

UC San Diego

UC San Diego Electronic Theses and Dissertations

Title

Physics-Constrained, Structure-Preserving Machine Learning Models for Structural Health Applications

Permalink

<https://escholarship.org/uc/item/2n15q9w2>

Author

Najera-Flores, David Aaron

Publication Date

2024

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA SAN DIEGO

Physics-Constrained, Structure-Preserving Machine Learning Models for Structural Health
Applications

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

in

Structural Engineering

by

David A. Najera-Flores

Committee in charge:

Professor Michael D. Todd, Chair
Professor Jiun-Shyan Chen
Professor Charles Farrar
Professor Boris Kramer

2024

Copyright

David A. Najera-Flores, 2024

All rights reserved.

The Dissertation of David A. Najera-Flores 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 wife, Daniela, for inspiring me everyday through your actions and words.

EPIGRAPH

The development of mathematics
is largely a history of deepening understanding of the concepts of infinity.

Emmy Noether

There is really nothing more to say-except why.
But since why is difficult to handle, one must take refuge in how.

Toni Morrison

I may remark that the curious transformations many formulae can undergo, the unsuspected and to a beginner apparently impossible identity of forms exceedingly dissimilar at first sight, is I think one of the chief difficulties in the early part of mathematical studies. I am often reminded of certain sprites and fairies one reads of, who are at one's elbows in one shape now, and the next minute in a form most dissimilar.

Ada Lovelace

TABLE OF CONTENTS

Dissertation Approval Page	iii
Dedication	iv
Epigraph	v
Table of Contents	vi
List of Figures	x
List of Tables	xxi
Acknowledgements	xxii
Vita	xxv
Abstract of the Dissertation	xxviii
Chapter 1 Introduction	1
1.1 Machine learning applications to physics	2
1.1.1 Purely data-driven approaches	2
1.1.2 Physics-informed approaches	3
1.1.3 Physics-constrained approaches	4
1.2 Research Objectives, Organization of Dissertation, and Contributions	4
Chapter 2 Physics-Constrained Neural Differential Operators for Health Prognosis ..	9
2.1 Applications to Lithium-ion battery remaining useful life prediction	9
2.1.1 Predictive Approaches for Battery RUL Forecast	13
2.1.2 Physics-Constrained Bayesian Neural Network	22
2.1.3 Demonstration of Methodology	29
2.1.4 Performance Metrics	29
2.1.5 Results	31
2.1.6 Comparison and discussions	33
2.1.7 Conclusions	40
2.1.8 Preview of Section 2.2	41
2.1.9 Remarks	41
2.2 Applications to corrosion morphology prediction in civil structures	41
2.2.1 Pitting Corrosion-Induced Failure of Large Civil Infrastructure	44
2.2.2 Phase-Field Pitting Corrosion Simulation	46
2.2.3 Surrogate Modeling for Physics-Based Pitting Corrosion Forecasting ..	47
2.2.4 Physics-Constrained Machine Learning as A Surrogate Model for Corro-	
sion Failure Forecasting	49
2.2.5 Numerical Results	54

2.2.6	Conclusions	60
2.2.7	Preview of Chapter 3	61
2.2.8	Remarks	61
Chapter 3	Structure-Preserving Neural Differential Operators for Structural Dynamics	70
3.1	Background	72
3.1.1	Hamiltonian Mechanics	72
3.1.2	Physics-Constrained Autoencoder	74
3.1.3	Structure-Preserving Neural Network Ensemble	76
3.1.4	Trained Model Deployment.....	80
3.1.5	Domain Shift Identification	80
3.2	Numerical Results and Discussion.....	81
3.2.1	Euler-Bernoulli Beam	81
3.2.2	Two Degree of Freedom System	83
3.2.3	Rigid Link with Suspended Mass	85
3.2.4	System Identification of a Duffing Oscillator	89
3.3	Conclusion	90
3.4	Preview of Chapter 4	91
3.5	Remarks	92
Chapter 4	State-Space Reconstruction from Partial Observables Using an Invertible Neural Network with Structure-Preserving Properties for Nonlinear Struc- tural Dynamics	97
4.1	Introduction	97
4.2	Background	99
4.2.1	Structure-Preserving Neural Network	99
4.2.2	State space reconstruction with time delay embedding	100
4.3	Coordinate transformation	102
4.3.1	Normalizing flow autoencoder	103
4.3.2	General MLP autoencoder.....	104
4.4	Neural network training and deployment	106
4.5	Numerical examples	110
4.5.1	Oscillator with isolated cubic stiffness nonlinearity	110
4.5.2	Two-dimensional membrane with isolated nonlinearities	119
4.5.3	Brake-Reuß Beam	124
4.5.4	S4 Beam	127
4.6	Practical Challenges	129
4.7	Conclusion	130
4.8	Preview of Chapter 5	131
4.9	Remarks	132
Chapter 5	Nonlinear System Identification in Structures with Isolated Deviatoric Be- havior	142
5.0.1	Isolated Nonlinearities	145

5.0.2	Structure-Preserving Machine Learning	149
5.1	Numerical Simulation	155
5.1.1	Example	155
5.1.2	Simulation	158
5.2	Conclusions	168
5.3	Preview of Chapter 6	169
5.4	Remarks	170
Chapter 6	Nonintrusive Structure-Preserving Nonlinear Model Order Reduction	171
6.1	Background	173
6.1.1	Lagrangian mechanics	174
6.1.2	Structure-preserving intrusive model reduction for mechanical systems ..	175
6.2	ML-enhanced structure-preserving operator inference method	177
6.2.1	Data projection via proper orthogonal decomposition	177
6.2.2	Problem formulation	178
6.2.3	Step 1: Lagrangian operator inference method for linear reduced operators	180
6.2.4	Step 2: Structure-preserving learning for nonlinear reduced operators ..	182
6.2.5	Training algorithm	184
6.3	Numerical results	185
6.3.1	Conservative rod model with spatially isolated elastic nonlinearities	185
6.3.2	Two-dimensional nonlinear membrane	192
6.3.3	Half Brake-Reuß beam	197
6.4	Conclusions	201
6.5	Preview of Chapter 7	202
6.6	Remarks	203
Chapter 7	Uncertainty Quantification of a Machine Learning Model for Identification of Isolated Nonlinearities with Conformal Prediction	204
7.1	Background and Methodology	205
7.1.1	Isolated Nonlinearity Formulation	205
7.1.2	Structure-Preserving Machine Learning	207
7.1.3	Statistical Inference on Deviatoric Forces	211
7.2	Results	213
7.2.1	Example description	213
7.2.2	Noise model description	215
7.2.3	Effect of noise on model performance	216
7.2.4	Prediction intervals and damage detection	217
7.3	Conclusions	228
7.4	Preview of Chapter 8	232
7.5	Remarks	232
Chapter 8	Uncertainty-Aware Structure-Preserving Digital Twin for Domain Shift Detection from Nonlinear Dynamic Response of Structural Systems	234
8.1	Background and Methodology	236

8.1.1	Isolated Nonlinearity Formulation	236
8.1.2	Structure-Preserving Machine Learning	238
8.1.3	Spectral-Normalized Neural Gaussian Process	241
8.1.4	Hypothesis Testing for Domain Shift	244
8.2	Results	247
8.2.1	Example description	247
8.2.2	Definition of domain shift	249
8.2.3	Effect of domain shift on baseline response	250
8.2.4	Domain shift detection	254
8.3	Conclusions	261
8.4	Preview of Chapter 9	263
8.5	Remarks	263
Chapter 9	A Probabilistic Reasoner Based on Bayes Risk for Damage Detection in Structural Systems	265
9.1	Background and Motivation	265
9.2	Uncertainty-aware ML model	266
9.3	Damage detector definition	268
9.4	Demonstration of damage detector performance	270
9.4.1	Problem description	270
9.4.2	Optimizing for probability of detection	272
9.4.3	Considering cost of wrong decisions	273
9.4.4	Detector performance when considering multiple damage types	279
9.5	Model updating	280
9.6	Conclusion	284
Chapter 10	Conclusions and Future Research	286
Appendix A	Chapter 2 Appendix	289
A.1	Hamiltonian network: autoencoder hyperparameters	289
Appendix B	Chapter 3 Appendix	291
B.1	Invertible neural network: hyperparameters	291
B.2	Comparison of MLP and INN autoencoders	291
Bibliography		294

LIST OF FIGURES

Figure 1.1.	Proposed framework for SHM system.	6
Figure 2.1.	Illustration of FFM used to generate posterior predictive plot	16
Figure 2.2.	Illustration of similarity-based RUL estimation method. Note that the data and curve shown in this figure are for explanation purpose only. They are not real capacity data and curve.	21
Figure 2.3.	Schematic of general neural network architecture.....	26
Figure 2.4.	Capacity data used in this example (Before and after removing outliers)...	29
Figure 2.5.	RUL predictions of test 0 and 18 for all the approaches considered.	34
Figure 2.6.	RUL predictions of test 62 and 108 for all the approaches considered.	35
Figure 2.7.	Histogram of error distributions over the test set for the different techniques considered.	36
Figure 2.8.	RUL curve predictions for four representative cells.	38
Figure 2.9.	Random discharge capacity curve samples generated by the BNNet (black) and actual experimental curve (dashed red).	39
Figure 2.10.	Proportion of cycles for which the null hypothesis $H_0 : \hat{L}_\mu = L$ was accepted.	40
Figure 2.11.	A miter gate in the dewatered state.	45
Figure 2.12.	Pit shapes under different stress conditions.	47
Figure 2.13.	Architecture of CNN-GP-NARX surrogate model.	48
Figure 2.14.	Schematic of the proposed general neural network architecture.	50
Figure 2.15.	Simulation results of two different corrosion cases	55
Figure 2.16.	Example of corrosion pattern evolution with pixel-by-pixel constraint (bottom) and total corrosion constraint (top). Simulation time step is indicated in figure.	57
Figure 2.17.	Comparison of MSE as a function of time for different methods.	58

Figure 2.18.	Results for test case 251. Shown: simulation reference (first), CNN-GP-NARX mean prediction (second), CNN-GP-NARX 95% confidence interval (third), BLSTEN-VAE mean prediction (fourth), and BLSTEN-VAE 95% confidence interval. Simulation time step is indicated.	62
Figure 2.19.	Results for test case 246. Shown: simulation reference (first), CNN-GP-NARX mean prediction (second), CNN-GP-NARX 95% confidence interval (third), BLSTEN-VAE mean prediction (fourth), and BLSTEN-VAE 95% confidence interval. Simulation time step is indicated.	63
Figure 2.20.	Results for test case 248. Shown: simulation reference (first), CNN-GP-NARX mean prediction (second), CNN-GP-NARX 95% confidence interval (third), BLSTEN-VAE mean prediction (fourth), and BLSTEN-VAE 95% confidence interval. Simulation time step is indicated.	64
Figure 2.21.	Corner plot showing the distribution of training set parameters and selected test set cases.	65
Figure 2.22.	Evolution of latent features as a function of time.	66
Figure 2.23.	Derivatives of pitting corrosion with respect to the latent features. Simulation time step is indicated at the top.	67
Figure 2.24.	Derivatives of pitting corrosion with respect to the physical parameters for a tensile normal load. Simulation time step is indicated at the top.	68
Figure 2.25.	Derivatives of pitting corrosion with respect to the physical parameters for a compressive normal load. Simulation time step is indicated at the top. ..	68
Figure 2.26.	Effect of physical parameter variation from low (left) to high (right) on corrosion shape.	69
Figure 3.1.	Schematic of ensemble of neural networks used.	78
Figure 3.2.	Phase space (top) and time (bottom) response of Euler-Bernoulli beam problem. Analytical response shown in color and network prediction shown in black.	82
Figure 3.3.	Modal frequencies (left) computed from analytical model (blue) and learned model (green), and cross-orthogonal check of analytical and predicted mode shapes (right).	83
Figure 3.4.	Diagram of two degree-of-freedom system example.	83
Figure 3.5.	Summary of phase space response for the various cases considered. Analytical response shown in color and network prediction shown in black. ..	86

Figure 3.6.	Analytical and predicted energy terms for undamped linear case.	87
Figure 3.7.	Eigenvectors of analytical (dashed red) and predicted (solid black) $\mathbf{M}$ (left) and $\mathbf{K}$ (right).	87
Figure 3.8.	Eigenvalues (left) and eigenvectors (right) of $\mathbf{M}^{-1}\mathbf{K}$	88
Figure 3.9.	State matrices of linear damping model (top) and updated cubic damping model (bottom).	88
Figure 3.10.	Eigenvectors of state matrices $\mathbf{M}$ (left), $\mathbf{K}$ (middle), and $\mathbf{C}$ (right), computed from original network (dashed red) and updated network (solid blue). The hue of the eigenvectors indicates the variation in time.	89
Figure 3.11.	Rigid link with suspended mass problem.	89
Figure 3.12.	Time (top) and phase space (bottom) response of the rigid link problem. Analytical response shown in color and network prediction shown in black.	93
Figure 3.13.	Eigenvectors of analytical (dashed red) and predicted (solid black) $\mathbf{M}$ (left) and $\mathbf{K}$ (right).	94
Figure 3.14.	State space representation of original (left) and learned system (right). ...	94
Figure 3.15.	Comparison between identified and actual parameters in the duffing oscillator system energy terms.	95
Figure 3.16.	Nonlinear forced response of the actual system (solid colored) and the trained network (dashed black) at different excitation amplitudes.	96
Figure 4.1.	Schematic of complete workflow.	106
Figure 4.2.	Example system with isolated nonlinearities. The nonlinearities are localized within the region $s \in (s_1, s_2)$, while the remainder of the rod is linear.	111
Figure 4.3.	Snapshots in time of rod deformation colored by displacement for the first mode excitation case (A). Measured locations shown in green.	111
Figure 4.4.	Snapshots in time of rod deformation colored by displacement for the first two modes excitation case (B). Measured locations shown in green.	112
Figure 4.5.	Snapshots in time of rod deformation colored by displacement for the first three modes excitation case (C). Measured locations shown in green.	112

Figure 4.6.	Original signals (color) and time-delayed copies (dashed black) for the first mode excitation case (A).	113
Figure 4.7.	Original signals (color) and time-delayed copies (dashed black) for the first two modes excitation case (B).	113
Figure 4.8.	Original signals (color) and time-delayed copies (dashed black) for the first three modes excitation case (C).	114
Figure 4.9.	Reconstructed response from neural network prediction for the first mode excitation case (A). Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.	115
Figure 4.10.	Reconstructed response from neural network prediction for the first two modes excitation case (B). Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.	116
Figure 4.11.	Reconstructed response from neural network prediction for the first three modes excitation case (C). Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.	117
Figure 4.12.	Reconstructed response from neural network prediction for the first three modes excitation case (C) with alternative initial conditions. Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.	118
Figure 4.13.	t-SNE embedding of full dimensional (left) and latent (right) space. The points are colored by their time index so that time evolution is indicated by going from bright to dark colors.	120
Figure 4.14.	Fast Fourier Transform of physical displacements (blue) and frequencies computed from linearized learned latent space model (black).	121
Figure 4.15.	Modal displacements for original physical system (color) and learned latent space model (dashed).	122
Figure 4.16.	Membrane structure with visible grid and measurement points shown in red. Colored by deformation pattern of the first bending mode.	123
Figure 4.17.	Original signals (color) and time-delayed copies (dashed black) for the membrane model.	124
Figure 4.18.	Reconstructed response from neural network prediction for the test set (alternative initial conditions). Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.	125

Figure 4.19.	Spectral embedding of full dimensional (left) and latent (right) space in three dimensions. The points are colored by their time index so that time evolution is indicated by going from bright to dark colors.	126
Figure 4.20.	FFT of the actual physical displacements (solid line, no marker) at the measured locations and the predicted response (circle marker). The eigenvalues extracted from the model are shown with a vertical black line.	127
Figure 4.21.	STFT of the actual physical displacements (top) at the measured locations and the predicted response (bottom). The eigenvalues extracted from the model are shown with a dashed line.	133
Figure 4.22.	Modal displacements for original physical membrane system (color) and learned latent space (dashed).	134
Figure 4.23.	Snapshots in time of HBRB deformation colored by axial displacement. Measured locations shown in green.	134
Figure 4.24.	Time histories of HBRB experimental (blue) and predicted (black) displacement, velocity, and acceleration for the three measured locations. Vertical dashed line indicates training/test data boundary.	135
Figure 4.25.	Phase portraits of the HBRB experimental data (blue) and predicted (dashed black) for the three measurement locations.	135
Figure 4.26.	Hilbert transform of the HBRB response at the three measurement locations (blue, orange, and green) and the predicted response (dashed black), as well as the error (magenta). Vertical dashed line indicates training/test data boundary.	136
Figure 4.27.	Frequency (left) and damping (right) backbone curves extracted from time histories of HBRB response.	137
Figure 4.28.	Time histories of S4 beam experimental (blue) and predicted (black) acceleration for three accelerometer locations. Vertical dashed line indicates training/test data boundary.	138
Figure 4.29.	Close-up of time histories of S4 beam experimental (blue) and predicted (black) acceleration for three accelerometer locations. Vertical dashed line indicates training/test data boundary.	139
Figure 4.30.	Hilbert transform of the S4 beam response at three measurement locations (blue, orange, and green) and the predicted response (dashed black), as well as the error (magenta). Vertical dashed line indicates training/test data boundary.	140

Figure 4.31.	Modal displacements for S4 beam experimental response (color) and learned latent space (dashed). The bottom figures show a close-up of the time histories.	140
Figure 4.32.	First two NNMs of the S4 beam learned model in latent space.	141
Figure 5.1.	General Structure; the nonlinearities are localized within the region $\mathcal{C}_2$. ..	146
Figure 5.2.	The nonlinear isolated region ($\mathcal{C}_2$) can be decomposed from the linear external region ($\mathcal{C}_1$).	149
Figure 5.3.	Architecture of MLP augmented with polynomial terms.	152
Figure 5.4.	Data-driven simulation incorporating the MLP.	154
Figure 5.5.	Example system with isolated nonlinearities. The nonlinearities are localized within the region $s \in (s_1, s_2)$, while the remainder of the rod is linear.	155
Figure 5.6.	Hysteretic force model. Note that this is in addition to the elastic and dissipative coupling described in Eq. (5.25).	157
Figure 5.7.	System response; direct numerical simulation	161
Figure 5.8.	Cubic nonlinearities and hysteretic damping	162
Figure 5.9.	Instantaneous amplitude of the Hilbert Transform; $\ \mathcal{H}[\delta Q](t)\ $: Blue, $\ \mathcal{H}[\text{Err}_{\delta Q}](t)\ $: Red.	163
Figure 5.10.	Cubic nonlinearities and hysteretic damping, alternative initial conditions .	164
Figure 5.11.	Cubic nonlinearities and hysteretic damping, forced response	164
Figure 5.12.	Cubic nonlinearities and hysteretic damping, forced response; steady-state response	165
Figure 5.13.	Cubic nonlinearities and hysteretic damping, altered $\mathcal{C}_1$	166
Figure 5.14.	Cubic nonlinearities and hysteretic damping, $\mathcal{C}_2$ relocated to $s \in [0.45, 0.55]$	167
Figure 5.15.	Cubic nonlinearities and hysteretic damping, $\mathcal{C}_2$ relocated to $s \in [0.45, 0.55]$	167
Figure 6.1.	ML-enhanced structure-preserving operator inference architecture.	181
Figure 6.2.	Conservative rod model. The nonlinearities are localized within the region $s \in (s_1, s_2)$, while the remainder of the rod is linear.	187

Figure 6.3.	Conservative rod model. (a) The LOpInf-SpML ROM correctly predicts $q_1(t)$ trajectory 100% outside the training time interval. (b) The L-OpInf SpML ROM exhibits bounded energy error behavior. The magenta line in both plots indicates the end of the training time interval at $T_{\text{train}} = 8$	189
Figure 6.4.	Conservative rod model. An accurate phase space portrait obtained using the LOpInf-SpML ROM demonstrates that the proposed approach has learned the underlying nonlinear dynamics.	190
Figure 6.5.	Conservative rod model. The LOpInf-SpML ROM starts at a significantly lower validation loss value and achieves the lowest error achieved by the POD-SpML ROM in $100\times$ less epochs. The dotted black line indicates the lowest validation error achieved by the POD-SpML ROM.	190
Figure 6.6.	Conservative rod model. The LOpInf-SpML ROM is robust to perturbations in the training initial condition whereas the POD-SpML ROM fails to provide accurate predictions for unseen initial conditions. The magenta star in both plots represents the training initial condition.	191
Figure 6.7.	Nonconservative membrane model. Plots in (a) and (b) show that the LOpInf-SpML ROM of reduced dimension $r = 12$ provides accurate predictions 100% outside the training time interval. The magenta line indicates the end of the training time interval.	192
Figure 6.8.	Nonconservative membrane model. The full-field solution comparison in plots (a), (b), and (c) shows that the LOpInf-SpML ROM provides accurate predictions over the entire computational domain.	193
Figure 6.9.	Nonconservative membrane model. In the training data regime, both ROMs capture the frequency content in the training data accurately. In the test data regime, we observe that the LOpInf-SpML ROM performs significantly better than the POD-SpML ROM.	194
Figure 6.10.	Half Brake-Reuß beam. Geometry of the jointed structure including a photo with DIC speckle pattern shown (a), top view (b), and front view (c). Figure reprinted from [45], Copyright (2022), with permission from Elsevier and the authors.	199
Figure 6.11.	Half Brake-Reuß beam. The LOpInf-SpML ROM of size $r = 3$ accurately predicts the amplitude-dependent frequency characteristics and yields backbone curves that appear to agree with the backbone curves obtained directly from the DIC measurements.	199

Figure 6.12.	Half Brake-Reuß beam. The amplitude-dependent damping plots based on the LOpInf-SpML ROM of dimension $r = 3$ are reasonably similar to the plots obtained from the DIC measurements.	200
Figure 6.13.	Half Brake-Reuß beam. The proposed approach accurately predicts the frequency-energy curve in (a) and captures the overall trend of the challenging damping-energy curve in (b).	201
Figure 7.1.	General Structure; the nonlinearities are localized within the region $\mathcal{C}_2$. ..	206
Figure 7.2.	Neural network architecture used to model the deviatoric force term. The inputs to the network and to the polynomial terms are $\mathbf{z}_\beta$ and $\dot{\mathbf{z}}_\beta$. The outputs from the network and the polynomial terms are added together to obtain the deviatoric force.	210
Figure 7.3.	Predictions produced with neural network trained with noisy and clean data for input data with noise levels of 5% and 20%. Network predictions: black, actual: red.	216
Figure 7.4.	Close-up of predictions for noisy input data (5%) and corresponding histograms at three points in time. Clean: green, noisy: magenta, actual: red.	218
Figure 7.5.	Close-up of predictions for noisy input data (20%) and corresponding histograms at three points in time. Clean: green, noisy: magenta, actual: red.	218
Figure 7.6.	Response obtained from time integration with trained neural network in the loop. Network predictions: black, actual: red.	219
Figure 7.7.	Deviatoric force predictions from noisy inputs (at 5% noise level) at three points in time alongside predictive confidence intervals. Actual response shown in red, 95% predictive confidence intervals shown in yellow. Green line indicates time point at which the histograms are computed.	220
Figure 7.8.	Deviatoric force predictions from noisy inputs (at 10% noise level) at three points in time alongside predictive confidence intervals. Actual response shown in red, 95% predictive confidence intervals shown in yellow. Green line indicates time point at which the histograms are computed.	221
Figure 7.9.	Deviatoric force predictions from noisy inputs (at 20% noise level) at three points in time alongside predictive confidence intervals. Actual response shown in red, 95% predictive confidence intervals shown in yellow. Green line indicates time point at which the histograms are computed.	222

Figure 7.10.	Deviatoric force predictions from noisy inputs (at 5% noise level) at three points in time alongside predictive confidence intervals. Low level damaged response shown in red, 95% predictive confidence intervals shown in yellow. Green line indicates time point at which the histograms are computed. . . .	223
Figure 7.11.	Deviatoric force predictions from noisy inputs (at 20% noise level) at three points in time alongside predictive confidence intervals. Low level damaged response shown in red, 95% predictive confidence intervals shown in yellow. Green line is time point at which the histograms are computed. .	224
Figure 7.12.	Deviatoric force predictions from noisy inputs (at 20% noise level) at three points in time alongside predictive confidence intervals. High level damaged response shown in red, 95% predictive confidence intervals shown in yellow. Green line is time point at which the histograms are computed. .	225
Figure 7.13.	Velocity time history (left) and power spectral density (PSD) across isolated region boundary. The baseline response is shown in black and the response for the damaged cases is shown in blue.	226
Figure 7.14.	Histograms of velocity time history samples corresponding to different damage levels.	227
Figure 7.15.	Histograms of velocity PSD samples corresponding to different damage levels.	227
Figure 7.16.	Velocity time histories (top left) and PSDs (top right) and p-values as a function of time (left) and frequency (right) with $p = 0.05$ shown with a dashed black line.	229
Figure 7.17.	P-values as a function of time and frequency computed from STFT of data computed with the KS test. Upper color bar limit is set to $\log(0.05)$ so that anything that is yellow exceeds $p = 0.05$	230
Figure 7.18.	P-values as a function of time and frequency computed from STFT of data computed with the CVM test. Upper color bar limit is set to $\log(0.05)$ so that anything that is yellow exceeds $p = 0.05$	231
Figure 8.1.	Neural network architecture used to model the deviatoric force term. The inputs to the network and to the polynomial terms are $\mathbf{z}_\beta$ and $\dot{\mathbf{z}}_\beta$. The outputs from the network and the polynomial terms are added together to obtain the deviatoric force.	240
Figure 8.2.	Phase portraits of a representative DOF (DOF = 11). Mean percent error annotated for each case. Actual response shown in a solid blue line, predicted response shown in a black dashed line.	252

Figure 8.3.	Top: nonlinear damping mean ($\mathcal{J}_{nl}$) as a function of time with one standard deviation shown around it. Bottom: variance of the nonlinear damping as a function of time.	253
Figure 8.4.	Top: box plots for each cases considered of the log-variance of the nonlinear potential (top) and the nonlinear damping prediction.	255
Figure 8.5.	Top: nonlinear damping mean ($\mathcal{J}_{nl}$) as a function of time with one standard deviation shown around it. Bottom: variance of the nonlinear damping as a function of time.	256
Figure 8.6.	Comparison of dispersion metrics as a function of domain shift. Blue: IQR, Black: Variance. Dashed lines represent the baseline dispersion level. ...	258
Figure 8.7.	Top: violin plots of IQR distributions. Bottom: Wald test statistic for each case vs. Baseline 0. Dashed line represents 95% confidence interval (or 5% p-value).	260
Figure 8.8.	Top: MDS of 128-dimensional state space. Bottom: MDS of 32-dimensional hidden space. Color groups are consistent with Fig. 8.4. The marker “ $\times$ ” indicates the Baseline 0 case.	262
Figure 9.1.	Left: ROC curves colored by WS value (blue to red: short to large). Right: P-value as a function of WS.	273
Figure 9.2.	IQR histograms for varying window sizes.	274
Figure 9.3.	Cost vs. WS as a function of prior probability of damage and λ	277
Figure 9.4.	IQR mean and one standard deviation band around it for the baseline (blue) and the damaged case (green).	278
Figure 9.5.	Cost vs. probability of false alarm as a function of prior probability of damage, λ , and γ	279
Figure 9.6.	Confusion matrix across all damage cases as a function of window size. Predicted condition shown across horizontal axis and actual conditions shown across vertical axis.	281
Figure 9.7.	F1 score as a function of window size.	282
Figure 9.8.	Nonlinear potential (top) and damping (bottom) time histories. One standard deviation intervals shown.	283
Figure 9.9.	Histograms of log-variance of nonlinear potential (left) and damping (right) network weights.	284

Figure 9.10.	Histograms of log-variance of nonlinear potential (left) and damping (right).	285
Figure A.1.	Validation loss as a function of epochs during training.....	289
Figure B.1.	Decoded oscillator time histories. Dashed black is the actual response, blue is the INN decoding, and yellow is the MLP decoding.....	293

LIST OF TABLES

Table 2.1.	Computational cost of a single RUL curve for each method.	33
Table 2.2.	Summary of RUL Prediction Performance	36
Table 2.3.	Ranges of Input Parameters	55
Table 2.4.	Computational cost of predicting 200 time steps for each method.	56
Table 2.5.	Average MSE of all test cases across 200 time steps for each method.	57
Table 3.1.	Summary of Neural Network Performance	85
Table 4.1.	Summary of results for rod model	119
Table 6.1.	Relative state error comparison between different methods.....	189
Table 8.1.	Domain shift cases. For the gradual stiffness increase, the max value is provided. The initial condition (IC) coefficients represent the contribution of each mode shape to the initial deformation.....	250
Table A.1.	Summary of hyperparameter grid search results.	290
Table B.1.	Hyperparameters for autoencoder	291
Table B.2.	Hyperparameters for energy networks	292
Table B.3.	Training settings	292

ACKNOWLEDGEMENTS

The work presented in this dissertation was possible thanks to the support of mentors, colleagues, co-workers, friends, and family who contributed in many ways to my doctoral journey. First and foremost, I am grateful to my advisor, Dr. Michael Todd who gave me the opportunity to go on this exciting adventure, and whose technical advice has been invaluable and constant. I look forward to continue collaborating with you in the future. I am also thankful for the technical advice I received from my PhD committee: Dr. Jiun-Shyan Chen, Dr. Charles Farrar, and Dr. Boris Kramer, whose direct advice was always insightful, and whose publications I consulted constantly throughout my academic journey.

I am also grateful to the various PIs at Sandia who continued to support me over the years: Dr. Adam Brink, Dr. Kurtis Ford, Mr. Steven Gomez, Dr. Lyndsay Shand, Dr. John Emery, and Dr. Anthony Garland. I am indebted to you for the trust you had in me. Thanks to all my colleagues at Sandia who have taught me so much about engineering. I would also like to acknowledge Dr. Dane Quinn, Dr. Zhen Hu, Dr. Justin Jacobs, and Dr. Eleni Chatzi for their technical mentorship over my academic career. The invaluable technical advice I received from you was fundamental to my academic success.

Some parts of this work would not have been possible without the excellent collaborators I had, including Dr. Mayank Chadha, Dr. Guofeng Qian, Dr. Harsh Sharma, and Mr. Konstantinos Vlachas. It has been a pleasure collaborating with you and I look forward to continued collaboration in the future. I would also like to thank my other peers in the SHM lab, Dr. Adrielly Hokama Razzini and Dr. Zihan Wu for the great moments we shared at IMAC.

I would also like to acknowledge the support I received from the leadership team at ATA Engineering, who continued to support me as I worked on this research. That includes Dr. Matthew Kaplan, Dr. Paul Blelloch, Mr. Tom Deiters, Mr. Kurt Knutson, and Mr. Josh Davis as well as the late Dr. Mary Baker whose encouragement and enthusiasm was instrumental in my decision to take on this challenge. I am grateful for the opportunity you gave me and the encouragement I always received. I am also thankful to all my co-workers who have been

supportive during this journey.

Lastly, I would like to express my love and gratitude to all my family and friends who were always there for me. To my parents, Martha and Ricardo, and my brother, Ricky, who have always been loving and supportive of everything I do. And to my wife, Daniela, who has been the best study buddy I could ever hope for, spending hours at Geisel or trapped at home during a global pandemic studying vastly different subjects, but always making every moment better.

The research in this dissertation was funded by Sandia National Laboratories and ATA Engineering, Inc. Portions of this dissertation have been published or are currently being considered for publications.

Chapter 2 is composed of the following first-authored publications:

David A. Najera-Flores, Zhen Hu, Mayank Chadha, and Michael D. Todd. A physics-constrained bayesian neural network for battery remaining useful life prediction. *Applied Mathematical Modelling*, 122:42–59, 2023

David A. Najera-Flores, Guofeng Qian, Zhen Hu, and Michael D. Todd. Corrosion morphology prediction of civil infrastructure using a physics-constrained machine learning method. *Mechanical Systems and Signal Processing*, 200:110515, 2023

Chapter 3 is composed of the following first-authored publication:

David A. Najera-Flores and Michael D. Todd. A structure-preserving neural differential operator with embedded Hamiltonian constraints for modeling structural dynamics. *Computational Mechanics*, 72:241–252, March 2023

Chapter 4 is composed of the following first-authored publication:

David Najera-Flores and Michael Todd. State-space reconstruction from partial observables using an invertible neural network with structure-preserving properties for nonlinear structural dynamics. Manuscript accepted for publication in *Nonlinear Dynamics*, 2024

Chapter 5 is composed of the following first-authored publication:

David A. Najera-Flores, D. Dane Quinn, Anthony Garland, Konstantinos Vlachas, Eleni Chatzi, and Michael D. Todd. A structure-preserving machine learning framework for accurate

prediction of structural dynamics for systems with isolated nonlinearities. *Mechanical Systems and Signal Processing*, 213:111340, 2024

Chapter 6 is composed of the following first co-authored publication:

Harsh Sharma, David A. Najera-Flores, Michael D. Todd, and Boris Kramer. Lagrangian operator inference enhanced with structure-preserving machine learning for nonintrusive model reduction of mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 423:116865, 2024

Chapter 7 is composed of the following first-authored publication:

David A. Najera-Flores, Justin Jacobs, D. Dane Quinn, Anthony Garland, and Michael D. Todd. Uncertainty Quantification of a Machine Learning Model for Identification of Isolated Nonlinearities With Conformal Prediction. *Journal of Verification, Validation and Uncertainty Quantification*, 9(2):021005, 06 2024

Chapter 8 is composed of the following first-authored publication:

David Najera-Flores, Justin Jacobs, D. Dane Quinn, Anthony Garland, and Michael Todd. Uncertainty-aware, structure-preserving machine learning approach for domain shift detection from nonlinear dynamic responses of structural systems. Manuscript accepted for publication in the Journal of Risk and Uncertainty in Engineering Systems Part B: Mechanical Engineering, 2024

Chapter 9 is composed of previously unpublished work.

VITA

2010	Associate of Science, Southwestern Community College
2013	Bachelor of Science, University of California San Diego
2014	Master of Science, University of California San Diego
2014-2024	Senior Project Engineer, ATA Engineering, Inc.
2020	Master of Computer Science, University of Illinois at Urbana-Champaign
2024	Doctor of Philosophy, University of California San Diego

PUBLICATIONS

Journal Publications

David Najera-Flores and Michael Todd. State-space reconstruction from partial observables using an invertible neural network with structure-preserving properties for nonlinear structural dynamics. Manuscript accepted for publication in *Nonlinear Dynamics*, 2024

David Najera-Flores, Justin Jacobs, D. Dane Quinn, Anthony Garland, and Michael Todd. Uncertainty-aware, structure-preserving machine learning approach for domain shift detection from nonlinear dynamic responses of structural systems. Manuscript accepted for publication in the *Journal of Risk and Uncertainty in Engineering Systems Part B: Mechanical Engineering*, 2024

Harsh Sharma, David A. Najera-Flores, Michael D. Todd, and Boris Kramer. Lagrangian operator inference enhanced with structure-preserving machine learning for nonintrusive model reduction of mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 423:116865, 2024

David A. Najera-Flores, Justin Jacobs, D. Dane Quinn, Anthony Garland, and Michael D. Todd. Uncertainty Quantification of a Machine Learning Model for Identification of Isolated Nonlinearities With Conformal Prediction. *Journal of Verification, Validation and Uncertainty Quantification*, 9(2):021005, 06 2024

David A. Najera-Flores, Jonel Ortiz, Moheimin Y. Khan, Robert J. Kuether, and Paul R. Miles. A Bayesian Multi-Fidelity Neural Network to Predict Nonlinear Frequency Backbone Curves. *Journal of Verification, Validation and Uncertainty Quantification*, 9(2):021003, 06 2024

David A. Najera-Flores, Zhen Hu, Mayank Chadha, and Michael D. Todd. A physics-constrained

bayesian neural network for battery remaining useful life prediction. *Applied Mathematical Modelling*, 122:42–59, 2023

David A. Najera-Flores, Guofeng Qian, Zhen Hu, and Michael D. Todd. Corrosion morphology prediction of civil infrastructure using a physics-constrained machine learning method. *Mechanical Systems and Signal Processing*, 200:110515, 2023

David A. Najera-Flores and Michael D. Todd. A structure-preserving neural differential operator with embedded Hamiltonian constraints for modeling structural dynamics. *Computational Mechanics*, 72:241–252, March 2023

David A. Najera-Flores, D. Dane Quinn, Anthony Garland, Konstantinos Vlachas, Eleni Chatzi, and Michael D. Todd. A structure-preserving machine learning framework for accurate prediction of structural dynamics for systems with isolated nonlinearities. *Mechanical Systems and Signal Processing*, 213:111340, 2024

David A. Najera-Flores and Robert J. Kuether. Approximating nonlinear response to harmonic base-excitation using fixed-interface quasi-static modal analysis. *Journal of Sound and Vibration*, 544:117381, 2023

Adam R. Brink, David A. Najera-Flores, and Cari Martinez. The neural network collocation method for solving partial differential equations. *Neural Comput. Appl.*, 33(11):5591–5608, jun 2021

Michael Ross, David Najera-Flores, and James Freymiller. Including Mean Stress in the Effective Equivalent Stress of the ASME BPVC Elastic–Plastic Fatigue Method. *Journal of Nondestructive Evaluation, Diagnostics and Prognostics of Engineering Systems*, 5(3):031006, 05 2022

David A. Najera-Flores and Robert J. Kuether. A Study of Whole Joint Model Calibration Using Quasi-Static Modal Analysis. *Journal of Vibration and Acoustics*, 142(5):051109, 06 2020

Conference Publications

David A. Najera-Flores and Michael D. Todd. Ensemble of numerics-informed neural networks with embedded hamiltonian constraints for modeling nonlinear structural dynamics. In Matthew R.W. Brake, Ludovic Renson, Robert J. Kuether, and Paolo Tiso, editors, *Nonlinear Structures & Systems, Volume 1*, pages 27–30, Cham, 2023. Springer International Publishing

Konstantinos Vlachas, David Najera-Flores, Carianne Martinez, Adam R. Brink, and Eleni Chatzi. A physics-based reduced order model with machine learning-boosted hyper-reduction. In Brandon J. Dilworth, Timothy Marinone, and Michael Mains, editors, *Topics in Modal Analysis & Parameter Identification, Volume 8*, pages 131–139, Cham, 2023. Springer International Publishing

David A. Najera-Flores and Michael D. Todd. State space reconstruction from embeddings of partial observables in structural dynamic systems for structure-preserving data-driven methods. In Ramin Madarshahian and François Hemez, editors, *Data Science in Engineering, Volume 10*, pages 139–141, Cham, 2023. Springer Nature Switzerland

Justin Jacobs, David Najera-Flores, Adam R. Brink, and Tatiana Flanagan. Parameter estimation for dynamical systems under continuous and discontinuous gaussian noise using data assimilation techniques. In Ramin Madarshahian and François Hemez, editors, *Data Science in Engineering, Volume 9*, pages 135–143, Cham, 2022. Springer International Publishing

David A. Najera, Christopher I. Van Damme, and Victoria Harris. Model reduction of geometrically nonlinear structures via physics-informed autoencoders. In Ramin Madarshahian and François Hemez, editors, *Data Science in Engineering, Volume 9*, pages 145–148, Cham, 2022. Springer International Publishing

Carianne Martinez, David A. Najera-Flores, Adam R. Brink, D. Dane Quinn, Eleni Chatzi, and Stephanie Forrest. Confronting domain shift in trained neural networks. In Luca Bertinetto, João F. Henriques, Samuel Albanie, Michela Paganini, and Gül Varol, editors, *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, volume 148 of *Proceedings of Machine Learning Research*, pages 176–192. PMLR, 11 Dec 2021

David Najera, Eric Jayson, Thomas Board, and Doug Melville. A framework for bayesian calibration of fiber-reinforced composite material properties. In *Proceedings of the American Society for Composites: Thirty-fifth Technical Conference*, 2020

Jonathan Buck, David Najera, Doug Melville, and Eric Jayson. Prediction of fiber-reinforced composite material properties using collaborative filtering techniques. In *Proceedings of the American Society for Composites: Thirty-third Technical Conference*, 2018

Robert J. Kuether and David A. Najera. Parameter estimation of joint models using global optimization. In Matthew S. Allen, Randall L. Mayes, and Daniel Jean Rixen, editors, *Dynamics of Coupled Structures, Volume 4*, pages 29–39, Cham, 2017. Springer International Publishing

Jonathan Buck, Eric Jayson, and David Najera. Using optimization to improve the quasi-isotropic status quo. In *Proceedings of the American Society for Composites: Thirty-First Technical Conference*, 2016

FIELDS OF STUDY

Major Field: Structural Engineering

ABSTRACT OF THE DISSERTATION

Physics-Constrained, Structure-Preserving Machine Learning Models for Structural Health Applications

by

David A. Najera-Flores

Doctor of Philosophy in Structural Engineering

University of California San Diego, 2024

Professor Michael D. Todd, Chair

Many different industrial sectors such as aerospace, civil, mechanical, and automotive rely on complex systems that have evolved beyond their original design due to, among other things, damage, aging, upgrades, or degradation. The mismatch between the original design life and the current state of these systems motivates the need for an improved representation of their as-built, as-deployed state to monitor their structural health. Furthermore, as these systems continue to evolve beyond their original design, this virtual representation needs to adapt as well. To help ensure the reliability of these systems, a structural health monitoring (SHM) system that relies on an accurate representation of the system it monitors is desired. This

dissertation presents the development of a workflow for integration of physics-constrained and structure-preserving machine learning (ML) models that can function as digital twins of a system with the purpose of enabling response forecasting to future states, as well as provide a basis for damage detection based on domain shifts observed in the data from the monitored system. The proposed approach leverages the well-established foundations of computational mechanics theory to constrain the parameter space of the ML models. This dissertation presents a series of extensions and variations of the basic principles of structure preservation in nonlinear dynamics to increase efficiency by reducing the data needs (through a sparse measurement requirement), reducing the computational burden through order reduction, and leveraging prior knowledge of the ideal system before it deviated from its nominal condition. Furthermore, this work proposes a strategy to deal with observational uncertainty (e.g., from measurement error) and epistemic uncertainty (e.g., from model limitations). Lastly, the digital twin model is used in combination with detection theory to establish a probabilistic reasoner that enables risk- and cost-informed decision-making. The entire workflow establishes a strategy for physically-interpretable ML models that can be interrogated to infer health states and to forecast their structural health state under future operational loads with increased accuracy.

Chapter 1

Introduction

The reliability of complex civil, aerospace, and mechanical structures is of utmost importance for their robust operation. These systems are often exposed to environments that may cause damage or changes to their structural performance over their lifetime. To prevent failure of these systems, time-based maintenance (TBM) or condition-based maintenance (CBM) techniques are employed. TBM is common in many applications where time to failure is known in some statistical manner. However, TBM lacks a connection to performing continuous monitoring, and therefore may be either over- or under-conservative, depending on the maintenance schedule; maintenance is performed on a fixed schedule, regardless of the actual need. On the other hand, CBM relies on monitoring (either continuous or regularly scheduled) to trigger maintenance actions. CBM requires a reliability model that can be used to inform the health state of the structure based on measurements, which leads to structural health monitoring (SHM).

SHM makes use of a condition monitoring workflow that combines a reliability model with the ability to extract informative features from these data and, subsequently, assess the state of the structure. To be effective, these SHM systems rely on a computational representation of the monitoring structure so that they can perform "forward" predictions based on the current state of the structure (otherwise known as "prognostics"). This virtual representation is also known as a "digital twin". Digital twins seek to provide a faithful representation of the deployed system in the field [47, 75, 262, 246]. To be truly effective, these computational models need continued

monitoring of the physical systems via embedded sensors [272, 165] to preserve their fidelity when structural changes occur. Since they are meant to operate in real-time aided by continuous data streams, they offer an enhanced ability to monitor the evolving health of a system and predict the future response to both nominal operating conditions and unexpected excursions in loading environments [250, 21, 246, 247].

Digital twins can be fully physics-based when sufficiently complete knowledge of the structure is available. To be effective, these physics-based models still require calibration to test data (e.g., modal tests in the case of vibration measurements). For example, finite element models may function as digital twins if they are an accurate representation of the as-built, as-deployed system. However, to be truly effective, these models need constant updating to reflect the latest state of the systems they represent. This updating can be expensive when dealing with physics-based models, which often need to be solved with high-performance computers. SHM often relies on data-driven models which have the advantage of being really fast but may be unreliable if not properly trained. A more robust alternative is a class of hybrid systems that combine data-driven approaches with physics-based models. This spectrum of data vs. physical knowledge will be explored in more detail in Section 1.1.

1.1 Machine learning applications to physics

In this section, we provide a description of how machine learning has been used over the years for modeling physical systems. The objective of this section is to provide an overview of how digital twins of physical systems can be obtained with data-driven methods.

1.1.1 Purely data-driven approaches

Purely data-driven modeling approaches are popular when experimental data are available because they can be readily obtained by employing a type of regression to these data. These approaches range from more traditional techniques such as singular value or proper orthogonal decomposition [119, 16, 17] to more recent work with neural networks, with applications in

areas such as climate modeling [168, 128], material modeling [266, 276, 116], and structural dynamics [236, 274]. Purely data-driven methods generally lack physical interpretability and are limited in the domain they may operate in (e.g., not outside their training domain), and as a result, they are not very robust. Researchers have shown [69] how these systems fail when presented with data outside the training regime (or in a shifted domain). While these methods can still be extremely useful for applications where the expected inputs are well characterized, they are often not very robust in SHM applications where the digital twin may be presented with data from a shifted domain (e.g., with some underlying structural change).

1.1.2 Physics-informed approaches

More recently, physics-informed neural networks (PINNs) have become more pervasive in the literature, ranging from applications in solving classical partial differential equations (PDEs) [203, 121, 277] including the Navier-Stokes equation [113], as well as applications in geostatistical modeling [290], defect detection [281], and solution of non-homogeneous dynamic systems and elasticity problems [27], to larger scale systems including hybrid mesh-free methods [237, 93]. These approaches typically operate by transforming the PDE residual into a loss term, as well as enforcing boundary and initial conditions with similar penalty methods. However, these methods require the most amount of physical knowledge to operate. PINNs and other closely related approaches rely on the full knowledge of the PDE and assume full access to the state variables needed for the PDE. While these approaches may offer numerous advantages over classical physics-based methods (e.g., FEA), they still require a very specific form of data that may restrict its applicability to complex engineering systems. For example, for material modeling, access to the stresses and deformation gradients may be required to evaluate the PDEs. For this reason, these methods are well equipped for replacing physics-based models in a context where high-fidelity modeling is required but may not be the best fit for SHM.

1.1.3 Physics-constrained approaches

Physics-constrained methods represent a middle ground between purely data-driven and physics-informed approaches. In these methods, a constraint representing known physical behavior is introduced either through the loss function or through the network architecture. For example, a common constraint is energy conservation or consistency with physical laws (such as thermodynamic laws) [89, 54]. Structure-preserving methods can be categorized under the general family of physics-constrained approaches. In contrast with physics-informed methods, structure-preserving approaches seek to enforce physical constraints through inductive biases built into the data-driven mechanism, e.g., the neural network architecture or in reduced order modeling via an appropriate choice of reduction basis [217, 197, 137]. This family of methods have been shown to be more robust than purely data-driven approaches while providing much more flexibility than PINNs. A key reason for why physics-constrained and structure-preserving techniques may be preferred over physics-informed methods is the fact that model nonlinearities may be unknown or hard to characterize with existing physics-based models. For instance, the physical phenomenon of friction in bolted joints still lacks a complete physics-based characterization at the structural level, necessitating approximations through whole joint models that aim to capture the observed behavior [163, 25]. Much like these models that have been developed by subject matter experts, physics-constrained and structure-preserving approaches aim to model physical behavior that may be hard to characterize through first principles by enforcing adherence of the model's behavior to physical laws.

1.2 Research Objectives, Organization of Dissertation, and Contributions

Based on the discussion in the previous sections, this research focuses on the application of physics-constrained and structure-preserving machine learning techniques to structural health monitoring applications. In this work, physics constraints are used to limit the function space of

data-driven methods in order to establish reliable and robust digital twins that can be deployed in SHM frameworks. The initial research was motivated by applications to transportation systems that require to operate robustly under normal environments (e.g., subjected to extended random vibration environments). However, the methods presented here were developed in an abstract framework that is applicable to any structural system. Ultimately, these tools are employed to detect damage in a system, which is understood as "intentional or unintentional changes to the properties of the system [...] which adversely affect the current or future performance of the systems", based on the definition provided by Farrar and Worden [65].

Figure 1.1 summarizes the proposed workflow which consists of two distinct phases: predictive modeling, or the digital twin phase, and the probabilistic modeling phase. In the first phase, physics-constrained or structure-preserving methods are developed to establish a reliable data-driven digital twin of a deployed system. In the second phase, probabilistic models are employed to characterize the uncertainty associated with the digital twin predictions and to realize the SHM paradigm by detecting changes in the system and providing the necessary tools for risk- and cost-informed decision-making. In the digital twin phase, there are a number of cases that were considered given the diversity of data that were of interest. However, the problems considered in this work all fall under the same general family of problems where data that was generated by a physical process are available. We recognize that each problem is unique and may require variations to the methods developed in this work, but we offer a variety of approaches that cover many of the challenges associated with complex engineering systems. To this end, the objectives of the research are summarized as follows:

- Establish a general family of physics-constrained and structure-preserving data-driven methods for deployment as digital twins in SHM applications, mostly focused on nonlinear structural dynamics
- Address the practical challenges of dealing with experimental data by providing uncertainty quantification, recognizing the challenge of partial observability in realistic systems, and

enabling model order reduction by leveraging linear projection methods thereby reducing the computational burden on the ML algorithms

- Propose an SHM workflow that enables the development of physically-interpretable models of as-built, as-deployed systems from data

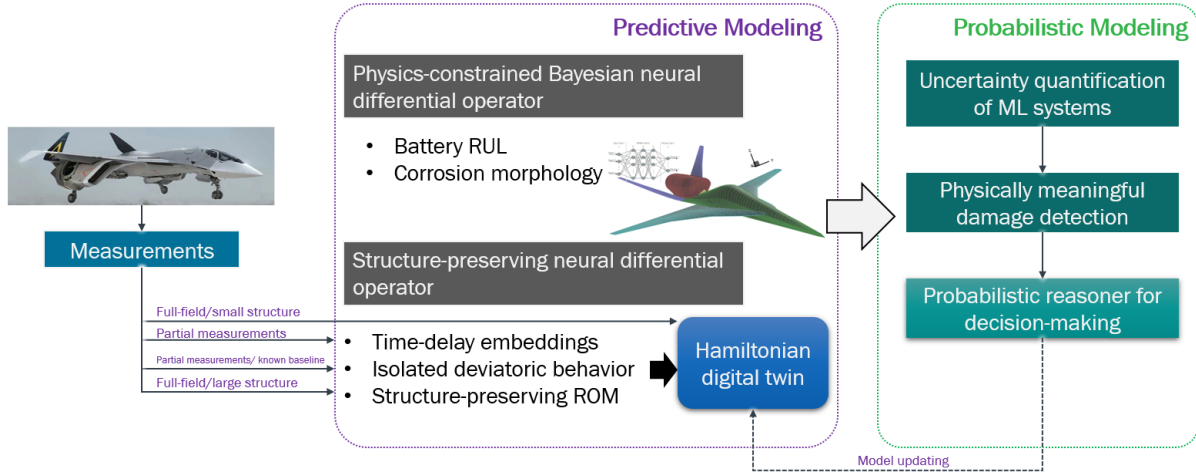


Figure 1.1. Proposed framework for SHM system.

