$A_{ij} = \sum_{k=1}^n \bar{Q}_{ij}^{(k)} t_k \quad (2.11a)$$

$$B_{ij} = \sum_{k=1}^n \bar{Q}_{ij}^{(k)} t_k \bar{z}_k \quad (2.11b)$$

$$D_{ij} = \sum_{k=1}^n \bar{Q}_{ij}^{(k)} \left(t_k \bar{z}_k^2 + \frac{t_k^3}{12} \right) \quad (2.11c)$$

being t_k is the k -th ply thickness and $\bar{z}_k$ is the k -th ply central position related to the laminate geometric mid-plane.

Laminates symmetric with respect to the mid-plane possess no coupling stiffness,

therefore $\mathbf{B} = \mathbf{0}$, and Equation (2.9) is simplified to

$$\{N\} = [A] \{\varepsilon^0\} \quad (2.12a)$$

$$\{M\} = [D] \{\kappa\}. \quad (2.12b)$$

Furthermore, if the stacking consists only of 0° , 90° , and $\pm 45^\circ$ plies then $A_{16} = A_{26} = 0$.

For a N -ply laminate with ply thickness t , the total thickness of the laminate is $h = Nt$, the effective moduli of this laminate, i.e., its properties when treated as an equivalent anisotropic solid, are estimated using the average stresses ($\bar{\sigma}_x, \bar{\sigma}_y$, and $\bar{\sigma}_{xy}$) and mid-plane strains ($\varepsilon_x^0, \varepsilon_y^0$, and γ_{xy}^0) via

$$\begin{Bmatrix} \varepsilon_x^0 \\ \varepsilon_y^0 \\ \gamma_{xy}^0 \end{Bmatrix} = h \begin{bmatrix} A_{11} & A_{12} & 0 \\ A_{21} & A_{22} & 0 \\ 0 & 0 & A_{66} \end{bmatrix}^{-1} \begin{Bmatrix} \bar{\sigma}_x \\ \bar{\sigma}_y \\ \bar{\sigma}_{xy} \end{Bmatrix} \quad (2.13)$$

such that the terms in the multiplication $h\mathbf{A}^{-1}$ correspond to the compliance matrix for the strain-stress relation shown below

$$\begin{Bmatrix} \varepsilon_{xx} \\ \varepsilon_{yy} \\ \gamma_{xy} \end{Bmatrix} = \begin{bmatrix} \frac{1}{E_x} & \frac{-\nu_{yx}}{E_x} & \frac{-\eta_{xy,x}}{G_{xy}} \\ \frac{-\nu_{xy}}{E_x} & \frac{1}{E_y} & \frac{-\eta_{xy,y}}{G_{xy}} \\ \frac{\eta_{x,xy}}{E_x} & \frac{\eta_{y,xy}}{E_y} & \frac{1}{G_{xy}} \end{bmatrix} \begin{Bmatrix} \sigma_{xx} \\ \sigma_{yy} \\ \sigma_{xy} \end{Bmatrix}. \quad (2.14)$$

Therefore, the laminate's effective moduli are

$$E_x = \frac{A_{11}A_{22} - A_{12}^2}{hA_{22}} \quad (2.15a)$$

$$E_y = \frac{A_{11}A_{22} - A_{12}^2}{hA_{11}} \quad (2.15b)$$

$$\nu_{xy} = \frac{A_{12}}{A_{22}} \quad (2.15c)$$

$$\nu_{yx} = \frac{A_{12}}{A_{22}} \quad (2.15d)$$

$$G_{xy} = \frac{A_{66}}{h} \quad (2.15e)$$

2.3.1 Engineering the Caps, Shear Clips, and Web

The caps, shear clips, and web were designed using AGP370-5H/3501-6S woven carbon fabric. The web is a carbon fiber - foam sandwich in which the foam used was the Divinycell Low Density Foam (H-45). The moduli of the composite and foam are presented in Table 2.3.

Table 2.3. Properties of the different materials used for construction.

Material	E_1	E_2	G_{12}	ν_{12}	Ply thickness
AGP370-5H/3501-6S	77.2 GPa	75.2 GPa	6.5 GPa	0.06	0.33 mm
Divinycell H-45 Foam	50 MPa	50 MPa	50 MPa	0.4	

The top and bottom caps were designed to have a laminate thickness h equivalent to the FE model's. Therefore, the number of plies in the caps is

$$n_{\text{Cap}} = \frac{h_{\text{FEA}}}{t_{\text{carbon fiber}}} = \frac{5\text{mm}}{0.33\text{mm}} \approx 15.14 = 16 \text{ plies.} \quad (2.16)$$

Limited by 16 plies, the stacking sequence of the caps was chosen by the combination of plies oriented at 0° , 90° , and $\pm 45^\circ$ that yielded the closest effective moduli to the properties used in the FE model. The number of plies in the clips and web were chosen with criterion of achieving effective moduli as close as possible to the effective moduli used in the FE model, independently of the number of plies. The stacking sequence and effective properties for each I-beam component are presented in Table 2.4.

Table 2.4. Effective moduli of the laminates analyzed.

Laminate	Stacking Sequence	E_x [GPa]	E_y [GPa]	G_{xy} [GPa]	ν_{xy}	ν_{yx}
Cap	$[90^\circ/0_3^\circ/\pm 45^\circ/0_2^\circ]_s$	67.5	66.5	13.9	0.1735	0.1709
Clips	$[45^\circ/0^\circ/-45^\circ]_s$	46.8	46.2	26.1	0.4283	0.4231
Web	$[\pm 45^\circ/0_{\text{Foam}}^\circ]_s$	33.9	33.6	23	0.4851	0.4806

2.4 Manufacturing

All components of the I-beam were manufactured using the wet layup method, where the two-part Poly Epoxy Structural Epoxy System manufactured by Aircraft Spruce was used as the resin.

2.4.1 Top and Bottom Caps

To create the top and bottom caps, rectangular sections (1.2 meters by 7 centimeters) of AGP370-5H/3501-6S woven carbon fabric were utilized. These dimensions exceeded those of the modeled 1-meter beam in finite elements. The additional material in both width and length allowed for trimming the cured part to the desired specifications and provided an extra 15 centimeters, resulting in a 1.15-meter-long beam. This excess length was intended to allow clamping the beam for a cantilever beam experimental setup.

The layup process involved stacking each layer of carbon fiber fabric onto an aluminum tool coated with release film. After applying a thin layer of epoxy to the designated layup area, each layer of carbon fiber fabric was carefully positioned and saturated with epoxy resin. Once the final ply was in place and thoroughly impregnated with resin, the layup was covered with release fabric and a layer of polyester breather to evenly distribute vacuum pressure. The entire assembly was then vacuum bagged and cured under a vacuum pressure of 14.7 psi. The schematics of this process are shown in Figure 2.26.

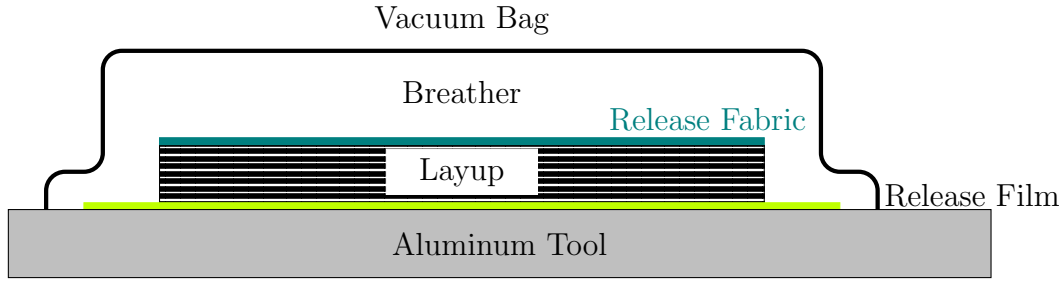


Figure 2.26. Top and bottom caps vacuum-bag curing schematics.

2.4.2 Shear Clips

The layup process for the shear clips followed the same manufacturing procedure as the caps. However, for the layup of the shear clips, a C-shaped aluminum tool was employed on the outer surface of the mold. This specific choice of tool ensured that the release fabric (also referred to as “peel ply”) would yield a consistent non-glossy surface finish on that side of the cured part, resulting in a better surface for bonding.

Rectangular sections (1.2 meters by 8 centimeters) of carbon fiber fabric were carefully stacked to form a C-shaped composite. This stacking configuration ensured symmetrical lateral edges measuring 2 centimeters in height and 4 centimeter wide. The schematics of the whole process is shown in Figure 2.27 and a picture of the shear clip manufacturing is shown in Figure 2.28.

2.4.3 Web

The layup process for manufacturing the web of the I-beam did not involve the use of an aluminum tool. Initially, the 1.2-meter-long, 5-centimeter-high, and 1-centimeter-wide foam core was saturated with resin along its lateral walls. Subsequently, layers of carbon fiber fabric pre-soaked in resin were alternately stacked onto each lateral wall of the foam core. This sandwich structure was then enveloped in release fabric, followed by a layer of polyester breather, and finally cured under pressure within a vacuum bag. The schematics of this process is shown in Figure 2.29. Notice that covering this structure

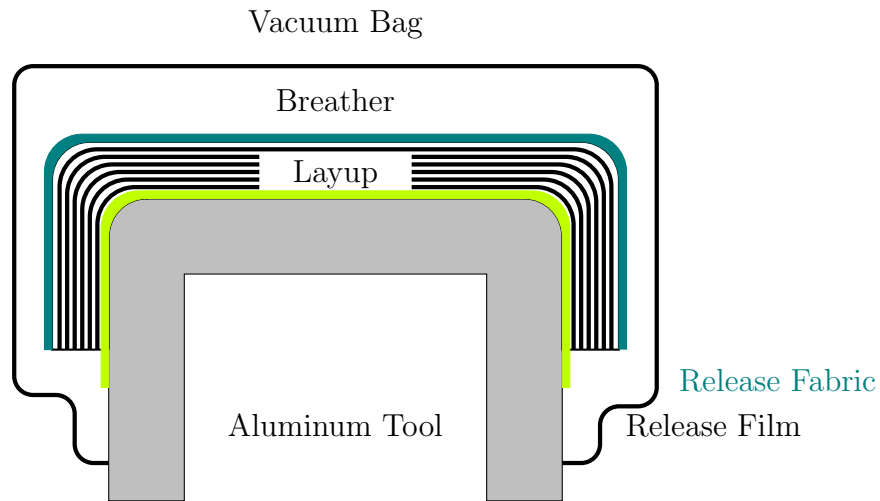


Figure 2.27. Shear clips vacuum-bag curing schematics.

with the release fabric for the curing process will created lateral walls with enhanced the bonding performance.



(a)



(b)

Figure 2.28. Manufacturing of the shear clips: (a) wet layup and (b) vacuum bag cure.

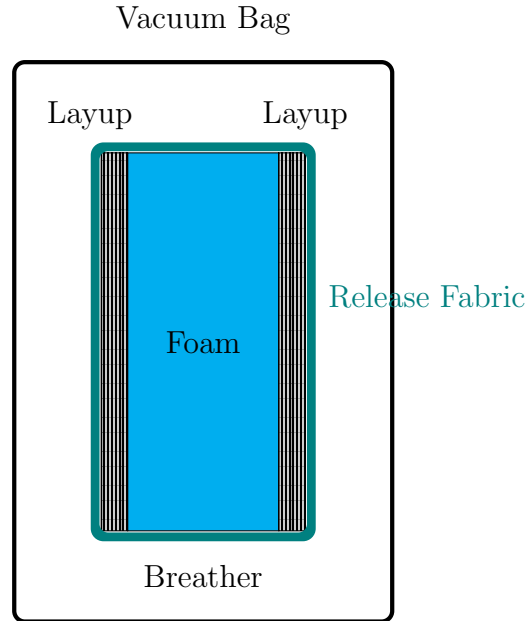


Figure 2.29. Web carbon fiber - foam sandwich vacuum-bag curing schematics.

2.4.4 Assembly

After curing, all parts were trimmed to the appropriate width and length. The two caps were trimmed to a width of 5 centimeters each. The web underwent trimming at the top and bottom to eliminate any excess composite material adhering to these surfaces. The four shear clips were made by cutting the C-shaped laminates in half and removing any excess height from the lateral walls. All components were cut to a length of 1.15 meters. The I-beam parts are shown in Figure 2.30.

The assembly of the I-beam was performed in two stages as illustrated in Figure 2.31:

1. The bottom cap, web and two shear clips were attached forming a “T” beam.
2. The remaining two shear clips and the top cap to the “T” beam formed in the first stage.

The structural adhesive used was the Aeropoxy Structural Adhesive ES6209 from PTM&W.

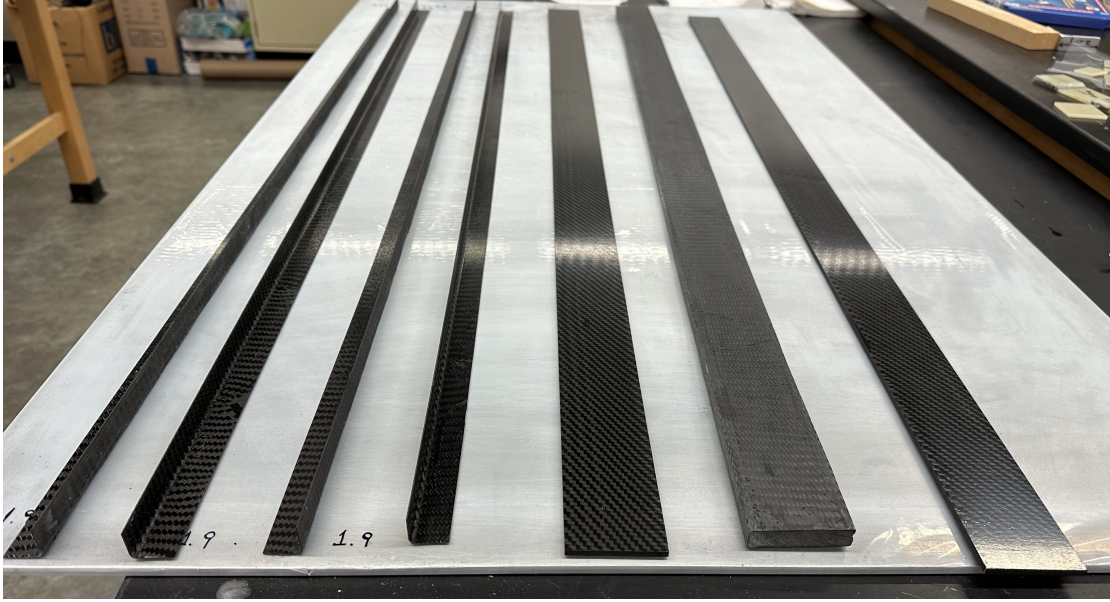


Figure 2.30. I-beam components after they were trimmed to the correct size.

To guarantee adequate bonding between the parts in stage 1, the a distributed pressure was applied the bonding area of the web and clip contact. This was achieved by positioning two metal beams with rectangular cross-sections on each side of the web, while multiple clamps compressed this sandwich during the adhesive curing cycle. Similarly, to guarantee a good bonding between the cap and shear clips, the same metal beams were utilized to exert pressure against the cap, ensuring proper contact between the shear clip and cap bonding areas. This assembly step is shown in Figure 2.32(a).

With an “T” beam assembled, the second step of the assembly was performed by pressing both shear clips against the web using multiple c-clamps, and by pressing the shear clips against the cap using spring clamps. This assembly step is shown in Figure 2.32(b).

One important aspect of the second stage of the assembly is that it involved deliberately creating a debonding between the top cap and both the shear clips and core. This intentional damage was generated by preventing bonding between the parts for a 10-centimeter segment beginning 25 centimeters from the root of the beam. No structural

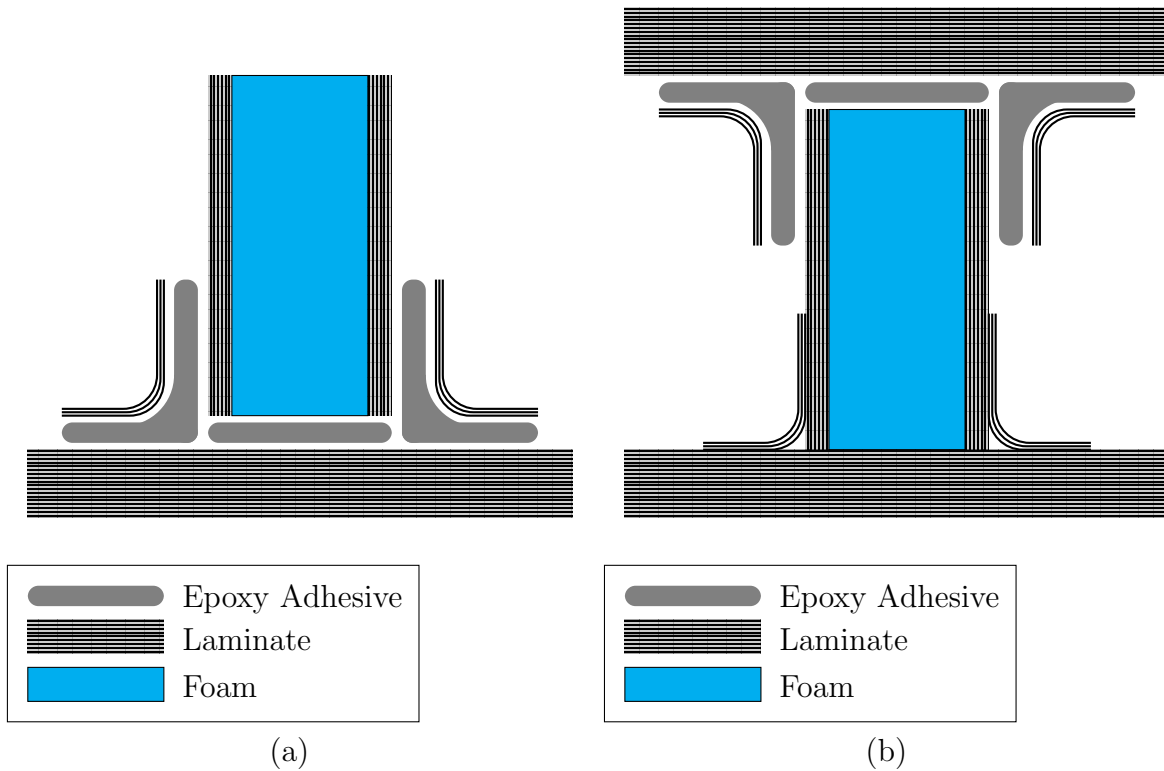
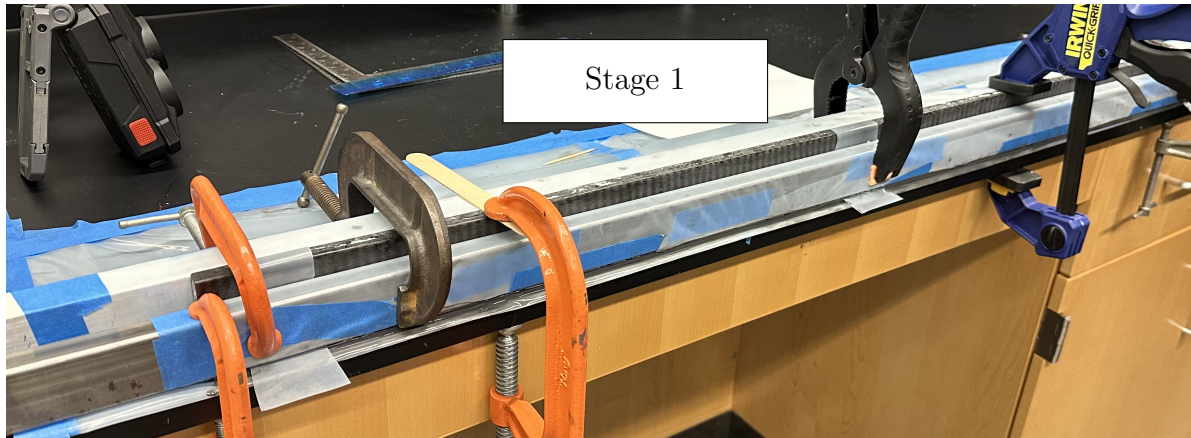
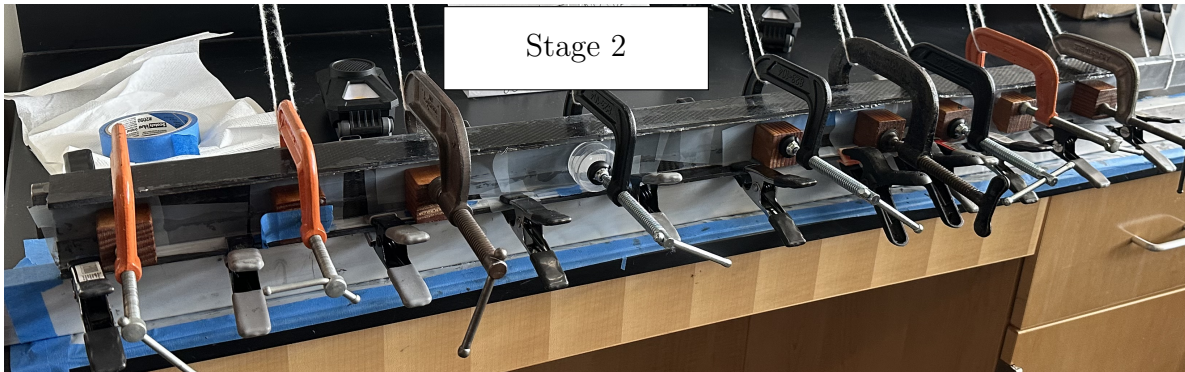


Figure 2.31. Two stages of the I-beam assembly: (a) the bottom cap, two shear clips and web were assembled, (b) remaining cap and two shear chips where assembled to the “T” beam formed in stage 1.



(a)



(b)

Figure 2.32. Assembly of the I beam components forming first “T” beam (a) and the final I-beam (b).

adhesive was applied in this specific region. Furthermore, a layer of release film was applied to cover the same area on the top cap, top shear clips, and the top surface of the web as shown in Figure 2.33. This precautionary measure was implemented to ensure that no unintended bonding occurred because of the adhesive spreading under the pressure applied during the assembly process.

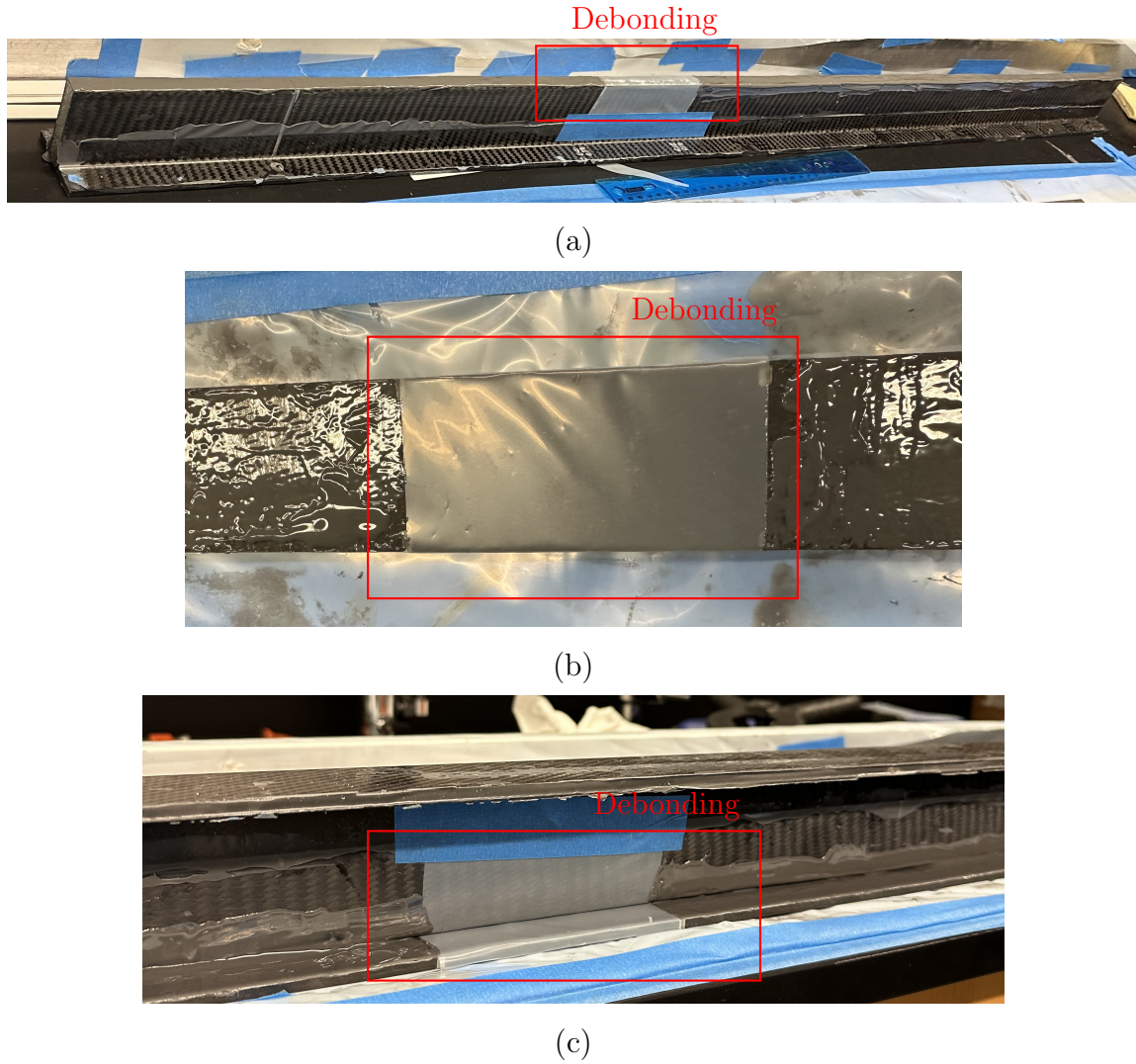
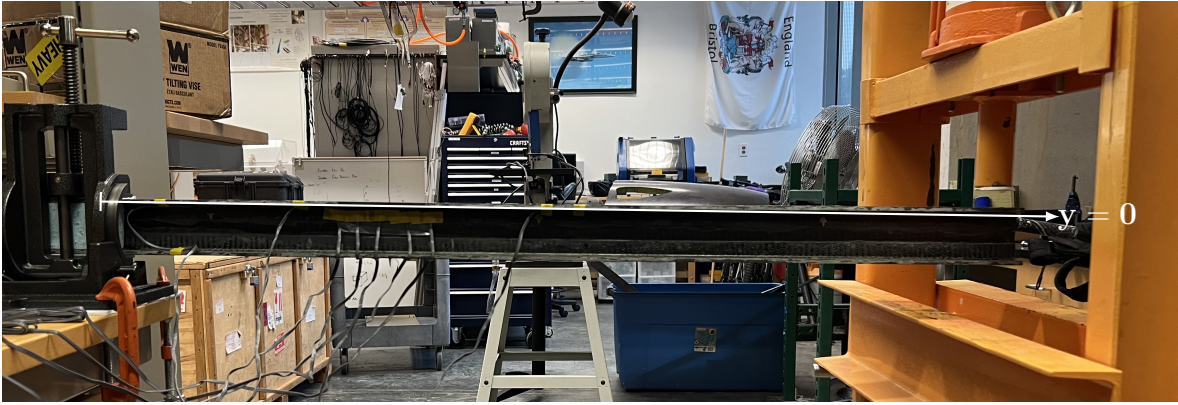


Figure 2.33. Debonding damage insertion by adding a release film layer (a) to some of the bonding area between the top cap and the web, (b) to the same bonding area of the top cap (redundancy). (c) Shows the debonding defect on the completely assembled composite I-beam.

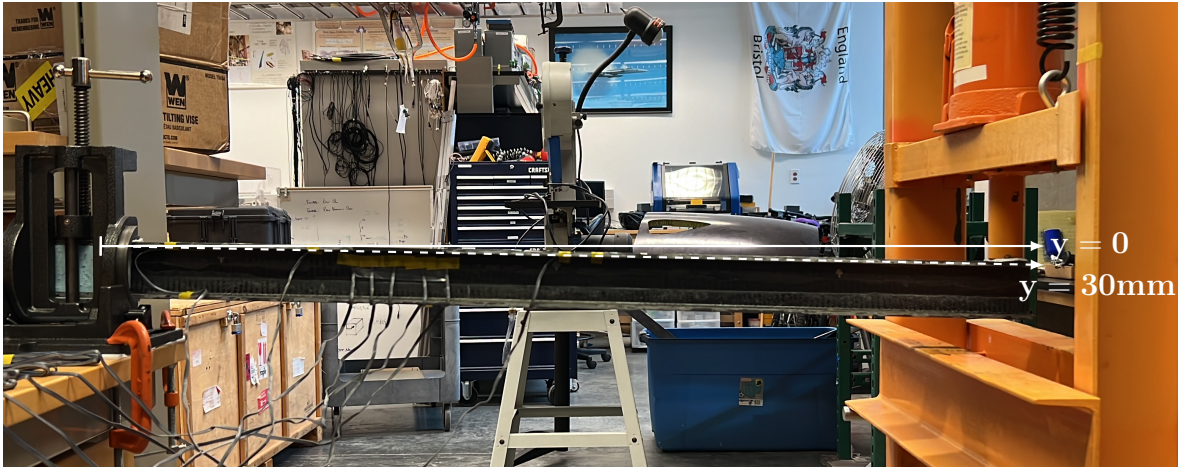
2.5 Experimental Setup

The manufactured I beam was setup as a cantilever beam subjected to a point load on the free end aiming to validate the strain behavior of the I beam modeled in finite elements (see Figure 2.34). The first 15 centimeters of the beam was clamped by a tilting vise rigidly attached to table using multiple c-clamps. The end of the clamped region of

the beam marks the $x = 0m$ position. The load was applied at $x = 1m$ as a prescribed tip displacement achieved using a hydraulic press as shown in Figure 2.34. The hydraulic press allowed for smooth and controlled deflection of the beam, as the displacement can be applied in small increments.



(a)



(b)

Figure 2.34. Cantilever beam at (a) no deflection and (b) 30 mm deflection due to a tip load.

To observe the strain behavior of beam, 15 strain gauges were glued to one of the shear clip that has an internal defect (debonding). However, strain gauges #8, #11, and #15 were defective. The strain gauge locations throughout the beam is presented in Table 2.5.

Table 2.5. Position of strain gauges along the I beam (units in [mm]).

Strain Gauges	1	2	3	4	5	6	7	8
Location	100	188	238	254	267	283	291	303
Strain Gauges	9	10	11	12	13	14	15	
Location	314	327	339	355	369	563	784	

The hardware setup for the experiment shown in Figure 2.35 consisted of FLA-5-11-3LT strain gauges manufactured by Tokyo Sokki Kenkyujo Co., Ltd, connected to a strain gauge bridge amplifier model 633 manufactured by the Encore Electronics, Inc.. The amplifiers output signal was acquired by National Instrument's NI-9234 module on a cDAQ-9178 CompactDAQ USB chassis. A MATLAB code was written for data acquisition.

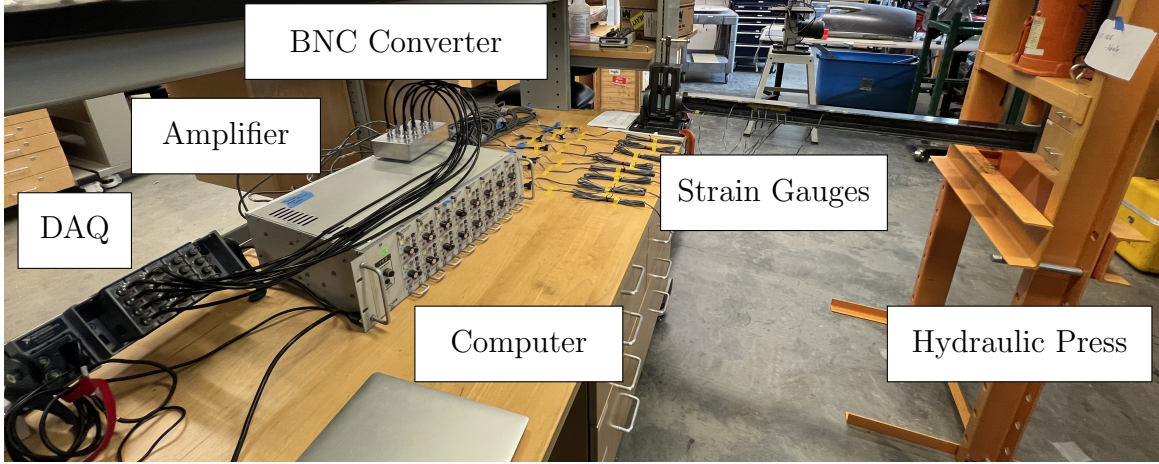


Figure 2.35. Experimental Setup.

While the beam was unloaded, data was acquired for 4 minutes at a sampling frequency of 25.6 k Hz to collect approximately 3 million samples of a baseline signal ($V_{\text{unstrained}}$) of the unstrained beam. The baseline signal is shown in Figure 2.36. Then, data from the strain gauges was collected for different tip displacement (V_{strained}). Figure 2.37 shows the voltage measured in each sensor at a 30 mm deflection of the tip.

The bias of the voltage measured at the strain gauges when the beam is deflected is removed by subtracting the baseline voltage - therefore, only the voltage output caused by the deflection will be observed. Furthermore, this measure can be normalized by the

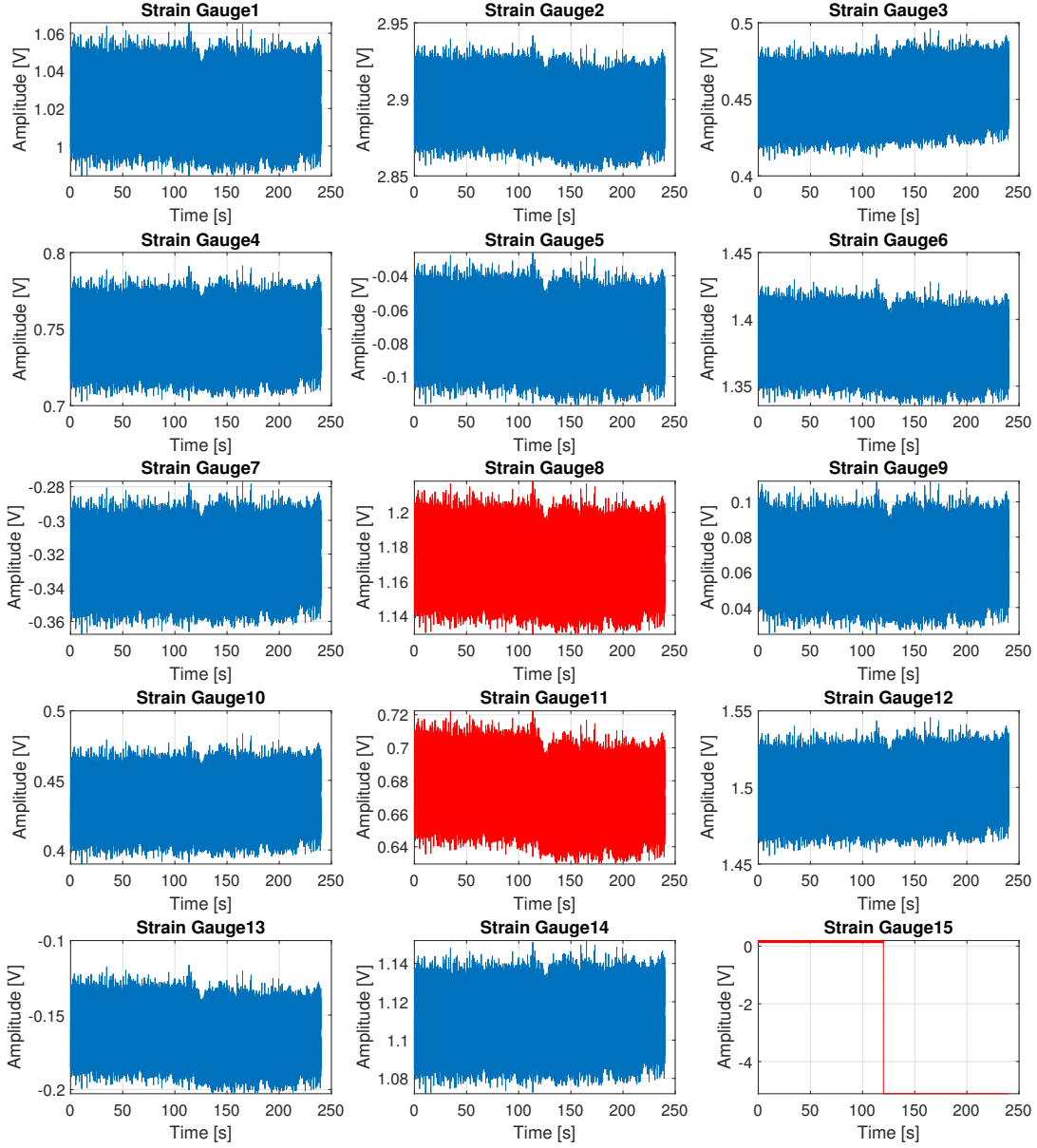


Figure 2.36. Baseline voltage at the 15 strain gauges. In blue are the operational strain gauges and in red the defective strain gauges.

voltage excitation provided to the strain gauge. Therefore the term V_r can be expressed by

$$V_r = \left[\left(\frac{V_{\text{out}}}{V_{\text{in}}} \right)_{\text{strained}} - \left(\frac{V_{\text{out}}}{V_{\text{in}}} \right)_{\text{unstrained}} \right] \quad (2.17)$$

in which V_{in} is the bridge excitation voltage excitation provided to the strain gauges [60].

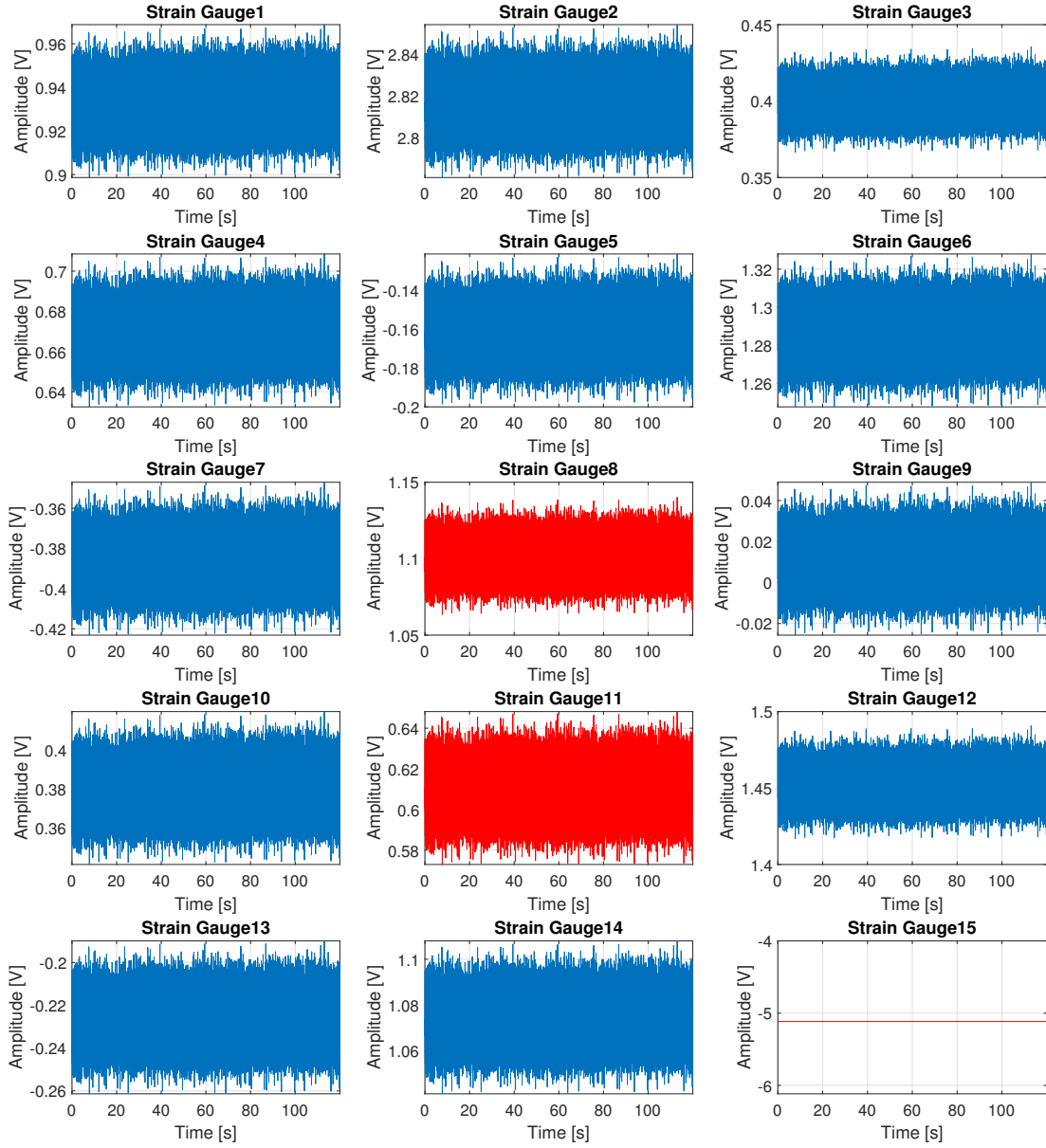


Figure 2.37. Voltage signal at the 15 strain gauges when the beam's tip is deflected 30 mm. In blue are the operational strain gauges and in red the defective strain gauges.

For this experiment, the input voltage was $V_{\text{in}} = 2V$. The amplitude V_r for the experimental data is shown in Figure 2.38 using the last 2 minutes of the baseline acquisition and the total duration (2 minutes) of the data acquisition when the beam was deflected 30 mm.

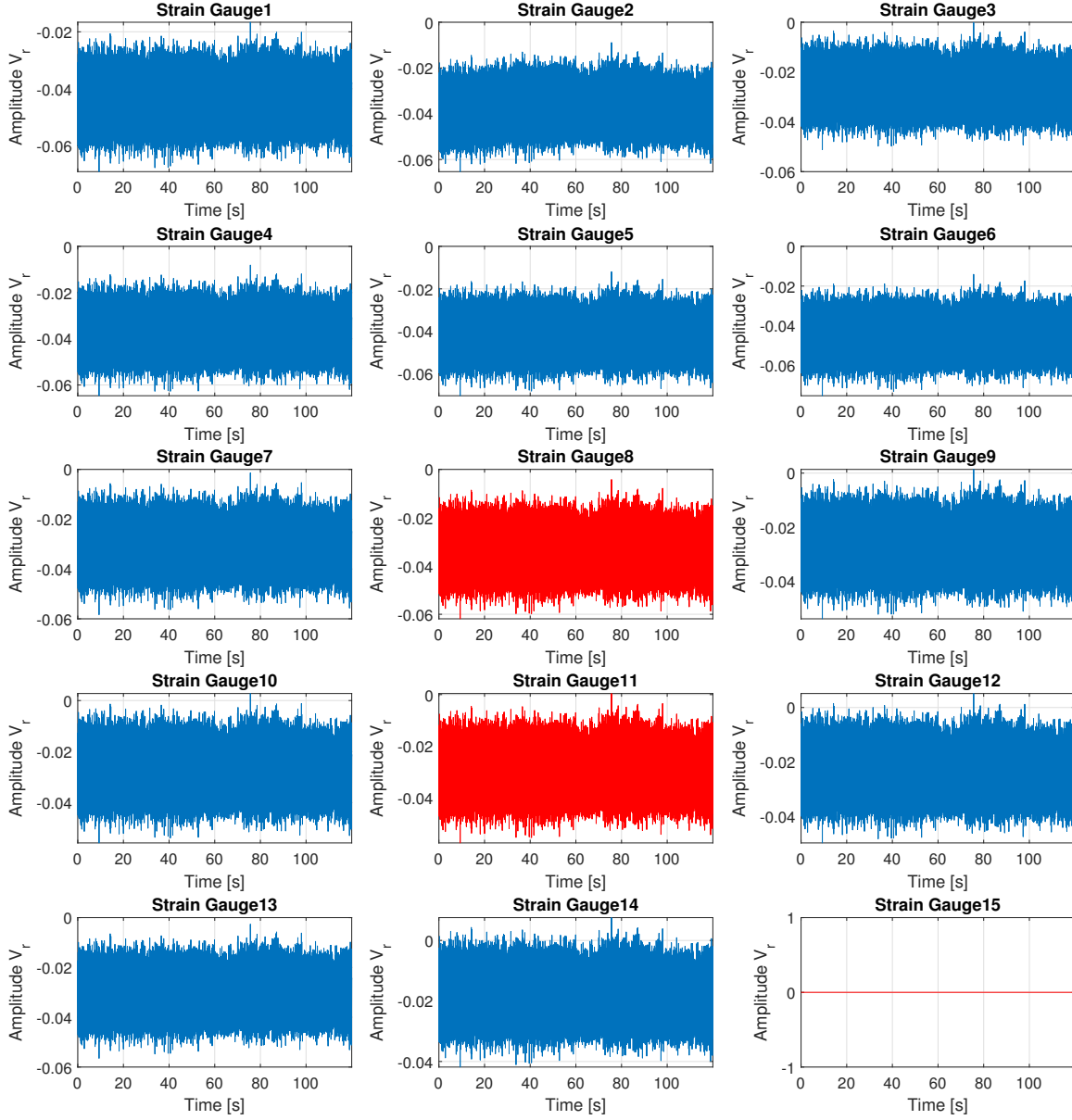


Figure 2.38. Amplitude V_r at the 15 strain gauges when the beam's tip is deflected 30 mm. In blue are the operational strain gauges and in red the defective strain gauges.

Now, the data acquired can be converted from voltage to strain via

$$\varepsilon = \frac{4V_r}{F_G(1 + 2V_r)} \left(1 + \frac{R_l}{R_g}\right) \left(\frac{1 + rK_t}{1 - \nu_0 K_t}\right) \quad (2.18)$$

in which F_G is the gauge factor (see Table 2.6), R_l is the lead wire resistance, $R_g = 120\Omega$

is the strain gauge's resistance, K_t is the strain gauge's transverse sensitivity (see Table 2.6), $r = 0.4283$ is the Poisson ratio of the material to which the strain gauge is attached (refer to Table 2.4), and $\nu_0 = 0.285$ is the Poisson ratio of the standard test material used in calibrating the strain gauge for its gauge factor [61].

Table 2.6. Strain gauge properties.

Strain Gauges	1	2	3	4	5	6	7	8
Strain Gauges Factor	2.13	2.1	2.13	2.11	2.13	2.13	2.11	2.13
Transverse Sensitivity	0	-0.2	0	0	0	0	0	0
Strain Gauges	9	10	11	12	13	14	15	
Strain Gauges Factor	2.11	2.11	2.13	2.13	2.11	2.13	2.11	
Transverse Sensitivity	0	0	0	0	0	0	-0.2	

The term $(1 + R_l/R_g)$ is known as the lead wire desensitization [60]. The wire resistance is computed via $R_l = \rho_l \ell$, in which $\rho_l = 0.44\Omega/m$ is the resistance per cable length, and $\ell = 3m$ is the lead wire length, therefore $R_l = 1.32\Omega$. The lead wire desensitization term is $(1 + R_l/R_g) \approx 1.011$.

The term $(1 + rK_t)(1 - \nu_0 K_t)$ is the correction factor for transverse sensitivity. This correction is 0.9736 for strain gauge's 2 and 15, and the correction term is 1 for all other sensors (no correction since the gauge has no transverse sensitivity).

Using Equation (2.18) to compute the strain measurements when the beam is under a 30 mm tip deflection boundary condition, the result of strain over the course of the 2 minute acquisition is shown in Figure 2.39.

The average strain ($\bar{\varepsilon}$) at each sensor d is calculated via

$$\bar{\varepsilon}[d] = \frac{\sum_{i=1}^{N_{\text{samples}}} \varepsilon[d]}{N_{\text{samples}} - 1} \quad (2.19)$$

with corresponding standard deviation

$$\sigma_{\varepsilon}[d] = \sqrt{\frac{\sum_{i=1}^{N_{\text{samples}}} (\varepsilon[d] - \bar{\varepsilon}[d])^2}{N_{\text{samples}}}} \quad (2.20)$$

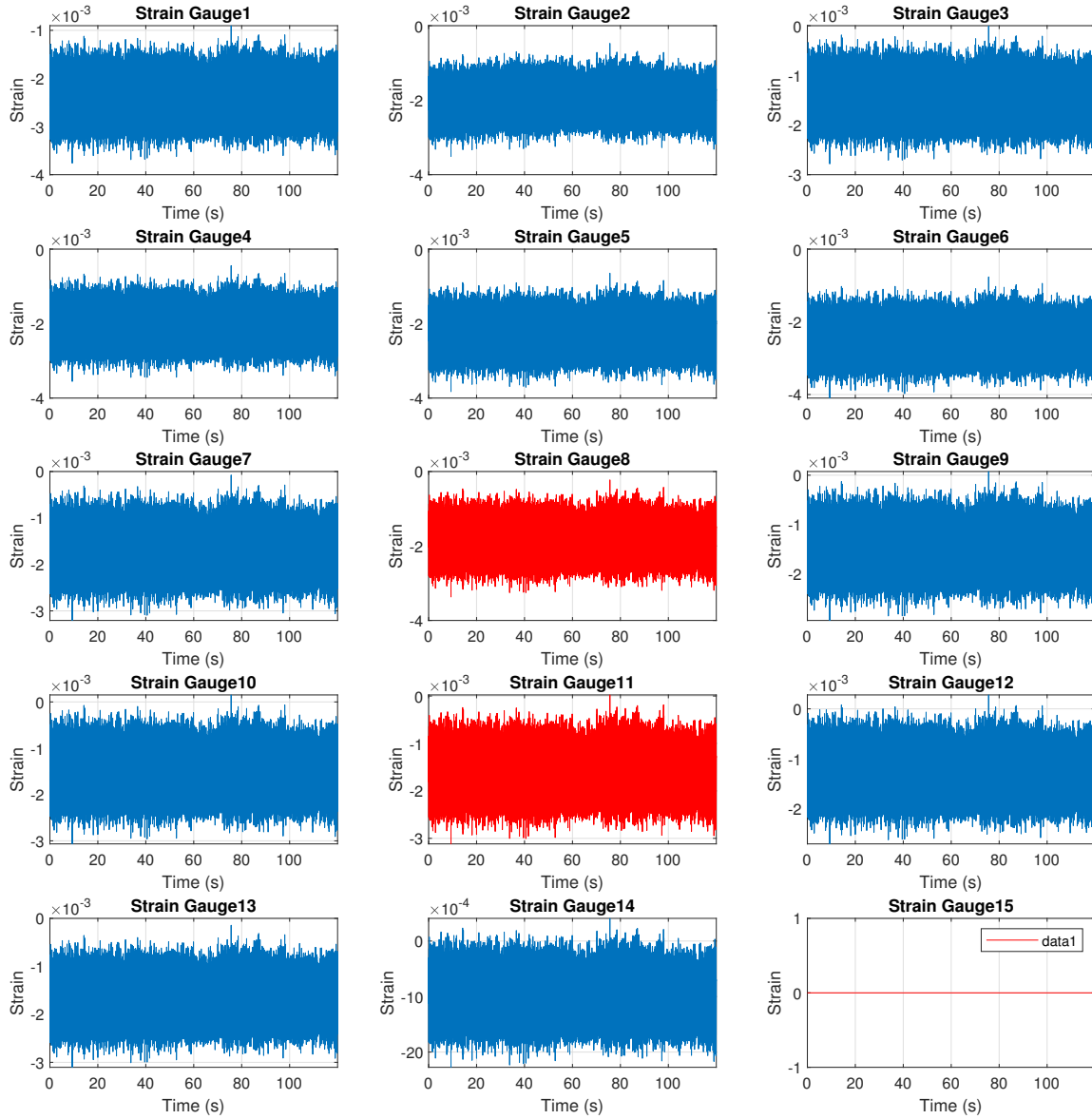
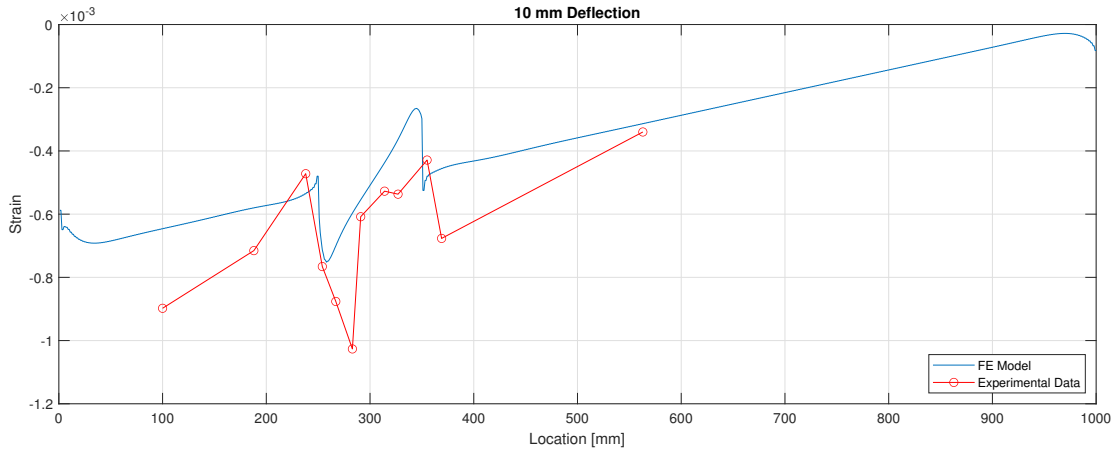


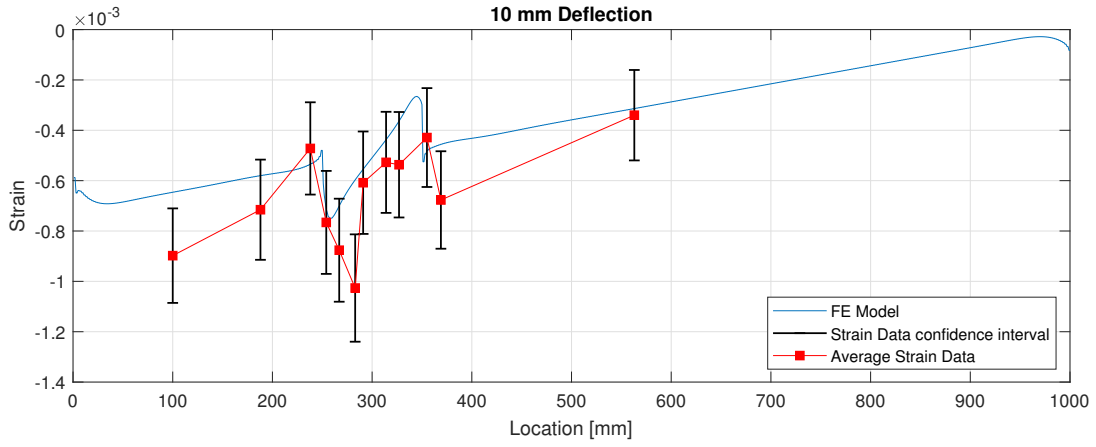
Figure 2.39. Strain at the 15 strain gauges when the beam's tip is deflected 30 mm. In blue are the operational strain gauges and in red the defective strain gauges.

To validate the strain behavior of the I beam modeled on Abaqus, the material properties used in the FE model were updated to correspond to the effective moduli estimated for the manufactured I beam (see Table 2.4). The load was also change to prescribe a displacement instead of a point force. Then, strain was extracted along the same path as where the strain gauges were attached (at the top part of the shear clip, 7

mm away from the top cap's edge).

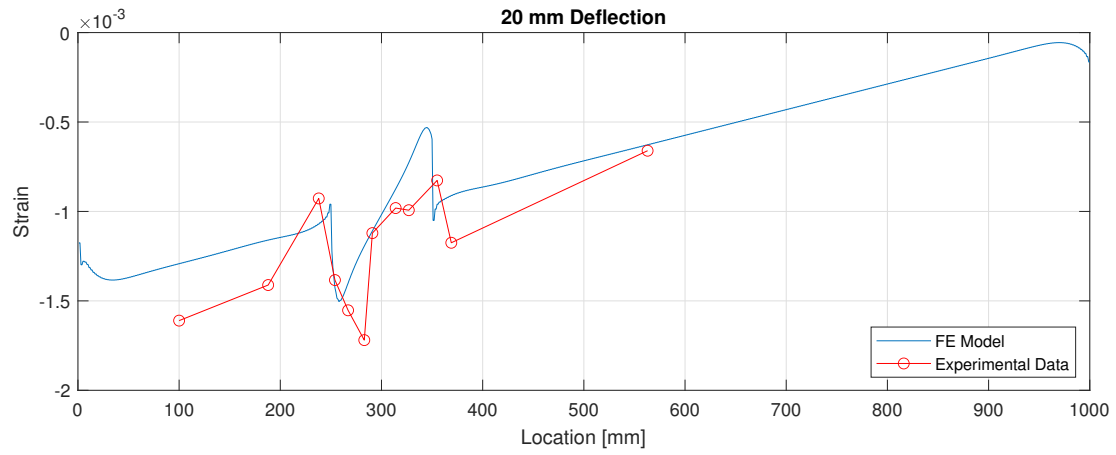


(a)

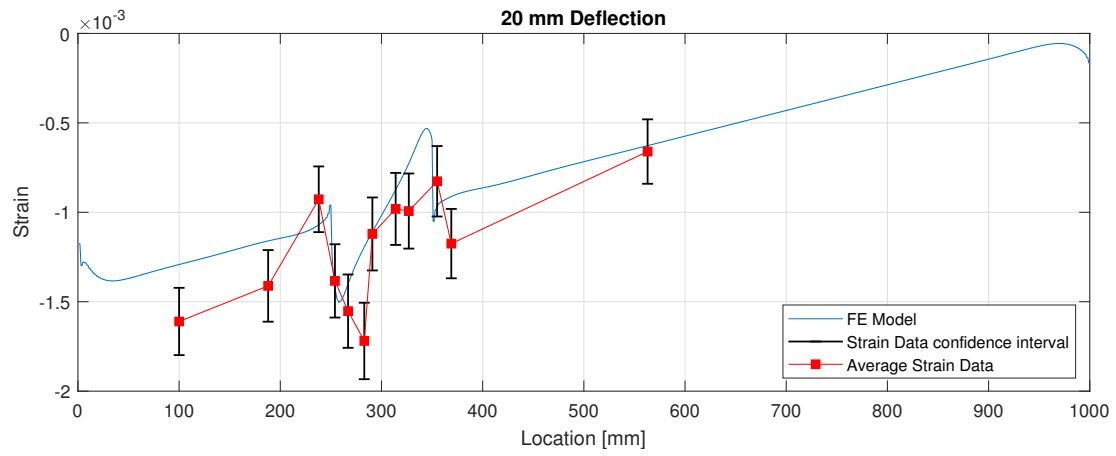


(b)

Figure 2.40. FE Model's strain versus strain from experimental data for a prescribed tip displacement of 10 mm: (a) average experimental strain and (b) average experimental strain and it's $\pm\sigma_\epsilon$ confidence intervals.

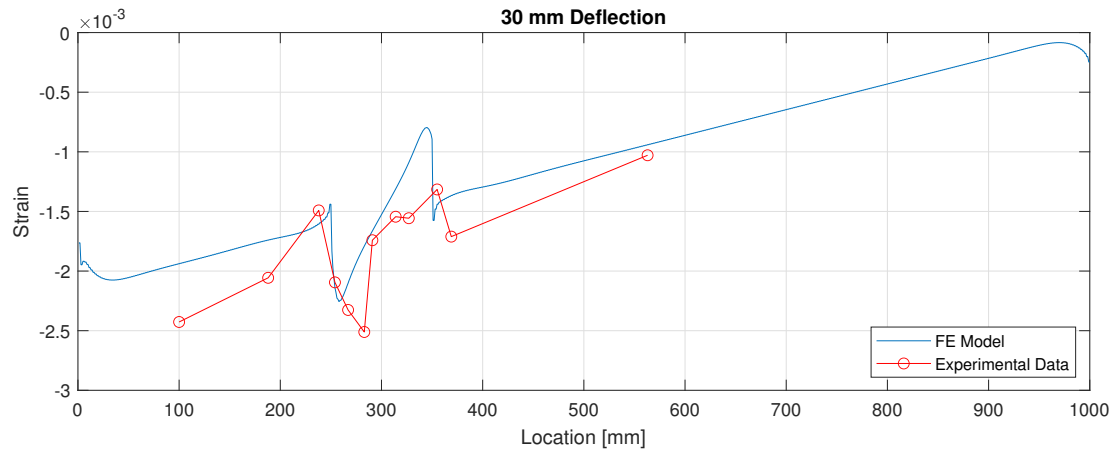


(a)

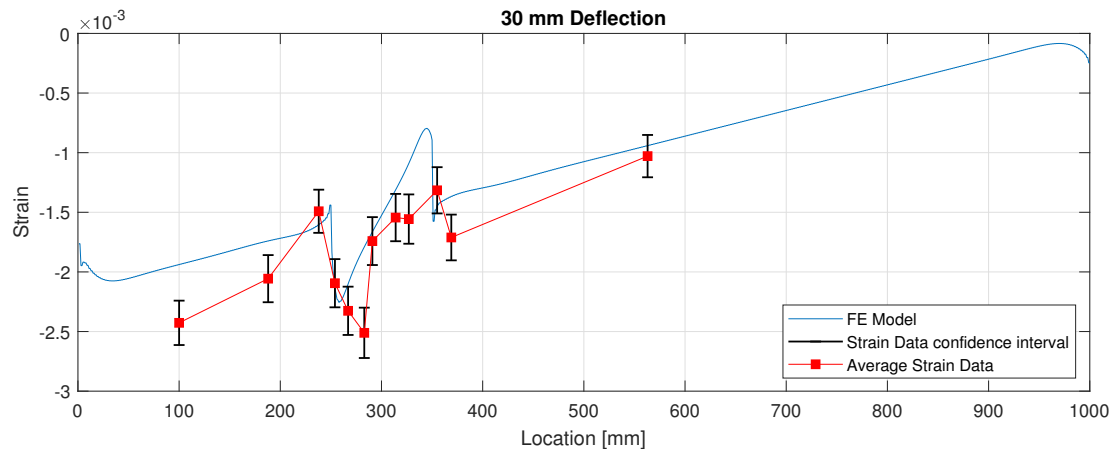


(b)

Figure 2.41. FE Model's strain versus strain from experimental data for a prescribed tip displacement of 20 mm: (a) average experimental strain and (b) average experimental strain and it's $\pm\sigma_\epsilon$ confidence intervals.

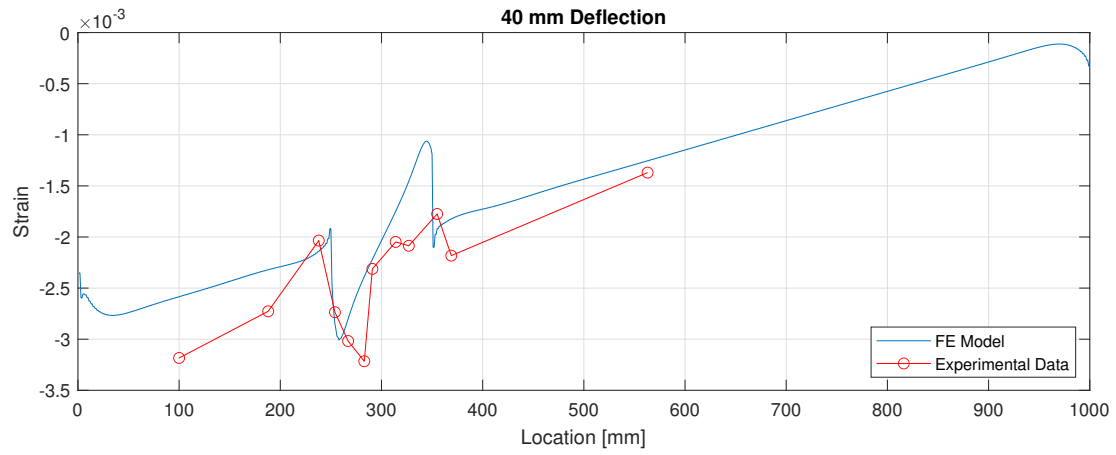


(a)

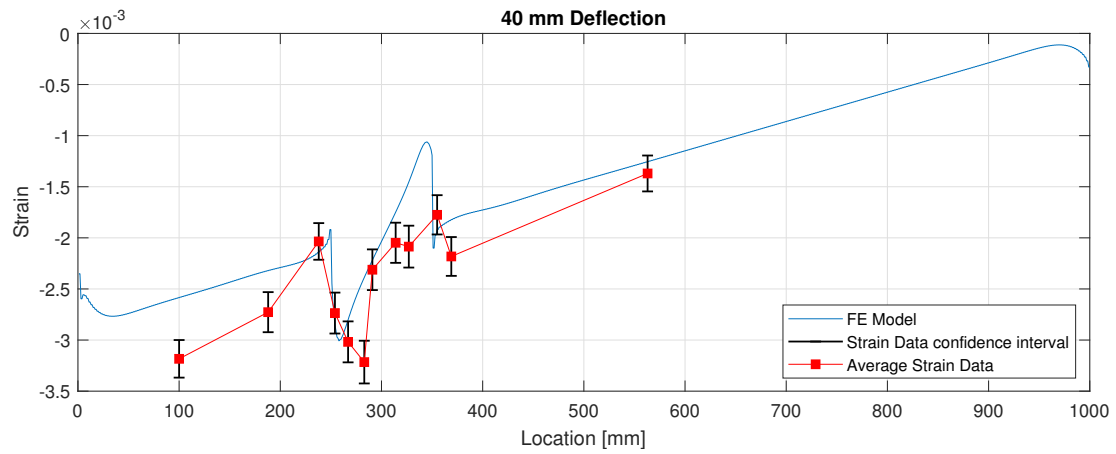


(b)

Figure 2.42. FE Model's strain versus strain from experimental data for a prescribed tip displacement of 30 mm: (a) average experimental strain and (b) average experimental strain and its $\pm\sigma_\epsilon$ confidence intervals.



(a)



(b)

Figure 2.43. FE Model's strain versus strain from experimental data for a prescribed tip displacement of 40 mm: (a) average experimental strain and (b) average experimental strain and its $\pm\sigma_\epsilon$ confidence intervals.

Figure 2.40(a) overlays the strain extracted from the FE simulation and the strain estimated from the strain gauges measurements, and Figure 2.40(b) shows the $\pm\sigma_\epsilon$ confidence interval around each measurement. From Figures 2.40, 2.41, 2.42, and 2.43 it can be seen that the overall strain behavior on top of the damage region is very similar to the one modeled via FE. There are two peaks and two valleys that can be consistently observed on all load levels. The measured strain, however, does not align nicely with the modeled strain. There are several sources of uncertainty which can cause that behavior:

1. The material properties used to update the FE model. Although they were calculated analytically, no tensile test was performed to confirm them.
2. The discrepancy between nominal and measured dimensions of each laminate after manufacturing, that cause a difference in the area moment of inertia of the beam.
3. Uncertainty in the alignment of the strain gauges with the longitudinal axis. In the FE model, this is a perfect line.
4. Distance between the centroid of the strain gauge to the edge of the shear clips, versus the FE model.
5. Noise in strain measurements.
6. How the boundary conditions are applied in the model versus the experiment.
7. The quality or lack of bonding between the parts.

In addition, there were two strain gauges placed in the damage region that malfunctioned, one loss of adhesion and the other because of a broken wire. Therefore, there were two less points that could be used to discretize the strain behavior along the damage area.

Even with all these uncertainty sources, the experiment serves as validation to the FE model, as the expected strain behavior occurred. These uncertainties would need to be further investigated for model calibration. In addition, no critical load that may cause

local buckling was applied to guarantee that the beam could be used for more tests in the future.

2.6 Summary

This chapter presented a detailed finite element model of the composite I-beam used as a computational test bed for this dissertation. The pristine model was presented, and then damage was introduced by removing the tie constraint between parts, simulating a debonding damage. This debonding may cause local buckling at loads smaller than the ultimate service load, which is considered critical to the operations.

This model represents a wing's spar made of prepreg carbon fiber materials and manufactured according to all aeronautical standards. The model calibration and validation was performed using experimental data from Israel, and the model is trustworthy. However, this information is proprietary and cannot be shared in this dissertation. To overcome this challenge, an adapted version of the spar was built in-house for academic purposes, using the materials and manufacturing tools available at the educational laboratory.

All the steps for the spar's manufacturing were detailed. Although there is a discrepancy between the experimental and modeled strain data, this experiment suffice to prove that the strain field has the same behavior under damage as the one obtained in the simulations. Therefore, it can be said that the model can serve as a data emulator in the computational framework for the remaining chapters of the dissertation.

Chapter 2, in part, has been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., Tur, M. , & Yehoshua, T.,2023. The title of this paper is “The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application”. The dissertation author was the primary investigator and author of this paper.

Chapter 3

Data generation - Surrogate models

3.1 Statistical Modeling

After being validated, the finite element model serves as the fundamental “ground truth” forward model in this project, i.e., the “physics simulator” for the test article of interest. Inputs to this forward model are damage parameters (size and location) and the load. For given damage parameters and given load, the outputs are what is intended to measure in practice, which is strain.

The debonding damage is fully characterized by two random variables, namely damage size, σ , and damage location along the beam, x . Thus, the damage is probabilistically characterized by the joint probability density function $p(\sigma, x)$. Its size and location are assumed to be independent (a reasonable assumption, although stress concentrations from the loading profile might induce some correlation between size and location), then the joint density of the damage parameter factors is:

$$p(\sigma, x) = p(\sigma)p(x) \tag{3.1}$$

The sponsors of this project provided some prior distributions on damage size and location, based on non-destructive tests of the spars after manufacturing, which can be

written as:

$$p(\sigma) = \text{Rayleigh}(50) = \frac{e^{(-\sigma^2/5000)}\sigma}{2500} \quad (3.2)$$

$$p(x) = \text{Uniform}(0, L) = 0.001 \quad (3.3)$$

which are independently plotted in Figure 3.1.

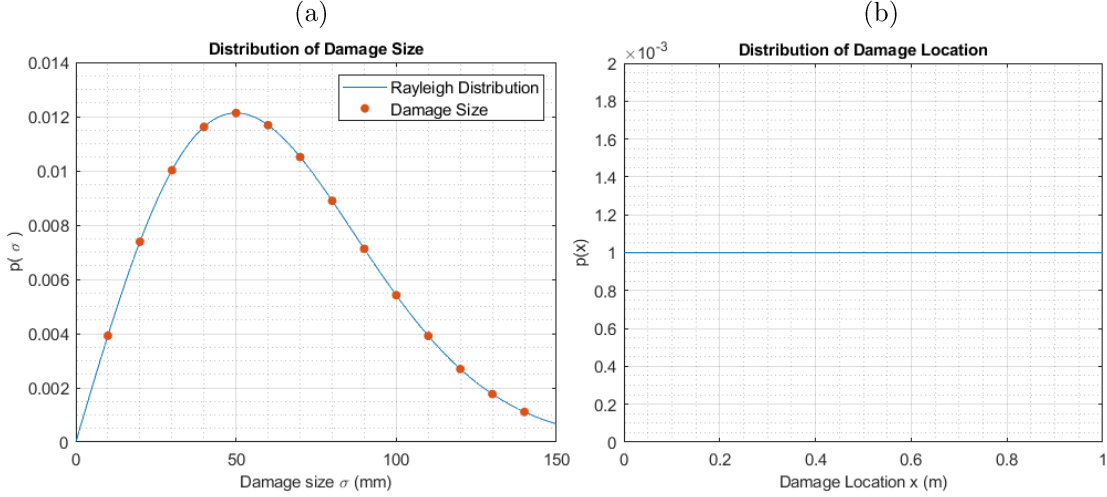


Figure 3.1. Prior distributions of damage size (a) and damage location (b).

The probabilistic load state was also defined. Given the static maximum service load, or limit load, shown in Figure 2.9, an equivalent maximum dynamic loading profile for a distribution of flight maneuvers (1 to 9) is found by dividing the global maximum service load (1.95 g) by the load factor corresponding to each case, from 1 to 9. Table 3.1 below reproduces previous load data provided by the sponsors and includes the effective maximum dynamic load factor obtained for each case, as well as the relative percentage of occurrence per flight.

This data can be fit to a probability density function model empirically, resulting in a form of probabilistic load $p(\mathbf{F})$ where bold face is used since the load state $\mathbf{F}$ will possibly depend on more than one parameter (i.e., however many are used to fit or otherwise appropriately characterize the data on Table 3.1. The specific fitting is not too important

Table 3.1. Dynamic load description.

Gust/ Maneuver type	Nz max (equivalent g)	Max load factor (Nz/1.95 g)	% Occurrences per flight
1	1.15	0.589744	81.697
2	1.25	0.641026	14.59
3	1.35	0.692308	2.85
4	1.45	0.74359	0.624
5	1.55	0.794872	0.158
6	1.65	0.846154	0.05
7	1.75	0.897436	0.019
8	1.85	0.948718	0.008
9	1.95	1	0.004

for formulating the problem but only relevant when actually solving the problem.

Thus, with the inputs specified now, the forward model can be written quite compactly as in Equation (3.4):

$$\varepsilon = h(\mathbf{F}, \sigma, x) + w \quad (3.4)$$

where ε is the output strain (at any location), and $h(*)$ is the finite element model (more specifically, the static step of the model) with probabilistic inputs of load state $\mathbf{F}$ and damage state characterized by σ and x . Some assumed random white noise, w is added to the finite element model, modeling measuring uncertainty in the output strain – this noise could be assumed Gaussian with zero mean and some standard deviation s_w .

Since $h(*)$ itself is complex, forward simulation mapping inputs to measurements is achieved by running the finite element model, sampling across all the probabilistic variables $\mathbf{F}$, σ , x , w . This is a very important step of this study and the execution of it will be discussed in Section 3.2 – running Abaqus with Python.

However, because failure is not determined by the presence/location of damage, but rather the effect on local buckling, two runs per probabilistic input sample point must be undertaken: one normal application of Equation (3.4) to get the strain ε , followed by a

buckling analysis to get the buckling eigenvalue λ , so Equation (3.4) is modified to:

$$\begin{aligned}\varepsilon &= h(\mathbf{F}, \sigma, x) + w \\ \lambda &= b(h(\mathbf{F}, \sigma, x))\end{aligned}\tag{3.5}$$

where $b(*)$ is the buckling analysis step applied to the finite element model h . From appropriate input space sampling, one obtains the distribution of ε while finding the corresponding buckling eigenvalue for the same input conditions. This is the critical link, since what is needed is the relationship between strain and the limit state for the buckling eigenvalue. If, somehow, the inverse of $b(*)$, $b^{-1}(*)$, is obtained, then Equations 3.5 are combined as

$$\varepsilon = b^{-1} + w = g(\lambda, w)\tag{3.6}$$

more explicitly showing that link. However, it is the inverse of Equation (3.6), $g^{-1}(*)$, that is wanted in practice – strain is measured in order to infer whether buckling has occurred or not. Probabilistic speaking, the goal is to find out the probability that buckling $p(\lambda)$ has occurred given measurements of strain:

$$p(\lambda) = p(\lambda|\varepsilon)p(\varepsilon) = p(\lambda|\varepsilon)p(\varepsilon|\mathbf{F}, \sigma, x, w)p(\mathbf{F})p(\sigma)p(x)p(w)\tag{3.7}$$

where it is explicitly assumed that load, damage parameters, and measurement noise are all independent.

The first conditional probability $p(\lambda|\varepsilon)$ on the right-hand-side is determined by the buckling analysis step of the FE model, and the remaining terms by static step, linear perturbation, of the FE model. Both require executions of the finite element model, which is time-consuming (around 15 minutes for a 1m-long submodel of the spar, 1 hour for the 5m-long spar model), to execute this framework.

It was decided that a surrogate model needed to be built in the forward direction

– such surrogate model will replicate the finite element model input/output to achieve sufficient number of Monte Carlo simulations with reasonable computational time. However, to build this surrogate model using supervised machine learning processes, enough data must be provided to build a trustworthy model. Therefore, the finite element model must be executed hundreds of times to generate training and testing data. In addition to the running time, it is also time consuming to remove the previous damage from the model and create a new one using the graphical user interface on Abaqus, so Python scripts were developed to execute Abaqus.

3.2 Running Abaqus with Python

The graphical user interface (GUI) is extremely helpful when creating, testing, and validating the finite element model. However, to generate data in bulk, it would be impossible to manually change the damage size and location hundreds/thousands of times, so Python scripts were written to help with this task.

These scripts are not a commercial, simplified codes, but rather customized codes that took significant effort to develop and are unique to this FE model – the scripts were created specifically for this FE model, including how the parts, forces, surfaces, etc. were named in the model. They are by no means trivial, and they execute steps that the user would execute in the GUI in an automated manner.

First, a file containing N samples must be generated (each sample corresponds to different damage sizes and locations to be simulated, one after the other). It was determined that the loads should be kept fix at the maximum service load of 1.95 g (Figure 2.9), so they are not a varying input to the simulation. Given that the model is linear, results for any other load level (other maneuvers) can be obtained by scaling them using the maximum load factor from Table 3.1.

With this file ready, the Python scripts do the following in a nested loop:

- Open the FE model on Abaqus on the background – no user interface is displayed.
- Look for partitions from previous damage and delete them.
- Select the damage size and location from the file. Create new partitions on all the surfaces based on those damage parameters.
- Re-assign masters and slaves for tie constraints.
- Re-assign local seeds on parts that were modified and re-generate meshes.
- Confirm that the stacking direction for material property is correct.
- The model already contained all the necessary information that is constant– material properties, geometry, boundary condition, loads, constraints, etc. Therefore, with the changes from the inputs, it is ready to run.
- Run the job.
- Save eigenvalues.
- Create paths along the beam to extract strain data (Figure 2.22). The location of each path is easily changed, if needed.
- Save strain data ($\varepsilon_{11}, \varepsilon_{12}, \varepsilon_{22}$) - 1 file per path, per sample.

It then moves to the next sample from the file and execute the same process until all N samples are executed. The process is summarized in a flow chart below (Figure 3.2).

This whole process runs Abaqus non-stop for days or even weeks, depending on the number of samples N, unless manually interrupted. In case there is an issue with any of the samples and the job could not be executed (e.g., invalid damage location), Python displays a message and moves on to the next sample. If the computer needs to be used for other computationally demanding tasks, it is easy to stop the Python scripts and later continue the simulations from the sample it stopped, without any information loss.

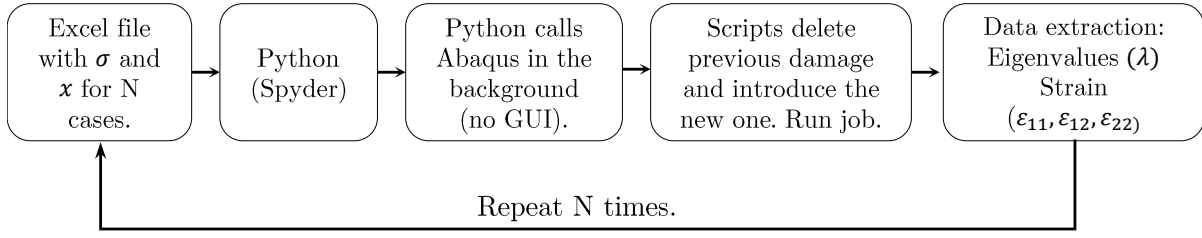


Figure 3.2. Flowchart of how Abaqus is executed using Python scripts.

3.2.1 1 meter sub-model

In order to obtain an initial understanding of the spar's behavior under damage and develop initial tests for surrogate models, an 1-meter long sub-model of the spar was generated, following the same procedures highlighted in Chapter 2. Shear force and bending moment diagrams were used to find the equivalent loads that should be applied to the model of the first 1m section of the beam, given the maximum service load of the entire beam provided in Figure 2.9 and Sub-section 2.1.4. The loads applied to the 1-m model were as follows:

- $q = +13729 \text{ Pa}$ is a uniformly distributed load, applied on the surface of the beam's bottom cap, representing the sum of its self-weight and an outside pressure.
- $P = +1765.08 \text{ N}$ are concentrated forces applied to the beam at the free-end (1m).
- $M = 2059.26 \text{ N.m}$ is a counter-clockwise pure moment applied at the free-end (1m).

This 1 meter model was used to generate the results that will be discussed in the upcoming sections.

3.2.2 Creating sampling distribution

Prior distributions for the damage size and damage location were provided in 3.1. In practice, the damage cases are limited by 2 things to be implemented in the FE model:

1. Mesh size in the longitudinal direction is of 2mm on the caps and cores.

2. Boundaries of the beam: damage location+damage size $\leq$ 1.0 m (or 5m for entire beam), considering that damage location is where it starts, not the center of the damage.

Considering these 2 factors, it was defined that the damage location should vary in 2mm steps, to avoid creating unstructured meshes that could cause conflict with the Python scripts. The damage size was limited to 10mm steps to decrease the number of possible combinations and provide changes in the strain field that are more observable, instead of a 2mm increase.

To create the sampling distribution, a space filling technique called Latin Hypercube sampling (LHS) was used. It is a statistical method for generating a near-random sample of parameter values from a multidimensional distribution [62]. For the first "long run" (running the model N times using Python), also known as first batch, $N = 800$ samples were created. The histograms of the damage parameters obtained using LHS are shown in Figure 3.3. It is easy to see that the damage size follows the Rayleigh distribution, and the damage location distribution is nearly uniform, considering the boundary constraint at 1m.

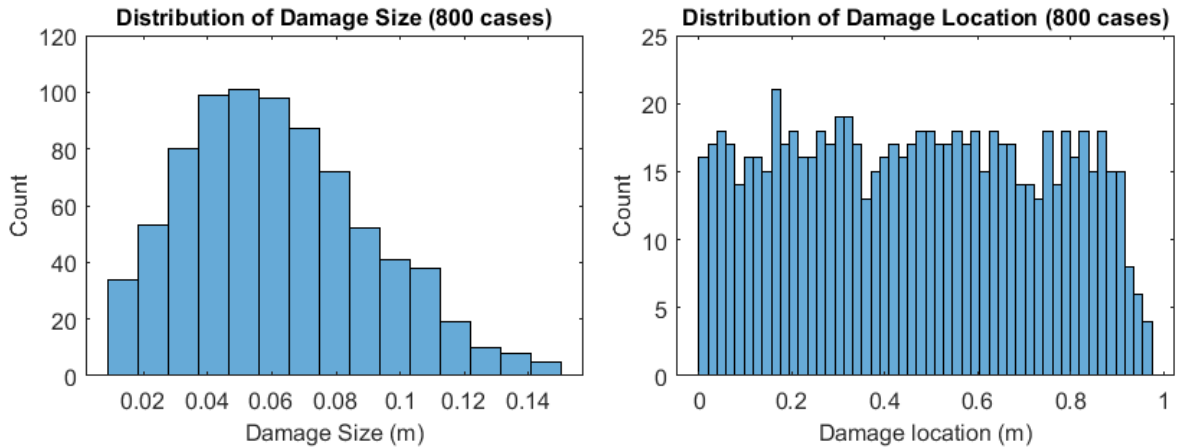


Figure 3.3. Histograms of damage size and damage location generated using Latin Hypercube sampling for 800 samples.

The sampling space for these 800 samples is shown in Figure 3.4 (left). These distributions are also plotted as a 3D histogram on the right. By observing these images, it is clear that some damage locations have much less samples than others, especially because the sum of the damage size and its location cannot exceed 1m. Lack of samples for some damage sizes is also evident, as they follow the given Rayleigh distribution, so there are not many cases bigger than 0.11m. Therefore, to fully represent the damage scenario, it was necessary to run more samples on Abaqus. This new batch was set with $N = 2,000$ samples.

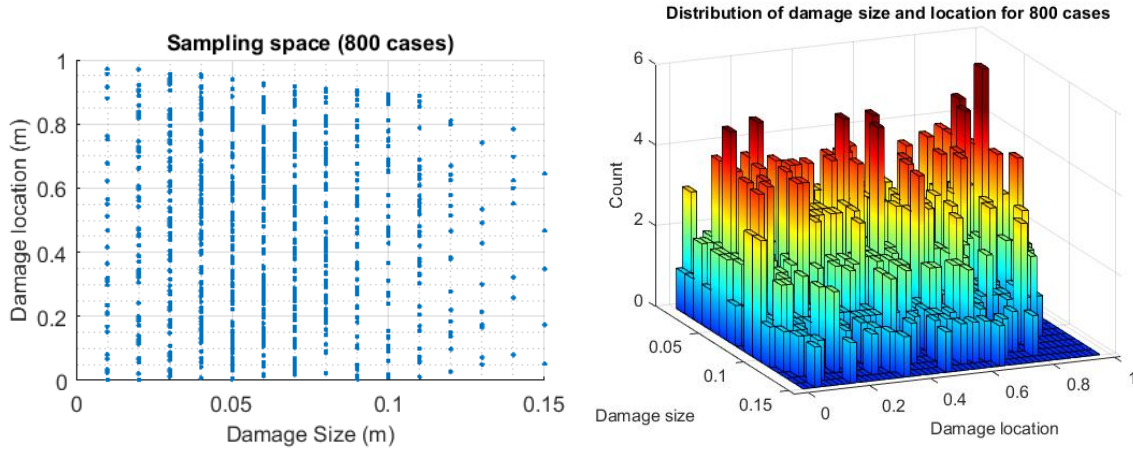


Figure 3.4. Sampling space for the first long run of the 1m beam.

To generate enough data to feed the machine learning models, it was defined that these 2,000 new samples would become a training set, and the previous 800 cases would serve as a test set, with no overlap between the two sets. However, running 2,000 cases takes about 2 weeks if running non-stop in the laboratory's workstation, thus the results from the previous 800 cases were analyzed to make a well-informed decision when creating the new data set. Some of these results are shown in Figure 3.5.

First, it is shown that a lot of the damage cases resulted in eigenvalues larger than 2. These eigenvalues occur for small damage sizes ($\sigma \leq 0.03m$), which were heavily sampled. Recall that in this buckling analysis the structure was loaded with the maximum

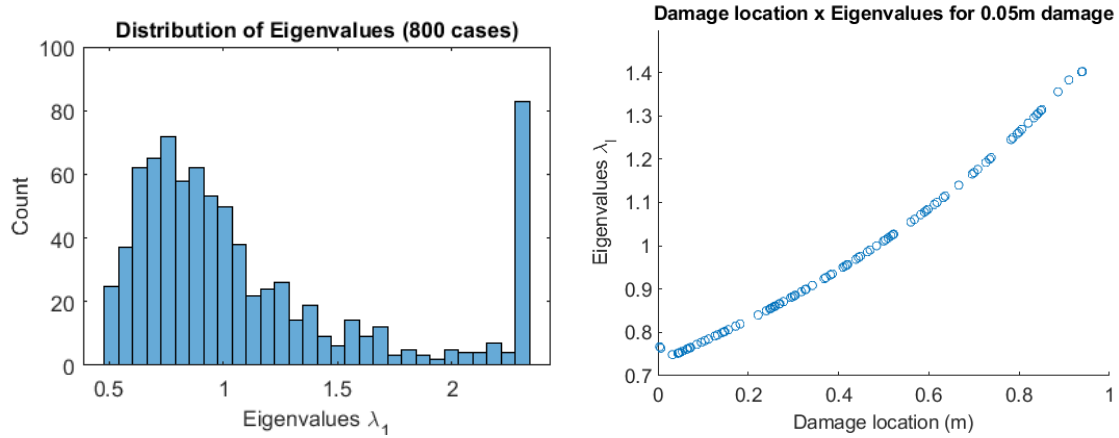


Figure 3.5. Results of the first long run (800 cases) - Distribution of eigenvalues (left) and eigenvalues per damage location for a 0.05m damage size (right).

load it is expected to have, and that the eigenvalue (λ) is the percentage of the load that the structure can carry without failure (local buckling). Thus, $\lambda = 2$ means that the structure can withstand 200% of the applied load without failing.

Even though it is very important that the forward model represent all damages with fidelity, damage cases smaller than 0.03m are not critical to the structure. On the other hand, damage cases bigger than 0.10m are extremely critical and will cause failure under full loads, no matter their location. Therefore, although larger damage occur less often (according to the prior distribution shown in Figure 3.1), it is extremely important to be able to represent them with the forward surrogate model, as the cost of missing this information would be extremely high.

The same idea applies to the damage location, as it is shown in Figure 3.5 that for a fixed damage size (0.05m), damages closer to the root (0 m) are more critical to the structure ($\lambda \leq 1.0$) than damages further away. This is true especially for the 1m beam, as in the 5m beam, the 1m and 2m sections are very damage-sensitive because of the concentrated loads applied in those regions.

Therefore, instead of sampling the new data set with the given prior distributions, a new distribution was created for the damage size for the second long run, that focuses on

larger damages. The damage location was kept as a uniform distribution. The sampling also considers the previous samples generated, so that there are no repeated cases, as this could cause over-fitting. The new PDF is shown on the upper left corner of Figure 3.6. The samples generated using this distribution are shown on the upper right corner. The sampling space for the 2,000 samples are shown in the lower right corner, where it is clear that the number of large damage cases ($x > 10cm$) is much higher than on the previous set of 800 cases.

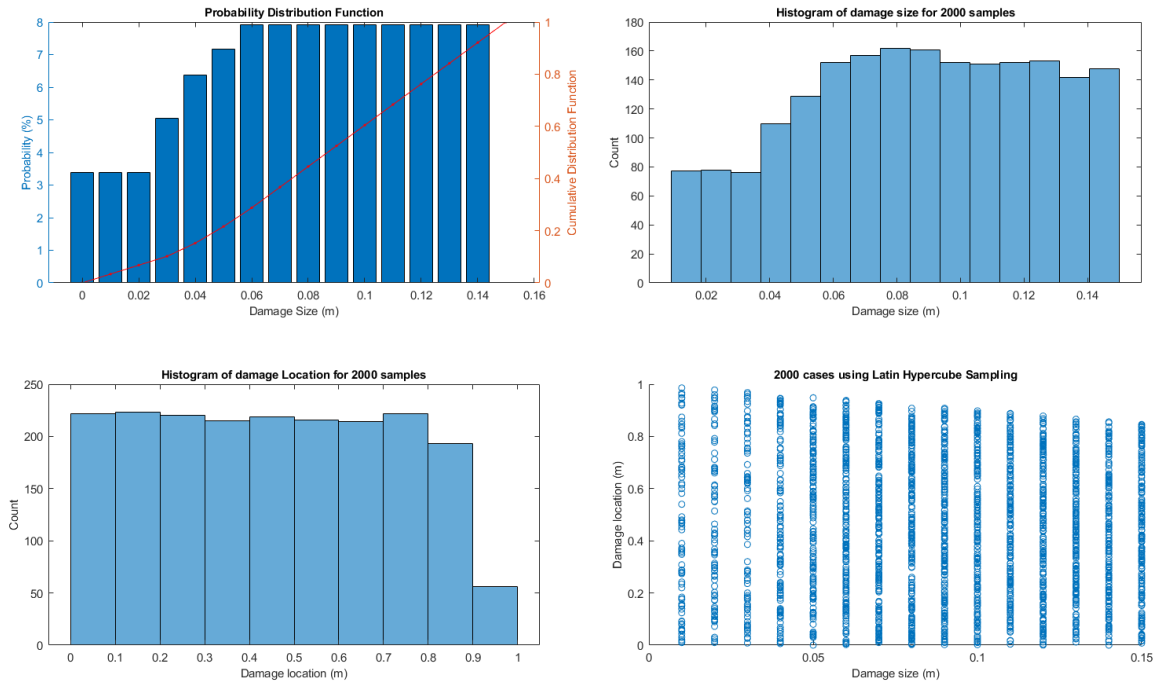


Figure 3.6. New samples for the second long run, with a new damage size distribution.

The eigenvalue results from the buckling analysis of the second long run are shown in Figures 3.7 and 3.8. These figures emphasize what was stated before: $\sigma \leq 0.03m$ do not cause $\lambda < 1$, independently of their location along the spar, and $\sigma \geq 0.10m$ always provide $\lambda < 1$, i.e., they are critical. The other damage sizes may be critical or not, depending on their location.

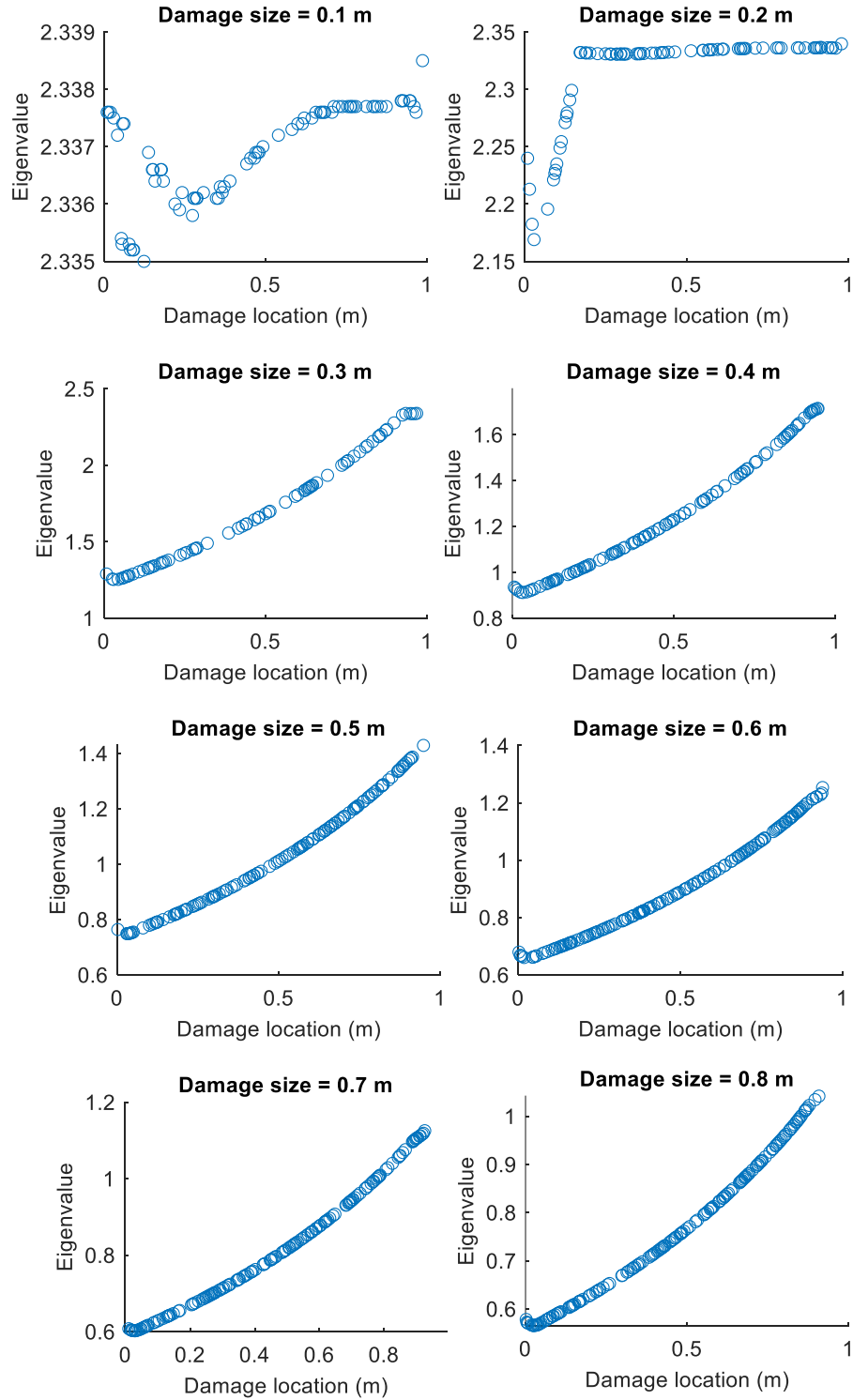


Figure 3.7. Eigenvalue results from the buckling analysis of the second long run (part 1 of 2).