The following eight chapters address these objectives and are organized as follows. Chapters 2 thru 6 constitute the predictive modeling portion of the research while Chapters 7 thru 9 represent the probabilistic modeling portion. Chapter 2 describes the development of two algorithms that incorporate physics constraints through a loss function penalty for health prognosis. Both algorithms incorporate epistemic uncertainty quantification through a Bayesian neural network architecture. The concept of neural differential operators is also introduced in this chapter. Chapter 3 shows the development of the structure-preserving algorithm that will act as the foundation for much of the rest of the dissertation. The general theory behind Hamiltonian and Lagrangian mechanics is covered in this chapter. Chapter 4 addresses the challenge of partial observability in realistic applications through the application of embedding theory. Chapter 5 leverages an existing isolated formulation of nonlinear dynamical systems to demonstrate how existing knowledge of the structural system can be integrated into the data-driven digital twin

to obtain accurate representations of an unknown nonlinear behavior. Chapter 6 demonstrates how the data-driven digital twin can be combined with projection-based reduced order modeling methods to reduce the computational burden on the neural network training. Chapter 7 develops a general approach of uncertainty quantification of the digital twin predictions given observational noise in the measurements that are informing the SHM system. Chapter 8 extends the uncertainty quantification of the digital twin to handle epistemic uncertainty via distance-aware neural networks while leveraging this measure of uncertainty to detect domain shift in the data. And lastly, Chapter 9 demonstrates how the features extracted from the data in Chapter 8 can be used for risk- and cost-informed decision making by computing the detector's Bayes risk.

To summarize, the contributions of this research are the following:

- Development of a physics-constrained surrogate model for pitting corrosion prediction
- Development of a physics-constrained model for remaining useful life prediction of lithium-ion batteries
- Development of a structure-preserving digital twin that preserves the Hamiltonian and/or Lagrangian structure of general dynamical systems
- Development of a workflow for training structure-preserving data-driven models when only partial measurements are available
- Development of a workflow to leverage existing knowledge of physical systems when available, limiting the data-driven aspect of the digital twin to an isolated nonlinear region
- Development of a structure-preserving model order reduction approach for nonlinear systems based on Lagrangian operator inference
- Development of an approach for uncertainty quantification of the digital twin system when observational noise is present through conformal regression

- Development of an approach for epistemic uncertainty quantification of the digital twin for use in a probabilistic reasoner to perform risk- and/or cost-informed decision-making

Chapter 2

Physics-Constrained Neural Differential Operators for Health Prognosis

This chapter describes the use of neural differential operators for health prognosis applications in two different applications: remaining useful life forecasting of Lithium-ion batteries, and pitting corrosion growth in metallic structures. These two applications employ different ML architectures but rely on a similar concept: a probabilistic data-driven ML model with physics constraints. In both applications, it is demonstrated that the proposed approach has several advantages over other state-of-the-art methods.

2.1 Applications to Lithium-ion battery remaining useful life prediction

Lithium-ion battery health prognosis, or the ability to predict its remaining useful life, is crucial for ensuring the reliability and efficiency of the systems that rely on its power. Monitoring and forecasting the remaining battery capacity provides valuable insight into the overall performance of the battery and allows for the prediction of battery failure. These predictions facilitate timely maintenance and/or replacement, improves the overall energy efficiency of the system, and maximizes system operational availability (i.e., minimizing downtime). Battery prognosis is a crucial requirement for various systems, including electric vehicles (for optimizing range and performance), renewable energy systems (for improving energy storage), aerospace

and defense applications (for predicting remaining battery life in satellites), Internet of Things (for monitoring device battery life to ensure seamless operation), among others.

Failure prognostics—the capability to predict remaining useful life (RUL) estimation—of lithium-ion batteries has been a very active research topic for a couple of decades with the rise of such battery usage across many applications [1, 3, 283, 83]. Approaches to failure prognostics of batteries may be roughly classified into three categories, namely physics-based, data-driven, and hybrid approaches. This classification is similar to that for failure prognostics in other engineering domains [246]. Physics-based methods heavily rely on modeling battery degradation patterns using physics-based or physics-informed models, e.g., taking advantage of electro-chemical principles. The physics-based model could be a high-fidelity simulation model [154], an empirical/semi-empirical model [187], or a simplified model derived from physical laws [188]. For physics-based methods, failure prognostics of batteries usually consists of two main steps. The first step is to update the uncertain model parameters based on actual observations of a battery, using Bayesian methods [104, 33, 263], optimization-based methods [62], or other methods [213, 287]. After the model updating, the physics-based model is then used to forecast the degradation of the battery and thus estimate its RUL. The commonly used models in the literature include the exponential function [40], the power-law function [263], the equivalent circuit models [11, 166], and multi-physics simulation-based degradation models [231]. There are multiple advantages of physics-based methods. For instance, most of the physics-based methods do not require any run-to-failure data. Furthermore, physics-based methods have good extrapolation capabilities if the model accurately captures the underlying degradation principle of the batteries. However, this potential advantage becomes the main disadvantage, because the current state of the art knowledge of battery degradation is very limited due to the highly complicated degradation process [60]. The discrepancy between the model used for prediction and that of the underlying true physics could lead to large errors in the RUL prediction, resulting in sub-optimal if not costly consequences in the decision-making.

With the rapid development of machine learning (ML) and especially deep learning

techniques in recent years, many data-driven approaches have been developed for prognostics to overcome the limitations of the lack of physical degradation knowledge [139, 143, 149, 147]. In data-driven approaches, a model is constructed based on run-to-failure data to achieve a mapping between observables (voltage, current, etc.) and battery capacity or monitoring data and life directly [244]. Various ML models have been employed to construct such data-driven models for failure prognostics of batteries. For instance, numerous neural network-based data-driven models, including regular artificial neural network models and models based on deep neural networks, have been proposed for the capacity estimation of lithium-ion batteries [36, 205, 102]. The Gaussian process regression model-based method has also been proposed to estimate the RUL of batteries, even though it is not as widely used as neural network architectures [145, 241]. Moreover, to overcome the limitations of individual ML models and improve the RUL estimation accuracy and robustness, ensemble learning methods have also been investigated in the literature and have shown promising performance [186, 227]. Other researchers have shown promise with the use of recurrent neural network architectures [286, 148], including combining these architectures with classical Bayesian approaches to enable uncertainty quantification [146]. The advantage of an ML-based data-driven model is that it is capable of generating the required mapping that is not conditioned upon an understanding of the underlying physics of degradation. The ML model learns the degradation pattern automatically purely based on run-to-failure data. The disadvantage is that the data used for training significantly affects the performance of the data-driven model. If the prediction region is outside of the training space, the data-driven models could have a very large prediction error. The model sometimes may also create predictions that violate the physical laws.

While both physics-based methods and emerging data-driven methods have their own advantages and have shown good performance for RUL estimation of batteries in some situations, their disadvantages have impeded the adoption of the methods in practical applications. This has motivated the development of hybrid approaches that combines physics-based knowledge with data-driven models [39, 245]. The hybrid approaches have been explored in two main directions.

In the first direction, run-to-failure data is used to correct the physics-based degradation models [275, 109, 114]. In the second direction, physics-based knowledge is incorporated into data-driven models to improve the extrapolation accuracy of data-driven models for failure prognostics of batteries [229, 141, 249, 125]. A comprehensive review of various hybrid approaches for battery RUL estimation is available in [141]. It is believed that the hybrid approaches will outperform the performance of physics-based and data-driven methods by combining the advantages of these two types of methods.

Building upon the existing lines of inquiry into hybrid modeling, the proposed approach combines physical insights from physics-based forecasting models and the ability to assimilate existing experimental data to capture the complex performance evolution of lithium-ion batteries. The proposed approach seeks to overcome a number of challenges associated with existing approaches, namely the lack of uncertainty quantification in hybrid methods, the difficulty associated with training recurrent neural networks, and the expert knowledge required to enforce physics constraints. To this end, the proposed method leverages the observation that battery discharge capacity is a monotonically decreasing phenomenon to enforce physical constraints in a neural network ensemble. Furthermore, the network is trained to act as a neural differential operator that learns the discharge capacity rate instead of learning the discharge capacity directly. This formulation provides a direct way to enforce constraints on time derivatives as well as a way to perform discharge capacity forecasting through direct time integration of the neural operator (avoiding the need for recurrent architectures). To learn the operator, an architecture inspired by DeepONet [152] is used, consisting of a branch network that encodes the input function space (i.e., the domain of battery cells) and a trunk that encodes the domain of the output function (i.e., the number of cycles). To capture the epistemic uncertainty associated with lack of knowledge of the exact battery degradation phenomenon, the network is formulated as a Bayesian neural network capable of expressing uncertainty over its own predictions. The resulting algorithm is a physics-constrained Bayesian neural network that provides high flexibility and improved accuracy for RUL prediction with respect to existing methods. Using the battery dataset presented

in the paper by Severson et al. [221] as an example, the proposed method was compared with a simplified physics-based degradation forecasting model and two data-driven prognostics models. The results show that the proposed physics-constrained neural network can provide more accurate and precise RUL estimation than the other methods using the same training data.

This chapter is organized as follows. Section 2.1.1 describes all the methods considered, including the proposed approach. Section 2.1.3 presents results and discussion for all the methods considered. Finally, section 2.1.7 provides concluding remarks and future research directions.

2.1.1 Predictive Approaches for Battery RUL Forecast

In this section, an implementation of the failure forecast model based on simplified physical laws and two data-driven failure prognostic methods are presented in Sections 2.1.1, 2.1.1, and 2.1.1 respectively. These three methods will be compared with this chapter’s proposed physics-constrained neural network method presented later in Section 2.1.2. In the following sections, capacity is used to refer to discharge capacity which is generally used to track capacity degradation. Capacity is always presented in Ampere-hour (Ah) units unless otherwise noted.

It should be noted that all three methods are distinct and, as a result, are able to take different types of inputs. In particular, the neural network approach is the most flexible and can incorporate a larger number of inputs than the other methods without dramatically sacrificing efficiency (e.g., the GPR methods require the computation of the kernel matrix which becomes expensive with more inputs). This flexibility is leveraged to include additional potentially useful information as input to the neural network.

A Physics-Based Failure Forecast Model

The Failure Forecast Model (FFM) form proposed by Voight [261] successfully models many geophysical (see [260]) and material-level failure mechanisms (see [260, 248, 191]) that share a common self-accelerating degradation characteristic as the system approaches failure. We observe that the inverse of the battery capacity evolution accelerates with time and can, therefore,

be modeled using the FFM. We assume a failure forecast model that has self-accelerating characteristics and perform Bayesian inference to predict the RUL and the uncertainty bounds in the prediction.

Let $C(t)$ denote the capacity of the battery at time (or equivalently, cycle) t . Let $R(t) = C(t)^{-1}$ denote the inverse of the discharge capacity. Since the discharge capacity is a monotonically decreasing phenomenon, the inverse of the discharge capacity is a monotonically increasing phenomenon. As per the general FFM model form, $R(t) = C(t)^{-1}$ is modeled by using a first order differential equation that takes the form

$$\frac{dR}{dt} = kR^\alpha, \text{ where } k > 0 \text{ and } \alpha > 1. \quad (2.1)$$

Here, the constants k and α are specific parameters for the physical process (discharge capacity degradation here). Using Eq. (2.1), the quantity $R(t)$ can be solved by assuming that $R(t = t_f) = R_f$, where, t_f is the time of failure, such that

$$R(t) = \left(R_f^{1-\alpha} + k(\alpha - 1)(t_f - t) \right)^{\frac{1}{1-\alpha}}. \quad (2.2)$$

The capacity failure threshold is defined as $C_f = R_f^{-1}$, which is assumed to be $C_f = 0.8$ in this work. The discharge capacity $C(t)$ is obtained as

$$C(t) = \left(C_f^{\alpha-1} + k(\alpha - 1)(t_f - t) \right)^{\frac{1}{\alpha-1}}. \quad (2.3)$$

The value $\alpha = 2$ represents a special linear case of the FFM, such that

$$C(t)|_{\alpha=2} - C_f = k(t_f - t). \quad (2.4)$$

The value of $\alpha < 2$ represents the case where the discharge capacity follows a convex down trend (curving upward along the horizontal axis). On the other hand, $\alpha > 2$ represents the case where

the discharge capacity follows a concave down trend (curving downward along the horizontal axis). As α increase, the value of $C(t)$ decays faster as the system approaches failure. Since discharge capacity is a monotonically degrading phenomenon with the rate of decay increasing over time, $\alpha \geq 2$ is of interest.

The goal here is to make a probabilistic estimate of the parameters k , α , t_f , and the noise parameter σ_l (standard deviation of the battery capacity data) using a portion of gathered capacity data $\mathcal{D}$. Let Θ denote the random vector consisting of the parameters to be estimated, i.e., $\{k, \alpha, t_f, \sigma_l\}$. The parameter vector can be inferred using Bayes theorem, such that

$$P(\Theta|\mathcal{D}) = \frac{P(\mathcal{D}|\Theta) \times P(\Theta)}{P(\mathcal{D})}. \quad (2.5)$$

In the equation above, $P(\Theta)$ denotes the prior belief of the parameter vector. Since $k > 0$, the prior for k is assumed to be a uniform distribution ranging from 0 to an arbitrarily large number (representing ∞). The prior for α is assumed to be uniform distribution ranging from 2 to 10. The prior to σ_l is chosen to be half normal, a weakly informative distribution as suggested by Gelman et al. [77, 76]. Since t_f is positive and greater than the current time, the prior for the time of failure is assumed to be a uniform distribution from the current time t to an arbitrarily large number (representing ∞). The term $P(\mathcal{D})$ in Eq. (2.5) is the evidence and is simply a normalizing quantity. Finally, the term $P(\mathcal{D}|\Theta)$ denotes the likelihood of observing data $\mathcal{D}$ given the parameter vector Θ . If a specific forward measurement model is adopted, a tailored likelihood function could be derived from that. In this chapter, the FFM (Eq. (2.3)) itself is used to model the likelihood function. To capture the heteroscedastic converging effect of the noise structure, the inference is performed in the logarithmic space $\tilde{\mathcal{D}} = \log(\mathcal{D})$. The likelihood of observing $\tilde{\mathcal{D}}$ given the parameters vector $\Theta = \{k, \alpha, t_f, \sigma_l\}$, denoted by $P(\tilde{\mathcal{D}}|\Theta)$, is modeled using a normal distribution with the standard deviation σ_l and mean μ_l , i.e., $P(\tilde{\mathcal{D}}|\Theta) = N(\tilde{\mathcal{D}}|\mu_l, \sigma_l)$, where the

mean of the likelihood function is given by

$$\mu_t = \log \left(\left(C_f^{\alpha-1} + k(\alpha-1)(t_f - t) \right)^{\frac{1}{\alpha-1}} \right). \quad (2.6)$$

To obtain the joint posterior distribution of parameters constituting the parameter vector Θ using Eq. (2.5), a sampling-based Markov Chain Monte Carlo (MCMC) approach is implemented using a Metropolis sampler within the PyMC3 Python package. Once the parameter vector is inferred using the recorded data $\tilde{\mathcal{D}}$, the future discharge capacity data $\hat{\mathcal{D}}$ (called the posterior predictive distribution) may be forecast using the samples of the inferred parameters run on the FFM model that was used to construct the likelihood function. Mathematically, $\hat{\mathcal{D}}$ is conditioned on the recorded data $\tilde{\mathcal{D}}$, such that

$$P(\hat{\mathcal{D}}|\tilde{\mathcal{D}}) = \int_{\Omega_{\Theta}} P(\hat{\mathcal{D}}|\Theta)P(\Theta|\tilde{\mathcal{D}}) d\Theta. \quad (2.7)$$

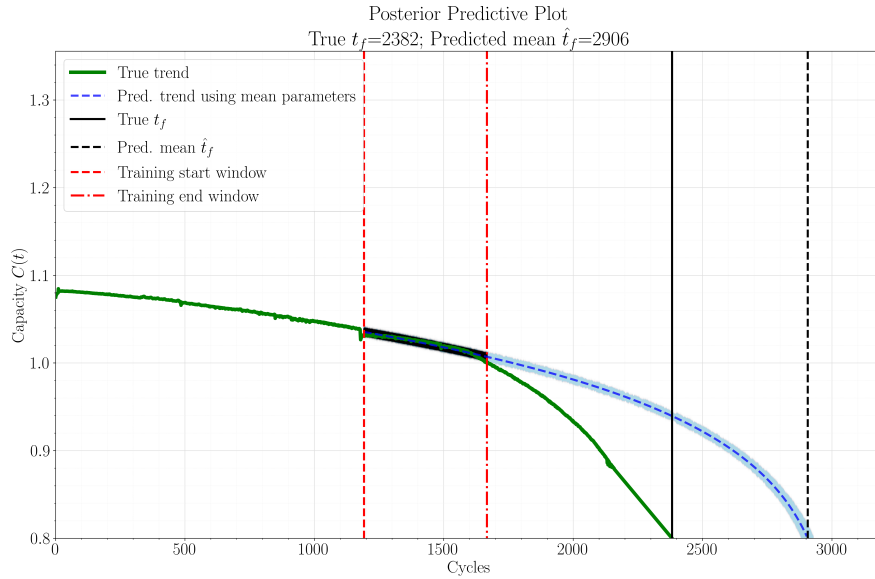


Figure 2.1. Illustration of FFM used to generate posterior predictive plot

Fig. 2.1 illustrates an example of a posterior predictive plot inferred using a window of data that forecasts the time of failure of the battery's capacity as well as yields uncertainty bounds

of the prediction. We observe that the prediction overshoots the true trend for the considered window. This is because the assumed failure model described in Eq. (2.3) does not fully capture the characteristics of battery capacity curves. It is designed for forecasting damage or fracture type of failure, where degradation notably accelerates towards the end. However, the capacity degradation of a battery behaves differently. Unlike the fracture type of failure, the slope of the capacity curve is almost constant towards the end of the battery's life. Consequently, the FFM model does not fully capture this degradation characteristic, resulting in an overshooting of the prediction until the data window comes closer to the failure threshold.

To predict the distribution of the time of failure (and by extension RUL), parameter estimations are performed using a moving window of recorded data because the data contained in the moving window will contain the latest stage information about the capacity degradation that informs the degradation curve. A data window size of 20% is used. Let $\hat{t}_f$ denote a realization of time, t denote the current instance of time, then the corresponding RUL denoted by $\hat{L}(\hat{t}_f)$ is given as

$$\hat{L}(\hat{t}_f, t) = \hat{t}_f - t. \quad (2.8)$$

We note that this is not a recursive approach since a particular data window does not affect the inferred parameters obtained by any other window and each window acts as an independent source of available data. For the purpose of demonstration, we assume non-overlapping windows spanning the entire time/cycle domain as shown later in Fig. 2.5 and 2.6.

Data-Driven Prognostic models

Two different types of data-driven prognostic models, including a Gaussian process regression (GPR)-based model and a similarity-based model, will be provided in this subsection. Both models use the run-to-failure data of batteries to build a degradation model for failure prognostics.

A GPR-Based method

Assuming that N run-to-failure data-sets of battery capacity with different lengths exist, denoted as $\mathbf{C}_i$, $i = 1, \dots, N$, where $\mathbf{C}_i = \{C_{i,j}, j = 1, \dots, T_i\}$ is the i -th capacity dataset with a length of T_i . In order to build a GPR model to capture the degradation pattern of the battery, a finite impulse response (FIR) filter with a Hanning window is first applied to the original capacity curves to remove high-frequency noise and outliers in the data and thus smooth the capacity degradation curves.

Since GPR model training will be computationally very slow for a large number data set and further suffers from the curse of dimensionality, a down-sampling step is performed to reduce the size of the data after the smoothing of the degradation curves. Denoting the run-to-failure data after down-sampling as $\tilde{\mathbf{C}}_i = \{C_{i,k}, k = 1, \dots, M_i\}$, $\forall i = 1, \dots, N$, where M_i is the length of the i -th capacity dataset after down-sampling, the finite difference of the capacity between two adjacent time steps is computed as

$$\Delta C_{i,k} = C_{i,k} - C_{i,k+1}, \quad i = 1, \dots, N; k = 1, \dots, M_i - 1, \quad (2.9)$$

where $\Delta C_{i,k}$ is the capacity degradation of the i -th battery from time step t_k to time step t_{k+1} after down-sampling. Note that the finite difference is computed by subtracting $C_{i,k+1}$ from $C_{i,k}$ since the capacity is decreasing over time.

To learn the underlying dynamics of the capacity degradation, a nonlinear autoregressive exogenous model (NARX) using the GPR model is constructed. NARX model is a nonlinear time-series modeling method that describe the response of a time series at the current time instant as a nonlinear function of the past values of the time series and current and past values of the driving factors (i.e., exogenous inputs). Various methods can be employed to model such a nonlinear functional relationship. When a GPR model is used to build this nonlinear function, the resulting model is called a GP-NRAX model [289]. The GP-NARX model in this chapter is

given by

$$\Delta\hat{C}_k = \mathcal{G}(\Delta C_{k-1}, \Delta C_{k-2}, \dots, \Delta C_{k-p}; \theta), \quad (2.10)$$

in which $\mathcal{G}(\cdot)$ is the GPR model with hyper-parameters θ , $\Delta\hat{C}_k \sim \mathcal{N}(\mu_{C,k}, \sigma_{C,k}^2)$ is the predicted capacity degradation at time step t_k which follows a Gaussian distribution with a mean of $\mu_{C,k}$ and a standard deviation of $\sigma_{C,k}$, and p is the number of lags in the NARX model. The hyper-parameters θ are estimated using the maximum likelihood estimate method based on the training data of $\Delta C_{i,k}$ obtained in Eq. (2.9).

For some situations, additional information from the run-to-failure data may be available. For instance, for the data used in the numerical example, the first 100 cycles of the discharge capacity curves are available. Denoting those additional parameters as ω and incorporating them into the GPR model, the GP-NARX model given in Eq. (2.10) can be rewritten as

$$\Delta\hat{C}_k = \mathcal{G}(\Delta C_{k-1}, \Delta C_{k-2}, \dots, \Delta C_{k-p}, \omega; \theta). \quad (2.11)$$

In the above equations, the optimal down-sampling rate and the number of lags p need to be determined through a group of validation data that is not used in the test. After the construction of the GP-NARX model, for a new battery with measured capacity data $C_{new,q}$, $q = 1, \dots, T$, where T is the current number of measured time steps, we first predict the battery capacity degradation of the new battery at time step t_{T+1} as

$$\Delta\hat{C}_{new,T+1} = \mathcal{G}(\Delta C_{new,T}, \Delta C_{new,T-2}, \dots, \Delta C_{new,T-p+1}, \omega; \theta). \quad (2.12)$$

Since $\Delta\hat{C}_{new,T+1} \sim \mathcal{N}(\mu_{new,T+1}, \sigma_{new,T+1}^2)$, we first generate N_{MCS} samples of $\Delta\hat{C}$ according to the Gaussian distribution using Monte Carlo simulation (MCS). Let the MCS samples be $\Delta C_{new,r}$, $r = 1, \dots, N_{MCS}$, the capacity at time step t_{T+1} is given as

$$\hat{C}_{T+1,r} = C_{new,T} - \Delta C_{new,r}, \quad r = 1, \dots, N_{MCS}, \quad (2.13)$$

where $\hat{C}_{T+1,r}$, $r = 1, \dots, N_{MCS}$ are the predicted MCS samples of battery capacity at time step t_{T+1} due to the uncertainty in the GPR prediction.

The above two equations (i.e., Eqs. (2.12) and (2.13)) are implemented recursively over time to forecast battery capacity over a long time period, and thereby allows one to predict the RUL of a new battery.

Similarity-Based Approach

The similarity-based method proposed by Wang et al. in Ref. [267] for data-driven prognostics is also employed in this chapter as a benchmark method for comparison. The basic idea of the similarity-based method is to match the degradation pattern of a new battery to that of the run-to-failure data in the library and thus estimate the RUL of the new battery based on the actual life of the existing units. The approach will be briefly summarized herein, and interested readers are directed to Ref. [267] for details. As shown in Fig. 2.2, a subset of the capacity data of the new battery is first extracted using a window size ΔT and denote the extracted data as $\mathbf{C}_s = \{C_{new,T-\Delta T}, \dots, C_{new,T}\}$. For a given run-to-failure capacity curve $\mathbf{C}_i$ in the training data, the time instant T_0^* that $\mathbf{C}_i$ has the highest similarity with $\mathbf{C}_s$ is identified by solving the following optimization model:

$$\begin{aligned} T_0^* &= \arg \min_{T_0} \{f(\mathbf{C}_s, \mathbf{C}_i(T_0))\}, \\ s.t. \quad & \\ T_0 &\in [0, T_{e,i} - \Delta T], \end{aligned} \tag{2.14}$$

where $\mathbf{C}_i(T_0) = \{C_{i,T_0}, \dots, C_{i,T_0+\Delta T-1}\}$ are the capacity data of the i -th capacity degradation curve starting from time T_0 with a window size ΔT , $f(\mathbf{C}_s, \mathbf{C}_i(T_0))$ is a loss function that quantifies the similarity of the two data-sets $\mathbf{C}_s$ and $\mathbf{C}_i(T_0)$, and $T_{e,i}$ is the end of life of the i -th battery obtained based on the capacity degradation curve. The loss function $f(\mathbf{C}_s, \mathbf{C}_i(T_0))$ could be defined in many different ways, such as likelihood function, Euclidean distance, or correlation

measures. In this work, the Euclidean distance is employed as the loss function.

The above optimization model basically moves the capacity degradation data of the new battery along the time axis to find the best match between the capacity degradation pattern of the new battery and that of the battery used for training (see the illustration in Fig. 2.2).

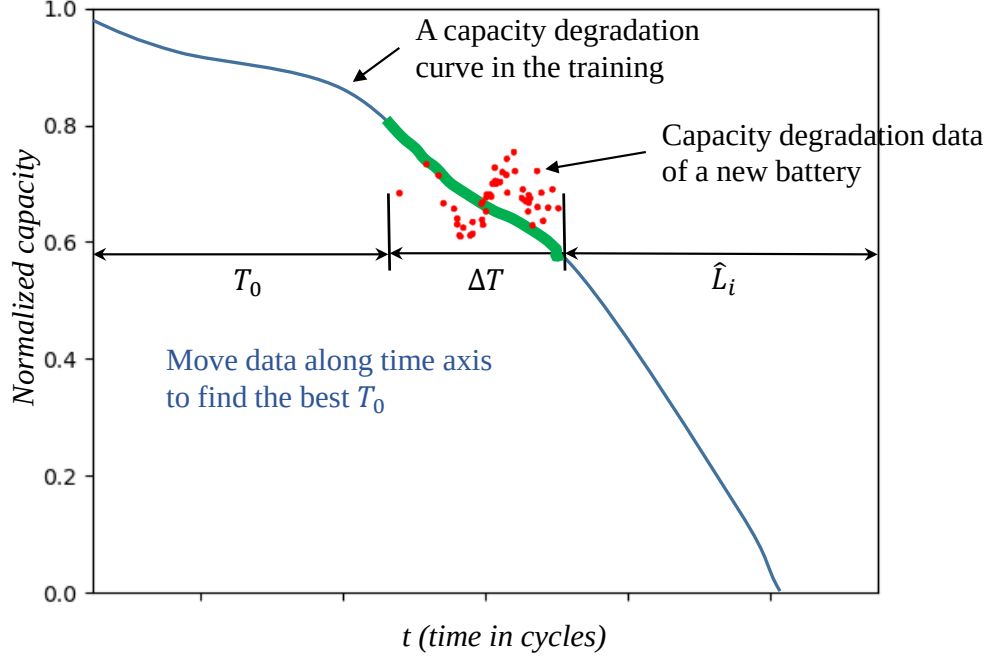


Figure 2.2. Illustration of similarity-based RUL estimation method. Note that the data and curve shown in this figure are for explanation purpose only. They are not real capacity data and curve.

After the identification of T_0^* , the similarity, S_i , between the degradation pattern of the new battery and that of the i -th battery for training, is given as

$$S_i = f(\mathbf{C}_s, \mathbf{C}(T_0^*)), i = 1, \dots, N. \quad (2.15)$$

The corresponding RUL estimate based on the i -th degradation curve is given by

$$\hat{L}_i = T_{e,i} - T_0^* - \Delta T, i = 1, \dots, N. \quad (2.16)$$

Finally, the $S_i, i = 1, \dots, N$, are used as weights and $\hat{L}_i, i = 1, \dots, N$ as corresponding RUL estimates, to implement a re-sampling of $\hat{L}_i$ based on the weights S_i to obtain the probabilistic distribution of the RUL estimate of the new battery. In this work, the re-sampling of $\hat{L}_i$ is performed using the particle filtering method.

2.1.2 Physics-Constrained Bayesian Neural Network

Neural networks have been shown to be universal function approximators [53, 103] as well as universal nonlinear operator approximators [152, 246]. Their extensive expressive power comes from performing a series of function composition operations. On its own, the individual functions that make up a neural network are often differentiable linear or simple nonlinear functions. The key aspect is that these operations are differentiable and thus, amenable to be used in conjunction with gradient-based optimization algorithms. However, because of the emergent complexity that neural networks exhibit, they can often overfit and produce non-physical behavior. To this end, physics constraints can be introduced into the network via penalty terms in the loss function [203]. The constraints introduced in this work will be described in the next section. Neural networks are particularly amenable for imposing Neumann-type constraints because gradients are easily computed with auto-differentiation. For these reasons, neural networks were chosen as the proposed approach.

Moreover, the present problem exhibits significant variance in the battery RUL. For this reason, it is desirable to opt for a Bayesian neural network that can model the epistemic uncertainty associated with the predictions. In this work, it is assumed that epistemic uncertainty is associated with the parametric part of the model and not the time-dependent portion. The rationale behind this choice is that there is model-form uncertainty associated with the parametric model due to not having access to direct state variables associated with the physical phenomena and instead relying on indirect measurements. The effects of this assumption could be explored in future work. The time-dependent trunk only takes time as input while the parametric branch of the network takes as input the following vector

$$\lambda = \{C_0, C_{20}, C_{40}, C_{60}, C_{80}, C_{100}, \mathbf{var}(Q_{100-10}), \int_0^{100} C dt\}. \quad (2.17)$$

This vector of input parameters seeks to capture the trend of the RUL curve in the first 100 cycles. Through an iterative process, it was determined that 5 sample points from the initial RUL curve were sufficient. In addition to these samples, it was determined that the area under the curve (AUC), computed by integrating the first 100 steps of the RUL curve, was also a useful predictor. In addition to these features, the authors of [221] found that the voltage discharge curves from early cycles correlated well with the final RUL value. For this reason, the variance of the difference between the voltage discharge curve at 100 and at 10 cycles is used as a predictor as well.

Bayesian Neural Network Overview

Unlike a traditional neural network which learns to predict point estimates, a Bayesian Neural Network learns the posterior distribution, $P(w|\mathcal{D})$, over the weights given an assumed prior distribution, $P(w)$ following Bayes theorem as

$$P(w|\mathcal{D}) = \frac{P(\mathcal{D}|w)P(w)}{\int P(\mathcal{D}|w')P(w')dw'}, \quad (2.18)$$

where w are the model weights to be estimated, $\mathcal{D}$ represents the training data, and $P(\mathcal{D}|w)$ is the likelihood of observing $\mathcal{D}$ for given weights w .

As described in [19], obtaining the exact predictive posterior distribution is intractable for neural networks of any practical size due to the challenge of computing the term in the denominator of Bayes theorem. A common approach for approximating the predictive posterior is to rely on variational inference, as described in [97, 79]. In this formulation, we begin by minimizing the Kullback-Leibler (KL) divergence between a variational posterior distribution $q(w|\mu)$ and the desired posterior distribution $P(w|\mathcal{D})$. The KL divergence term is defined as

$$\mathbf{KL}(q(w|\mu)||P(w|\mathcal{D})) = \int q(w|\mu) \log \left(\frac{q(w|\mu)}{P(w|\mathcal{D})} \right) dw, \quad (2.19)$$

where μ are the parameters of the variational distribution $q(w|\mu)$.

This formulation leads to the familiar problem of minimizing the variational free energy or maximizing the expected lower bound (ELBO) which consists of two terms: a KL distance between the variational distribution and the weights prior distribution (effectively acting as a complexity cost) and the negative log-likelihood term that effectively captures the reconstruction loss term. The total loss term is defined as

$$\mathcal{H}(\mathcal{D}, \mu) = \mathbf{KL}[q(w|\mu)||P(w)] - \mathbf{E}_{q(w|\mu)}[\log P(\mathcal{D}|w)], \quad (2.20)$$

This formulation is used to train the network by employing the Bayes by Backprop algorithm, as described in [19]. In practice, the loss function is approximated by employing Monte Carlo sampling such that the loss over a batch of size n is computed as

$$\mathcal{H}(\mathcal{D}, \mu) \approx \sum_i^n \log q(w^{(i)}|\mu) - \log P(w^{(i)}) - \log P(\mathcal{D}|(w^{(i)})), \quad (2.21)$$

where $w^{(i)}$ represents the i -th Monte Carlo sample drawn from the variation posterior $q(w|\mu)$. In this way, the network can be trained more or less as a traditional neural network using the tools available in Tensorflow and Tensorflow Probability [58]. Specifically, dense variational layers are used in Tensorflow to keep track of the variational posterior distribution.

Neural Network Architecture and Constraints

A Bayesian Neural Network architecture was used to model the battery discharge capacity rate. A Bayesian neural network, unlike a traditional network, treats its weights as random variables as described previously. As a result, the network functions as a probabilistic model where the weight distributions have been determined through variational inference (i.e., during training). The trained network functions as a neural differential operator that can be deployed

by performing time integration over the desired inference time range. The concept of a neural differential operator refers to a neural network that learns the mapping between state variables and the derivative of a field variable (e.g., the discharge capacity) and it has been used in previous work [183] by the authors. The network architecture was inspired by DeepONet [152] and is shown in Fig. 2.14. DeepONet leverages the concept of neural networks as universal operator approximators (based on the universal nonlinear operator approximation theorem introduced by Chen and Chen [44]). DeepONet consists of

- a branch network (i.e., $\mathcal{G}$) that encodes the discrete input function space and,
- a trunk network (i.e., $\mathcal{F}$) that encodes the domain of the output functions.

This architecture is desirable here because the objective is to construct the neural network as an operator that models the time derivative of the capacity discharge over a function space representing the different battery cells (parameterized by their initial capacity performance) and evaluated at an arbitrary point along their lifetime.

Consistent with Fig. 2.3, the trunk $\mathcal{F}$ processes the time dependence and the branch $\mathcal{G}$ handles the history or parametric dependence as

$$\frac{dC}{dt} \sim p \left(\frac{dC}{dt} | \lambda, t \right) = \mathcal{G}_{\Phi}(\lambda) \mathcal{F}_{\Psi}(t), \quad (2.22)$$

where Φ and Ψ are the weights of the branch and the trunk, respectively, and λ is the vector of input parameters related to the first 100 cycles of the discharge capacity curve. The branch learns the encoding over the discrete universe of battery cells (based on limited observations) while the trunk encodes the output domain, i.e., the number of cycles. In this case, both networks are exactly the same but they could be different if, for example, the branch needed to act as a functional over image data or high-dimensional functions.

The two networks consisted of 8 layers with 10 units, each with a nonlinear swish activation function. The swish activation function is defined as $f(x) = x/(1 + \exp(-x))$ and was

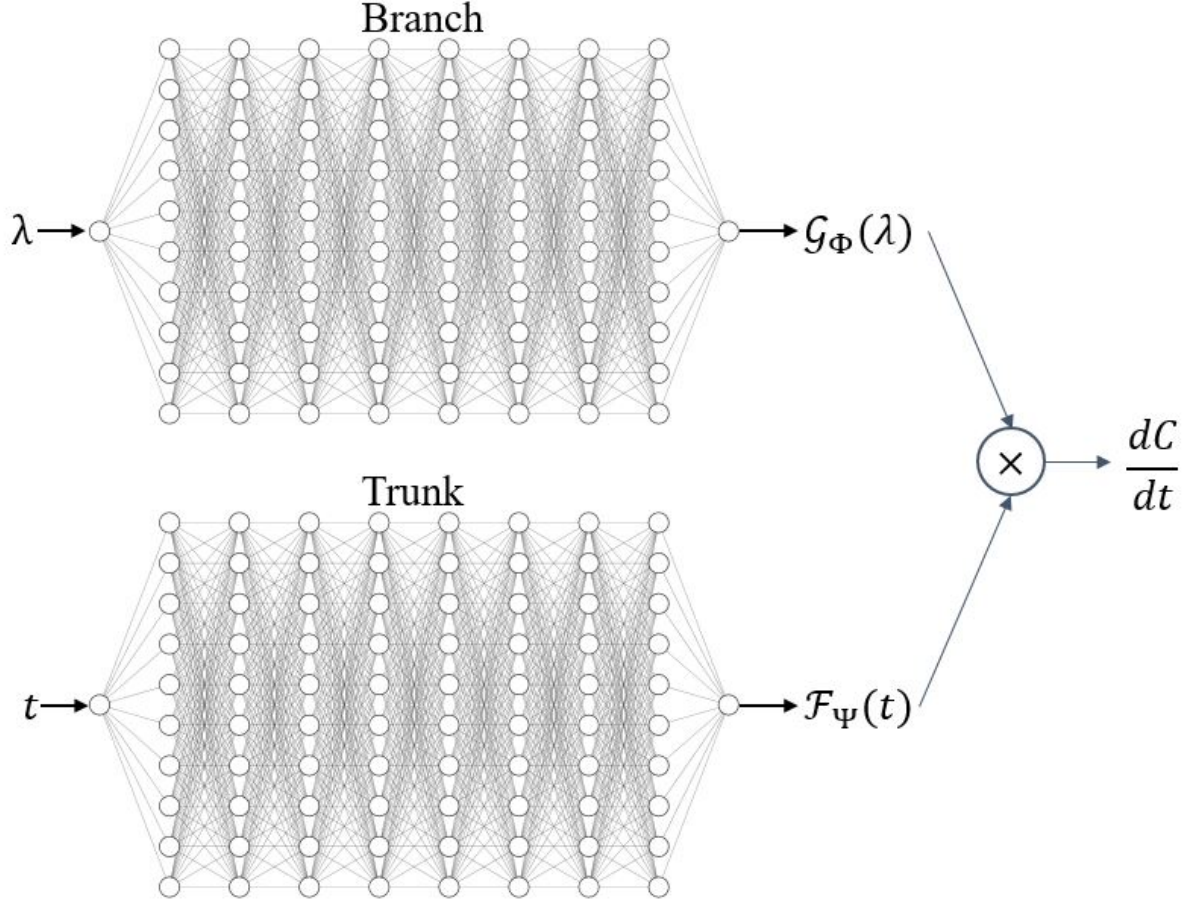


Figure 2.3. Schematic of general neural network architecture.

chosen because it is part of the Tauber-Wiener functions used in DeepONet [152] and was shown to behave well based on experiments performed by the authors. The last layer of the branch network has a variational dense layer with a trainable posterior approximate Gaussian mean field and an isotropic Gaussian prior. This variational layer is meant to introduce the capability to predict epistemic uncertainty. The assumption that epistemic uncertainty is only present in the branch network was made to avoid having to deal with random variables when enforcing physical constraints associated with the discharge rate. If the physical constraint could be expressed as a probability density function, it would be desirable to include random variables in the trunk as well, but this was not the case here.

As the network predicts the discharge capacity rate, the residual of a multi-step integration

evaluation is included to enforce mathematical consistency. The constraint is enforced via the following loss term,

$$\mathcal{L}_{num} = \left\| \sum_{j=0}^s \gamma_j C_{k+j} - h \sum_{j=0}^s \eta_j \left(\frac{dC}{dt} \right)_{k+j} \right\|_2^2, \quad (2.23)$$

where γ and η are coefficients from the Butcher tableau corresponding to the Adams-Bashforth linear multistep integration method, and h is the time step associated with the training data. The C_{k+j} term represents the $(k+j)_{th}$ time step in the C vector.

It is observed that the discharge capacity is a monotonically decreasing phenomenon. Based on this observation, a constraint enforcing negative self-acceleration is incorporated. The constraint is expressed mathematically by

$$\frac{d^2C}{dt^2} < 0. \quad (2.24)$$

As the network was constructed to model the discharge capacity, its second time derivative can be obtained by computing the gradient of the network via auto-differentiation. Furthermore, because the trunk is the only component with time dependence, the loss term associated with this physical constraint can be computed as

$$\mathcal{L}_{phys} = \left[\mathbf{ReLU} \left(\frac{d\mathcal{F}(t)}{dt} \right) \right]^2 \quad (2.25)$$

where **ReLU** is the rectified linear unit function which is non-zero only for positive values.

Lastly, the reconstruction loss was computed by evaluating the negative log-likelihood of the network with respect to the supervised discharge capacity rate as

$$\mathcal{L}_{recon} = - \sum_i^N \log p \left(\left(\frac{dC}{dt} \right)_i | \lambda_i, t \right). \quad (2.26)$$

where N is the number of samples in a batch. Note that this term corresponds to the third term in the variational loss term shown in Eq. (2.21). In addition the the reconstruction term, a

complexity term is also computed as

$$\mathcal{L}_{complex} = \sum_i^N \log q(w^{(i)}|z) - \log P(w^{(i)}), \quad (2.27)$$

where $q(w^{(i)}|z)$ is the variational distribution as described in Section 2.1.2 and $P(w^{(i)})$ denotes the non-trainable isotropic Gaussian prior distribution. Because a Gaussian distribution was also assumed for the variational posterior, the parameter vector z contains the mean and standard deviation of the distribution.

The total loss was obtained by adding the four individual loss components as

$$\mathcal{L} = \zeta \mathcal{L}_{recon} + \beta \mathcal{L}_{phys} + \kappa \mathcal{L}_{num} + \rho \mathcal{L}_{complex}, \quad (2.28)$$

where ζ , β , κ , and ρ are used to weight the contribution of each loss term and are determined manually by inspecting the relative magnitude of the different loss terms. The loss is minimized using the Adam optimizer with a learning rate set to 10^{-4} .

Once trained, the discharge capacity is obtained by time-integrating the capacity rate as

$$C(t) = \int_0^t \frac{dC}{dt} dt, \quad (2.29)$$

where $\frac{dC}{dt}$ is sampled from the Bayesian neural network as shown in Eq. (2.22). The time T_e at which the discharge capacity will reach its limit state (i.e., its end of life), can then be predicted from which the RUL, $\hat{L}(t)$, is estimated as

$$\hat{L}(t) = T_e - t, \quad (2.30)$$

where t denotes the cycle at which the RUL is being estimated.

2.1.3 Demonstration of Methodology

Dataset and pre-processing

The method was evaluated using the metrics described in section 2.1.4 by predicting the RUL of the battery dataset provided in [221]. For consistency, the same 80/20 training/test random split was used for all methods. The capacity degradation curves were pre-processed to remove outliers and extend them beyond their endpoint by linearly extrapolating them to zero discharge capacity by following the instruction given in Ref. [246]. Figure 2.4 shows the battery capacity as a function of cycles for the original data and after removing outliers.

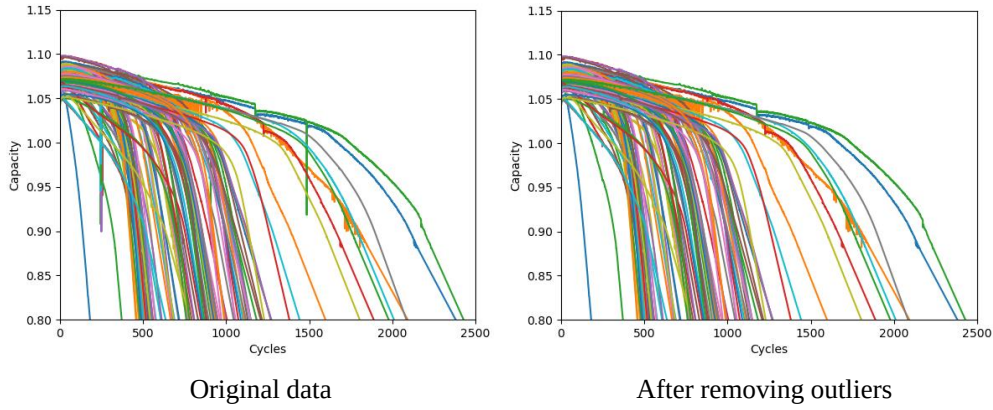


Figure 2.4. Capacity data used in this example (Before and after removing outliers).

2.1.4 Performance Metrics

To evaluate the performance of the algorithms, the following performance metrics were considered.

The error was computed simply as the difference between the actual RUL curve and the prediction as

$$\varepsilon_i = L_i - \hat{L}_i, \quad (2.31)$$

where i indicates the i -th test case.

The root mean squared error (RMSE) is a simple way to evaluate the accuracy of the prediction. The RMSE was computed by comparing the mean RUL predictions to the actual RUL curve as

$$RMSE = \frac{1}{N} \sum_i^N (\varepsilon_i)^2, \quad (2.32)$$

where the sum is performed over the N test cases.

The S-metric is based on the prognosis metric suggested by the authors of [216]. This score is an asymmetric metric that penalizes late predictions more heavily than early predictions. The metric is defined as

$$s = \begin{cases} \sum_i^N e^{\left(-\frac{\varepsilon_i}{a_1}\right)} - 1, & \text{for } \varepsilon_i < 0 \\ \sum_i^N e^{\left(\frac{\varepsilon_i}{a_2}\right)} - 1, & \text{for } \varepsilon_i \geq 0, \end{cases} \quad (2.33)$$

where s is the computed score, N is the number of test cases, and the parameters were defined as $a_1 = 1000$ and $a_2 = 1300$ following the guidance provided in [216] but scaling the parameters to be appropriate for the task at hand.

The F1-score was computed to test the robustness of the algorithms. To this end, the definition of true positives, false positives, and false negatives was taken from [215]. False positives were defined to be early predictions while false negatives were defined to be late predictions. These quantities were computed as

$$FP = \begin{cases} 1, & \text{if } \varepsilon_i > t_{FP} \\ 0, & \text{otherwise,} \end{cases} \quad (2.34)$$

$$FN = \begin{cases} 1, & \text{if } -\varepsilon_i > t_{FN} \\ 0, & \text{otherwise,} \end{cases} \quad (2.35)$$

where t_{FP} and t_{FN} are threshold values that are used to determine the acceptable region over

which the prediction is considered a false prediction. For this work, these two threshold were set to 50 cycles. Predictions that were within 50 cycles of the experimental data were considered true positives (TP). Lastly, the F1-score was computed as

$$F_1 = \frac{2TP}{2TP + FP + FN}. \quad (2.36)$$

2.1.5 Results

The four methods described in Section 2.1.1 were exercised to predict the battery RUL. Test 0 has an unusually high RUL while test 62 has one of the lowest RUL values of all the test cases observed. On the other hand, test 18 and 108 are typical of the battery RUL observed in most of the cells. For this reason, these four cases were chosen to exemplify the performance of each of the approaches. Fig. 2.5 and Fig. 2.6 show the mean RUL prediction and associated 95% confidence interval for all the methods considered.

For the physics-based FFM method, a linear and nonlinear version of the model is shown. The FFM is a nonlinear regressor guided by a failure forecasting model equation that accounts for varying rate of capacity degradation over time, as observed in the battery's decline in capacity as well as other failure mechanisms such as fracture and crack propagation. By using the FFM, the remaining useful life of the battery can be probabilistically inferred. A specific version of the general nonlinear FFM is a linear FFM, which uses a forecasting model that assumes a constant degradation rate throughout the battery's life, as determined by the data used for forecasting.

Because all four methods are different, a direct comparison of computational cost is not too meaningful but runtimes for computing a single RUL curve are reported for completeness in Table 2.4. For the GPR, Similarity-based, and the FFM methods, a single CPU was used for computation of the RUL curve. For the BNNet approach, a single GPU was used to train the network for 50,000 epochs without early-stopping criteria. As shown in Table 2.1, even though the physics-constrained BNNet approach takes a very long time for training, it is the most efficient one among the four methods for RUL prediction. Because the GPR-based method

implements a recursive uncertainty propagation scheme to quantify the uncertainty in the RUL prediction (see description in Sec. 2.1.1), it is the slowest. In addition, the similarity-based method and physics-based FFM method do not require any training time, due to the fact that these two methods use a sliding window to either check the similarity between the data of the window and the available run-to-failure data or to infer and update the model parameters of the FFM model using Bayesian method for RUL prediction. The computational time of these two methods for prediction is therefore mainly caused by similarity checking or Bayesian inference.