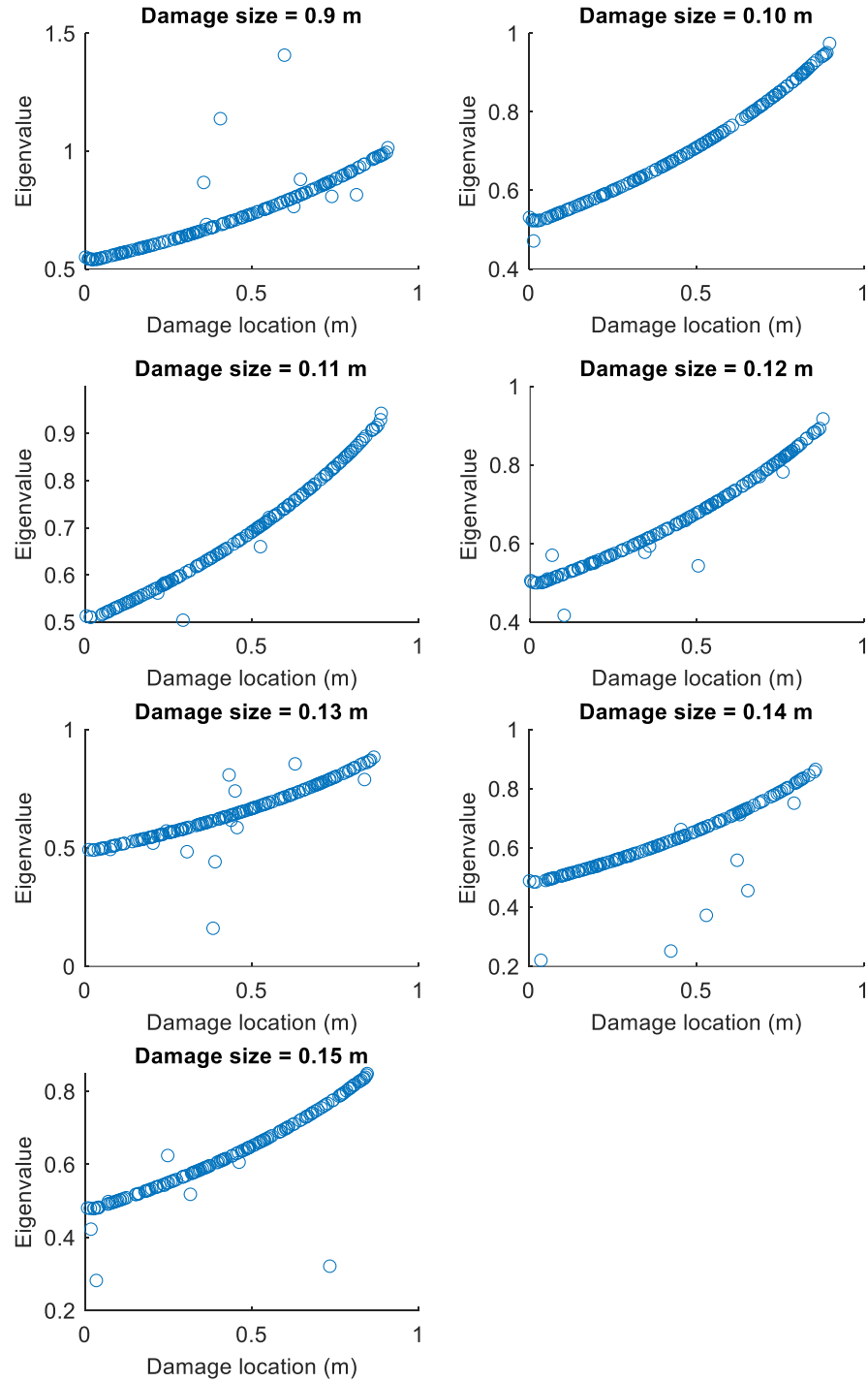


Figure 3.8. Eigenvalue results from the buckling analysis of the second long run (part 2 of 2).

3.2.3 Data assembly

As mentioned on Section 3.2, the scripts create paths along the beam to extract strain data after the static analysis on the FE model. There are 9 paths shown in Figure 2.22 and an extra one at the middle of the bottom cap. Strain data $(\varepsilon_{11}, \varepsilon_{12}, \varepsilon_{22})$ is extracted continuously along these paths, at the nodal points (e.g., the beam is 1m long and the cap's mesh is of 2mm, so $1000mm/2mm = 500$ nodal points).

The scripts extract strain data at the nodal points and saves it into three columns, creating one file per sample, per path. This generates $10 \times N$ files. This data needs to be reorganized into an acceptable format for machine learning, so another code was generated to assemble the data from all samples, creating 1 file per strain type, per path (ε_{11} path 1, $\dots$, ε_{11} path 10, ε_{12} path 1, and so on). These new files contain N rows, that refers to the samples (observations), and M columns that refer to the strain value at nodal points (features). This process is illustrated in Figure 3.9.

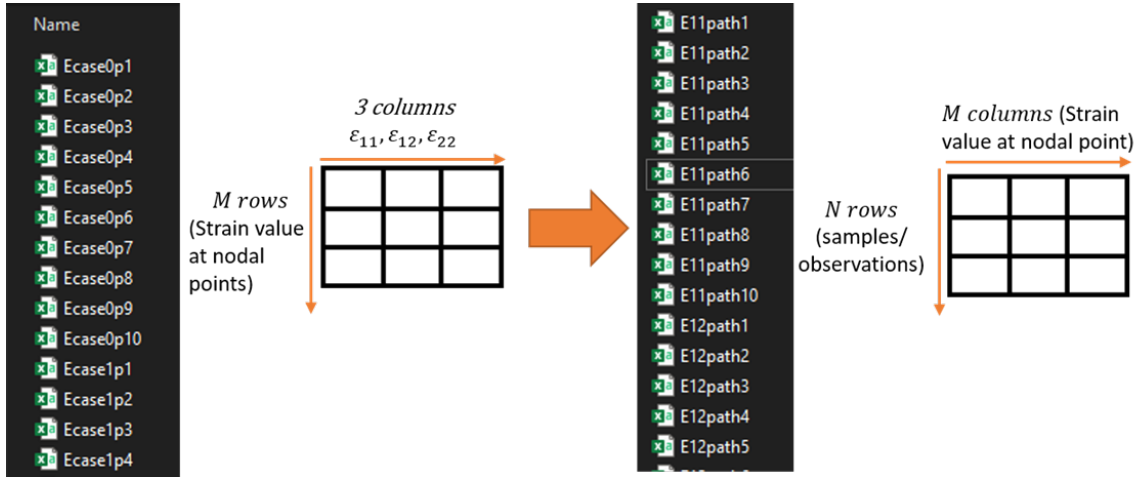


Figure 3.9. Data assembly process. Left: files extracted from scripts. Right: files after assembly.

3.2.4 Two-step outlier detector

It is rare, but after long-hours running non-stop, the workstation may get too hot and thermal throttling [63] may occur. If this occurs, it was observed that either the

strain measurements or eigenvalue results have an atypical behavior (Figure 3.10). To avoid inserting this data into the machine learning algorithms, a 2-step outlier detector was developed using a moving average of 5 points. First, it is applied to the assembled strain data. Then, in a nested loop, the eigenvalues are separated by damage size and the eigenvalue detector is applied to each group of eigenvalues. Figure 3.11 shows an example of the outlier detector applied to 15cm damages at all locations. The problematic samples found by the outlier detector are saved to a file, and the user can run these cases again by inserting this file to the Python scripts.

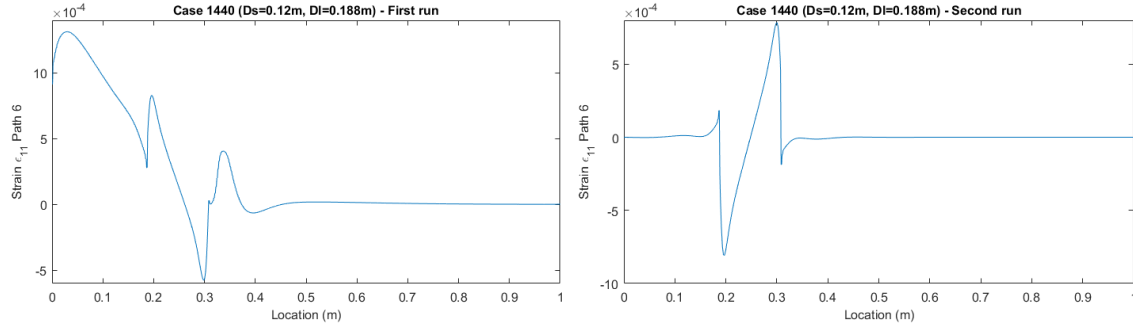


Figure 3.10. Left: example of an atypical strain behavior found by the outlier detector. Right: same sample after another simulation, with the correct result (baseline signal removed, so there is no slope on the strain measurement).

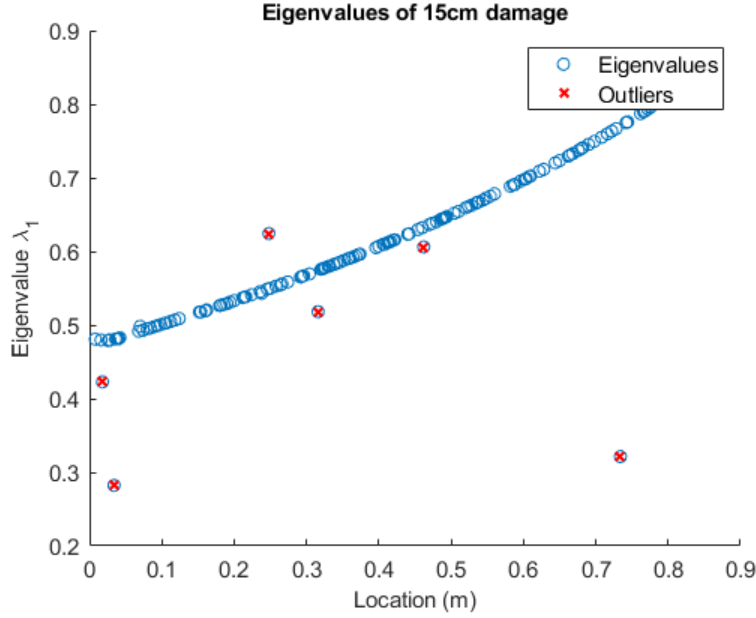


Figure 3.11. Outlier eigenvalues found by outlier detector

3.2.5 Data fusion

After running all the necessary cases, a data fusion script is executed to assemble data from all batches (e.g., 800 cases from the first long run and 2,000 cases from the second long run, plus any other case executed separately). Then, it looks for repeated cases within and removes them from the matrix, and the data is reorganized by damage size and damage location for ease of use. Finally, it must be decided which path and strain type will be used to create the machine learning models. This entire workflow is summarized in Figure 3.12.

To help decide which path to work with, the strain data for different damages were plotted for all paths – a sample of this is shown in Figure 2.23, where strain ε_{11} collected for $\sigma = 0.05m$ and $x = 0.704m$ is plotted. The y-axis was kept fix to all plots, to compare the amplitude of the signal at different locations. This was repeated for ε_{22} (Figure 2.24) and ε_{12} (not displayed), for a variety of sizes and locations.

From this analysis, ε_{11} for paths 6 and 8 (symmetrical) and path 9 seemed like

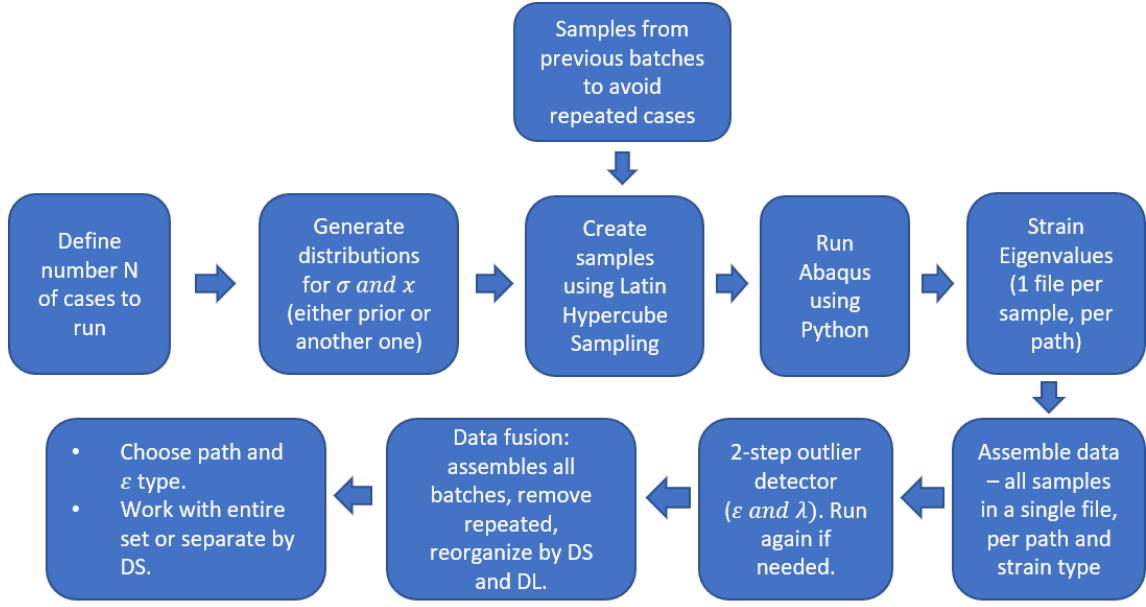


Figure 3.12. Flowchart of data fusion process.

strong candidates, along with ε_{22} from these paths. Then, two factors were taken into account:

- The damage of interest is debonding between the top cap and both shear clips and web. Therefore, either paths 6 or 8 seemed like suitable options to monitor this damage. Given that they are symmetrical, there is no preference between them.
- Placing the FBG sensors in the orthogonal direction to read ε_{22} would force the fiber to be bent multiple times along the beam, which would result in Bragg transmission loss [64]. Thus, it is desired to maintain the fiber with as minimal bending as possible, and measuring ε_{11} seems more feasible than measuring ε_{22} .

Therefore, a machine learning model was built for ε_{11} measured at path number 6, and it will be explored in the upcoming sections. Figure 3.13 shows the original strain measurement extracted from path 6, and the same measurement with the baseline subtracted, where the baseline is the strain measurement from the pristine structure (damage free).

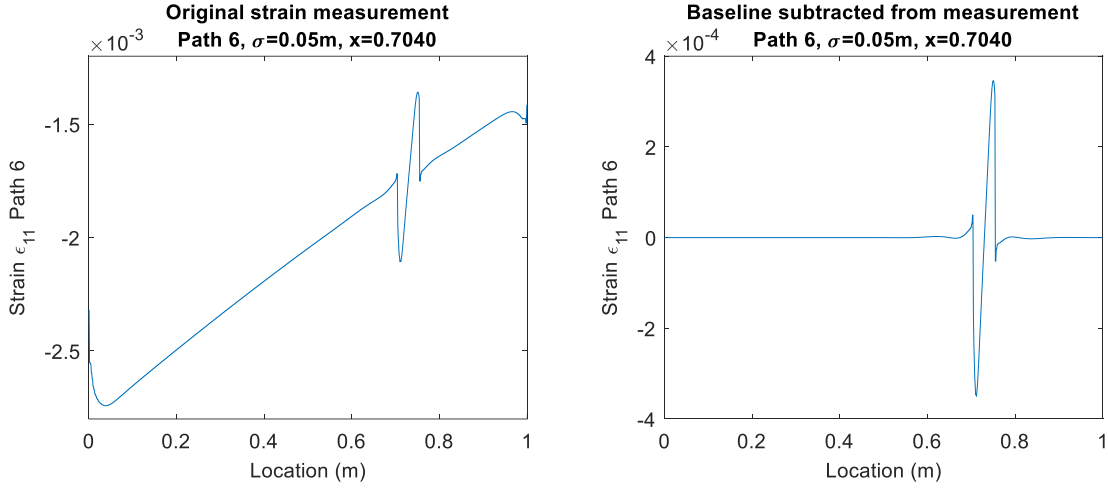


Figure 3.13. Measurements extracted from path 6, for damage size 0.05m and damage location 0.7040m. Left: original measurement. Right: Baseline subtracted from results.

3.3 Machine learning surrogate models

As mentioned in Section 3.1, a surrogate model must be created to replicate the finite element model input/output mapping to achieve sufficient number of Monte Carlo simulations with reasonable computational time. In fact, two models are necessary in the forward direction: one to replace the buckling analysis (output is eigenvalue) and the other one to replace the static analysis (output is strain data). The inputs to both models are damage size and damage location. This scheme is represented in Figure 3.14.

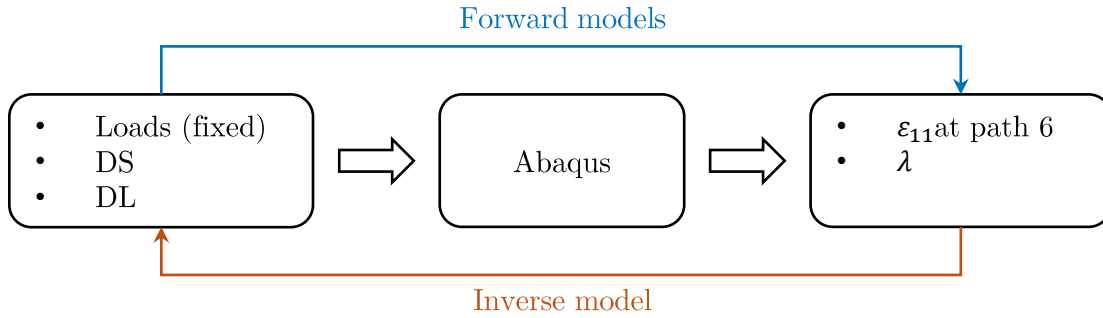


Figure 3.14. Representation of forward and inverse models created to replace Abaqus.

The inverse model works in the opposite direction: given the strain measurement,

the outputs are damage size and damage location. It does not make sense to create a model to predict the damage features given eigenvalues, as the eigenvalues are not measurable in practice. As explained Section 3.1, what is needed in the practical application is the relationship between strain and the limit state for the buckling eigenvalue – strain is measured to infer whether buckling has occurred or not. Probabilistic speaking, the goal is to find out the probability that buckling $p(\lambda)$ has occurred given measurements of strain. The models developed to achieve this goal are described in the following sections.

The initial models were created using data from the two long runs executed for the 1m beam. After removing repeated and problematic cases, the data set contained around 2,759 samples. As described in Section 3.3, the samples contained damage sizes from 1 to 15cm, and locations from 0.002m to 0.980m (respecting the boundaries), varying in steps of 2mm. The models created for the 5m beam will be based on the 1m models, as it is an extended version of the problem.

3.4 Surrogate model - eigenvalues

The first surrogate model developed was to predict the buckling eigenvalue, for the fixed loads, given damage size and damage location. Therefore, replacing Abaqus' buckling analysis. Eigenvalue data used for this model was shown in Figures 3.7 and 3.8, where the results were separated by damage size.

The first approach was to fit a polynomial to the eigenvalue results of each damage size. The results for the 15cm damages are shown in Figure 3.15, where the original data is shown in blue, and the fitted curve is shown in orange. The fitting worked well, with an RMSE=0.0411, without removing the outliers. However, if this approach was chosen, it would be necessary to create 15 models, as there are 15 possible damage sizes being analyzed. It is advantageous to have the lowest order model possible that captures the changes for all damage sizes, hence a Gaussian process (GP) regressor (GPR) was chosen

as the surrogate model.

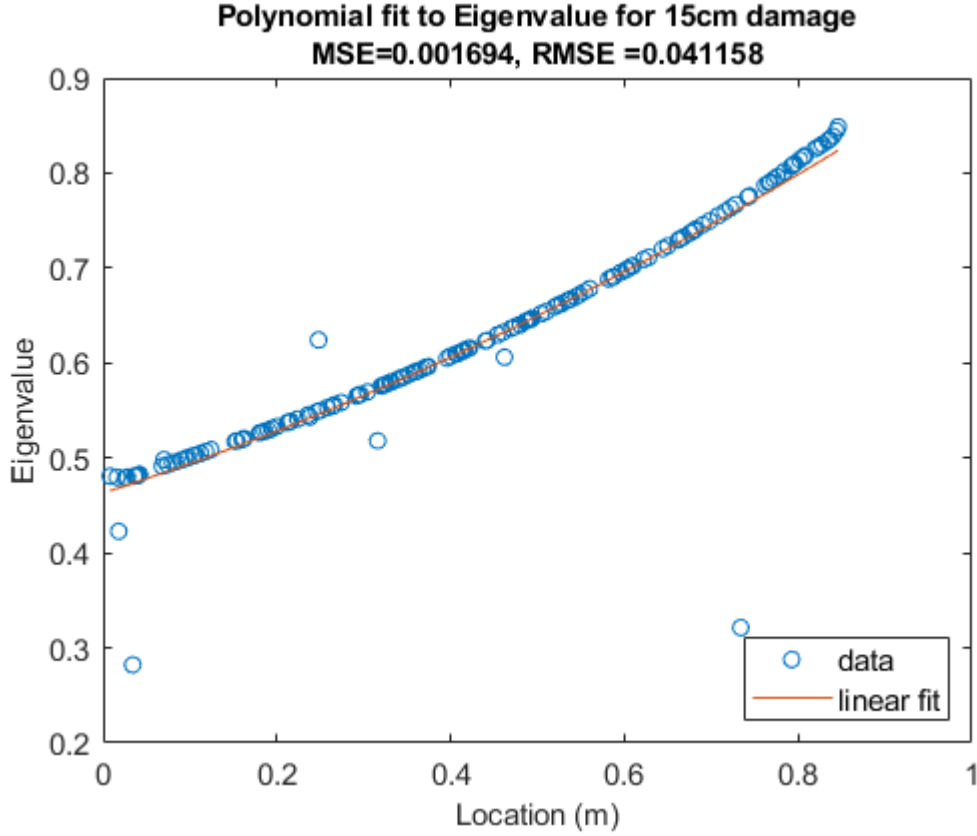


Figure 3.15. Actual (blue) and predicted (orange) eigenvalue from the polynomial fit for 15cm damage.

3.4.1 Gaussian process regressor

A Gaussian process regressor is a probabilistic model commonly used for regression tasks. It defines a distribution over functions, where each function represents a possible relationship between input and output variables. In simpler terms, a GP regressor provides a way to estimate a function that best fits the data, along with a measure of uncertainty for each prediction [65].

The key idea behind a GP regressor is that any finite set of function values should follow a multivariate Gaussian distribution. The GP is defined by a mean function, which gives the expected function value at each input point, and a covariance function (also

called a kernel), which specifies how the function values at different input points co-vary with each other.

Mathematically, a Gaussian process is defined by its mean function $m(x)$ and covariance function $k(x, x')$, where x and x' are input points [65]:

$$f(x) \sim \mathcal{GP}(m(x), k(x, x')) \quad (3.8)$$

For simplicity, let's consider a GP regressor with zero mean function (i.e., $m(x) = 0$). The covariance function commonly used is the squared exponential (SE) or Gaussian kernel:

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{(x - x')^2}{2l^2}\right) \quad (3.9)$$

where σ_f^2 is the variance parameter, which determines the vertical variation of the functions, and l is the lengthscale parameter, which determines the horizontal length scale over which the function varies.

Another commonly used covariance function is the exponential, defined by:

$$k(x, x') = \sigma_f^2 \exp\left(-\frac{r}{l}\right) \quad (3.10)$$

where $r = \sqrt{(x - x')^T(x - x')}$ is the Euclidean distance between x and x' .

Given a set of N training points $\{x_1, \dots, x_N\}$ with corresponding function values $\{y_1, \dots, y_N\}$, the GP regressor predicts the function value $f(x_*)$ at a new input point x_* as well as the uncertainty associated with this prediction.

The predictive distribution of the GP regressor at x_* is a Gaussian distribution:

$$p(f(x_*)|x_*, X, y) = \mathcal{N}(\mu_*, \sigma_*^2) \quad (3.11)$$

where X is the matrix of training inputs, y is the vector of training outputs, μ_* is the

mean of the predictive distribution, and σ_*^2 is the variance of the predictive distribution.

The mean and variance of the predictive distribution are given by:

$$\begin{aligned}\mu_* &= k_*^T (K + \sigma_n^2 I)^{-1} y \\ \sigma_*^2 &= k_{**} - k_*^T (K + \sigma_n^2 I)^{-1} k_*\end{aligned}\tag{3.12}$$

where:

k_* is the vector of covariances between the training points and x_* ,

k_{**} is the covariance between x_* and itself,

K is the covariance matrix between the training points, and

σ_n^2 is the noise parameter, which accounts for noise in the training data.

In summary, a Gaussian process regressor provides a flexible framework for regression, allowing for uncertainty estimation in predictions.

The first attempt on fitting the GPR model was done using the entire data set, but it was observed that the damage of sizes 1 and 2cm had an eigenvalue response too different from the rest of the data set. The eigenvalues per damage location for these two sizes are shown in 3.16. They were causing errors on the predictions of the smaller sizes, so they were removed from the data set.

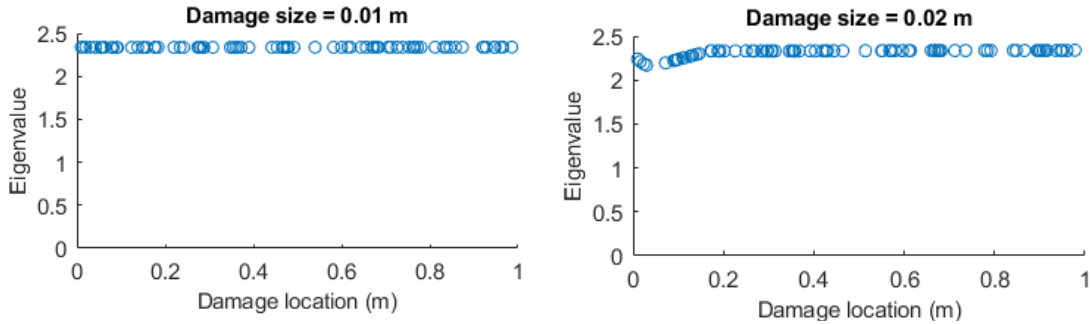


Figure 3.16. Eigenvalues per damage location, for damage size of 1cm (left) and 2cm (right).

The new data set contained around 2,500 samples, after removing the outliers and all $0 \leq \sigma \leq 2\text{cm}$. The train/test split was done using a 30% holdout, and the kernel function used was the squared exponential. The predictions for the eigenvalues forward model are shown in Figure 3.17, where the real data (from Abaqus) is shown in blue and the predicted data is in orange. The 95% confidence intervals are also calculated, but not shown in this picture to not make it overcrowded. The root mean square error (RMSE) is 0.0028, confirming that this surrogate model is an accurate choice to replace the buckling analysis of the FE model.

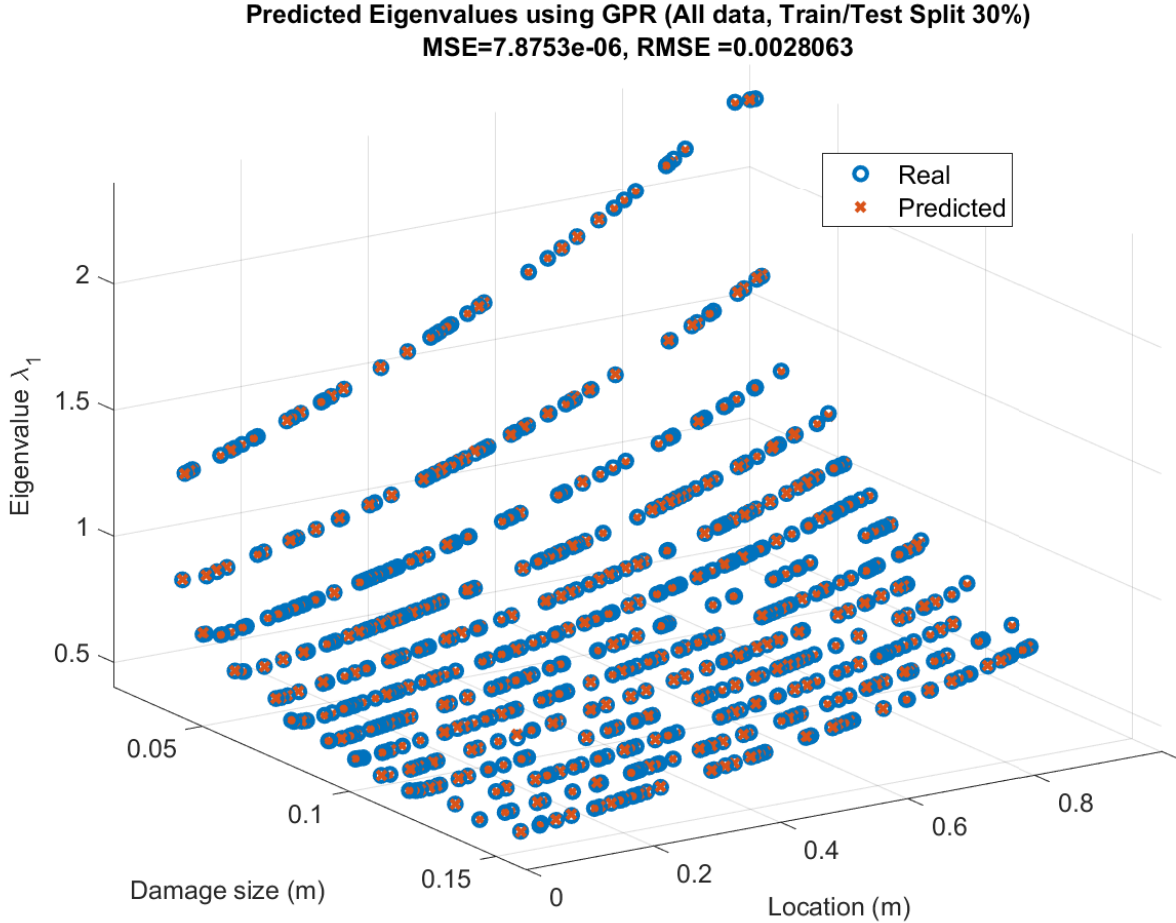


Figure 3.17. Eigenvalue predictions of the Gaussian Process Regression model, for damage sizes between 3 and 15cm for all damage locations.

As this model does not include damage sizes of 1 and 2cm, the polynomial fit idea

mentioned previously could be used for these two sizes. However, it will later be shown that these small damage cases are completely removed from the optimal sensor placement study, as they do not represent a critical damage in any given position.

3.5 Surrogate model - Strain measurements

As mentioned in the previous section, two models are necessary in the forward direction: one to replace the buckling analysis (output is eigenvalue) and the other one to replace the static analysis (output is strain data). The further is much more complex, as from only 2 inputs (damage size and damage location) it needs to predict measurements at 1,000 points (1 meter beam discretized with 1mm mesh size, or 5,000 points for the entire 5m beam). It was shown in 3.13 that strain ε_{11} from path 6 (below the top shear clips) was one of the most sensitive to damage, so the surrogate model will focus on the reconstruction of that path.

One alternative to develop this forward model would be to use a principal component analysis (PCA), by projecting each data point onto only the first few principal components to obtain lower-dimensional data while preserving as much of the data's variation as possible. However, as shown in Figure 3.18, at least the first 50 principal components would be necessary to represent 90% of the data. This would mean training 50 different models to predict each principal component and then putting them all together to reconstruct strain.

Several ways to reconstruct the strain measurement were tried, such as fitting a Lorentzian function, 2 Gaussian distributions, 4 Gaussian distributions, among others. Developing a matrix that mapped the relationship between the inputs and strain to be solved as a least squares problem was also attempted, but none of these alternatives provided a good fit. Thus, a new approach was taken.

That approach consists of the following:

- Remove the baseline from the signal.

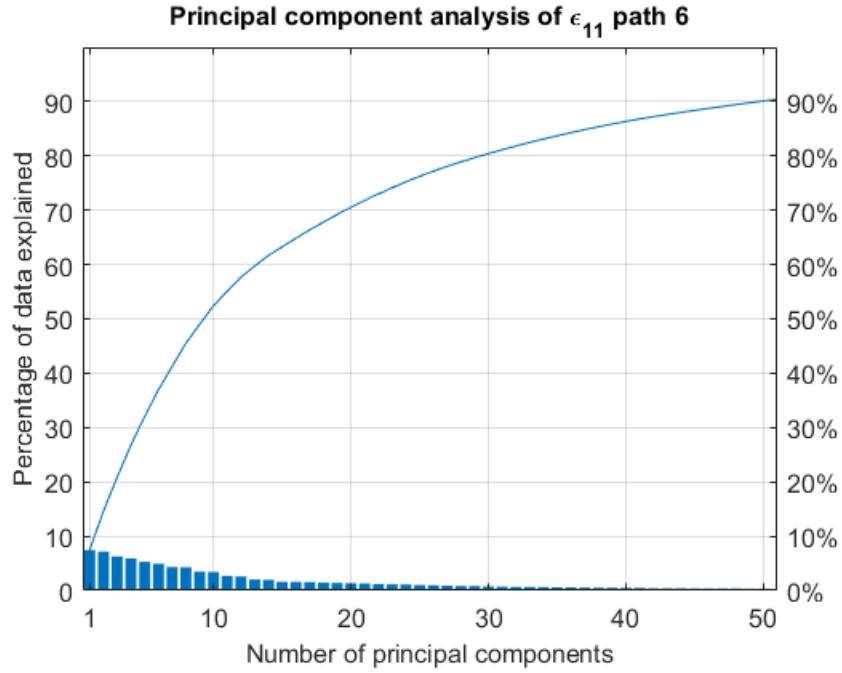


Figure 3.18. Principal component analysis of strain data for path 6.

- Find the points p1, p2, p3, and p4 (shown by the black arrows Figure 3.19), which are described by their (x, y) coordinates,
- Break the signal down into 5 segments (curves) based on these 4 points.
- Find the equations that accurately describe each segment for all damage sizes and locations to reconstruct the signal.

Finally, it is necessary to develop and train the models that can predict the x-y coordinates of each point for a given damage size and location. All these steps will be described in detail throughout the following sections.

3.5.1 Code to find the coordinates of points p1, p2, p3, and p4

The first step was to develop a code that would take in a strain measurement, remove its baseline, and output the x and y coordinates of points p1, p2, p3, and p4. While this may seem to be a simple task, not all strain measurements have peaks and valleys

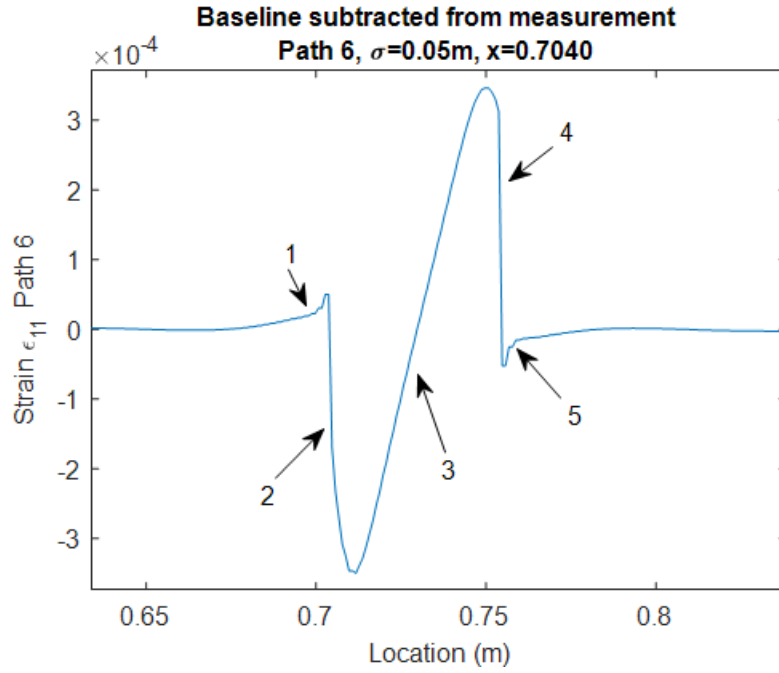


Figure 3.19. Strain measurement minus baseline broken down in 5 segments, with peaks and valleys highlighted by points p1 to p4.

as clearly as they are shown in Figure 3.18. A few different behaviors are exemplified in Figure 3.20. Plot (1,1) in the grid shows an example of the global valley having another local minimum point right next to it. Plot (1,2) does not really have a local maximum point p1 where the damage starts, it is a nearly flat curve. This also happens at plot (2,2), where points p1 and p4 are preceded and succeeded by a flat curve, respectively, and this is a common behavior for very small damages. Plot (2,1) shows a damage at the very end of the 1m beam, where the strain suddenly decreases at the last node of the mesh but point p4 was correctly identified.

Therefore, simply using Matlab’s “findpeaks” function would not be robust enough to find the peaks and valleys locations for all the different measurements. Also, one could claim that it works fine if you give the locations where “findpeaks” should look for the local maxima and minima, but this approach fails if you are blindly given a strain signal without any information of the damage location or if it is a noisy measurement.

To overcome these challenges, a Matlab function was written to first finds the

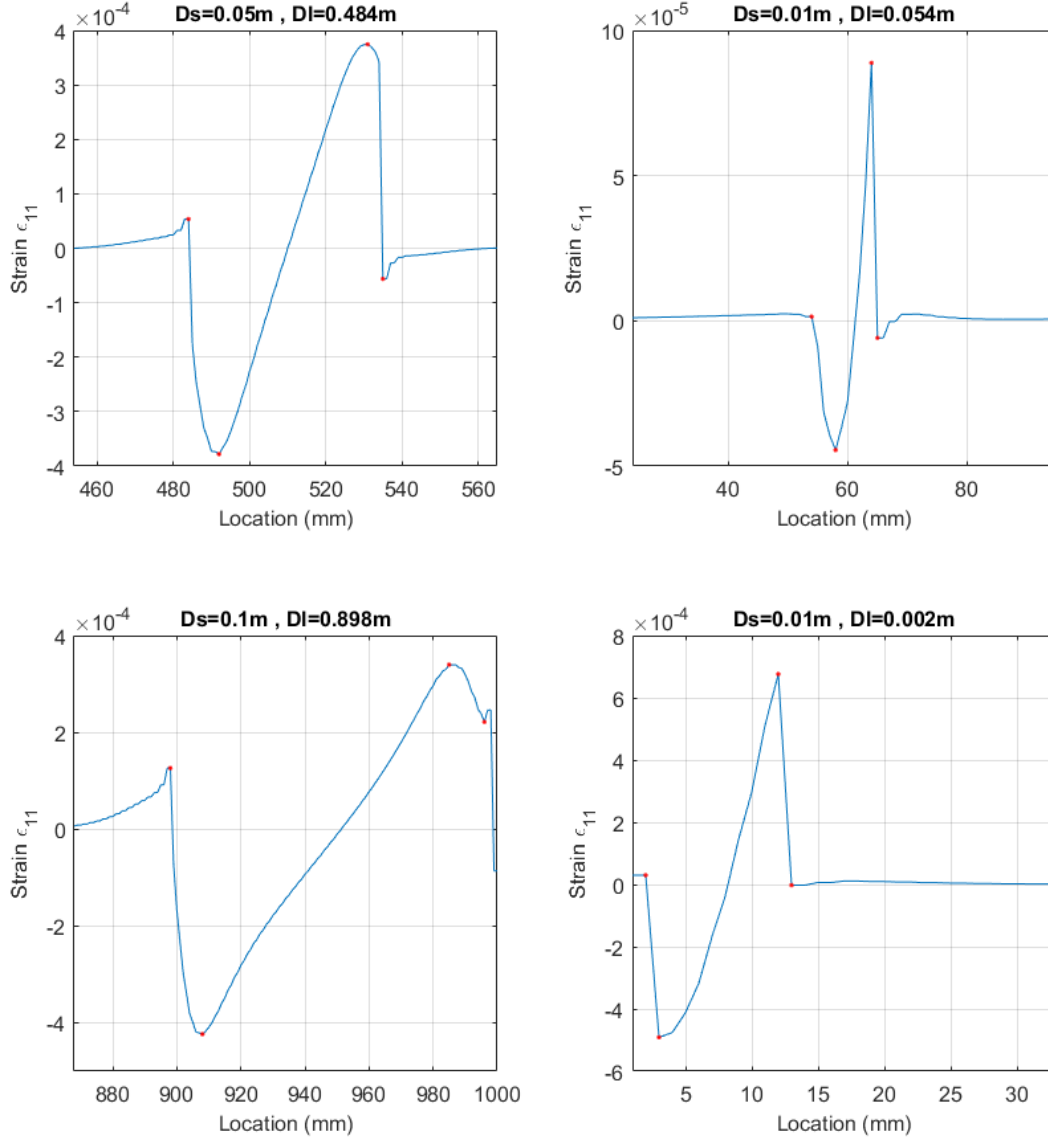


Figure 3.20. Samples of strain measurements for different damage sizes and locations.

coordinates of the local maximum (p3) and of the local minimum (p2). Then, from the local minimum, it moves back one point at a time while its amplitude keeps increasing. When it stops, it looks at ± 2 neighboring points to make sure that it reached an inflection point and marks it as p1. The same approach is done, in the opposite direction starting on the global maximum, to find p4. As shown in Figure 3.20, this method worked well for different strain measurements (confirmed by the red dots on the plots), and it was confirmed by checking all the different strain measurements. All this is done without any

prior information about the damage size or damage location.

To prove the robustness of the method, the function was applied to the same measurement as plot (1,1) from Figure 3.20, but adding white noise with zero mean and standard deviation 2010^{-6} . The result is shown in Figure 3.21. This could be used as a feature extraction tool in a real measurement (not created artificially).

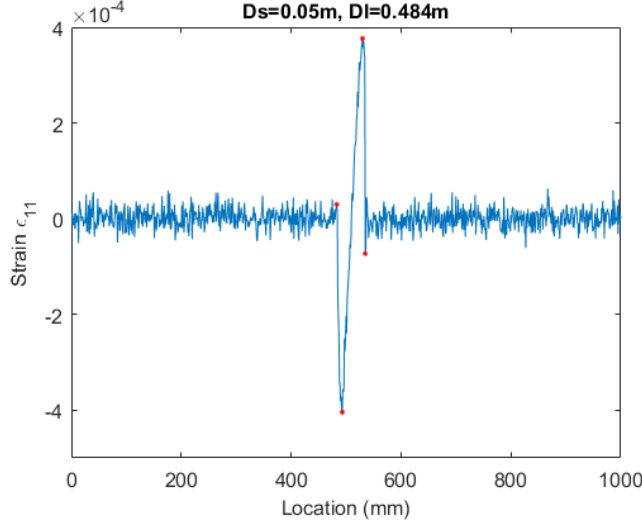


Figure 3.21. Example of function to find peaks being used under noise.

3.5.2 Strain reconstruction of each of the 5 segments

Before developing any machine learning model to predict the coordinates of points p1 – p4, a “proof-of-concept” was necessary to guarantee that it would be possible to reconstruct the entire ε_{11} measurement of path 6 given these coordinates. As shown in Figure 3.19, the signal was broken down into 5 segments, detailed in the middle column of Table 3.2.

The general equations that govern each segment are shown in Table 3.2. Recall that the goal is to develop a surrogate model to substitute Abaqus and serve as a data generator, so the error between the original and the reconstructed measurements must be below 1%. With that goal, there were no universal parameters a, b, c, N, M that worked

Table 3.2. Description of each of the 5 segments used for strain reconstruction.

Segment Name	From-to points	Equation of the curve
Curve 1	-N to p1	$f(x) = a \exp(bx)$
Curve 2	p1 to p2	$f(x) = ax^2 + bx + c$
Curve 3	p2 to p3	$f(x) = ax + b$
Curve 4	p3 to p4	$f(x) = ax^2 + bx + c$
Curve 5	p4 to +M	$f(x) = a \exp(bx)$

well for all damage sizes and locations. Thus, a function that adapts these parameters based on the x and y coordinates of points p1, p2, p3, and p4 was created. No other information is necessary.

For example, by looking at plots (2,1) and (2,2) from Figure 3.20, it is noticeable that N must be much larger for the damage far from the root and with an exponential curve #1, than it is for a damage starting 2mm away from the root and a flat curve #1. That behavior also affects the b parameter in the exponential equation. When looking at curve #2 for these two plots, it can be visually estimated that the a parameter in $f(x) = ax^2 + bx + c$ should be close to zero at plot (2,2), as it seems to be a flat curve, while plot(2,1) clearly has a parabolic behavior of curve #2.

After the strain is reconstructed for each of the 5 curves, the rest of the measurement that is not around the damage region is zero-padded to make sure that it has the same amount of points as the original signal, and the baseline measurement is added back to reconstruct the rest of the signal. The results for the reconstruction around the damage location and for the entire signal are shown in Figure 3.22.

The strain reconstructions of the 4 measurements shown in Figure 3.20 are shown in Figure 3.23. The RMSE value shown in the title of each subplot is calculated only around the damage region where the signal was reconstructed, to guarantee that this value is not underestimated by considering the region that was zero-padded, and it is in the order of magnitude of 10^{-6} . These results also show that the function that adapts the parameters of the equations works well for different damage sizes and locations, even if

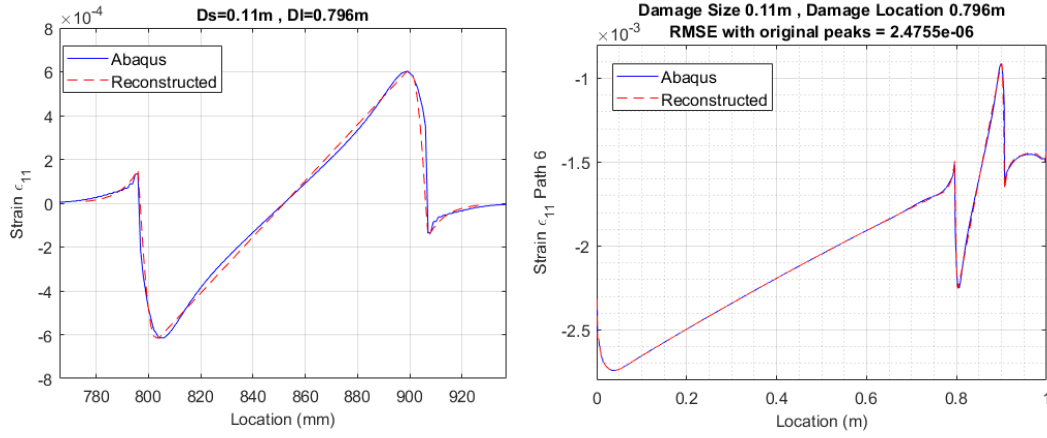


Figure 3.22. Strain reconstruction based on equations from Table 3.2. Left: reconstruction zoomed on damage location. Right: global strain reconstruction.

they are very close to the root or to the end of the beam.

3.5.3 Artificial Neural Network to predict coordinates of p_1 , p_2 , p_3 , and p_4 .

The strain reconstruction results shown in Section 3.5.2 were based on the coordinates obtained by applying the algorithm to find the coordinates on the strain measurements from Abaqus. However, the goal is to create a model that will predict these coordinates and reconstruct strain for a new data set, replacing Abaqus. Thus, a machine learning model was built to predict these 8 coordinates, given the 2 inputs (damage size and damage location).

Although this usually would require 8 different artificial neural networks (or any machine learning model of choice), one to predict each x coordinate and y coordinate of points p_1 , p_2 , p_3 , and p_4 , it is possible to perform a multi-output regression to predict two or more numerical values. Unlike normal regression where a single value is predicted for each sample, multi-output (or multi-target) regression requires specialized machine learning algorithms that support outputting multiple variables for each prediction. Deep learning neural networks are an example of an algorithm that natively supports multi-output

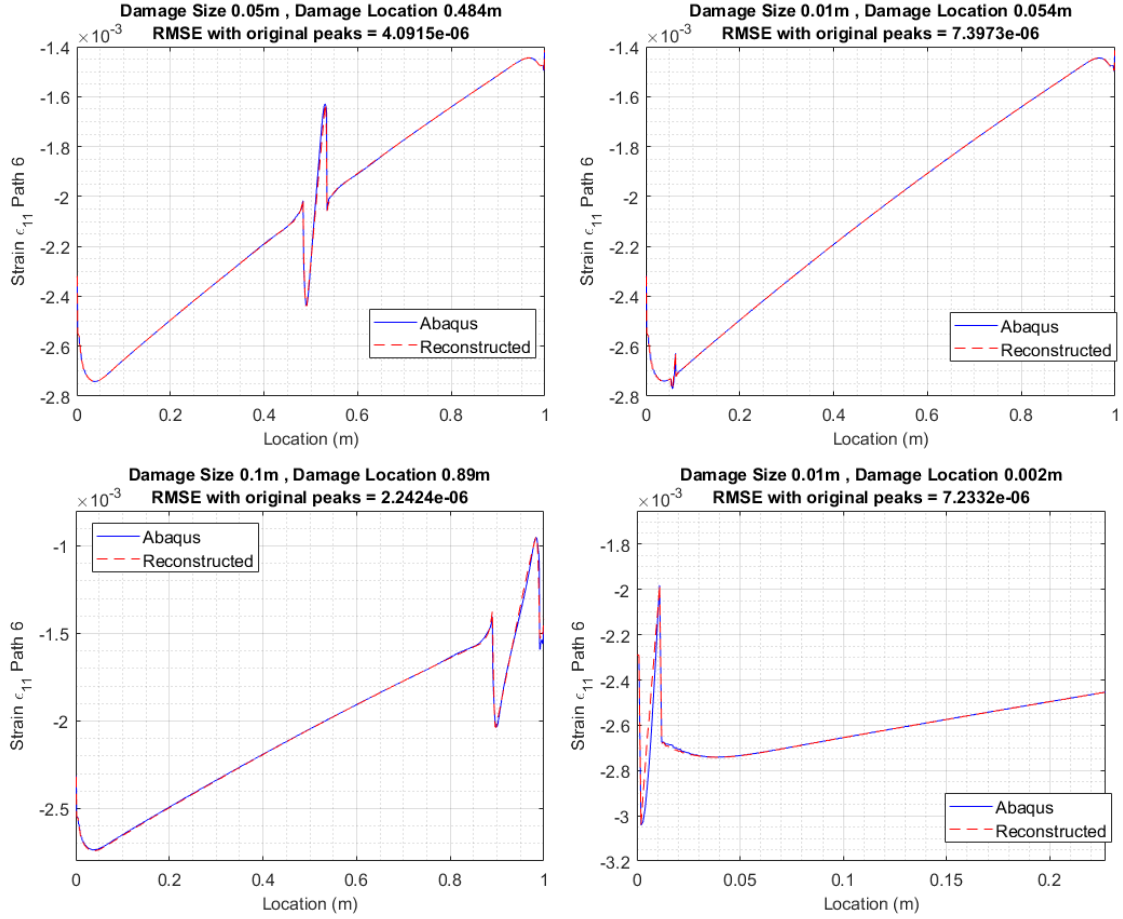


Figure 3.23. Reconstruction of strain measurements shown in Figure 3.20, based only on x-y coordinates of points p1, p2, p3, and p4. Plot (2,2) is zoomed in as the damage is too small.

regression problems.

An advantage of these multi-output approaches is that they might produce simpler models with improved computational efficiency [66]. A further advantage is that they might provide the means to effectively model the multi-output data sets by considering not only the underlying relationships between the features and the corresponding targets, but also the relationship between the targets, providing a better representation and interpretability of the problems [66, 67].

Neural network models for multi-output regression tasks can be defined and evaluated using Matlab's deep learning functions, which was the option of choice for this model.

These functions are already set to be executed in GPU arrays, accelerating the code by automatically running on a graphics processing unit (GPU) using Matlab's Parallel Computing Toolbox [™], as training deep networks is computationally intensive. These networks could also be implemented in Python using Keras deep learning library.

Preparing the data

As already mentioned, there are 2 inputs to the neural network, which are damage size and damage location, and 8 outputs: $x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4$, which are the x and y coordinates of points $p1, p2, p3$, and $p4$, respectively. Both the inputs and outputs were normalized, using the following equation so that their ranges were $[0, 1]$:

$$X_{norm} = \frac{X_i - X_{min}}{X_{max} - X_{min}} \quad (3.13)$$

Normalization helps stabilize and speed up networking training using gradient descent. If the data is poorly scaled, the loss can become 'NaN' (not a number) and the network parameters can diverge during training. Also, the damage size varied from 0 to 15mm while the damage location varied from 0 to 1000mm, so this guarantees that one parameter will not have more weight than the other when training the network. The same is valid for the outputs, as the x-coordinates were in the order of 10^3 while the y-coordinates were in the order of 10^{-4} .

After normalization, the data was randomly divided into 3 data sets, and there were no repeated cases among the entire set:

- Training data, consisting of 70% of the data.
- Validation data, consisting of 15% of the data.
- Testing data, consisting of 15% of the data.

The code trains the network on the training data and calculates the accuracy on

the validation data at regular intervals during training. The validation data is not used to update the network weights, it is just a way of keep track of the training progress and to make sure that the model is not overfitting based on the training data.

3.5.4 Defining the neural network’s architecture

For this problem, the first layer consists of a “feature input layer”, as the input is a data set of numeric scalars representing features. The last couple of layers are a fully connected layer with 8 neurons (as there are 8 outputs), followed by a regression layer.

A regression layers computes the half-mean-squared error loss for regression tasks. For a single observation, the mean-squared-error is given by:

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2 \quad (3.14)$$

where N is the number of responses, y_i is the target output, and $\hat{y}_i$ is the network’s prediction for response i . When training, the Matlab calculates the mean loss over the observations in the mini-batch.

What comes between the input layer and the output layer are the so-called hidden layers. Defining the neural network’s architecture can be a challenging process, as there is no pre-defined network that works well for all sorts of regression problems, so the number of hidden layers and the number of neurons on each layer should be defined to each specific problem.

To estimate a starting point, keeping all training parameters fixed, a simple algorithm so that for i layers and j neurons, it trains the neural network and calculates the average of the root mean square error from the 8 parameters. Then, it chooses the number of neurons in layer i that provided the lowest average error and repeated the same cycle for all j in layer $i+1$. Thousands of cases were run, with varying i and j values. Figure 3.24 shows part of these results, where the plots contain $i = 1 : 4$ layers and $j = 25 : 25 : 400$

neurons possible on each layer.

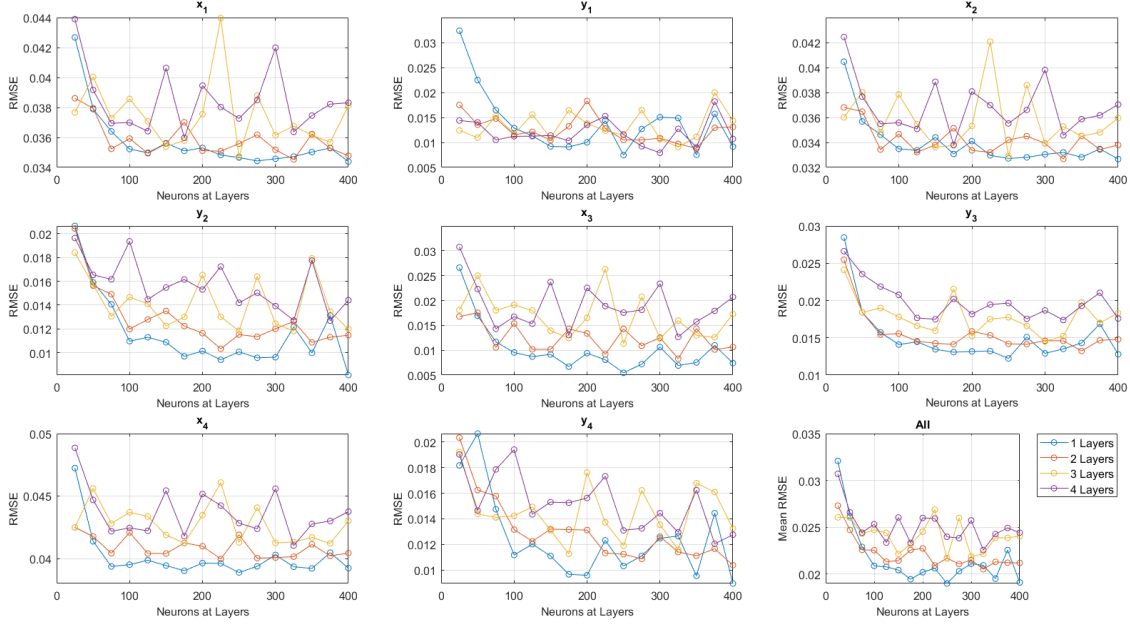


Figure 3.24. Root-mean-squared error for each output of the neural network, with different number of layers and neurons on each layer.

The test went up to 7 layers with 600 possible neurons on each, with a step 2. The results consistently showed that while additional layers may be beneficial to lower the error of some specific outputs, the average RMSE for all 8 outputs was the lowest for 1 single layer, so it was decided to move forward with just one hidden layer between input and output layers.

The final ANN architecture is illustrated in Figure 3.25. The number of neurons on the hidden layer were chosen in another optimization that will be discussed in upcoming sections. Between the hidden and the output layer, there is also a dropout layer and an activation layer.

At training time, the dropout layer randomly sets input elements to zero with a given probability (for this problem, it was defined as 25%). This operation effectively changes the underlying network architecture between iterations and helps prevent the network from overfitting [68].

The activation layer chosen was a rectified linear unit (ReLU) layer, which performs a threshold operation to each element of the input, where any value less than zero is set to zero. This operation is equivalent to

$$f(x) = \begin{cases} x, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases} \quad (3.15)$$

In addition to choosing the number of neurons in the hidden layer, it was also necessary to define the training options. The solver chosen for training the network was the Adam optimizer, as it performed much better than the classic standard gradient descent solver. The other hyperparameters were defined using a Bayesian optimization.

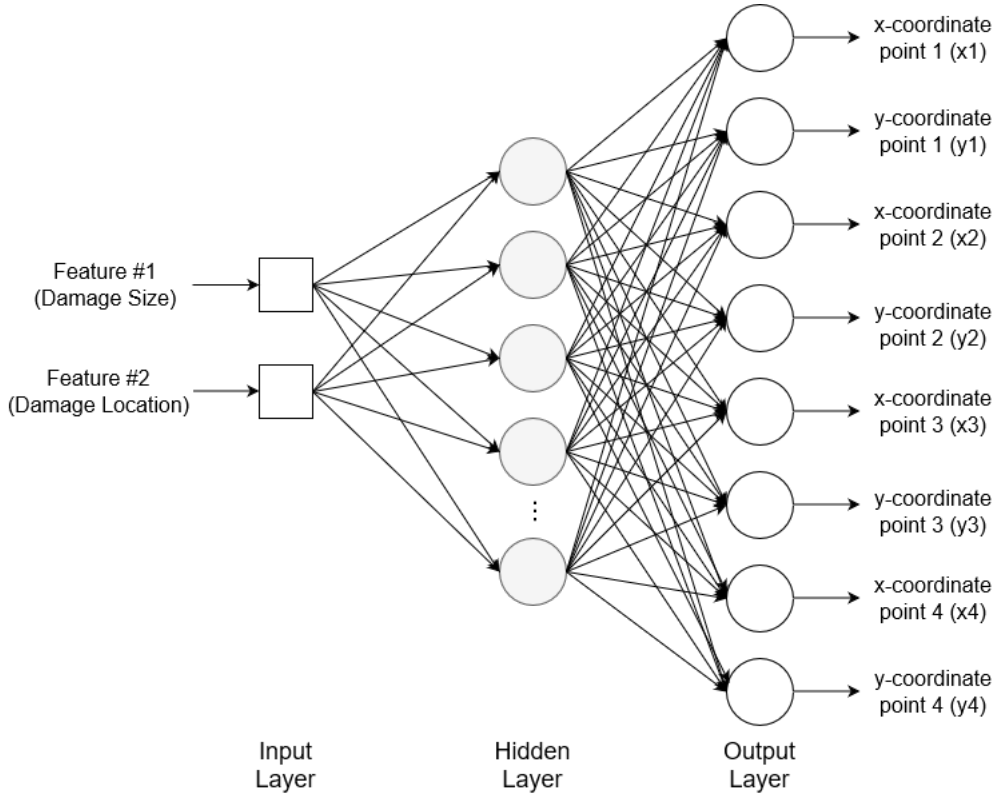


Figure 3.25. Chosen artificial neural network (ANN) architecture, with a single hidden layer.

Bayesian hyperparameters optimization

The aim of hyperparameter optimization in machine learning is to find the hyperparameters (the model settings) of a given machine learning algorithm that return the best performance as measured on the validation data set. The parameters being optimized for this model were:

- Number of neurons in the hidden layer.
- Mini-batch size: size of mini-batch used for each training iteration. A mini-batch is a subset of the training set that is used to evaluate the gradient of the loss function and update the weights.
- Number of epochs: the full pass of the training algorithm over the entire training set using mini-batches is defined as one epoch.
- Initial learn rate used for training. If the learning rate is too low, training takes a long time. If it is too high, then training might reach a suboptimal result.
- Factor for L2 regularization. Adding a regularization term for the weights to the loss function helps reduce overfitting.

An objective function $f(x)$ was written to take in these six hyperparameters, train the neural network, and return the objective score to minimize, which here is the average RMSE from the 8 outputs from the validation data set. This can be represented in equation form as:

$$x^* = \arg \min_{x \in X} f(x) \quad (3.16)$$

where x^* is the set of hyperparameters that yields the lowest value of the score.

The problem with hyperparameter optimization is that evaluating the objective function to find the score is extremely expensive, as each time a different set of hyperparameters is chosen, the model needs to be trained on the training data, make predictions

on the validation data, and calculate the error. The larger the number of hyperparameters being optimized, the longer the process will take.

Grid search and random search are slightly better than manually tuning the parameters, but they are still relatively inefficient because they do not choose the next hyperparameters to evaluate based on previous results. In other words, grid and random search are completely uninformed by past evaluations, and as a result, they often spend a significant amount of time evaluating “bad” hyperparameters.

Bayesian optimization, in contrast to random or grid search, keeps track of past evaluation results which they use to form a probabilistic model mapping hyperparameters to a probability of a score on the objective function: $P(score|hyperparameters)$. This model is called a “surrogate” for the objective function [69], and it is much easier to optimize than the objective function. Bayesian methods work by finding the next set of hyperparameters to evaluate on the actual objective function by selecting hyperparameters that perform better on the surrogate function. In summary, Bayesian optimization chooses the next hyperparameters in an informed manner, so it can find better model settings than random search in fewer iterations [69]. The optimization stops when the defined maximum time or maximum number of iterations is reached.

This optimization was executed for 1,000 iterations and it was defined that the best settings would be 517 neurons in the hidden layer, a mini-batch size of 128 samples, 1000 epochs, a factor of 6.699×10^{-6} for L2 regularization, and an initial learn rate of 1×10^{-3} . These settings applied to the architecture previously defined yielded an average RMSE of 0.0067, which is an improvement of nearly 70% when compared to the results from non-optimized settings.

3.5.5 Results from the ANN model

After defining the architecture of the model (Figure 3.25) and the optimized hyperparameter values, the artificial neural network (also called a feed-forward multi-layer

perceptron model) model was ready to predict the values for the testing input data.

Recall that from only two inputs (damage size and damage location) the model can predict 8 outputs at once ($x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4$, which are the x and y coordinates of points p1, p2, p3, and p4, respectively). The results for the predictions on the testing data set (input values never seen by the model before) are shown in Figure 3.27 and Figure 3.28. These results confirm that the model is performing well. The steps necessary to develop the ANN model are summarized in 3.26.

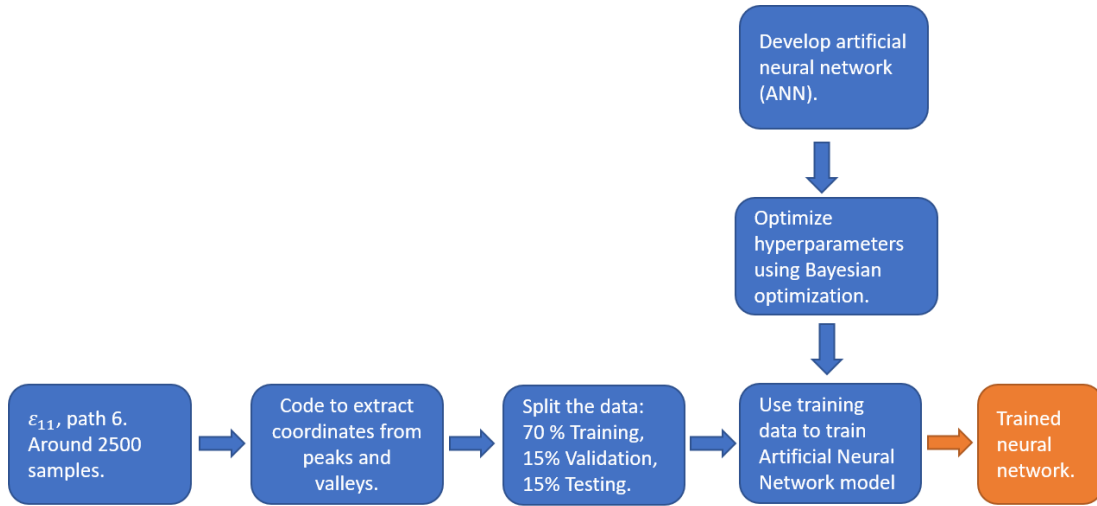


Figure 3.26. Flow chart of the procedure to develop the artificial neural network model.

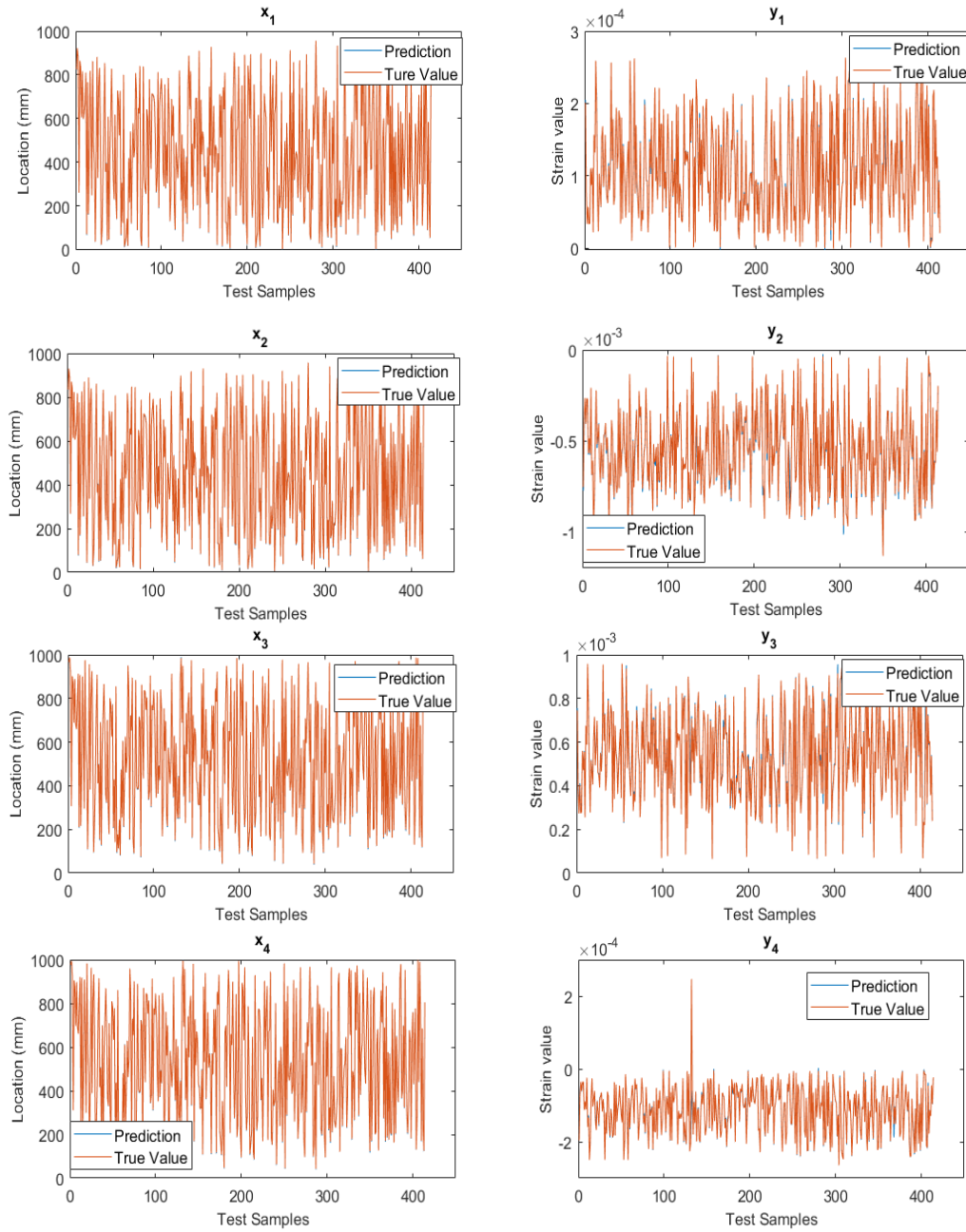


Figure 3.27. Results from artificial neural network predicting coordinates of points p1, p2, p3, and p4. The plots show the results for all samples, overlaying a comparison of the predicted and true value (from Abaqus) for all samples.

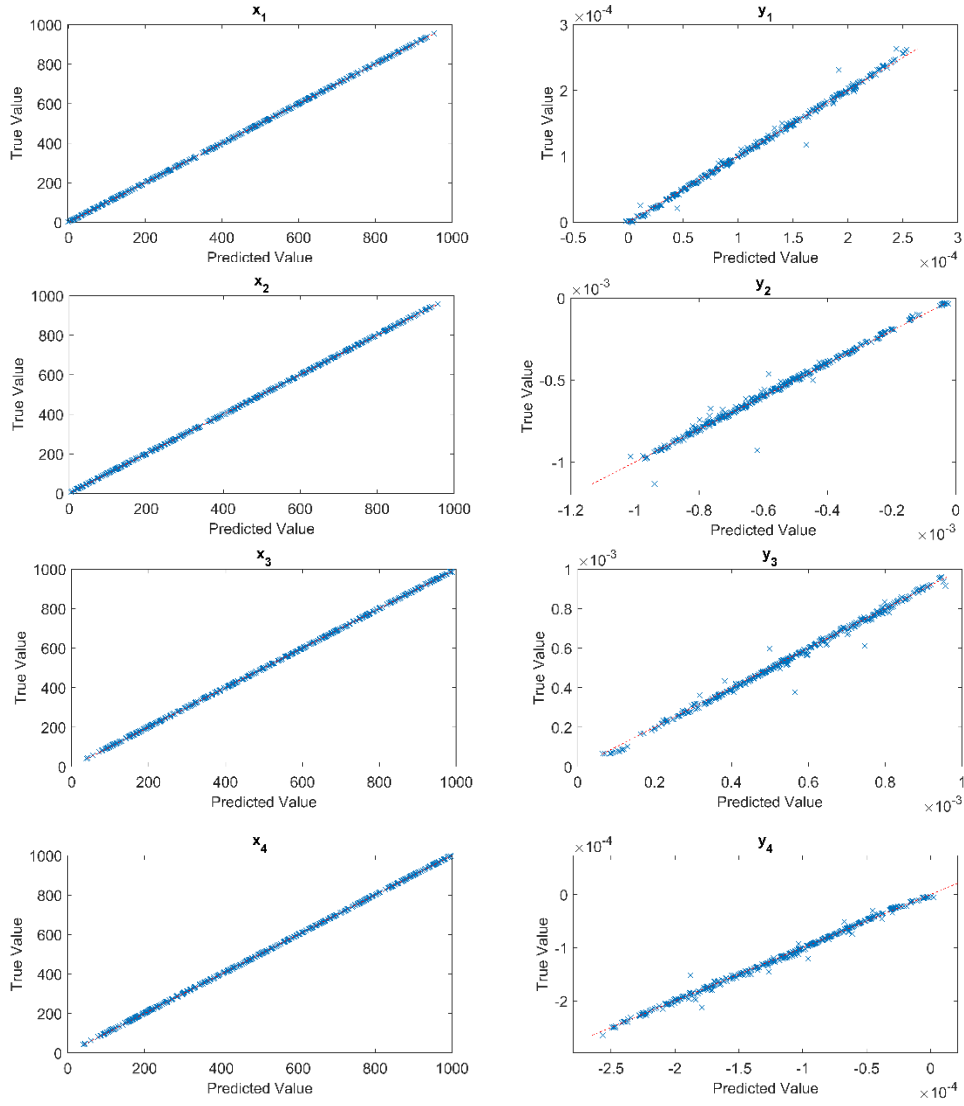


Figure 3.28. Results from artificial neural network predicting coordinates of points p1, p2, p3, and p4. The plots show the results of the predicted values against the true value, and a line in red.

3.5.6 Strain reconstruction using coordinates obtained with the forward model

After developing the ANN model, the methodology described in Section 3.5.2 reconstruct strain. Figure 3.29 shows a flowchart of how the model works. Given the damage size and damage location, the neural network predicts the 8 coordinates corresponding to points p1 to p4. Using these coordinates, another algorithm developed is used to reconstruct the strain ε_{11} from path 6 (the longitudinal direction under the top shear clips). Samples of results obtained from the forward model are shown in Figure 3.30.

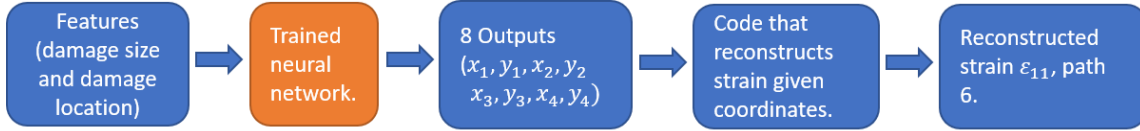


Figure 3.29. Flow chart of the procedure to reconstruct strain ε_{11} from path 6.

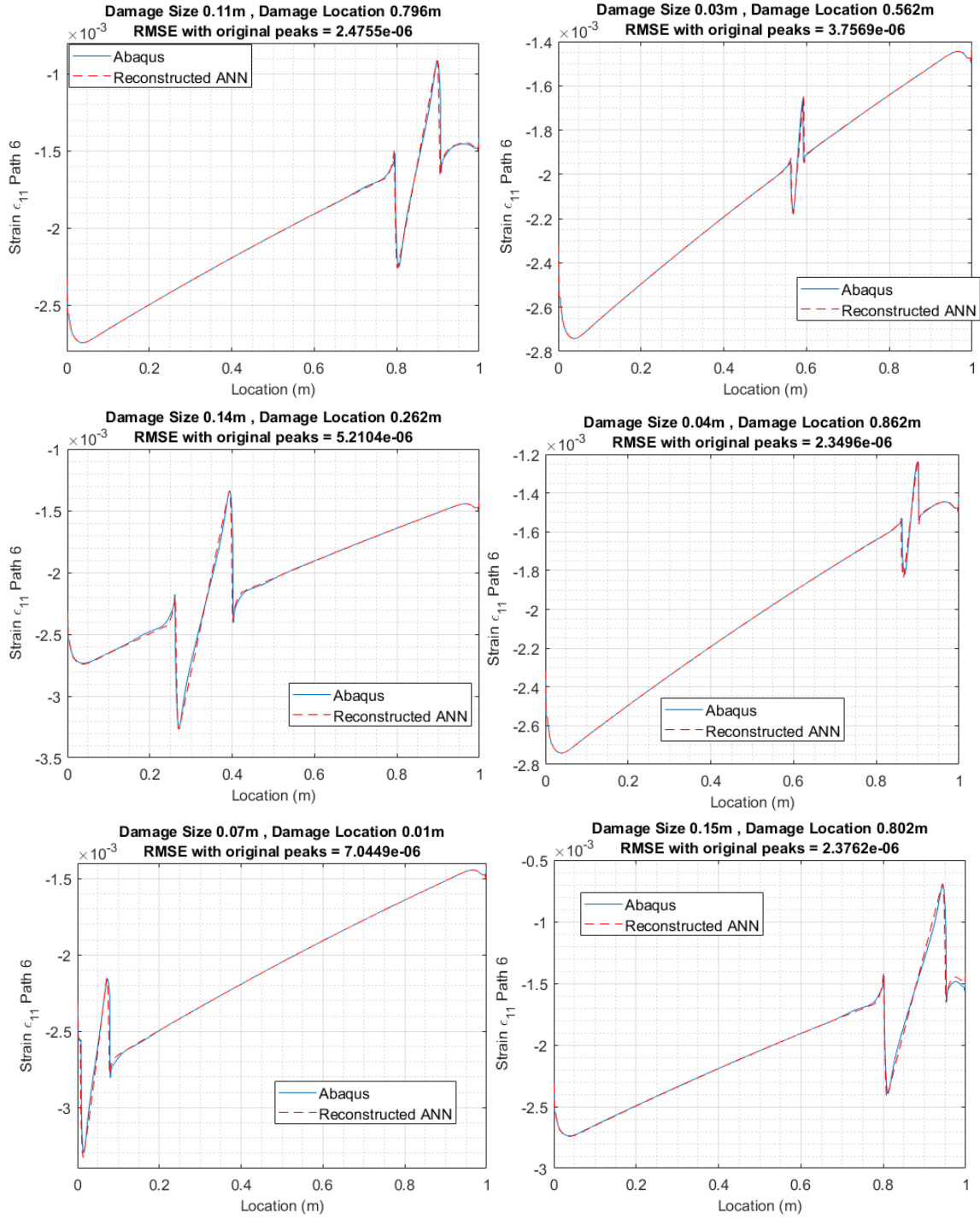


Figure 3.30. Strain reconstruction of different damage sizes and locations, generated with the forward model.

3.6 Expansion to the 5m model

The model of the 1m section of the beam is executed much faster than the 5m model, thus the large computational campaign for the 1m model was a fundamental step to obtain an initial understanding of the spar’s behavior under damage and develop initial tests for surrogate models. However, there are advantages on running the model of the entire beam.

The main advantage is that it is possible to create damage on top of the loading region at 1m and 2m, which is not possible when we only analyze the [0,1]m length. Another advantage of modeling the entire 5m beam is that there is no influence on the buckling eigenvalues due to an “increased stiffness” behavior of the shorter beam when compared to its full-length version.

Figure 3.31 shows the results of the strain measurements along path 6 (on the bottom of the wet layup plies) for the undamaged beam. It is clear how the concentrated force and pure moments applied to the 1m and 2m cross-sections is an area of high stress concentration and affects the strain measurements, as expected. Figure 3.31 (right) shows a zoomed image around the 1m loading region, showing this strain behavior that was not observable on the 1m-long beam.

3.6.1 Adjustments to Python Scripts

The Python scripts discussed in Section 3.2 were re-used for the 5m model with minor changes. The first change is that all the dimensions were extended to 5 meters. Then, differently from the 1m beam, there was no surface at the 1m and 2m cross-sections that were readily selectable to apply the concentrated forces and moments on, so they had to be applied to nodal points instead. Therefore, in addition to all the steps executed by the scripts, every time that the parts were re-meshed, the scripts needed to:

- Select the nodal points at the 1m and 2m cross-sections (Figure 3.32, (a)).

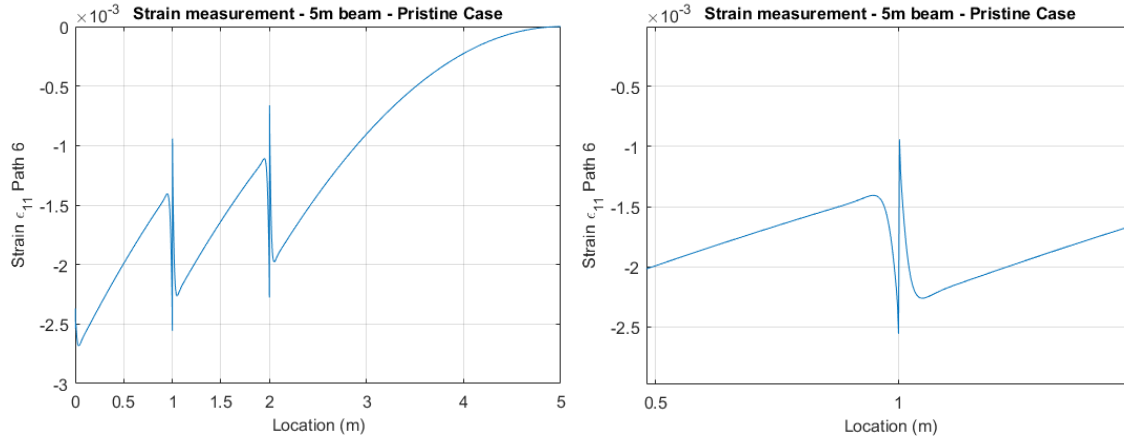


Figure 3.31. Results of strain measurements along path 6 for the 5m beam without damage (pristine). Left: full beam, right: zoomed in around 1m load region.

- Create coupling constraints (Figure 3.32, (b)).
- Apply forces at reference points.

3.6.2 Long run of 5m model

After making the necessary adjustments to the model and scripts, a long computational campaign was executed for the 5m model, containing 800 samples. To create the damage size and location distribution, an importance sampling strategy was taken. From lessons learned from the 1m model and individual runs of the 5m model, it was observed that damage cases that occurred around the root and around the 1m loading region tended to be more critical, usually providing a buckling eigenvalue $\lambda < 1$. It was also observed that damage sizes smaller than $0.03m$ were not critical, so they were sampled less heavily. The damage parameters distributions and the sample space, sampled using Latin hypercube sampling, are shown in Figure 3.33.

In addition to these 800 cases, 40 cases were randomly selected from the 1m long runs, so that the results from the 5m and 1 models could be compared, and 165 cases were sampled around the 1m loading region. Therefore, there were 1005 total cases simulated using the 5m model on Abaqus.

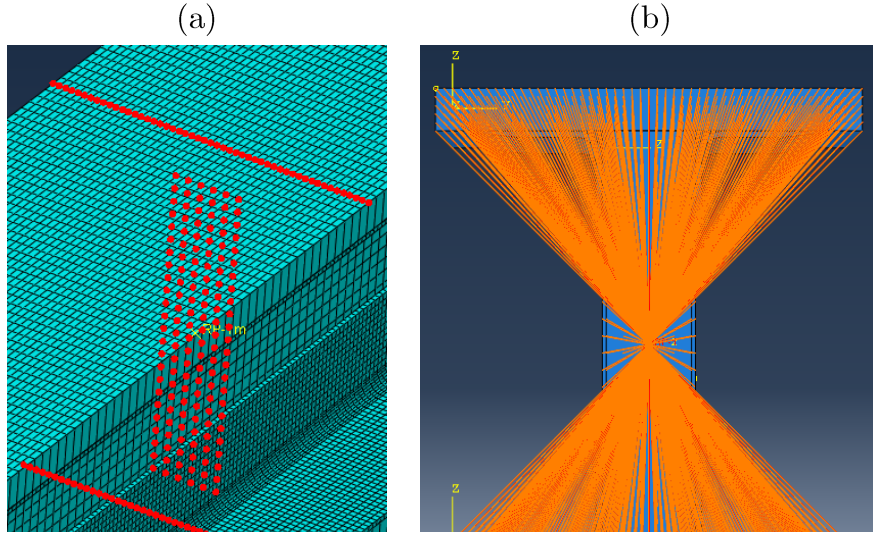


Figure 3.32. (a) Nodal points of loading region highlighted in red, after being automatically selected by the Python scripts. (b) Creation of coupling constraints.

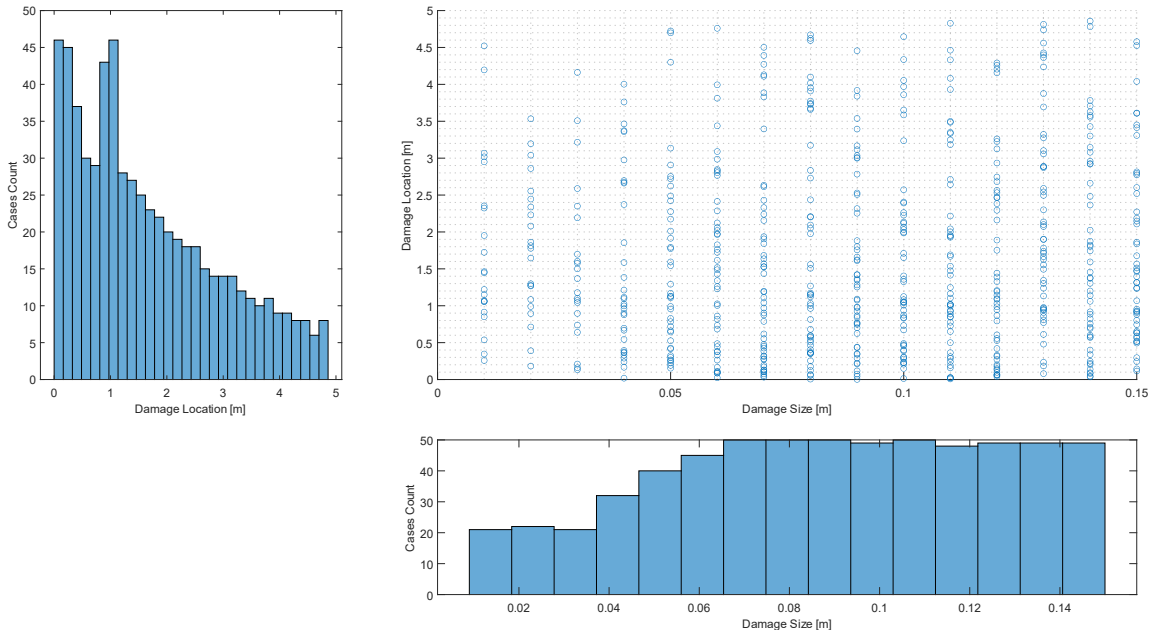


Figure 3.33. Damage size and damage location histograms, and sampling space, for 800 samples of the 5m beam.

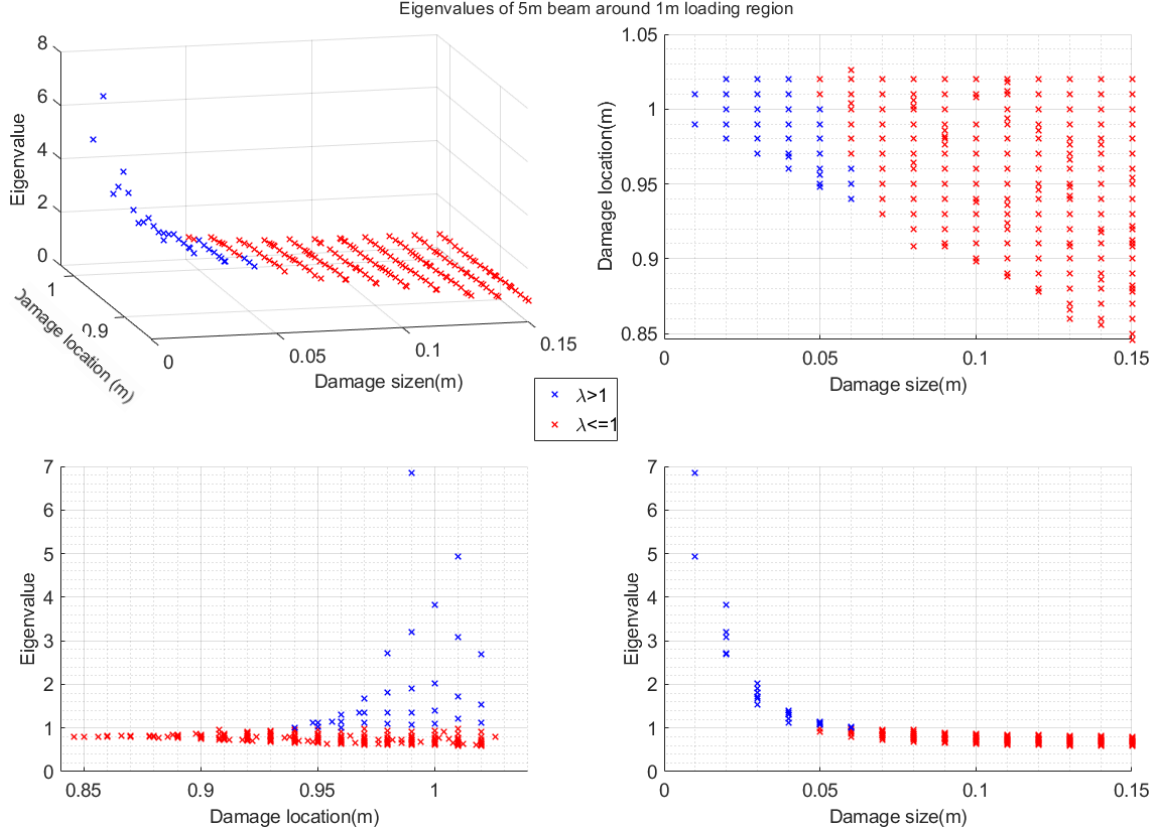


Figure 3.34. Eigenvalues obtained from 5m beam simulation, with damage around loading region (1m cross-section).

3.6.3 Eigenvalue results

As mentioned, the main advantage of having a model of the full beam that it is possible to create damage on top of the loading region at 1m and 2m. Upon analyzing the eigenvalue results of damage located around the 1m cross-section (Figure 3.34) it is clear that most damage in this area are critical, with $\lambda \leq 1$ shown in red. These results emphasize the importance of analyzing the entire beam, instead of only a 1m-long section of it.

3.6.4 Strain results

The strain measurements have the same behavior as they did in the 1m beam in most regions, as they should. However, when damage cases are implemented around the

1m or 2m loading regions, the strain measurements cannot be reconstructed using the artificial neural network model used for the 1m beam (Section 3.5.3). Some of these results are shown in Figure 3.35. Notice that DS and DL stand for damage size and location to make it straightforward, but they are more formally defined as σ and x , respectively, in other sections.

While only 6 examples are shown in Figure 3.35, by looking at all 205 cases around the 1m loading region, it was possible to identify some trends in behavior:

- Case 1: Damage cases that ended right before 1m ($DS + DL < 1m$) had a behavior similar to the one shown in the first row, first column.
- Case 2: Damage cases that ended exactly at 1m ($DS + DL = 1m$) had a behavior similar to the one shown in the first row, second column.
- Case 3: Damage cases that ended passed through the loading region ($1m < DS + DL < 1.15m$) had a behavior similar to the ones shown in the second row.
- Case 4: Damage cases that ended passed through the loading region and started around 1m ($1m < DS + DL < 1.15m \& 0.98m < DL < 0.998m$) had a behavior similar to the one shown in the third row, column 1.
- Case 5: Finally, damage cases that started right after 1m ($1m < DL < 1.02m$) had a behavior similar to the one shown in the third row, column 2.

Therefore, in addition to the “usual” strain signature (e.g., 3.13), there are at least 5 different behaviors that can be expected when looking at strain measurements from the 5m beam, and it is hard to find a universal model that captures all these different behaviors.