The results of the GPR-based method show that the GPR method can effectively quantify the uncertainty in the RUL predictions. As shown in Fig. 2.5 and Fig. 2.6, the true RUL curves lie within the 95 confidence bounds of the RUL prediction. The GPR-based method can accurately predict the RUL for tests 18, 62, and 108. During the construction of the GPR model, we only rely on the training data and did not leverage any physics knowledge. As a result, the prediction of the GPR model could have large prediction errors when the degradation pattern of the test is quite different from that of the training. For instance, the predicted RUL for test 0 from GPR-based method has large errors while it is pretty accurate for the other test cases. This is because the degradation of test 0 is unusually slow as mentioned above, which is quite different from that of tests used for training. The similarity-based method has a similar pattern as the GPR-based method. It can also effectively quantify the RUL prediction uncertainty and is pretty accurate for most cases. Very similar to the GPR-based method, the similarity-based method also has large prediction errors for test 0. This is attributed to the fact that both the GPR-based method and similarity-based method are interpolation-based methods. For interpolation-based methods, they could have good prediction accuracy when the prediction is within the training domain and may produce large errors if the prediction is outside the domain of the training data. The results of the FFM-based method shows that it overshoots the prediction in the early phase and the prediction gradually converges to the true value towards the end. This is because the FFM model is not suitable for forecasting battery capacity discharge because it is a nonlinear regression model designed for forecasting failure of a damage or fracture type of failure where

degradation accelerates towards the end. The capacity degradation of battery behaves differently and can be divided into three stages: initial, mid and final phase. In the initial phase, the capacity degrades at a constant rate, in the mid phase the rate of degradation accelerates and in the final phase, the slope becomes constant. Unlike the fracture type problem, the degradation rate of battery capacity does not change much in the final phase. As a result, the FFM model form does not fully capture this degradation characteristics, resulting in overshooting of the prediction until the data window comes closer to the failure threshold. We observe that the general nonlinear FFM, which is capable of capturing changes in slope, leads to better predictions than the linear FFM. In contrast, the BNNet approach is fully nonlinear and does not enforce a specific form of nonlinearity, making it more flexible while still able to enforce the monotonically decreasing discharge capacity. By leveraging this physics knowledge within the neural network training, the BNNet approach performs the best among the presented four methods for the four test cases.

A qualitative and quantitative comparison of the RUL predictions for all the methods is presented in the next section.

Table 2.1. Computational cost of a single RUL curve for each method.

Method	Training	Prediction
Physics-BNNet	5 hours	0.23s
GPR	0.42 hours	142s
Similarity	N/A	0.47s
Physics-FFM	N/A	60s

2.1.6 Comparison and discussions

The performance was evaluated by comparing the RUL curves to the actual data. Fig. 2.7 illustrates the distribution of errors calculated as shown in Eq. 2.31. A positive error indicates early end-of-life prediction while a negative error indicates late end-of-life. For this reason, a positive value can be considered conservative. As illustrated, the residuals from the physics-constrained BNNet are narrower and centered around zero, whereas the physics-based FFM and

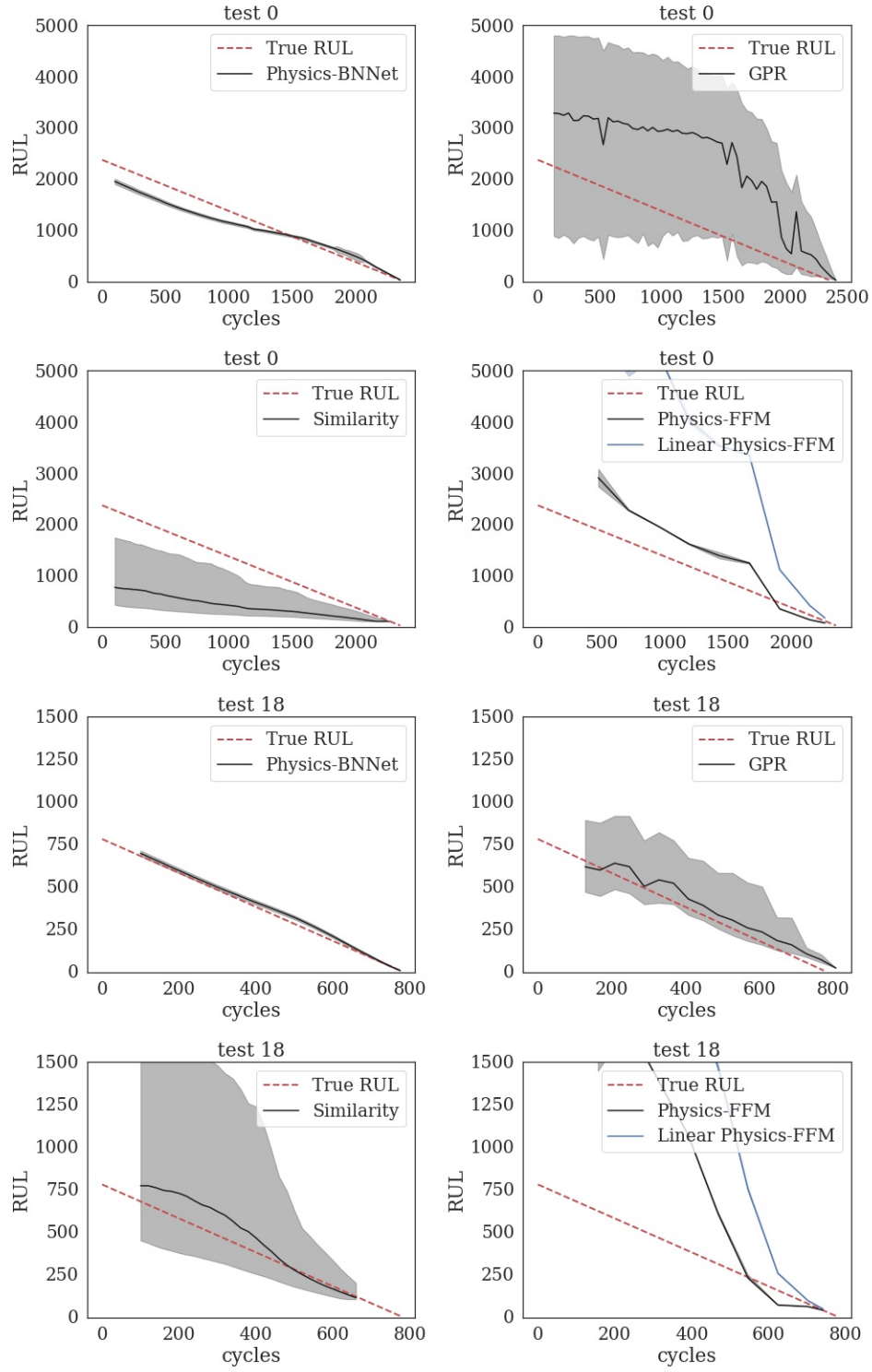


Figure 2.5. RUL predictions of test 0 and 18 for all the approaches considered.

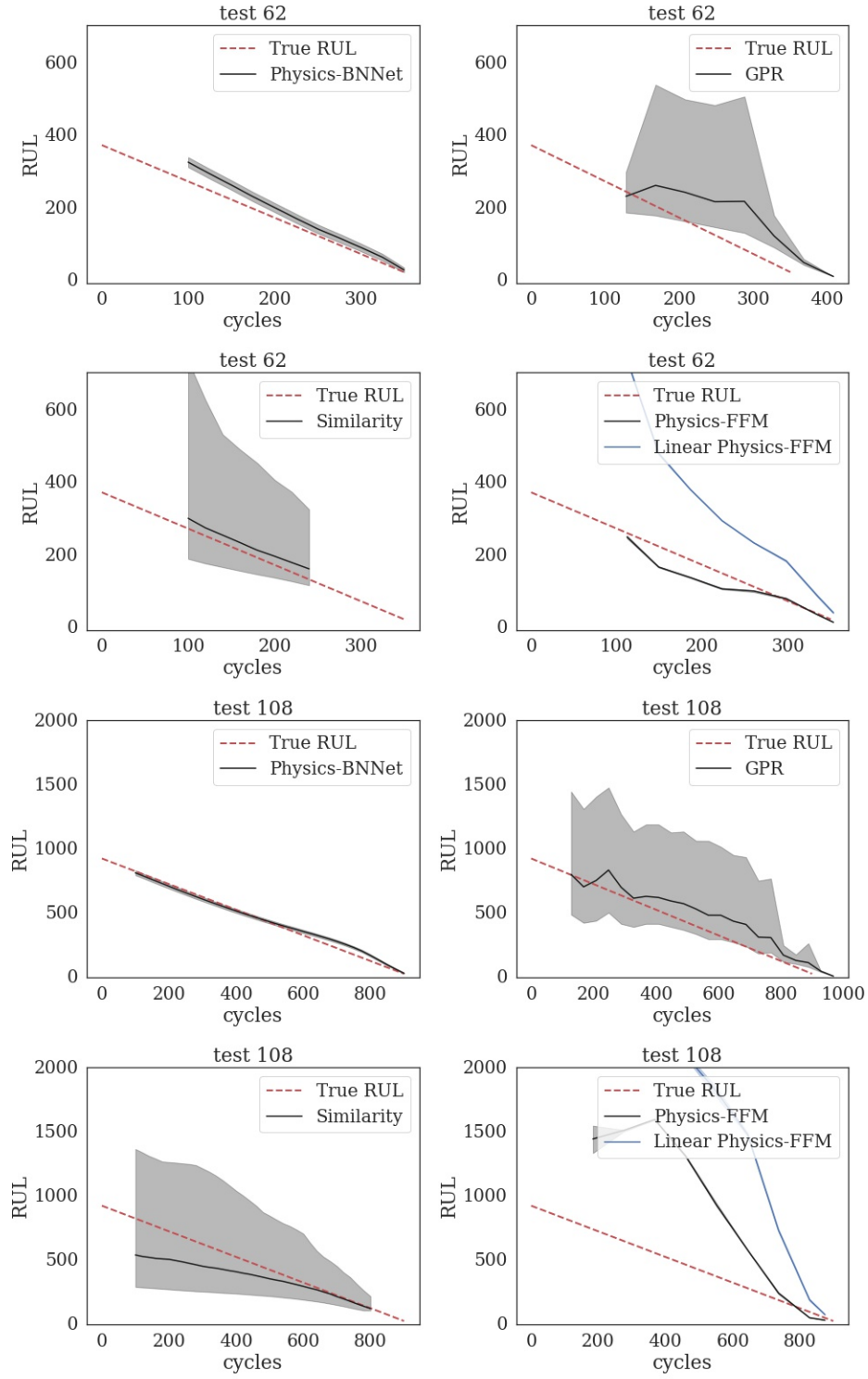


Figure 2.6. RUL predictions of test 62 and 108 for all the approaches considered.

the GPR predictions are generally broader. The similarity-based approach has similarly narrower distributions than the other GPR approach, but broader than the physics-constrained BNNet method. One of the trends that can be appreciated from this plot is the fact that the BNNet and similarity-based approach are generally conservative (i.e., early prediction of the RUL) whereas the other methods are unconservative (i.e., late predictions). To quantify the trade-off between precision and accuracy of these methods, the F1-score is reported in Table 2.2. As shown, the computed F1-score is consistent with the visual observations about the BNNet method being more precise while having a good recall rate.

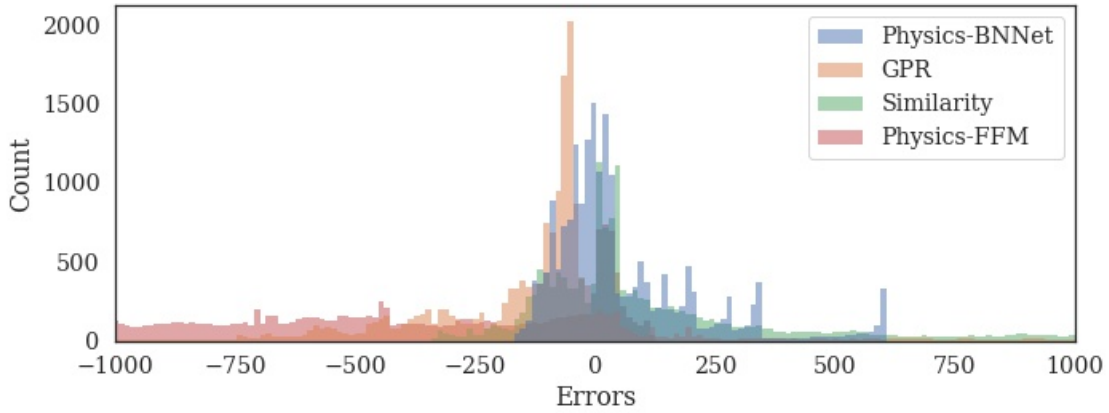


Figure 2.7. Histogram of error distributions over the test set for the different techniques considered.

Table 2.2. Summary of RUL Prediction Performance

Method	RMSE	S-metric	F1-score
Physics-BNNet	166	87	0.62
GPR	733	302	0.28
Similarity	425	101	0.46
Physics-FFM	5643	982	0.33

To illustrate the type of predictions obtained with the different methods considered, the RUL curves are plotted together in Fig. 2.8. Only four representative RUL curves are shown for easiness of visualization (these are the same curves shown in Fig. 2.5 and Fig.

2.6). As illustrated, both similarity-based approach and the GPR-based method have large 95% intervals, whereas the physics-based FFM and physics-constrained BNNet methods have generally narrower bounds. The similarity-based approach and the GPR-based method convey the uncertainty of their predictions through this large confidence interval which is useful because even when the predictions are not accurate, the model is able to express this uncertainty in the prediction. The physics-constrained BNNet method has significantly narrower bands but it is also more accurate, as judged by the performance metrics shown in Table 2.2. The BNNet's narrow 95% intervals seem to correspond to increased accuracy. The prediction uncertainty for time of failure using the FFM model, which is a nonlinear regressor, is dependent on the variability of the battery capacity data that is used to probabilistically infer the model parameters for the FFM regressor equation. In the case of battery capacity curves shown in Fig. 2.4, the variability is low, leading to a narrow prediction interval for the time of failure and the remaining useful life (RUL). Additionally, the prediction uncertainty is highest in the early phase of the prediction as uncertainty propagates during extrapolation for future capacity, and becomes narrower as the data window nears the failure threshold (see Fig. 2.1). As mentioned, quantitative performance metrics are shown in Table 2.2. The metrics show a consistent trend, with the physics-constrained BNNet method being the most accurate followed by the similarity-based approach method, then the GPR-based method, and lastly the physics-based FFM.

As the BNNet exhibits superior accuracy, its performance merits further exploration. One way to evaluate the performance of the proposed method is to perform hypothesis testing. We obtain the predicted RUL, $\hat{L}$ from the probabilistic model as described in section 2.1.2. We can then test $H_0 : \hat{L}_\mu = L$ versus $H_1 : \hat{L}_\mu \neq L$, that is, whether the actual RUL curve matches the mean of the samples of $\hat{L}$ generated from the physics-constrained BNNet. To that end, a T-test was performed by generating 500 samples from the BNNet model, and then computing the RUL curve and comparing it to the experimental data. The test was performed at each cycle along the RUL curve to obtain a distribution of T-tests. The proportion of times the null hypothesis was accepted (i.e., the proportion of cases for which the experimental mean agrees with the predicted

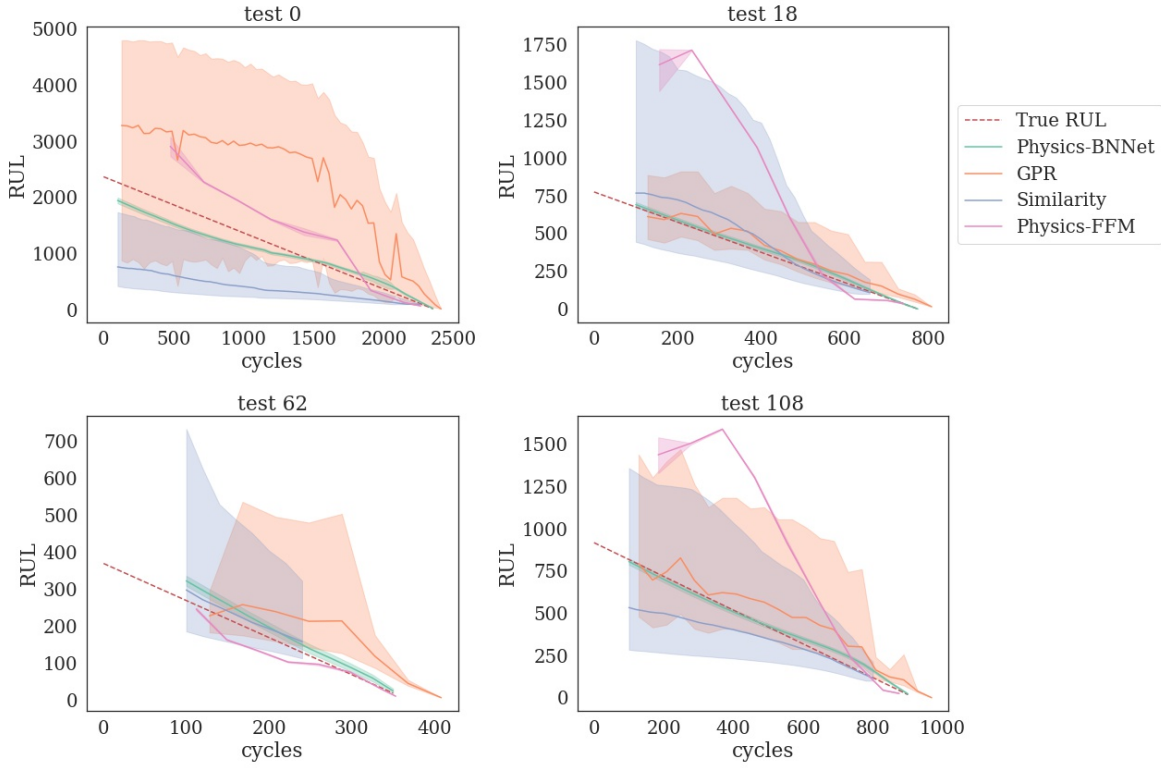


Figure 2.8. RUL curve predictions for four representative cells.

mean) with a 95% confidence level (two-sided test with p-value less than 0.05) is plotted in Fig. 2.10 for all the test cases. As illustrated, the lowest acceptance rate is 90% which indicates that the distribution sampled from the BNNNet generative process agrees well with the experimental data most of the time. To further illustrate this result, Fig. 2.9 shows 500 random discharge capacity curve samples generated by the BNNNet plotted alongside the actual experimental curve (shown in red). The prediction of each curve takes on average 0.23 seconds. Note that the uncertainty does not grow significantly with time because the predictions are not recursive due to the network learning the capacity discharge rate and forward computation achieved through time integration (with the RK45 integrator).

The superior performance of the BNNNet compared to the other approaches may be attributed to the fact that the BNNNet approach combines many of the attributes of the other methods. Due to the Bayesian architecture of the BNNNet method, this approach is similar to GPR,

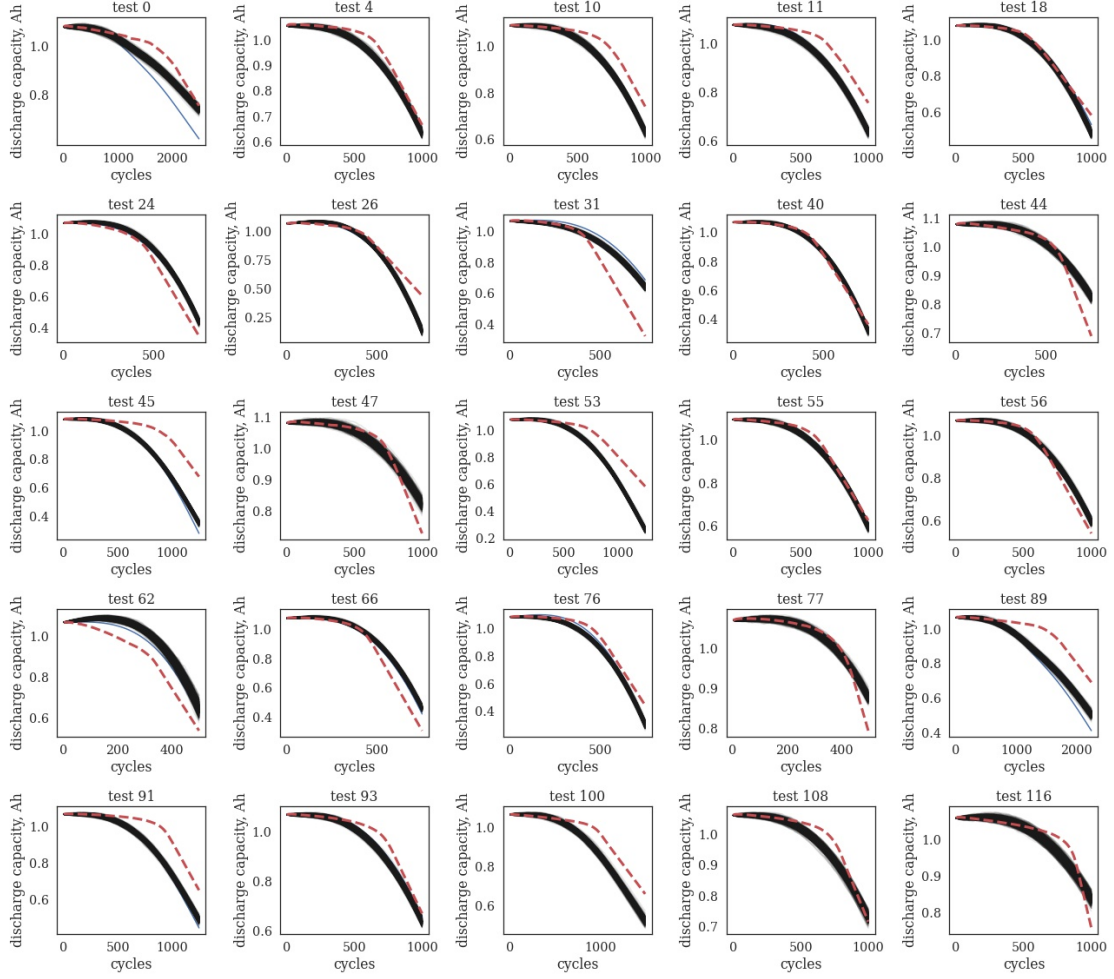


Figure 2.9. Random discharge capacity curve samples generated by the BNNNet (black) and actual experimental curve (dashed red).

as explored in [134] which allows the BNNNet to generate good predictions even when the data exhibit variability. Furthermore, the physics constraints of the BNNNet approach were inspired by the constraints utilized in the FFM approach, namely the fact that the battery degradation process exhibits monotonic behavior and thus can be constrained to have a concave down trend through its second derivative. In addition to these features, the BNNNet algorithm incorporated ideas from operator inference in its architecture which presumably increased its expressiveness. All of these features may have contributed to the good performance of the BNNNet method.

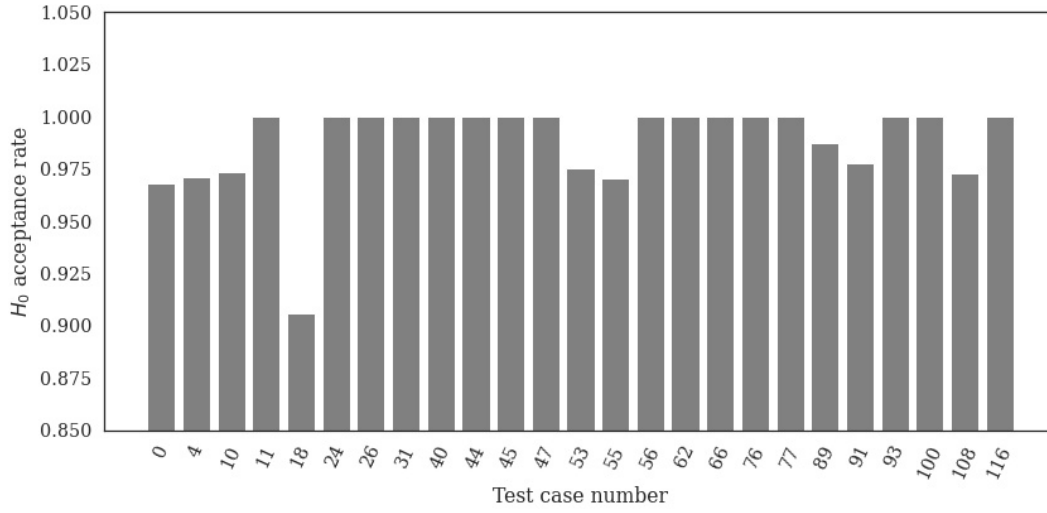


Figure 2.10. Proportion of cycles for which the null hypothesis $H_0 : \hat{L}_\mu = L$ was accepted.

2.1.7 Conclusions

This section presented a comparison of four distinct approaches to perform RUL prediction of a lithium-ion battery dataset. As described in the original paper ([221]), the battery dataset introduces several challenges due to the broad variability in capacity discharge trends for nominally identical battery cells. For this reason, this work was focused on the application of probabilistic methods that can account for the variability inherent in this dataset. Three of the four methods considered are well-established techniques in the field of RUL forecasting. It was shown that the FFM approach was too restrictive in the assumptions it makes about the degradation trends, and that it may not be appropriate to capture the physics of the battery degradation phenomenon. On the other hand, the data-driven approaches (GPR and Similarity-based) lacked the physical constraints to achieve high accuracy but they were able to express this epistemic shortcoming via large uncertainty bounds. Lastly, the BNNet approach combined features from the physics-based and data-driven approaches to achieve higher accuracy while predicting confidence intervals that were generally wide enough to capture the behavior of the actual dataset. The BNNet outperformed the other approaches in all the metrics considered which implies that the BNNet method is more robust, precise, and accurate than the other methods.

Future opportunities for research include extending the network architecture to be recurrent or memory-aware for cases in which the physical phenomenon considered is not monotonic and to include additional physical constraints by leveraging insights from physical modeling and simulation.

2.1.8 Preview of Section 2.2

Section 2.1 introduced the use of a data-driven approach to predict remaining useful life of Lithium-ion batteries. This approach consisted of a Bayesian neural network that combined the concept of ordinary differential operators with physics constraints that enforced self-accelerating degradation of the battery discharge capacity. Section 2.2 continues with this theme by applying a similar (but distinct) architecture to a pitting corrosion problem. In contrast with the battery dataset, the pitting corrosion problem deals with (simulated) images of corrosion patterns, therefore relying on a convolutional architecture instead.

2.1.9 Remarks

Portions of this dissertation have been published or are being considered for publication. Section 2.1 is composed of the following first authored publications:

David A. Najera-Flores, Zhen Hu, Mayank Chadha, and Michael D. Todd. A physics-constrained bayesian neural network for battery remaining useful life prediction. *Applied Mathematical Modelling*, 122:42–59, 2023

2.2 Applications to corrosion morphology prediction in civil structures

Pitting corrosion under stress is a common type of degradation in civil infrastructure [151, 87]. Forecasting the damage evolution of pitting corrosion over time is essential for maintenance planning and life-cycle cost optimization for any structure where corrosion-induced consequences are important. Specifically, one limit state is defined as the corrosion-induced cracking initiation

within the time of interest. The curvature information from corrosion morphology, together with stress condition, is utilized in the crack initiation criterion based on fracture mechanics. The detailed information can be found in Ref. [198]. Damage limit state forecasting may be typically performed using some form of a data-driven method, a physics-based approach, or a combination thereof [246, 247]. For large civil infrastructures such as the miter gates considered in this work, data that indicate degradation caused by pitting corrosion are usually unavailable. Physics-based corrosion damage prediction using high-fidelity computer simulations provides a promising way to overcome this challenge.

The operational and environmental loading (demand) play a significant role in corrosion evolution over long-term life cycle operations [206]. In order to account for these factors in physics-based simulation of pitting corrosion growth, different kinds of computational models have been proposed such as the finite volume method (FVM) [218], cellular automata (CA) techniques [233], peridynamic (PD) formulations [282], and phase-field (PF) models [42]. Among them, the phase-field method has the advantage of having the flexibility to couple the mechanical stress into the electrochemical equations to simulate the corrosion growth under mechanical stress [198].

When physics-based models such as the phase-field model are employed for failure forecasting, various sources of uncertainty (e.g., material properties, load conditions, etc.) in the simulation model need to be considered [106]. Proper consideration usually requires many (thousands or more) evaluations of the high-fidelity simulation model, particularly if a Monte Carlo simulation (MCS)-based method is adopted. However, the phase-field simulation is computationally very expensive due to multiple coupled nonlinear partial differential equations (PDE). Advanced uncertainty quantification or reliability analysis methods are therefore usually needed to tackle the computational challenge. Among existing uncertainty quantification or reliability analysis methods, MCS based on surrogate modeling is one of the most widely used. In such a strategy, a computationally cheaper yet accurate surrogate model is first constructed to replace the computationally expensive computer simulation models. MCS is then conducted

using the surrogate model to quantify the distribution of features of interest (e.g., corrosion depth, shape) or predict the probability of failure (limit state analysis) [105].

Multiple approaches have been made in recent years to build surrogate models emulating phase-field simulations. For instance, Shen et al. [228] used the least square regression (LSR) and the back-propagation neural network (BPNN) to produce an analytical expression for the breakdown strength based on phase-field simulation data. Eduardo et al. [56] utilized k-nearest neighbor (k-NN) and artificial neural networks (ANN) algorithms to detect the presence and location of failure based on simulation data. For time-dependent processes, Montes et al. [167] applied a long-short-term (LSTM) network to perform microstructure evolution predictions as the surrogate model of the phase-field simulation. The principal component analysis (PCA) technique was applied for dimension reduction; however, the reconstructed microstructure from the LSTM-trained surrogate model had a considerably noisy boundary. Nevertheless, in the corrosion simulation, the boundary between the corroded material and the solution is of main interest. Qian et al. [198] implemented a convolutional neural network with Gaussian process regression imbued with a nonlinear autoregressive exogenous structure (CNN-GP-NARX) network. In that work, a CNN-Autoencoder is trained to map between the high-dimensional image space and the low-dimensional latent space. The mean values of prediction are accurate, but there are some spurious corrosion shapes because of the uncertainty in the prediction. Even though these data-driven surrogate models show great potential in overcoming the computational cost for probabilistic physics-based failure forecasting, their accuracy may be significantly affected by the training data, and the prediction sometimes may violate physical law due to the limited number of training data.

The goal of this work is to build a physics-constrained machine learning model as the surrogate model for the phase-field stress corrosion simulation. The scope of this section is mainly about the corrosion morphology prediction, which can provide curvature information for corrosion-induced cracking initiation limit state. The direct prediction of corrosion morphology provides richer information for corrosion failure prognostics than the conventional approaches

which are usually based on highly abstract features, such as corrosion depth, volume loss, etc. To accomplish this goal, a convolutional variational autoencoder (CVAE) is designed to reduce the dimensionality of the corrosion images. Then, a Bayesian Latent Space Time Evolution Network (BLSTEN) is implemented to model the pitting corrosion evolution in the latent space. The CVAE model and the BLSTEN model are co-trained together to achieve good accuracy in the surrogate modeling. The corrosion rate is constrained by incorporating a physics term in the loss function in the proposed BLSTEN-VAE network. The results, when compared to a parallel method without physics-informed constraint, show that the proposed method is more accurate and robust. In addition, the trained BLSTEN-VAE network provides insight into the physical process it learned from data by inspecting its disentangled latent space. Furthermore, the sensitivity of the corrosion shapes to the environmental parameters is studied.

The remainder of this chapter is organized as follows: Section 2.2.1 introduces pitting corrosion damage in the infrastructure and phase-field simulation, as well as the previous GP-based surrogate model. Section 2.2.4 describes the architecture and formulation of the proposed network. Finally, Section 2.2.5 presents the results and the discussion of the proposed method.

2.2.1 Pitting Corrosion-Induced Failure of Large Civil Infrastructure

Pitting corrosion is a localized surface degradation form concentrating on a confined area. It is considered one of the most destructive forms among others [4]. The influence of pitting corrosion on structural integrity is difficult to detect at early stages. However, it can lead to material loss or induce cracking when combined with mechanical stress [126], resulting in abrupt structural failure. Numerous failures of infrastructure have occurred because of pitting corrosion; for example, a pre-stressed concrete bridge in Slovenia failed because of the corrosion on prestressing cables [256], the roof of an indoor swimming pool in Switzerland collapsed due to corrosion on multiple stainless steel elements [271], and the partial collapse of the Berlin Congress Hall happened because of corrosion in the tendons bearing for the roof arch [92].

Pitting corrosion is a complex process influenced by various factors including environ-

mental parameters, temperature, surface condition, metallic composition, among others [4]. In large civil infrastructure, mechanical stress also plays a significant role in corrosion evolution during long service periods. It is difficult to gauge pitting corrosion accurately considering the variability in every factor. Nevertheless, most corrosion growth models used for failure forecasting are empirical formulas [118, 235, 285, 35]. Most of these formulas cannot take the multiple factors mentioned above into consideration as they lack physical insight into the actual degradation process. In addition, empirical data required for empirical models for certain categories of infrastructure might not be available or exist, such is the case for the miter gates shown in Fig.2.11. Therefore, the physics-based pitting corrosion growth model plays a vital role in simulating and predicting corrosion growth by considering multiple factors.



Figure 2.11. A miter gate in the dewatered state.

2.2.2 Phase-Field Pitting Corrosion Simulation

The phase-field (PF) method has been actively applied for corrosion simulation [234, 158, 142]. The PF method shows competence and flexibility in simulating corrosion evolution [158]. The phase transition is described with a continuous variable ξ instead of a discrete phase transition. Multiple PDEs are coupled together to describe the phase change considering ionic concentration and diffusion. The mechanical stress influence on the generalized chemical potential can also be included in the PF model, as shown in the last term of Eq. (2.37) and coupled with other PDEs [198].

$$\mu = \mu_t + \mu_{el} + \mu_{gr} + \mu_{me}, \quad (2.37)$$

where μ is the generalized chemical potential, consisting of thermal potential μ_t , electrical potential μ_{el} , gradient potential μ_{gr} , and mechanical potential μ_{me} . The mechanical stress term μ_{me} is then coupled into the reaction equation in the activation overpotential term η_a

$$\eta_a = \Delta\phi - \frac{\mu_{M^{n+}} + n\mu_e - \mu_M + \mu_{me}}{nF}, \quad (2.38)$$

where $\mu_{M^{n+}}$, μ_e and μ_M are the activities of components, and $\Delta\phi$ is the interfacial potential difference. The reaction equation for the phase transition can be then written as the following

$$\frac{\partial \xi}{\partial t} = -L_\sigma (g'(\xi) - \kappa \nabla^2 \xi) - L_\eta h'(\xi) \left(\exp \frac{(1-\alpha)\eta_a}{RT} - \bar{c}_+ \exp \frac{-\alpha n F_r \eta_a}{RT} \right), \quad (2.39)$$

where L_σ represents the interfacial mobility and L_η represents the reaction coefficient which can be calibrated by the experimental data, R is molar gas constant, T is the temperature, and F_r is Faraday's constant. The detailed derivation may be found in [198]. This technique was implemented in that work on a two-dimensional plate with mechanical stress applied on one side of the plate and the pitting corrosion grows from the top center area. The simulation was run

for 200 time steps which corresponds to approximately 9.5 years (or 17.36 days per time step). Different shapes of pits are simulated under different stress conditions as shown in Fig. 2.12. The detailed relation between the stress and the corrosion growth is discussed in detail in [198].

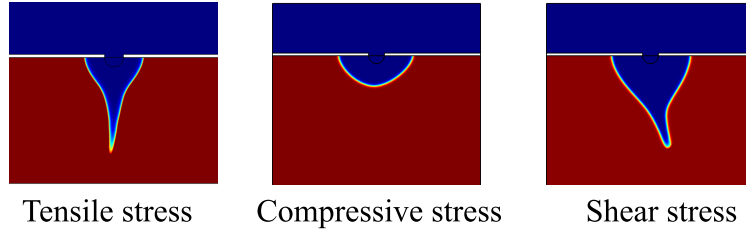


Figure 2.12. Pit shapes under different stress conditions.

Executing such a PF model requires solving multiple coupled PDEs, which requires substantial computational cost; one simulation may take more than thirty minutes [198]. In most probability-based failure analyses, the model needs to be run with different inputs on the order of least thousands of times. Efficient surrogate models (emulators) are preferred to replace the time-consuming simulation while maintaining acceptable accuracy compared to the full physics-based PF model evaluation.

2.2.3 Surrogate Modeling for Physics-Based Pitting Corrosion Forecasting

A machine learning-based surrogate model was proposed in previous research [198] to accelerate the simulation process as illustrated in Fig. 2.13. Similar to the dynamic PF simulation, the current pit shape, two stress parameters, and two environmental parameters are the inputs, and the pit shape of the next time-step is the output. Since the time-dependent prediction for high-dimensional features requires large amounts of training data, the time-dependent prediction is performed in the latent space after dimension reduction. A convolutional neural network (CNN) autoencoder is employed to reduce the dimension of the pit shape data from thousands to six dimensions in the latent space.

To perform the time-dependent prediction at any time step, a short-term Gaussian Process (GP) model together with a long-term GP-based nonlinear autoregressive network with exogenous inputs (NARX) are integrated. The short-term GP model generates the corrosion shapes of the first five time steps based on the mechanical load and the environmental factors $\theta = [\sigma, \tau, L_\eta, D^e]$, which are the normal stress, shear stress, reaction constant, and diffusion coefficient. Then, the GP-NARX model is responsible for recursively predicting the next time step based on the corrosion shapes of five previous time steps together with θ . Finally, the low-dimensional features in the latent space of the next step are transformed back to the image space with the CNN-Autoencoder structure mentioned above (see Fig. 2.13).

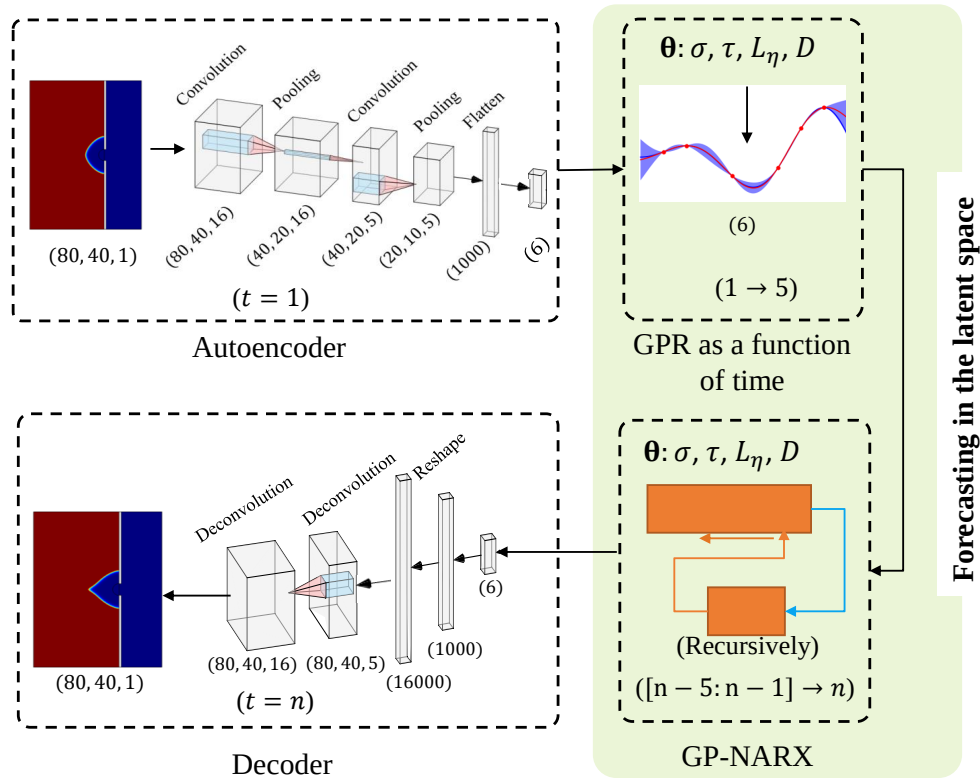


Figure 2.13. Architecture of CNN-GP-NARX surrogate model.

This CNN-GP-NARX surrogate modeling method has shown success in accelerating the simulation of the computationally expensive PF model for corrosion prediction. However, it was

observed that the prediction accuracy of the surrogate model suffered in some cases, as evidenced by predicted corrosion shapes that sometimes violated physical law (irreversibility). In order to address these issues, in the next section, we propose a physics-constrained machine learning method. The proposed physics-constrained method will be compared with the CNN-GP-NARX method in section 2.2.5.

2.2.4 Physics-Constrained Machine Learning as A Surrogate Model for Corrosion Failure Forecasting

A physics-constrained ensemble of neural networks is proposed in this section as a surrogate model to predict the pitting corrosion evolution over time. As shown in Fig. 2.14, the ensemble had two main components: a convolutional variational autoencoder (CVAE) that reduced the dimensionality of the pitting corrosion images, and a multilayer perceptron network (i.e., BLSTEN) that modeled the time evolution of the corrosion. The machine learning components were implemented using Tensorflow 2.10.0 and Tensorflow Probability 0.18.0. Multiple loss terms including a reconstruction loss, a latent space disentanglement loss, a BLSTEN loss, and physics loss are combined together to co-train the CVAE and BLSTEN networks, in order to ensure the effectiveness of the training and to incorporate physical laws into the training of the neural networks. The following subsections explain the models and different loss terms in detail.

Convolutional Variational Autoencoder

The first network is a CVAE designed to reduce the dimensionality of the corrosion images, represented by ξ . The encoder $q_{\Phi}(\mathbf{z}|\xi)$ consists of three 2D convolutional layers with a kernel size of 3, stride of 2, and 9 filters. The swish activation function is used for all the layers. The output from the 2D convolutional layers is then flattened and passed to a dense variational layer with a trainable posterior mean field. The output of the encoder is a random latent vector $\mathbf{z}$

$$\mathbf{z} \sim q_{\Phi}(\mathbf{z}|\xi), \quad (2.40)$$

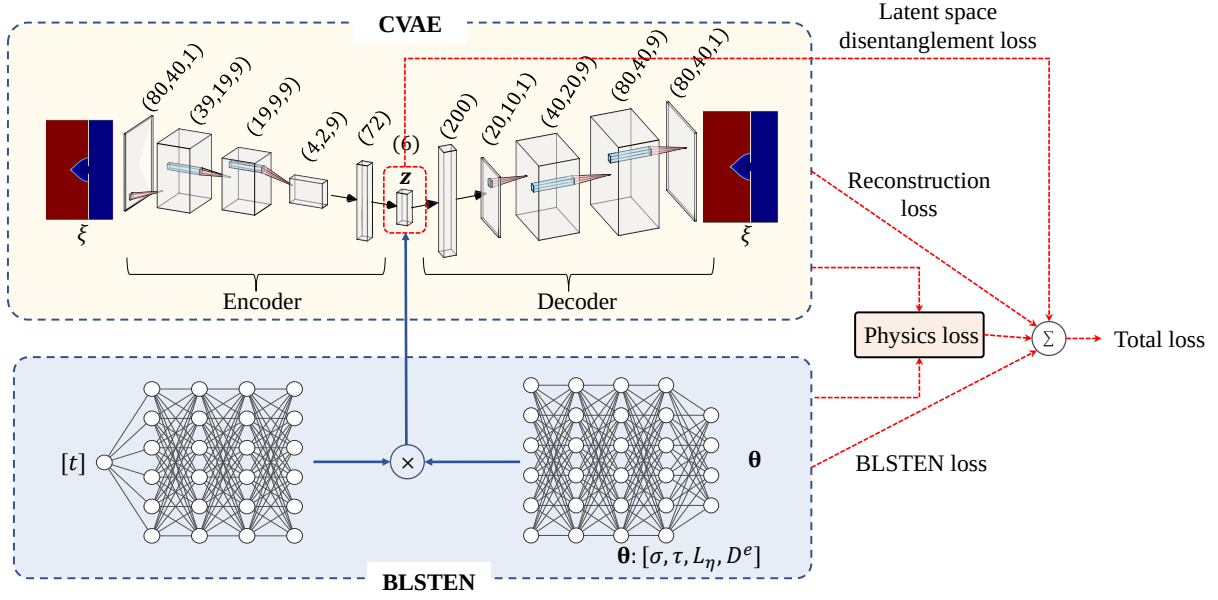


Figure 2.14. Schematic of the proposed general neural network architecture.

where Φ represents the encoder weights.

The second component of the autoencoder architecture is the decoder $p_{\Psi}(\xi|\mathbf{z})$ which takes the latent vector and maps it back to the full original dimension. The decoder consists of three 2D convolutional layers with a kernel size of 3, stride of 2, and 9 filters, except for the last one which had a single filter to match the original corrosion image. The output of the decoder is a matrix corresponding to a reconstruction of the original image

$$\hat{\xi} \sim p_{\Psi}(\xi|\mathbf{z}), \quad (2.41)$$

where Ψ represents the decoder weights.

Following this formulation, the loss function for the CVAE model corresponds to the negative evidence lower bound (ELBO), given by

$$\mathcal{L}_i^{\text{VAE}} = -\mathbf{E}_{\mathbf{z} \sim q_{\Phi}(\mathbf{z}|\xi_i)} [\log p_{\Psi}(\xi_i|\mathbf{z})] + \beta \mathbf{KL}(q_{\Phi}(\mathbf{z}|\xi_i) || p(\mathbf{z})), \quad (2.42)$$

where the subscript i represents the loss associated with the i -th training sample, $\mathbf{KL}(\cdot)$ is the

Kullback–Leibler (KL) divergence, $p(\mathbf{z})$ is assumed to be a spherical multivariate standard distribution, and β is a parameter used to weigh the contribution of the KL divergence term. The first term in Eq. (2.42) represents the reconstruction loss and the second term controls the level of latent space disentanglement, as described in [32]. As shown in Fig. 2.14, this loss term will be combined with other loss terms to co-train the CVAE along with the BLSTEN model (see the detailed description in the subsequent section).

Bayesian Latent Space Time Evolution Network

A separate Bayesian multilayer perceptron network (known as the Bayesian Latent Space Time Evolution Network or BLSTEN) is used to model the time evolution of the pitting corrosion phenomenon in the latent space. The CVAE and the BLSTEN are both trained concurrently. The BLSTEN architecture is based on DeepONet [152] and is shown in Fig. 2.14. DeepONet leverages the concept of neural networks as universal operator approximators (based on the universal nonlinear operator approximation theorem introduced by Chen and Chen [44]). As depicted in Fig. 2.14, the DeepONet consists of

- a branch network (i.e., $g_{\lambda}^B(\theta)$) that encodes the discrete input function space and,
- a trunk network (i.e., $g_V^T(t)$) that encodes the domain of the output functions.

This architecture is desirable here because the objective is to construct the neural network as an operator that models the latent space vector over a function space representing different scenarios (e.g., loading cases) to evaluate the corrosion evolution over time. The trunk is a deterministic neural network while the branch is a fully Bayesian neural network. The rationale behind this architectural choice was that it is expected that the majority of the network prediction uncertainty will be associated with the model-form error coming from the parametric portion of the model because it is known that the network’s functional form will only be an approximation of the true underlying solution. Both the trunk and branch are multilayer perceptrons (MLP) with 3 fully connected layers, with 6 units each, and a swish activation function. Unlike a

traditional neural network which learns to predict point estimates, a Bayesian Neural Network learns the posterior distribution, $P(w|\mathcal{D})$, over the weights given an assumed prior distribution, $P(w)$ following Bayes theorem as

$$P(w|\mathcal{D}) = \frac{P(\mathcal{D}|w)P(w)}{\int P(\mathcal{D}|w')P(w')dw'}, \quad (2.43)$$

where w are the model weights to be estimated, $\mathcal{D}$ represents the training data, and $P(\mathcal{D}|w)$ is the likelihood of observing $\mathcal{D}$ for given weights w .

As described in [19], obtaining the exact predictive posterior distribution is intractable for neural networks of any practical size due to the challenge of computing the term in the denominator of Bayes theorem. A common approach for approximating the predictive posterior is to rely on variational inference, as described in Refs. [97, 79]. In this formulation, we begin by minimizing the KL divergence between a variational posterior distribution $q(w|\mu)$ and the desired posterior distribution $P(w|\mathcal{D})$. The KL divergence term is defined as

$$\mathbf{KL}(q(w|\mu)||P(w|\mathcal{D})) = \int q(w|\mu) \log \left(\frac{q(w|\mu)}{P(w|\mathcal{D})} \right) dw, \quad (2.44)$$

where μ are the parameters of the variational distribution $q(w|\mu)$.

This formulation leads to the familiar problem of minimizing the variational free energy or maximizing the expected lower bound (ELBO) which consists of two terms: a KL distance between the variational distribution and the weights prior distribution (effectively acting as a complexity cost) and the negative log-likelihood term that effectively captures the reconstruction loss term. The total loss term is defined as

$$\mathcal{H}(\mathcal{D}, \mu) = \mathbf{KL}[q(w|\mu)||P(w)] - \mathbf{E}_{q(w|\mu)}[\log P(\mathcal{D}|w)]. \quad (2.45)$$

This formulation is used to train the network by employing the Bayes by the Backprop algorithm, as described in [19]. In practice, the loss function is approximated by employing

Monte Carlo sampling such that the loss over a batch of size n is computed as

$$\mathcal{H}(\mathcal{D}, \mu) \approx \sum_i^n \log q(w^{(i)}|\mu) - \log P(w^{(i)}) - \log P(\mathcal{D}|(w^{(i)})), \quad (2.46)$$

where $w^{(i)}$ represents the i -th Monte Carlo sample drawn from the variational posterior $q(w|\mu)$. In this way, the network can be trained more or less as a traditional neural network using the tools available in Tensorflow and Tensorflow Probability [58]. Specifically, a dense variational layer with a Gaussian mean posterior field is used in Tensorflow to keep track of the variational posterior distribution. Thus, the loss term $\mathcal{H}$ is internally added by Tensorflow during training.

The inputs to the network are the set of parameters Θ , which contains the applied normal (σ) and shear (τ) stress, the reaction constant (L_η), and the diffusion coefficient (D^e). In addition to the parameter vector Θ , the time vector is also included as part of the input. A new latent vector sample is obtained from the BLSTEN as

$$\tilde{\mathbf{z}} \sim g_\Pi(\tilde{\mathbf{z}}|\Theta, t) = g_\lambda^B(\Theta)g_v^T(t), \quad (2.47)$$

where $g_\lambda^B(\Theta)$ and $g_v^T(t)$ are the branch and trunk networks, respectively, and $\Pi : [\lambda, v]$ are the network weights.