Principal component analysis (PCA) is common technique used for dimensionality reduction when strain is available from a large number of sensors (in this case, nodal points

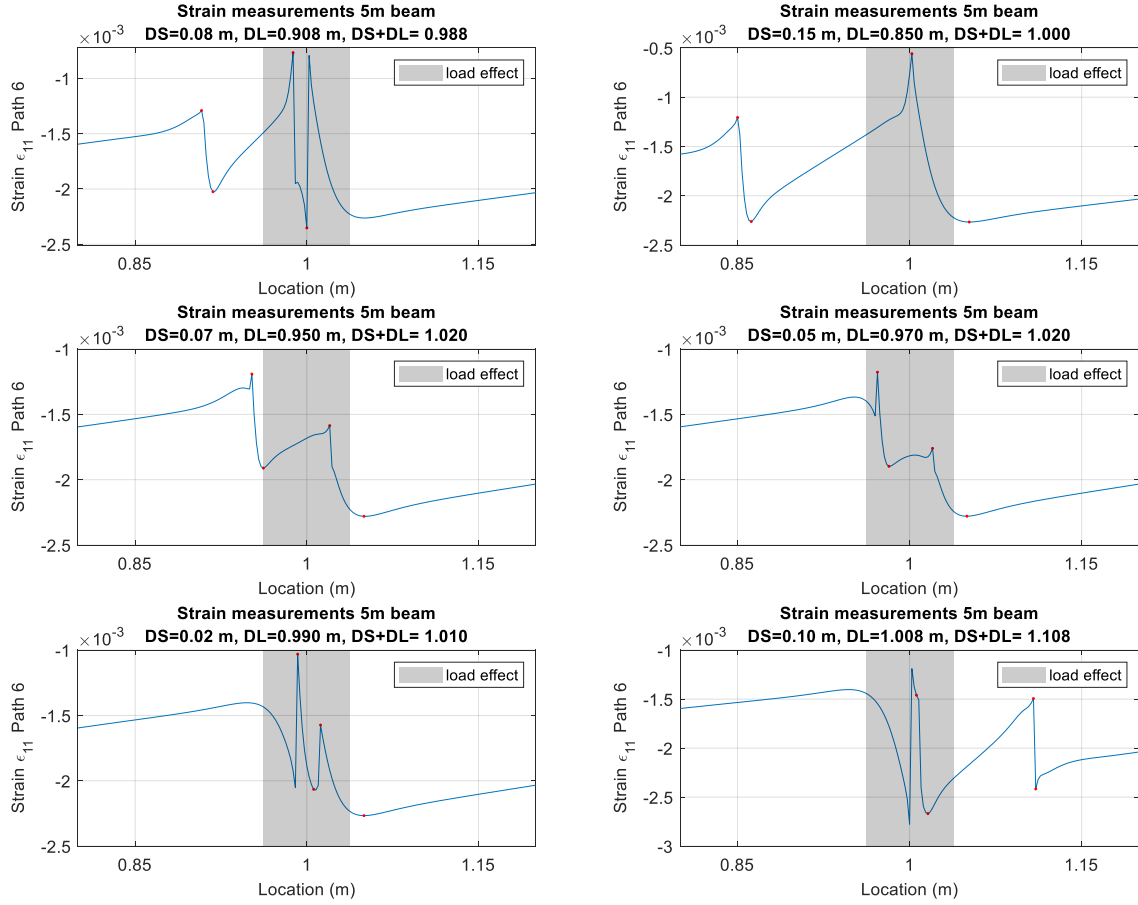


Figure 3.35. Strain ϵ_{11} obtained from path 6 on the 5m-long beam. Results shown focus on damage cases that are around the loading region (shown in gray). The small red dots will be used for strain reconstruction later in this section.

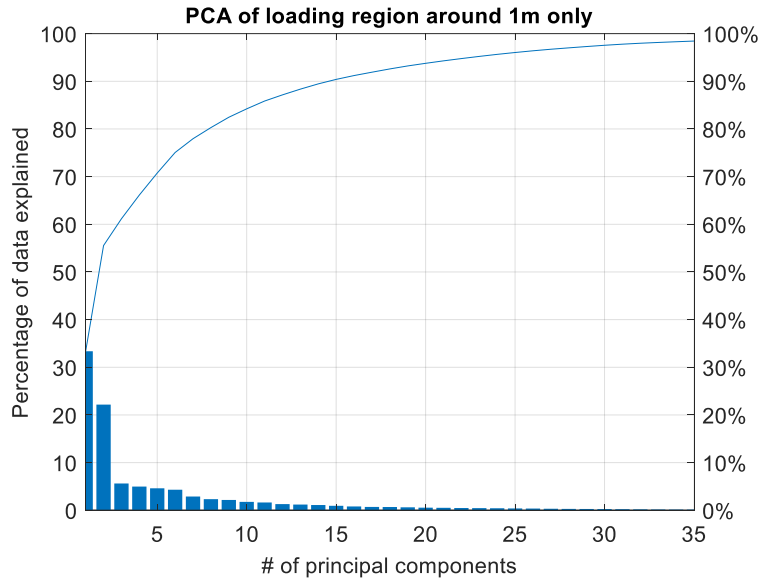


Figure 3.36. Principal component analysis result of 5m beam, focusing on 1m loading region only.

from the FE model). By grabbing the strain measurements only around the 1m loading region (0.85m to 1.15m) and performing the PCA, it is shown on the pareto chart (Figure 3.36) that 35 principal components are necessary to represent 98.44% of the measurements.

The first concern with using this method for strain reconstruction is that 35 machine learning models would need to be trained to predict each principal component for new damage cases, that were not simulated on Abaqus. The second concern is that while 98.44% signal representation is very good, the reconstruction is not very good, as shown in Figure 3.37. In some cases, it tends to recreate characteristics that are not present in all cases, as shown in Figure 3.37 subplots (1,1) and (3,2), where it adds a peak to the signals. In other cases, it tends to reduce the magnitude of the peaks and valleys, as shown in the middle row of Figure 3.37.

As it will be explained in upcoming chapters, the strain reconstruction is used to calculate the likelihood of a measurement given a certain damage case, so that the code can calculate how useful it is to have a sensor at a certain location (how much information is gained by placing a sensor at that specific location being analyzed). Therefore, having

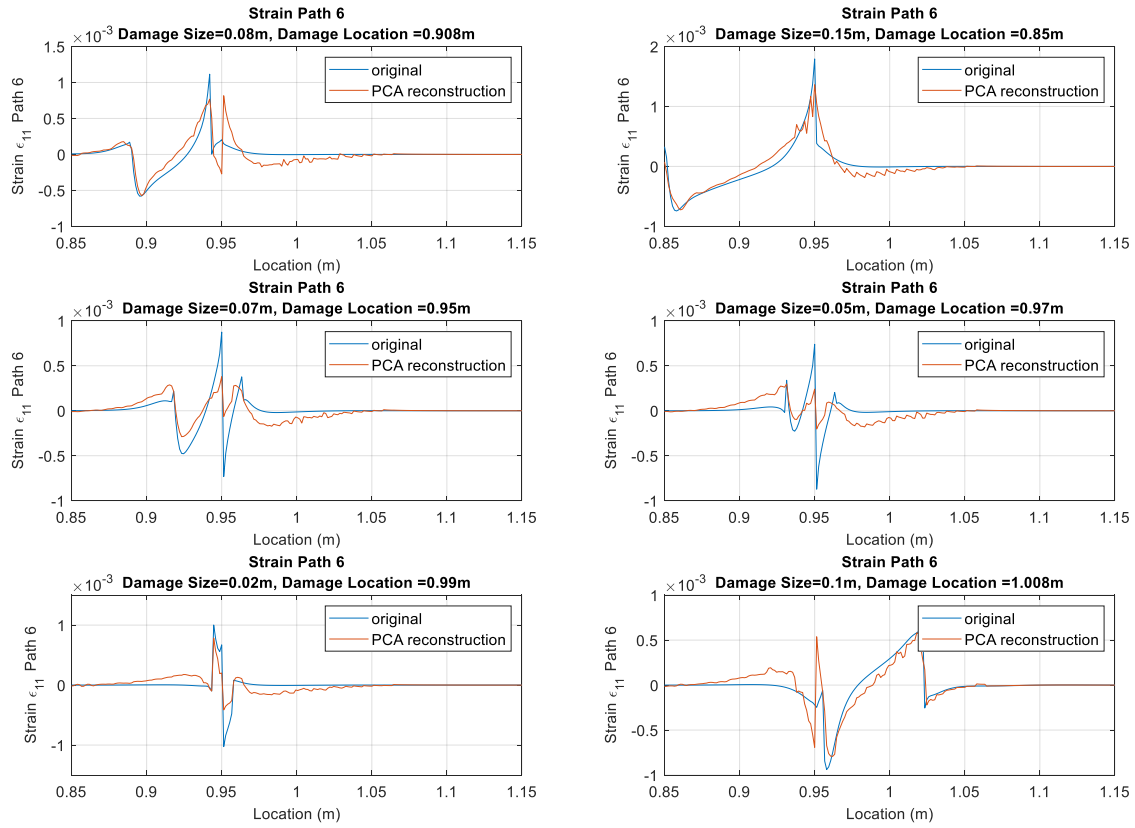


Figure 3.37. Strain reconstruction of several damage cases, using the first 35 principal components.

a good strain reconstruction with small error values is crucial for the sensor placement optimization step.

3.6.5 Updated forward models for the 5m beam

Section 3.5 detailed the strategy to reconstruct the strain measurements for the 1m-long beam. A critical part of it was breaking down the signal into 5 curves, as shown in Figure 3.19, and finding the (x, y) coordinates of the two peaks and two valleys that were used on the governing equations. These coordinates were used as proof-of concept to show that it was possible to reconstruct the strain based on these 4 coordinates. Then, a neural network model was developed to predict the coordinates of new cases. A similar strategy will be used for the 5m beam. However, as shown in Figure 3.35, the strain measurements behave differently on damage cases around the loading region, so a new model was built for the loading region only, and it will be detailed below.

The “find peaks” code developed for the 1m model (Section 3.5.1) was adapted to find 2 peaks and 2 valleys that consistently appeared in the new measurements, shown as small red dots in Figure 3.35. Some of the signals also had 1 additional peak and valley, so the code must distinguish between the 4 points that correspond to the effect of the damage on the measurement (these will be the points used), and the other 2 points that corresponded to the effect of the loads. The new code was tested, and it worked well to capture the (x, y) coordinates of such 4 points. Each coordinate was saved, and the results are shown in Figure 3.38.

Figure 3.38 shows that each coordinate has a similar behavior within their different damage sizes, and most of them had non-linearities. Attempts were made to fit curves that captured the results of each coordinate, but most of the results had a high RMSE - specially because the smaller damage were not described by the same curves as the larger damage.

Therefore, the reconstruction of strain for the 5m beam will be separated into two

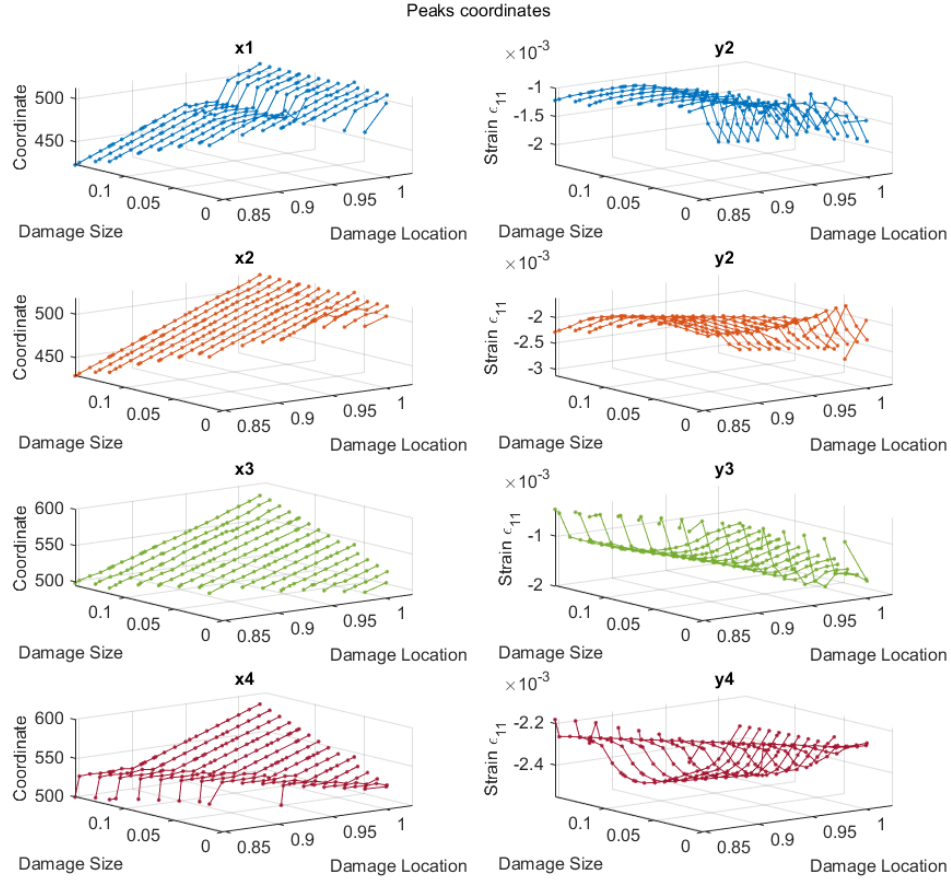


Figure 3.38. Results of the (x,y) coordinates of the 4 points that correspond to the damage effect on the strain measurements. Results are shown for each coordinate, plotted against damage size and damage location.

problems. Strain measurements for damage cases that occur more than 15cm away from the loading region will continue to be reconstructed using the ANN surrogate model from Section 3.5. Damage cases around the loading regions at 1m and 2m will need a new surrogate model to capture their strain behavior, which will be described in the upcoming section.

Gaussian process regressors to predict (x,y) coordinates

The first surrogate model attempted was a deep artificial neural network, similar to the one described in Section 3.5 – a model that with 2 inputs (damage size and damage location) was able to predict all the 8 coordinates at once. However, this model performed poorly for the loading region, especially when predicting the y coordinates, and this is attributed to the lack of training data. Recall that the 1m model had 2,000 samples for training. For this new model, by combining all the cases simulated on Abaqus with damage cases around the 1m loading region, there were results available from only 205 different samples.

Therefore, it was decided to break the problem into 8 different models, and to use Gaussian Process regressors to solve it. 85% of the cases were used for training (174 cases) and the other 15% were used for testing. The inputs to each model were damage size and damage location, and the output was either the x or y coordinates of points 1, 2, 3, and 4. Table 3.3 describes the output of each model, and the kernel function used for each of them.

The exponential kernel was already described in Equation (3.10). In a GP regressor, the automatic relevance determination (ARD) exponential kernel is given by:

$$k_{\text{ARD-Exp}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp \left(-\frac{1}{2} \sum_{i=1}^D \frac{(x_i - x'_i)^2}{\lambda_i^2} \right) \quad (3.17)$$

where:

Table 3.3. Description of the output of each GPR model and the kernel function used.

Model number	Coordinate	Kernel function
1	x_1	Exponential
2	y_1	ARD Exponential
3	x_2	ARD Exponential
4	y_2	ARD Exponential
5	x_3	ARD Exponential
6	y_3	ARD Exponential
7	x_4	ARD Squared Exponential
8	y_4	ARD Squared Exponential

σ^2 is the variance parameter,

D is the dimensionality of the input space,

$\mathbf{x}$ and $\mathbf{x}'$ are input vectors,

λ_i are the length-scale parameters for each input dimension i .

This kernel allows each input dimension to have its own length-scale parameter λ_i , providing flexibility in modeling the importance of each input dimension. The ARD squared exponential kernel in a GP regressor is given by:

$$k_{\text{ARD-SE}}(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp \left(-\frac{1}{2} \sum_{i=1}^D \frac{(x_i - x'_i)^2}{\lambda_i^2} \right) \quad (3.18)$$

This kernel is similar to the ARD exponential kernel, but it uses the squared exponential function for the distance calculation.

After partitioning the data, both the inputs and outputs were normalized, using the Equation (3.13) so that their ranges were $[0, 1]$. Normalization helps stabilize and speed up model training. Also, the damage size varied from 0 to 15mm while the damage location varied from 0 to 1000mm, so this guarantees that one parameter will not have more weight than the other when training the network. The same is valid for the outputs, as the x-coordinates were in the order of 10^3 while the y-coordinates were in the order of

Table 3.4. MSE results using 85% of samples for training the GPR models.

Coordinate	MSE
X1	0.0018
Y1	$6.668e^{-4}$
X2	$2.237e^{-4}$
Y2	$6.0335e^{-4}$
X3	$6.024e^{-5}$
Y3	0.0105
X4	0.0042
Y4	0.0033

10^{-4} .

The results from each of the 8 models are shown in Figures 3.39 and 3.40. The 95% confidence interval for each prediction is shown in gray.

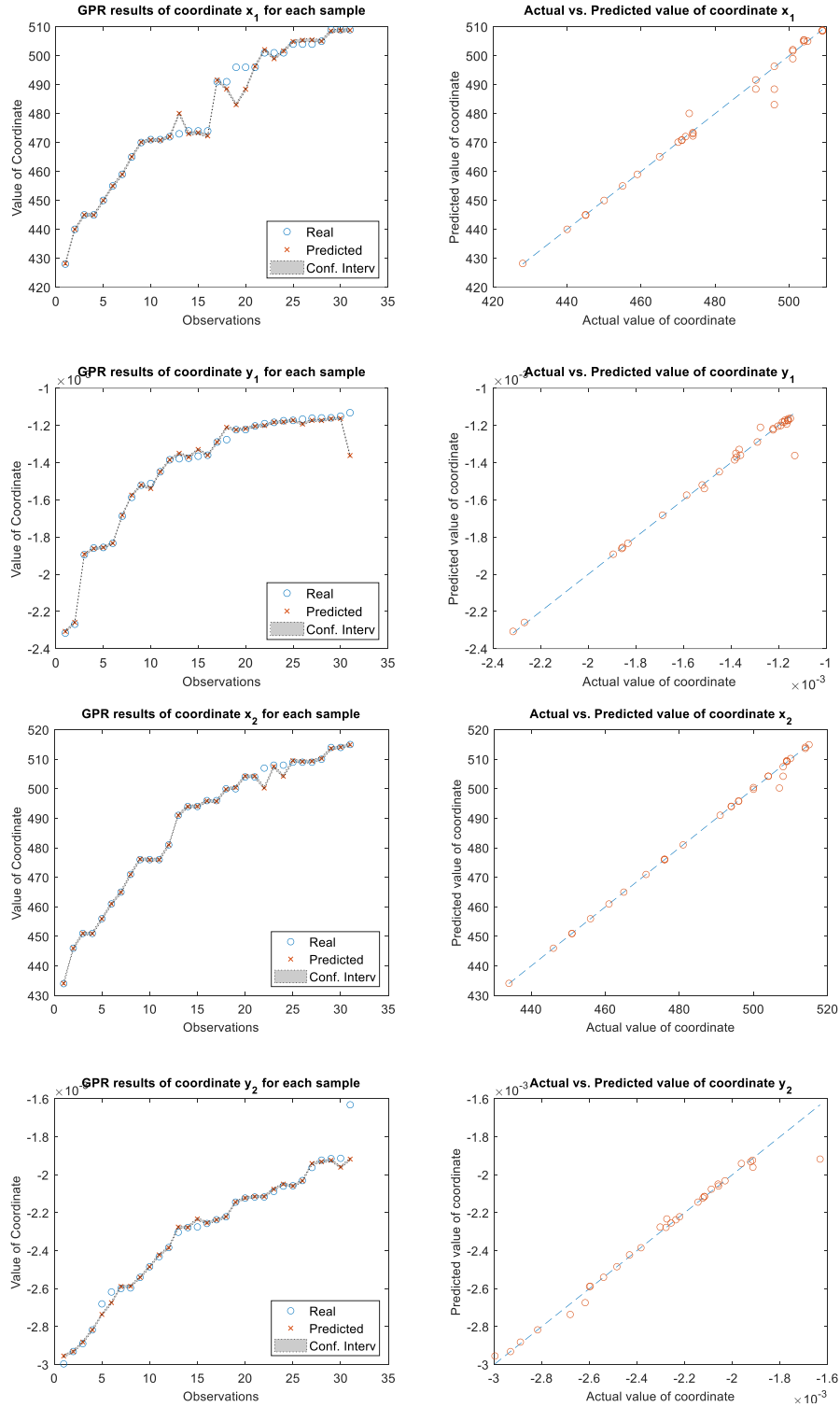


Figure 3.39. GPR results of the predictions of the testing data for (x, y) coordinates of points 1 and 2 (models 1 to 4).

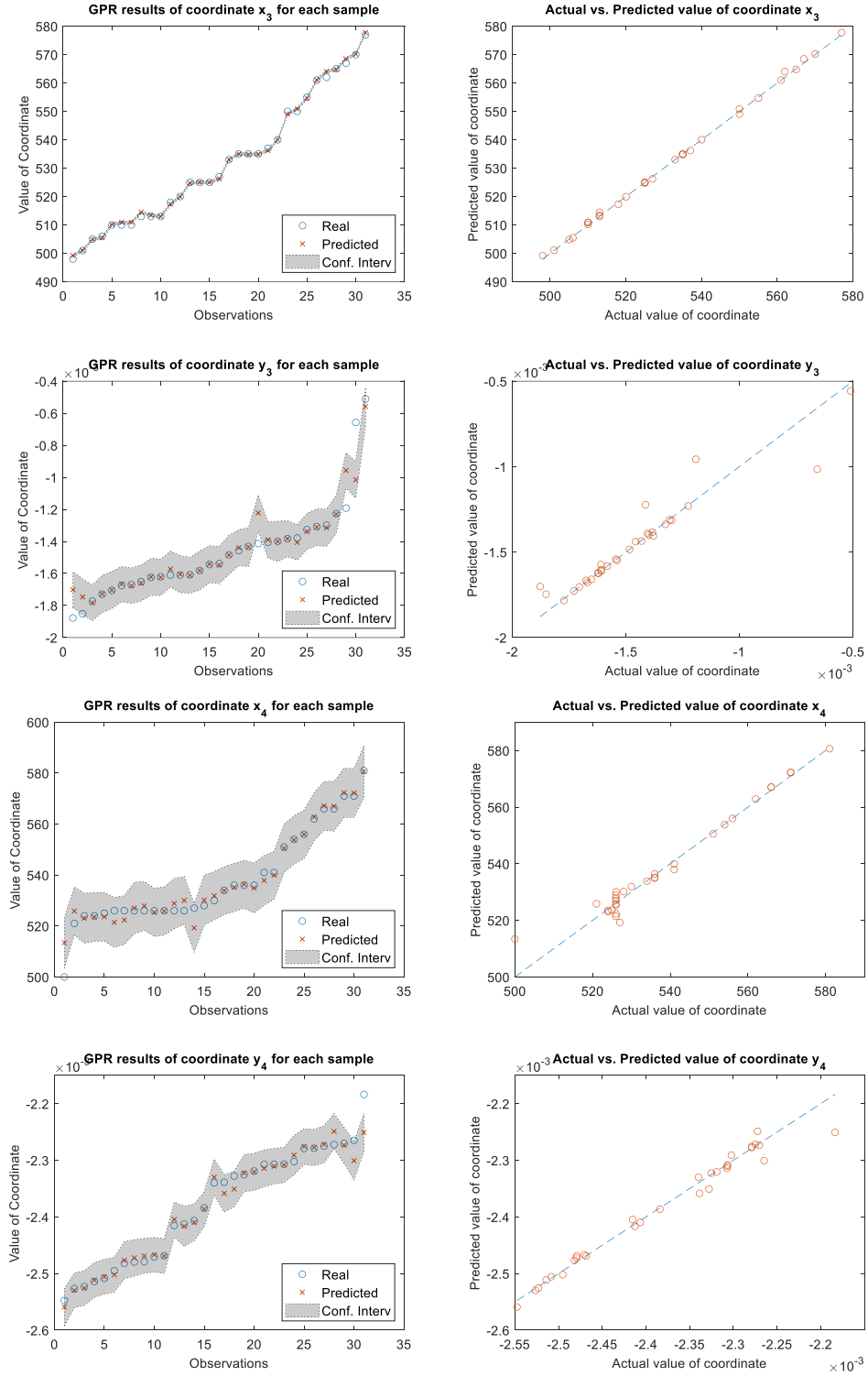


Figure 3.40. GPR results of the predictions of the testing data for (x, y) coordinates of points 3 and 4 (models 5 to 8).

3.6.6 Strain reconstruction of the 5m beam

The final piece of this surrogate model is finding the governing equations to reconstruct the strain behavior of path 6 around the 1m and 2m loading regions, given the eight (x, y) coordinates that the GP regressors predict. The governing equations of each of the 5 curves for each of the 5 cases described in Section 3.6.4 are shown in Table 3.5. There are no equation parameters a,b,c,and d that work for all cases, so they were implemented in the code as functions of the x and y coordinates of points p1, p2, p3, and p4.

Figures 3.41, 3.42, 3.43, 3.44, and 3.45 show the strain reconstruction using the governing equations described in Table 3.5. The average RMSE for all 205 cases was 1.1099e-08. The RMSE of each case was calculated from 0.850m to 1.15m only, to guarantee that it was calculating the error of the segments reconstructed around the damage only.

The strain reconstruction worked well for all the different cases previously described, as shown by the RMSE values. Therefore, by combining the previous forward model (Section 3.5.3) that works well for damage cases far away from the damage region with this new model presented, the strain from path 6 can be reconstructed for the entire 5m beam.

In addition, a new GP regressor was trained for the 5m beam, using 90% of the 1005 total cases simulated using the 5m model on Abaqus as training data, and 10% as testing data. Results from this new model are shown in Figure 3.46.

Table 3.5. Governing equations of each strain segment, as a function of the x coordinate of points p1-p4, for each case described in 3.6.4.

	Curve 1	Curve 2	Curve 3	Curve 4	Curve 5
1	$a \exp(bx) + cx + d$	$ax + b$	$b(\tan \theta(x) - a)$	$ax + b$	Baseline
2	$a \exp(bx) + cx + d$	$ax + b$	$b(\tan \theta(x) - a)$	$ax + b$	Baseline
3	$a \exp(bx) + cx + d$	$ax + b$	$ax + b$	$ax + b$	$a \exp(-bx) + cx + d$
4	$a \exp(bx) + cx + d$	$ax + b$	$ax + b$	$ax + b$	$a \exp(-bx) + cx + d$
5	Baseline	$ax + b$	$ax + b$	$ax + b$	$a \exp(-bx) + cx + d$

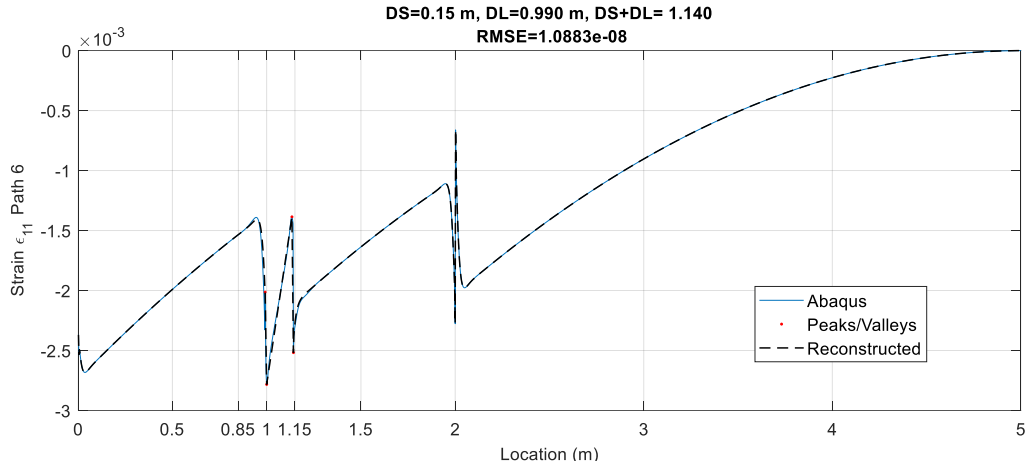


Figure 3.41. Strain reconstruction for path 6 along the entire 5m beam, for damage size= 15cm and damage location = 0.990m.

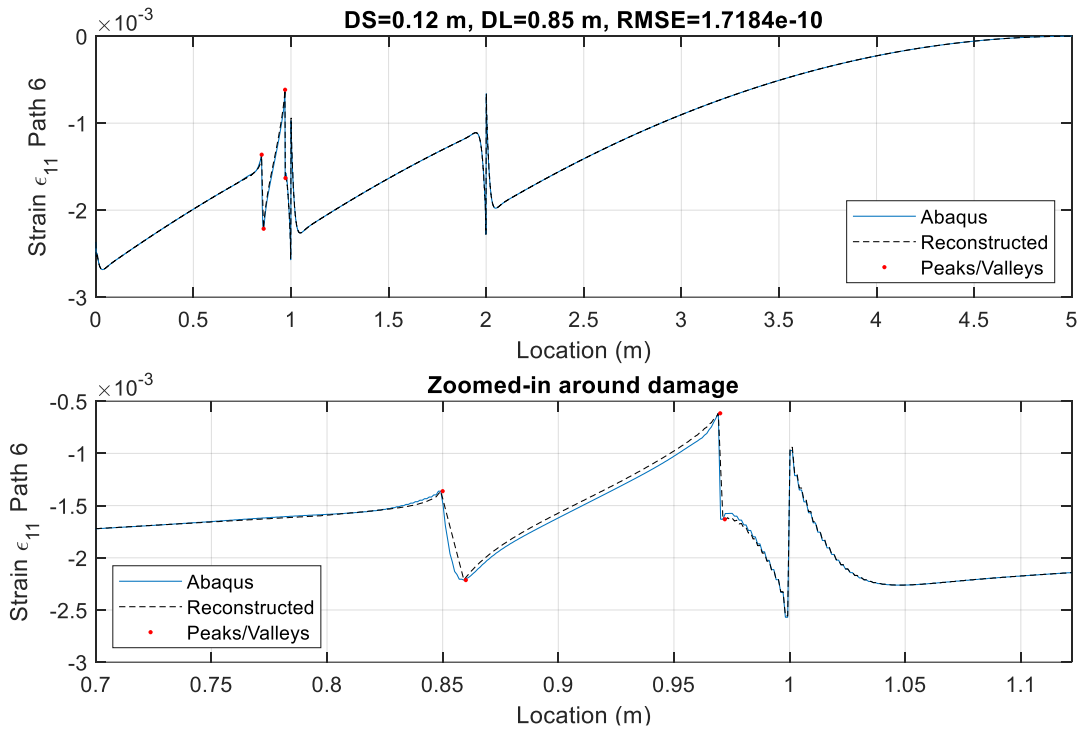


Figure 3.42. Reconstruction of a damage case that ended right before 1m.

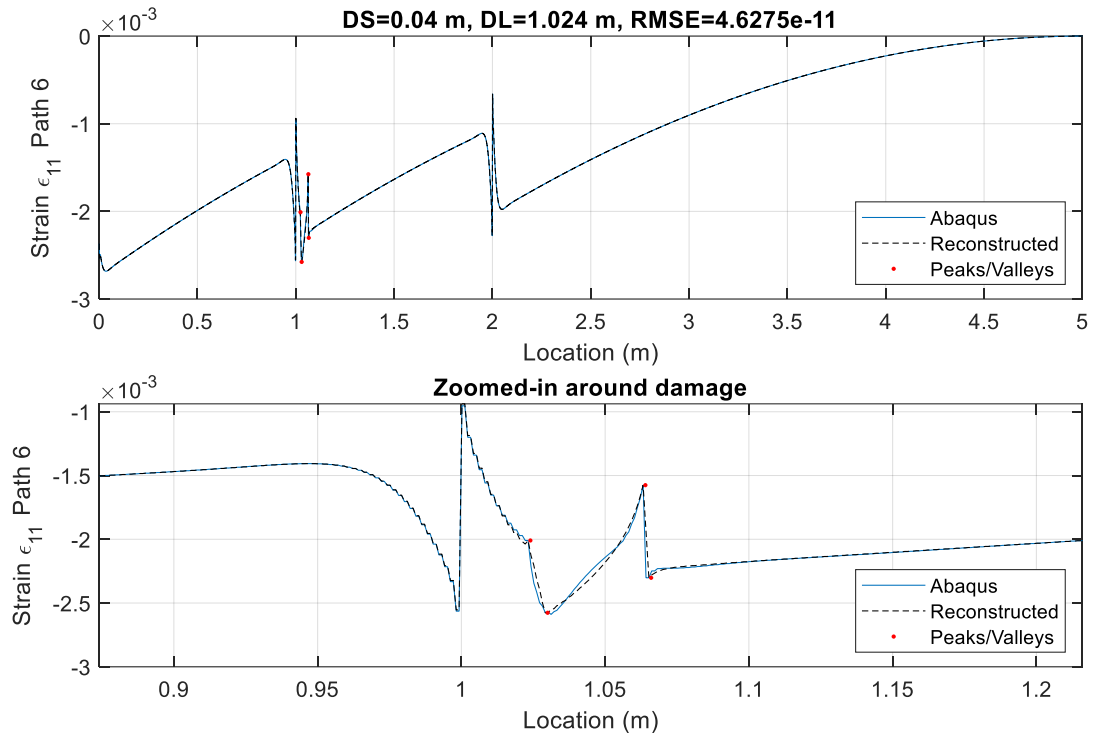


Figure 3.43. Reconstruction of a damage case that occurs through the 1m loading region.

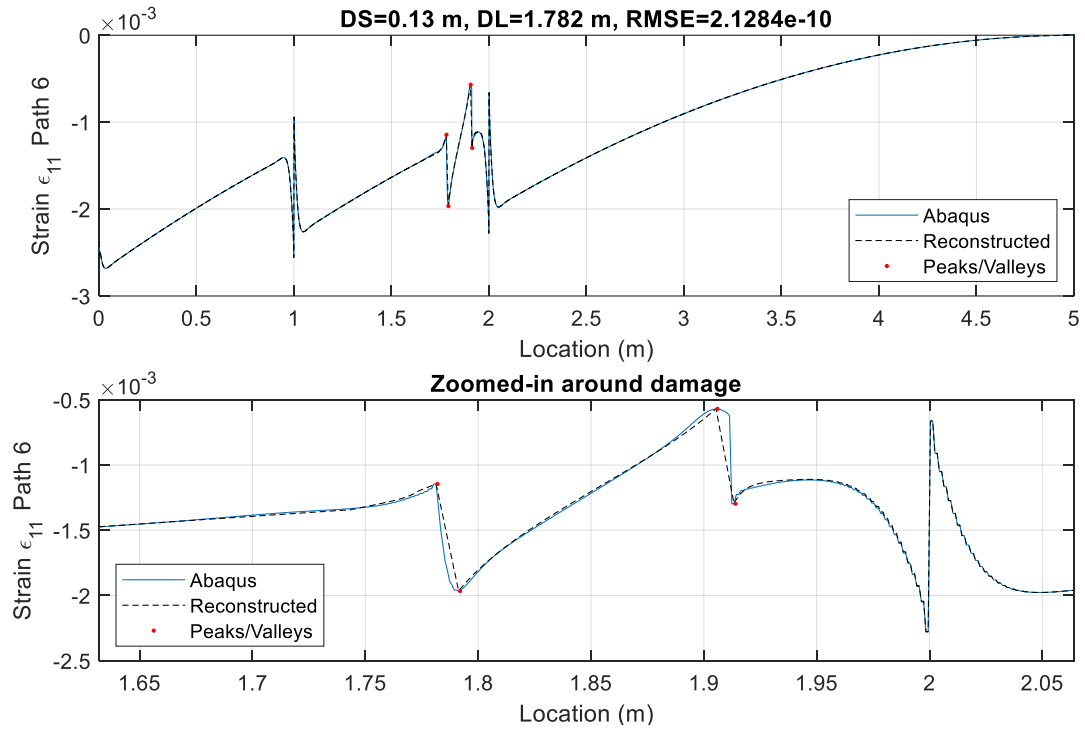


Figure 3.44. Reconstruction of a damage case that occurs right before the second loading region (2 meters).

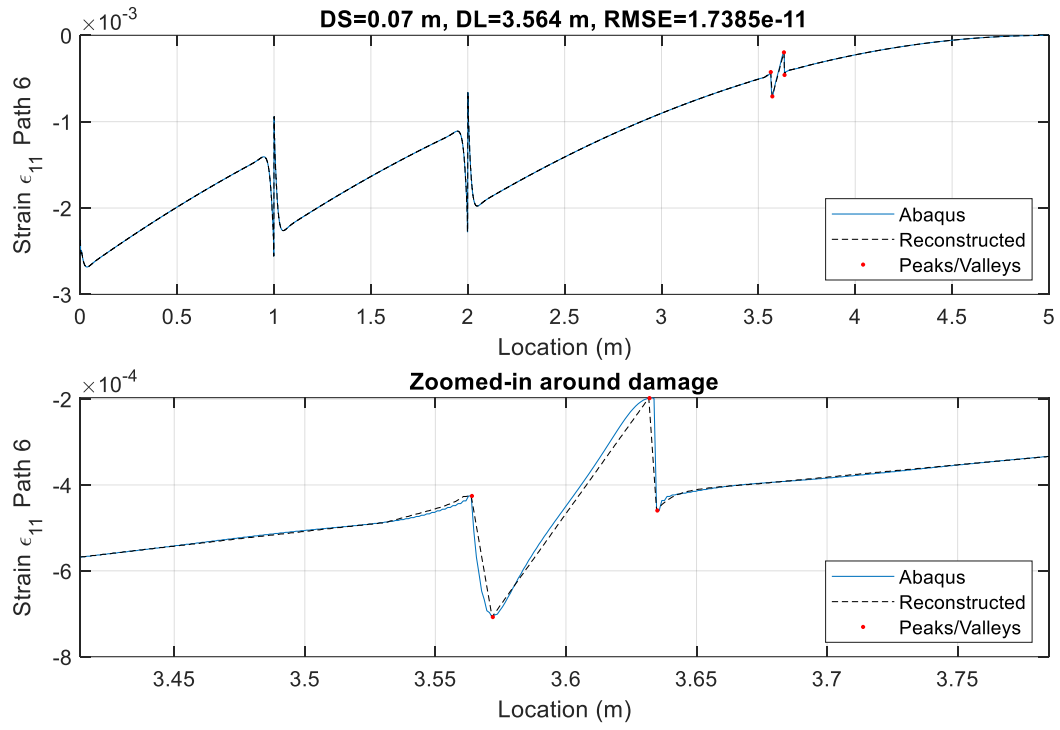


Figure 3.45. Reconstruction of a damage case far from loading region.

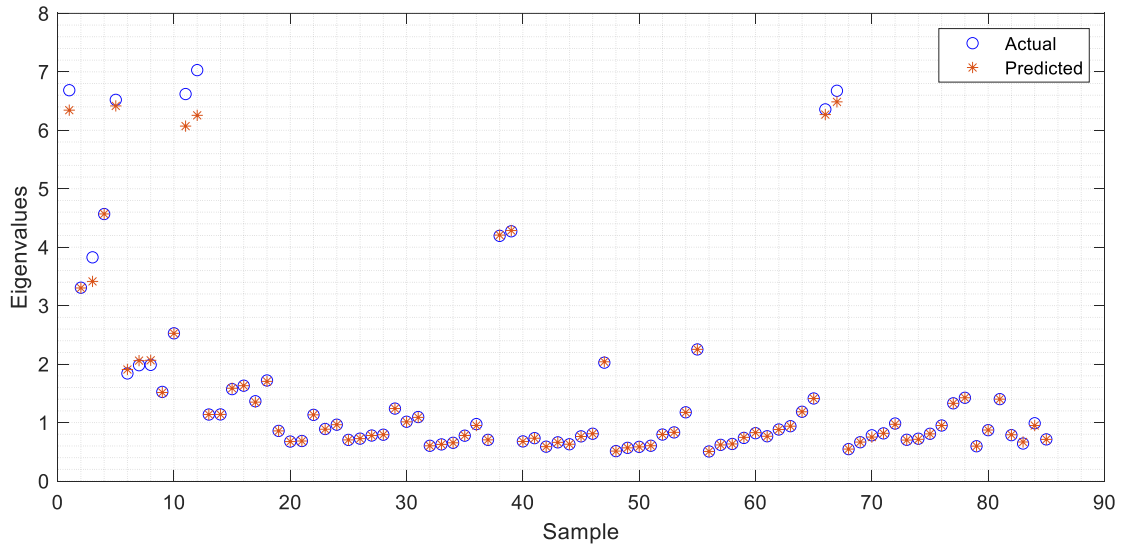


Figure 3.46. Eigenvalues predicted using GP regressor model for 5m beam.

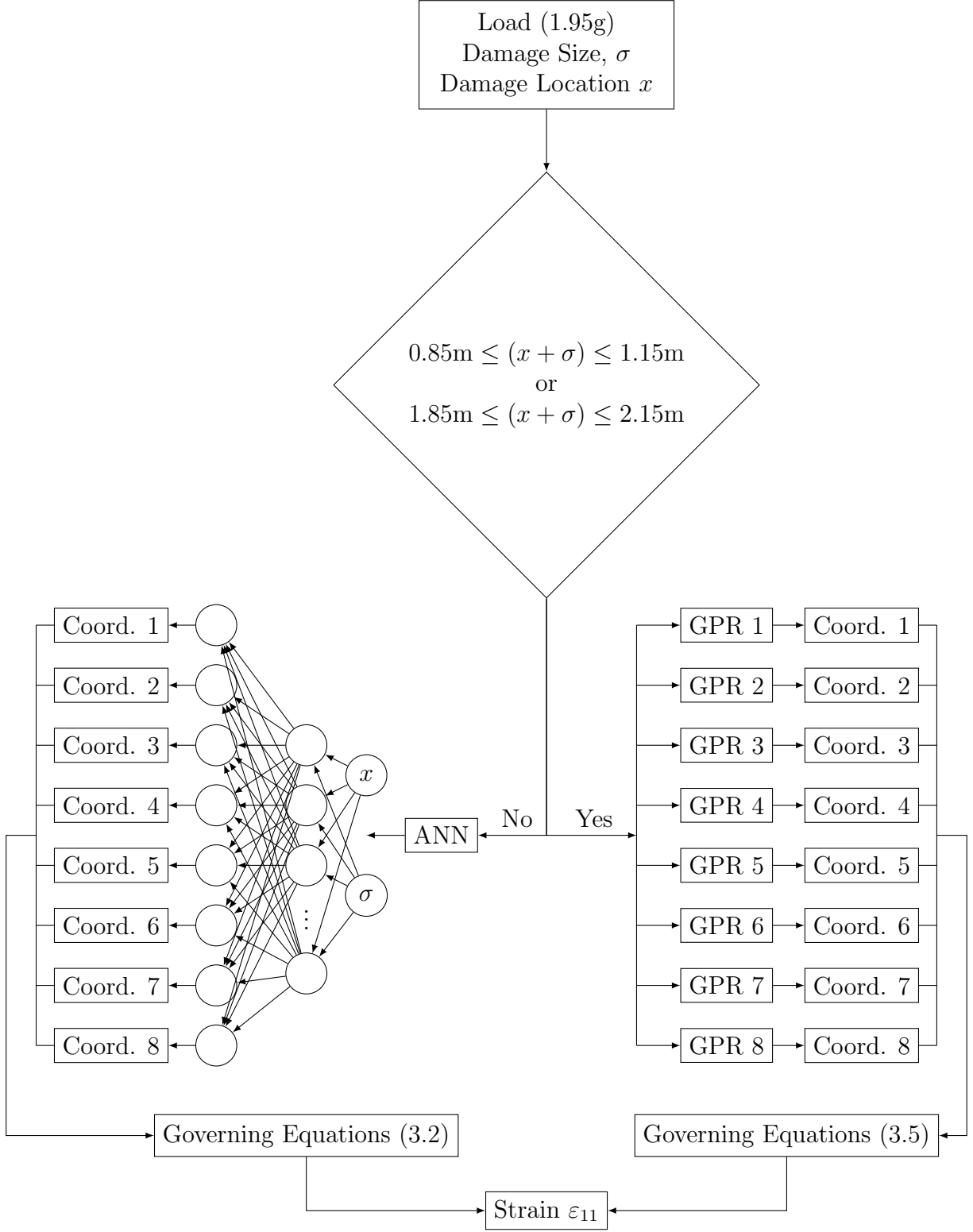


Figure 3.47. Flowchart of the surrogate model for strain reconstruction.

3.7 Improvement in computational performance

The purpose of developing a forward surrogate model was to replicate the finite element model input/output results to achieve sufficient number of Monte Carlo simulations with reasonable computational time. Recall that Abaqus takes around 15 minutes to make a full run of one damage case on the 1m beam, and 50 minutes on the 5m beam, which becomes computationally expensive when it needs to be run thousands of times.

By combining the eigenvalue predictor surrogate model (Section 3.4) with the strain surrogate model described in Section 3.5.3, a forward model for the 1m beam that can “replace” Abaqus and predict the buckling eigenvalues and reconstruct strain ε_{11} from path 6 in less than $10^{-1}seconds$ was developed, which is an incredible performance gain.

3.8 Summary

This chapter introduced a formal statistical modeling approach to the damage inference problem. Then, it was stated that in order to achieve sufficient Monte Carlo simulations to represent the entire damage state sampling space, it was necessary to build “run-time” surrogate models in the forward direction to replace Abaqus. Python scripts were developed to collect enough data from the FE model to train and test the surrogates, as manually executing such a large computational campaign would be impossible.

Then, two surrogate models were developed - an ANN to reconstruct strain measurements, and a GP regressor to provide buckling eigenvalues. The models were first developed to an 1-meter subsection of the beam, and then expanded to the entire 5-meters spar. It was shown that the use of surrogate models reduce the run time from 50 minutes to mere 0.1 seconds, which is an astronomical performance gain. Chapter 4 will describe how this data is used to build a damage diagnosis model.

Chapter 3, in part, has been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., Tur,

M. , & Yehoshua, T.,2023. The title of this paper is “The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application”. The dissertation author was the primary investigator and author of this paper.

Chapter 3, in part, has been published in *Data Science in Engineering, Volume 9. Conference Proceedings of the Society for Experimental Mechanics Series.*, Razzini, A.H., Kressel, I., Ofir, Y., Tur, M., Yehoshua, T., & Todd, M.D., 2023. The title of this paper is “Development of a Surrogate Model for Structural Health Monitoring of a UAV Wing Spar”. The dissertation author was the primary investigator and author of this paper.

Chapter 4

Damage diagnosis

Once enough data is obtained or generated, the next step in the optimal sensor placement framework is to develop a diagnosis model that is able to distinguish changes in sensor readings caused by damage from those caused by operational and environmental variations.

In this study, the debonding damage θ is fully characterized by two random variables, namely damage size, σ , and damage location along the beam, x . Thus, the damage is probabilistically characterized by the joint probability density function $p(\sigma, x)$.

For this problem, the damage diagnosis consists of two steps:

1. Estimate θ given a set of measurements $\mathbf{y}$ obtained from sensors placed on the structure under a certain load level. This is performed via Bayesian inference.
2. Obtain the buckling eigenvalue (λ) associated with the inferred damage parameters (θ) and load level. This can be done via FE model or surrogate model, which is much faster and more feasible in a time-sensitive decision.

The surrogate model that provides buckling eigenvalues was already detailed in Section 3.4, so the next section will focus on the Bayesian inference implementation and how these two steps can be combined to provide an estimation of how critical the current damage state is to the structure.

4.1 Bayesian inference

Bayesian inference is a statistical method that applies the principles of probability to update the state of knowledge about uncertain parameters, such as damage parameters in structural health monitoring applications, based on new data. It is grounded in Bayes' theorem, which provides a mathematical framework for updating the probability distribution of parameters as new data are observed.

The advantages of Bayesian inference include its flexibility in handling complex models and uncertainties, the ability to incorporate prior knowledge or expert opinions, the probabilistic interpretation of results, and the capability for sequential updating as new data become available. This makes it a powerful tool for continuously refining one's understanding of structural damage and making informed decisions based on probabilistic assessments of structural health. Overall, Bayesian inference provides a powerful framework for updating the understanding of structural damage as new data are collected, helping engineers and researchers make informed decisions based on probabilistic assessments of the structural health.

In this work, $\mathbf{y}$ are noisy strain measurements obtained by FBG sensors placed in discrete locations of the structure, $\mathbf{d}$, or equivalent computationally simulated measurements. In this context, Bayes' theorem can be written as follows:

$$p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{d}) = \frac{p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{d})p(\boldsymbol{\theta}|\mathbf{d})}{p(\mathbf{y}|\mathbf{d})} \quad (4.1)$$

where:

$p(\boldsymbol{\theta}|\mathbf{d})$ is the prior distribution of the damage parameters.

$p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{d})$ is the likelihood of the measured data.

$p(\mathbf{y}|\mathbf{d})$ is the evidence, and

$p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{d})$ is the posterior distribution of the damage parameters.

Notice that the prior distribution of the damage parameters is independent of the design (sensor placement), so the prior distributions can be rewritten as:

$$p(\boldsymbol{\theta}|\mathbf{d}) = p(\boldsymbol{\theta}) \quad (4.2)$$

The prior distribution is formulated using domain knowledge or expert opinion, as it represents an initial belief about the distribution of the damage parameters. As mentioned in Section 3.4, the sponsors of this project provided the prior distributions on damage size and location, based on non-destructive tests of the spars after manufacturing, which can be written as:

$$p(\sigma) = \text{Rayleigh}(50) = \frac{e^{(-\sigma^2/5000)}\sigma}{2500} \quad (4.3)$$

$$p(x) = \text{Uniform}(0, L) = 0.001 \quad (4.4)$$

which were independently plotted in Figure 3.1.

The likelihood function ($p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{d}) \equiv L(\boldsymbol{\theta})$) describes the probability of observing the data given specific values of the damage parameters. $L(\boldsymbol{\theta}_1) > L(\boldsymbol{\theta}_2)$ indicates that the observed measurement data is more likely for $\boldsymbol{\theta} = \boldsymbol{\theta}_1$ than for $\boldsymbol{\theta} = \boldsymbol{\theta}_2$. This function is crucial for updating the prior distribution based on the new data.

The evidence $p(\mathbf{y}|\mathbf{d})$ is the probability of observing the data under any possible value of the parameters. It is computed by integrating the product of the likelihood and the prior distribution over all possible values of the parameters. In practice, the evidence is simply a normalizing constant that ensures that the posterior distribution $p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{d})$ integrates to one. Therefore, Equation (4.1) can be simplified as

$$p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{d}) \propto p(\mathbf{y}|\boldsymbol{\theta}, \mathbf{d})p(\boldsymbol{\theta}) \quad (4.5)$$

4.2 Likelihood computation

The numerical evaluation of the likelihood in Bayesian inference depends on the specific problem and the nature of the likelihood function. If the likelihood function has a simple analytical form, it can be evaluated directly using mathematical expressions. For example, the likelihood for a normal distribution with known variance is simply the normal probability density function. For more complex likelihood functions, numerical integration techniques such as quadrature methods (e.g., Gaussian quadrature) can be used to approximate the integral of the likelihood function over the parameter space. Approximation methods, such as Laplace approximation or variational inference, can be used to approximate the likelihood function with a simpler, analytically tractable distribution.

In some cases, the likelihood function can be evaluated by simulating data from the model and comparing it to the observed data. This approach is common in Bayesian computation for models where the likelihood function is intractable, and it will be detailed below.

Consider a structural system subjected to a set of loads $\mathbf{F}$, and define the variable $\mathbf{v}$ as the system's response computed through a numerical method. More specifically, consider $\mathbf{F}$ as the maximum expected load applied to the model, and $\mathbf{v}$ the strain measurements from path 6 obtained from the FE model.

In statics, the measurements y can be related to the model's response through the following relation:

$$\mathbf{y} = \mathbf{L}\mathbf{v} + \boldsymbol{\epsilon} \quad (4.6)$$

where ϵ is the measurement noise. The related probability density function of the prediction error is defined as $p_{\epsilon}(\boldsymbol{\epsilon})$.

In Equation (4.6), $\mathbf{L}$ is the configuration vector, linking the measurements $\mathbf{y}$ to the

FE model's response $\mathbf{v}$. It is a Boolean vector, with either 1 or 0 values, corresponding to the measured and non-measured response components. In other words, it is a selection operator that discretize the locations where strain is actually measured on the structure, out of the complete vector $\mathbf{v}$ extracted from all the elements from path 6 on the FE model. $\mathbf{L}$ depends pm the spatial position of the sensors, hence $\mathbf{L} = \mathbf{L}(\mathbf{d})$ is a function of the design variable $\mathbf{d}$. Because the model's response $\mathbf{v}$ depends on the parameters $\boldsymbol{\theta}$, Equation 4.6 can be generalized as:

$$\mathbf{y} = \mathbf{G}(\mathbf{d}, \boldsymbol{\theta}) + \boldsymbol{\epsilon} \quad (4.7)$$

where

$$\mathbf{G}(\mathbf{d}, \boldsymbol{\theta}) = \mathbf{L}(\mathbf{d})\mathbf{v}(\boldsymbol{\theta}) \quad (4.8)$$

is the forward model operator that maps the design variables and the parameters to be estimated onto the measurements domain $\mathbf{Y}$.

Then, according to [70], the likelihood function may be numerically evaluated as

$$p(\mathbf{y}^i | \boldsymbol{\theta}^j, d) = p_{\epsilon}[\mathbf{y}^i - \mathbf{G}(\mathbf{d}, \boldsymbol{\theta}^j)] \quad (4.9)$$

which is the probability of measuring the noise represented by the difference between measurement $\mathbf{y}^i$ and model $\mathbf{G}(\mathbf{d}, \boldsymbol{\theta}^j)$. The index i represents a measurement obtained at a time instance i , or a database of measurements i . The index j represents a large number of samples that describe the damage parameters domain. The likelihood evaluation can be done by performing the following steps:

1. Define the Noise Model:

The statistical properties of measurement noise $\boldsymbol{\epsilon}$ need to be specified. It can be assumed that noise is normally distributed with mean 0 and standard deviation σ_{ϵ} , then

$$\boldsymbol{\epsilon} \sim N(0, \sigma_{\epsilon}^2) \quad (4.10)$$

2. Map the Model Predictions:

For a given set of parameters $\boldsymbol{\theta}^j$ and $\mathbf{d}$ that maps the damage parameter domain, use the computational model $\mathbf{G}$ to generate the predicted data $\mathbf{G}(\mathbf{d}, \boldsymbol{\theta}^j) = \mathbf{L}(\mathbf{d})\mathbf{v}(\boldsymbol{\theta}^j)$. j should consist of enough samples to represent the damage parameters spectra. This is rapidly achieved using the surrogate models, instead of the FE model.

3. Calculate the Residuals:

Compute the residuals, which are the differences between the observed data y^i and the model predictions G^j :

$$\mathbf{r}^{ij} = \mathbf{y}^i - \mathbf{G}(\mathbf{d}, \boldsymbol{\theta}^j) \quad (4.11)$$

4. Compute the Likelihood:

Given the residuals r and the noise model, calculate the likelihood of observing the data y given the parameters θ and d using Equation 4.9.

If the noise is normally distributed, the likelihood equation can be re-written as:

$$p(\mathbf{y}^i | \boldsymbol{\theta}^j, \mathbf{d}) = \prod_{k=1}^N \frac{1}{\sqrt{2\pi\sigma_\epsilon^2}} \exp\left(-\frac{\mathbf{r}_k^{ij^2}}{2\sigma_\epsilon^2}\right) \quad (4.12)$$

where N is the number of sensor, and r_k is the k -th residual. Notice that it was assumed that the noise is statistically independent in each sensor, so the productory represents the joint probability distribution of each residual r at a sensor k , and d is the sensor array, comprised of N sensors.

4.2.1 Maximum likelihood estimation

Maximum Likelihood (ML) estimation is a method for estimating the parameters of a statistical model by maximizing the likelihood function. The likelihood function measures how likely the observed data are given the model and its parameters. In other

words, MLE finds the parameter values that make the observed data most probable under the assumed model. Mathematically, it can be expressed as:

$$\hat{\boldsymbol{\theta}}_{ML}^i = \arg \max_{\boldsymbol{\theta}^j} p(\mathbf{y}^i | \boldsymbol{\theta}^j, \mathbf{d}) \quad (4.13)$$

To demonstrate this using an example, suppose that the wing spar is 1m long and it has a damage $\boldsymbol{\theta}^1 = \{\sigma = 0.04m, x = 0.006m\}$. The damage parameters are unknown to the user and need to be estimated.

Suppose that there are 14 sensors evenly spaced along the length of the spar. Then, a vector $\mathbf{y}^i = y_1, y_2, \dots, y_{14}$ can be obtained. The user generates $j = 6,000$ unique samples of damage, $\boldsymbol{\theta}^j$ (or any other arbitrarily large number). For each j , the strain measurement obtained via the surrogate model (noise free) is estimated for each of the 14 sensor locations and compared to the observed strain measurement y^i (noisy, since it is obtained from the sensors). The likelihood is then estimated following Equation (4.12).

Then, the damage state is inferred by selecting $\boldsymbol{\theta}^j$ that provided the maximum likelihood. This prediction is called $\hat{\boldsymbol{\theta}}^1$. Maximum likelihood results for $\boldsymbol{\theta}^1$ are shown in Figure 4.1 (a). A larger damage case $\boldsymbol{\theta}^2 = \{\sigma = 0.11m, x = 0.006m\}$ was tested at the same location, and the results are shown in Figure 4.1 (b).

A damage size of 4cm is relatively small, given that the effect of damage on the strain field is very localized and it only causes small disruptions in the strain measurements. Therefore, Figure 4.1 (a) shows several locations and sizes that are nearly as likely to occur. When the sensors measure data on a structure that has a larger damage (considering the same location), as shown Figure 4.1 (b), the maximum likelihood estimator is much more precise – one combination of size and location has a very high likelihood, and all the other likelihood values are very small (barely visible in the plot).

Another example will be provided in Section 4.3.

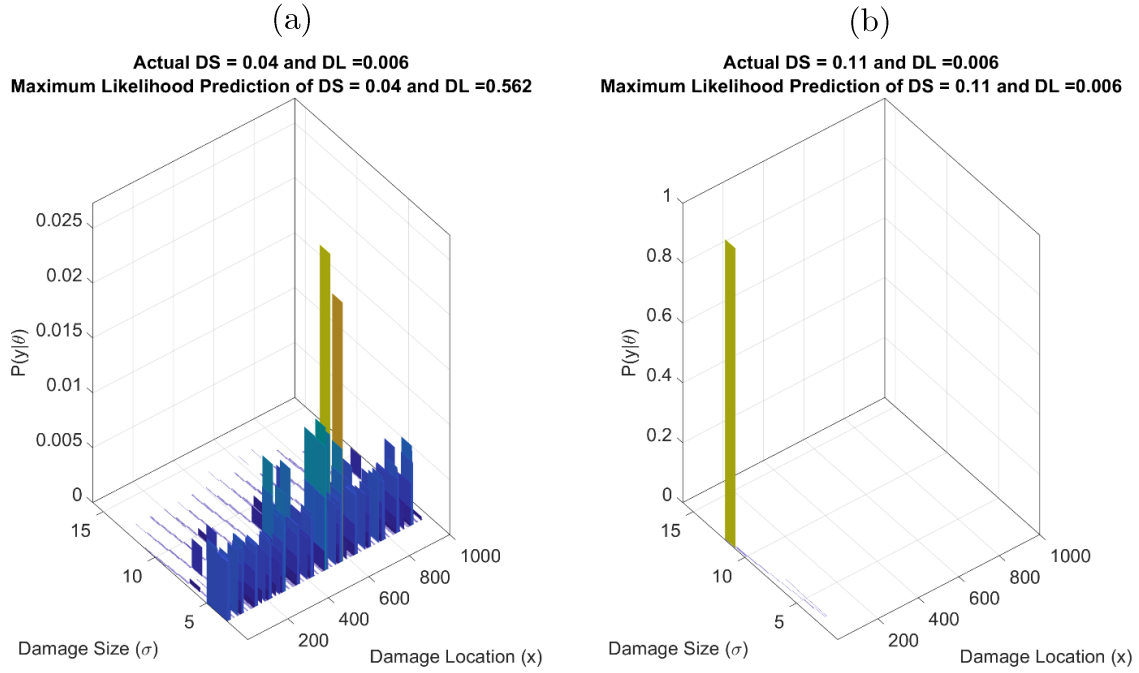


Figure 4.1. Prior distributions of damage size (a) and damage location (b).

4.2.2 Maximum a posteriori estimation

After the likelihood has been obtained, it can be multiplied by the priors to provide the posterior distributions of the damage parameters, $p(\boldsymbol{\theta}|\mathbf{y}, \mathbf{d})$, as described in Equation (4.1). Mathematically, it can be expressed as:

$$\hat{\boldsymbol{\theta}}_{MAP}^i = \arg \max_{\boldsymbol{\theta}^j} p(\boldsymbol{\theta}^j | \mathbf{y}^i, \mathbf{d}) \quad (4.14)$$

For now, it will be considered that only one batch of data is available. However, one of the main advantages of Bayesian inference is the capability of updating the model as new data become available. This makes it a powerful tool for continuously refining one's understanding of structural damage state, and it will be discussed in Section Model Updating.

Table 4.1. Seven classes of damage criticality, defined by the buckling eigenvalue, from not critical (Class 0) to most critical (Class 6).

Class	Eigenvalue
0	$\lambda > 1$
1	$0.9 < \lambda \leq 1$
2	$0.8 < \lambda \leq 0.9$
3	$0.7 < \lambda \leq 0.8$
4	$0.6 < \lambda \leq 0.7$
5	$0.5 < \lambda \leq 0.6$
6	$\lambda \leq 0.5$

4.2.3 Linking damage diagnosis to criticality

Once the damage parameters are estimated via Bayesian inference, the next step is estimating how critical that damage is. This is done using the GPR surrogate model that, given damage parameter inputs, estimates the buckling eigenvalue, as shown in Figure 4.2.

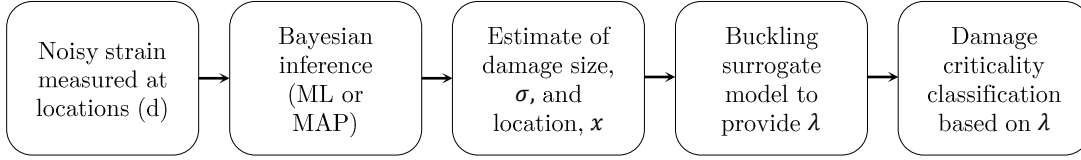


Figure 4.2. Flow chart of how a noisy strain measurement processed to be converted into damage criticality level.

Recall that Equation (2.1) define that $P_{critical} = \lambda \times P_{applied}$. Therefore, in a binary classification problem, the SHM system could define that “damage is critical (if $\lambda \leq 1$)” versus “damage is not critical (if $\lambda > 1$)”, which could carry implied actions, such as “stop operations for repair” or “continue operations”, respectively.

Damage criticality could also be classified in several different levels, such as shown in Table 4.1.

4.2.4 Performance of Bayesian inference model considering all sensors are present

Considering that FBG sensors will be implemented in practice, it is assumed that each sensor has a spatial resolution of 6mm and no sensor can be overlapped (their centroids must be at least 2mm apart). Therefore, there are $5000/6 = 833$ possible locations to place the sensors at path 6 along the longitudinal direction of the beam.

Following the classes specified on Table 4.1, 833 sensors were virtually placed on the 5m beam surrogate model. Then θ^i containing 1,000 unique damage cases were created, each case containing a batch of 10 measurements with random noise applied to them. The noise is defined as a Gaussian white noise, with 0 mean and $\sigma_\epsilon = 5e - 6$. This noise level was compatible with measurements obtained in-flight by the sponsors of this work, as shown in [45]. The damage location was distributed uniformly along the beam, while the damage sizes varied from 3cm to 15cm.

Since all damage cases were created computationally using the surrogate models, the true class labels are known, so the confusion chart can be generated. The confusion chart for damage classification is shown in Figure 4.3. It can be seen that the classification is perfect, which is expected if 833 sensors are placed on a single structure, which is a very unrealistic scenario that serves to check if the damage diagnosis model is working properly.

Figure 4.4 shows the mean squared error results. It is relevant to notice that the MSE is higher for small damage sizes, from 3 to 5cm, and for damage locations greater than 4.5 meters. This is an observability issue, that will be discussed in the upcoming section.

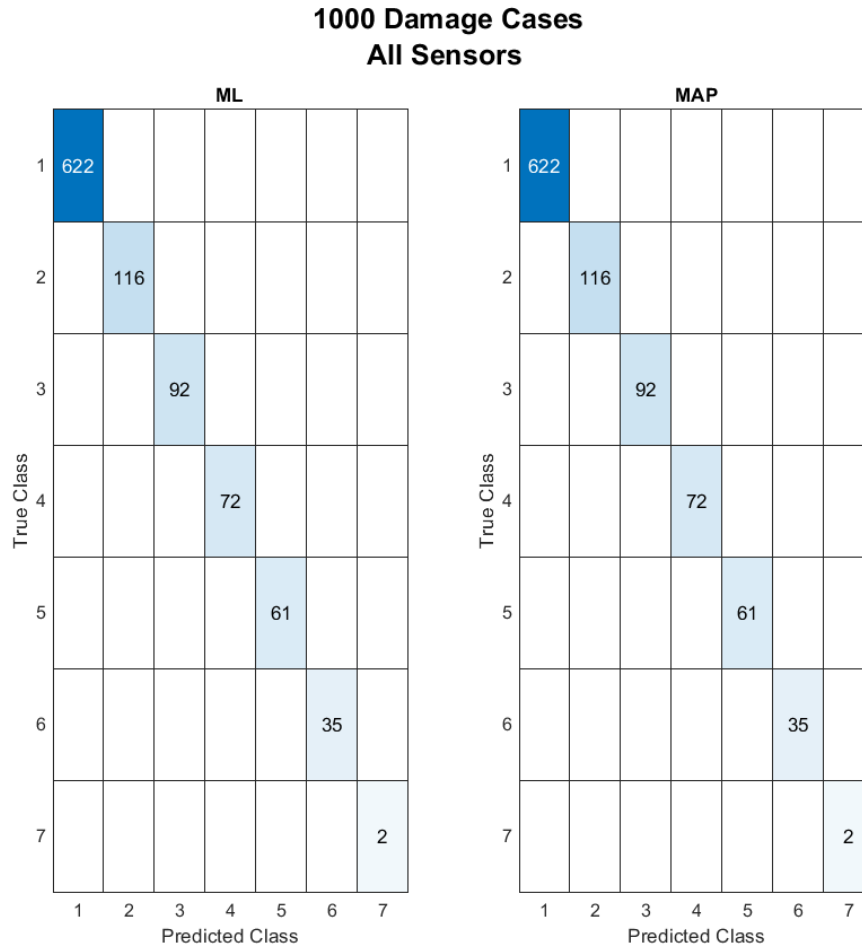


Figure 4.3. Confusion charts showing the performance of 833 sensors for criticality assessment using the maximum likelihood estimation (left) and maximum a posteriori estimation (right).

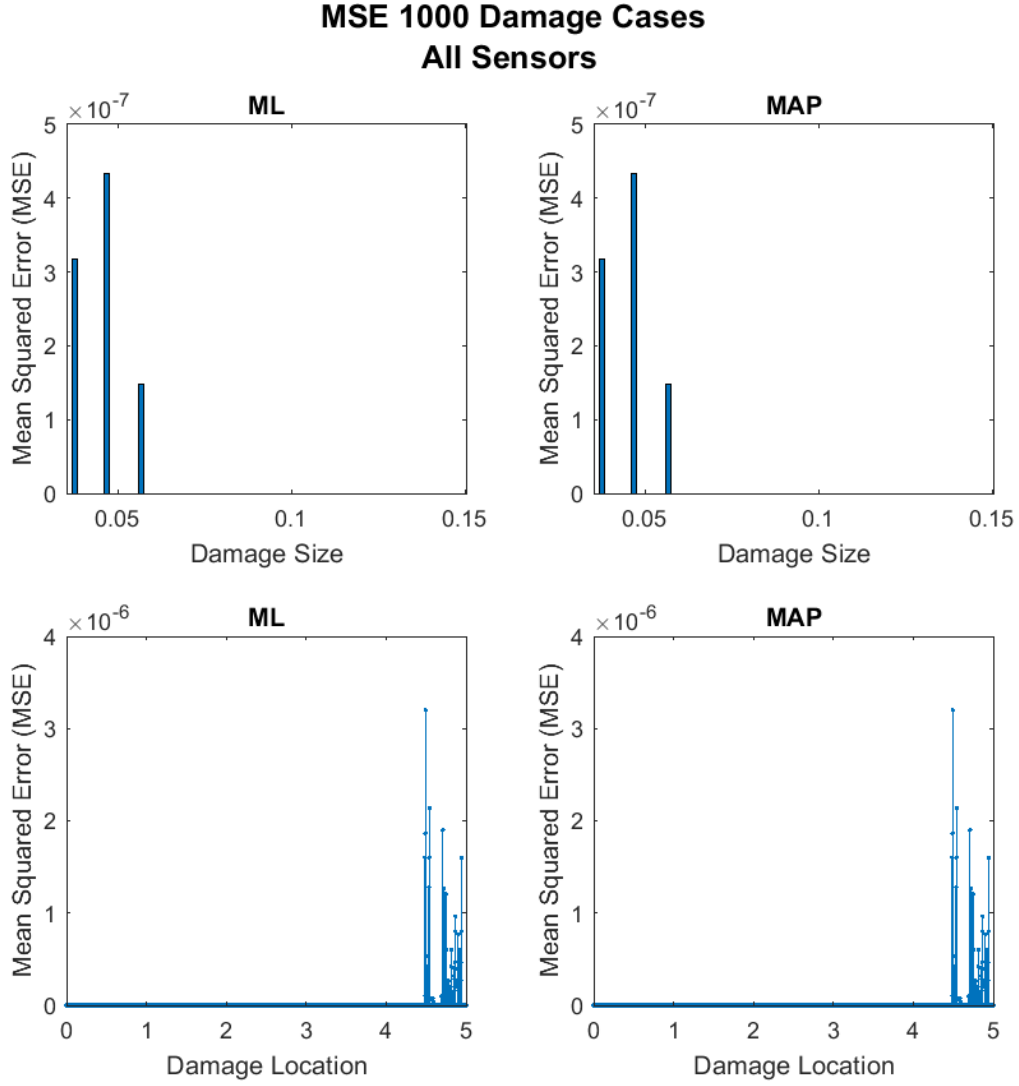


Figure 4.4. MSE from 833 sensors used for damage inference using the maximum likelihood estimation (left) and maximum a posteriori estimation (right).

4.3 Observability limitations

The low performance of the detector under low signal to noise ratio (SNR) is an observability issue. To understand the influence of the low SNR, recall that a key part of the damage detection is reconstructing the strain measurements. This is done based on the coordinates of 4 points (p1 to p4) shown in red in Figure 4.5.

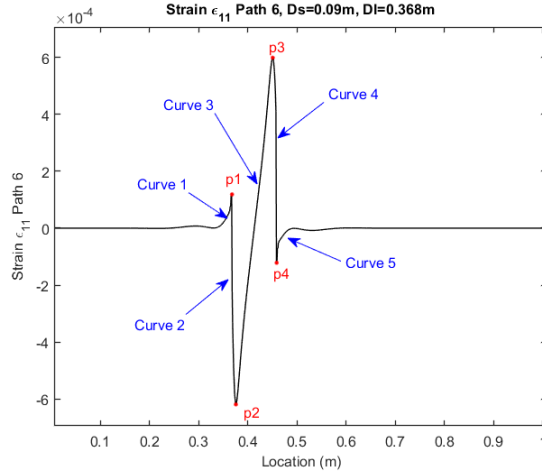


Figure 4.5. Strain signature of damage cases far from the loading region, after subtracting the baseline.

Figures 4.6, 4.7, 4.8, and 4.9 show the amplitude (y coordinate) of these points per x -coordinate (location along the beam). The orange points (right-axis label) represent the buckling eigenvalues per location. The black dashed lines represent the noise level, a Gaussian white noise with zero mean and $\sigma_\epsilon = 5e - 6$. The blue lines are the cases separated by damage size. It can be seen that points 2 and 4 are embedded in noise anywhere after 4 m, and even before that for some cases.

From strain observations from several damage cases, it was noticed that the region from 0 to 2 meters has absolute strain measurements of around $2,000\mu\epsilon$, while from 3 meters onward, the absolute strain measurements are smaller than $1,000\mu\epsilon$.

When a damage is present near the root, the strain measurement will have a large

amplitude, while the amplitude will be very low if the damage is close to the free end of the beam. This causes a poor SNR at anywhere after 3.5m on the beam, and specially from 4.5 to 5m, where the strain is minimal. This poor SNR generates a lot of uncertainty in the measurements, so the optimizations tend to place a lot of sensors in that region (see Figure 4.10), as will be discussed in the next chapter.

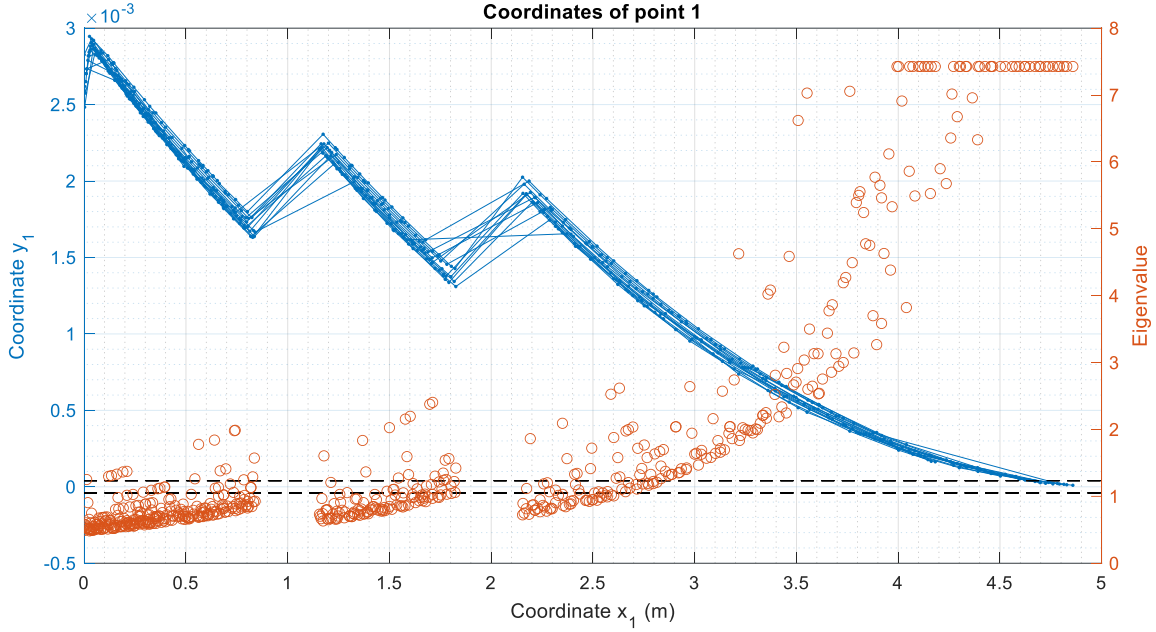


Figure 4.6. Left axis: Coordinates of point 1 compared to the noise level (black dashed line). Right axis: eigenvalues per x coordinate.

Figure 4.10 shows an example of how the Bayesian inference works (detailed in Section 4.1). Blue lines correspond to strain measurements obtained every 5mm (either on Abaqus or strain reconstruction surrogate model). Orange dots are where sensors were placed by the optimization, this is where “experimental” data would be collected, also known as observed noisy data. In a summary, the steps are the following:

1. Collect data (Figure 4.10, Top). In a real setup, data is already noisy and collected at discrete locations where there are sensors present. In this case, as the data is simulated, it is generated at every 5mm and Gaussian white noise $\epsilon \sim N(0, \sigma_\epsilon^2 \mathbf{I})$ is

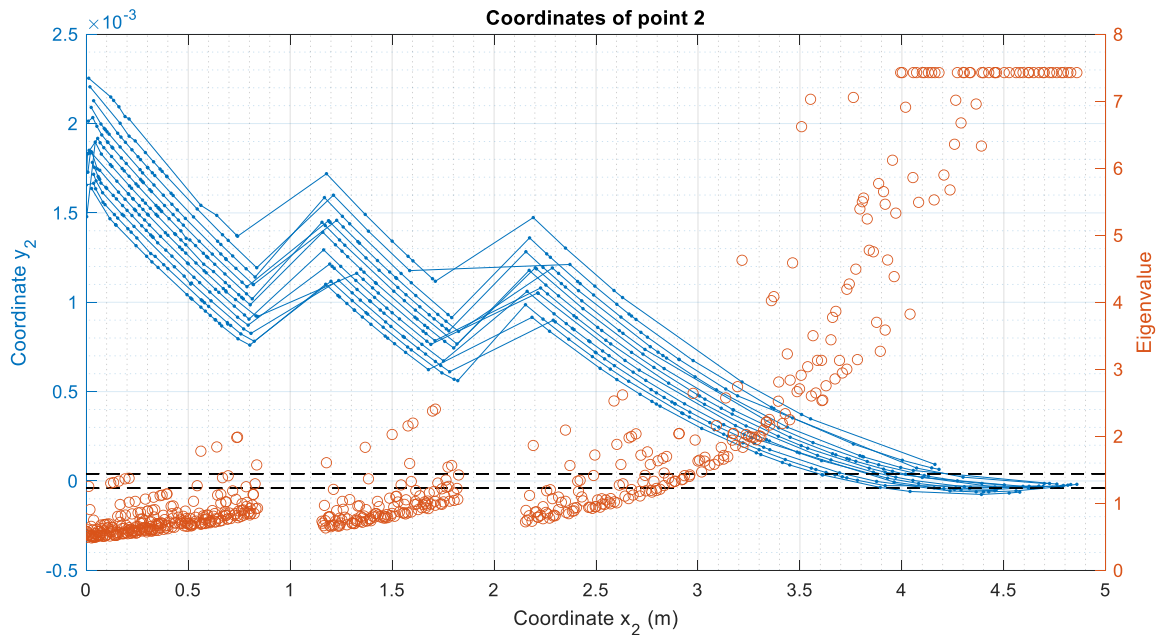


Figure 4.7. Left axis: Coordinates of point 2 compared to the noise level (black dashed line). Right axis: eigenvalues per x coordinate.

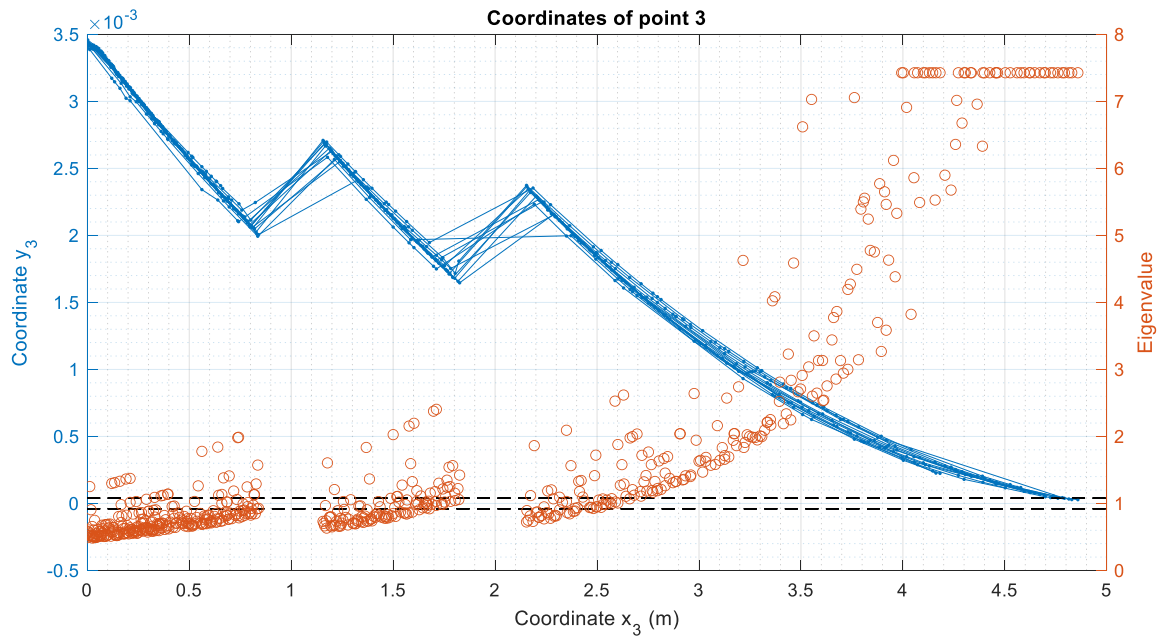


Figure 4.8. Left axis: Coordinates of point 3 compared to the noise level (black dashed line). Right axis: eigenvalues per x coordinate.

added to it. Then, the data is saved only where the sensors are located, the rest is thrown away.

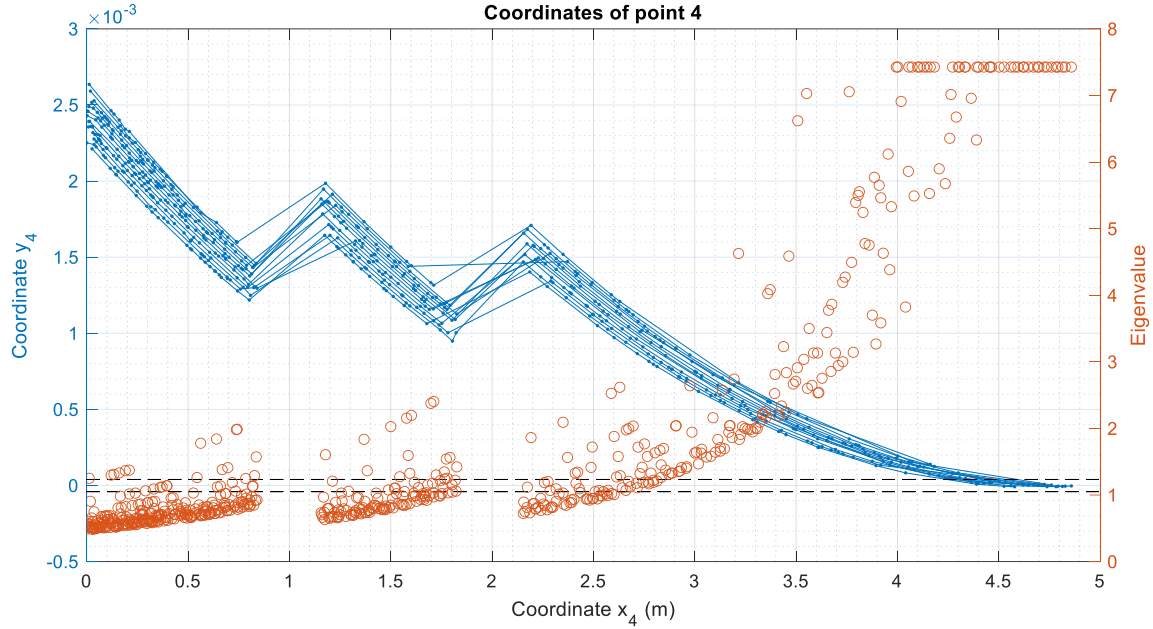


Figure 4.9. Left axis: Coordinates of point 4 compared to the noise level (black dashed line). Right axis: eigenvalues per x coordinate.

2. Compare this data with noise free data from damage from all possible damage size and location - for this simulation, $\theta^j = 31,912$ cases, which is the combination of all possible damage sizes defined from $0.03m$ to $0.15m$ in steps of $0.01m$, and damage locations from 0 to 5 meters, in $0.002m$ steps and considering boundary limitations. In this example, the true damage size is $0.03m$ long and true damage location is $0.762m$. In the middle picture, it is comparing the data to a randomly selected damage size of $0.05m$ located at $4.572m$ (there is a tiny peak if zoomed in).
3. Subtract the clean measurement from the noisy data (Figure 14, bottom). Compare what is left to the probability of the noise. If the data being compared is extremely similar to the experimental data (i.e., if the strain signature of the peaks and valleys are close enough), all that will be left is noise. Therefore, the likelihood of that being the damage present in the experimental data is very high.
4. Repeat for all $\theta^j = 31,912$ cases and choose the one with the highest likelihood (for

ML estimation).

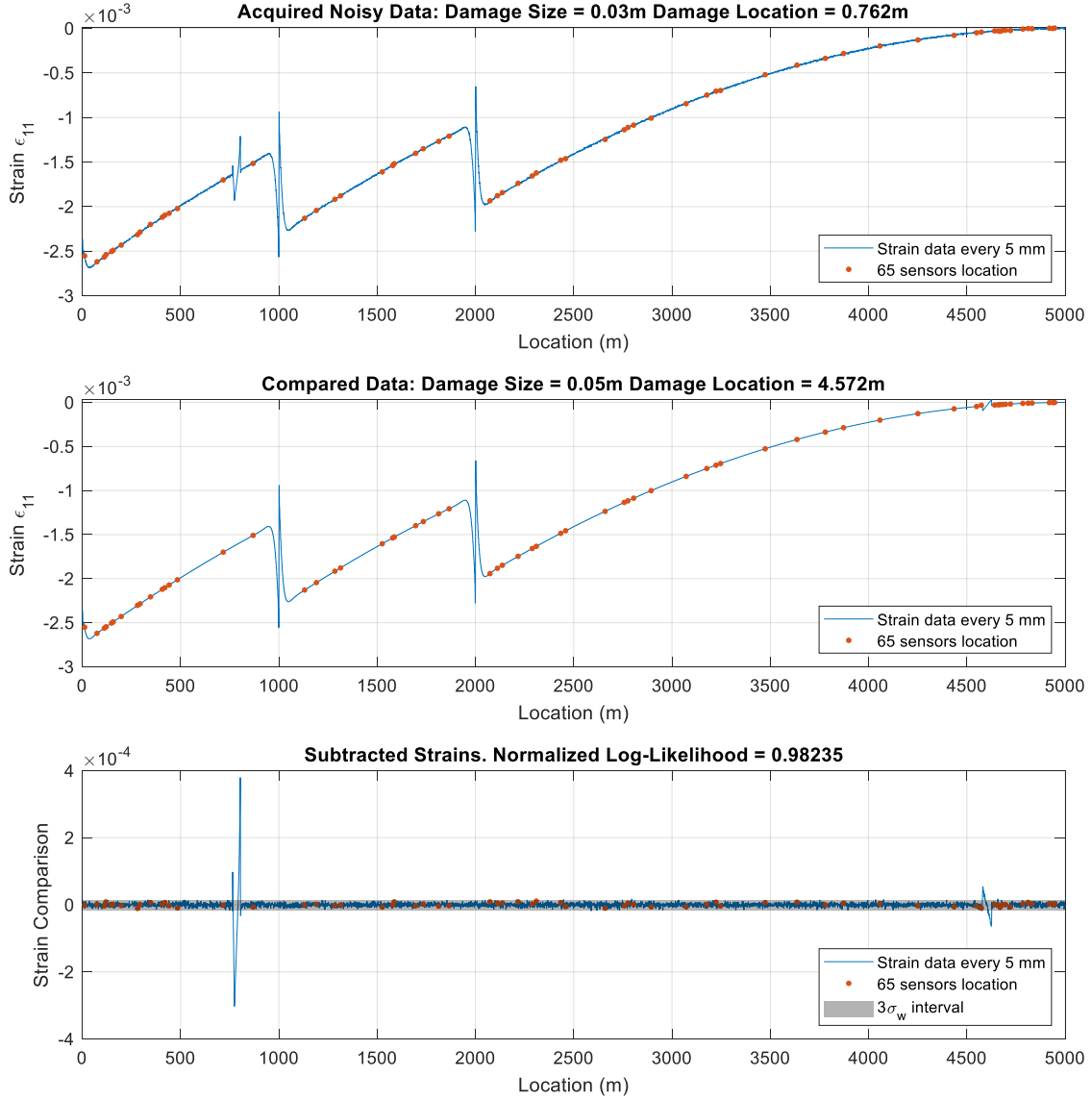


Figure 4.10. Example of how the Bayesian inference works. Blue lines correspond to strain measurements obtained every 5mm and orange dots are where sensors were placed by the optimization.

It can be seen in Figure 4.10 that the compared data is nothing alike the actual damage. However, the sensors are blind to that information. The sensors around 0.762m are far enough from it so that the strain measurements are not affected by the damage, as shown in the zoomed-in version in Figure 4.11, top. When the 5cm damage at 4.572m is

compared to the experimental data, the subtraction (where the sensors can measure it) is only noise. Therefore, it mistakenly thinks that this is the most likely damage location according to the likelihood function $p(\mathbf{y}^i|\boldsymbol{\theta}^j, d) = p_\epsilon[\mathbf{y}^i - \mathbf{G}(\mathbf{d}, \boldsymbol{\theta}^j)]$. The damage size is estimated correctly. In addition, Figures 4.6 to 4.9 show that almost all damage at 0.762 are critical ($\lambda < 1$), while no damage after 4 m is critical. Therefore, the prediction from Figure 4.12 is a false negative.

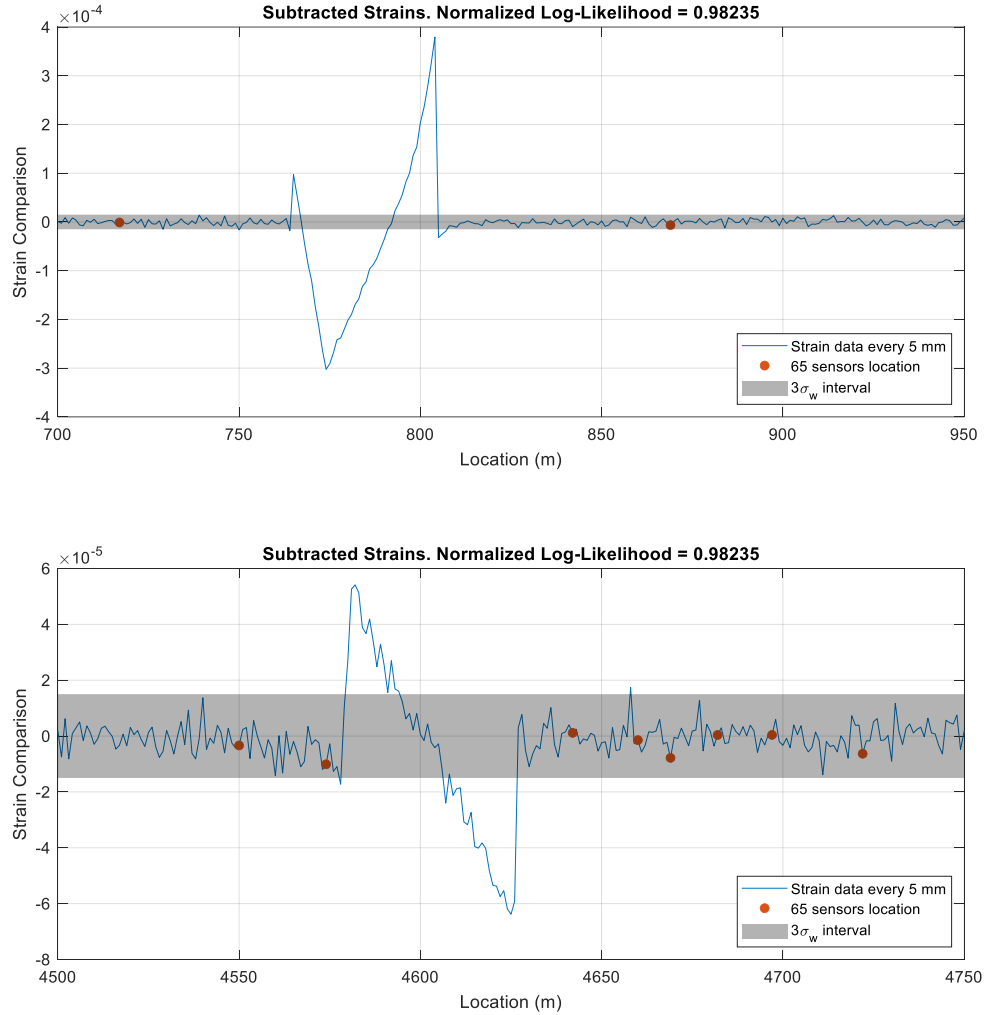


Figure 4.11. Zoomed-in view of Figure 4.10, bottom, of the subtracted strains.

A heatmap of the maximum likelihood and maximum a posteriori is shown in Figure Figure 4.12. The green cross indicates the true damage location, while the red circle indicates the prediction. The lighter the color in the heatmap, the higher the probability.

The same sensor array was used to predict a damage of size 7cm located at 0.392 m, as

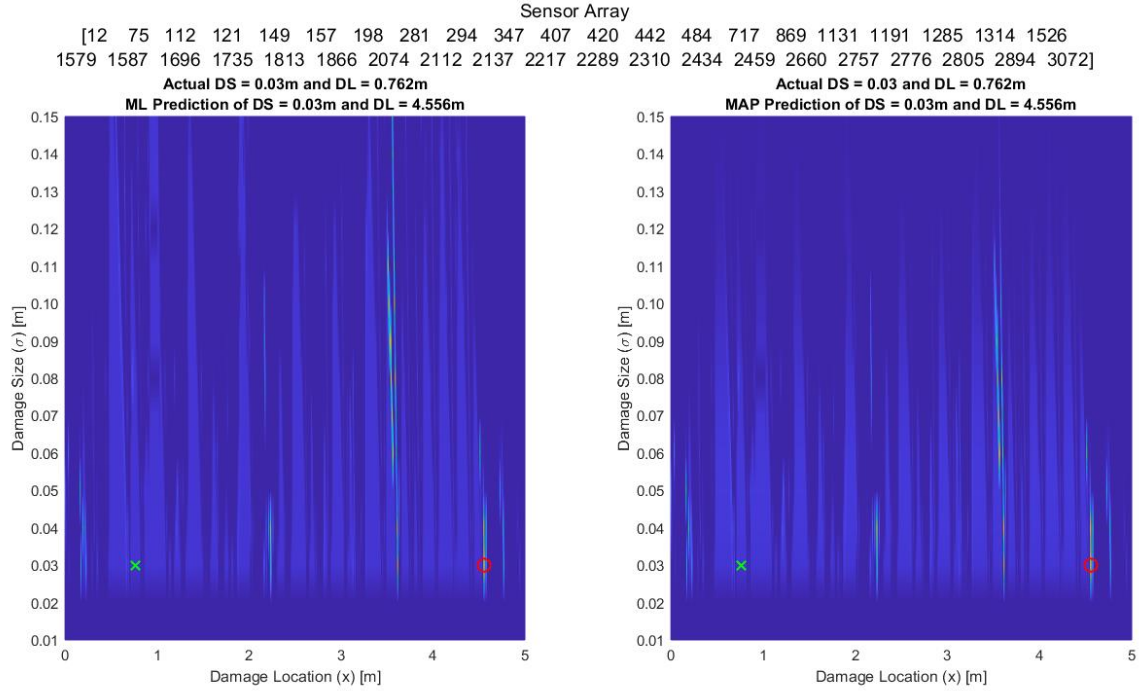


Figure 4.12. Incorrect prediction of damage location using a sensor array of 65 sensors.

shown in Figures 4.13 and 4.14. Notice that there are much more sensors in that region, including some that go over the damage. When subtracting both strain measurements, even though they are different, the remainder strain where the sensors are (orange dots) is mostly noise. This provided a likelihood of 98.76% of the damage being at that location and around that size. When all other cases are compared, the likelihood is calculated, and used to generate the heatmap. As shown in Figure 18, the prediction was precise in this case.