The fact that pitting corrosion is an irreversible process was implemented into the architecture by constraining the corrosion rate to be negative via a physics loss term

$$\mathcal{L}_i^{PHYS} = \left[\mathbf{ReLU} \left(\frac{d\xi}{dt} \right)_i \right]^2, \quad (2.48)$$

where $\mathbf{ReLU}$ is the rectified linear unit function which is non-zero only for positive values. The corrosion rate can be obtained via autodifferentiation from the decoder and the BLSTEN as

$$\frac{d\xi}{dt} = \frac{\partial \xi}{\partial \tilde{\mathbf{z}}} \frac{\partial \tilde{\mathbf{z}}}{\partial t} = \frac{\partial p_\Psi(\xi|\tilde{\mathbf{z}})}{\partial g_\Pi(\tilde{\mathbf{z}}|\Theta, t)} \frac{\partial g_\Pi(\tilde{\mathbf{z}}|\Theta, t)}{\partial t}. \quad (2.49)$$

Lastly, the output from the encoder and the BLSTEN have to be reconciled. This result is achieved by minimizing the negative log-likelihood as

$$\mathcal{L}_i^{BLSTEN} = -\mathbf{E}_{\mathbf{z} \sim q_{\Phi}(\mathbf{z}|\xi_i)}[\log g_{\Pi}(\mathbf{z}|\Theta, t)]. \quad (2.50)$$

The total loss that combines the above loss terms as illustrated in Fig. 2.14 is computed over the N training samples as

$$\mathcal{L} = \frac{1}{N} \sum_i^N (\alpha \mathcal{L}_i^{VAE} + \gamma \mathcal{L}_i^{BLSTEN} + \phi \mathcal{L}_i^{PHYS}), \quad (2.51)$$

where $\alpha = 500$, $\gamma = 1$, and $\phi = 1$ are the weights used to scale the different loss terms to have roughly the same order of magnitude. The values were determined based on trial and error until it was observed that all the terms were converging to similar orders of magnitude. A more systematic strategy could be used in the future to address this challenge such as gradient surgery for multi-task learning [278] or competitive loss functions [280].

The networks are trained using the Adam optimizer with a learning rate set to 10^{-4} . Next, we use a numerical example to demonstrate the efficacy of the proposed physics-constrained surrogate model.

2.2.5 Numerical Results

Data Collection

Four environmental parameters, including external normal stress, external shear stress, reaction constant, and diffusion coefficient in the phase-field corrosion simulation, are selected as the four input parameters in the ML-based surrogate model. The external normal and shear stress are the mechanical load applied to the model. In the electrochemical system, the reaction constant represents the rate of the reaction, while the diffusion coefficient describes the ratio of flux density to the concentration gradient. These four parameters are major factors that influence the corrosion process. The input parameters are calibrated and normalized according to the

previous work and the literature [198, 31, 41]; the ranges of the parameters are shown in Table 2.3. The Sobol sampling technique is then applied to generate a low-discrepancy sequence of the input parameters combination. A total of 256 corrosion cases are simulated and 255 simulations are completed (one case did not converge). The simulation results of two corrosion cases are plotted in Figure 2.15 for illustration purposes.

Table 2.3. Ranges of Input Parameters

Parameter Name	Ranges
Normal stress (σ)	[-50, 50]
Shear stress (τ)	[0, 50]
Reaction constant (L_η)	[0, 10]
Diffusion coefficient (D^e)	[0, 5]

θ : -5.86, 21.68, 9.58, 0.86 θ : 44.14, 46.68, 4.68, 3.26

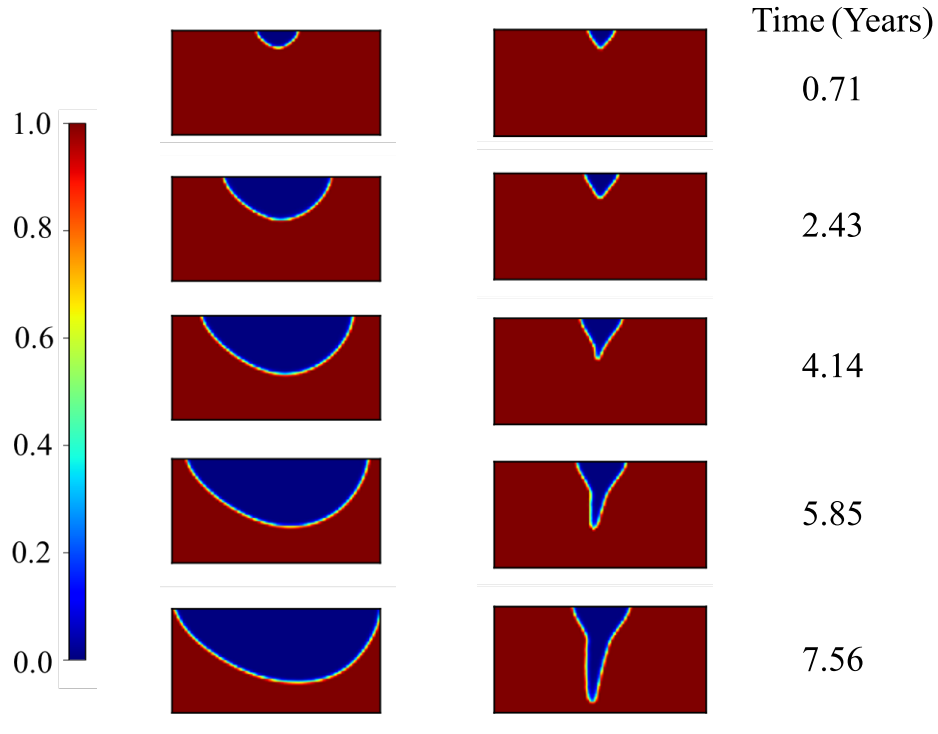


Figure 2.15. Simulation results of two different corrosion cases

To evaluate the performance of the proposed method, images of 220 training samples out of the 255 simulated corrosion cases were used for training. The remaining 25 cases were

used for validation, and 10 were used for testing all the algorithms considered. All the results presented in this section correspond to images from the test set. A six-dimensional latent space was assumed for all algorithms considered based on the work performed in [198].

Comparison Between the CNN-GP-NARX and BLSTEN-VAE Approach Results

The GP-NARX and BLSTEN-VAE approach were trained with the same dataset described above. The GP-NARX and CNN autoencoder portions of the CNN-GP-NARX were trained separately (on 1 GPU and 10 CPUs, respectively) while the BLSTN and VAE were trained simultaneously (on a single GPU). Table 2.4 shows the computational cost of the two approaches.

Table 2.4. Computational cost of predicting 200 time steps for each method.

Method	Training	Prediction
CNN-GP-NARX	7 hours	5.40s
BLSTEN-VAE	96 hours	1.30s

Two variations of the physical constraint were considered. In the first version, the monotonic constraint in Eq. (2.48) was enforced on a pixel-by-pixel basis, while in the second version, the constraint was enforced on the total corrosion (by summing the corrosion rate across all pixels). The reason for considering these two cases was that while the pixel-by-pixel approach captures the correct physics, it is prone to evolve spurious corrosion patterns originating from noisy predictions. The total corrosion constraint allows for small corrections of small spurious pixels while still encouraging the correct physics. An example of this behavior is shown in Fig. 2.16, where it can be seen that the pixel constraint allows a spurious corrosion pattern to grow discontinuously while the total corrosion constraint does not suffer from this. The two approaches were compared with the CNN-GP-NARX approach by computing the mean squared error (MSE) of the mean predictions across all test cases. Fig. 2.17 shows the MSE as a function of time and Table 2.5 presents the average of the MSE across time. As illustrated, the overall error of the BLSTEN-VAE approach is smaller than that of the CNN-GP-NARX approach. In

particular, the BLSTEN-VAE approach with a physics constraint on the total corrosion results in a significant reduction in error. One observation is that the error of the BLSTEN-VAE approach is relatively flat across time while the CNN-GP-NARX's error tends to grow with time. This result is a consequence of the autoregressive approach used in the time evolution mechanism of the NARX algorithm which tends to accumulate error due to its recursive predictions. The BLSTEN algorithm is not autoregressive and, as a result, does not accumulate significant error with time.

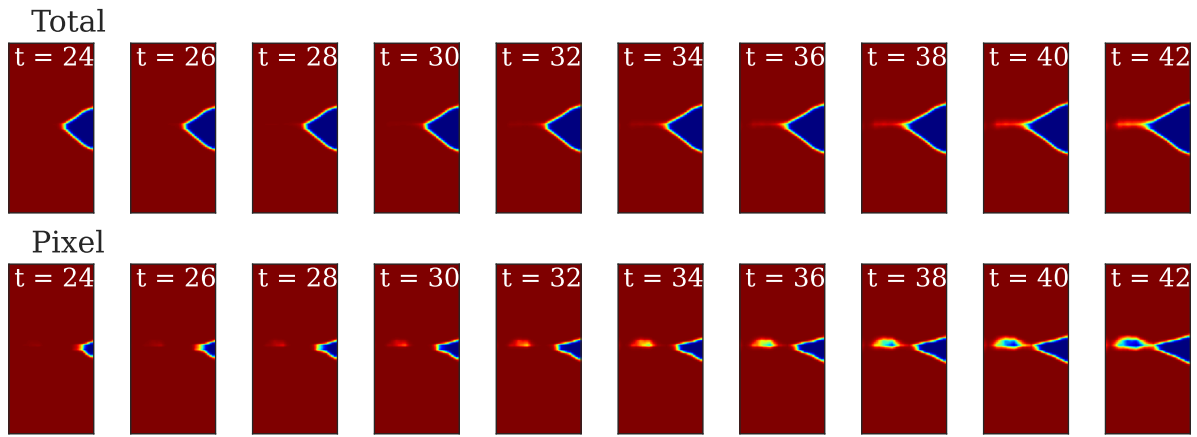


Figure 2.16. Example of corrosion pattern evolution with pixel-by-pixel constraint (bottom) and total corrosion constraint (top). Simulation time step is indicated in figure.

Table 2.5. Average MSE of all test cases across 200 time steps for each method.

Method	Average MSE
CNN-GP-NARX	0.0125
BLSTEN-VAE - pixel	0.0084
BLSTEN-VAE - total	0.0066

One particular example that was hard to predict for the CNN-GP-NARX approach was case 251 with $\theta = [37.89, 6.05, 8.97, 4.76]$. Fig. 2.18 illustrates the results for case 251 for the CNN-GP-NARX approach and the BLSTEN-VAE (with the total corrosion constraint). To show the confidence interval bounds, the edges from the truth data were extracted using a Canny edge

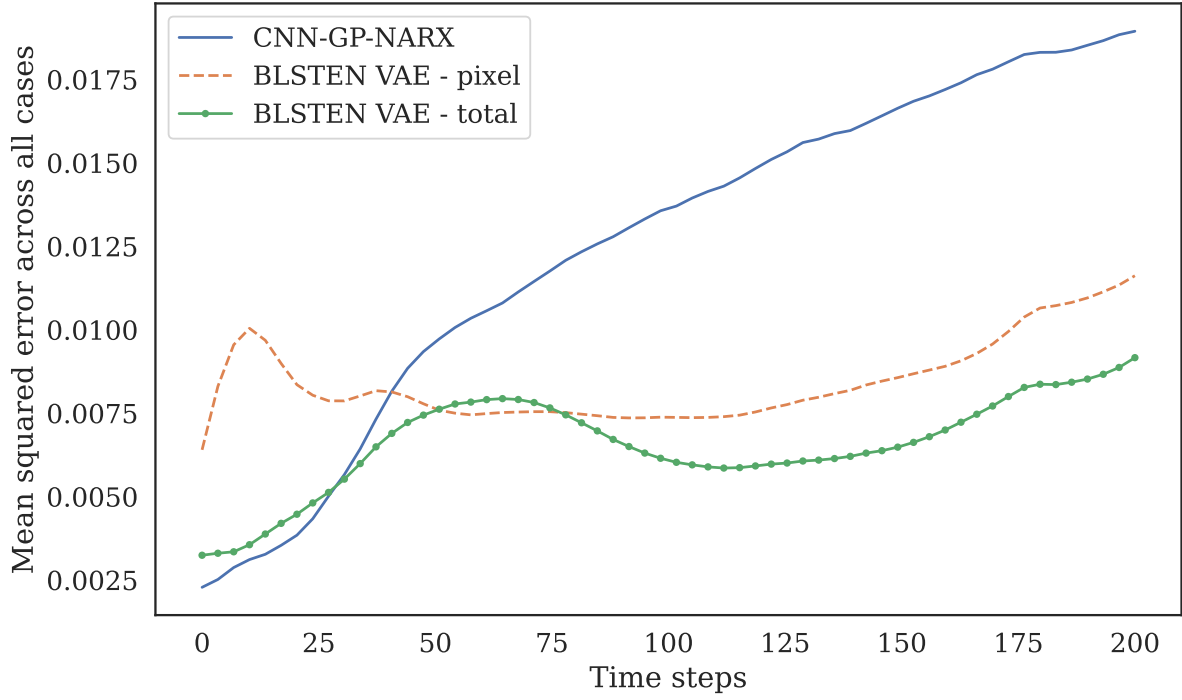


Figure 2.17. Comparison of MSE as a function of time for different methods.

detection algorithm [24] and the predicted confidence interval was drawn around it. As shown, the CNN-GP-NARX prediction is fuzzy around the edges, and this uncertainty is expressed in large 95% level confidence intervals. The area between the two-sided confidence interval is shaded in Fig. 2.18 to represent the boundaries of the predicted corrosion. The BLSTEN-VAE results show a much narrower confidence interval around the true boundary as well as a mean prediction that is much closer to the actual image. A similar behavior can be observed for cases 246 with $\theta = [44.14, 46.68, 4.68, 3.26]$ and 248 with $\theta = [-30.86, 34.18, 2.23, 4.46]$ which are presented in Fig. 2.19 and 2.20, respectively. These cases were chosen because they were challenging to predict and are representative of the three types of corrosion shapes observed. These results demonstrate that the BLSTEN-VAE approach is robust at predicting a variety of corrosion shapes with relatively high confidence.

To understand why these cases are challenging, a corner plot of the parameter space was plotted in Fig. 2.21. The off-diagonal subplots show pair plots of the four physical parameters

samples while the diagonal plots show a histogram of the marginal distribution of each parameter. As illustrated, the parameters closely follow a uniform distribution. The three cases discussed in this section were plotted in color with markers on top to illustrate where they fall relative to the training data. The most significant observation is that all three cases are near edges in at least one of the pair plots, indicating that they are near the training set boundaries. In particular, case 251 (drawn as an orange diamond) is close to the corner in all the pair plots. This observation indicates that these cases may represent near out-of-distribution predictions which is why the CNN-GP-NARX approach struggles with it while the BLSTEN-VAE approach is more robust given the physical constraints built into the algorithm.

BLSTEN-VAE Results Discussion

The trained BLSTEN-VAE networks can be queried to gain physical insight from the data. One of the effects of the VAE algorithm is the disentanglement of latent features [95]. This disentanglement is achieved by the KL divergence term in the loss function shown in Eq. (2.42). Fig. 2.22 shows the evolution of the latent features as a function of time. As illustrated, four of the six latent features exhibit significant change as a function of time, while two of them (latent feature 4 and 5) remain almost flat and are in general much smaller than the other features. This result seems to indicate that the corrosion shape effective dimensionality is four (i.e., the images reside in a four-dimensional manifold) instead of the assumed six latent dimensions. To explore what these latent dimensions represent, the corrosion images were differentiated with respect to the latent dimensions. These spatially varying derivatives are plotted in Fig. 2.23. As illustrated, the latent dimensions seem to be controlling different aspects of the corrosion shape. Latent feature 1 seems to discretize the corrosion boundary into five regions to allow for refined control of the boundary geometry. Latent feature 2 and 6 seem to be more focused on controlling the symmetry of the shape. Latent feature 3 is always negative and seems to be controlling the overall area of the corrosion pattern. This feature is also the only one with a significant constant offset at the initial time (see Fig. 2.22) and may be in charge of enforcing the physical constraint

of Eq. (2.48). As described previously, features 4 and 5 do not have significant contributions based on their magnitude plotted in Fig. 2.22 but are shown for completeness.

A further step is taken to explore the sensitivity of the corrosion shape to the physical parameters. Fig. 2.24 and Fig. 2.25 illustrate the spatially varying derivatives of the corrosion patterns with respect to the physical parameters for a tensile and compressive normal load, respectively. It is observed that the tensile normal load has a significant effect on the sharp corrosion tip area and the compressive normal stress mainly affect the flat pit corrosion area. As shown, the stress parameters seem to have a large influence on the curvature of the corrosion shape, with the normal stress (σ) driving the shape of the corrosion tip, while the shear stress (τ) influence is more pronounced close to the edges. The reaction constant (L^η) seems to control the overall corrosion area, while the diffusion coefficient (D^e) has a minor effect on the corrosion area or shape. All of these effects may also be observed in Fig. 2.26 where the physical parameters were varied one at a time while keeping all the other parameters fixed at their mean value. The trends observed in this result are consistent with physical intuitions of what their effect should be.

2.2.6 Conclusions

This work presented a novel combination of an operator learner (i.e., BLSTEN) and a nonlinear dimensionality reduction technique (i.e., VAE) with a constraint based on the physical phenomenon of irreversible corrosion growth to construct a surrogate model that can be used to forecast corrosion shape based on environmental parameters. A Bayesian approach was used to provide prediction confidence intervals to inform subsequent maintenance decisions down the line. The approach was demonstrated to be more accurate and robust than the established CNN-GP-NARX algorithm by avoiding error accumulation inherent in autoregressive models and through the introduction of physics-inspired constraints. The surrogate model was also queried to understand the effective dimensionality of the corrosion shape as well as its sensitivity to the environmental parameters. It was found that the surrogate model sensitivities were physically

sensible, and that the corrosion shape images reside in a four-dimensional manifold. Future work will focus on improving the algorithm’s performance as well as expanding it to deal with time-varying loads.

2.2.7 Preview of Chapter 3

Chapter 2 described the application of physics-constrained neural differential operators to forecast health indicators for two applications. This chapter demonstrated how a physics-constrained, data-driven approach can be implemented in cases where the physics are too complex to model but some knowledge of physical behavior is known. In both of the previous cases, this knowledge was associated with the monotonicity of the degradation mechanism. This constraint is consistent with the principles of thermodynamics which prevent the physical system from ”repairing” itself in the absence of external energy. Chapter 3 will extend this concept to the general field of structural dynamics. Consistent with thermodynamics principles, the next chapter employs the principle of least action (or Hamilton’s principle) to constrain the functional space of the data-driven model. In the following chapter, the family of ML algorithms is extended to include structure-preserving methods which go beyond just applying physics constraints via the loss function, but have this geometric structure embedded in their architecture.

2.2.8 Remarks

Portions of this dissertation have been published or are being considered for publication. Section 2.2 is composed of the following first authored publications:

David A. Najera-Flores, Guofeng Qian, Zhen Hu, and Michael D. Todd. Corrosion morphology prediction of civil infrastructure using a physics-constrained machine learning method. *Mechanical Systems and Signal Processing*, 200:110515, 2023

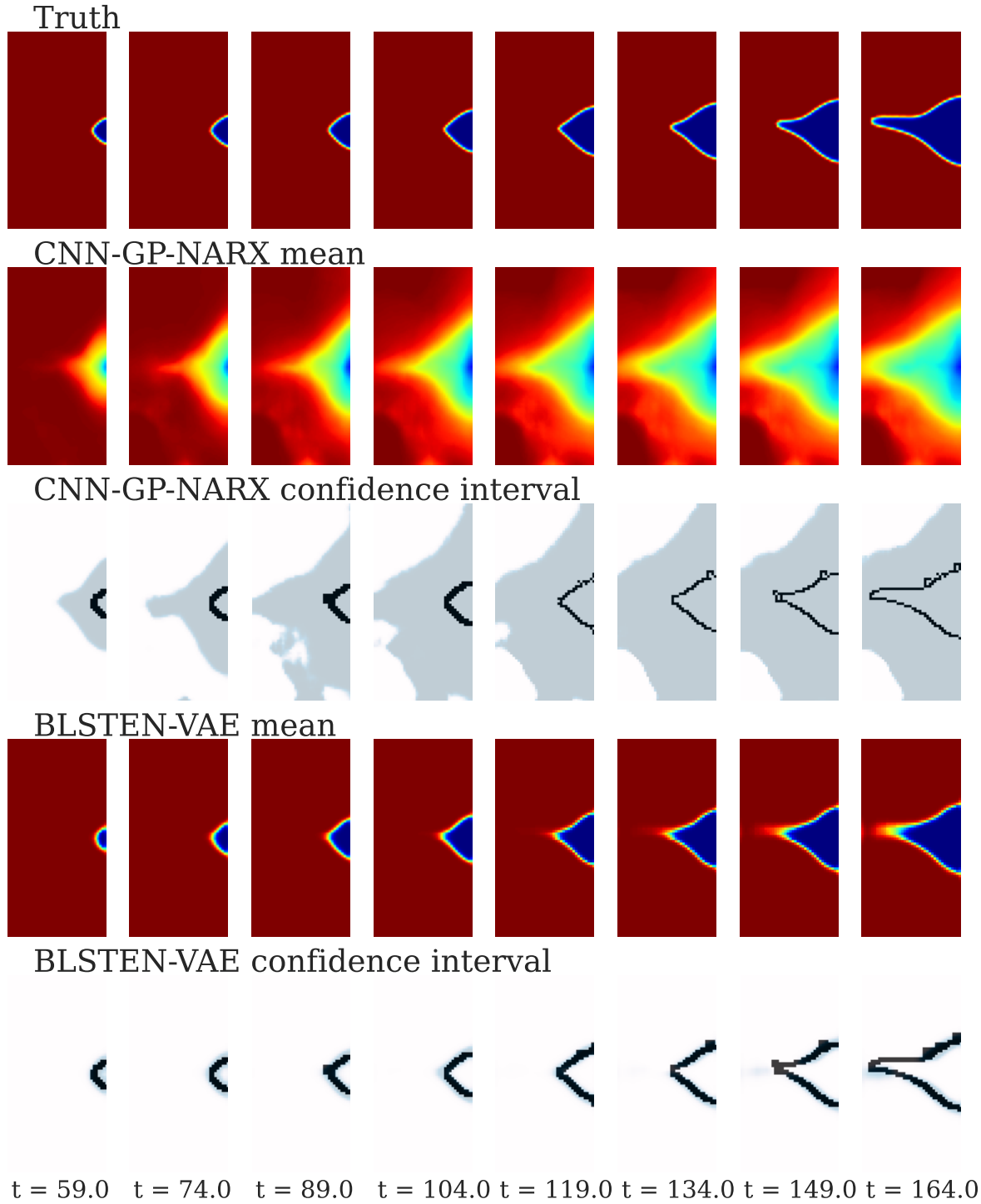


Figure 2.18. Results for test case 251. Shown: simulation reference (first), CNN-GP-NARX mean prediction (second), CNN-GP-NARX 95% confidence interval (third), BLSTEN-VAE mean prediction (fourth), and BLSTEN-VAE 95% confidence interval. Simulation time step is indicated.

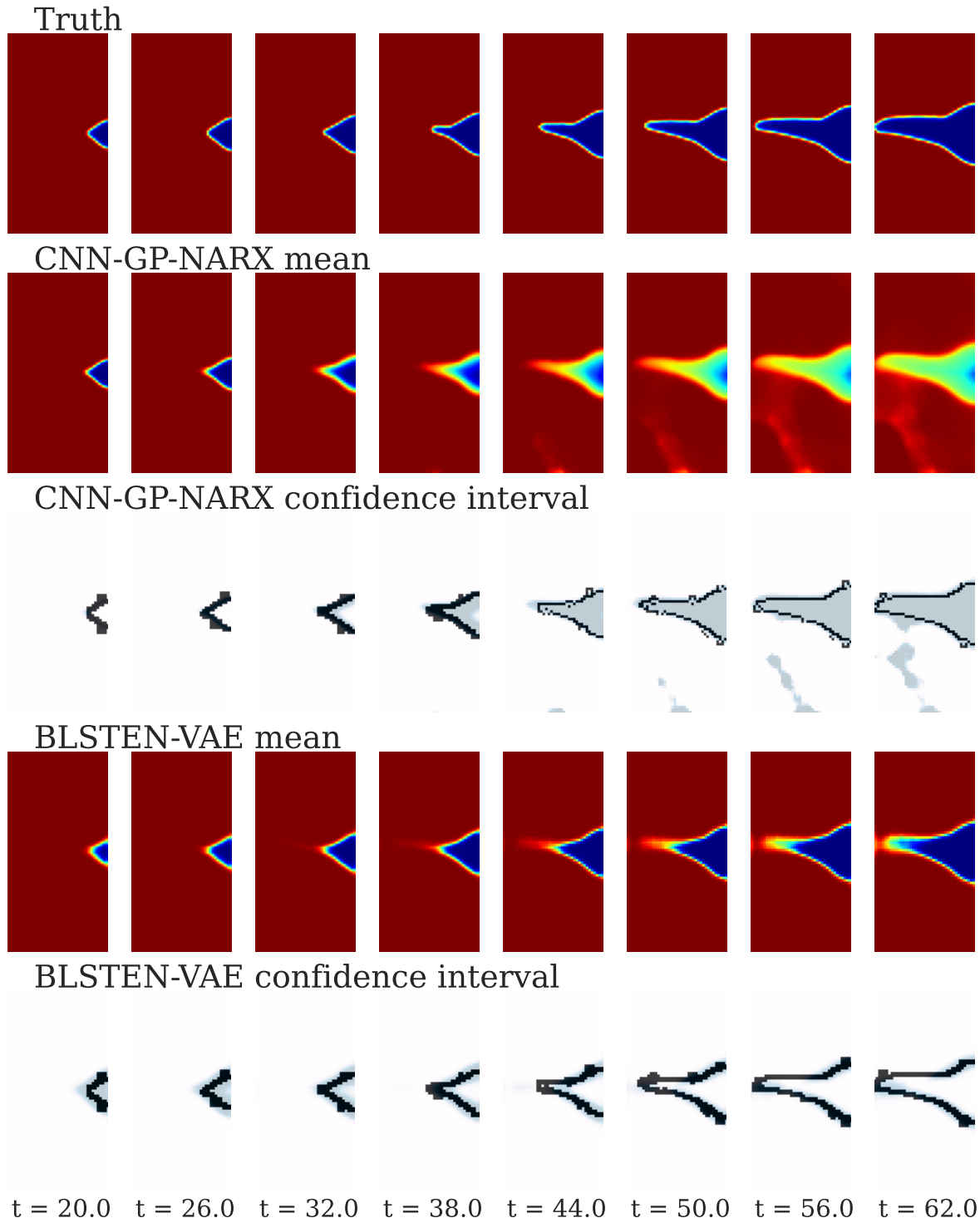


Figure 2.19. Results for test case 246. Shown: simulation reference (first), CNN-GP-NARX mean prediction (second), CNN-GP-NARX 95% confidence interval (third), BLSTEN-VAE mean prediction (fourth), and BLSTEN-VAE 95% confidence interval. Simulation time step is indicated.

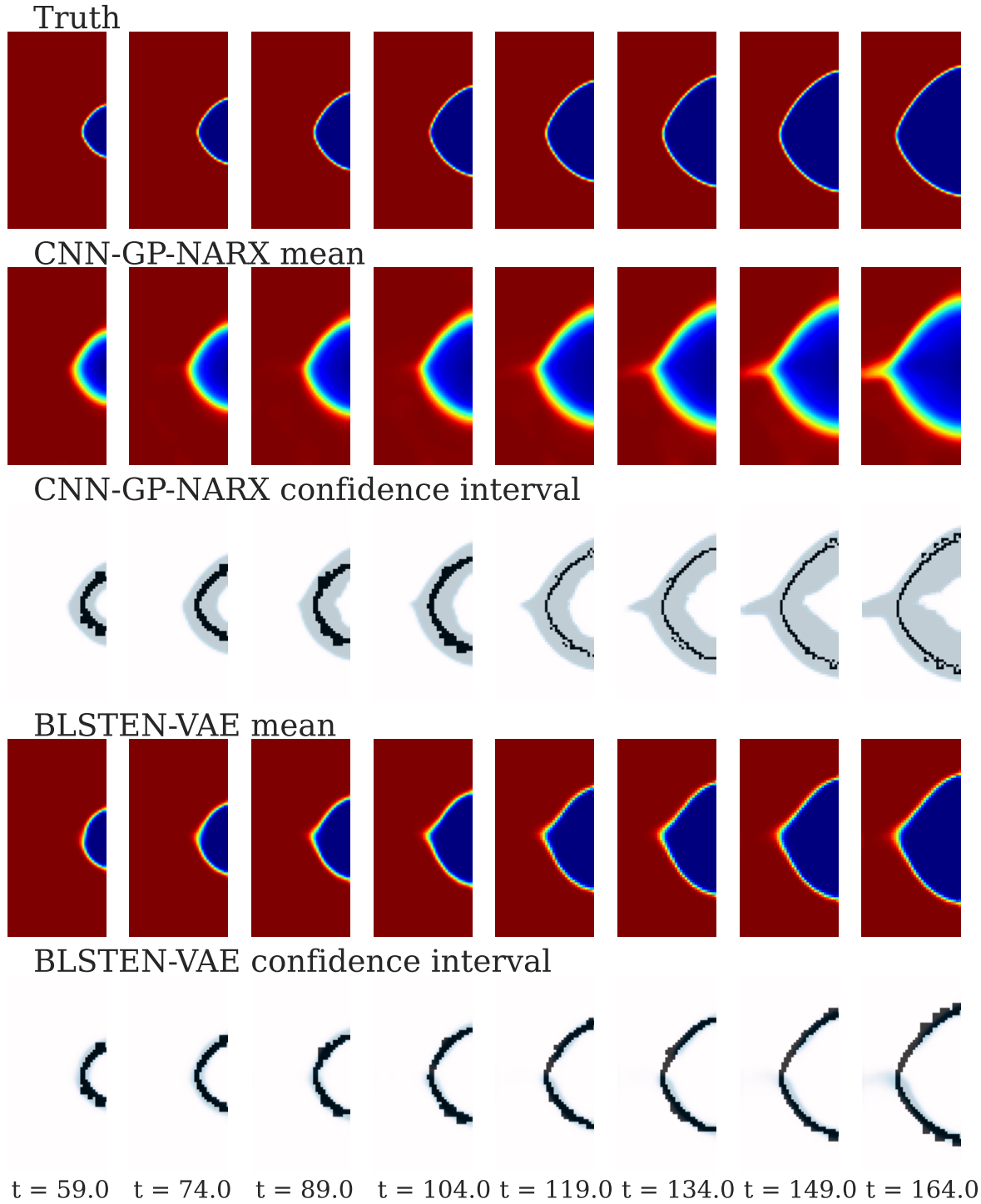


Figure 2.20. Results for test case 248. Shown: simulation reference (first), CNN-GP-NARX mean prediction (second), CNN-GP-NARX 95% confidence interval (third), BLSTEN-VAE mean prediction (fourth), and BLSTEN-VAE 95% confidence interval. Simulation time step is indicated.

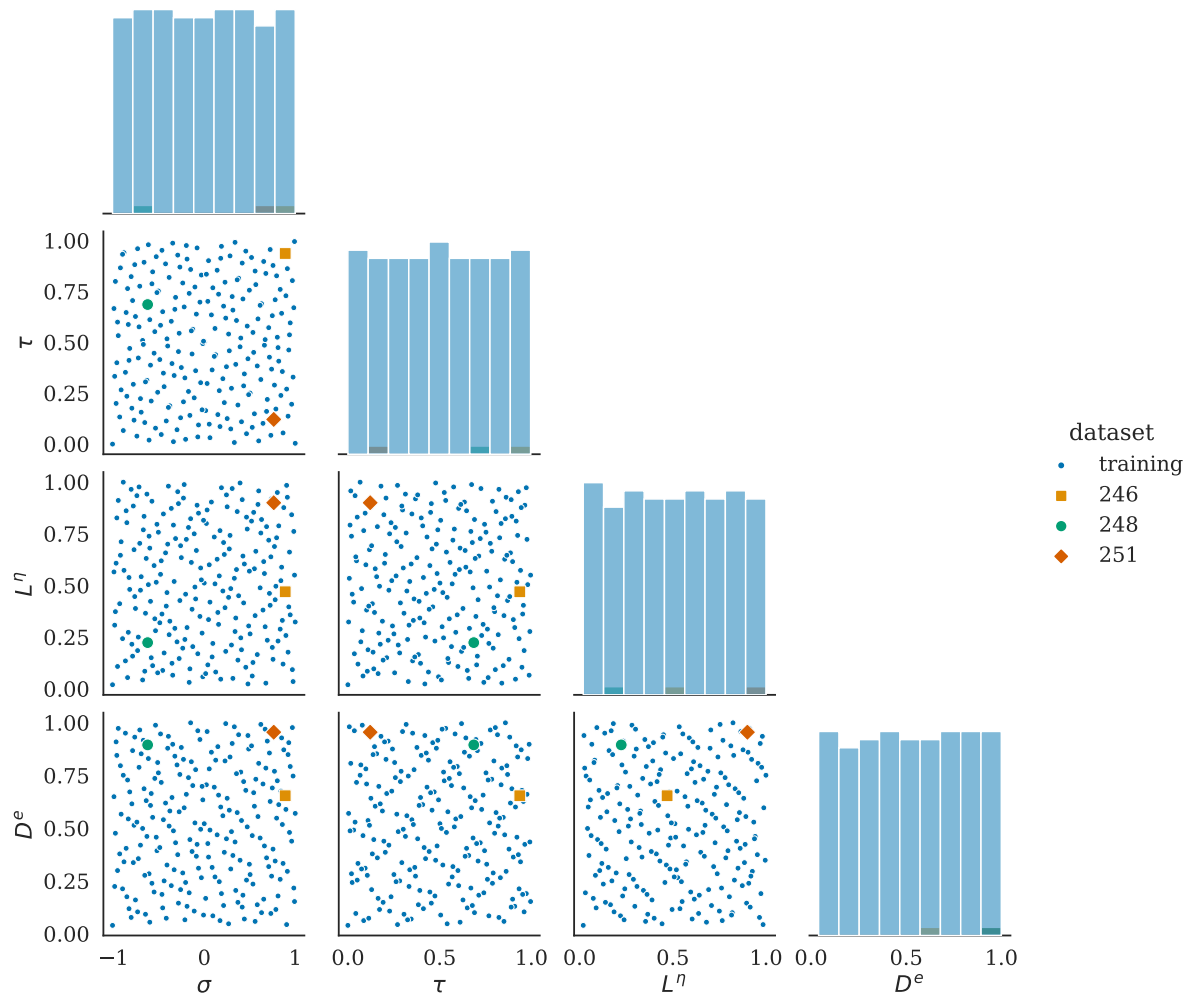


Figure 2.21. Corner plot showing the distribution of training set parameters and selected test set cases.

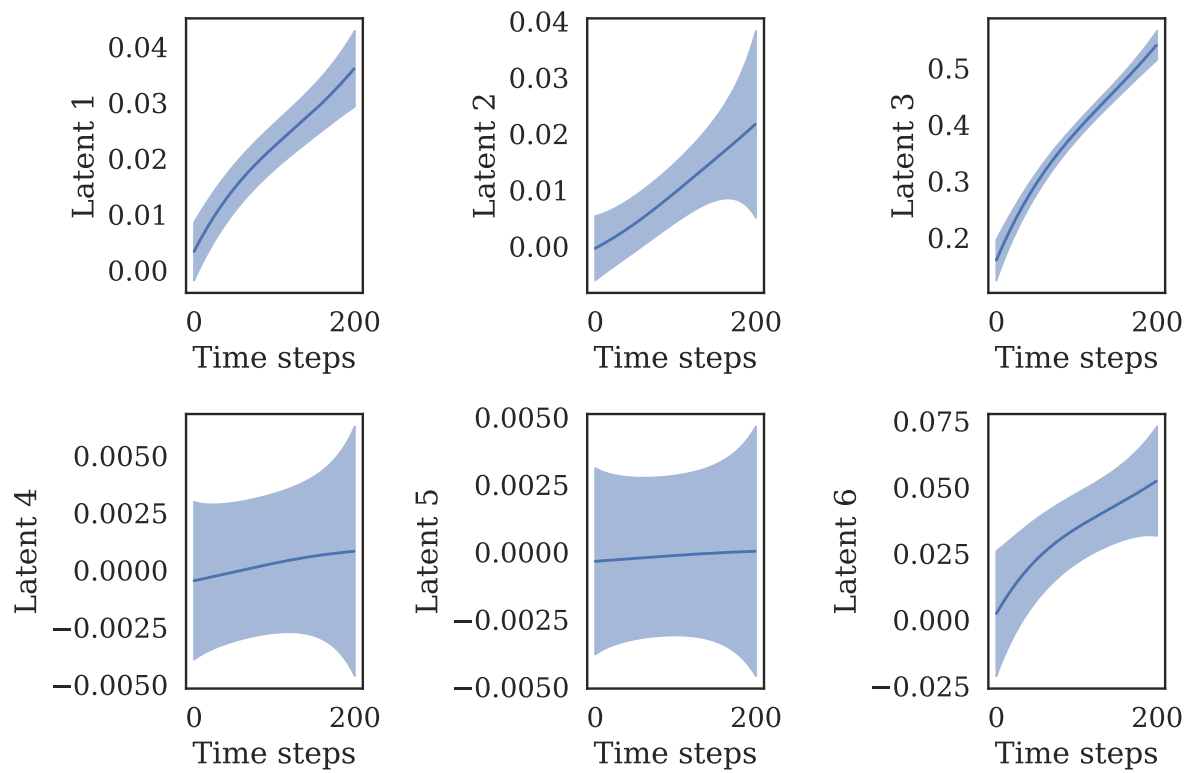


Figure 2.22. Evolution of latent features as a function of time.

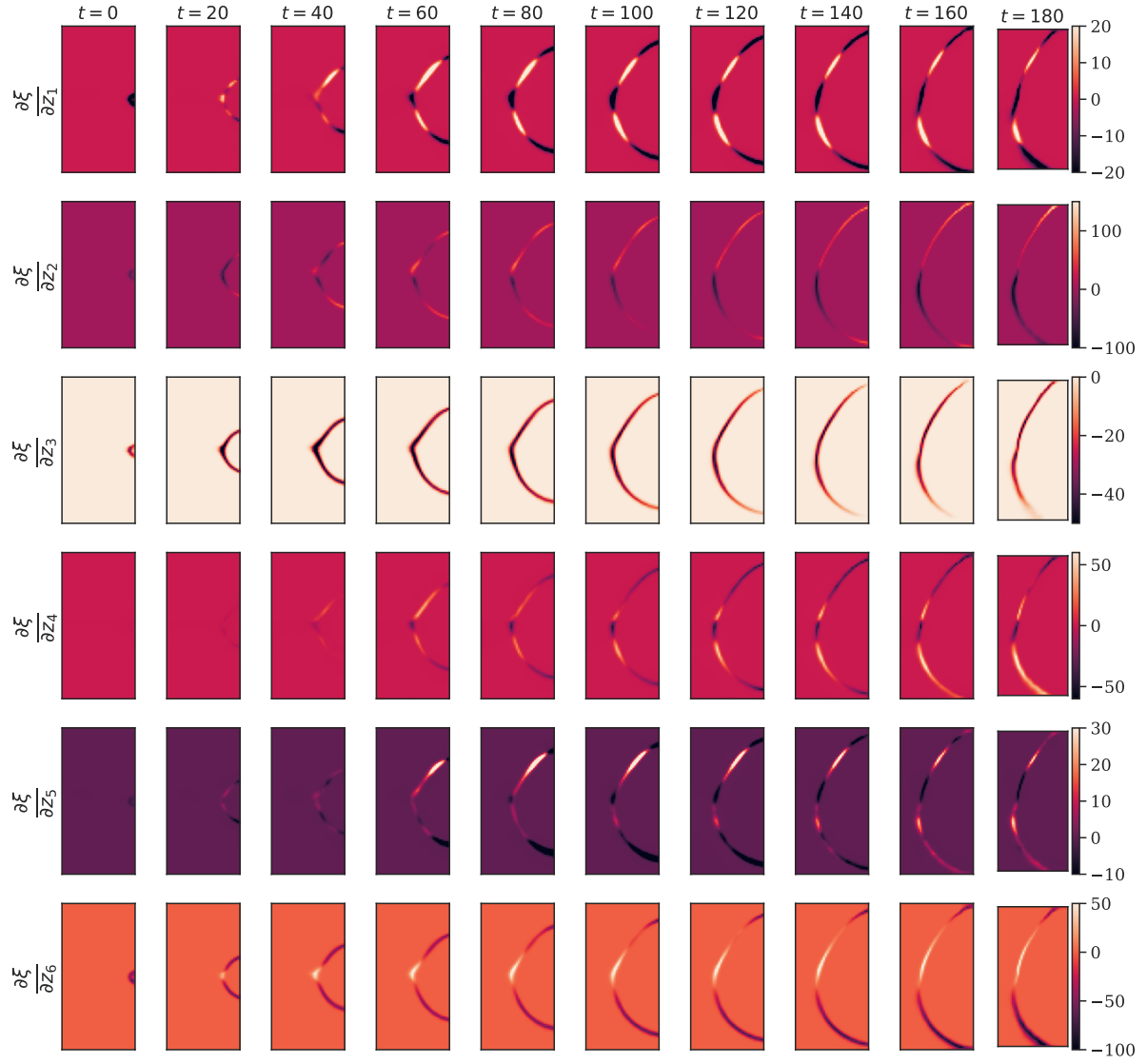


Figure 2.23. Derivatives of pitting corrosion with respect to the latent features. Simulation time step is indicated at the top.

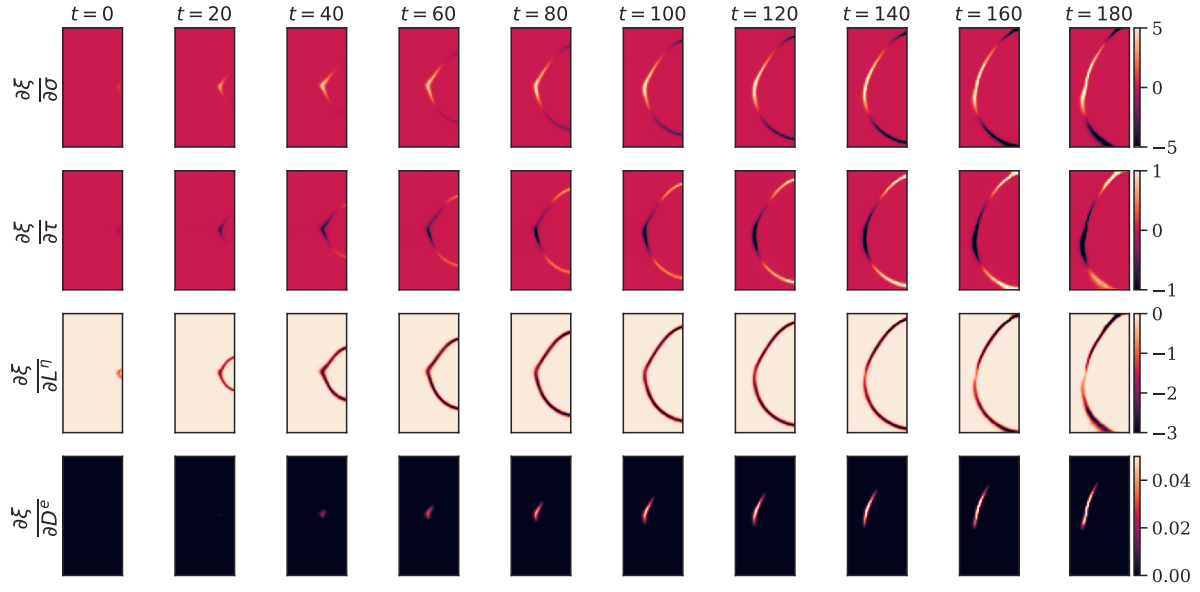


Figure 2.24. Derivatives of pitting corrosion with respect to the physical parameters for a tensile normal load. Simulation time step is indicated at the top.

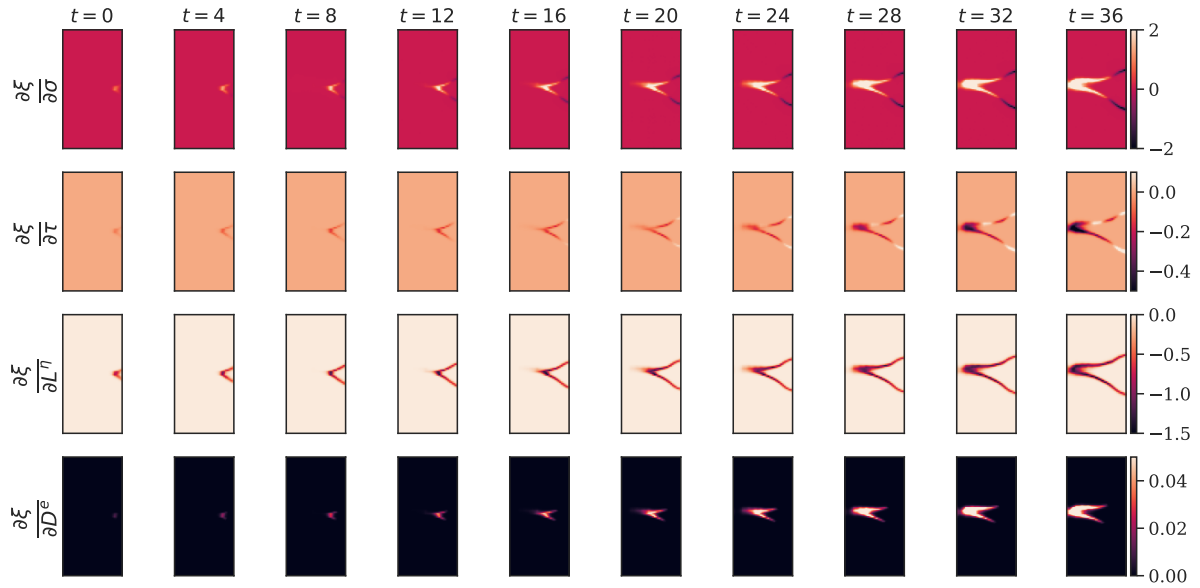


Figure 2.25. Derivatives of pitting corrosion with respect to the physical parameters for a compressive normal load. Simulation time step is indicated at the top.

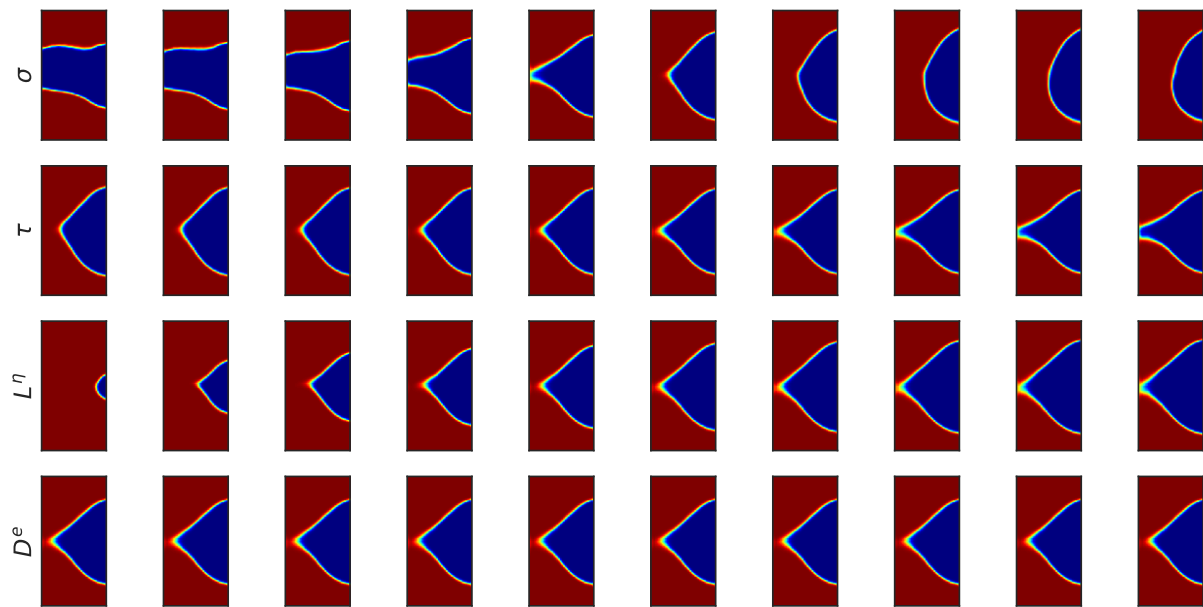


Figure 2.26. Effect of physical parameter variation from low (left) to high (right) on corrosion shape.

Chapter 3

Structure-Preserving Neural Differential Operators for Structural Dynamics

Modeling and simulation of physical systems are an essential part of the the design cycle and maintenance in many disciplines, including aerospace, structural, mechanical, and civil engineering. Physics-based models provide interpretable predictive results that can help guide analysts and designers to understand and predict the performance of mechanical systems. However, accurate and precise physics-based model development can be expensive and time-consuming. Moreover, not all physical phenomena are readily modeled *a priori*. In many cases the final as-built, as-deployed mechanical system is not accurately represented by an existing model, necessitating careful model updating to bridge the gap between model and observation. The differences between the model and the as-deployed configuration may be due to a variety of reasons including damage, unknown physics, and inadequate boundary conditions.

On the other hand, data-driven modeling approaches are popular when experimental data are available. These approaches range from more traditional techniques such as singular value or proper orthogonal decomposition [119][16][17] to more recent work with neural networks, with applications in areas such as climate modeling [168][128], material modeling [266][276][116], and structural dynamics [236][274]. Purely data-driven methods generally lack physical interpretability and are limited in the domain they may operate in (e.g., not outside their training domain), and as a result, they are not very robust. More recently, physics-informed neural

networks (PINNs) have become more pervasive in the literature, ranging from applications in solving classical partial differential equations (PDEs) [203][121][277], including the Navier-Stokes equation [113], as well as applications in geostatistical modeling [290], defect detection [281], and solution of non-homogeneous dynamic systems and elasticity problems [27], to larger scale systems including hybrid mesh-free methods [237][93]. These approaches typically operate by transforming the PDE residual into a loss term, as well as enforcing boundary and initial conditions with similar penalty methods.

In contrast with physics-informed methods, structure-preserving approaches seek to enforce physical constraints through inductive biases built into the data-driven mechanism, e.g., the neural network architecture or in reduced order modeling via an appropriate choice of reduction basis [217][197][137]. One way to preserve the underlying geometric structure of a dynamical system is by constraining the data-driven model to satisfy a Lagrangian or Hamiltonian system structure [226][111][164][80] [212]. Many of these approaches assume that the variables/observables are in canonical or generalized coordinate form, which is generally not the case for experimental data. Another common theme in past research is to start with the assumption that at least a few state parameters are known (e.g., the mass matrix), but this also is not always true in many real applications. The present research seeks to build upon these methods to address practical problems of observability and the fact that canonical or generalized coordinates are not usually measurable.