It will be discussed in the next Chapter that while evenly spaced sensors might not be the best option, they end up covering more regions that are prone to critical damage (i.e., from the root to 2m) than the optimized results (see orange dots on Figures 14 and 17), if the optimization is not set up appropriately. The optimized results are spending sensors at locations that are not important (after 4.5 m) and at the cost of neglecting

critical regions. With that, they end up biasing the performance by predicting that the damage is where the subtracted strain is only noise (where it has several sensors embedded in noise) and not where the damage actually is, as it is placed in a blind spot.

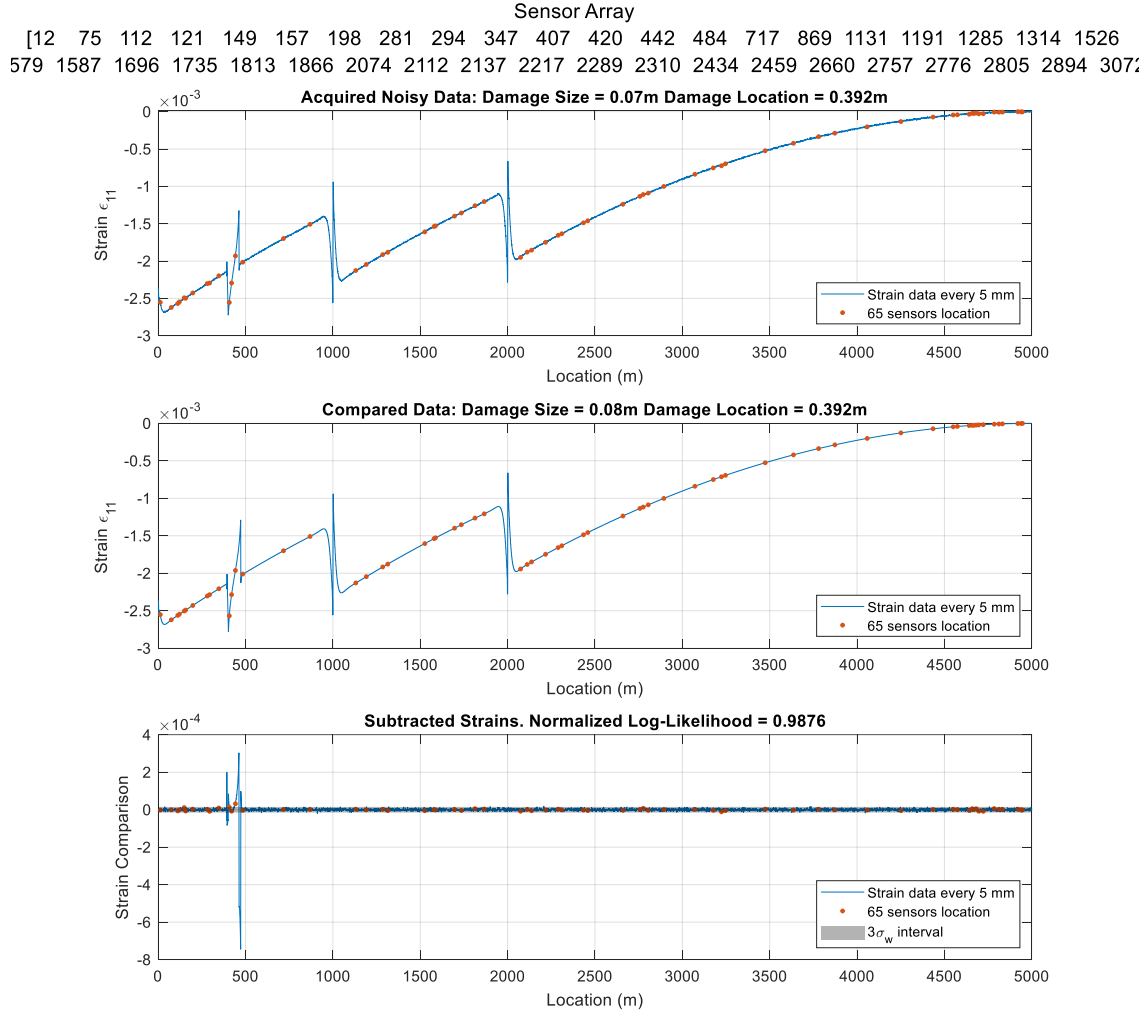


Figure 4.13. Another example of how the Bayesian inference works. Blue lines correspond to strain measurements obtained every 5mm and orange dots are where sensors were placed by the optimization.

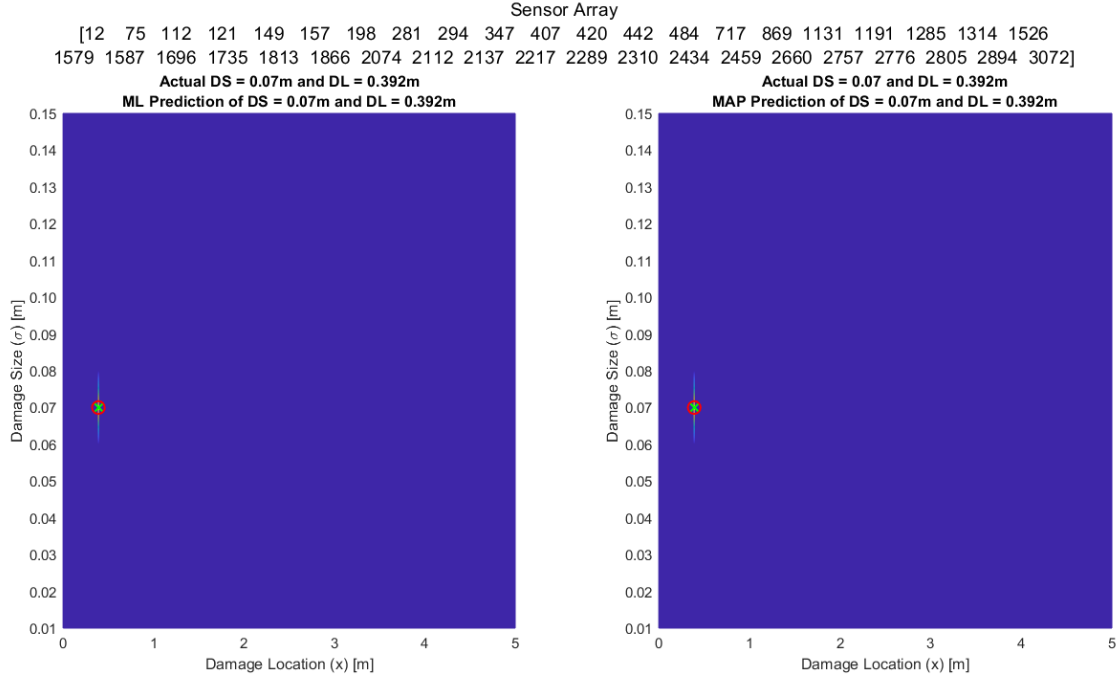


Figure 4.14. Correct prediction of damage size and location using a sensor array of 65 sensors optimized sequentially.

4.4 Bayesian model updating

It was mentioned in Section 4.2.2 that one of the main advantages of Bayesian inference is the capability of updating the model as new data become available. In simple terms, the estimated posterior distributions become the new priors of the subsequent prediction.

There are several ways to implement that numerically, and common Bayesian inference algorithms include:

- Markov Chain Monte Carlo (MCMC): MCMC methods, such as the Metropolis-Hastings algorithm and Gibbs sampling, are used to approximate the posterior distribution of parameters by generating a Markov chain.
- Variational Inference (VI): VI approximates the posterior distribution with a simpler distribution by minimizing the Kullback-Leibler (KL) divergence between them.

- **Sequential Monte Carlo (SMC):** SMC methods, also known as particle filters, approximate the posterior distribution by propagating a set of weighted particles through a series of sequential importance sampling and resampling steps.

This work uses the SMC method, and it will be detailed below.

4.4.1 Sequential Monte Carlo

When implementing Sequential Monte Carlo (SMC) for Bayesian inference, the likelihood $p(\mathbf{y}_t^i | \boldsymbol{\theta}_t^j, \mathbf{d})$ is typically assumed to follow a parametric distribution that models the noise in the data.

In this work the likelihood follows a Gaussian distribution, as defined in Equation (4.12). To incorporate the time-dependency, Equation (4.12) can be written as:

$$p(\mathbf{y}_t^i | \boldsymbol{\theta}_t^j, \mathbf{d}) = \prod_{k=1}^N \frac{1}{\sqrt{2\pi\sigma_\epsilon^2}} \exp\left(-\frac{\mathbf{r}_{kt}^{ij^2}}{2\sigma_\epsilon^2}\right) \quad (4.15)$$

Then, SMC is implemented as follows.

Initialization: Start by generating N particles (samples) from the prior distribution of $\boldsymbol{\theta}$, denoted as $p(\boldsymbol{\theta})$. Assign equal weight to each particle $\mathbf{w}_t = \frac{1}{N}$.

Sequential Importance Sampling (SIS): For each time step t , repeat the following steps:

1. **Prediction (evolution):** Propagate particles through the state transition model to predict their new states: $\boldsymbol{\theta}_t^j \sim p(\boldsymbol{\theta}_t | \boldsymbol{\theta}_{t-1}, \mathbf{y})$, such that the particles represent the updated state.
2. **Weight Updating:** Compute the likelihood of observing the data given each particle's state: $p(\mathbf{y}_t | \boldsymbol{\theta}_t)$.

Update the weights of particles using Bayes' rule:

$$\mathbf{w}_t \propto p(\mathbf{y}_t | \boldsymbol{\theta}_t) \times \mathbf{w}_t$$

3. **Normalization:** Normalize the weights so that they sum to 1:

$$\tilde{\mathbf{w}}_t = \frac{\mathbf{w}_t}{\sum_{i=1}^N \mathbf{w}_t}$$

4. **Resampling:** Resample particles with replacement according to their weights.

Particles with higher weights are more likely to be sampled. This step prevents particle degeneracy, where only a few particles carry most of the weight. After resampling, all particles will have equal weight again.

5. **Repeat:** Repeat steps 2-4 for each time step or iteration.

Posterior Estimation: After iterating through all time steps, the final set of particles represents samples from the posterior distribution $p(\boldsymbol{\theta}|\mathbf{y})$

$$p(\boldsymbol{\theta}|\mathbf{y}) \approx \sum_{i=1}^N \mathbf{w}_t \delta(\boldsymbol{\theta} - \boldsymbol{\theta}_t)$$

where δ is the Dirac delta function.

In summary, SMC methods use a set of particles to represent the posterior distribution and sequentially update these particles based on the observed data. By iteratively updating and resampling particles, SMC provides an approximation of the posterior distribution, particularly useful in cases where the posterior distribution is complex or cannot be computed analytically.

Results from model updating

Some results obtained after model updating are shown in Figures 4.15, 4.16, and 4.17. Figure 4.15 shows results for $\sigma = 0.03m$ and $x = 2.072m$. The first plot shows the maximum likelihood estimation of the damage state, and the remaining plots show the evolution of the maximum a posteriori estimation from the 1st to the 50th estimation. It can be seen that the uncertainty in the estimate decreases as more observations are added.

In Figures 4.16 and 4.17, only the 1st and 50th MAP predictions are shown. In Figure 4.16 it can be seen that the model is not able to estimate the correct damage location, but the prediction improves as more data becomes available. In Figure 4.17 it can be seen that the prediction is made correctly at the 1st iteration, but there is a lot of uncertainty around the damage size. At the 50th iteration, this uncertainty has decreased significantly.

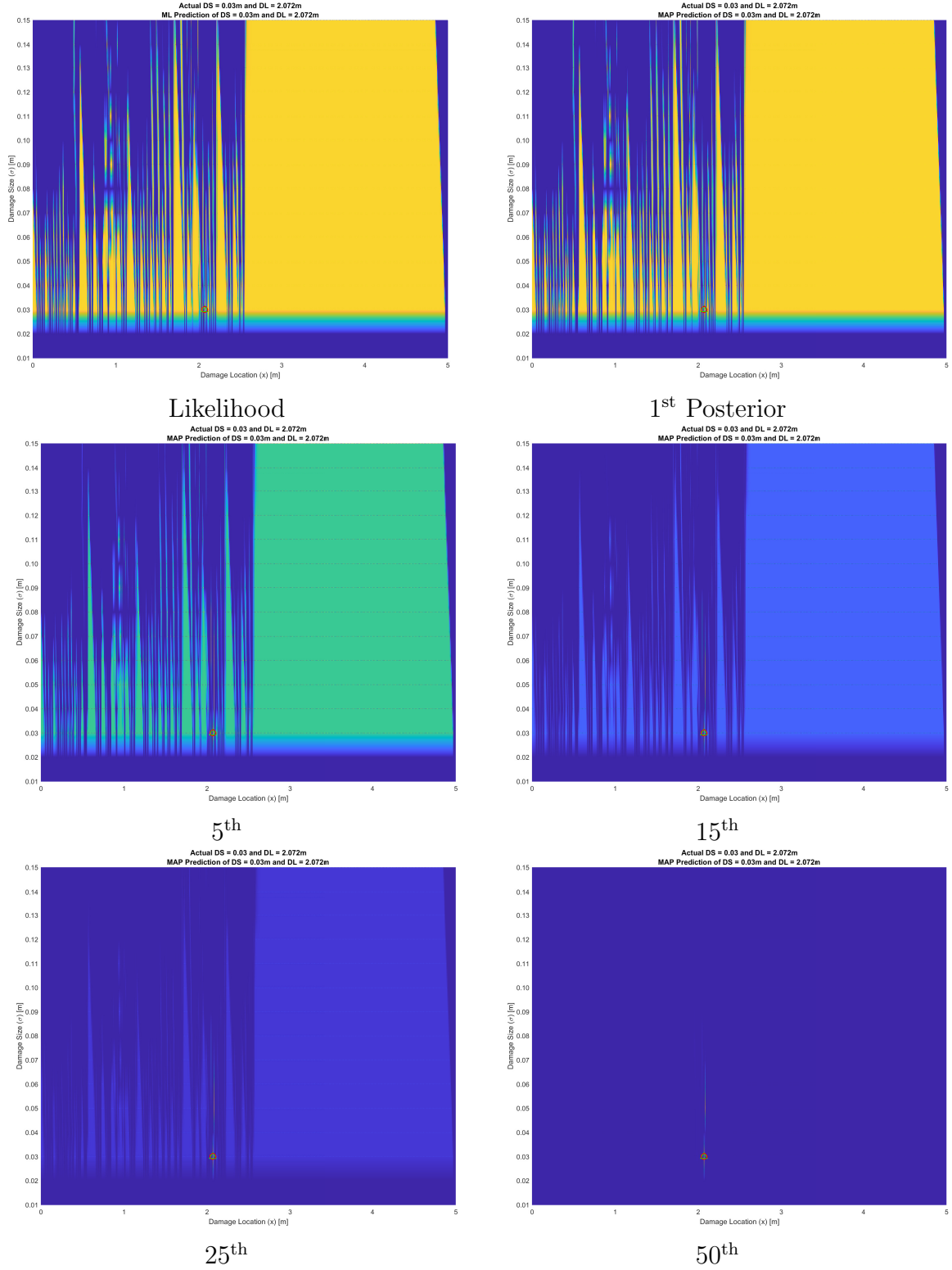


Figure 4.15. Bayes model updating for $\sigma = 0.03m$ and $x = 2.072m$. The first plot shows the maximum likelihood estimation of the damage state, and the remaining plots show the evolution of the maximum a posteriori estimation. Green triangle: true value, red circle: prediction.

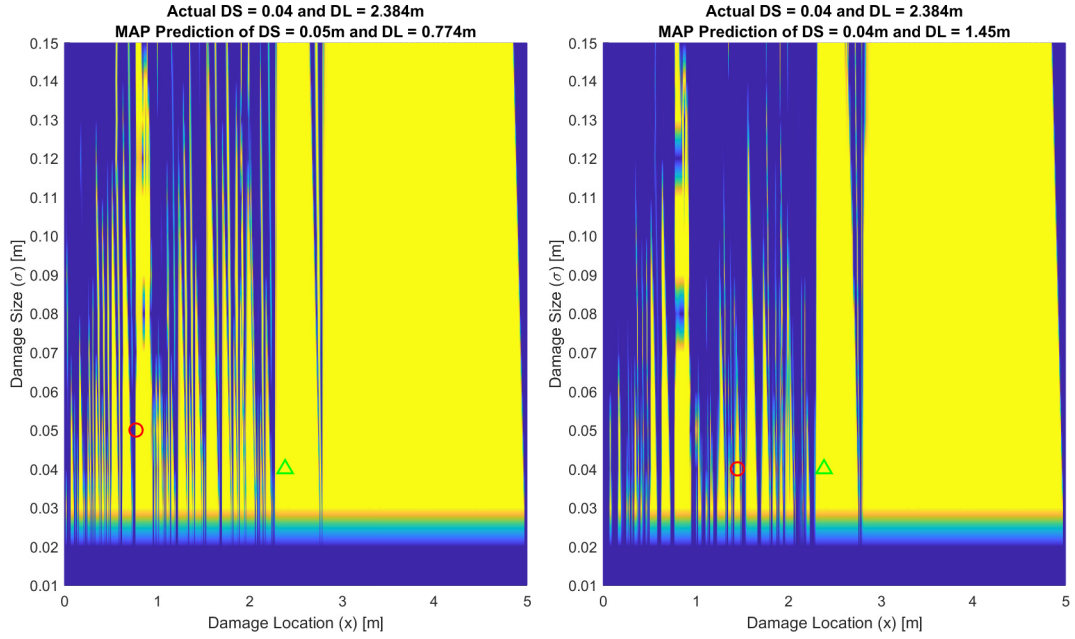


Figure 4.16. Bayes model updating for $\sigma = 0.04m$ and $x = 2.384m$. The first plot shows the 1st MAP estimate and the second plot shows the 50th MAP estimate. Green triangle: true value, red circle: prediction.

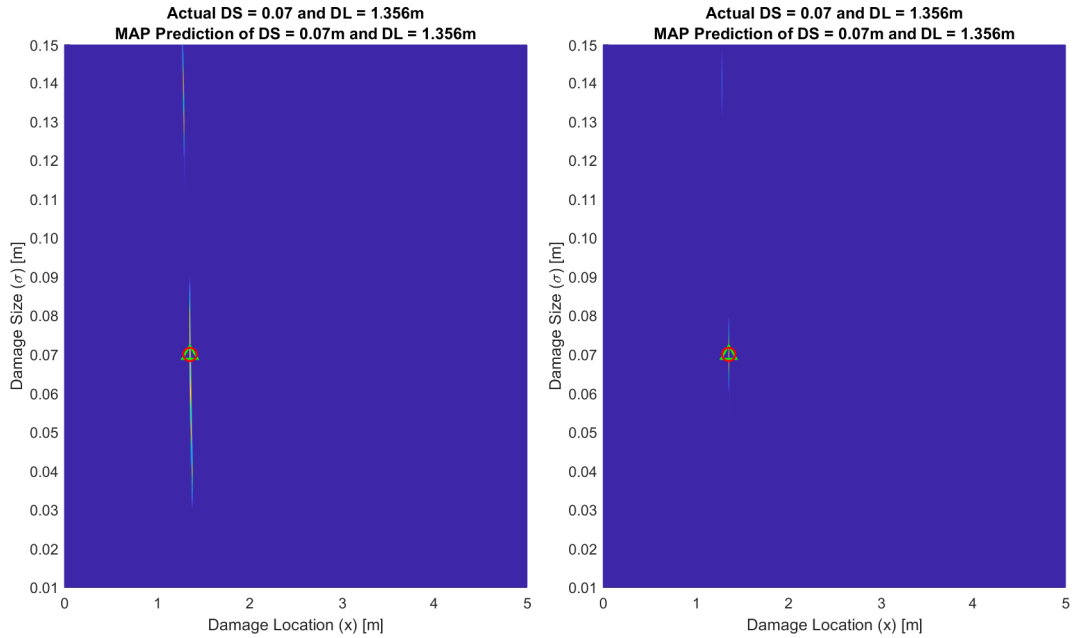


Figure 4.17. Bayes model updating for $\sigma = 0.11m$ and $x = 1.356m$. The first plot shows the 1st MAP estimate and the second plot shows the 50th MAP estimate. Green triangle: true value, red circle: prediction.

4.5 Summary

This chapter defined an entire framework that defines how measured strain data can be used on the Bayesian inference model to infer the damage state parameters, characterized by damage size and location, how the corresponding buckling eigenvalues are obtained and transformed in damage criticality level.

The entire Bayesian inference process was detailed, including the numerical approximation of the likelihood and posterior distributions. Overall, Bayesian inference provides a powerful framework for updating the understanding of structural damage as new data are collected, and this was shown by using sequential Monte Carlo to update the posterior distributions as more data became available. Observability issues in regions of low signal-to-noise ratios were also discussed. Chapter 5 details the last step of the OSP framework, which consists in defining the cost function to be minimized and implementing the chosen algorithm to do so.

Chapter 4, in part, has been published in *Data Science in Engineering, Volume 10. SEM 2023. Conference Proceedings of the Society for Experimental Mechanics Series*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., & Tur, M. , 2023. The title of this paper is “Optimal Fiber-Optic Sensor Placement Framework for Structural Health Monitoring of an Aircraft’s Wing Spar”. The dissertation author was the primary investigator and author of this paper.

Chapter 4, in part, has been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., Tur, M. , & Yehoshua, T., 2023. The title of this paper is “The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application”. The dissertation author was the primary investigator and author of this paper.

Chapter 5

Optimal Sensor Placement

A crucial step to implement a structural health monitoring system is defining the instrumentation to be used. From the operational evaluation step in the four-step statistical pattern recognition paradigm [21] the engineer designing the system must define the concept of damage for the system, understand the operational and environmental conditions, and determine data acquisition limitations. This evaluation helps tailor the SHM process to the specific features and unique aspects of the system being monitored.

In many applications, the damage state cannot be observed directly. Instead, sensors must be placed on the structure to obtain measurements that will be used to infer the damage state. Therefore, the type of damage being monitored will directly influence the selection of the type(s) of sensor(s) that will be used. For example, the detection of broken wires in suspension cable bridges is a completely different problem to monitoring debonding damage on composite structures.

The hardware of an SHM system is constrained by the costs associated with instrumentation, power consumption, and the impact on the structural system (such as weight, aerodynamics, and aesthetics), and locations that might not be accessible. To address these limitations, optimal sensor and/or actuator placement (OSP) strategies aim to maximize a chosen system performance metric. This presents two main challenges. The first is to derive an appropriate utility function for the system design, which should reflect

the intended purpose of the SHM system and may include constraints like placement limitations or sensor count. The second challenge is to develop an efficient optimization algorithm to select a design that maximizes or nearly maximizes the performance function.

A notable OSP approach for structural monitoring was developed by Kammer in 1991 [71], which focuses on minimizing a norm (like the trace or determinant) of the Fisher information matrix, known as the D-optimality criterion. F This approach is considered non-Bayesian (classical) as it does not assume any prior probability distribution of the state to be estimated. The D-optimality design criterion is particularly suitable for estimation problems.

However, the primary goal of SHM is not just estimation but also reducing the lifetime cost of a structure (in terms of money and/or safety) through estimation and decision-making. Therefore, the OSP problem is more appropriately addressed through a general framework based on Bayesian experimental design theory.

The framework proposed throughout this dissertation aims to optimize the allocation of sensing network resources to minimize the utility (cost) function, allowing for the optimization of sensor network designs considering non-uniform probabilities of damage and costs of detection errors.

5.1 Defining the cost function

The study mentioned in the introduction [71] is a classic example of finding the optimal sensor design, $\mathbf{d}^*$ that can maximize the gain in information. However, the goal of an SHM system is not necessarily to make the most accurate decisions, but rather to make the least costly ones in terms of economics and safety. In this perspective, a cost function that reflects this goal must be defined.

To formally define the costs associated with making a certain decision, let C_{ij} be the cost or regret of deciding that the structural state is $\mathbf{M}_i$ when the true state is

$\mathbf{M}_j$, where $\mathbf{M} = \{M_0, M_1, \dots, M_M\}$ is the set of all possible damage states. $\mathbf{M}$ can be {damage present, damage not present}, classes of damage types in a composite structure {debonding, delamination, cracks}, or anything else that represents the problem being studied. For this dissertation, where a link between the damage parameters (size and location) and failure is clearly defined, a certain damage can be classified as $\mathbf{M} = \{M_0 = \text{critical}, M_1 = \text{not critical}\}$. Therefore, C_{IJ} can be:

- C_{00} : true positive
- C_{01} : false positive
- C_{10} : false negative
- C_{11} : true negative

A confusion chart can be derived as follows:

		PREDICTED VALUES	
		Positive	Negative
ACTUAL VALUES	Positive	True Positive (TP) C_{00}	False Negative (FN) C_{10}
	Negative	False Positive (FP) C_{01}	True Negative (TN) C_{11}

Figure 5.1. Confusion chart and costs related to each prediction.

From the confusion chart (Figure 5.1), the probability of detection (POD), also known as the true positive rate (TPR), can be defined as:

$$\text{POD} = \frac{\text{True Positive}}{\text{actual critical cases}} \quad (5.1)$$

The false positive rate (FPR), also known as the probability of false alarm (PFA), can be defined as:

$$\text{PFA} = \frac{\text{False Positive}}{\text{actual non-critical cases}} \quad (5.2)$$

The false negative rate (FNR) or miss rate is:

$$\text{FNR} = \frac{\text{False Negative}}{\text{actual critical cases}} \quad (5.3)$$

And the true negative rate (TNR) is:

$$\text{TNR} = \frac{\text{True Negative}}{\text{actual non-critical cases}} \quad (5.4)$$

The cost of decision can be defined as:

$$C_{\text{decision}} = \sum_{i,j=0}^1 C_{ij} P(D_i|M_j) P(M_j) \quad (5.5)$$

Notice that C_{ij} can be defined such that it only penalizes incorrect decisions (false positives and false negatives), given that a correct damage detection (true positive) has intrinsic costs that should not be considered, i.e., the repair of a damage should be performed regardless of an SHM system being in place.

In addition, every SHM design incurs a cost that depends on its design parameters that drives the necessity of optimization. This cost can be the cost of the SHM hardware, the cost of installing and maintaining the SHM system, and even potential increase in operation costs of the structure (such as the extra weight in an aerospace system). In order for an SHM system to be feasible, its ability to decrease risk must be enough to at least offset the cost of the system. To construct a total cost function, the cost of action and the cost of design need to be in terms of the same units (usually monetary).

The design cost, C_{SHM} , can be defined for general applications as:

$$C_{SHM} = C_{\text{sensors}} + C_{\text{DAQ}} + C_{\text{ground station}} + C_{\text{installation}} \quad (5.6)$$

Table 5.1. SHM system design costs.

Item	Estimated cost (USD)
Single FBG sensor	100
Interrogator (4 channels, up to 10 FBG per channel)	30000
Ground station (without Non-Recurring Expenses)	10000
Installation	5000
UAV (one unit)	8M
False positive (unnecessary grounding for inspection)	5000
False negative – small repair (40mm)	10000
False negative – large repair (40mm)	15000
False negative – loss of a UAV	8M

For this study, the costs of the SHM system design are shown in Table 5.1. Initial costs for the false negatives and false positives are also shown, but they will be adjusted later on. The total cost, or Bayes risk, is then given by:

$$\hat{C}(d) = \frac{C_{\text{decision}} + C_{\text{design}}}{N} \quad (5.7)$$

where N is the number of samples used to evaluate the cost. The optimal sensor design, $\mathbf{d}^*$, can be defined as:

$$\mathbf{d}^* = \arg \min_d \hat{C}(\mathbf{d}) \quad (5.8)$$

5.2 Bayesian Optimization

The last step in the proposed OSP framework is defining which optimization function to be used and actually running the optimization. Section 1.5 mentioned a few algorithms that are used for OSP, such as Bayesian optimization, genetic algorithms, gravitational search algorithms, and particle swarm. This study uses Bayesian optimization to define the optimal sensor placement, as it is a powerful technique for optimizing complex functions that are expensive to evaluate.

It works by building a probabilistic surrogate model, usually a Gaussian Process, to approximate the unknown function. This model is then used to predict the function's

behavior and its uncertainty. An acquisition function, such as Expected Improvement or Upper Confidence Bound, is employed to decide where to sample next by balancing exploration of uncertain regions with exploitation of promising areas. Over successive iterations, the surrogate model is updated with new observations, and the acquisition function guides the search towards the global optimum.

Algorithm 1 shows how the optimization works. Then, algorithms 2 and 3 show how the algorithm is applied for optimal sensor placement in this study. The key difference between algorithms 2 and 3 is that, in the first one, the optimization is performed sequentially for k sensors, while algorithm 3 explains the combinatory optimization.

It is considered that each sensor has a spatial resolution of 6 mm and no sensor can be overlapped, thus there are $5000/6 = 833$ possible locations to place the sensors. Therefore, the two different ways that the optimization works can be explained as follows:

1. **Sequentially:** where one sensor is optimally defined. Then it is fixed in that location, and the next sensor is optimally defined. This is repeated until k sensors is reached. Therefore, the optimization dimension is 833, then 832, etc.
2. **Simultaneously (or combinatory):** in this case, the code optimizes for all k sensors at once. The dimension of the optimization is given by:

$$\overline{d_{N_{\text{sensors}}}} = \prod_{i=1}^{N_{\text{sensors}}} (n - i), \text{ where } n = 833 \quad (5.9)$$

Figure 5.2 shows an example of the Bayesian optimization analyzing the objective function and searching for its global minimum.

In this work, the acquisition function used was expected improvement, the surrogate model was a GP regressor, and the constraints consisted of strain resolution, boundaries, and minimal distance between sensors. The results from the optimization will be discussed in the upcoming sessions.

Algorithm 1. Bayesian Optimization Algorithm

- 1: **Input:** Cost function $C(\mathbf{d})$, surrogate model, acquisition function, and constraints
 - 2: **Output:** $\mathbf{d}_{\min}, C_{\min}$
 - 3: Initialize $\mathbf{D}_{\text{init}}$, evaluate $C(\mathbf{d})$ at $\mathbf{D}_{\text{init}}$ to obtain C_{init}
 - 4: Set $S \leftarrow \{(\mathbf{d}_i, C(\mathbf{d}_i)) \mid \mathbf{d}_i \in \mathbf{D}_{\text{init}}\}$
 - 5: **for** $t = 1$ to T **do**
 - 6: Fit surrogate model to S
 - 7: $\mathbf{d}_{\text{next}} \leftarrow \arg \min \text{AcquisitionFunction}(\mathbf{d} \mid S, \text{SurrogateModel})$
 - 8: Evaluate $C(\mathbf{d}_{\text{next}})$
 - 9: Update $S \leftarrow S \cup \{(\mathbf{d}_{\text{next}}, C(\mathbf{d}_{\text{next}}))\}$
 - 10: Optionally, check for convergence or stopping criteria
 - 11: **end for**
 - 12: $\mathbf{d}_{\min}, C_{\min} \leftarrow \arg \min_{(\mathbf{d}, C(\mathbf{d})) \in S} C(\mathbf{d})$
 - 13: **return** $\mathbf{d}_{\min}, C_{\min}$
-

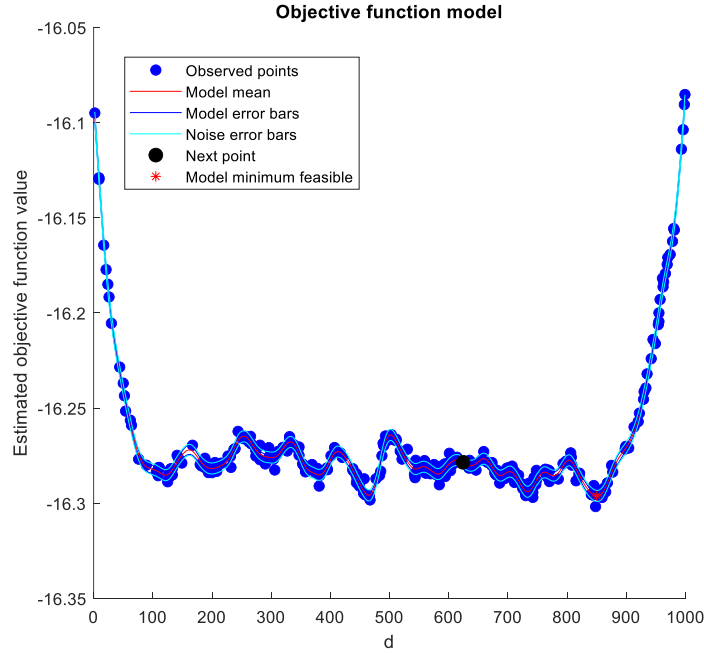


Figure 5.2. Bayesian optimization looking for the best location for the first sensor in a sequential optimization.

Algorithm 2. Bayesian Optimization for Sequential Sensor Placement in Structural Health Monitoring

- 1: **Input:** Number of output combinations N_i , Number of input combinations N_j , Strain forward model (ANN), Buckling forward model (GPR), Number of sensors k
 - 2: **Output:** Optimal sensor placement d^*
 - 3: Generate N_i combinations of damage size and location $\theta^i = \{\sigma, x\}$ following $P(\sigma) = \text{Rayleigh}(50)$ and $P(x) = \text{Uniform}(0, 1)$
 - 4: Generate N_j combinations of damage size and location $\theta^j = \{\sigma, x\}$ via permutation
 - 5: **for** $i = 1$ to N_i **do**
 - 6: Compute strain measurements $v(\theta^i)$ using ANN
 - 7: **end for**
 - 8: **for** $j = 1$ to N_j **do**
 - 9: Compute strain measurements $v(\theta^j)$ using ANN
 - 10: **end for**
 - 11: **for** $k = 1$ to k **do**
 - 12: Initialize Bayesian Optimization Algorithm (1) for the k -th sensor
 - 13: **while** not converged **do**
 - 14: Select a candidate sensor placement d_k by maximizing the acquisition function
 - 15: Specify the sensor array as $\mathbf{d} = \{d_1, d_2, \dots, d_k\}$
 - 16: Create Boolean representation of sensor placement $L(\mathbf{d})$
 - 17: Emulate measured strain $\mathbf{y} = G(\mathbf{d}, \theta^i) + \epsilon$, where $\epsilon \sim N(\mathbf{0}, \sigma^2 \mathbf{I})$
 - 18: Estimate likelihood $p(\mathbf{y}^i | \theta^j, \mathbf{d})$ using Equation 4.12
 - 19: Infer damage parameters using Maximum Likelihood Estimator (Equation 4.13)
 - 20: Estimate buckling eigenvalue λ using the forward GPR, and estimate the associated damage criticality
 - 21: Compute the averaged cost $\hat{C}(\mathbf{d})$
 - 22: Update the surrogate model with the new data point $(s, \hat{C}(\mathbf{d}))$
 - 23: **end while**
 - 24: Set the optimal sensor placement $\mathbf{d}^*$ for the k -th sensor
 - 25: **end for**
 - 26: **return** Optimal sensor placement $\mathbf{d}^*$ for all sensors
-

Algorithm 3. Bayesian Optimization for Combinatory Sensor Placement in Structural Health Monitoring

- 1: **Input:** Number of output combinations N_i , Number of input combinations N_j , Strain forward model (ANN), Buckling forward model (GPR), Number of sensors k
 - 2: **Output:** Optimal sensor placement d^*
 - 3: Generate N_i combinations of damage size and location $\theta^i = \{\sigma, x\}$ following $P(\sigma) = \text{Rayleigh}(50)$ and $P(x) = \text{Uniform}(0, 1)$
 - 4: Generate N_j combinations of damage size and location $\theta^j = \{\sigma, x\}$ via permutation
 - 5: **for** $i = 1$ to N_i **do**
 - 6: Compute strain measurements $v(\theta^i)$ using ANN
 - 7: **end for**
 - 8: **for** $j = 1$ to N_j **do**
 - 9: Compute strain measurements $v(\theta^j)$ using ANN
 - 10: **end for**
 - 11: Initialize Bayesian Optimization Algorithm (1) for all k -sensor at once
 - 12: **while** not converged **do**
 - 13: Select a candidate sensor array $\mathbf{d}$ by maximizing the acquisition function
 - 14: Create Boolean representation of sensor array $L(\mathbf{d})$
 - 15: Emulate measured strain $\mathbf{y} = G(\mathbf{d}, \theta^i) + \epsilon$, where $\epsilon \sim N(\mathbf{0}, \sigma^2 \mathbf{I})$
 - 16: Estimate likelihood $p(\mathbf{y}^i | \theta^j, \mathbf{d})$ using Equation 4.12
 - 17: Infer damage parameters using Maximum Likelihood Estimator (Equation 4.13)
 - 18: Estimate buckling eigenvalue λ using the forward GPR, and estimate the associated damage criticality
 - 19: Compute the averaged cost $\hat{C}(\mathbf{d})$
 - 20: Update the surrogate model with the new data point $(s, \hat{C}(\mathbf{d}))$
 - 21: **end while**
 - 22: **return** Optimal sensor array $\mathbf{d}^*$
-

5.2.1 Results from first optimization

For the first optimization, the cost of decision was defined following the confusion chart from Figure 5.3 as a binary classification problem, where a false negative costs \$8,000,000 and a false positive costs \$5,000. The costs were obtained from $N^i = 1,000$ samples of unique damage cases, obtained following the prior probabilities.

True Class	0	0	5k
	1	8M	0
		0	1
		Predicted class	

Figure 5.3. Costs for binary classification of 1st optimization.

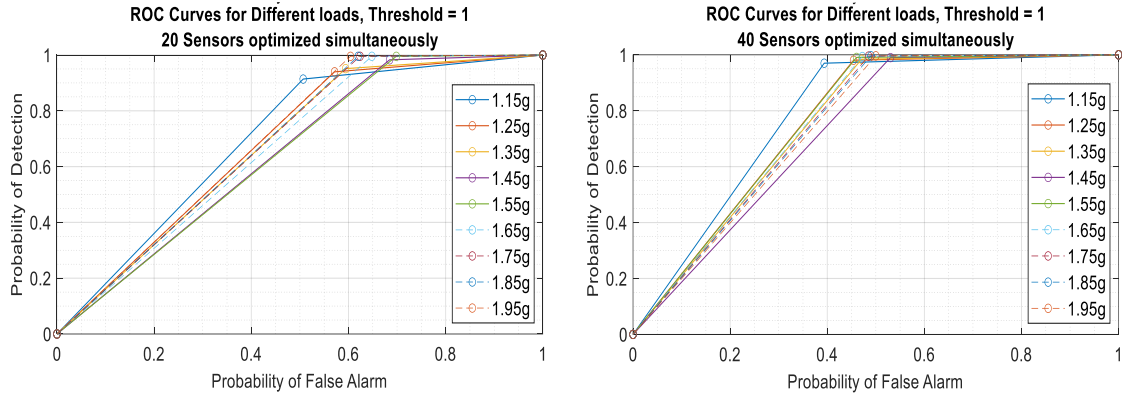


Figure 5.4. ROC curves for sequential optimization for different load levels. Left: 20 sensors, right: 40 sensors.

Figure 5.4 shows ROC curves for sequential optimization for different load levels., for 20 and for 40 sensor locations optimized sequentially. These different load levels follow Table 3.1. It is assumed that that the structure is always under loads that can be linearly scaled from maneuver 9 (1.95g). I.e., if the structure is under maneuver 8, the applied loads correspond to 94.87% of the maximum service load, and the strain measurements can be scaled to represent the loads from maneuver 8. The buckling eigenvalues can also be

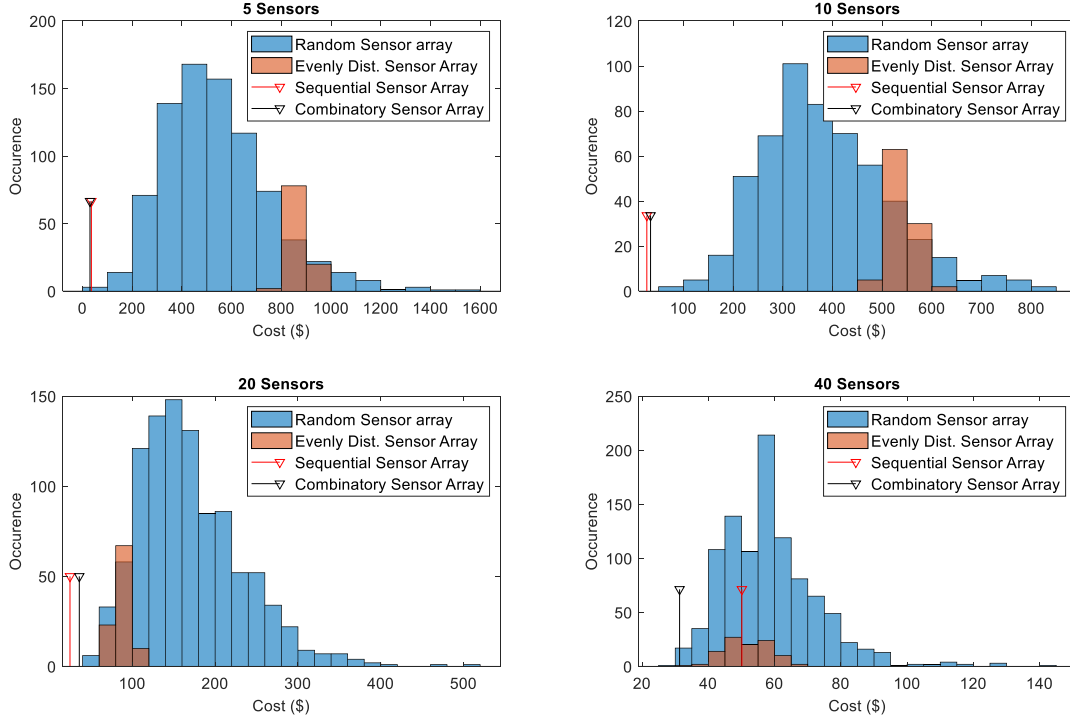


Figure 5.5. Results from the Bayesian optimization for different number of sensors. Notice that the axes do not have the same limits.

scaled, given that $P_{critical} = \lambda \times P_{applied}$, as shown in Equation (2.1). Therefore, a damage case that causes a $\lambda < 1$ (i.e., critical) at the maximum service load, may not be critical under other maneuvers.

On Figure 5.4, it can be seen that except for the lowest load level (1.15g), the probability of detection was around 98%. This nearly perfect POD comes at the cost of a very high PFA. This comes at no surprise, given that the cost of a false negative is 1,600 times higher than the cost of a false positive. Therefore, the optimization chooses the sensor set that avoids false negatives at all costs.

Figure 5.5 shows a histogram in which the cost was evaluated for 1,000 different randomly selected sensor arrays. Then, the same damage cases were evaluated multiple times by evenly distributed sensor arrays. Finally, the damage cases were used to evaluate the cost obtained via optimization, both sequential (red) and combinatory (black). It can

be seen that the optimized sensors consistently perform better than randomly selected sensors. The evenly distributed sensors are a very poor option when there are only 10 sensors to be placed on the structure, but their performance improves as more sensors are allowed. Finally, both sequential and combinatory optimization had a similar performance, but for 40 sensors, the combinatory sensor placement had a better performance.

5.2.2 Results from second optimization

For the first optimization, the cost of decision was defined following the confusion chart from Figure 5.6 as a binary classification problem, where a false negative costs \$8,000,000 and a false positive costs \$2,400,000. The costs were obtained from $N^i = 1,000$ samples of unique damage cases, the strain measurement of each damage cases consisted of a batch of 10 samples. The samples were obtained following the prior probabilities.

True Class	0	0	2.4 M
	1	8M	0
		0	1
		Predicted class	

Figure 5.6. Costs for binary classification of 1st optimization.

Figure 5.7 shows ROC curves for sequential optimization for the same load level (1.95g) for 10 and for 40 sensor locations optimized sequentially and as a combinatory. It can be seen that the change in cost function completely shifted the ROC curves for 40 sensors. The probability of detection went down to approximately 86% for the combinatory results, a 12% decrease, but the PFA went down around 40%. This means that there are much less interruptions in operations in this case.

Figure 5.8 shows a histogram in which the cost was evaluated for 1,000 different randomly selected sensor arrays. Then, the same damage cases were evaluated multiple

times by evenly distributed sensor arrays. Finally, the damage cases were used to evaluate the cost obtained via optimization, both sequential (red) and combinatory (black).

Notice that these histograms cannot be directly compared with the histograms from Figure 5.5, as the costs used to obtain the performance are completely different. With the exception of the cost value on the x-axis, the trends observed can still be compared. Once again, both optimized results performed better than the randomly distributed and the mean evenly distributed value. The combinatory case performed better than the sequential array.

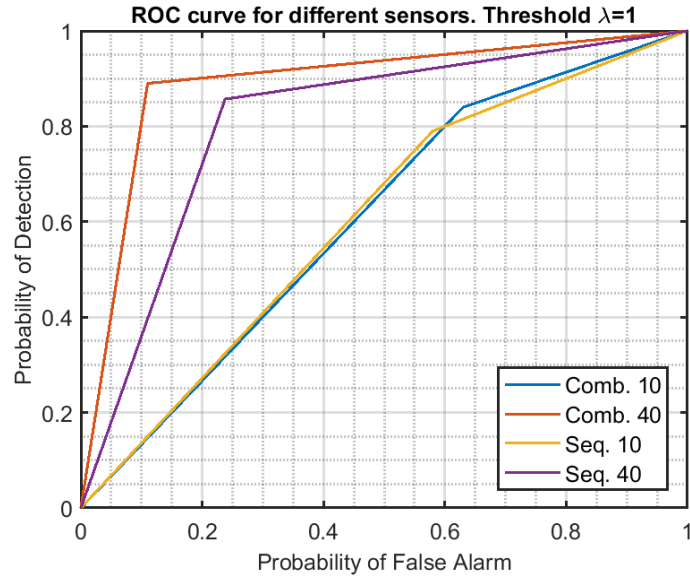


Figure 5.7. ROC curves for sequential optimization for different load levels, 2nd run.

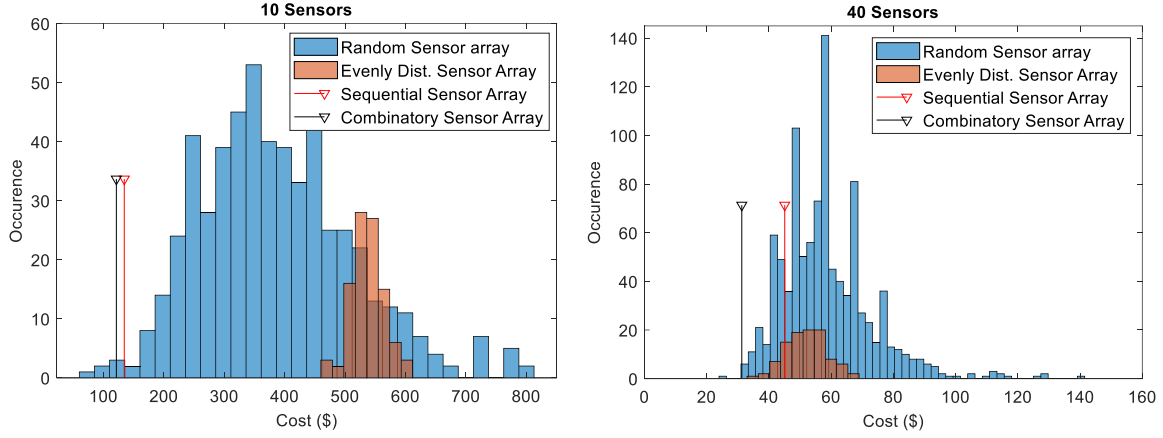


Figure 5.8. Results from the 2nd Bayesian optimization for different number of sensors. Notice that the axes do not have the same limits.