In this work, we propose a structure-preserving neural network framework based on Hamiltonian mechanics. The Hamiltonian framework allows us to relate the energy of the system to the measured quantities (e.g., accelerations) through the Euler-Lagrange equations of motion. In this work, the potential, kinetic energy, and Rayleigh damping terms are each modeled with a multilayer perceptron (MLP). Auto-differentiation is used to compute partial derivatives and to assemble all the relevant equations, including computing the generalized inertia matrix, by forming the Hessian of the kinetic energy with respect to the generalized coordinates. Moreover, a system identification strategy is used to predict domain shifts in the

data. The network incorporates a numerics-informed loss function via the residual of a multi-step integration term, allowing the ensemble of networks to be time-integrated with new initial conditions and an arbitrary external force after it has been trained, effectively functioning as a differential operator. Our approach incorporates a physics-constrained autoencoder to perform coordinate transformation between measured coordinates and a generalized coordinate set. This approach results in a physics-informed, data-driven model of the structure that can respond to changes in the structure by updating its own parameters and maintain a physically meaningful interpretation of the updates. Such an approach can form the basis of a digital twin for many appropriate applications. The technique is demonstrated on simulated data (computational) examples.

This chapter is organized as follows. Section 3.1 describes the principles of Hamiltonian mechanics and their integration into the neural network architecture. Section 3.2 presents a series of numerical examples that demonstrate different aspects of the approach. Finally, section 3.3 provides concluding remarks and future research directions.

3.1 Background

Section 3.1.1 reviews the basic theory behind the Hamiltonian mechanics framework. In section 3.1.2 we introduce the general neural pipeline for learning the system energy terms. In section 3.1.3 we describe the physics-constrained autoencoder used to perform coordinate transformation. Section 3.1.4 goes over the methodology for detection of shifts in the data.

3.1.1 Hamiltonian Mechanics

The foundation for Hamiltonian mechanics is the principle of least action or Hamilton's principle of stationary action. We start with the action functional $\mathcal{S}$ which is defined as

$$\mathcal{S} = \int_{t_1}^{t_2} L(\mathbf{q}, \dot{\mathbf{q}}, t) dt. \quad (3.1)$$

The scalar action $\mathcal{S}$ is a functional of the Lagrangian $L(\mathbf{q}, \dot{\mathbf{q}}, t)$. This functional is integrated between arbitrary times t_1 and t_2 . Hamilton's principle states that the realized trajectory of the system corresponds to that which makes the action stationary with respect to change of variables. To achieve this, the variational principle is invoked by considering a virtual infinitesimal displacement, δ , such that the perturbed action is

$$\mathcal{S}_B = \int_{t_1+\Delta t}^{t_2+\Delta t} L(\mathbf{q} + \delta\mathbf{q}, \dot{\mathbf{q}} + \delta\dot{\mathbf{q}}) dt. \quad (3.2)$$

To determine the stationary action condition, we then consider the total variation of the action functional

$$\delta\mathcal{S} = \mathcal{S}_B - \mathcal{S} = \int_{t_1}^{t_2} \sum_{j=1}^n \left(\frac{\partial L}{\partial q_j} - \frac{d}{dt} \frac{\partial L}{\partial \dot{q}_j} \right) \delta q_j dt + \mathcal{O}(\delta^2), \quad (3.3)$$

where $\mathcal{O}(\delta^2)$ indicates higher-order terms and are usually ignored to obtain the first variation of the action functional. This first variation is extremized to determine the trajectory that makes the action functional stationary. The additional deterministic assumption that the starting and end points are known is usually made ($\delta q_i(t_1) = \delta q_i(t_2) = 0$). In that case, the action scalar can be extremized by setting the first variation to zero which results in the homogeneous version of the Euler-Lagrange equations of motion

$$\frac{\partial L}{\partial q_j} - \frac{d}{dt} \frac{\partial L}{\partial \dot{q}_j} = 0. \quad (3.4)$$

This expression corresponds to the conservative case. In the presence of dissipative and external forces, the Euler-Lagrange equations of motion are modified as follows,

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_j} - \frac{\partial L}{\partial q_j} + \frac{\partial I}{\partial \dot{q}_j} = Q_j^{nc}, \quad (3.5)$$

where I represents the Rayleigh damping term and $\mathbf{Q}^{nc}$ is the vector of generalized non-conservative forces.

These principles are the basis for the structure imposed on the framework described in the next sections.

3.1.2 Physics-Constrained Autoencoder

One important feature of operating in a Lagrangian or Hamiltonian frame of reference is the necessity to express the degrees-of-freedom (DOFs) of a discrete system in a generalized or canonical coordinate system. In this work, we assume that the original measured DOFs may not be in generalized coordinates, $\mathbf{q}$, and are instead expressed in an arbitrary coordinate system, $\mathbf{r}$. To perform a coordinate transformation, we propose to use an autoencoder architecture. The coordinate transformation operation is then performed as follows,

$$\mathbf{q} = \mathcal{E}_{nn}(\mathbf{r}) \quad (3.6)$$

$$\mathbf{r} = \mathcal{D}_{nn}(\mathbf{q}), \quad (3.7)$$

where $\mathcal{E}_{nn}$ and $\mathcal{D}_{nn}$ are neural networks representing the encoder and decoder, respectively. Both of these are constructed by performing a nonlinear projection, where a transformation matrix $\mathbf{T}$ is learned from an MLP. This specific construction serves as an additional constraint to the transformation type allowed and has been shown to speed-up the training process significantly for the problems considered in this chapter compared to learning the coordinate transformation directly from an MLP,

$$\mathcal{E}_{nn} = \mathbf{T}_{\mathcal{E}}(\mathbf{r}) \cdot \mathbf{r} \quad (3.8)$$

$$\mathcal{D}_{nn} = \mathbf{T}_{\mathcal{D}}(\mathbf{q}) \cdot \mathbf{q}. \quad (3.9)$$

In this work we assume that the velocities and accelerations can be obtained from mea-

surements. These also need to be transformed back and forth between arbitrary and generalized coordinates. The generalized velocities and acceleration can be obtained by applying the chain rule,

$$\dot{q}_i = \sum_{j=1}^n \frac{\partial q_i}{\partial r_j} \cdot \dot{r}_j, \quad (3.10)$$

$$\ddot{q}_i = \sum_{j=1}^n \frac{\partial q_i}{\partial r_j} \cdot \ddot{r}_j + \sum_{j=1}^n \frac{\partial \dot{q}_i}{\partial r_j} \cdot \dot{r}_j. \quad (3.11)$$

The expressions are left unsimplified to show the actual operations that are performed with auto-differentiation. The decoded velocities and accelerations in the original coordinate system can be obtained from the generalized coordinates in a similar manner,

$$\dot{r}_j = \sum_{i=1}^k \frac{\partial r_j}{\partial q_i} \cdot \dot{q}_i, \quad (3.12)$$

$$\ddot{r}_j = \sum_{i=1}^k \frac{\partial r_j}{\partial q_i} \cdot \ddot{q}_i + \sum_{i=1}^k \frac{\partial \dot{r}_j}{\partial q_i} \cdot \dot{q}_i. \quad (3.13)$$

The physical forces, assumed to be velocity-independent, are also transformed into generalized forces by following the principle of virtual work and arriving at the following expression,

$$Q_i^{nc} = \sum_{j=1}^n \frac{\partial r_j}{\partial q_i} \cdot F_j. \quad (3.14)$$

Because the autoencoder is constructed using differentiable neural networks, all of these operations are performed with auto-differentiation using the Python package Jax [23] via computing the forward Jacobian of either the encoder or decoder output with respect to its input.

The only supervised aspect of the training procedure is performed on the decoded acceleration via a reconstruction loss term as

$$\mathcal{L}_{recon} = \|\hat{\mathbf{r}} - \ddot{\mathbf{r}}\|_2^2, \quad (3.15)$$

where $\hat{\mathbf{r}}$ represents the reconstructed accelerations in the original physical coordinate system. Note that the decoder outputs the displacements directly, and the accelerations are then computed from the displacements and velocities using Eq. 3.12 and 3.13.

3.1.3 Structure-Preserving Neural Network Ensemble

To define the dynamics of a general non-conservative Lagrangian system, we first determine the kinetic energy $T(\mathbf{q}, \dot{\mathbf{q}})$, the potential energy $U(\mathbf{q})$, and the Rayleigh damping term $I(\mathbf{q}, \dot{\mathbf{q}})$. These quantities are in the most general form a function of the generalized displacements $\mathbf{q}$ and generalized velocities $\dot{\mathbf{q}}$. Each of these terms is modeled with a multilayer perceptron (MLP) that is augmented with complete polynomial terms, inspired by SINDy [28]. For example, the kinetic energy function would be defined as

$$\begin{aligned} T_{nn}(\mathbf{q}, \dot{\mathbf{q}}) = & \sum_{i,j,\dots,p}^{i+j+\dots p=M} \alpha_{i,j,\dots,p} q_1^i q_2^j \dots q_k^p \\ & + \beta_{i,j,\dots,p} \dot{q}_1^i \dot{q}_2^j \dots \dot{q}_k^p \\ & + \sum_n^N \theta_n \mathcal{G}^{(n)}(\mathbf{q}, \dot{\mathbf{q}}), \end{aligned} \quad (3.16)$$

where the coefficients α , β , and θ are learned during training, M is the degree of the polynomial, and the function $\mathcal{G}^{(n)}$ represents an MLP. In general, a fully connected neural network is used to define $\mathcal{G}^{(n)}$. The dimensionality of the MLP is problem-dependent, but for this work, the network depth generally did not exceed six layers, each with a maximum of 22 units. As shown in Eq. 3.16, the energy terms do not impose any assumptions that constrain the approach to linear problems. The combination of polynomial terms of arbitrary order and the nonlinear activation functions in the MLP ensure that the network can model arbitrary nonlinear problems.

The potential energy and Rayleigh damping terms are similarly defined. The Lagrangian is constructed from its definition as

$$L(\mathbf{q}, \dot{\mathbf{q}}) = T_{nn}(\mathbf{q}, \dot{\mathbf{q}}) - U_{nn}(\mathbf{q}), \quad (3.17)$$

and the generalized momenta are subsequently computed as

$$p_k = \frac{\partial L(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{q}_k}. \quad (3.18)$$

Next, the Hamiltonian is computed from the Lagrangian and the generalized momenta as

$$H(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{p}\mathbf{q} - L(\mathbf{q}, \dot{\mathbf{q}}), \quad (3.19)$$

and then the momentum rates are computed from the Hamiltonian, the generalized forces, and the dissipative terms as

$$\dot{p}_k = -\frac{\partial H(\mathbf{q}, \dot{\mathbf{q}})}{\partial q_k} + Q_i^{nc} - \frac{\partial I_{nn}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{q}_k}. \quad (3.20)$$

The generalized inertia matrix may be obtained by taking the Hessian of the kinetic energy term with respect to the generalized velocities, where the computation in this manner enforces symmetry (another appropriate physical constraint on an inertia matrix):

$$M_{ij} = \frac{\partial^2 T_{nn}(\mathbf{q}, \dot{\mathbf{q}})}{\partial \dot{q}_i \partial \dot{q}_j}. \quad (3.21)$$

Finally, the generalized accelerations are obtained by inverting the generalized inertia matrix. A Moore-Penrose inverse is used to avoid numerical issues, which are common during early stages of training when the matrix $\mathbf{M}$ may be near singular due to random initialization of the weight parameters in the energy terms,

$$\ddot{q}_n = M_{ni}^{-1}(\dot{p}_i - M_{ij,k}\dot{q}_k\dot{q}_j). \quad (3.22)$$

A schematic of the ensemble of neural networks is illustrated in Figure 3.1. To further illustrate the neural network workflow, the process is presented as pseudocode below:

Algorithm 1. Neural network ensemble workflow

- 1: **for** every batch **do**
 - 2: Compute T_{nn} , U_{nn} , and I_{nn}
 - 3: Compute the Lagrangian L (Eq. 3.17)
 - 4: Compute the momentum $\mathbf{p}$ (Eq. 3.18)
 - 5: Compute the Hamiltonian H (Eq. 3.19)
 - 6: Compute the momentum rate $\dot{\mathbf{p}}$ (Eq. 3.20)
 - 7: Compute the generalized inertia matrix $\mathbf{M}$ (Eq. 3.21)
 - 8: Compute the Moore-Penrose inverse of the generalized inertia matrix
 - 9: Compute the generalized accelerations $\ddot{\mathbf{q}}$ from the momentum rate and inverse generalized inertia matrix (Eq. 3.22)
 - 10: **end for**
-

Examples of the Python implementation of the method can be found at https://github.com/dnajera27/hamiltonian_mech_node_jax.

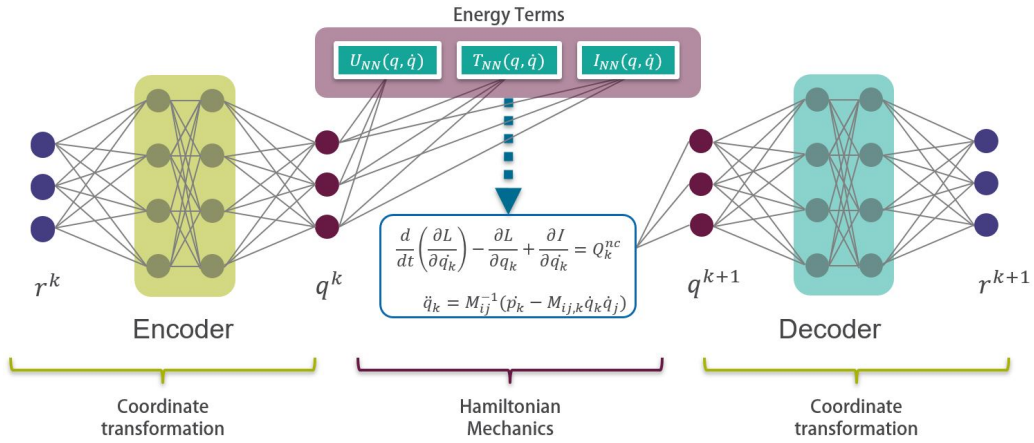


Figure 3.1. Schematic of ensemble of neural networks used.

Numerics- and Physics-Informed Loss Terms

The prescribed structure described above ensures that the network adheres to physical constraints. However, a few additional numerics- and physics-informed loss terms were defined to ensure that the network is self-consistent and time-integrable. The most significant constraint was implemented by evaluating the residual of a multi-step integration step to ensure that rates are accurately predicted. The general loss term has the following form:

$$\mathcal{L}_{diff} = \left\| \sum_{j=0}^s \gamma_j \begin{pmatrix} \mathbf{q}_{k+j} \\ \dot{\mathbf{q}}_{k+j} \end{pmatrix} - h \sum_{j=0}^s \eta_j \begin{pmatrix} \dot{\mathbf{q}}_{k+j} \\ \ddot{\mathbf{q}}_{k+j} \end{pmatrix} \right\|_2^2, \quad (3.23)$$

where γ and η are coefficients from the Butcher tableau corresponding to the Adams-Bashforth linear multistep integration method, and h is the time step associated with the training data. The $\mathbf{q}_{k+j}$ term represents the $(k+j)_{th}$ time step in the $\mathbf{q}$ vector. The same general loss function form was applied to the momenta and its time derivative, $\mathbf{p}$ and $\dot{\mathbf{p}}$. This loss term is used to ensure mathematical consistency in the differential equation by evaluating the residual of a multi-step integration step between a variable and its time derivative. This loss term is used for all time derivatives since time is not an explicit input in the network and hence auto-differentiation cannot be used to evaluate time derivatives in this case.

In addition, the following physical constraints were included in the loss term to ensure self-consistency:

$$\begin{aligned} \mathcal{L}_{phys} = & \|T_{nn}(\mathbf{q}, \dot{\mathbf{q}}) - 1/2\mathbf{q} \cdot \mathbf{p}\|_2^2 \\ & + \|H(\mathbf{q}, \dot{\mathbf{q}}) - (T_{nn}(\mathbf{q}, \dot{\mathbf{q}}) + U_{nn}(\mathbf{q}))\|_2^2 \\ & + \|\mathbf{p} - \mathbf{M}\dot{\mathbf{q}}\|_2^2. \end{aligned} \quad (3.24)$$

3.1.4 Trained Model Deployment

While the ensemble of neural networks was trained in a supervised manner to reproduce the dynamics of observed data, the inference stage is set up as an ordinary differential equation (ODE) operator. We begin with a set of initial conditions, $\mathbf{x}_0$ and applied external force vector, $\mathbf{F}_{ext}$. The first step is to use the trained encoder to obtain the initial conditions in transformed generalized coordinates, $\mathbf{q}_0$ and $\dot{\mathbf{q}}_0$ using Eq. 3.6 and 3.10. Similarly, the generalized forces, $\mathbf{Q}^{nc}$, are obtained using Eq. 3.14. Given the initial conditions and forcing function, the trained neural networks are used as a neural ODE operator expressed in state space,

$$\begin{pmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{pmatrix} = \mathcal{N}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{Q}^{nc}), \quad (3.25)$$

where the generalized acceleration is computed using Eq. 3.22. The system can now be solved using any ODE solver. In this work we used the RK45 method [61] implemented in the Scipy package [257] in Python. Finally, the accelerations in the original coordinate system are computed by passing the time-integrated generalized responses to the decoder, $\mathcal{D}_{nn}(\mathbf{q})$.

3.1.5 Domain Shift Identification

Once the neural ODE has been deployed, the network can be re-trained at any time to assimilate new data. In the case that new data are available that exhibit shifts from the original data domain, the network can be used to localize and identify the source of these changes. This action is accomplished by running the training process (i.e., optimization via backpropagation) for a few epochs and then looking for changes in the statistical distribution of the weight shifts associated with the energy neural networks. Localization is accomplished by computing the system state matrices and looking for changes in specific entries. The state matrices $\mathbf{M}$, $\mathbf{K}$, and $\mathbf{C}$ are computed by taking the Hessian of the kinetic, potential, and Rayleigh damping networks, respectively, similar to what was done in Eq. 3.21. The task of automatically identifying shifts to

trigger a model updating step is left for future work. In this work, we only look at the effect of domain shifts on the model parameters.

3.2 Numerical Results and Discussion

This section will demonstrate the approach applied to four computational examples: an Euler-Bernoulli beam, a two degree-of-freedom system with cubic nonlinearities in the stiffness and damping terms, a rigid link with a geometric nonlinearity, and a single degree-of-freedom duffing oscillator. For all the numerical examples shown, a single training example was used. The training data was long enough to include at least two periods of oscillation of the lowest frequency in each example. All the results shown were computed for different initial conditions and external forcing than those used for training.

3.2.1 Euler-Bernoulli Beam

This example demonstrates the core capability of the framework to learn the kinetic, potential, and Rayleigh damping terms, as well as derived state matrices. For this example, we consider a linear Euler-Bernoulli beam problem to demonstrate the relationship between the learned quantities and the modal response of the original system. The problem considered was a three-element beam with stiffness $EI = 1.5$, length $l = 1.0$, density $\rho = 15.0$ and clamped ends, resulting in four DOFs (or an 8-dimensional state space). Proportional damping was defined as $\mathbf{C} = 1\text{e-}5\mathbf{K} + 1\text{e-}1\mathbf{M}$.

The network was trained with simulated data generated by applying a random vibration force input. The following results are for a case with a different random vibration force and random initial conditions. After training, the network is deployed as described in section 3.1.4. The runtime per time step evaluation was $2.72\text{e-}4$ seconds and the displacements mean squared error (MSE) was $3.27\text{e-}9$. The phase space response is shown in Figure 3.2, illustrating that the trained network is capable of predicting the response of the system without noticeable loss of accuracy after 3000 time steps.

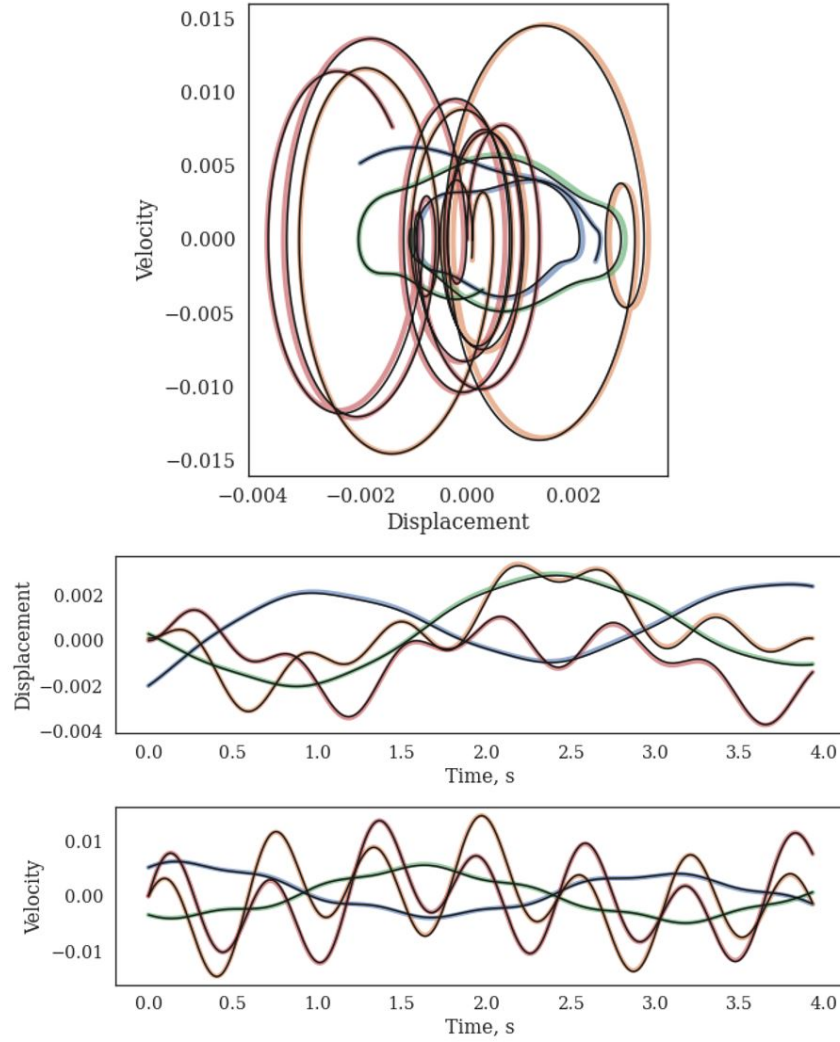


Figure 3.2. Phase space (top) and time (bottom) response of Euler-Bernoulli beam problem. Analytical response shown in color and network prediction shown in black.

As described in section 3.1, the trained network can be used to obtain the state matrices $\mathbf{M}$, $\mathbf{K}$, and $\mathbf{C}$ from the observed data. We used these matrices to perform modal analysis via the trained network. Figure 3.3 shows a comparison of the four natural frequencies computed from the network state matrices and those computed analytically. Moreover, the pseudo-orthogonality matrix was computed to compare the mode shapes. As illustrated in Figure 3.3, the pseudo-orthogonality matrix is the identity matrix, indicating that the modes computed by performing modal analysis with the trained network and the analytical model are the same. This result

demonstrates the network's capability to learn the modal characteristics of the system without having access or prior knowledge of any material or geometric parameters.

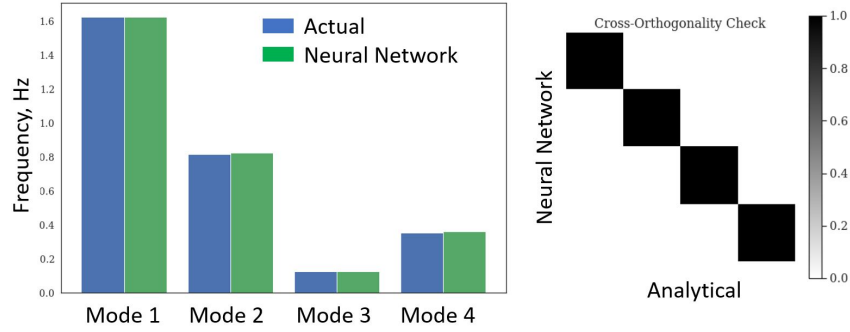


Figure 3.3. Modal frequencies (left) computed from analytical model (blue) and learned model (green), and cross-orthogonal check of analytical and predicted mode shapes (right).

3.2.2 Two Degree of Freedom System

The next example is a 2DOF system with cubic nonlinearities in the stiffness and damping terms as shown in Figure 3.4. This example demonstrates the applicability of the framework to nonlinear problems, as well as an example of domain shift identification. For all the results, the following parameter values were used: $m_1 = 1.1$, $m_2 = 1.5$, $k_1 = 1.0$, $k_2 = 1.3$, $k_3 = 0.8$, $k_{nl} = 10.0$, $c_1 = 0.01$, $c_2 = 0.08$, and $c_{nl} = 0.095$.

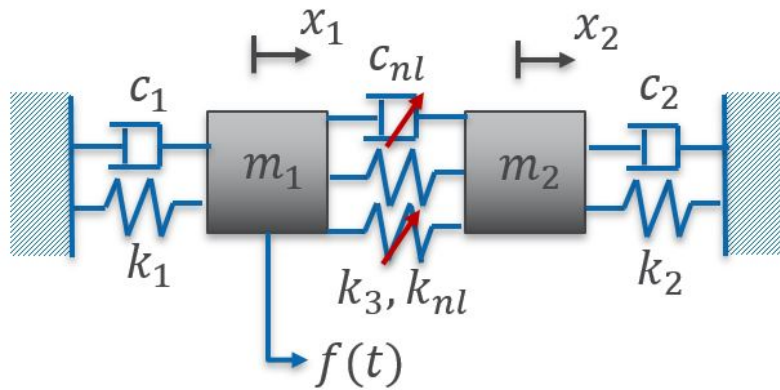


Figure 3.4. Diagram of two degree-of-freedom system example.

Table 3.1 summarizes the results for various cases considered. Figure 3.5 presents the corresponding phase plots illustrating how there is no apparent loss of accuracy after 3000 time

steps. The results shown in Table 3.1 and Figure 3.5 were computed by time integrating the trained neural network as described in Section 3.1.4 with random initial conditions. For the forced vibration cases, the network was trained with a random signal generated by summing 100 harmonics with random phase and random amplitude, and it was then tested with a different random signal.

To understand what the neural networks are learning, we looked at the undamped linear case. Figure 3.6 shows a comparison of the learned and analytical time-varying energies. As illustrated, the learned energies do not correspond exactly to the physical quantities even though the constant Hamiltonian is correctly learned for this undamped case. Furthermore we can look at the eigenvectors of the corresponding $\mathbf{M}$ and $\mathbf{K}$ to understand how the learned and analytical state matrices vary. As seen in Figure 3.7, the learned state matrices are transformations of the physical quantities. Moreover, Figure 3.8 shows the eigenvalues and eigenvectors of $\mathbf{M}^{-1}\mathbf{K}$. The eigenvalues are presented as a function of time since the network is allowed to learn a time-varying $\mathbf{M}$ and $\mathbf{K}$, but it learns the correct constant behavior for the linear case. As shown, the neural network learns to reproduce the eigenvalues and eigenvectors exactly. We can then determine that the network only needs to reproduce the state space representation of the dynamic system and not necessarily the exact energy terms. This result makes sense since we do not supervise the energy terms directly, as these are assumed to be unknown because they cannot be easily measured. This analysis provides insight into what the neural network is learning to reproduce the dynamic response of the observed system. We see that the network is learning an equivalent dynamic system that preserves important physical properties, such as the state space representation.

Domain Shift Identification

To demonstrate how the trained network can be used to identify a domain shift in new data, we started with the trained network corresponding to the system where $c_{nl} = 0.0$. We confronted the network with new data generated from a dynamic system with $c_{nl} = 0.095$ and

Table 3.1. Summary of Neural Network Performance

Terms set to 0.0	Runtime per evaluation (s)	Displacement MSE
$c_1, c_2, c_{nl}, k_{nl}, F_{ext}$	9.24e−9	0.22e−3
$c_1, c_2, c_{nl}, F_{ext}$	1.29e−5	1.19e−3
c_{nl}, F_{ext}	2.12e−5	1.15e−3
c_{nl}	6.76e−6	2.11e−3
F_{ext}	1.01e−5	0.58e−3
—	1.38e−5	2.19e−3

ran the training algorithm for 5000 additional iterations (this takes 5 minutes).

A snapshot of the state matrices from the original and trained model are shown in Figure 3.9. As illustrated, the only matrix that changed is **C** while **M** and **K** remained unchanged. To further illustrate this change, the eigenvectors of the state matrices were computed as a function of time and are shown in Figure 3.10. As illustrated, the eigenvectors of **M** and **K** are unchanged but the eigenvectors of **C** have been transformed. Specifically, the original network exhibits eigenvectors that are orthogonal to each other which indicates that the two masses are uncoupled through damping, which is the actual physical state of the original problem. In the updated model, the eigenvectors show some degree of coupling between the two masses, which is consistent with the update performed.

Lastly, the shift was quantified by computing the variance of the residual between the updated and original weights corresponding to the kinetic, potential, and Rayleigh damping networks which is 1.88e−8, 2.44e−5, and 1.36e−3, respectively. This result indicates that the damping network weights underwent the largest changes during training. All of these results can be used to identify the nature of the domain shift in new data. Future work will focus on automatic domain shift detection for automated model updating.

3.2.3 Rigid Link with Suspended Mass

The next example demonstrates the use of the autoencoder to perform coordinate transformation from a Cartesian coordinate system that has redundant coordinates due to the kinetic

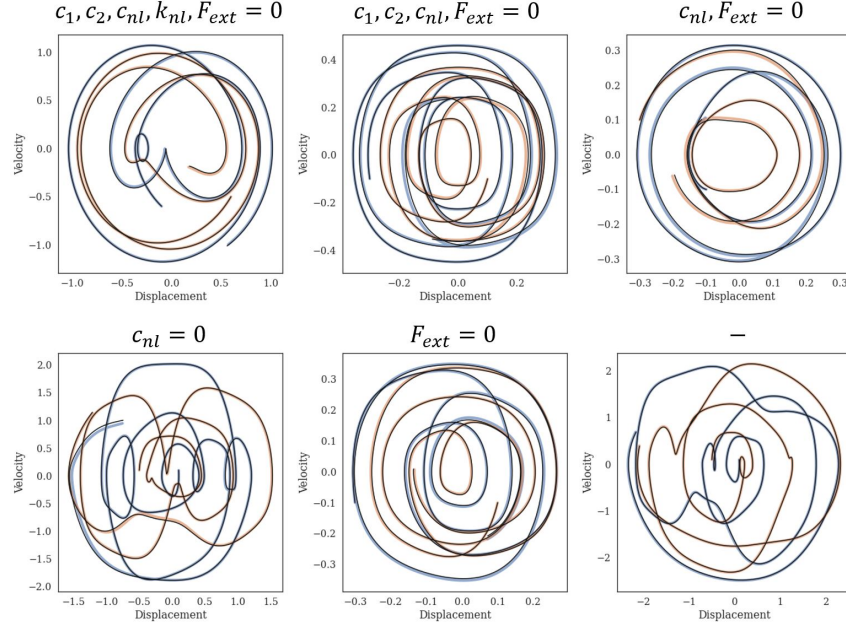


Figure 3.5. Summary of phase space response for the various cases considered. Analytical response shown in color and network prediction shown in black.

constraints of the problem and, as a result, does not constitute a generalized coordinate system. The problem is illustrated in Figure 3.11. The parameters used in this example were $a = \frac{1}{3}$, $b = 1.1$, $m = 4.0$, $M = 2.0$, $k = 100.0$, $c = 0.1$.

Coordinate Transformation

To demonstrate the coordinate transformation autoencoder, we assume that we measure the response of the rigid link structure at points B and the response of the mass M relative to the point A in a Cartesian coordinate system (i.e., only in the x- and y-directions). However, this choice of coordinates does not constitute a generalized coordinate system because they are redundant given the kinematic constraints of the problem. Instead, there is only two independent coordinates that need to be discovered by the autoencoder. A very natural choice of coordinates would be the rigid link angle θ , and the vertical displacement y of the mass M relative to point A . We did not provide the network with any inductive biases regarding these coordinate and instead we let the autoencoder discover its own set of generalized coordinates.

The autoencoder hyperparameters (i.e., number of layers and hidden units) were de-

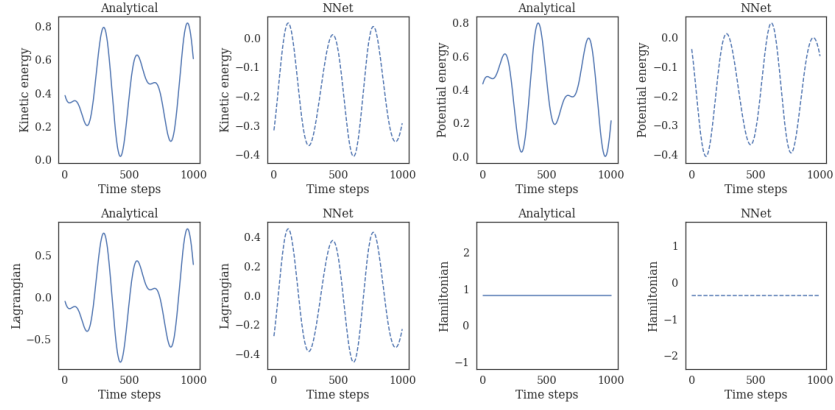


Figure 3.6. Analytical and predicted energy terms for undamped linear case.

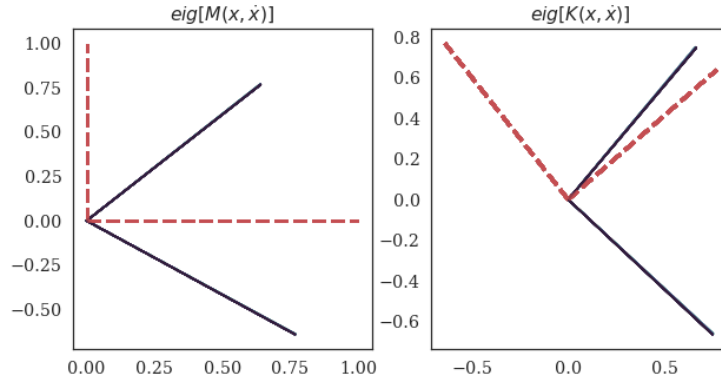


Figure 3.7. Eigenvectors of analytical (dashed red) and predicted (solid black) $\mathbf{M}$ (left) and $\mathbf{K}$ (right).

terminated by running a grid search and the results are presented in the Appendix A.1. The network was trained with data generated with Mathematica by numerically solving the problem by prescribing random initial conditions and a random vibration input. The results presented here are for different set of random initial conditions and random vibration input. Figure 3.12 shows the reconstructed phase space response. The runtime per evaluation was $1.42\text{e}-2$ seconds and the displacement MSE was $2.52\text{e}-5$. The system is solved by running the Runge-Kutta solver in the learned latent space and then projecting it back to the original measured coordinate system (i.e., the assumed Cartesian reference frame).

The learned latent space is not constrained in any way to correspond to the identified

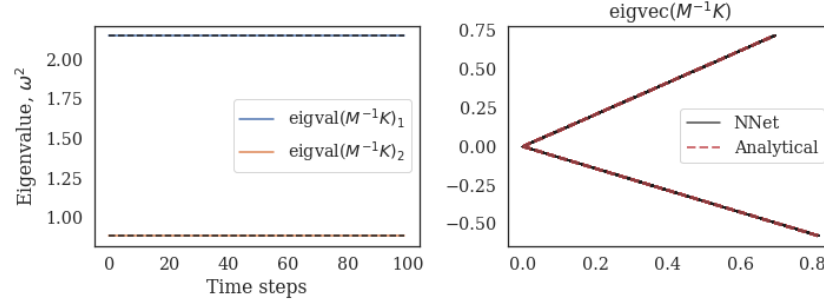


Figure 3.8. Eigenvalues (left) and eigenvectors (right) of $M^{-1}K$.

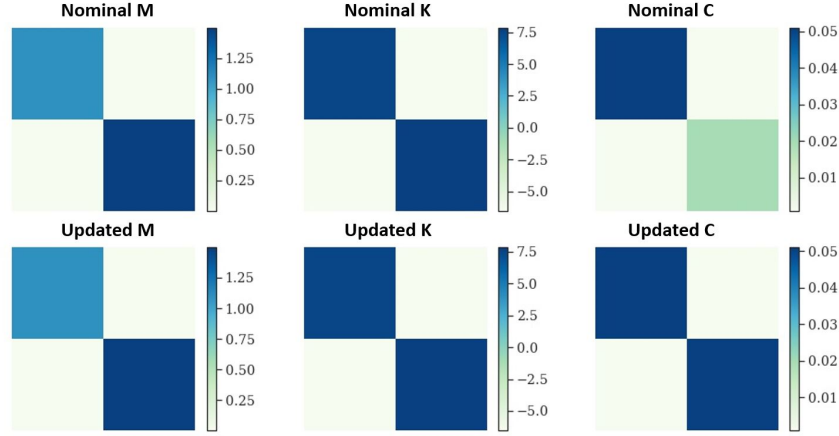


Figure 3.9. State matrices of linear damping model (top) and updated cubic damping model (bottom).

generalized coordinates (θ, y) . As a result, the autoencoder projects to an alternative set of generalized coordinates. Figure 3.13 illustrates how the corresponding $\mathbf{M}$ and $\mathbf{K}$ matrices have been transformed by looking at the matrix eigenvectors. For instance, the original system has an uncoupled $\mathbf{K}$ matrix (illustrated by the orthogonal eigenvectors) but the learned system does not. On the other hand, the original $\mathbf{M}$ matrix is heavily coupled but the learned latent space is only weakly coupled. Interestingly the learned state space is topologically similar to the original state space, as shown in Figure 3.14.

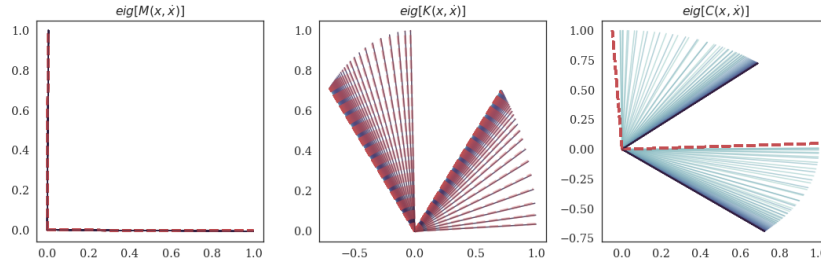


Figure 3.10. Eigenvectors of state matrices $\mathbf{M}$ (left), $\mathbf{K}$ (middle), and $\mathbf{C}$ (right), computed from original network (dashed red) and updated network (solid blue). The hue of the eigenvectors indicates the variation in time.

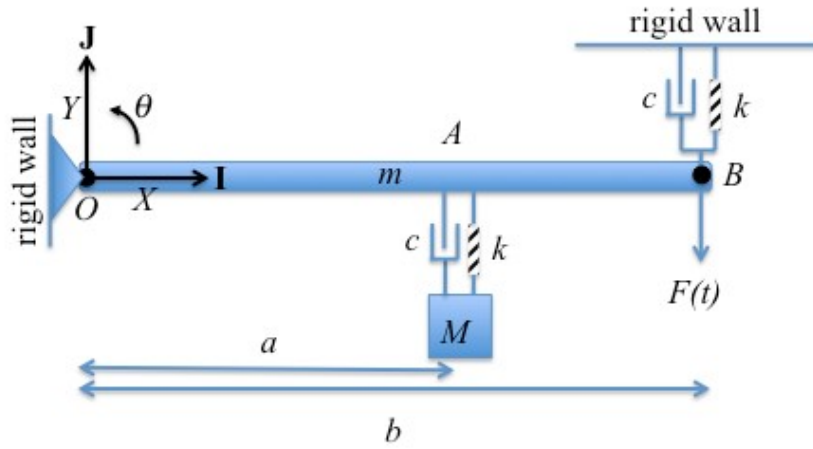


Figure 3.11. Rigid link with suspended mass problem.

3.2.4 System Identification of a Duffing Oscillator

To demonstrate how the approach may be used in system identification tasks, a single degree-of-freedom duffing oscillator system is considered. The system is defined by the nonlinear ordinary differential equation

$$\ddot{q} + 0.05\dot{q} + 1.2q + 5.0q^3 = f(t) \quad (3.26)$$

The neural network corresponding to the term $\mathcal{G}$ in Eq. 3.16 had three layers of four units each for all the energy terms. Fifth-degree polynomials were assumed for each of the energy terms. The network was trained with a single realization obtained by exciting the oscillator with

a random signal with a peak amplitude of 0.01. The identified parameters for the energy terms are presented in graphical form in Fig. 3.15. As shown, there is excellent agreement between the identified parameters and the actual values. The largest discrepancy is in the damping term with a relative error of 2%. The network correctly identified the appropriate non-zero terms. All of the neural network coefficients were on the order of 10^{-6} , which demonstrates the network's capability to prune terms that do not fit the physical formulation.

To demonstrate the network's capability to generalize, the trained system was used in a multi-harmonic balance algorithm [195] to obtain the nonlinear forced response of the system. As shown in Fig. 3.16, the network was able to predict an accurate response amplitude even at large amplitudes. The network was originally trained on a random signal input with a peak amplitude of 0.01 but was able to predict an accurate response at three times the training excitation amplitude as well as capture the nonlinear trends at this higher amplitude. This result is not surprising given the fact that the system identification was very accurate and, as a consequence, the trained network is a good representation of the actual physical system.

3.3 Conclusion

This chapter presented an original approach for obtaining a physically interpretable data-driven model of a structural dynamics system. The approach is based on enforcing structure-preserving properties of an ensemble of neural networks that learn the kinetic and potential energies, as well as the Rayleigh dissipation terms from observed responses. These neural networks promote parsimonious models through regularization and inductive priors imposed via a dictionary of polynomial terms. Furthermore, an autoencoder is used to perform coordinate transformation to map arbitrary coordinate systems to generalized coordinates. This transformation is also physically constrained through the definition of generalized coordinates that necessitates a very particular transformation to generalized forces. In addition, a numerics-informed loss term was added to ensure the trained network could be time integrated with traditional ODE

solvers. The trained ensemble was then deployed by extracting the learned differential operator and exercised in state space. An initial demonstration of how the results can be used to detect domain shift was provided. A more thorough and complete exploration of this feature will be explored in future work. The results presented in this chapter show that the approach is robust and does not suffer from excessive error accumulation. It has also been found that limited data (i.e., a single time realization containing two oscillation cycles) may be sufficient for training the network to be used outside of its training regime (i.e., different initial conditions and external forcing). These results are a direct consequence of the physical and numerical constraints, including the time integration residual loss term. So far, ideal conditions have been assumed (i.e., ideal sampling rate, noise-free systems, etc.). Future work will focus on addressing practical challenges encountered in real experimental and deployed systems. Furthermore, future work will extend the method to larger systems and to be fully Bayesian for use in a probabilistic reasoning framework to inform structural health diagnosis and prognosis, as well as focusing on hysteretic systems which will need enhancing the neural network architectures to consider history dependence.

3.4 Preview of Chapter 4

Chapter 3 introduced the general formulation of an ensemble of neural networks that work together to learn the geometric structure of the underlying dynamic space from data. The method was demonstrated on low-dimensional systems where the observables constituted the dynamic state space. In general, data obtained via measurement sensors may not represent the underlying state space, making it impossible to enforce physics directly. To overcome this challenge, Chapter 4 presents a strategy that relies on embedded time delays of measured signals to uncover the state space via a diffeomorphic mapping learned with an invertible neural network. The results presented in Chapter 4 show how the structure-preserving formulation can be applied to cases where only partial observables are available.

3.5 Remarks

Portions of this dissertation have been published or are being considered for publication. Chapter 3 is composed of the following first authored publications:

David A. Najera-Flores and Michael D. Todd. A structure-preserving neural differential operator with embedded Hamiltonian constraints for modeling structural dynamics. *Computational Mechanics*, 72:241–252, March 2023

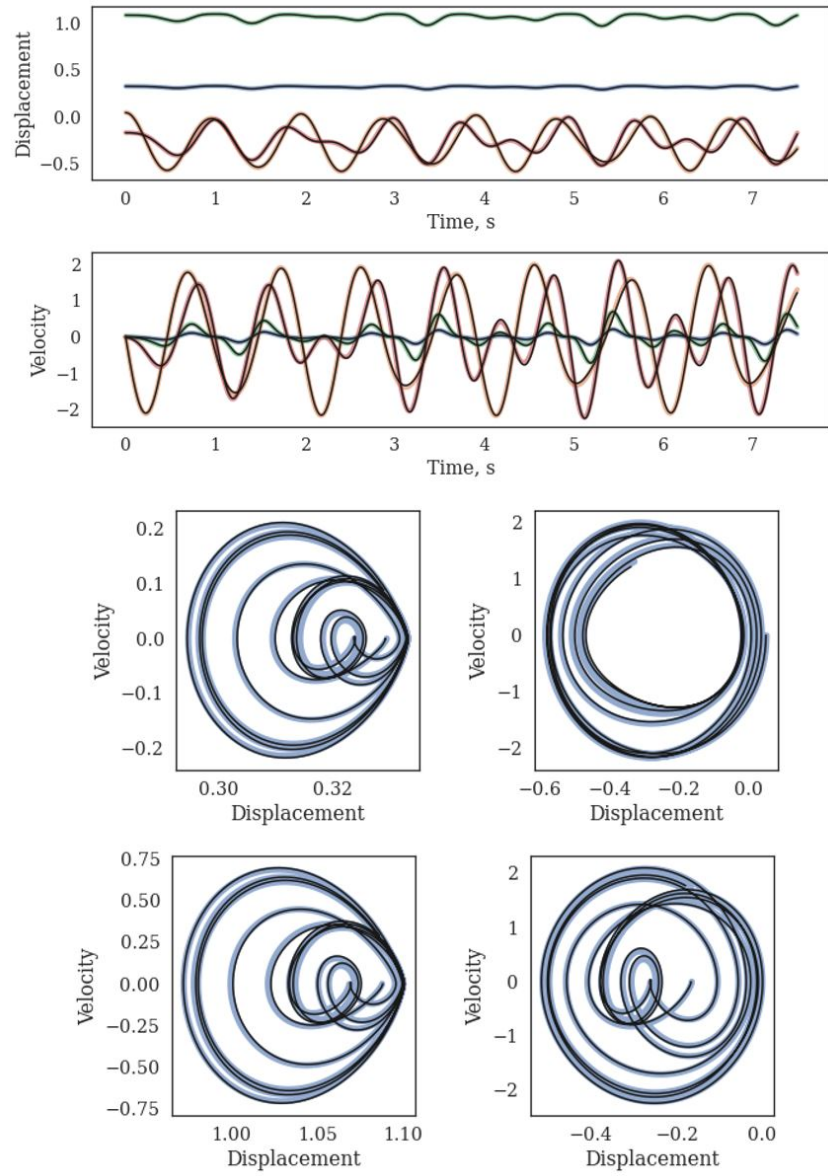


Figure 3.12. Time (top) and phase space (bottom) response of the rigid link problem. Analytical response shown in color and network prediction shown in black.

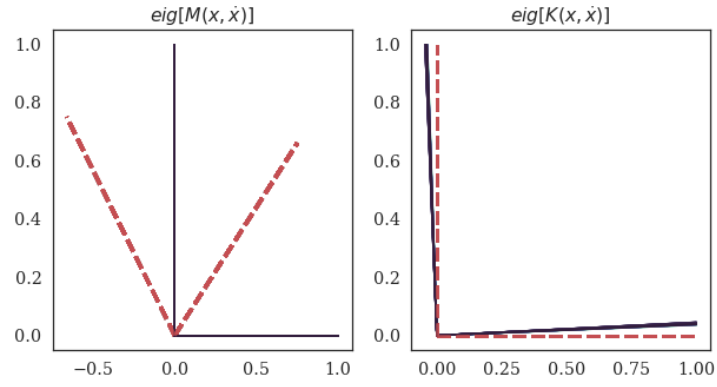


Figure 3.13. Eigenvectors of analytical (dashed red) and predicted (solid black) $\mathbf{M}$ (left) and $\mathbf{K}$ (right).

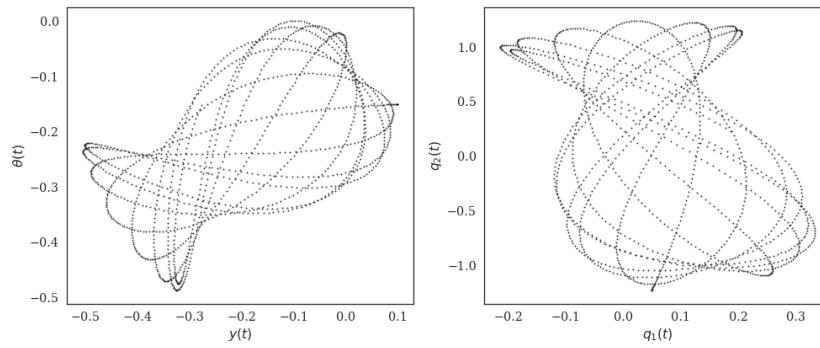


Figure 3.14. State space representation of original (left) and learned system (right).

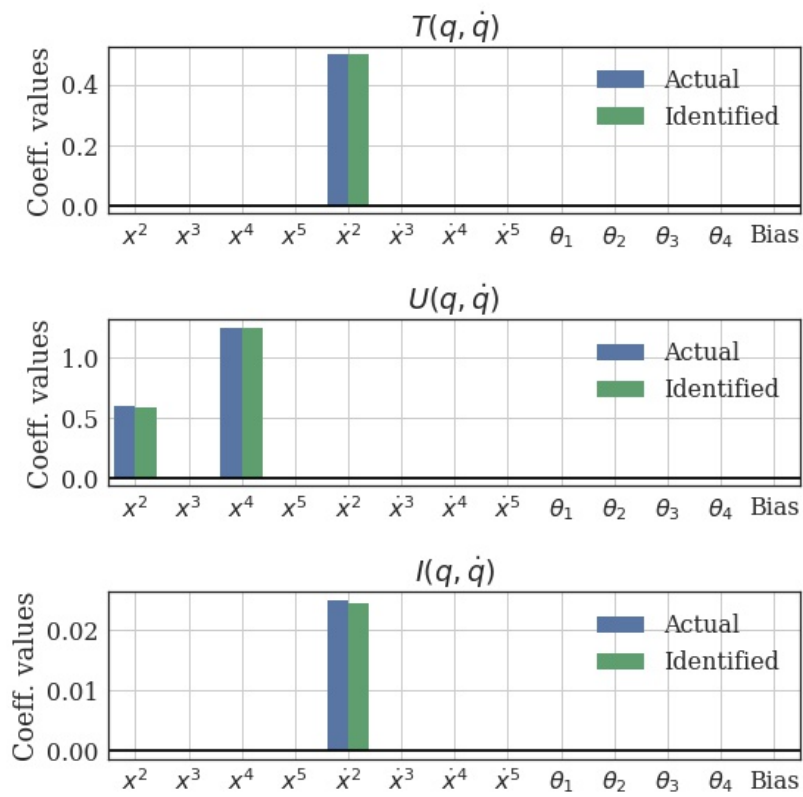


Figure 3.15. Comparison between identified and actual parameters in the duffing oscillator system energy terms.

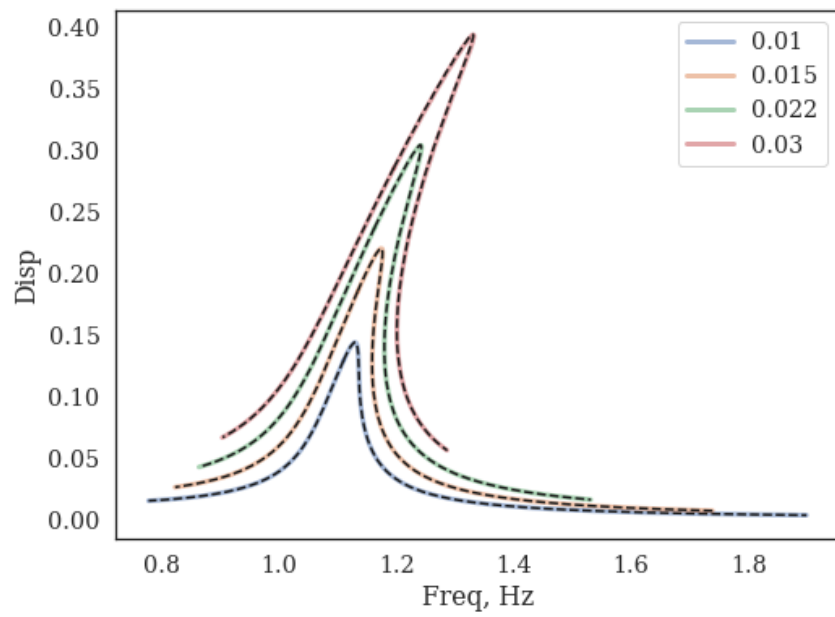


Figure 3.16. Nonlinear forced response of the actual system (solid colored) and the trained network (dashed black) at different excitation amplitudes.

Chapter 4

State-Space Reconstruction from Partial Observables Using an Invertible Neural Network with Structure-Preserving Properties for Nonlinear Structural Dynamics

4.1 Introduction

This chapter builds upon the work presented in Chapter 3. As motivated previously, complex physical systems often rely on digital models to inform their design cycle and maintenance schedule. Chapter 3 introduced a structure-preserving approach that stands in contrast to purely data-driven modeling approaches, which are popular when experimental data are available. These data-driven approaches range from more traditional techniques such as singular value or proper orthogonal decomposition [119, 16, 17] to more recent work with neural networks, applied to areas such as climate modeling [168, 128], material modeling [266, 276, 116], and structural dynamics [236, 274]. Nonlinear system identification also typically falls under this category where models are learned through a variety of techniques from experimental data as described in detail in [120, 189]. However, purely data-driven methods generally lack physical interpretability and are limited in the domain they may operate in (e.g., not outside their training domain), resulting in a lack of robustness. More recently, physics-informed neural networks (PINNs) have become popular in the literature, ranging from applications in solving classical

partial differential equations (PDEs) [203, 121, 277], including the Navier-Stokes equation [113], as well as applications in geostatistical modeling [290], defect detection [281], material modeling [63, 90, 89], and solution of non-homogeneous dynamic systems and elasticity problems [27], to larger scale systems including hybrid mesh-free methods [237, 93, 153, 284, 10]. These approaches typically operate by transforming the PDE residual into a loss term, as well as enforcing boundary and initial conditions with similar penalty methods, either through the strong or weak form of the PDE.

In contrast with physics-informed methods, structure-preserving approaches seek to enforce physical constraints through inductive biases built into the data-driven mechanism, e.g., the neural network architecture or in reduced order modeling via an appropriate choice of reduction basis [217, 197, 137]. Structure-preservation can be performed by prescribing the form of the model *a priori* through discovery of a symbolic expression [20, 28, 98]. Other approaches uncover the underlying spectral submanifolds from data and construct data-driven models assuming a normal form [85, 37, 38, 9]. While these models preserve a type of structure, other methods are focused on explicitly preserving the underlying Lagrangian or Hamiltonian system structure [226, 111, 164, 80, 212]. Many of these approaches require access to the generalized or canonical coordinates of the observed system. The authors previously designed a framework to transform from arbitrary coordinate systems to generalized coordinates while performing structure preservation [183]. However, that work assumed that the state-space was fully observable, which is not always the case when only partial observations are available. Recent work has leveraged Takens theorem [242] to extract spectral submanifolds from the data [37, 38, 9] and for discovery of partial differential equations [12, 208].

The present work is built on the idea that the underlying state space may be uncovered from partial observables via time-delayed embedding. Moreover, this chapter is focused on learning models that preserve the underlying geometric structure of the dynamic system. In contrast with the previous chapter, this chapter develops a generalized framework for discovering a diffeomorphic mapping between time-delayed embeddings of partial observables to a latent

space in generalized coordinates where structure preservation is enforced via invertible neural networks which preserve a bijective transformation. The approach is demonstrated on two computational examples and two experimental datasets. Section 4.2 describes previous work to establish the foundation upon which this work is built. Section 4.3 introduces the proposed approach while section 4.4 go over the details of the algorithmic implementation. Section 4.5 presents results and discussion of these while section 4.6 describes some of the caveats and challenges associated with the implementation. Finally, section 4.7 provides concluding remarks and future research directions.

4.2 Background

Section 4.2.1 reviews previous work by the authors that serves as the foundation for this work while section 4.2.2 provides a quick review of the theory of time-delayed embedding.

4.2.1 Structure-Preserving Neural Network

This work leverages a framework developed by the authors in [183] to construct a structure-preserving neural differential operator from data. This workflow requires data samples in generalized coordinates to enforce a Hamiltonian structure to the data-driven model. The framework proposed in [183] models a general non-conservative Lagrangian system. The workflow relies on learning the kinetic energy $T_{nn}(\mathbf{q}, \dot{\mathbf{q}}) : \mathbb{R}^{2d} \rightarrow \mathbb{R}^1$, the potential energy $U_{nn}(\mathbf{q}) : \mathbb{R}^d \rightarrow \mathbb{R}^1$, and the Rayleigh damping term $I_{nn}(\dot{\mathbf{q}}) : \mathbb{R}^d \rightarrow \mathbb{R}^1$ where d is the dimensionality of the generalized coordinates. These quantities are assumed to be a function of the generalized displacements $\mathbf{q}$ and generalized velocities $\dot{\mathbf{q}}$. Each of these terms is modeled with a multilayer perceptron (MLP) that is augmented with complete polynomial terms. As an example, the kinetic energy function is defined as

$$\begin{aligned}
T_{nn} = & \sum_{i,j,\dots,p}^{i+j+\dots p=M} \alpha_{i,j,\dots,p} \dot{q}_1^i \dot{q}_2^j \dots \dot{q}_k^p \\
& + \beta_{i,j,\dots,p} \dot{q}_1^i \dot{q}_2^j \dots \dot{q}_k^p \\
& + \sum_n^N \delta_n \mathcal{G}^{(n)}(\mathbf{q}, \dot{\mathbf{q}}),
\end{aligned} \tag{4.1}$$

where the coefficients α , β , and δ are learned during training, M is the degree of the polynomial, and the function $\mathcal{G}^{(n)}$ represents an MLP. In general, a fully connected neural network is used to define $\mathcal{G}^{(n)}$. For conciseness, only the kinetic energy function is illustrated but the potential and Rayleigh damping are modeled similarly. The learned ensemble of networks is used to derive the Lagrangian and Hamiltonian of the system, from which the generalized accelerations are computed as

$$\ddot{\mathbf{q}}_n = \mathbf{M}_{nn}^{-1}(\dot{\mathbf{p}}_i - \mathbf{M}_{ij,k} \dot{q}_k \dot{q}_j), \tag{4.2}$$

where $\mathbf{M}$ is the generalized inertia matrix (obtained from the learned kinetic energy) and $\dot{\mathbf{p}}$ is the momentum rate (obtained from relating the Hamiltonian to the non-conservative forces acting on the system). To compute the inverse of the generalized inertia matrix, a Moore-Penrose inverse is used to avoid numerical issues during the initial stages of training.

The details of the implementation are presented in the original article [183] and the algorithmic implementation is described in Algorithm 2.

4.2.2 State space reconstruction with time delay embedding

The data-driven, structure-preserving approach described in section 4.2.1 requires that the inputs to the neural networks be in generalized coordinates. However, this requirement is not guaranteed to be satisfied when the data is coming from sensors providing partial observations of the dynamic system. In such cases, space reconstruction techniques can be used to recover

predictive state space models from the partial observables. A well-known approach is the Method of Delays developed by Takens [242], which can reconstruct a state space that is topologically equivalent (i.e., diffeomorphic) to the original dynamic system. The concept of time delay embedding for state space reconstruction is derived from Taken's theorem which asserts that, in the absence of noise, the underlying dynamics of the system may be recovered from a time delay embedding of dimension m with $m > 2d + 1$, where d is the original system's dimension (which is often unknown *a priori*). The time delay embedding consists of lagged copies of the observed response vector, $x(t)$, such as

$$x(t) = [x(t), x(t - \tau), \dots, x(t - (m - 1)\tau)], \quad (4.3)$$

where τ is a uniform time delay.

A number of challenges associated with the practical application of Taken's theorem were described by Pecora et al. [192]. These challenges include the selection of embedding parameters, size and number of delays (τ), and embedding dimension m . Pecora et al. [192] proposed an approach consisting of employing statistical tests to determine these heuristic parameters in a systematic way. In Pecora's work, a continuity statistic is used to determine the functional independence of new candidate components to the reconstruction vector. To accomplish this, Pecora et al. proposed a k -nearest neighbor (k-NN) approach that tests whether the points considered around a fiducial point are within a certain range by chance or due to a functional relationship between the new candidate component and the existing reconstruction vectors. The details of the implementation of the algorithm are provided in the original paper [192].

Kraemer et al. [127] subsequently identified several challenges with the implementation of Pecora's algorithm. One of the most significant challenges was the choice of an optimal delay τ given that the definition of relative maximum, which is used to determine τ , is not obtained systematically. To address this challenge, Kraemer et al. proposed combining Pecora's method

with an approach proposed by Uzal et al. [251]. Uzal’s approach consists in employing an objective cost function known as the L-statistic to quantify the goodness of reconstruction. The details of the implementation of their algorithm are provided in the original paper [251].

The method proposed by Kraemer et al. is known as PECUZAL (a combination of Pecora’s and Uzal’s names). This new method combines the continuity statistic and the L-statistic to simultaneously determine the size and number of delays that optimize both Pecora’s and Uzal’s objectives. The algorithm operates recursively by first preselecting the potential delays and time series to use in each embedding cycle, and then picking a relative maximum according to the L-statistic. Specifically, a series of time delay candidates are proposed by applying Pecora’s approach while simultaneously computing their associated L-statistic. The combination of time delay and time series that yields the minimum change of L-statistic with respect to the current reconstruction vectors is then added as a new reconstruction vector. The steps are repeated until the L-statistic cannot be minimized any further. The detailed steps are provided in the original paper [127]. This approach provides an automated way to build a reconstruction vector from partial observables.

4.3 Coordinate transformation

The time delay embedding structure found by applying the PECUZAL algorithm are presumed to be topologically equivalent to the underlying state space of the dynamic system, according to Taken’s theorem. However, the diffeomorphic mapping from the time delay embedding to the underlying state space is not clearly defined yet. To this end, a neural network is trained to learn the appropriate mapping from time-delayed data to the generalized coordinate system. Two options are explored: (1) a normalizing flow network used as an autoencoder and (2) a general multilayer perceptron (MLP) used as an autoencoder. Both of these options are used to encode the embeddings into a generalized coordinate system and decode it back to the original coordinate space. The difference between these approaches is that the normalizing flow

is invertible and thus bijective which is in line with the theory of a diffeomorphic mapping. The MLP architecture is non-invertible but it is more general and does not require any assumptions about the structure of the mapping with the disadvantage that the inverse mapping also has to be learned (i.e., the decoder).

4.3.1 Normalizing flow autoencoder

The normalizing flow architecture is based on the work by Dinh et al. [59]. Let $\mathbf{x} \in \mathbb{R}^D$ be the reconstruction vector composed of time delay embeddings (augmented with the original observations) in D -dimensional space. An order reduction is first achieved by performing a linear projection as

$$\tilde{\mathbf{z}} = \mathbf{V}^\top \mathbf{x}, \quad (4.4)$$

where $\mathbf{V} \in \mathbb{R}^{D \times d}$ is the truncated projection matrix obtained through singular value decomposition (SVD). The reduced dimension d is chosen to be the embedding dimension determined by the magnitude of the singular values. A second linear transformation is performed to allow for linear mixing of the reduced-order components before passing it to the normalizing flow. The second linear transformation is performed as

$$\mathbf{z} = \mathbf{T}\tilde{\mathbf{z}}, \quad (4.5)$$

where $\mathbf{T} \in \mathbb{R}^d$ is a coefficient matrix that is learned during training.

The $\mathbf{z}$ vector is then passed to a normalizing flow based on the architecture proposed in [59]. The normalizing flow $f_\theta : \mathbf{z} \rightarrow \mathbf{q}$ is composed of a stacked sequence of nonlinear bijective functions with learnable parameters θ . Following the work by Dinh, two affine coupling layers (ACL) were stacked to keep the computation of the Jacobian determinant of the transformation tractable and efficient. As shown by Dinh, this particular architectural choice results in a lower triangular Jacobian matrix, $\frac{\partial f(\mathbf{z})}{\partial \mathbf{z}}$ which is desirable for efficient computation of its determinant.

The ACL requires the split of the input and output vectors into two halves: $\mathbf{z} = [\mathbf{z}_A, \mathbf{z}_B]$ and $\mathbf{q} = [\mathbf{q}_A, \mathbf{q}_B]$. Given this split, the forward operations (in the normalizing direction) are

$$\mathbf{q}_A = \mathbf{z}_A \odot \exp(s_B(\mathbf{z}_B)) + t_B(\mathbf{z}_B), \quad (4.6)$$

$$\mathbf{q}_B = \mathbf{z}_B \odot \exp(s_A(\mathbf{z}_A)) + t_A(\mathbf{z}_A), \quad (4.7)$$

where $\odot$ represents the element-wise multiplication operator, $s_A(\cdot)$ and $s_B(\cdot)$ represent nonlinear scaling functions, and $t_A(\cdot)$ and $t_B(\cdot)$ represent nonlinear translation functions. These nonlinear scaling and translation functions define the affine transformation and are modeled with multilayer perceptrons in this work. Inverting the normalizing flow direction results in the following expressions, in the order they would be computed:

$$\mathbf{z}_B = (\mathbf{q}_B - t_A(\mathbf{q}_A)) \odot \exp(-s_A(\mathbf{q}_A)), \quad (4.8)$$

$$\mathbf{z}_A = (\mathbf{q}_A - t_B(\mathbf{q}_B)) \odot \exp(-s_B(\mathbf{q}_B)). \quad (4.9)$$

The decoding operation can then be completed by inverting the operations in Eq. 4.4 and 4.5 as

$$\tilde{\mathbf{z}} = \mathbf{T}^{-1}\mathbf{z}, \quad (4.10)$$

and

$$\hat{\mathbf{x}} = \mathbf{V}\tilde{\mathbf{z}}, \quad (4.11)$$

where $\hat{\mathbf{x}}$ is the reconstruction of the original time delay embeddings.

4.3.2 General MLP autoencoder

As an alternative to the normalizing flow architecture, a general non-invertible MLP architecture is also studied. The reason for studying this alternative approach is that the INN

architecture may be too restrictive. For instance, dimension contraction and expansion are not possible with the INN because the dimensionality of the layers must be kept constant. As with the INN, the task is that of mapping the time delay embedding to the generalized coordinate system via an encoding $f_\pi : \mathbf{x} \rightarrow \mathbf{q}$ and back to the physical space via decoding $f_v : \mathbf{q} \rightarrow \mathbf{x}$, where θ and v are the learnable hyperparameters of the encoder and decoder, respectively. Both of these are constructed by performing a nonlinear projection, where a transformation matrix $\mathbf{T} \in \mathbb{R}^{D \times d}$ is learned from an MLP. As shown in previous work [183], this form of the autoencoder imposes additional physical constraints on the neural networks and helps with convergence. The transformation is then performed as

$$\mathbf{q} = \mathbf{T}_\theta(\mathbf{x}) \cdot \mathbf{x} \quad (4.12)$$

$$\hat{\mathbf{x}} = \mathbf{T}_v(\mathbf{q}) \cdot \mathbf{q}, \quad (4.13)$$

where $\hat{\mathbf{x}}$ is the reconstruction of the original time delay embedding.

The chain rule is then used to transform back and forth between the time delay embeddings and the generalized coordinates as

$$\dot{q}_i = \sum_{j=1}^n \frac{\partial q_i}{\partial x_j} \cdot \dot{x}_j, \quad (4.14)$$

$$\ddot{q}_i = \sum_{j=1}^n \frac{\partial q_i}{\partial x_j} \cdot \ddot{x}_j + \sum_{j=1}^n \frac{\partial \dot{q}_i}{\partial x_j} \cdot \dot{x}_j. \quad (4.15)$$

Similarly, the inverse operation is performed as

$$\dot{x}_j = \sum_{i=1}^k \frac{\partial x_j}{\partial q_i} \cdot \dot{q}_i, \quad (4.16)$$

$$\ddot{x}_j = \sum_{i=1}^k \frac{\partial x_j}{\partial q_i} \cdot \ddot{q}_i + \sum_{i=1}^k \frac{\partial \dot{x}_j}{\partial q_i} \cdot \dot{q}_i. \quad (4.17)$$

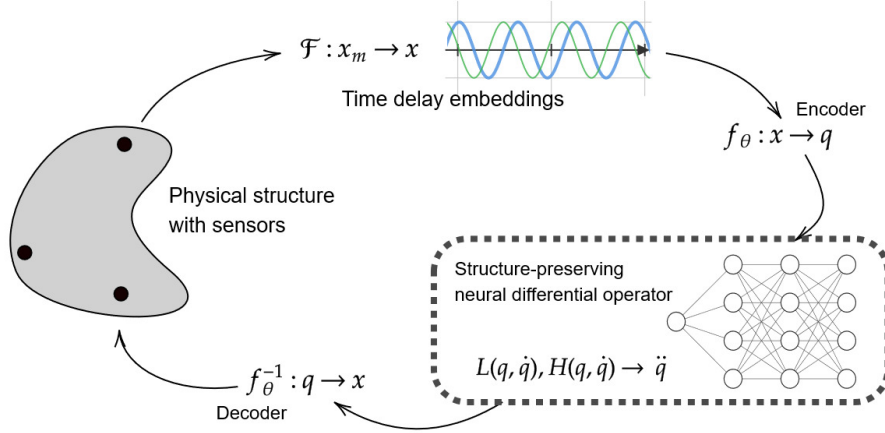


Figure 4.1. Schematic of complete workflow.

4.4 Neural network training and deployment

The entire workflow is illustrated in Fig. 4.1 and described in Algorithm 2. The neural network training is performed by adding all the loss terms considered. These terms are described in this section. Details for the hyperparameters and training settings are provided in Appendix B.1.

The first loss term corresponds to the structure-preserving architecture which is supervised by predicting the generalized accelerations

$$\mathcal{L}_{struct} = \|\ddot{\mathbf{q}} - \hat{\ddot{\mathbf{q}}}\|_2^2, \quad (4.18)$$

where $\ddot{\mathbf{q}}$ is the vector of encoded accelerations and $\hat{\ddot{\mathbf{q}}}$ are the computed accelerations from Eq. 4.2. In addition, the following other physical constraints are included to ensure self-consistency:

$$\begin{aligned}
\mathcal{L}_{phys} = & \|T_{nn}(\mathbf{q}, \dot{\mathbf{q}}) - 1/2\mathbf{q} \cdot \mathbf{p}\|_2^2 \\
& + \|H(\mathbf{q}, \dot{\mathbf{q}}) - (T_{nn}(\mathbf{q}, \dot{\mathbf{q}}) + U_{nn}(\mathbf{q}, \dot{\mathbf{q}}))\|_2^2 \\
& + \|\mathbf{p} - \mathbf{M}\dot{\mathbf{q}}\|_2^2.
\end{aligned} \tag{4.19}$$

Since time is not an explicit input to the network, a numerics-informed constraint was implemented to ensure that time derivatives are correctly predicted. This constraint consisted of the residual of a multi-step integration step. The general loss term has the following form:

$$\mathcal{L}_{diff} = \left\| \sum_{j=0}^s \gamma_j \begin{pmatrix} \mathbf{q}_{k+j} \\ \dot{\mathbf{q}}_{k+j} \end{pmatrix} - h \sum_{j=0}^s \eta_j \begin{pmatrix} \dot{\mathbf{q}}_{k+j} \\ \ddot{\mathbf{q}}_{k+j} \end{pmatrix} \right\|_2^2, \tag{4.20}$$

where γ and η are coefficients from the Butcher tableau corresponding to the Adams-Bashforth linear multistep integration method, and h is the time step associated with the training data. The $\mathbf{q}_{k+j}$ term represents the $(k+j)_{th}$ time step in the $\mathbf{q}$ vector. As described in [183], the same general form of this loss function is used for the momentum and its corresponding time derivative.

And lastly the reconstruction term was the same for both autoencoder architectures considered:

$$\mathcal{L}_{recon} = \|\mathbf{x}_m - \hat{\mathbf{x}}_m\|_2^2, \tag{4.21}$$

where $\mathbf{x}_m$ is a subset of the time delay embeddings corresponding to the original measurements (i.e., not the time-delayed copies). Due to the symmetry of the autoencoder, the entire time delay embedded matrix is reconstructed but only the original measurements are supervised because reconstruction of the time-delayed copies is not required. So the total loss is constructed by adding all these terms as

$$\begin{aligned}\mathcal{L} = & \mu_1 \mathcal{L}_{struct} + \mu_2 \mathcal{L}_{phys} + \\ & + \mu_3 \mathcal{L}_{diff} + \mu_4 \mathcal{L}_{recon},\end{aligned}\tag{4.22}$$

where the μ_i coefficients are tuned manually to scale the different loss terms to have the same order of magnitude.

Since the goal of this model is to act as a digital twin in an online setting, the ensemble of neural networks is deployed as an ordinary differential equation (ODE) operator during the inference stage. The model requires a set of initial conditions, $\mathbf{x}_0$. The first step is to use the trained encoder to obtain the initial conditions in transformed generalized coordinates, $\mathbf{q}_0$ and $\dot{\mathbf{q}}_0$ using Eq. 4.12 and 4.14. Given the initial conditions, the trained neural networks are used as a neural ODE operator expressed in the discovered state space,

$$\begin{pmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{pmatrix} = \mathcal{N}(\mathbf{q}, \dot{\mathbf{q}}),\tag{4.23}$$

where the generalized acceleration is computed using Eq. 4.2. After training, the system is solved using the RK45 method [61] implemented in the Scipy package [257] in Python but any solver can be used. After time integration, the accelerations in physical coordinates can be obtained by passing the generalized coordinate (i.e., in latent space) solution to the decoder, $f_v(\mathbf{q})$.

One practical consideration that needs to be addressed when dealing with response data from real structures is the discrepancy in the order of magnitude between displacements, velocities, and accelerations. These differences arise from the difference in orders of magnitudes associated with the mass and stiffness properties of the structure. This feature may lead to poor training performance since the neural network weights may have very disparate values leading to convergence issues due to the uniform gradient updates taken. To address this, when these discrepancies are observed, uniform time warping is applied to scale all the response quantities

Algorithm 2. INN with neural differential operator

Input: K snapshots of partial measurements $\mathbf{x}_m \in \mathbb{R}^{m \times K}$. Additional hyperparameters: size of training set T_{train} and validation set T_{val} , starting learning rate η , batch size N_b , maximum number of epochs N_{epochs} , and number of minibatches $N_{mb} = (T_{\text{train}} - T_{\text{val}})/N_b$.

Output: Latent space representation in generalized coordinates $\mathbf{q}(t)$ and learned neural differential operator $\mathcal{N}(\mathbf{q}, \dot{\mathbf{q}})$

- 1: Run PECUZAL algorithm to obtain the matrix of time delay embeddings $\mathbf{x} \in \mathbb{R}^{D \times K}$ and use the same time delays for $\dot{\mathbf{x}}, \ddot{\mathbf{x}}$.
 - 2: Build basis matrix $\mathbf{V} \in \mathbb{R}^{D \times d}$ from SVD of $\mathbf{x} \in \mathbb{R}^{D \times K}$.
 - 3: Get projection $\mathbf{z} \in \mathbb{R}^{d \times K}$ following Eq. 4.4. Similarly for $\dot{\mathbf{z}}, \ddot{\mathbf{z}}$.
 - 4: Split training data into $\mathbf{z}_{\text{train}} = [\mathbf{z}_{\text{subtrain}}, \mathbf{z}_{\text{val}}]$.
 - 5: Randomly initialize weights $\mathbf{\Theta}^0$.
 - 6: Initialize epoch counter $n_e = 0$.
 - 7: **while** $n_e < N_{\text{epochs}}$ **do**
 - 8: **for** $i = 1 : N_{mb}$ **do**
 - 9: Get a minibatch from the sub-training set, $\mathbf{z}_{\text{batch}} \subseteq \mathbf{z}_{\text{subtrain}}$.
 - 10: Pass time delay embeddings through INN encoder to obtain $\mathbf{q}_{\text{batch}}, \dot{\mathbf{q}}_{\text{batch}}, \ddot{\mathbf{q}}_{\text{batch}}$
 - 11: Compute $T_{nn}(\dot{\mathbf{q}}_{\text{batch}})$, $U_{nn}(\mathbf{q}_{\text{batch}})$, and $I_{nn}(\dot{\mathbf{q}}_{\text{batch}})$
 - 12: Compute all of the derived quantities from energy and the generalized accelerations $\hat{\mathbf{q}}_{\text{batch}}$ (Eq. 4.2)
 - 13: Pass $\hat{\mathbf{q}}_{\text{batch}}$ through the INN decoder to obtain $\hat{\mathbf{x}}_{\text{batch}}$
 - 14: Evaluate loss function terms to form total loss (4.22)
 - 15: Accumulate loss and compute gradient of loss function $\nabla_{\mathbf{\Theta}} \mathcal{L}$.
 - 16: Update weights $\mathbf{\Theta}^{N_{mb}n_e+i+1} = \text{ADAM}(\eta, \nabla_{\mathbf{\Theta}} \mathcal{L}, \mathbf{\Theta}^{N_{mb}n_e+i})$.
 - 17: **end for**
 - 18: Repeat **for loop** on $\mathbf{z}_{\text{val}}$ with updated weights.
 - 19: Accumulate loss on $\mathbf{z}_{\text{val}}$ to evaluate network convergence.
 - 20: $n_e = n_e + 1$
 - 21: **end while**
-

to the same order of magnitude. Uniform time warping is achieved by choosing a new time step, $d\tau$, such that the time derivatives of the displacement are the same order of magnitude as the displacement itself. The displacements are unchanged after this warping but the velocities are now scaled by Γ and the accelerations by Γ^2 where $\Gamma = dt/d\tau$. In the following examples, the data were time warped for better performance but the results are presented in the original unscaled forms.

4.5 Numerical examples

The following subsections demonstrate the proposed approach on two computational examples and two experimental datasets. All the trajectories considered were generated from free vibration response. The hyperparameters for the networks used for each example are provided in Appendix B.1.

4.5.1 Oscillator with isolated cubic stiffness nonlinearity

The first example used to demonstrate the proposed approach is a chain of 64 masses ($m = 1.0$) connected with linear springs ($k = 1.0$) with nonlinearities over the interval $s \in (s_1, s_2)$, representing a one-dimensional rod with a nonlinear region, as shown in Fig. 4.2. The nonlinearity is present in all the elements within the interval s and is represented as $f_{nl,i} = \kappa_i(x_{i+1} - x_i)^3$, where κ_i represents the cubic stiffness of the i -th element and x_i represents the corresponding displacement. In this work, the following parameters were used: $s_1 = 0.25$, $s_2 = 0.35$, and $\kappa_i = 2.6214 \times 10^5$. Proportional Rayleigh damping was applied corresponding to $\mathbf{C} = 1 \times 10^{-2} \mathbf{K}$.

To study this structure, the model was excited by applying initial conditions meant to excite either only the first linear mode, the first two linear modes, or the first three linear modes as initially described in [180]. These cases are denoted as A, B, and C, respectively. Because the structure is nonlinear, however, modal coupling leads to multiple modes always being excited, as will be illustrated in the results presented later. Fig. 4.3 thru 4.5 show how the rod deforms

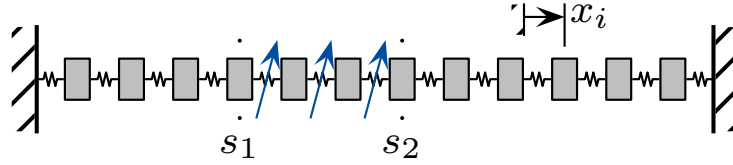


Figure 4.2. Example system with isolated nonlinearities. The nonlinearities are localized within the region $s \in (s_1, s_2)$, while the remainder of the rod is linear.

shortly after the initial conditions are applied. The six green nodes indicate the measured DOFs.

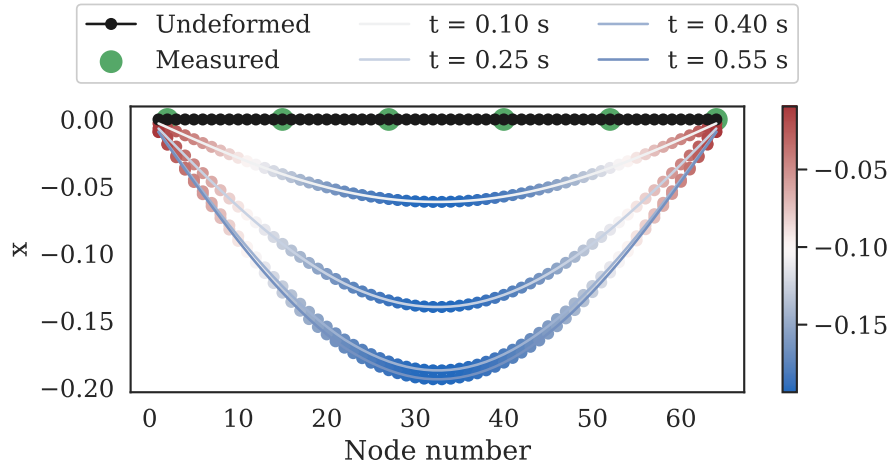


Figure 4.3. Snapshots in time of rod deformation colored by displacement for the first mode excitation case (A). Measured locations shown in green.

Figures 4.6 thru 4.8 illustrate a short window of the original model response at the measurement points as well as the time-delayed copies found by the PECUZAL algorithm. In all three cases, the PECUZAL algorithm added only one time-delayed copy in addition to the original signals. The time-delayed embeddings were then passed to the encoder described in section 4.3.1. The singular values were used to determine the number of dimensions kept. A threshold of 1×10^{-2} was used to determine the number of dimensions. This threshold value was chosen through an initial iterative study to determine acceptable reconstruction accuracy. The number of dimensions kept for each case are summarized in Table 4.1.

The structural response is obtained by following the procedure outlined in Fig. 4.1 and

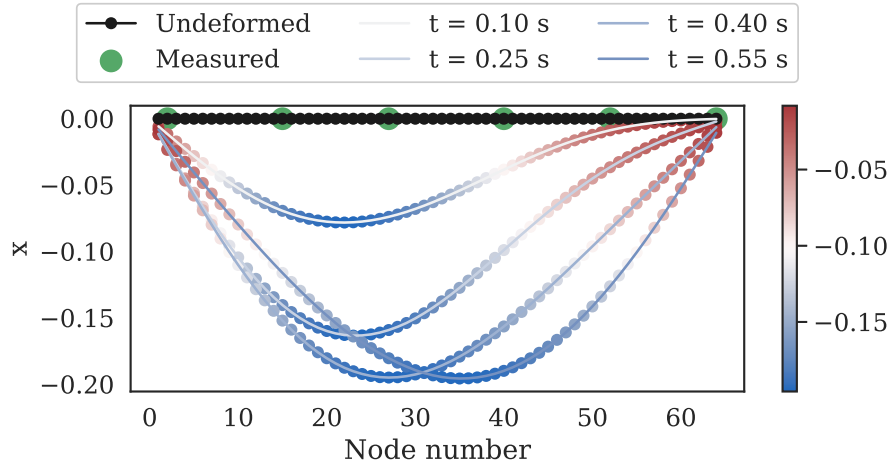


Figure 4.4. Snapshots in time of rod deformation colored by displacement for the first two modes excitation case (B). Measured locations shown in green.

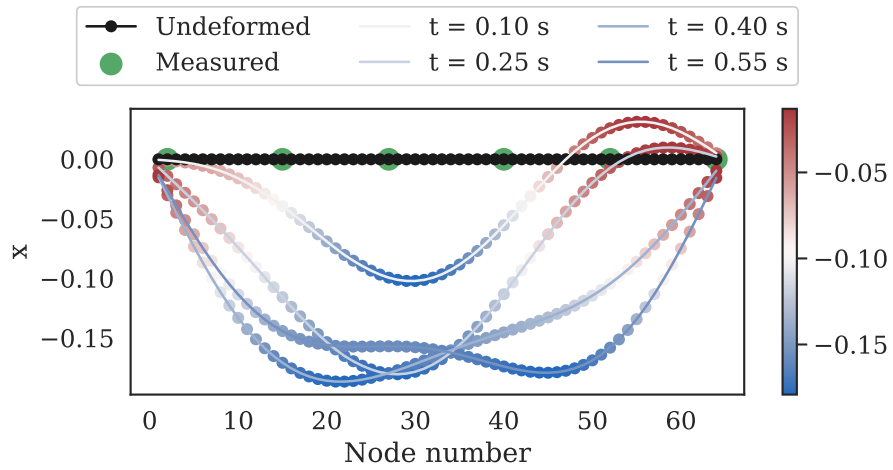


Figure 4.5. Snapshots in time of rod deformation colored by displacement for the first three modes excitation case (C). Measured locations shown in green.

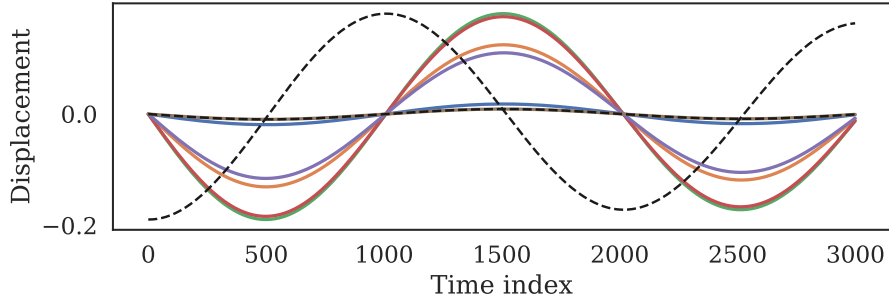


Figure 4.6. Original signals (color) and time-delayed copies (dashed black) for the first mode excitation case (A).

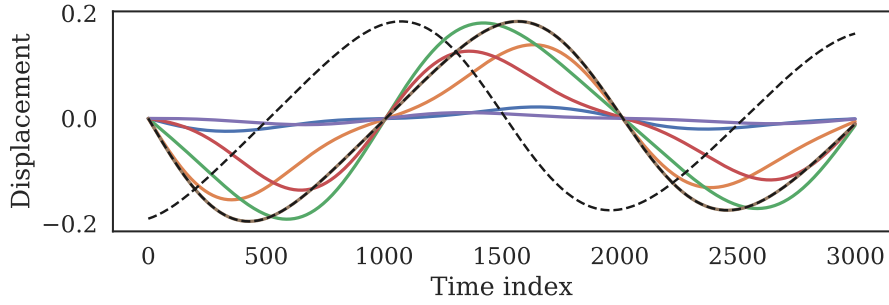


Figure 4.7. Original signals (color) and time-delayed copies (dashed black) for the first two modes excitation case (B).

Algorithm 2. The time-delay embeddings are obtained by running the PECUZAL algorithm. For this problem, the PECUZAL algorithm was run in Matlab on a single processor and it took 30 minutes to complete. The parameters of the PECUZAL algorithm are determined by first computing the average mutual information of the signals to get an idea of the magnitude of the delay required. The time-delayed copies are then appended to the original signals to obtain the complete set of time-delay embeddings. As outlined in Eq. 4.4, SVD is used to project the embeddings to a lower dimensional space. The reduced embeddings, $\tilde{\mathbf{z}}$, are linearly mixed as shown in Eq. 4.5. These linearly mixed, reduced order embeddings are then passed to the normalizing flow encoder which learns the mapping between these coordinates and the generalized coordinates ($\mathbf{q}$) that will satisfy the constraints described in section 4.2.1. The learned $\mathbf{q}$ vector is transformed back into the physical measured coordinate space by inverting the forward transformation, as described in section 4.3.1. The learned system is time-integrated

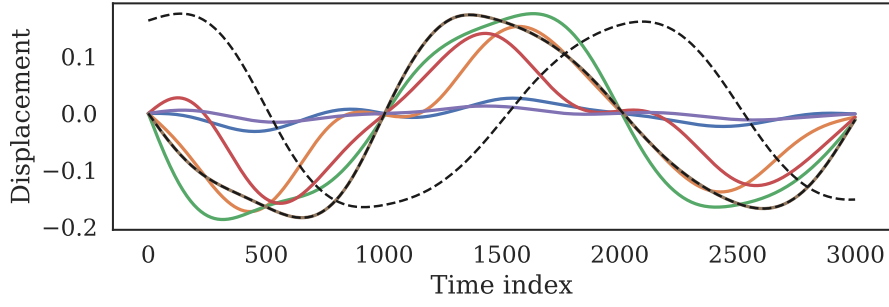


Figure 4.8. Original signals (color) and time-delayed copies (dashed black) for the first three modes excitation case (C).

in the latent space since these are the coordinates that satisfy the Hamiltonian structure. The error is ultimately evaluated by comparing the inverse transformation back to the physical domain with the original measurements. The reconstructed responses for this problem are shown in Fig. 4.9 thru 4.11 and the mean squared errors (MSE) are reported in Table 4.1. In addition to demonstrating temporal extrapolation, 10 different random initial conditions were prescribed for case C. The average mean squared error for these alternative initial conditions was 6.59×10^{-5} . One example of the reconstructed response is shown in Fig. 4.12. As illustrated, the reconstruction for these alternative initial conditions is not as accurate as it is for the original set but the general trajectories are still captured with acceptable accuracy. As mentioned in section 4.3, the INN encoder may be overly constraining. For this reason, the autoencoder described in 4.3.2 was also explored for case C. The architecture of the encoder and decoder network was the same as the MLP architectures used for the INN and was trained for the same number of epochs (150,000) to be able to make a direct comparison. The test MSE is also reported in Table 4.1 which was a bit higher for the test set with the original initial conditions. However, the MSE computed on the alternative initial conditions was 2.02×10^{-6} which is considerably lower than the MSE obtained with the INN encoder. However, training the MLP autoencoder is around 30% more time consuming and results in twice as many parameters. If a parsimonious and truly bijective mapping is desired, the INN is the best choice. For this reason, the remainder of the chapter (except for the last example) uses the INN encoder but a brief exploration of the

sensitivity of the inverse transformation to the latent space is provided in Appendix B.

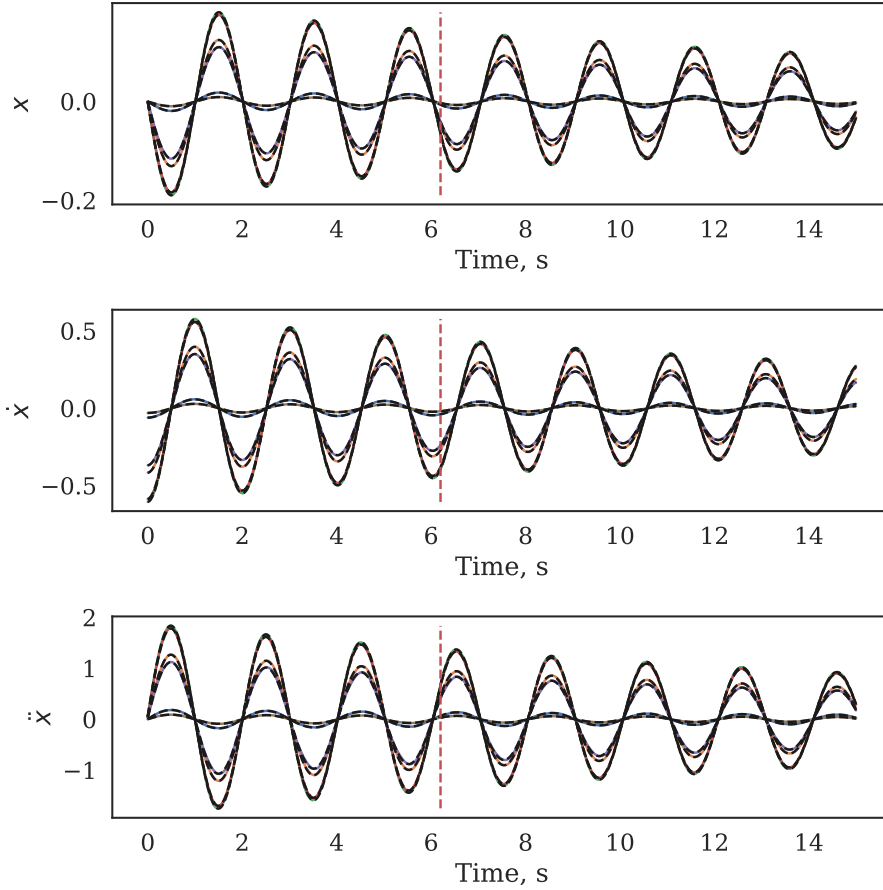


Figure 4.9. Reconstructed response from neural network prediction for the first mode excitation case (A). Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.

To understand the low-dimensional manifold that the neural network is learning, the responses from the full dimensional model were embedded into a low-dimensional space and compared to the embeddings from the latent space. The t-distributed stochastic neighbor embedding (t-SNE) [254] method was used to find the low-dimensional embedding of the data. To this end, the scikit-learn [193] implementation of t-SNE was used. The results are shown in Fig. 4.13. As illustrated, both the 64-dimensional space from the original system and the low-dimensional latent space are embedded in manifolds that are topologically similar. This result indicates that the learned two-dimensional manifolds are consistent between the original full-order model and the learned latent space. While this result does not guarantee that the

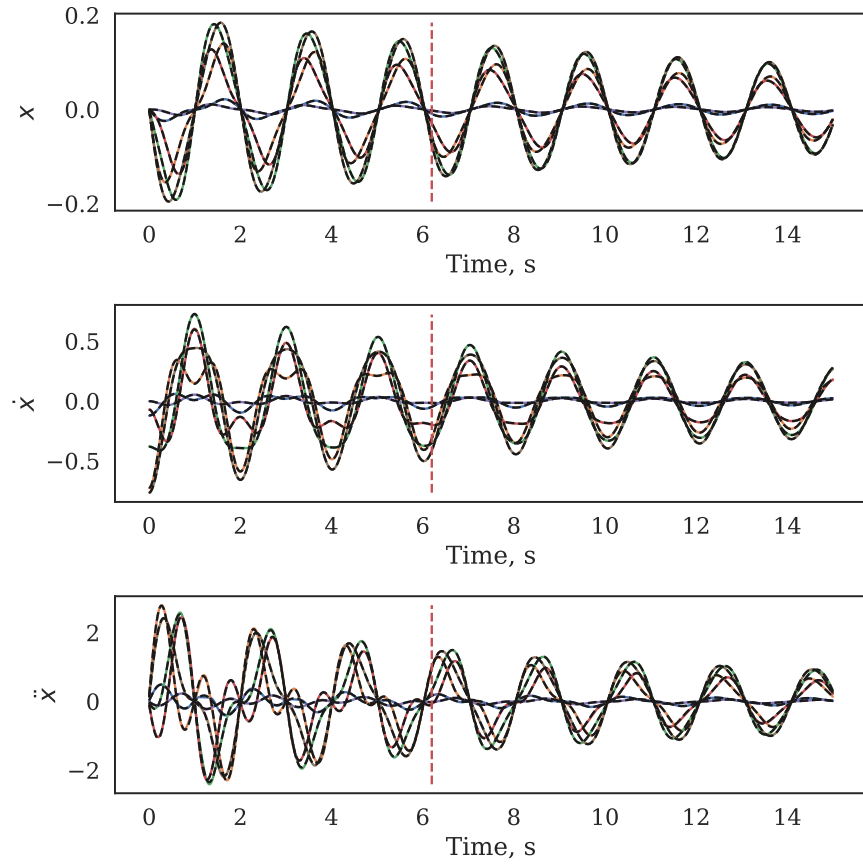


Figure 4.10. Reconstructed response from neural network prediction for the first two modes excitation case (B). Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.

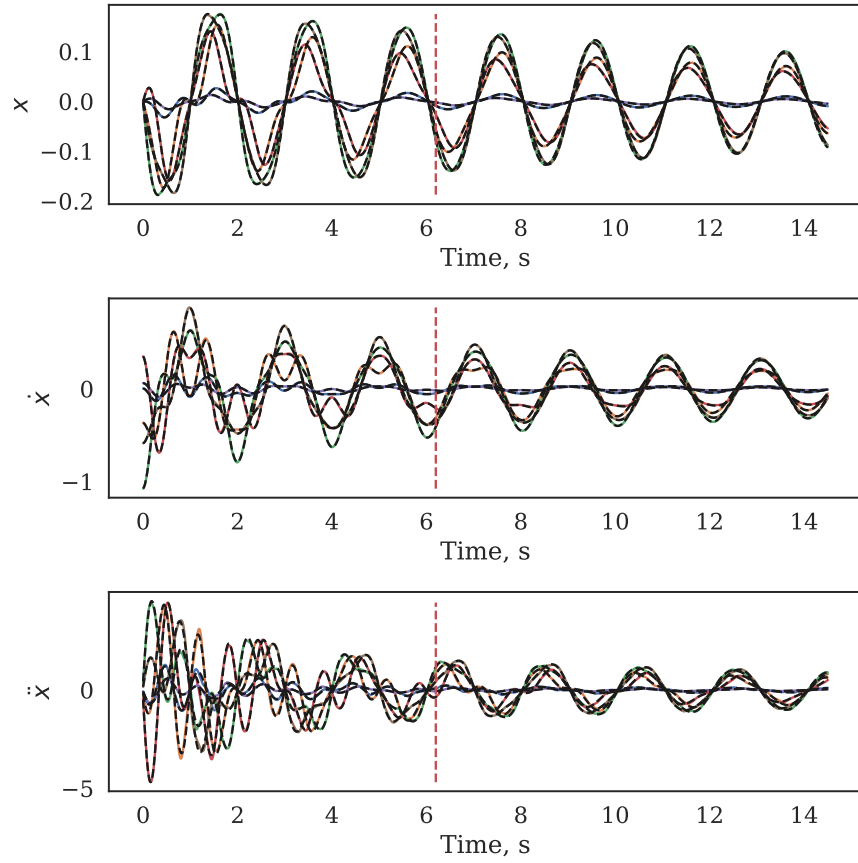


Figure 4.11. Reconstructed response from neural network prediction for the first three modes excitation case (C). Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.

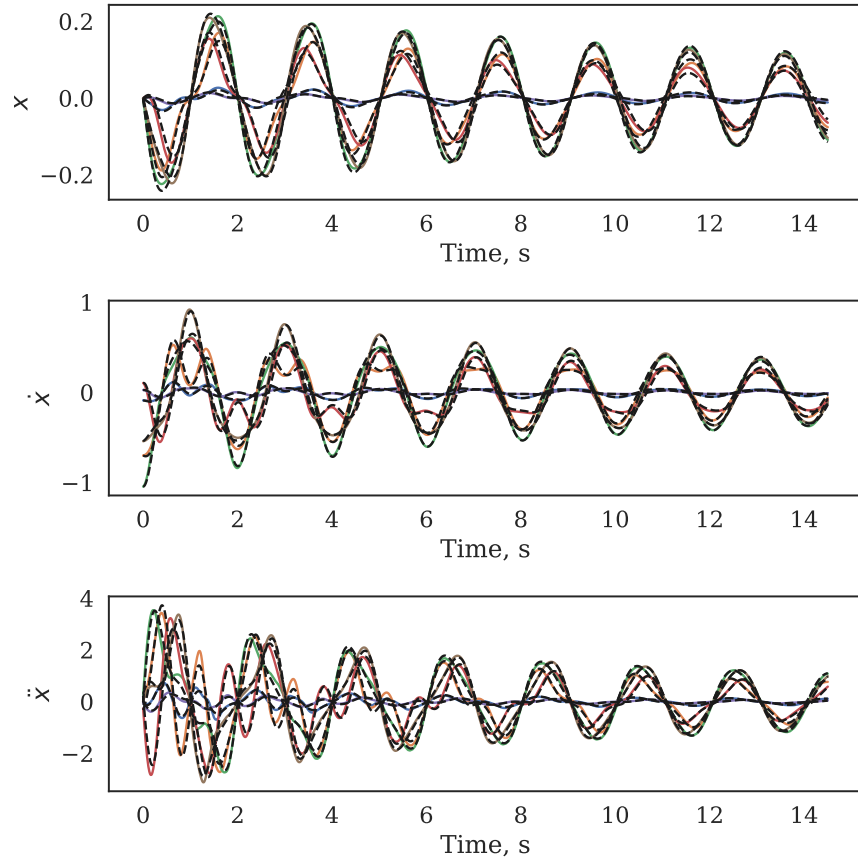


Figure 4.12. Reconstructed response from neural network prediction for the first three modes excitation case (C) with alternative initial conditions. Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.

Table 4.1. Summary of results for rod model

Case	Latent Dimension	Displacements MSE
A	4	2.64×10^{-8}
B	6	1.41×10^{-7}
C (INN)	6	8.53×10^{-7}
C (MLP)	6	1.21×10^{-6}

learned latent space is the same as the original underlying latent space, it suggests an equivalency between them.