5.2.3 Results from the third optimization

With lessons learned from the first and second optimization runs, some major updates were made.

The cost function is not defined as a binary classification anymore. Instead, there are 7 different damage critically levels, as defined in 4.1. Class 0 means that the damage is not critical $\lambda > 1$, class 1 is the least critical damage with $0.9 \leq \lambda$, and so on, until class 6 is defined as the most critical damage, with $\lambda < 0.5$. Recall that λ links the applies load to a load in which local buckling will occur under the presence of debonding damage. Therefore, $\lambda = 0.5$ means that the structure will buckle locally with 50% of the applied load, while cases with $\lambda > 1$ will not suffer buckling at the ultimate load level.

Thus, a new cost function was built with the idea that different λ value represent different damage criticality, and their classifications should be treated differently. For example, a false negative that has $\lambda_{true} = 0.98$ but $\lambda_{estimated} = 1.02$ is not nearly as critical as a $\lambda_{estimated} = 0.49$ that is misclassified as not critical. This new cost function is represented in Figure 5.9

In addition, new constraints were applied to the model. After analysis of the observability issues from Section 4.3, it was concluded that there was no advantage in

0	0	200 k	400 k	600 k	800 k	1M	1.2 M
1	250 k	0	200 k	400 k	600 k	800 k	1M
2	500 k	250 k	0	200 k	400 k	600 k	800 k
3	1M	500 k	250 k	0	200 k	400 k	600 k
4	2M	1M	500 k	250 k	0	200 k	400 k
5	4M	2M	1M	500 k	250 k	0	200 k
6	8M	4M	2M	1M	500 k	250 k	0
	0	1	2	3	4	5	6

Predicted class

Figure 5.9. Costs for binary classification of 3rd optimization.

placing sensors anywhere after $x > 3.5m$, given that the signal will present a low SNR. This poor SNR generates a lot of uncertainty in the measurements, so the optimizations tended to place a lot of sensors in that region to minimize that uncertainty. By placing these now constraints, the optimization is forced to ignore these regions and place sensors where the presence of damage will, in fact, cause local buckling to occur.

The costs were obtained from $N^i = 1,000$ samples of unique damage cases, the strain measurement of each damage cases consisted of a batch of 10 samples. However, in this third run, these samples were not obtained from the prior distributions anymore, but they were randomly sampled from damage cases that caused $\lambda < 1.1$. This guarantees that the sensors are trained to classify critical damage cases, and not any damage case that might not even be critical to the structure.

5.2.4 Results and Summary

Results from the third optimization are shown below. Figure 5.10 shows the cost per number of sensors. Since the sequential optimization chooses one sensor at a time, it is possible to calculate the cost every time that a new best sensor is added to the structure. It can be seen that the cost decreases as more sensors are added, as the improvement in decision making offsets the cost of adding a new sensor ($\$100/N_{samples}$). However, a new data acquisition system is needed after 40 sensors are allocated, so the cost of ($\$30,000/N_{samples}$) is not worth the improvement in information gain. It is also seen that the sequential optimization performed better than the combinatory one. This occurred because the sample space tends to infinity in the combinatory optimization, so it needs many more evaluations to provide a good result. The combinatory optimization ran for about 50% less times then the sequential one, given its computational costs.

Figure 5.11 shows the histogram. Notice that this histogram cannot be directly compared with the histograms from Figure 5.5 and 5.8, as the cost function is defined differently for each of them. Regardless of that, the overall trend is still the same, where the sequential and combinatory optimization performed much better than the random sensor placement.

Figures 5.12, 5.13, and 5.14 show the confusion charts for the evenly spaced, combinatory, and sequential sensor placements, respectively. These figures show that the sensors are performing as they were designed to - misclassification between 1 level of different classes occurs as they were not heavily penalized, but no damage of classes 5 or 6 goes undetected.

Figures 5.15, 5.16, and 5.17 show the ROC curves. Notice that although the sensors were trained as a multi-class cost, the results are treated as a binary classification for the ROC curves, so that they can be compared to the previous ROCs. In this case, damage is simply said to be critical if $\lambda_{estimated} < 1$. Figure 5.15 shows that 45 sensors provide a

better performance than only having 40 sensors, but this comes at a higher cost, as shown in Figure 5.10.

Comparing Figure 5.15 to results from the first and second optimizations, the performance improvement after updating the cost function is clear. For 40 sensors obtained sequentially, the POD is around 94%, with a PFA of around 16%. By comparing the ROC curves from the 1st optimization (Figure 5.4) to Figures 5.16 and 5.17, it can be seen that for lower load levels, the new POD is around 98%, a value equivalent to the 1st optimization, but the PFA now is close to 0%, and for the highest load level the POD is around 94%, with a PFA of around 17%.

Table 5.2 shows the layout obtained after each run. Notice that the results for the sequential optimization are shown in the same order as the sensors were selected, while the combinatory optimization selects all 40 sensors at once. Finally, Table 5.3 compares the cost, POD, and PFA of the 40 sensors layout obtained after each run. Once again, it can be seen that the third run of the sequential optimization provided the best performance and the lowest cost. The final layout after the third run is shown in Figure 5.18, where the pristine data from path 6 is shown as a blue line, and the sensors location are shown as red dots.

These results emphasize the importance of rigorously defining and testing a cost function that reflects the goal of the SHM system being implemented. Having a computational test bed to do so allows for trial and error, and to learn more about the system's behavior before the sensors are permanently placed on the structure.

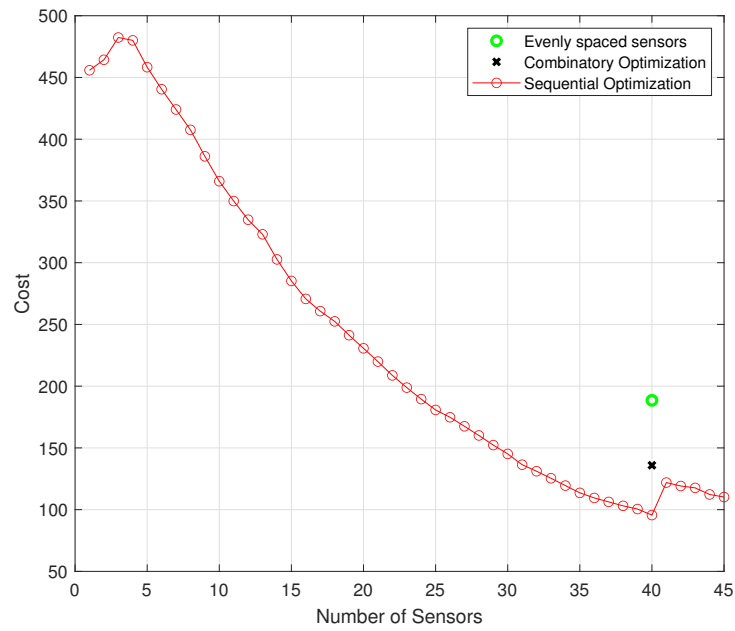


Figure 5.10. Cost per number of sensors.

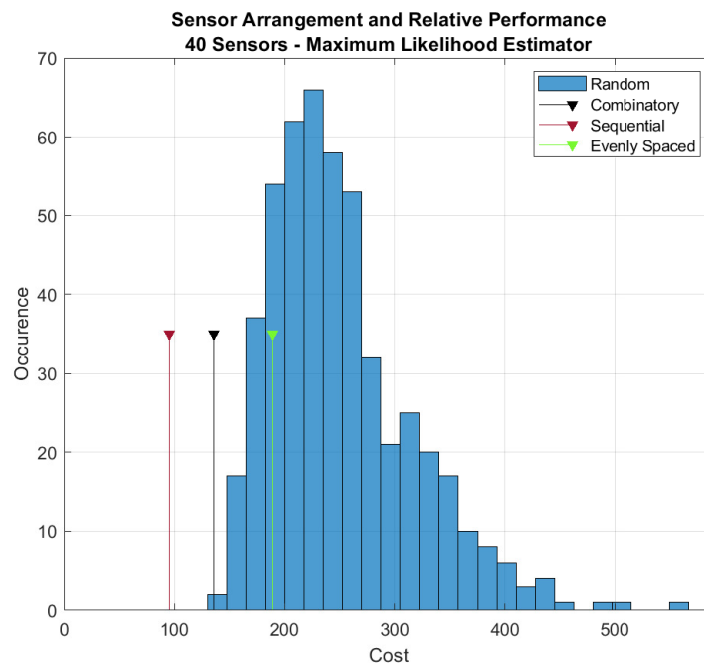


Figure 5.11. Histogram of costs per random sensor designs, versus optimized designs.

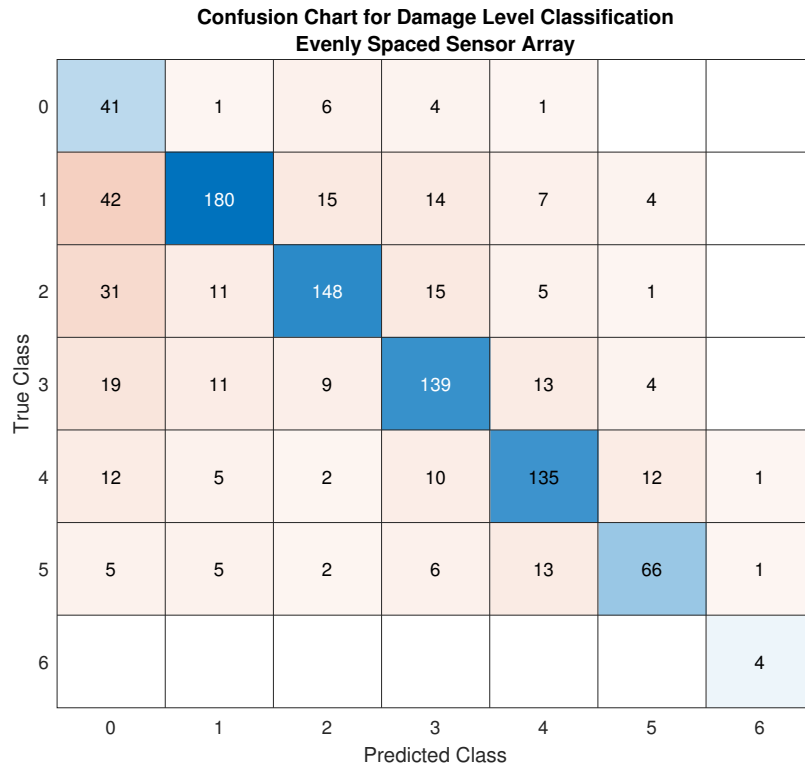


Figure 5.12. Confusion chart for the evenly spaced sensor array.

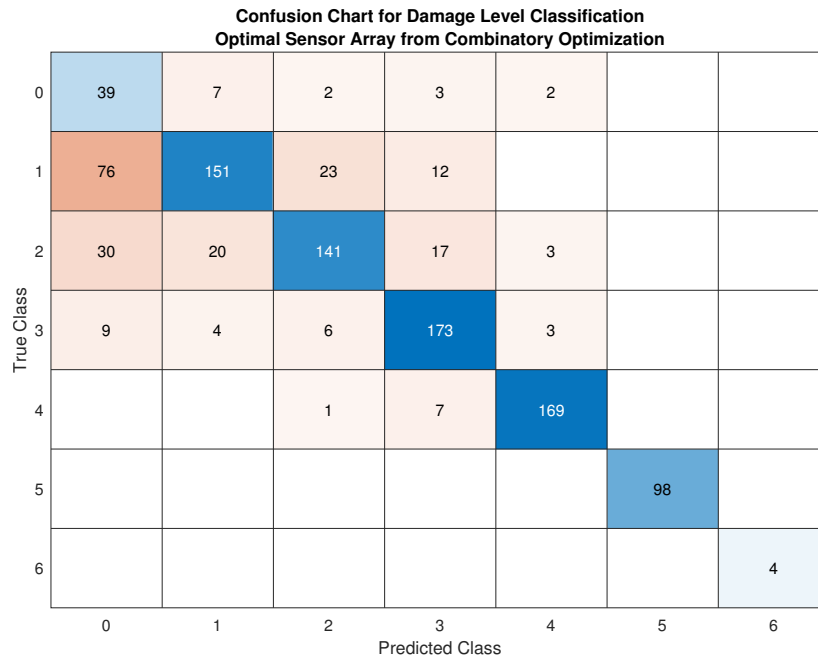


Figure 5.13. Confusion chart for the sensor array from the combinatory optimization.

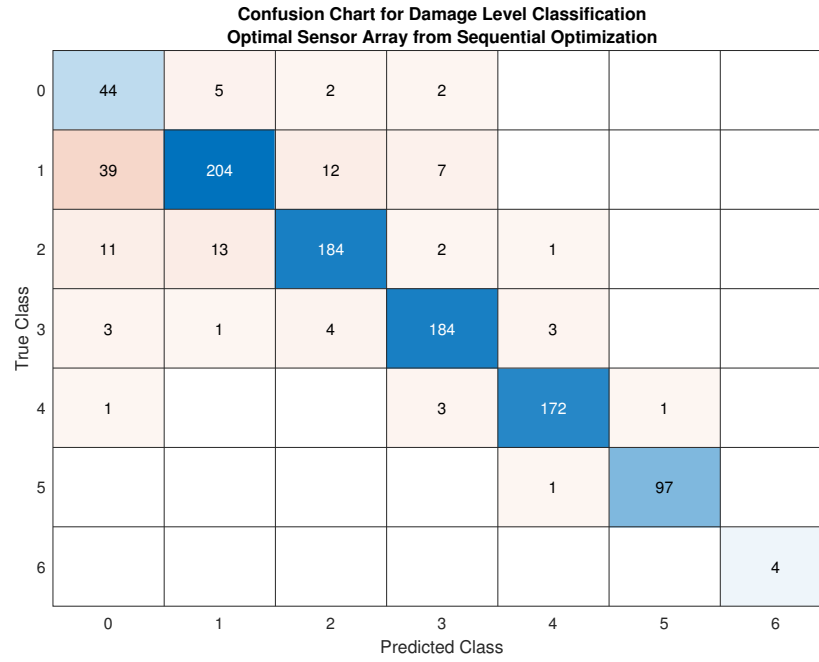


Figure 5.14. Confusion chart for the sensor array from the sequential optimization.

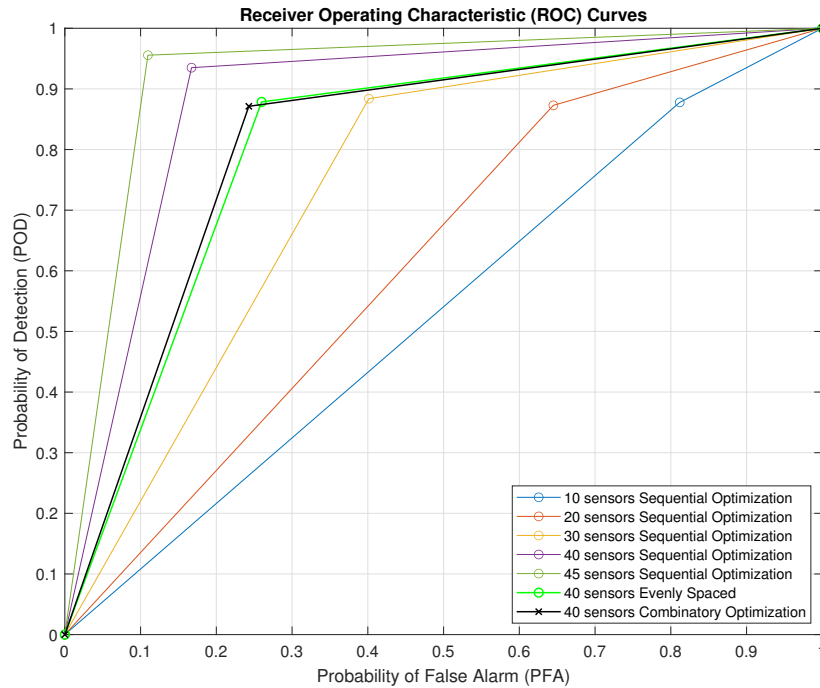


Figure 5.15. Receiver Operating Characteristic (ROC) curves for a variety of sensor sets, obtained at the ultimate load level.

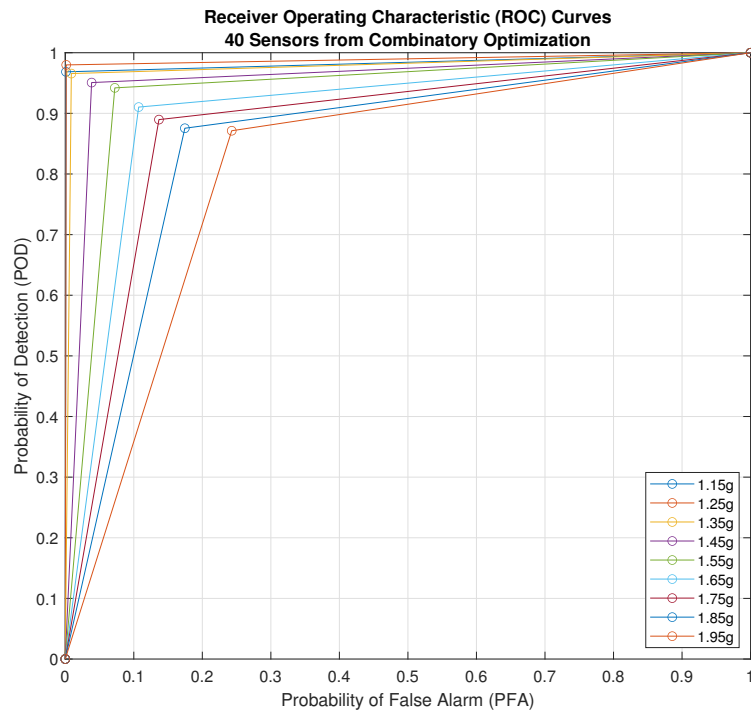


Figure 5.16. ROC curves for 40 combinatory sensors, for different load levels.

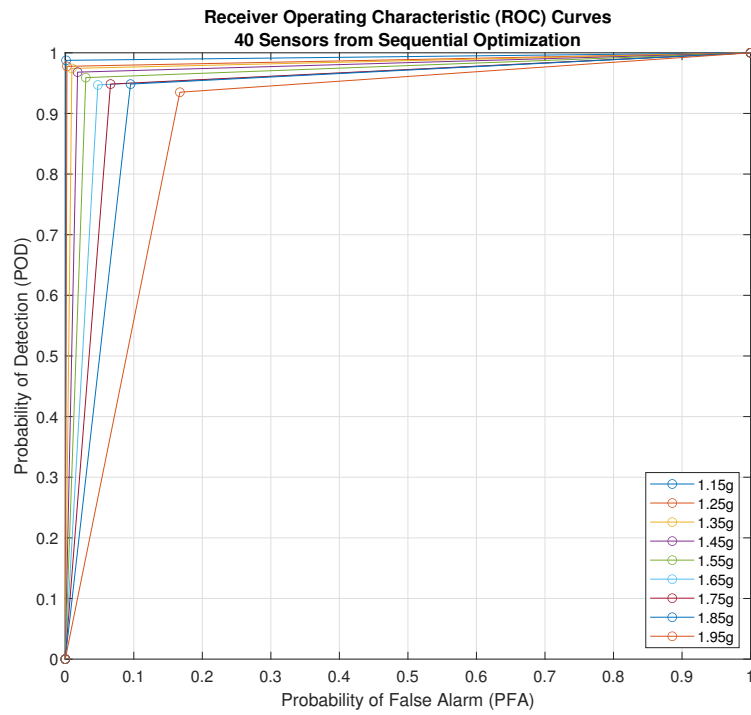


Figure 5.17. ROC curves for 40 sensors obtained sequentially, for different load levels.

Table 5.2. Final layout of each optimization run. Sensor location displayed in [mm].

Sensor	Seq. 1	Comb. 1	Seq. 2	Comb. 2	Seq. 3	Comb. 3
1	554	48	701	86	571	30
2	4	58	161	231	4	70
3	11	427	4	335	1301	78
4	681	526	11	519	1104	140
5	1424	586	418	750	1493	146
6	1321	599	536	808	126	166
7	451	611	2947	906	250	241
8	2278	617	1188	946	2214	257
9	1200	837	260	974	89	310
10	95	988	2160	1110	220	344
11	4977	1013	993	1210	2415	411
12	317	1117	1353	1242	561	468
13	4996	1234	1809	1298	367	517
14	232	1403	2360	1346	325	631
15	2081	1468	2283	1434	1269	637
16	1744	1484	2824	1498	1023	764
17	2461	1490	744	1615	2169	777
18	1207	1502	872	1736	1700	978
19	4989	1643	879	1797	407	1008
20	23	1665	2236	1844	704	1032
21	783	1771	1402	1952	932	1091
22	88	1822	1268	2078	1419	1109
23	30	1977	1866	2121	2557	1161
24	71	2087	2058	2169	1140	1224
25	1036	2138	2557	2273	734	1235
26	44	2158	2548	2287	1863	1355
27	1949	2181	1734	2317	2377	1419
28	2172	2351	2612	2334	2054	1518
29	37	2400	324	2353	466	1539
30	51	3017	142	2501	1563	1700
31	126	3148	2208	2598	849	1709
32	4970	3234	2378	2649	493	1832
33	610	3347	1968	2661	152	1906
34	58	3599	984	3022	2092	1987
35	78	3687	1208	3179	827	2095
36	151	3711	2033	3619	1643	2120
37	379	4091	388	3663	2509	2180
38	4963	4179	2758	3755	291	2268
39	1525	4274	1571	3809	1949	2282
40	3334	4831	1013	3930	1380	2794

Table 5.3. Summary of results from optimization runs.

Sensor	Seq. Run 1	Comb. Run 1	Seq. Run 2	Comb. Run 2	Seq. Run 3	Comb. Run 3
Cost	216.75	431.86	180.58	255.98	95.59	135.94
POD	0.85	0.76	0.89	0.84	0.94	0.87
PFA	0.56	0.28	0.20	0.23	0.17	0.24

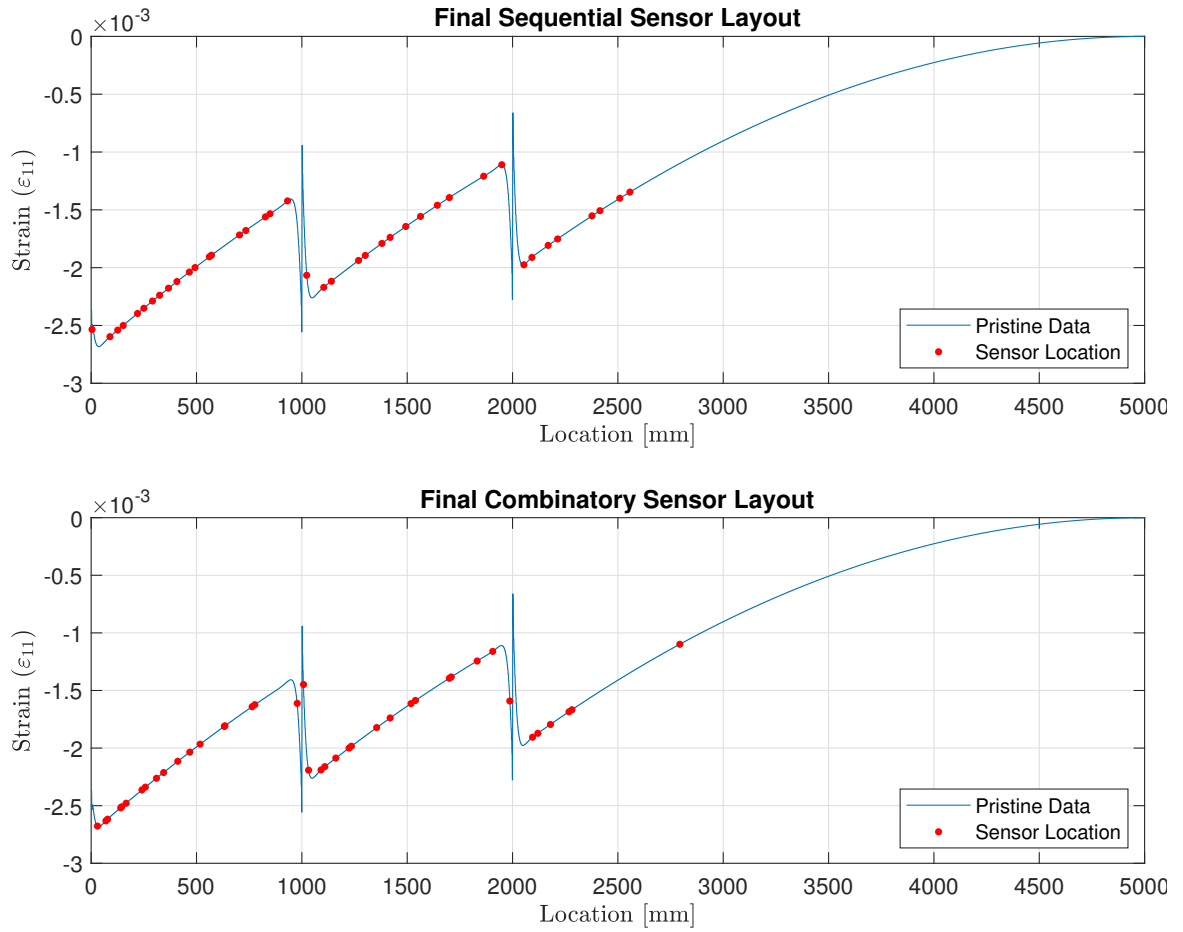


Figure 5.18. Final sensor layout after third run.

Chapters 5, in part, has been published in *Data Science in Engineering, Volume 10. SEM 2023. Conference Proceedings of the Society for Experimental Mechanics Series*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., & Tur, M. , 2023. The title of this paper is “Optimal Fiber-Optic Sensor Placement Framework for Structural Health Monitoring of an Aircraft’s Wing Spar”. The dissertation author was the primary investigator and author of this paper.

Chapters 5, in part, has been published in *Proceedings of the International Workshop on Structural Health Monitoring 2023*, Razzini, A. H., Todd, M. D., Kressel, I., Ofir, Y., Tur, M. , & Yehoshua, T., 2023. The title of this paper is “The Application of Risk Minimization to the Selection of Fiber Optic Sensors for an Aerospace Structural Monitoring Application”. The dissertation author was the primary investigator and author of this paper.

Chapter 6

Conclusions and future work

This dissertation proposed an optimal sensor placement framework that considers the minimization of the Bayes risk to determine the optimal system layout, consisting of the number and location of sensors to be deployed. The Bayes risk, or cost, was defined in terms of money, so that the stakeholders of any structure can obtain a practical understanding of the problem.

It was demonstrated that SHM system designs are highly dependent on the decision cost function. It was shown that if the cost function assumes that the penalty for not detecting a critical damage is 1,600 times higher than the penalty for having a false alarm, the system will take a very conservative approach to avoid false negatives "at all costs". Hence, it was shown that the primary goal of an SHM system cannot simply be the maximum gain of information to estimate the true damage state, but instead, it should consider the costs associated with a given decision and how heavily weighted it should be. In the context of this dissertation, a false negative that has estimated the buckling eigenvalue $\lambda_{true} = 0.98$ when the true value is $\lambda_{estimated} = 1.02$ is not nearly as critical as a $\lambda_{estimated} = 0.49$ that is misclassified as not critical. Therefore, the goal of SHM is not just estimation but also reducing the lifetime cost of a structure (in terms of money and/or safety) through estimation and decision-making. This dissertation provided a detailed framework on how to achieve that goal, demonstrated on a wing's spar entirely made of

composite materials.

Chapter 2 presented a high-fidelity finite element model of the composite I-beam that was used as a computational test bed for this dissertation. Since the information used to calibrate and validate the model was proprietary, an adapted version of the spar was built in-house for academic purposes. The entire modeling, construction, and testing procedure for this beam was presented. It was shown that regardless of many uncertainties sources that cause discrepancies between the model and experimental data, the overall strain behavior measured on the beam around the damage region match what was expected from the model and that the FE model is an essential piece to understanding the structure's behavior under different damage states and loading conditions, especially if field data is limited or not available.

Executing the finite element model several times comes with a high computational cost. Thus, in Chapter 3 showed how surrogate models can be useful to generate enough samples from the damage distributions in a time-efficient manner. The problem was presented in detail, and steps taken to guarantee that the lowest order model possible was used were discussed. The downside, however, is that the surrogates still need enough data to be trained and tested, so Python scripts were written to execute this large computational campaign.

The need for fast surrogate models became evident in Chapter 4 as the Bayesian inference model was derived, and the surrogate models were used to numerically estimate the likelihood and posterior distributions. It was shown that Bayesian inference provides a powerful framework for updating the understanding of structural damage as new data are collected, helping engineers and researchers make informed decisions based on probabilistic assessments of the structural health. Observability issues in regions of low signal-to-noise ratios were also discussed.

Finally, Chapter 5 discussed the last steps in the OSP framework, formally the Bayes risk as the costs associated with making a certain decision plus the cost of the SHM

system itself. It was shown that Bayesian optimization is a suitable algorithm to perform the optimization, as it makes a probabilistic decision of the problem to decide where to take the next sample from, instead of searching the entire sample space for the optimal solution. Three different risks were evaluated and the results were discussed, emphasizing the importance of rigorously defining and testing a cost function that reflects the goal of the SHM system being implemented. Having a computational test bed to do so allows for trial and error, and to learn more about the system's behavior before the sensors are permanently placed on the structure.

The significant contributions of this dissertation are summarized as follows:

1. The use of a multi-loading buckling approach to analyze the occurrence of local buckling in a composite wing's spar subject to debonding damage between the laminates.
2. Development of an statistical framework that links strain measurements to damage parameters, buckling eigenvalues, and damage criticality.
3. Development of a high-fidelity finite element model of an all-composite wing's spar to serve as a reliable data emulator of the physical problem.
4. In-house construction and testing of a composite spar to validate the strain behavior obtained via FE model.
5. Development of two machine learning-based surrogate models with high prediction accuracy to overcome the high computational cost of the FE model.
6. Development and demonstration of an optimal sensor placement framework that uses Bayesian optimization as the algorithm of choice.
7. Analysis of different cost functions for the optimal sensor placement algorithm and their impact on the overall performance of the SHM system.

6.1 Future work

There are some natural directions to continue the work developed in this dissertation. The first one is accounting for sensor reliability and including sensor failure in the optimization process. During the experimental setup to test the composite beam, 15 strain gauge sensors were used, and they were inherently associated with an amount of cables and heavy signal conditioner system that make their use challenging for real aerospace applications. The FBG sensors proposed in this dissertation's framework seem much more suitable for this application. However, if sensor failure is considered, one must evaluate the cost of losing a single strain gauge that malfunctions, versus losing measurements from up to 10 FBG sensors embedded on a single fiber that may break.

The second direction for the work is incorporating a damage growth model to perform a prognosis step and infer the remaining useful life of the structure before it needs to be repaired or replaced. The Bayesian inference can be used for updating the posterior probability of damage distributions when more data becomes available, so the missing link to perform the prognosis step is defining a damage growth model for debonding damage.

Finally, while the presented framework supports the optimization of any set of design parameters, this dissertation work focused primarily on the optimization of optimal FBG sensors placement. Consideration of other components of the design problem, such as sensor types and/or actuators, interrogation scheduling, temperature variation, etc., could enhance the performance of SHM systems.

Bibliography

- [1] Victor Giurgiutiu. *Structural health monitoring of aerospace composites*. Academic Press, 2015. doi:10.1016/B978-0-12-409605-9.00001-5.
- [2] Don Sears. Today's aircrafts wings. URL: <https://www.mscdirect.com/betterMRO/sites/default/files/Wing%20structure%20.jpg>.
- [3] Hyonny Kim and Keith T. Kedward. A method for modeling the local and global buckling of delaminated composite plates. *Composite Structures*, 44(1):43–53, 1999. URL: <https://www.sciencedirect.com/science/article/pii/S0263822398001172>, doi:10.1016/S0263-8223(98)00117-2.
- [4] Michael J Yeager. *Health Monitoring of Composites with Embedded Fiber Bragg Gratings*. University of California, San Diego, 2017.
- [5] Adam Quilter. Composites in aerospace applications. *IHS White Paper*, 444(1):264, 2001.
- [6] Isaac M Daniel, Ori Ishai, Issac M Daniel, and Ishai Daniel. *Engineering mechanics of composite materials*, volume 1994. Oxford university press New York, 2006.
- [7] Pankar K Mallick. *Fiber-reinforced composites: materials, manufacturing, and design*. CRC press, 2007.
- [8] Muhammad Aamir, Majid Tolouei-Rad, Khaled Giasin, and Ataollah Nosrati. Recent advances in drilling of carbon fiber-reinforced polymers for aerospace applications: A review. *The International Journal of Advanced Manufacturing Technology*, 105:2289–2308, 2019.
- [9] Justin Hale. 787 from the ground up. *Aero Magazine*, 4, 09 2006. URL: http://www.lb.boeing.com/commercial/aeromagazine/articles/qtr_4_06/AERO_Q406_article4.pdf.
- [10] A350 - less operating costs, more capabilities. URL: <https://aircraft.airbus.com/en/aircraft/a350-clean-sheet-clean-start/a350-less-operating-cost-more-capabilities>.
- [11] Natalie Zimmermann and Peng Hao Wang. A review of failure modes and fracture analysis of aircraft composite materials. *Engineering Failure Analysis*, 115:104692, 2020. doi:10.1016/j.engfailanal.2020.104692.

- [12] Umberto Polimeno and Michele Meo. Detecting barely visible impact damage detection on aircraft composites structures. *Composite Structures*, 91(4):398–402, 2009. Sixth International Conference on Composite Science and Technology. doi:10.1016/j.compstruct.2009.04.014.
- [13] Xiaochuan Sun and Stephen R. Hallett. Barely visible impact damage in scaled composite laminates: Experiments and numerical simulations. *International Journal of Impact Engineering*, 109:178–195, 2017. doi:10.1016/j.ijimpeng.2017.06.008.
- [14] Chen Ciang Chia, Hyo-Mi Jeong, Jung-Ryul Lee, and Gyuhae Park. Composite aircraft debonding visualization by laser ultrasonic scanning excitation and integrated piezoelectric sensing. *Structural Control and Health Monitoring*, 19(7):605–620, 2012.
- [15] Roberto Brighenti, Andrea Carpinteri, and Daniela Scorza. Mechanics of interface debonding in fibre-reinforced materials. *Journal of Composite Materials*, 50(19):2699–2718, 2016. doi:10.1177/0021998315612537.
- [16] Kaleeswaran Balasubramaniam, Shirsendu Sikdar, Rohan Soman, and Paweł Malinowski. Multi step structural health monitoring approaches in debonding assessment in a sandwich honeycomb composite structure using ultrasonic guided waves. *Measurement*, 194:111057, 2022. doi:10.1016/j.measurement.2022.111057.
- [17] North Atlantic Treaty Organization (NATO). Aep-4671 unmanned aircraft systems airworthiness requirements (usar). pages 70–79, 2019.
- [18] Hessamodin Teimouri, Abbas S Milani, Jason Loeppky, and Rudolf Seethaler. A gaussian process-based approach to cope with uncertainty in structural health monitoring. *Structural Health Monitoring*, 16(2):174–184, 2017.
- [19] Manuel A Vega and Michael D Todd. A variational bayesian neural network for structural health monitoring and cost-informed decision-making in miter gates. *Structural Health Monitoring*, 21(1):4–18, 2022.
- [20] W Grodzki and Andrzej Łukaszewicz. Design and manufacture of unmanned aerial vehicles (uav) wing structure using composite materials: Planung und bau einer flügelstruktur für unbemannte luftfahrzeuge (uav) unter verwendung von kompositwerkstoffen. *Materialwissenschaft und Werkstofftechnik*, 46(3):269–278, 2015.
- [21] Charles R Farrar and Keith Worden. An introduction to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365(1851):303–315, 2007.
- [22] Charles R Farrar and Keith Worden. *Structural health monitoring: a machine learning perspective*. John Wiley & Sons, 2012.
- [23] Peter J Shull. *Nondestructive evaluation: theory, techniques, and applications*. CRC press, 2002.

- [24] Carosena Meola, Simone Boccardi, and Giovanni Maria Carlomagno. *Infrared thermography in the evaluation of aerospace composite materials: infrared thermography to composites*. Woodhead Publishing, 2016.
- [25] Arun Manohar and Francesco Lanza di Scalea. Detection of defects in wind turbine composite blades using statistically enhanced lock-in thermography. *Structural Health Monitoring*, 12(5-6):566–574, 2013.
- [26] Clifford Bueno, Marion D Barker, Robert A Betz, Richard C Barry, and Robert Buchanan. Nondestructive evaluation of aircraft structures using high-resolution real-time radiography. In *Nondestructive Evaluation of Aging Aircraft, Airports, Aerospace Hardware, and Materials*, volume 2455, pages 114–124. SPIE, 1995.
- [27] Liang Cheng, Bin Gao, Gui Yun Tian, Wai Lok Woo, and Gerard Berthiau. Impact damage detection and identification using eddy current pulsed thermography through integration of pca and ica. *IEEE Sensors Journal*, 14(5):1655–1663, 2014.
- [28] Yi-Qing Ni and KY Wong. Integrating bridge structural health monitoring and condition-based maintenance management. In *4th International workshop on civil structural health monitoring*, pages 6–8, 2012.
- [29] Wieslaw Ostachowicz, Rohan Soman, and Pawel Malinowski. Optimization of sensor placement for structural health monitoring: a review. *Structural Health Monitoring*, 18(3):963–988, 2019. doi:10.1177/1475921719825601.
- [30] Yi Tan and Limao Zhang. Computational methodologies for optimal sensor placement in structural health monitoring: A review. *Structural Health Monitoring*, 19(4):1287–1308, 2020. doi:10.1177/1475921719877579.
- [31] Costas Argyris, Sharmistha Chowdhury, Volkmar Zabel, and Costas Papadimitriou. Bayesian optimal sensor placement for crack identification in structures using strain measurements. *Structural Control and Health Monitoring*, 25(5):e2137, 2018.
- [32] Eric B Flynn and Michael D Todd. A bayesian approach to optimal sensor placement for structural health monitoring with application to active sensing. *Mechanical Systems and Signal Processing*, 24(4):891–903, 2010.
- [33] Xiaowei Ye, Ye Q Ni, Kaiyuen Wong, and Jinming Ko. Statistical analysis of stress spectra for fatigue life assessment of steel bridges with structural health monitoring data. *Engineering Structures*, 45:166–176, 2012.
- [34] Johannes F. C. Markmiller and Fu-Kuo Chang. Sensor network optimization for a passive sensing impact detection technique. *Structural Health Monitoring*, 9(1):25–39, 2010. doi:10.1177/1475921709349673.
- [35] Cynthia Swann and Aditi Chattopadhyay. Optimization of piezoelectric sensor location for delamination detection in composite laminates. *Engineering Optimization*, 38(5):511–528, 2006. doi:10.1080/03052150600557841.

- [36] Robert F Guratzsch and Sankaran Mahadevan. Structural health monitoring sensor placement optimization under uncertainty. *AIAA journal*, 48(7):1281–1289, 2010.
- [37] Theodoros Loutas and Alexandros Bourikas. Strain sensors optimal placement for vibration-based structural health monitoring. the effect of damage on the initially optimal configuration. *Journal of Sound and Vibration*, 410:217–230, 2017.
- [38] Carl Malings and Matteo Pozzi. Value of information for spatially distributed systems: Application to sensor placement. *Reliability Engineering & System Safety*, 154:219–233, 2016.
- [39] Sergio Cantero-Chinchilla, Juan Chiachío, Manuel Chiachío, Dimitrios Chronopoulos, and Arthur Jones. Optimal sensor configuration for ultrasonic guided-wave inspection based on value of information. *Mechanical Systems and Signal Processing*, 135:106377, 2020.
- [40] Richard Abbott. Analysis and design of composite and metallic flight vehicle structures, 2016.
- [41] Local buckling of stitched composite laminate. *Composites Part B: Engineering*, 30(8):833–840, 1999. doi:10.1016/S1359-8368(99)00047-5.
- [42] Alan D Kersey, Michael A Davis, Heather J Patrick, Michel LeBlanc, Kee P Koo, Charles G Askins, Martin A Putnam, and E Joseph Friebele. Fiber grating sensors. *Journal of lightwave technology*, 15(8):1442–1463, 1997.
- [43] Rui De Oliveira, CA Ramos, and António Torres Marques. Health monitoring of composite structures by embedded fbg and interferometric fabry–pérot sensors. *Computers & structures*, 86(3-5):340–346, 2008.
- [44] Damien Kinet, Patrice Mégret, Keith W Goossen, Liang Qiu, Dirk Heider, and Christophe Caucheteur. Fiber bragg grating sensors toward structural health monitoring in composite materials: Challenges and solutions. *Sensors*, 14(4):7394–7419, 2014.
- [45] I Kressel, B Dorfman, Y Botsev, A Handelman, J Balter, ACR Pillai, MH Prasad, N Gupta, AM Joseph, and R Sundaram. Flight validation of an embedded structural health monitoring system for an unmanned aerial vehicle. *Smart Materials and Structures*, 24(7):075022, 2015.
- [46] S Takeda, Y Aoki, T Ishikawa, N Takeda, and H Kikukawa. Structural health monitoring of composite wing structure during durability test. *Composite structures*, 79(1):133–139, 2007.
- [47] Kenneth O Hill, Y Fujii, Derwyn C Johnson, and Brian S Kawasaki. Photosensitivity in optical fiber waveguides: Application to reflection filter fabrication. *Applied physics letters*, 32(10):647–649, 1978.

- [48] Michael J Asher, Barry FW Croke, Anthony J Jakeman, and Luk JM Peeters. A review of surrogate models and their application to groundwater modeling. *Water Resources Research*, 51(8):5957–5973, 2015.
- [49] Reza Alizadeh, Janet K Allen, and Farrokh Mistree. Managing computational complexity using surrogate models: a critical review. *Research in Engineering Design*, 31:275–298, 2020.
- [50] Adrielly H. Razzini, Michael D. Todd, Iddo Kressel, Yoav Ofir, Moshe Tur, and Tal Yehoshua. Damage assessment of an aircraft’s wing spar using gaussian process regressors. In Piervincenzo Rizzo and Alberto Milazzo, editors, *European Workshop on Structural Health Monitoring*. Springer International Publishing, 2023.
- [51] Adrielly H. Razzini, Iddo Kressel, Yoav Ofir, Moshe Tur, Tal Yehoshua, and Michael D. Todd. Development of a surrogate model for structural health monitoring of a uav wing spar. In Ramin Madarshahian and Francois Hemez, editors, *Data Science in Engineering, Volume 9*, pages 99–102, Cham, 2022. Springer International Publishing.
- [52] Daniel C Kammer and Michael L Tinker. Optimal placement of triaxial accelerometers for modal vibration tests. *Mechanical systems and signal processing*, 18(1):29–41, 2004.
- [53] KH Tong, Norhisham Bakhary, ABH Kueh, and AY Yassin. Optimal sensor placement for mode shapes using improved simulated annealing. *Smart Struct. Syst*, 13(3):389–406, 2014.
- [54] Jie-Sheng Wang, Shu-xia Li, and Jie Gao. Som neural network fault diagnosis method of polymerization kettle equipment optimized by improved pso algorithm. *The Scientific World Journal*, 2014, 2012.
- [55] Yichao Yang, Mayank Chadha, Zhen Hu, Manuel A. Vega, Matthew D. Parno, and Michael D. Todd. A probabilistic optimal sensor design approach for structural health monitoring using risk-weighted f-divergence. *Mechanical Systems and Signal Processing*, 161:107920, 2021. doi:10.1016/j.ymssp.2021.107920.
- [56] Adrielly Hokama Razzini, Michael D Todd, Iddo Kressel, Yoav Ofir, and Moshe Tur. The application of risk minimization to the selection of fiber optic sensors for an aerospace structural monitoring application. *Proceedings of the International Workshop on Structural Health Monitoring 2023*, 2023.
- [57] Adrielly Hokama Razzini, Michael D. Todd, Iddo Kressel, Yoav Ofir, and Moshe Tur. Optimal fiber-optic sensor placement framework for structural health monitoring of an aircraft’s wing spar. In Ramin Madarshahian and François Hemez, editors, *Data Science in Engineering, Volume 10*, pages 129–132, Cham, 2023. Springer Nature Switzerland.

- [58] Abaqus. Abaqus 6.14 analysis user’s guide. URL: <http://130.149.89.49:2080/v2016/books/usb/default.htm>.
- [59] Carlos A Mota Soares, Cristóvão M Mota Soares, and Manuel JM Freitas. *Mechanics of composite materials and structures*, volume 361. Springer Science & Business Media, 2013.
- [60] Omega Engineering. Practical strain gage measurements, 1999. URL: https://www.omega.co.uk/techref/pdf/StrainGage_Measurement.pdf.
- [61] Vishay Precision Group. Strain gages and instruments tech note tn-504-1: Strain gage thermal output and gage factor variation with temperature. URL: https://intertechnology.com/Vishay/pdfs/TechNotes_TechTips/TN-504.pdf.
- [62] Wei-Liem Loh. On Latin hypercube sampling. *The Annals of Statistics*, 24(5):2058 – 2080, 1996. doi:10.1214/aos/1069362310.
- [63] Théo Benoit-Cattin, Delia Velasco-Montero, and Jorge Fernández-Berni. Impact of thermal throttling on long-term visual inference in a cpu-based edge device. *Electronics*, 9(12), 2020. doi:10.3390/electronics9122106.
- [64] Alex Thompson, W.G.A. Brown, Paul Stoddart, and Scott Wade. Bend effects on fibre bragg gratings in standard and low bend loss optical fibres. pages 1 – 4, 01 2011. doi:10.1109/ACOFIT.2010.5929958.
- [65] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 1. Springer, 2006.
- [66] Dragi Kocev, Sašo Džeroski, Matt D White, Graeme R Newell, and Peter Griffioen. Using single-and multi-target regression trees and ensembles to model a compound index of vegetation condition. *Ecological Modelling*, 220(8):1159–1168, 2009.
- [67] Devis Tuia, Jochem Verrelst, Luis Alonso, Fernando Pérez-Cruz, and Gustavo Camps-Valls. Multioutput support vector regression for remote sensing biophysical parameter estimation. *IEEE Geoscience and Remote Sensing Letters*, 8(4):804–808, 2011.
- [68] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.
- [69] Ian Dewancker, Michael McCourt, and Scott Clark. Bayesian optimization for machine learning: A practical guidebook. *arXiv preprint arXiv:1612.04858*, 2016.
- [70] Giovanni Capellari, Eleni Chatzi, and Stefano Mariani. Structural health monitoring sensor network optimization through bayesian experimental design. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering*, 4(2):04018016, 2018.

- [71] Daniel C Kammer. Optimal sensor placement for modal identification using system-realization methods. *Journal of Guidance, Control, and Dynamics*, 19(3):729–731, 1996.