Because the learned network is structure-preserving, we can compute the eigenvectors and eigenvalues of the learned system and compare them to the physical system. For conciseness we focused the discussion on case C. Fig. 4.14 illustrates the Fast Fourier Transform (FFT) computed for all 64 DOFs of the full-order system alongside the natural frequencies computed from the linearized learned latent space. As shown, the three main modes are captured well. However, higher-frequency modes are not exactly captured but the network is presumably reconstructing these frequencies from a combination of the lower-frequency modes that it is learning. To illustrate this result, Fig. 4.15 shows the modal displacements from the physical system (color) and the learned system (dashed). As shown, the learned model captures the modal response very accurately. This plot also illustrates how mode 4 is significantly excited even though this mode was not directly excited through initial conditions.

4.5.2 Two-dimensional membrane with isolated nonlinearities

The second example consists of a two-dimensional membrane with an isolated nonlinearity that was originally introduced in [200]. The membrane model is non-dimensional. Its dimensions are $(l_x, l_y) = (\sqrt{2}, 1/\sqrt{2})$. The membrane was discretized into a rectangular grid with dimensions $(N_x, N_y) = (21, 13)$. The ends were all clamped which results in a total of $n = 240$ DOFs. The total mass of the membrane is $m = 1$ and it was modeled with a lumped mass matrix where $M_{ii} = m/n$ and $M_{i \neq j} = 0$. The linear stiffness of each element was defined as

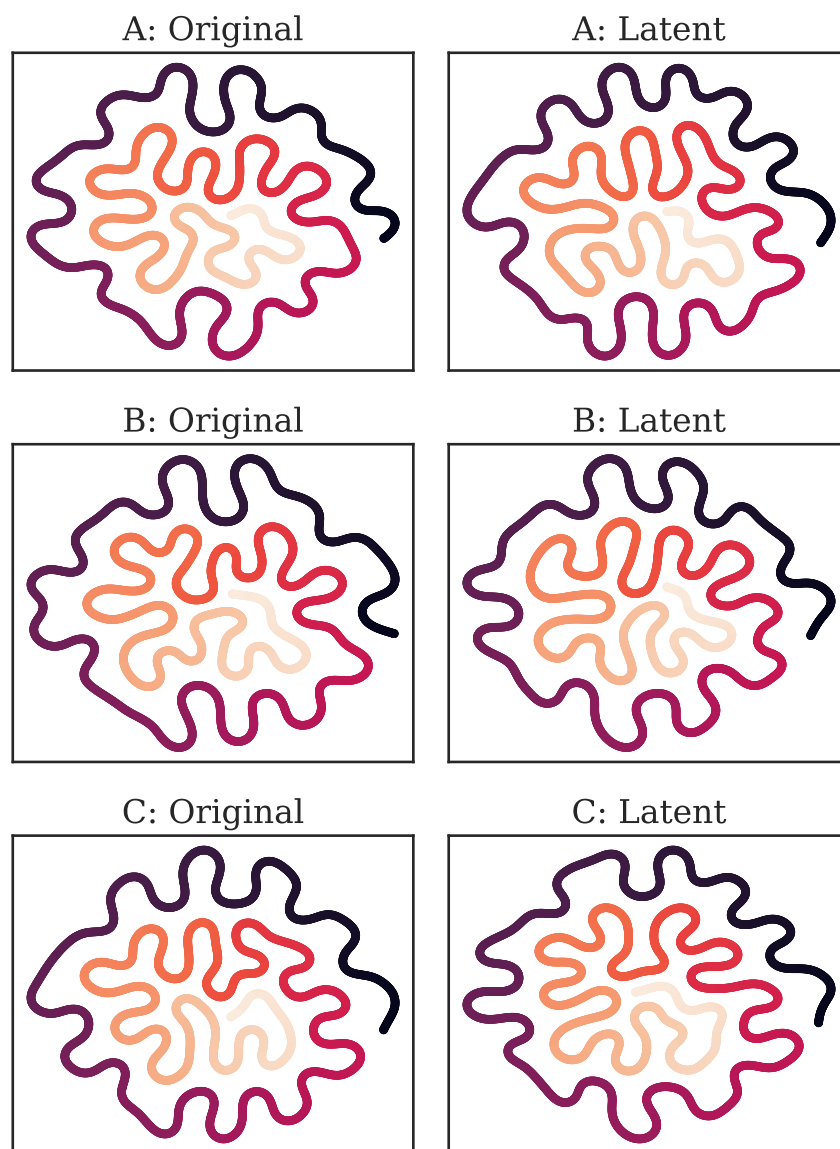


Figure 4.13. t-SNE embedding of full dimensional (left) and latent (right) space. The points are colored by their time index so that time evolution is indicated by going from bright to dark colors.

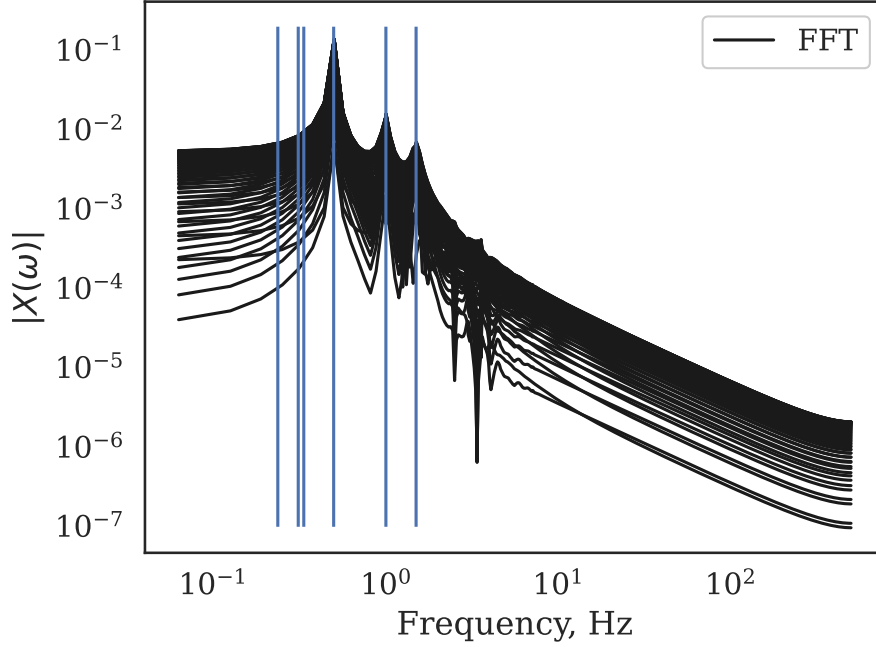


Figure 4.14. Fast Fourier Transform of physical displacements (blue) and frequencies computed from linearized learned latent space model (black).

$(k_x, k_y) = (n \cdot (l_x/N_x)^2, n \cdot (l_y/N_y)^2)$ and the stiffness matrix $\mathbf{K}$ is assembled from these individual elements. The linear damping matrix was assumed to be proportional to the linear stiffness matrix as $\mathbf{C} = 0.002\mathbf{K}$. A nonlinear region was modeled by adding cubic coupling terms to the stiffness and damping matrices. The cubic stiffness and damping coefficients were defined as $k_{nl} = c_{nl} = 0.02n^{3/2} = 74.361$. For this example, a total of 9 sensor locations were assumed as shown in Fig. 4.16. Similarly to the rod model in section 4.5.1, the initial conditions were defined as described in [180] by exciting the first two modes which excites a larger number of modes through modal coupling due to the nonlinearity.

Following the same workflow as in the previous example and described in Algorithm 2, the networks are trained with the membrane dataset to learn a model. The PECUZAL algorithm ran for 138 minutes on a desktop computer and determined that three additional time-delayed copies were required, as shown in Fig. 4.17 for a total of 12 dimensions in the embeddings matrix. After performing SVD, only six dimensions were retained based on the singular values of the training data. Following Algorithm 2, the neural differential operator was trained to learn

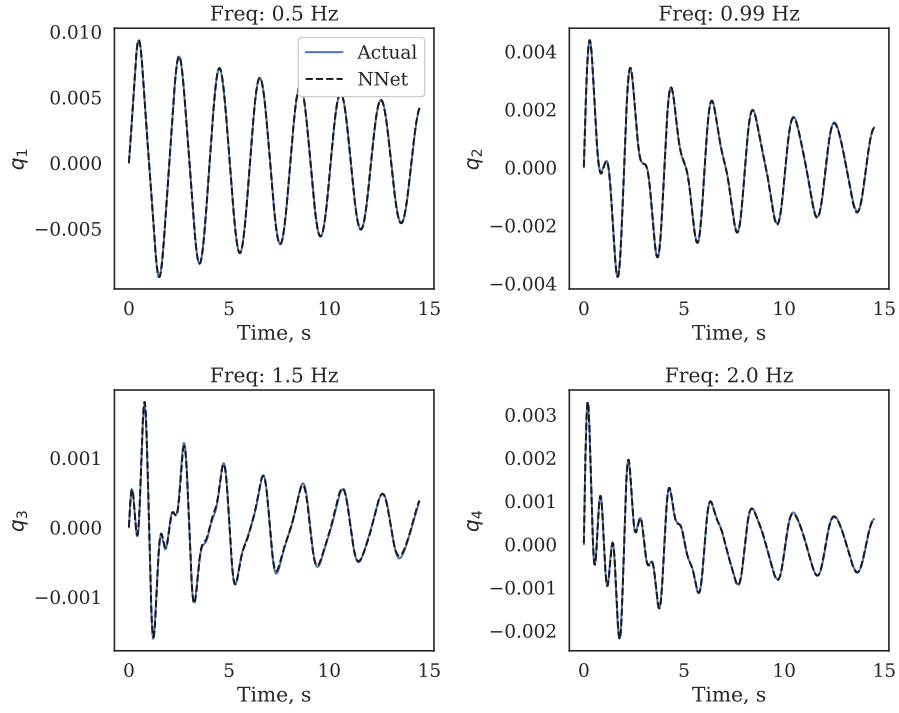


Figure 4.15. Modal displacements for original physical system (color) and learned latent space model (dashed).

the model in this six-dimensional latent space. Once trained, the model was then used to predict the response of the membrane under different initial conditions than the ones used for training. The energy is similarly injected into the first two modes but at 60% higher amplitude. For conciseness, only the results for the alternative initial conditions is presented. The time history responses for these test initial conditions for the 9 measured locations are shown in Fig. 4.18. The mean squared error for the displacements was 2.93×10^{-6} .

As with the previous example, manifold learning techniques were used to extract a lower dimensional representation of the trajectories to compare the manifold formed by the full order data and the learned latent space. To this end, a spectral embedding representation [14] of the data is shown in Fig. 4.19. Spectral embedding was chosen because it produced the most easily interpretable manifold and it was implemented using scikit-learn [193]. As shown, the spectral embedding representations between the full 240-dimensional space and the six dimensional latent space are topologically similar which implies that the underlying data structure is learned

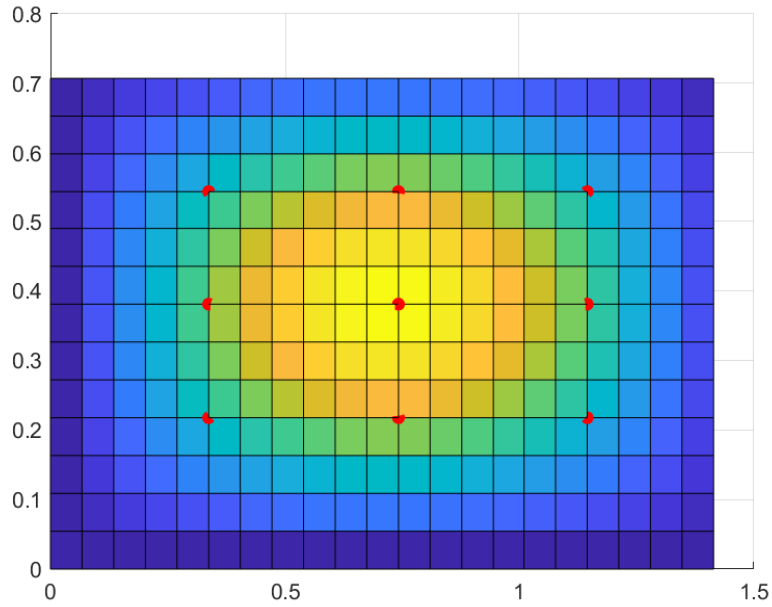


Figure 4.16. Membrane structure with visible grid and measurement points shown in red. Colored by deformation pattern of the first bending mode.

by the neural differential operator. Furthermore, the FFT of the response at the 9 measured locations was computed and compared to the experimental data as shown in Fig. 4.20. In addition to the FFT, the eigenvalues of the learned dynamic system were extracted by assembling the stiffness and mass matrices from the energy networks, and are shown as vertical lines in Fig. 4.20. As illustrated, the predicted FFT matches the first two modes really well but misses some of the higher frequency, lower amplitude responses but it captures some of the anti-resonant behavior present in the reference data. Similar to the rod model, the network tends to favor lower frequency modes and reconstructing the higher frequency response from the combination of these lower frequency components. Moreover, because the frequency content is changing as a function of time, the Short Time Fourier Transform (STFT) for the center of the membrane was also computed and is plotted in Fig. 4.21, which confirms some of the observations previously noted such the fact that the learned model matches the first two modes really well, but its accuracy decreases at higher frequencies where the response amplitude is smaller. The time histories

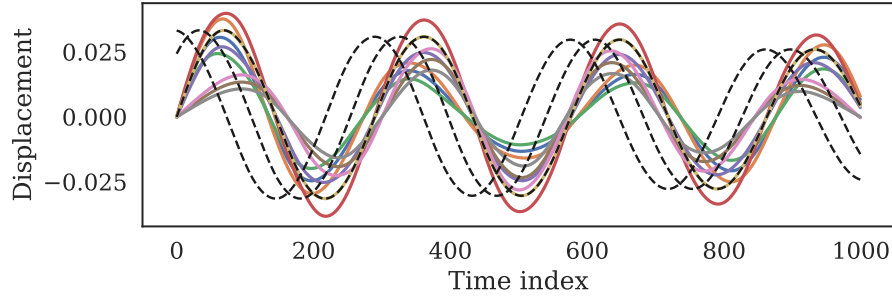


Figure 4.17. Original signals (color) and time-delayed copies (dashed black) for the membrane model.

were then projected onto modal space using the analytical mode shapes to inspect how well the predicted response matches the reference. This comparison is shown in Fig. 4.22. As illustrated, the model matches the modal responses really well in the modal coordinates shown despite the eigenvectors of the state space not corresponding exactly to the natural frequencies (except for the first two modes). This result indicates that the network is able to reconstruct the response with high accuracy by learning an equivalent representation of the model.

4.5.3 Brake-Reuß Beam

The next example corresponds to data obtained from a physical experiment presented in [45, 110] where the authors used high-speed cameras combined with Digital Image Correlation (DIC) to obtain full-field response of the structure known as the “Half Brake-Reuß beam” (HBRB). The HBRB consists of two beams that are joined through a three bolt lap joint as shown in the experimental setup in [45]. In the two papers cited, the authors used the data to perform nonlinear system identification. To this end, they performed experiments meant to excite the nonlinear behavior of the structure. In this work, we leverage the free response data that the authors of [45] produced. As described in their paper, the free response data was obtained by first exciting the beam at the frequency of interest with a fixed sinusoidal signal and then detaching the shaker once resonance is identified. This procedure is meant to isolate a mode of interest whereas in reality the response may include multiple modes responding due to modal coupling.

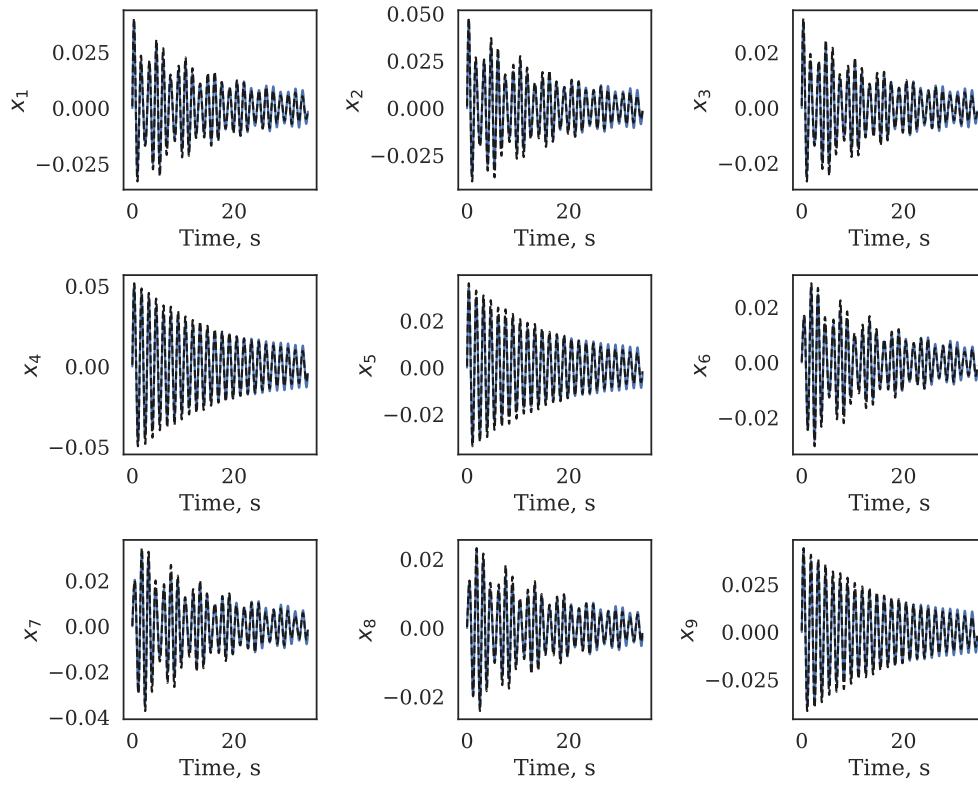


Figure 4.18. Reconstructed response from neural network prediction for the test set (alternative initial conditions). Color: ground truth. Dashed black: prediction. Dashed red: training/test boundary.

The nonlinearity in this structure arises from the bolted joint due to energy dissipation caused by slipping as well as possible intermittent contact in the joint. The authors of [110] found a slight change in frequency (less than 1.0 Hz) observed in the response but a significant increase in instantaneous damping ratio as a function of amplitude.

There were 206 DIC points that were provided by the authors. It was assumed that measurements were only available at three locations across the HBRB length (shown in green in Fig. 4.23). As in the previous examples, the embeddings were found by running the PECUZAL algorithm. For this example, it was found that one additional delayed copies were required. The embedding matrix consists of a four-dimensional vector (three original measurements plus one delayed copies). As described in section 4.3.1, SVD was performed on this time-delay embeddings matrix to obtain a low dimensional projection of the data. It was determined that two

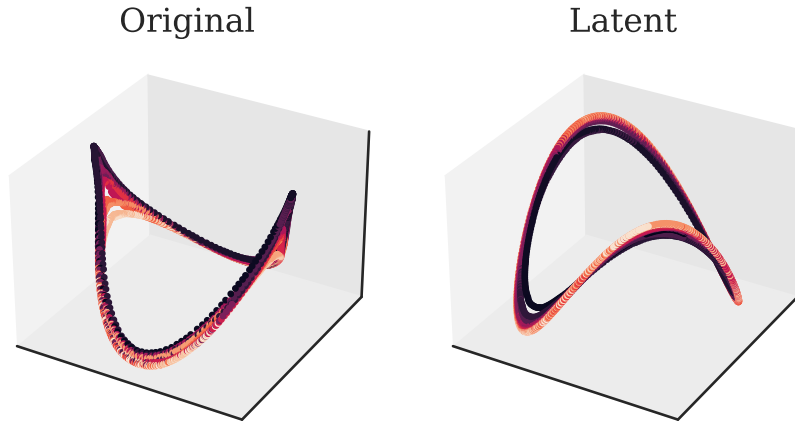


Figure 4.19. Spectral embedding of full dimensional (left) and latent (right) space in three dimensions. The points are colored by their time index so that time evolution is indicated by going from bright to dark colors.

dimensions would be retained based on the singular values obtained with a threshold of 1×10^{-2} . This lower dimensional representation of the embeddings is passed to the neural network as outlined in Eq. 4.5 thru 4.7 to obtain the generalized coordinates representation of the data. Only data up to 1s were included in the training set. Fig. 4.24 illustrates the predicted response (shown in blue) compared to the experimental response (shown in black). As the predicted response is really close to the actual, it is hard to distinguish between them. To this end, the phase portraits of the three measurement locations are also provided in Fig. 4.25. In addition, the Hilbert transform was applied to the data to extract an envelope of the time histories. The results are plotted in Fig. 4.26 alongside the error (computed as the difference between experimental and predicted, and shown in magenta) to illustrate that the error is orders of magnitude lower than the response, and that the error does not grow significantly with time. Another way of looking at the response is by extracting the amplitude-dependent frequency and damping from the free vibration time histories. The backbone curves were extracted using the method outlined in [131]. These nonlinear modal characteristics are shown in Fig. 4.27. As shown, the predicted response captures the softening

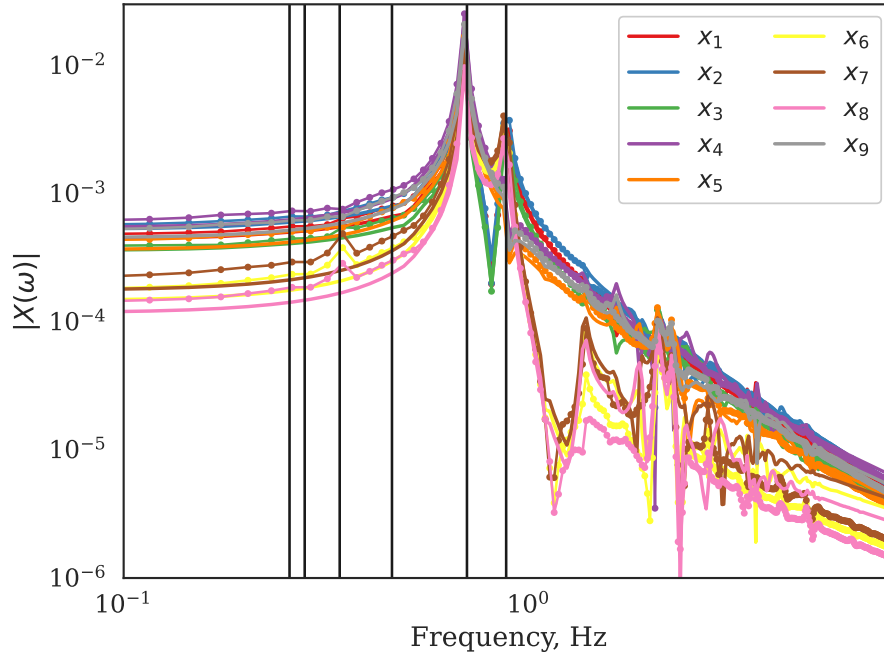


Figure 4.20. FFT of the actual physical displacements (solid line, no marker) at the measured locations and the predicted response (circle marker). The eigenvalues extracted from the model are shown with a vertical black line.

followed by a stiffening in the response, as well as the significant increase in damping as the amplitude increases. The authors of [110] described this change in frequency and damping as a result of energy dissipation occurring in the bolted lap joint which induces hysteretic behavior, a feature that is lacking from the learned model which may explain some of the discrepancies observed in the backbone curves.

4.5.4 S4 Beam

Another experimental example that was considered was a structure composed of two bolted steel beams known as the “S4 Beam” (or the “C-Beam”) that has been used in many experimental studies in the past [108, 265, 230] to study nonlinearities due to frictional dissipation in the bolted joints. The data used in this work were published in [264] where the authors conducted experiments by exciting the structure with a modal hammer at various amplitudes. The original paper contains photos of the experimental setup. In this work, the dataset corresponding

to a hammer impact in drive point one (DP1) at an amplitude of 280 N was used. The raw time history data was filtered with a band-pass filter between 200 and 400 Hz which corresponds to the range of the first two modes since these modes were shown to be coupled in [264]. According to the authors of [264], the impact at DP1 excited both modes and produced a noticeable shift in frequency and damping (for example, shown in Fig. 21 in [264]). The accelerometer data was integrated using the trapezoid rule twice to obtain velocities and displacements. The linear and quadratic trends were removed from the data using the “detrend” function in Matlab to prevent artificial drift.

This example proved to be more difficult to model with the INN (i.e., the learned model had stability issues). For this reason, and to demonstrate the performance of the alternative approach, it was decided that the MLP autoencoder would be used since it is more flexible (though it is not bijective). The same workflow was applied to the S4 Beam data which consisted of five accelerometer measurements. After applying the PECUZAL algorithm, it was determined that three additional time-delayed copies were required, making the time-delay embeddings an eight-dimensional vector. After performing SVD on the embeddings, it was determined that three latent dimensions would be used based on the magnitude of the singular values. Because the experimental data consisted only of accelerations, only these are compared to the simulated results obtained. Fig. 4.28 illustrates the time history comparison of the response at the three accelerometer locations with the largest response amplitude (the other two accelerometers’ response was two orders of magnitude smaller). Fig. 4.29 shows a zoomed-in version of the same plot illustrating the good agreement between predicted and measured responses outside of the training range. Similar to the previous example, we provide the Hilbert transform of the data in Fig. 4.30 to illustrate how the damping was captured accurately over the entire time period for the three locations considered. Furthermore, Fig. 4.31 illustrates the modal accelerations obtained by modally filtering the accelerations using the Moore-Penrose inverse of the mode shape matrix provided in the original dataset. As shown, the predicted time histories match the modally filtered measurements even later in time. Moreover, the learned model can be used to

compute the nonlinear normal modes (NNMs) of the structure. To this end, a multi-harmonic balance code previously presented in [252] was used to compute the NNMs of the learned system in the latent space. These NNMs are shown in Fig. 4.32 where it is observed that mode 1 exhibits a stiffening behavior while mode 2 exhibits a softening behavior. This behavior is consistent with the observations made by the authors in [264]. The physical interpretation of this result is not fully explored in the original paper but it is observed that the behavior of mode 2 is primarily driven by microslip in the joint while mode 1 causes the joint to open and close which is why it does not follow typical microslip behavior.

4.6 Practical Challenges

There are many practical considerations that need to be taken into account when dealing with data-driven approaches beyond standard pre-processing practices (e.g., standardizing or normalizing data). Some of these challenges listed here are based on the experience of going through the examples in section 4.6. These challenges present opportunities for future improvements but also provide caveats that need to be considered in this line of research.

- The data used in this work typically resides in relatively low-dimensional manifolds which enables the discovery of a low-dimensional model from partial observations. When the data are highly multi-modal, the number of observables also needs to increase, as postulated by Takens theorem.
- Related to the previous item, the dimensionality of the latent space is not obvious from the data *a priori*. Instead, an iterative approach had to be taken in order to determine the latent dimension based on neural network convergence.
- The proposed approach requires measurements of displacement, velocity, and acceleration which may not always be available. When displacement data are available (and provided the data has been appropriately filtered and cleaned), velocity and acceleration

can be approximated fairly robustly. However, obtaining displacements from acceleration measurements is more challenging because of the error introduced in the approximations obtained from numerical integration of the data. Accelerometer data requires additional detrending and filtering for use in this application.

- For some cases, the DOFs in the latent space have very different magnitudes which leads to placing more emphasis on the larger magnitudes. The reconstruction errors of these DOFs can be scaled by dividing over the maximum absolute value of their respective signals but this strategy leads to placing too much emphasis on signals that are too small. In this work, in certain cases, it was helpful to scale them by the softplus function instead of the maximum, because it softens the effect of dividing by small numbers.
- In this work, only homogeneous systems (i.e., no external forcing) were considered. Practical structural dynamics applications typically involve external loading conditions that are hard to characterize because they're arising from operational (often unmeasurable) conditions, unless the data are from a well controlled experiment. Future work will consider systems with external loads.
- This work did not consider multiple solutions arising from bifurcations. In such cases, this approach may fail if the multiple solutions are not observable in the training data. Future work will focus on studying these cases.

4.7 Conclusion

This research presented a general framework for learning structure-preserving machine learning models from sparse measurements by employing Taken's theorem to recover the underlying state space from partial measurements. The presented method combines the PECUZAL algorithm for finding time-delayed embeddings of partial measurements with an INN autoencoder that preserves the diffeomorphic mapping assumed in Taken's theorem. The Lagrangian

structure of the system is preserved in the learned latent space by learning the energy of the system as well as the dissipation, and relating these to the measured accelerations through the corresponding Hamiltonian. The proposed approach was demonstrated on two computational examples with isolated nonlinearities and two experimental examples with nonlinearities due to frictional dissipation in bolted joints. In all of the examples, it was shown that the learned network was capable of reproducing the response accurately outside of its training regime by either varying initial conditions or evaluating its performance past the training regime in the time domain when only one dataset was available (i.e., for the experimental cases). In addition to time domain comparisons, various other metrics were used to compare the response qualitatively, including learned manifolds, frequency domain metrics, and backbone curves. This work is a step forward to the goal of having a physically-interpretable digital twin for use in structural health monitoring applications where the learned model will be used alongside real-time monitoring to detect structural changes and perform prognostic tasks given the current health state of the structure. Future work will consider system with external loads, other types of nonlinearities, and include an explicit measure of uncertainty quantification.

4.8 Preview of Chapter 5

Chapter 4 introduced a strategy for dealing with more realistic cases where an as-built, as-deployed structure is only instrumented sparsely and a predictive data-driven, structure-preserving digital twin is required. Another situation where the digital twin may be required to be deployed consists of the case where a model is actually available but it is no longer representative of the as-built, as-deployed state because deviations have occurred. In such case, the digital twin can still leverage the knowledge about the existing structure but may be able to assimilate the current state from measured data. Chapter 5 addresses this situation by incorporating knowledge from an idealized representation of the structure and combining it with data from the deployed structure to learn the deviatoric behavior.

4.9 Remarks

Portions of this dissertation have been published or are being considered for publication. Chapter 4 is composed of the following first authored publications:

David Najera-Flores and Michael Todd. State-space reconstruction from partial observables using an invertible neural network with structure-preserving properties for nonlinear structural dynamics. Manuscript accepted for publication in Nonlinear Dynamics, 2024

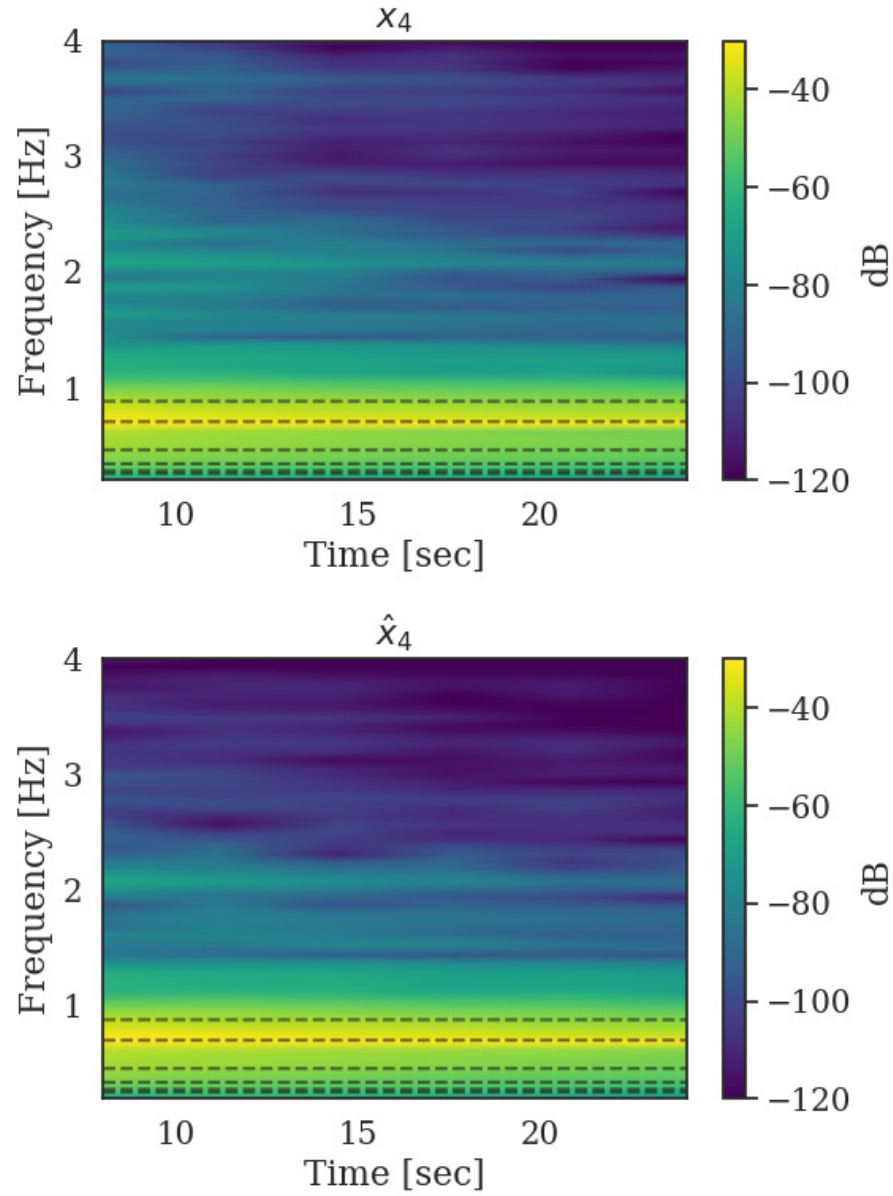


Figure 4.21. STFT of the actual physical displacements (top) at the measured locations and the predicted response (bottom). The eigenvalues extracted from the model are shown with a dashed line.

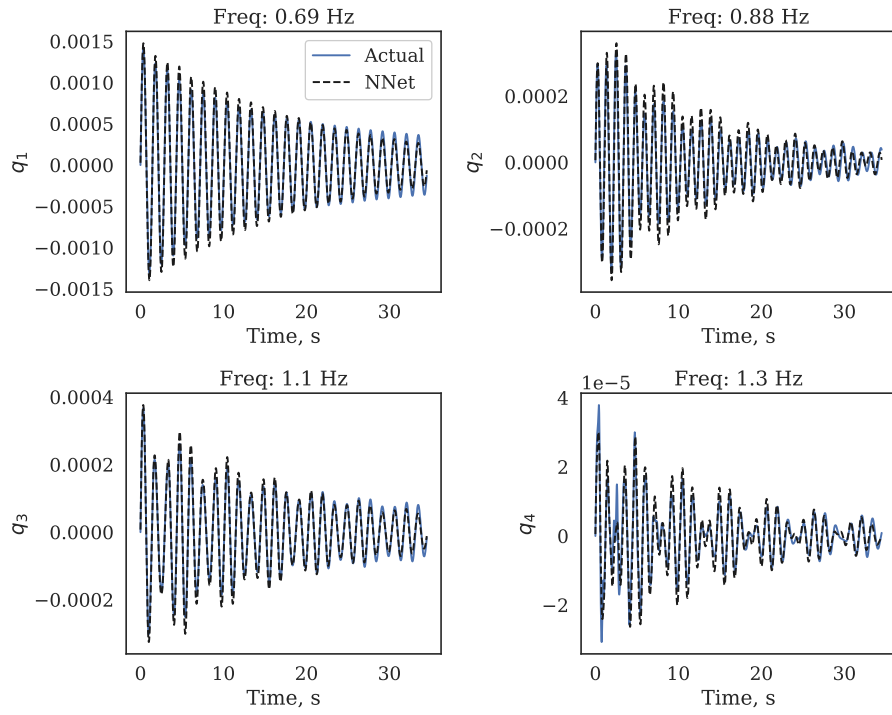


Figure 4.22. Modal displacements for original physical membrane system (color) and learned latent space (dashed).

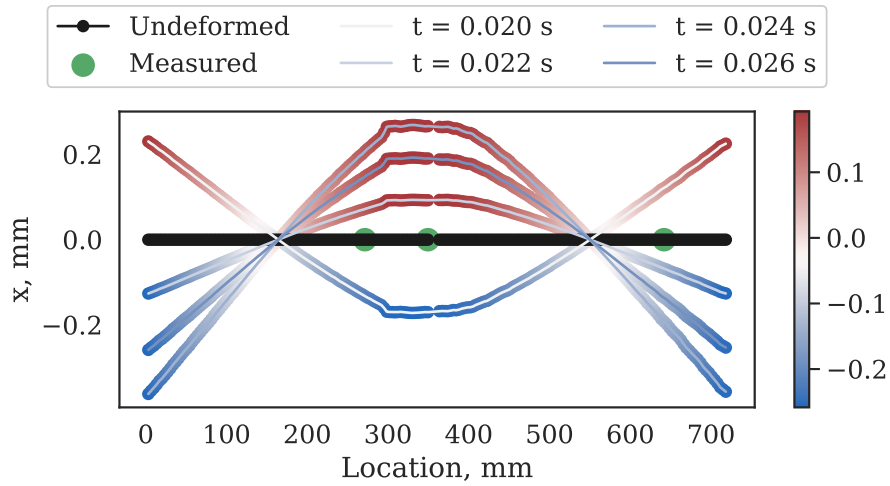


Figure 4.23. Snapshots in time of HBRB deformation colored by axial displacement. Measured locations shown in green.

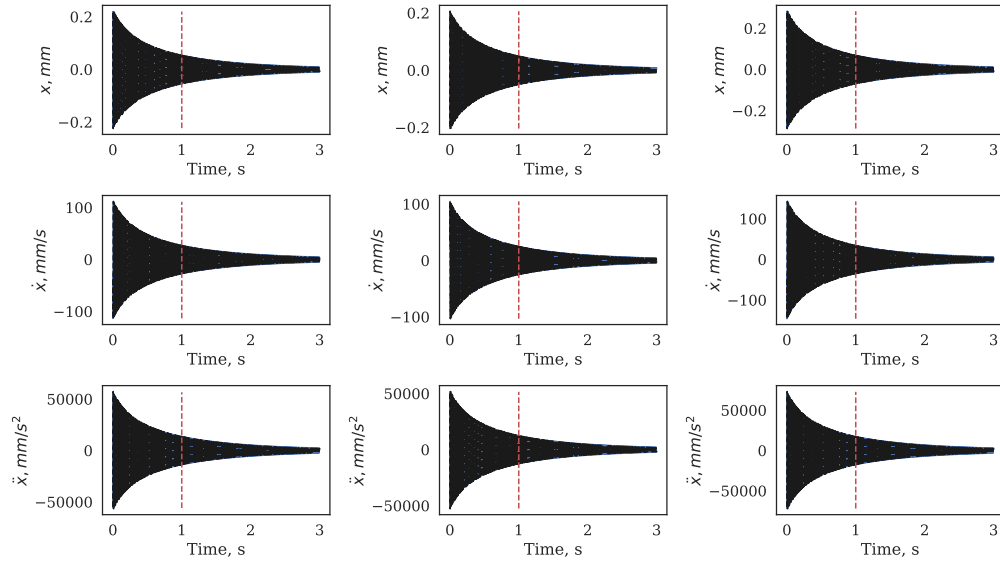


Figure 4.24. Time histories of HBRB experimental (blue) and predicted (black) displacement, velocity, and acceleration for the three measured locations. Vertical dashed line indicates training/test data boundary.

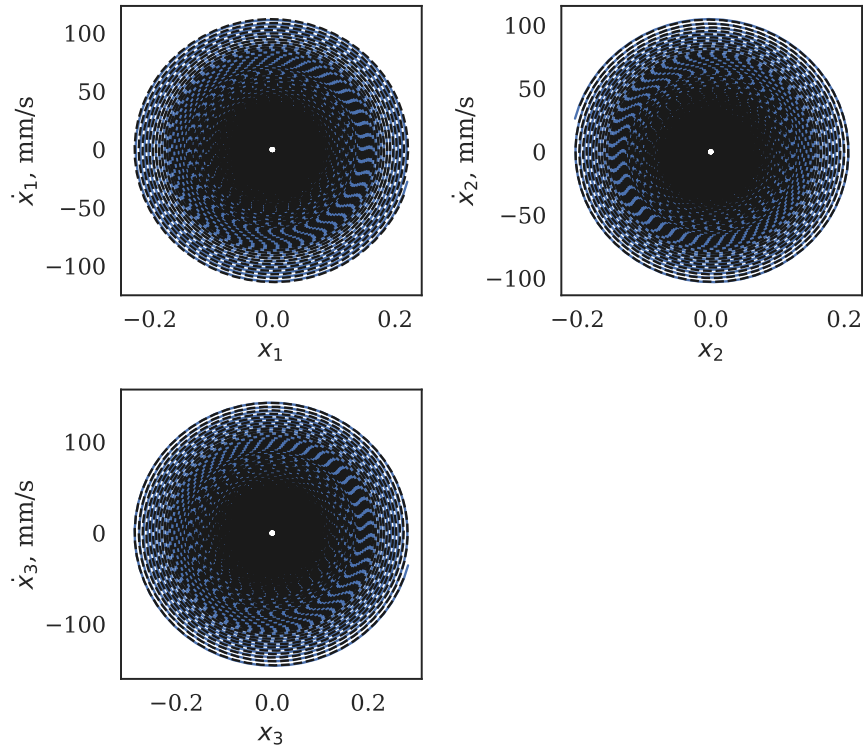


Figure 4.25. Phase portraits of the HBRB experimental data (blue) and predicted (dashed black) for the three measurement locations.

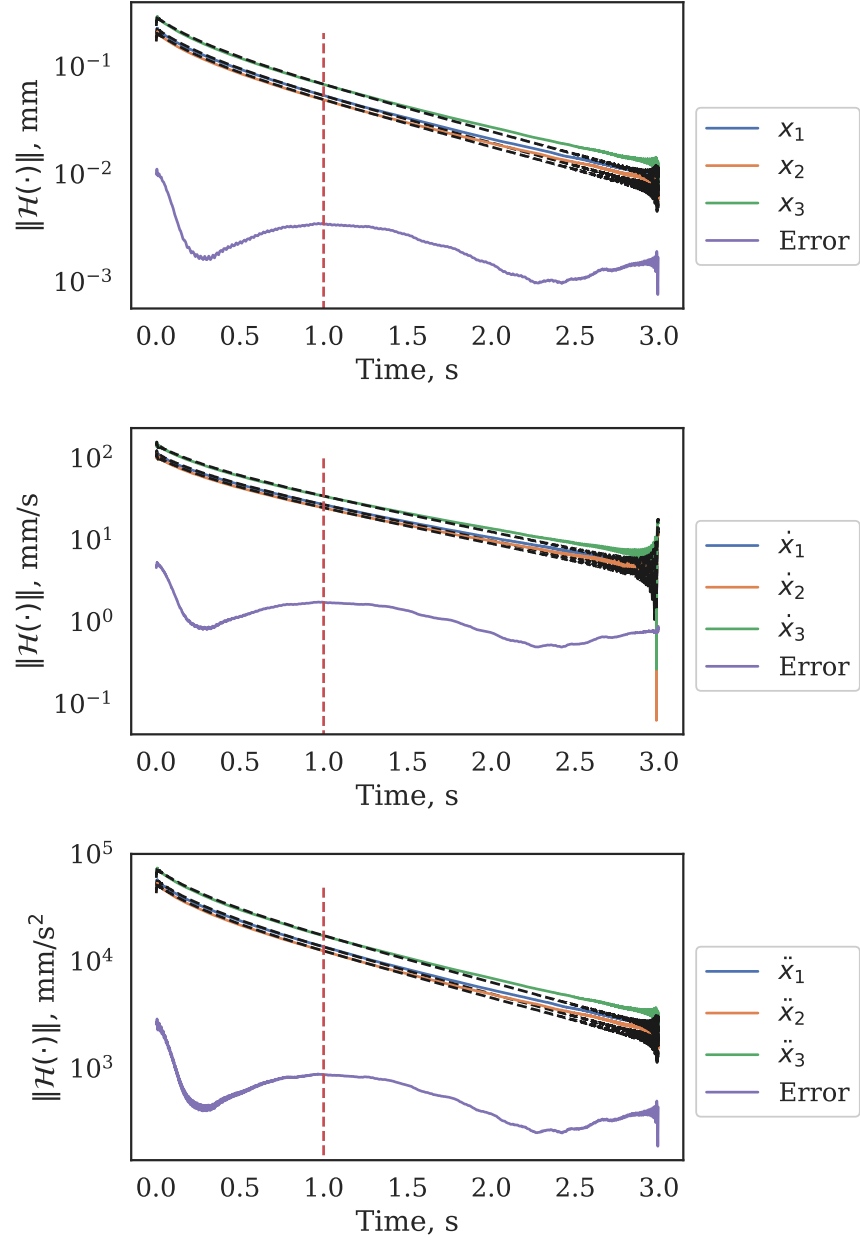


Figure 4.26. Hilbert transform of the HBRB response at the three measurement locations (blue, orange, and green) and the predicted response (dashed black), as well as the error (magenta). Vertical dashed line indicates training/test data boundary.

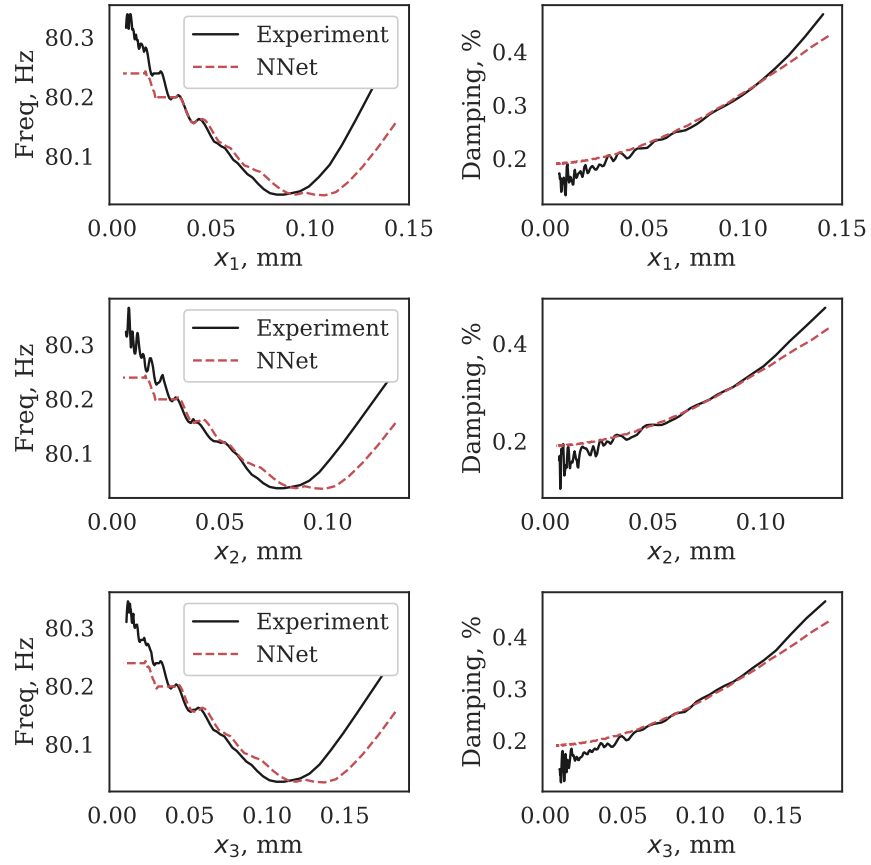


Figure 4.27. Frequency (left) and damping (right) backbone curves extracted from time histories of HBRB response.

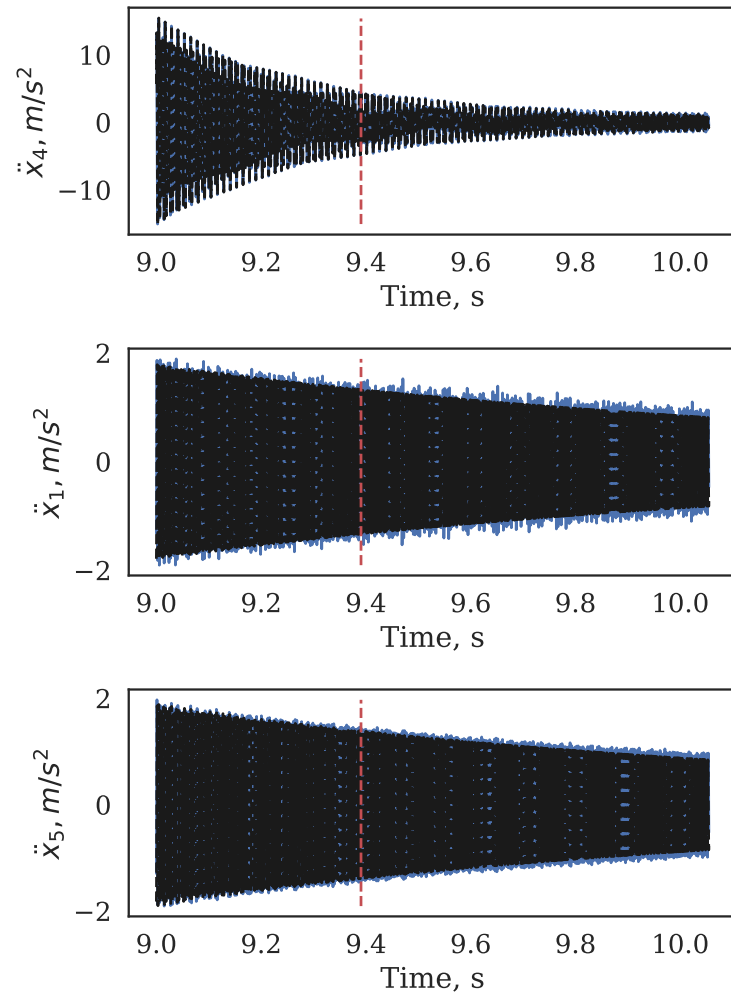


Figure 4.28. Time histories of S4 beam experimental (blue) and predicted (black) acceleration for three accelerometer locations. Vertical dashed line indicates training/test data boundary.

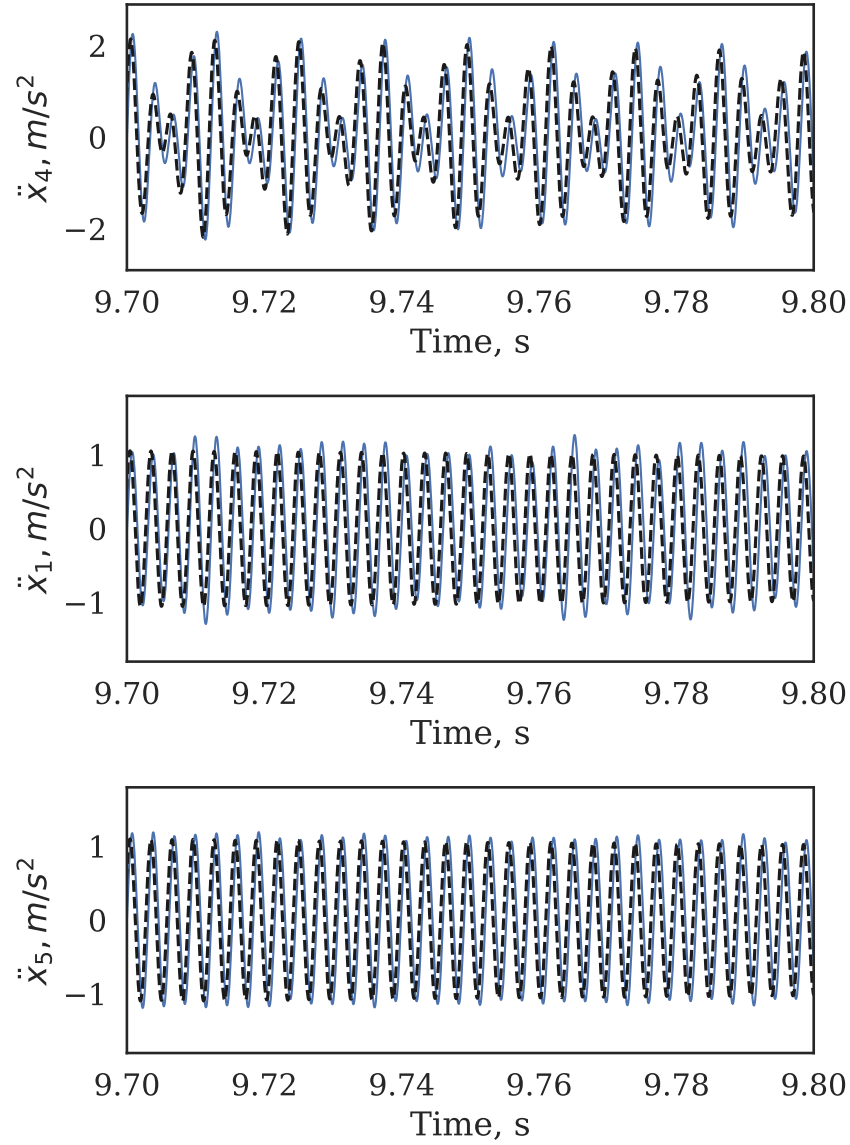


Figure 4.29. Close-up of time histories of S4 beam experimental (blue) and predicted (black) acceleration for three accelerometer locations. Vertical dashed line indicates training/test data boundary.

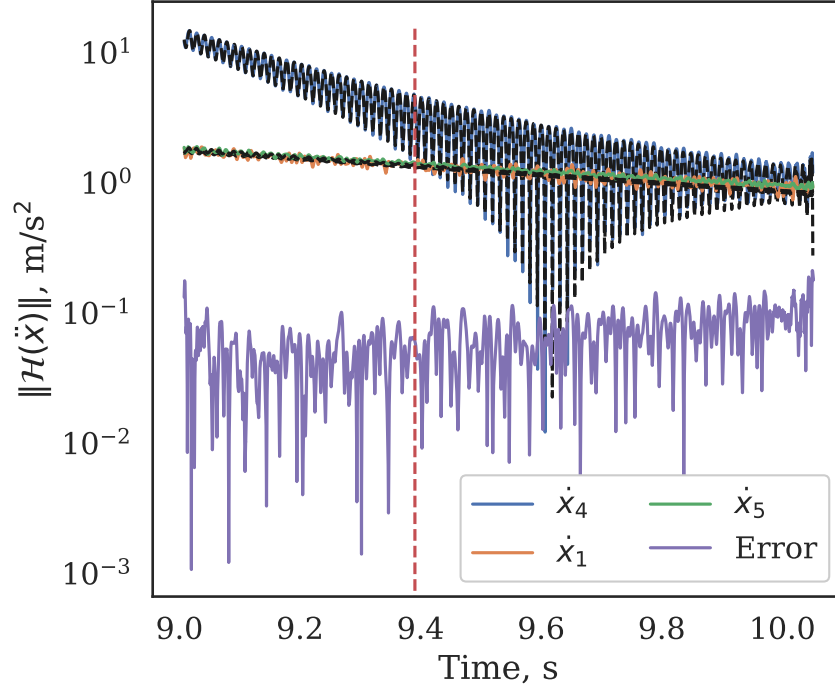


Figure 4.30. Hilbert transform of the S4 beam response at three measurement locations (blue, orange, and green) and the predicted response (dashed black), as well as the error (magenta). Vertical dashed line indicates training/test data boundary.

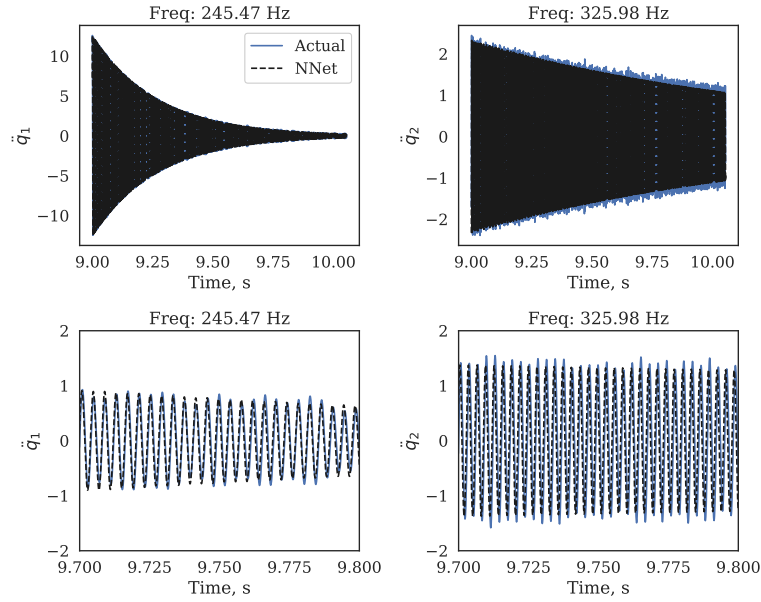


Figure 4.31. Modal displacements for S4 beam experimental response (color) and learned latent space (dashed). The bottom figures show a close-up of the time histories.

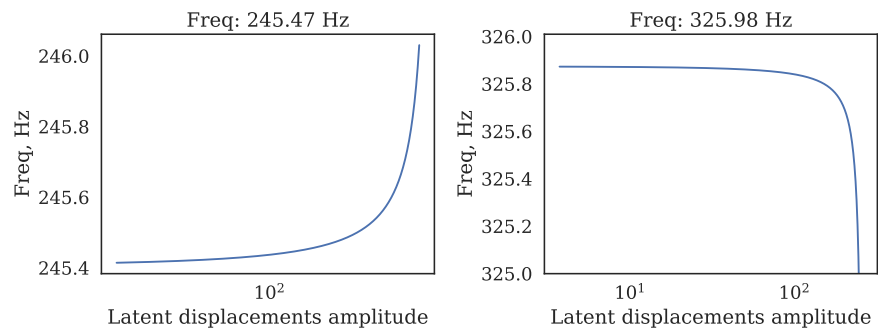


Figure 4.32. First two NNMs of the S4 beam learned model in latent space.

Chapter 5

Nonlinear System Identification in Structures with Isolated Deviatoric Behavior

Chapter 3 and 4 presented a methodology for developing structure-preserving ML models that can act as digital twins of a system. In these previous chapters, it was assumed that no knowledge about the physical system was available. In contrast, this chapter leverages existing knowledge about the nominal state of the structure. This work deals with the types of structures that are increasingly employed in environments that challenge the limits of their operational regime. Furthermore, maintenance cycles are being lengthened to avoid unnecessary and costly interruptions to ongoing operation. Engineering systems are no longer simply placed into service and maintained at regular intervals; there has been substantial investment in developing structural health monitoring (SHM) and damage prognosis (DP) solutions, which seek to provide actionable information about both the current health state of the system as well as predict future limit states (e.g., remaining useful life, time to next maintenance action, etc.) [64]. The ongoing development of SHM/DP approaches, together with advances in computational power, sensing, and data-driven modeling, have enabled the concept of digital twins. Digital twins can have many uses, but they are essentially computational models of engineering systems or components that are faithful representations of the deployed system in the field [47, 75, 262]. Digital Twins represent advanced models with detailed physics that are augmented and updated by ongoing data obtained from sensor systems embedded in the physical systems [272, 165]. As

a result they offer an enhanced ability to monitor the evolving health of a system and predict the future response to both nominal operating conditions and unexpected excursions in loading environments [250, 21, 246, 247].

To be truly effective, digital twins must operate at or near real time, and this requirement may be restrictive for high-fidelity models. The development of Reduced Order Models (ROMs) offers a potential capability to provide simulations on timescales that are commensurate with the need for rapid predictions [119]. For linear systems, such reduction methods are well established and primarily rely on a modal decomposition of the system, retaining only the dominant modes of the system, i.e., those that contain the most energy. Moreover, because of the linear nature of the system appropriately chosen modes are decoupled, so that energy is unable to transfer from one mode to another. If an analyst is able to determine those modes to be included based on the loading environment and initial energy distribution, the error introduced by neglected modes will remain small. However, in the presence of nonlinearities, reduced order models based on modal decomposition no longer segregate the energy across the modes. Significant mode coupling can occur so that energy is exchanged between those modes retained in the ROM as well as to neglected modes. Hence, accurate ROMs required for Digital Twins must account for the presence of nonlinearities in the system.

If the effect of the nonlinearities is in some sense small, the response of the nonlinear system can be characterized in terms of Nonlinear Normal Modes (NNMs), which are generalizations to the vibration modes of the corresponding linear system [207, 96]. If the system is operating at low amplitudes, the nonlinearities are negligible and each NNM can be identified with a linear mode, and vice versa. However, as the amplitude of the response and the effect of the nonlinearities increase an NNM begins to contain contributions from multiple if not all linear modes [273]. Moreover, as mentioned above the NNMs are not decoupled from one another but instead allow for energy transfer between modes. Nonetheless, NNMs provide a useful framework for developing ROMs of nonlinear structural systems. Given a set of NNMs to be used in the model reduction, normal forms can be used to find the resulting equations the contain

the minimal complexity. Such concepts have recently be extended to spectral submanifolds to estimate the NNMs that should be retained for a desired level of accuracy [86, 240].

The nonlinearities present within structural systems are often located within localized regions of the structure [72]. Examples include nonlinearities introduced by joints and interfaces, localized damage, or isolated connections with nonlinear connections. As such, the majority of the system can be characterized as “linear” yet the overall system response is significantly influenced by the presence of the isolated nonlinearities and accurate models must resolve these nonlinear regions. In such cases the linear modes of the system still provide an adequate description of the majority of the structure external to the nonlinear regions, but the coupling between these modes arising from the nonlinearities can no longer be neglected.

One common approach to such systems is substructuring, whereby different components or subregions are characterized separately through their internal dynamics, and are subsequently tied together with boundary degrees-of-freedom that describe the connectivity between regions [49, 209, 270, 5]. If appropriate ROMs are developed for each component, then when the global system is reassembled the resulting global model is itself a ROM [82, 243]. However, the global modes of the system may not be well approximated by the individual component modes, thus necessitating an increase in the order of the component ROMs to accurately describe the global dynamics [129].

A recent approach seeks to combine advantages of substructuring with the global modes that serve as the basis of NNMs. In particular, for systems with regions that contain the nonlinearities, [199] formulate the equations of motion in such a way as to isolate the effect of the nonlinearities on the underlying linear structure. In this work the effect of the nonlinearities is introduced through a deviatoric force acting on the ideal linear system, representing only the contribution from the nonlinearities and localized to the boundary of the nonlinear region. As a result, the global modes of the linear structure continue to serve as framework for the development of ROMs, while only the isolated nonlinear regions need to be resolved in an associated nonlinear system whose domain is only over the nonlinear region [199]. The present

work takes a step further, by using a structure-preserving Machine Learning (ML) algorithm in the form of a multilayer perceptron to predict the deviatoric force arising from the nonlinearities within the isolated region. As a result, the resolution of the nonlinear component is replaced by predictions from the ML system.

This organization of this chapter is as follows: Section 2 reviews the decomposition presented in [199] and identifies the deviatoric force component that describes the effect of the isolated nonlinearities on the remaining structure. The structure-preserving machine learning approach used to predict these deviatoric forces is described in Section 3. In Section 4 the developed data-driven approach is illustrated on a numerical example consisting of the discretization of a rod with an internal section that contains both cubic nonlinearities as well as hysteretic damping. Finally, Section 5 summarizes the formulation and results, and discusses limitations and future approaches for this methodology.

5.0.1 Isolated Nonlinearities

We consider a general structure that can be decomposed into two adjacent regions $\mathcal{C}_1$ and $\mathcal{C}_2$ as illustrated in Figure 5.1. Note that region $\mathcal{C}_1$ is described by the internal variables $\underline{\mathbf{x}}_1 \in \mathbf{R}^{N_1}$, while $\underline{\mathbf{x}}_2 \in \mathbf{R}^{N_2}$ are the corresponding internal variables for $\mathcal{C}_2$. In this decomposition it is assumed that any nonlinearities and/or mistunings are localized within the region $\mathcal{C}_2$, so that the equations of motion within $\mathcal{C}_1$ are linear and assumed known¹. As a result, the equations of motion for this structure can be written as

$$\mathbf{M}\ddot{\underline{\mathbf{x}}} + \mathbf{C}\dot{\underline{\mathbf{x}}} + \mathbf{K}\underline{\mathbf{x}} + \underline{\mathbf{N}}_\star = \underline{\mathbf{f}}(t), \underline{\mathbf{N}}_\star = [\underline{\mathbf{0}} \quad \underline{\mathbf{N}}(\underline{\mathbf{x}}_2)], \quad \underline{\mathbf{N}} \in \mathbf{R}^{N_2}, \quad \underline{\mathbf{x}} = [\underline{\mathbf{x}}_1, \underline{\mathbf{x}}_2] \in \mathbf{R}^{(N_1+N_2)} \quad (5.1)$$

The nonlinearities are localized to $\mathcal{C}_2$ through $\underline{\mathbf{N}} \in \mathbf{R}^{N_2}$ and could include isolated nonlinearities arising from, for example, the presence of joints and interfaces, nonlinear attachments, or accumulating damage. The following formulation is derived from [199], in which the isolated

¹The terms described in $\underline{\mathbf{N}}$ could also include linear mistunings, but any contribution to $\underline{\mathbf{N}}$ will be referred to as a “nonlinearity”.

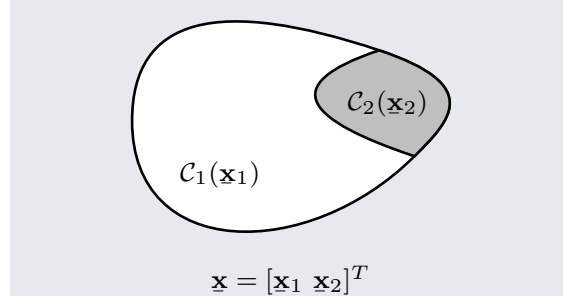


Figure 5.1. General Structure; the nonlinearities are localized within the region $\mathcal{C}_2$.

region was assumed to separate two linear regions. This restriction is relaxed in the current work, so that the reduction presented below holds for a nonlinear region that exists anywhere in the structure and need not be connected [258].

Isolated System

Within the linear region $\mathcal{C}_1$ the coordinates are further subdivided as $\underline{\mathbf{x}}_1 = [\underline{\mathbf{x}}_c \quad \underline{\mathbf{x}}_\alpha]$, where $\underline{\mathbf{x}}_\alpha$ are specifically those degrees-of-freedom that are coupled to the isolated region $\mathcal{C}_2$. Likewise, within $\mathcal{C}_2$ the coordinates that are coupled to region $\mathcal{C}_1$ are identified as $\underline{\mathbf{x}}_\beta$ so that $\underline{\mathbf{x}}_2 = [\underline{\mathbf{x}}_\beta \quad \underline{\mathbf{x}}_n]$. With this, the stiffness matrix $\mathbf{K}$ and displacement vector $\underline{\mathbf{x}}$ can be expressed as

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{cc} & \mathbf{K}_{c\alpha} & \mathbf{0} & \mathbf{0} \\ \mathbf{K}_{c\alpha}^T & \mathbf{K}_{\alpha\alpha} & \mathbf{K}_{\alpha\beta} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{\alpha\beta}^T & \mathbf{K}_{\beta\beta} & \mathbf{K}_{\beta n} \\ \mathbf{0} & \mathbf{0} & \mathbf{K}_{\beta n}^T & \mathbf{K}_{nn} \end{bmatrix}, \quad \mathbf{K}_1 \equiv \begin{bmatrix} \mathbf{K}_{cc} & \mathbf{K}_{c\alpha} \\ \mathbf{K}_{c\alpha}^T & \mathbf{K}_{\alpha\alpha} \end{bmatrix}, \quad \mathbf{K}_2 \equiv \begin{bmatrix} \mathbf{K}_{\beta\beta} & \mathbf{K}_{\beta n} \\ \mathbf{K}_{\beta n}^T & \mathbf{K}_{nn} \end{bmatrix}$$

where $\mathbf{K}_1$ and $\mathbf{K}_2$ are identified as the appropriate stiffness matrices within each component.

Likewise the displacement vector $\underline{\mathbf{x}}$ can be expressed as

$$\underline{\mathbf{x}} = \begin{bmatrix} \underline{\mathbf{x}}_c & \underline{\mathbf{x}}_\alpha & \underline{\mathbf{x}}_\beta & \underline{\mathbf{x}}_n \end{bmatrix}^T, \quad \underline{\mathbf{x}}_c \in \mathbf{R}^{N_c}, \quad \underline{\mathbf{x}}_\alpha \in \mathbf{R}^{N_\alpha}, \quad \underline{\mathbf{x}}_\beta \in \mathbf{R}^{N_\beta}, \quad \underline{\mathbf{x}}_n \in \mathbf{R}^{N_n}, \quad (5.2)$$

For this system it is expected that $N_c \gg N_n > (N_\alpha, N_\beta)$. Note that this is similar to the initial formulation for component mode synthesis, with $\underline{\mathbf{x}}_\alpha$ and $\underline{\mathbf{x}}_\beta$ serving as the boundary degrees-of-freedom for $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively.

Over the isolated region $\mathcal{C}_2$ the response of the as-built, as-deployed system can be expressed as

$$\underline{\mathbf{x}}_2 = \underline{\mathbf{y}} + \underline{\mathbf{z}}. \quad (5.3)$$

where $\underline{\mathbf{y}} \in \mathbf{R}^{N_2}$ is introduced as the response of an ideal system to be subsequently identified, while $\underline{\mathbf{z}} \in \mathbf{R}^{N_2}$ is the difference between the ideal and actual response. With this, the elastic term $\mathbf{K} \underline{\mathbf{x}}$ is

$$\mathbf{K} \underline{\mathbf{x}} = \begin{bmatrix} \mathbf{K}_{cc} & \mathbf{K}_{c\alpha} & \mathbf{0} & \mathbf{0} \\ \mathbf{K}_{c\alpha}^T & \mathbf{K}_{\alpha\alpha} & \mathbf{K}_{\alpha\beta} & \mathbf{0} \\ \mathbf{0} & \mathbf{K}_{\alpha\beta}^T & \mathbf{K}_{\beta\beta} & \mathbf{K}_{\beta n} \\ \mathbf{0} & \mathbf{0} & \mathbf{K}_{\beta n}^T & \mathbf{K}_{nn} \end{bmatrix} \begin{bmatrix} \underline{\mathbf{x}}_c \\ \underline{\mathbf{x}}_\alpha \\ \underline{\mathbf{y}}_\beta + \underline{\mathbf{z}}_\beta \\ \underline{\mathbf{y}}_n + \underline{\mathbf{z}}_n \end{bmatrix} = \mathbf{K} \underline{\mathbf{w}} + \begin{bmatrix} \underline{\mathbf{0}}_c \\ \mathbf{K}_{\alpha\beta} \underline{\mathbf{z}}_\beta \\ \underline{\mathbf{0}}_2 \end{bmatrix} + \begin{bmatrix} \underline{\mathbf{0}}_c \\ \underline{\mathbf{0}}_\alpha \\ \mathbf{K}_2 \underline{\mathbf{z}}, \end{bmatrix}$$

where $\underline{\mathbf{w}}$ is a mixed displacement vector can be defined as

$$\underline{\mathbf{w}} \equiv \begin{bmatrix} \underline{\mathbf{x}}_c & \underline{\mathbf{x}}_\alpha & \underline{\mathbf{y}}_\beta & \underline{\mathbf{y}}_n \end{bmatrix}^T = \begin{bmatrix} \underline{\mathbf{x}}_1 & \underline{\mathbf{y}} \end{bmatrix}^T. \quad (5.4)$$

In the above expression $\underline{\mathbf{0}}_c$ the subscript indicates the dimension of the null vector, so that $\underline{\mathbf{0}}_c \in \mathbf{R}^{N_c}$ and similarly for other expressions. The term $\mathbf{K}_{\alpha\beta} \underline{\mathbf{z}}_\beta$ acts on the degrees-of-freedom associated with $\underline{\mathbf{x}}_\alpha$, with the coupled term $\mathbf{K}_{\alpha\beta}^T \underline{\mathbf{x}}_\alpha$ included in $\mathbf{K} \underline{\mathbf{w}}$.

With the linear inertial and damping terms decomposed similarly, the equations of motion

can be written as

$$\begin{aligned}
 & \underbrace{\left\{ \mathbf{M} \ddot{\mathbf{w}} + \begin{bmatrix} \mathbf{0}_c \\ \mathbf{M}_{\alpha\beta} \ddot{\mathbf{z}}_\beta \\ \mathbf{0}_2 \end{bmatrix} + \begin{bmatrix} \mathbf{0}_c \\ \mathbf{0}_\alpha \\ \mathbf{M}_2 \ddot{\mathbf{z}} \end{bmatrix} \right\}}_{\mathbf{M} \ddot{\mathbf{x}}} + \underbrace{\left\{ \mathbf{C} \dot{\mathbf{w}} + \begin{bmatrix} \mathbf{0}_c \\ \mathbf{C}_{\alpha\beta} \dot{\mathbf{z}}_\beta \\ \mathbf{0}_2 \end{bmatrix} + \begin{bmatrix} \mathbf{0}_c \\ \mathbf{0}_\alpha \\ \mathbf{C}_2 \dot{\mathbf{z}} \end{bmatrix} \right\}}_{\mathbf{C} \dot{\mathbf{x}}} \\
 & + \underbrace{\left\{ \mathbf{K} \mathbf{w} + \begin{bmatrix} \mathbf{0}_c \\ \mathbf{K}_{\alpha\beta} \mathbf{z}_\beta \\ \mathbf{0}_2 \end{bmatrix} + \begin{bmatrix} \mathbf{0}_c \\ \mathbf{0}_\alpha \\ \mathbf{K}_2 \mathbf{z} \end{bmatrix} \right\}}_{\mathbf{K} \mathbf{x}} + \underbrace{\begin{bmatrix} \mathbf{0}_c \\ \mathbf{0}_\alpha \\ \mathbf{N}(\mathbf{y} + \mathbf{z}) \end{bmatrix}}_{\mathbf{N}_*(\mathbf{x})} = \mathbf{f}(t)
 \end{aligned}$$

These equations of motion are now partitioned into two separate systems of equations terms of $\mathbf{w}$ and $\mathbf{z}$ as

$$\begin{aligned}
 \mathbf{M} \ddot{\mathbf{w}} + \mathbf{C} \dot{\mathbf{w}} + \mathbf{K} \mathbf{w} &= \mathbf{f}(t) - \underline{Q}_*, \quad \mathbf{M}_2 \ddot{\mathbf{z}} + \mathbf{C}_2 \dot{\mathbf{z}} + \mathbf{K}_2 \mathbf{z} + \mathbf{N}(\mathbf{y} + \mathbf{z}) = \mathbf{0}, \\
 \underline{Q}_* &= \begin{bmatrix} \mathbf{0}_c & \underline{Q} & \mathbf{0}_2 \end{bmatrix}^T, \quad \underline{Q} \equiv \mathbf{M}_{\alpha\beta} \ddot{\mathbf{z}}_\beta + \mathbf{C}_{\alpha\beta} \dot{\mathbf{z}}_\beta + \mathbf{K}_{\alpha\beta} \mathbf{z}_\beta,
 \end{aligned} \tag{5.5}$$

which are coupled together through the ideal displacement $\mathbf{y}$ and deviatoric force $\underline{Q}$. Within $\mathcal{C}_2$ the deviatoric response $\mathbf{z}$ is driven by the ideal system $\mathbf{y}$ through the nonlinearities, while $\mathcal{C}_1$ is coupled to the deviatoric displacement $\mathbf{z}$ through $\underline{Q}$. The degree-of-freedom of the coupling terms can be significantly smaller than the overall degree-of-freedom of the system, so that

$$[\mathbf{M}_{\alpha\beta}, \mathbf{C}_{\alpha\beta}, \mathbf{K}_{\alpha\beta}] \in \mathbf{R}^{(N_\alpha \times N_\beta)}, \quad \text{with} \quad (N_\alpha \times N_\beta) \ll N^2. \tag{5.6}$$

Within the mixed displacement vector $\mathbf{w}$, $\mathbf{x}_1$ represents the exact response in the exterior region $\mathcal{C}_1$ while $\mathbf{y}$ describes the response of the ideal system in the isolated region. In addition, the mixed formulation is linear in terms of $\mathbf{w}$, while the nonlinear behavior is contained only in the equations of motion for $\mathbf{z}$, which is localized to the isolated region $\mathcal{C}_2$. The equation of

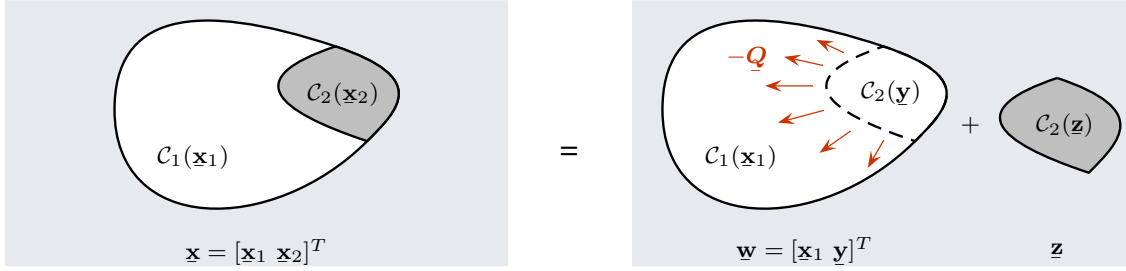


Figure 5.2. The nonlinear isolated region ($\mathcal{C}_2$) can be decomposed from the linear external region ($\mathcal{C}_1$).

motion in terms of $\mathbf{w}$ is referred to as the ideal linear system, with known mass, linear damping, and linear stiffness matrices defined over the entire structure. As illustrated in Figure 5.2, this decomposition isolates the nonlinear regions from the underlying linear representation of the overall system, while the effect of the nonlinearities on the response are introduced as a generalized force acting at the boundary $\delta\mathcal{C}_2$.

5.0.2 Structure-Preserving Machine Learning

The present work seeks to use machine learning approaches to estimate an appropriate constitutive relationship between $\mathbf{Q}$ and the response of the structure $\mathbf{x}_\alpha$, both defined at the boundary of the isolated region $\mathcal{C}_2$. Thus the exact numerical simulation incorporating the nonlinearities within $\mathcal{C}_2$ can be replaced in the simulation framework with an appropriately trained neural network.

The structure-preserving ML formulation is based on the work presented in [184] but is extended to handle nonlinear force identification. The dynamic system is represented using a Hamiltonian and Lagrangian formulation because this framework provides a generalized way of constructing the equations of motion in terms of the energy, which are scalar fields and therefore may require smaller networks to represent them. We start by defining the kinetic energy $\mathcal{T}$, the

potential energy $\mathcal{V}$, and the Rayleigh damping $\mathcal{J}$ terms of the known system as

$$\mathcal{T}(\dot{\underline{\mathbf{w}}}) = \frac{1}{2} \dot{\underline{\mathbf{w}}}^T \mathbf{M} \dot{\underline{\mathbf{w}}}, \quad \mathcal{V}(\underline{\mathbf{w}}) = \frac{1}{2} \underline{\mathbf{w}}^T \mathbf{K} \underline{\mathbf{w}}, \quad \mathcal{J}(\dot{\underline{\mathbf{w}}}) = \frac{1}{2} \dot{\underline{\mathbf{w}}}^T \mathbf{C} \dot{\underline{\mathbf{w}}} \quad (5.7)$$

From which the Lagrangian can be obtained as

$$\mathcal{L}(\underline{\mathbf{w}}, \dot{\underline{\mathbf{w}}}) = \mathcal{T}(\dot{\underline{\mathbf{w}}}) - \mathcal{V}(\underline{\mathbf{w}}). \quad (5.8)$$

The Lagrangian is used to compute the momentum as

$$p_k = \frac{\partial \mathcal{L}(\underline{\mathbf{w}}, \dot{\underline{\mathbf{w}}})}{\partial \dot{w}_k}. \quad (5.9)$$

Next, the Hamiltonian is computed from the Lagrangian and the generalized momenta as

$$H(\underline{\mathbf{w}}, \dot{\underline{\mathbf{w}}}) = \underline{\mathbf{p}} \cdot \dot{\underline{\mathbf{w}}} - \mathcal{L}(\underline{\mathbf{w}}, \dot{\underline{\mathbf{w}}}), \quad (5.10)$$

and then the momentum rates are computed from the Hamiltonian, the generalized forces, and the dissipative terms as

$$\dot{p}_k = -\frac{\partial H(\underline{\mathbf{w}}, \dot{\underline{\mathbf{w}}})}{\partial w_k} + Q_k^{\text{nc}} - \frac{\partial \mathcal{J}(\dot{\underline{\mathbf{w}}})}{\partial \dot{w}_k}, \quad (5.11)$$

where Q_k^{nc} is the k^{th} term of the generalized force vector, including both the external forces and the deviatoric term,

$$\underline{\mathbf{Q}}^{\text{nc}} = \underline{\mathbf{f}}(t) + \underline{\mathbf{Q}}_{\star}. \quad (5.12)$$

The deviatoric term is modeled with a multilayer perceptron (MLP) composed of Swish activation functions and is augmented with a dictionary of polynomial terms up to K^{th} degree. The inclusion of polyonimial terms is intended to provide flexibility and augmented interpretability in case

a polynomial form is desired. Moreover, previous work [28] has shown that polynomials are appropriate for representing nonlinearities in dynamical systems. The MLP takes in the boundary terms as input,

$$\mathbf{Q}_\star = \mathcal{G}(\mathbf{z}_\beta, \dot{\mathbf{z}}_\beta) + \sum_{i,j,\dots,p}^{i+j+\dots=p=K} \lambda_{i,j,\dots,p} z_{\beta 1}^i z_{\beta 2}^j \dots z_{\beta k}^p + \gamma_{i,j,\dots,p} \dot{z}_{\beta 1}^i \dot{z}_{\beta 2}^j \dots \dot{z}_{\beta k}^p, \quad (5.13)$$

where $\mathbf{z}_\beta$ is defined as the discrepancy between the measured and ideal response at the boundary of the isolated region. The coefficients λ and γ are learned during training, and the function $\mathcal{G}(\cdot)$ represents the MLP. Note that the form of the polynomial terms must be chosen prior to training the MLP. In the current example the polynomial terms are purely displacement or velocity based, with no cross terms included. This choice is somewhat arbitrary and any specific insight into the nonlinearities present could be leveraged in this choice. The overall structure of the MLP is illustrated in Figure 5.3.

Once the vector of generalized forces, including the contribution from the deviatoric force term, is defined, the accelerations can be predicted as

$$\ddot{\mathbf{w}}_n = \mathbf{M}_{ni}^{-1}(\dot{\mathbf{p}}_i - M_{ij,k} \dot{\mathbf{w}}_k \dot{\mathbf{w}}_j), \quad (5.14)$$

where $\mathbf{M}$ is the known mass matrix of the idealized system, and the momenta rate $\dot{\mathbf{p}}$ is defined in Eq. (5.11). As $\mathbf{w}$ is a vector of known quantities, the learning can be supervised by minimizing the squared error between the predicted and known accelerations,

$$\mathcal{L}_{\text{recon}} = \|\ddot{\mathbf{w}} - \ddot{\mathbf{w}}\|_2^2, \quad (5.15)$$

Once trained, the deviatoric force can be computed from the discrepancy term, $\mathbf{z}_\beta = \mathbf{x}_\beta - \mathbf{y}_\beta$, as

$$\mathbf{Q}_\star = \mathcal{N}(\mathbf{z}_\beta, \dot{\mathbf{z}}_\beta), \quad (5.16)$$

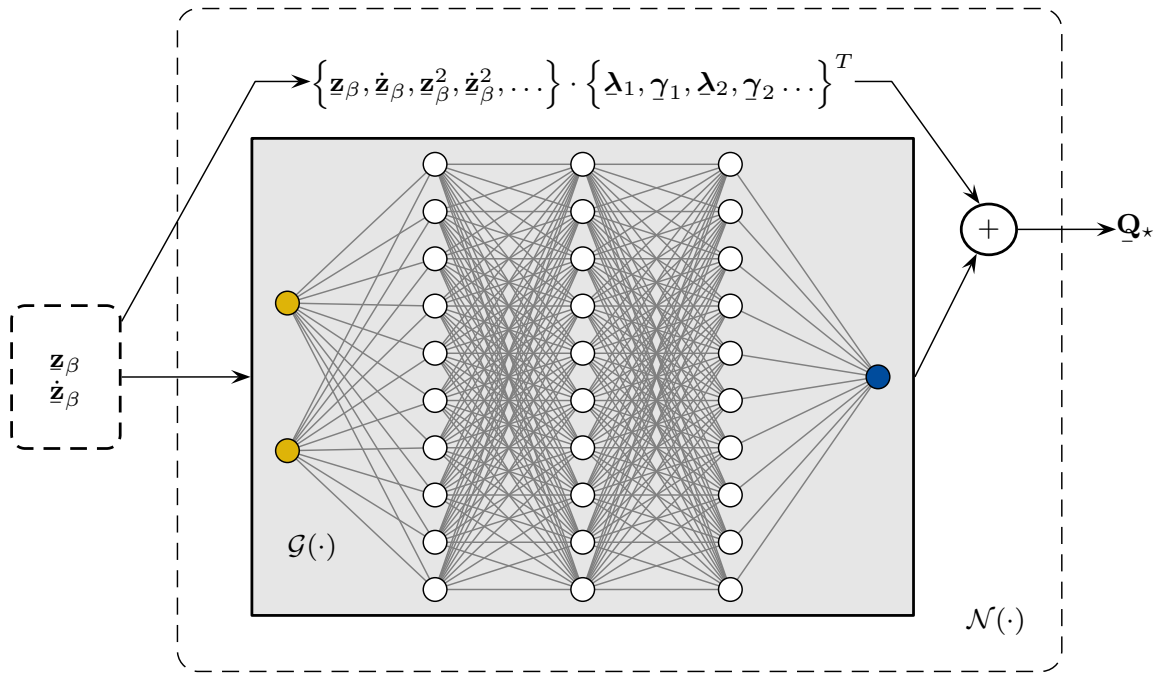


Figure 5.3. Architecture of MLP augmented with polynomial terms.

for any set of initial conditions or forcing function. It should be noted that the deviatoric force is not directly used for training. Instead, the residual of the equation of motion of the system is used to train the network. This formulation enforces preservation of the structure of the dynamic system.

Implementation and Deployment of ML system

The neural network $\mathcal{G}(\cdot)$ and the coefficients of the dictionary of polynomial terms defined in Eq. (5.13) are determined during training by minimizing the loss term defined in Eq. (5.15). To this end, an Adam optimizer with learning rate set to 10^{-4} was used. For all cases, the network $\mathcal{G}(\cdot)$ consisted of three hidden layers with 10 units and a Swish activation function [204], with the last layer being linear. The implementation of the neural network was done in Jax [23]. Training and inference was performed on a Google Colab session with one GPU. Training for the examples presented in this chapter took on the order of 1 hour to reach convergence (i.e., the loss stopped changing). The online evaluation of the deviatoric force takes on the order of microseconds.

An additional loss term was employed to ensure mathematical consistency in the network. This loss term ensures that the momentum rate represents the time derivative of the momentum. To perform this check, the residual of a multistep integration step was evaluated as

$$\mathcal{L}_{\text{diff}} = \left\| \sum_{j=0}^m v_j \mathbf{p}_{k+j} - h \sum_{j=0}^m \theta_j \dot{\mathbf{p}}_{k+j} \right\|_2^2, \quad (5.17)$$

where v and θ are coefficients from the Butcher tableau corresponding to the Adams-Bashforth linear multistep integration method, and h is the time step associated with the training data. The $\mathbf{p}_{k+j}$ term represents the $(k+j)^{\text{th}}$ time step in the $\mathbf{p}$ vector and m represents the order of the integration method. In this work $m = 3$ was used. This loss term was added to the reconstruction

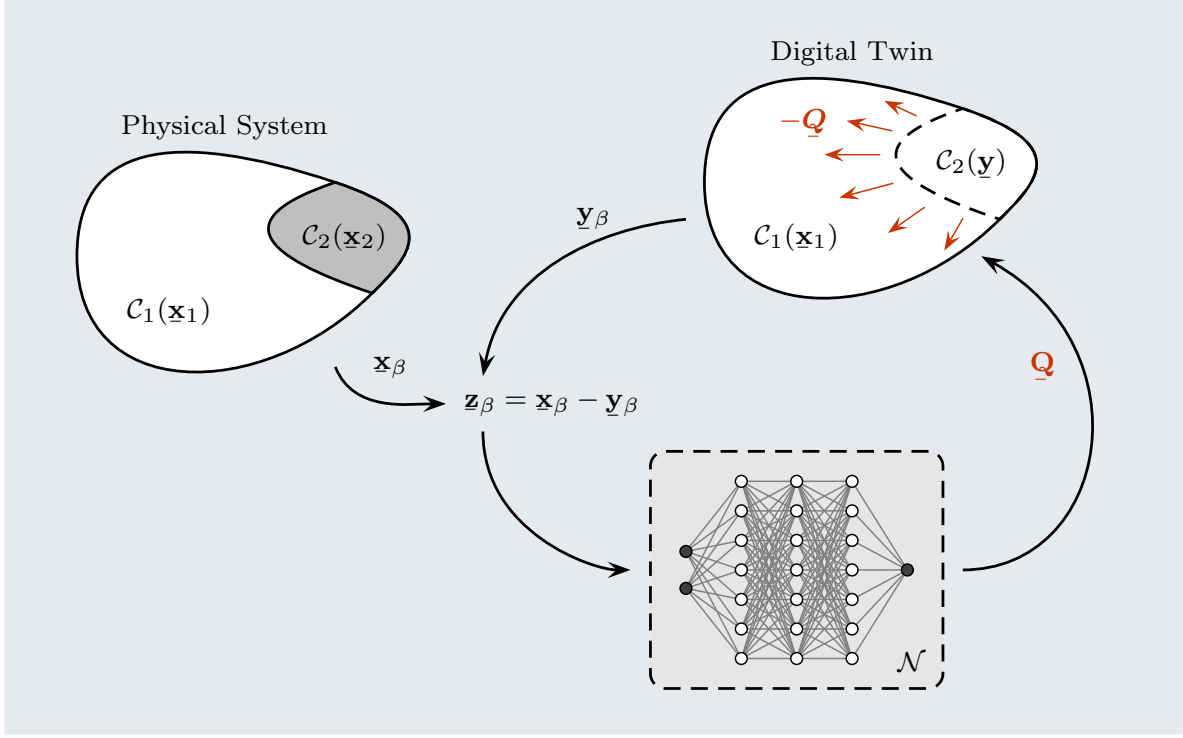


Figure 5.4. Data-driven simulation incorporating the MLP.

loss computed in Eq. (5.15) to form the total loss term as

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \mathcal{L}_{\text{diff}}. \quad (5.18)$$

Finally, the incorporation of the ML system within the overall data-driven simulation is illustrated in Figure 5.4. In all the examples presented in this chapter, a single realization with random initial conditions was used for training. All results presented correspond to different random initial conditions or forcing functions to those used for training.

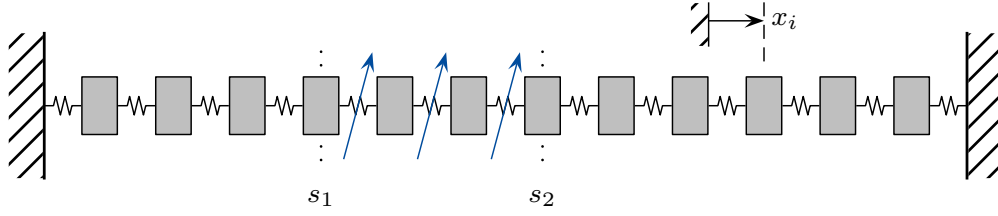


Figure 5.5. Example system with isolated nonlinearities. The nonlinearities are localized within the region $s \in (s_1, s_2)$, while the remainder of the rod is linear.

5.1 Numerical Simulation

5.1.1 Example

As an example, consider a chain of N elements ($\sigma_i = i/(N+1) \in [0, 1]$, $i = 1, \dots, N$) with nonlinearities over the interval $\sigma_i \in [s_1, s_2] \equiv \mathcal{C}_2$ as shown in Figure 5.5. The underlying linear system is described by mass and stiffness matrices of the form

$$\mathbf{M} = \frac{1}{N} \begin{bmatrix} 1 & & & \\ & 1 & & \mathbf{0} \\ & & \ddots & \\ & \mathbf{0} & & 1 \\ & & & & 1 \end{bmatrix}, \quad \mathbf{K} = (N+1) \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & & & \mathbf{0} \\ & & \ddots & & \\ & \mathbf{0} & & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}, \quad (5.19)$$

while the system is assumed to be proportionally damped with $\mathbf{C} \equiv \xi \mathbf{K}$.

In this formulation the degrees-of-freedom associated with boundaries of the nonlinear interval have indices

$$\text{(left)} \quad \ell = s_1 \cdot (N+1), \quad \text{(right)} \quad r = s_2 \cdot (N+1), \quad (5.20)$$

so that the state vector $\underline{\mathbf{x}}$ is decomposed as

$$\underline{\mathbf{x}}_c = \begin{bmatrix} x_1 & \dots & x_{\ell-2} & x_{r+2} & \dots & x_N \end{bmatrix}^T, \quad \underline{\mathbf{x}}_\alpha = \begin{bmatrix} x_{\ell-1} & x_{r+1} \end{bmatrix}^T, \quad (5.21)$$

$$\underline{\mathbf{x}}_\beta = \begin{bmatrix} x_\ell & x_r \end{bmatrix}^T, \quad \underline{\mathbf{x}}_n = \begin{bmatrix} x_{\ell+1} & \dots & x_{r-1} \end{bmatrix}^T. \quad (5.22)$$

Note that regardless of the number of elements, both $\underline{\mathbf{x}}_\alpha$ and $\underline{\mathbf{x}}_\beta$ only contain two coordinates each. The deviatoric force $\underline{Q}$ is then applied to elements $x_{\ell-1}$ and x_{r+1} at the boundary between $\mathcal{C}_1$ and $\mathcal{C}_2$. The associated coupling matrices from Eq. (5.5) are identified as

$$\mathbf{M}_{\alpha\beta} = \mathbf{0}, \quad \mathbf{K}_{\alpha\beta} = (N+1) \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad \mathbf{C}_{\alpha\beta} = \xi \mathbf{K}_{\alpha\beta}. \quad (5.23)$$

As a result the force $\underline{Q}$ can be written as

$$\underline{Q} = \begin{bmatrix} Q_{\ell-1} \\ Q_{r+1} \end{bmatrix} = (N+1) \begin{bmatrix} -(z_\ell + \xi \dot{z}_\ell) \\ -(z_r + \xi \dot{z}_r) \end{bmatrix}. \quad (5.24)$$

Nonlinearities

The nonlinearities in the isolated region are assumed to contain linear mistuning and cubic nonlinearities in terms of both stiffness and damping. Finally, hysteretic nonlinear damping is also present between the elements within $\mathcal{C}_2$, so that the nonlinear force is represented as

$$f_{nl,i} = \kappa_i (x_{i+1} - x_i) + \chi_i (\dot{x}_{i+1} - \dot{x}_i) + \eta_i (x_{i+1} - x_i)^3 + \zeta_i (\dot{x}_{i+1} - \dot{x}_i)^3 + g_{hys,i}. \quad (5.25)$$

The hysteretic force $g_{hys,i}$ is assumed to arise from a single Jenkins element, with a regularized friction model to eliminate any stick-slip discontinuities in the element. As illustrated in Figure 5.6, with $r_i \equiv u_{i+1} - u_i$ the hysteretic force $g_{hys,i}$ is assumed to depend on $\dot{r}_i$, so that $g_{hys,i} \equiv h(\dot{r}_i)$. This element is in series with the elastic components, so that

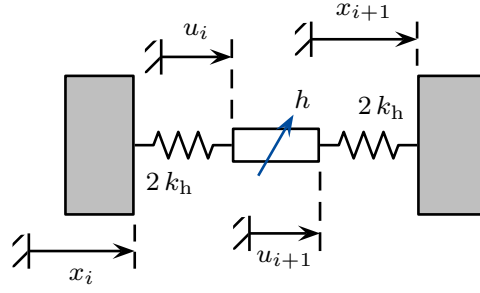


Figure 5.6. Hysteretic force model. Note that this is in addition to the elastic and dissipative coupling described in Eq. (5.25).

$$2k_h (u_i - x_i) = h(\dot{r}_i) = 2k_h (x_{i+1} - u_{i+1}). \quad (5.26)$$

Simplifying these and eliminating the internal displacements (u_i, u_{i+1}) yields

$$h(\dot{r}_i) = k_h ((x_{i+1} - x_i) - r_i). \quad (5.27)$$

Finally inverting the nonlinear relation, an evolution equation for the internal state variable z_i is obtained as

$$\dot{r}_i = h^{-1} \left(k_h ((x_{i+1} - x_i) - r_i) \right). \quad (5.28)$$

Finally, the hysteretic component applied to the elements is

$$g_{\text{hys},i} = k_h ((x_{i+1} - x_i) - r_i), \quad (5.29)$$

so that only the function h^{-1} need be defined in this formulation. One common regularization of Coulomb friction assumes that $h(\dot{r}) \equiv \mu \arctan(\dot{r}/\varepsilon)$, where ε is a regularization parameter.

With this, the evolution equation for the internal variable reduces to

$$\dot{r}_i = \varepsilon \tan \left(\frac{k_h}{\mu} ((x_{i+1} - x_i) - r_i) \right) \quad (5.30)$$

5.1.2 Simulation

Consider a system with $N = 64$. The nonlinear region $\mathcal{C}_2$ is located in the interval $(s_1, s_2) = (0.25, 0.35)$, so that the isolated region exists between elements $\ell \equiv 16$ and $r \equiv 22$. The deviatoric force across the isolated region is identified as

$$\delta Q \equiv Q_{23} - Q_{15} = -(N+1) \left\{ \xi ((\dot{x}_{22} - \dot{x}_{16}) - (\dot{y}_{22} - \dot{y}_{16})) + ((x_{22} - x_{16}) - (y_{22} - y_{16})) \right\}, \quad (5.31)$$

with the displacement and velocity across the interval defined as

$$\Delta \equiv x_{22} - x_{16}, \quad \dot{\Delta} \equiv \dot{x}_{22} - \dot{x}_{16}. \quad (5.32)$$

Unless noted otherwise the parameters of the nonlinear region are chosen as

$$\begin{aligned} \kappa_i &= 0, & \chi_i &= 0, & \eta_i &= 2.621 \times 10^5, & \zeta_i &= 2.621 \times 10^3, \\ k_{h,i} &= 6.50 \times 10^1, & \mu_i &= 1.00 \times 10^{-1}, & \varepsilon_i &= 1.00 \times 10^{-2}, \end{aligned} \quad (5.33)$$

with $\xi = 10^{-4}$. The initial conditions of the system are specified in terms of the modal displacements and velocities $(\mathbf{q}(0), \dot{\mathbf{q}}(0))$, where

$$\mathbf{x}(0) = \Phi \mathbf{q}(0), \quad \dot{\mathbf{x}}(0) = \Phi \dot{\mathbf{q}}(0), \quad (5.34)$$

with Φ the modal transformation matrix.

The present work then seeks to use a machine learning approach to estimate an appropriate

constitutive relationship between $(\mathbf{z}_\beta, \dot{\mathbf{z}}_\beta)$ and $\mathbf{Q}$, which can be used within the simulation framework. The ML network is trained using the initial $T = 2.50$ interval of data. For the training the system is simulated with zero initial displacement, so that $\mathbf{q}(0) = \mathbf{0}$ and nonzero initial modal velocities for the lowest three modes, with

$$\begin{aligned}\dot{q}_1(0) &= 4.4721 \times 10^{-1}, & \dot{q}_2(0) &= 2.2361 \times 10^{-1}, \\ \dot{q}_3(0) &= 3.1623 \times 10^{-1}.\end{aligned}\tag{5.35}$$

Although the identified modes decouple the undamped linear system, in the presence of non-proportional damping and/or nonlinearities significant modal coupling may occur. To reduce this initial coupling, the remaining initial modal velocities are chosen so that the initial modal accelerations vanish. Specifically the initial modal acceleration can be written as

$$\ddot{\mathbf{q}}(0) = - [(\Phi^T \mathbf{C} \Phi) \dot{\mathbf{q}}(0) + (\Phi^T \mathbf{K} \Phi) \mathbf{q}(0) + \Phi^T \mathbf{N}_*(\Phi \mathbf{q}(0), \Phi \dot{\mathbf{q}}(0))], \tag{5.36}$$

and with $\mathbf{q}(0) \equiv \mathbf{0}$ these nonlinear equations are solved numerically for $\dot{\mathbf{q}}(0)$. This approach is similar to Quasi-static Modal Analysis [67].

Figure 5.7a presents the response of the underlying linear system, and note that the modal amplitudes are almost constant given the small value of λ . In contrast, the introduction of nonlinearities within $\mathcal{C}_2$ leads to the response shown in Figure 5.7b, and is markedly different. While some dissipation is observed in the fundamental mode, the damping is most influential in the third mode as described by q_3 . Note that the location of the interface is centered around $s = 0.30$, nearby an extrema of the second mode so that the coupling between mode 2 and the nonlinearities is expected to be weak, and q_2 is relatively unaffected by the nonlinearities. Finally, both the displacement at $s = 0.75^2$, given by $x|_{s=0.75}$, and Δ , the displacement across $\mathcal{C}_2$ are altered by the nonlinearities. The presence of the nonlinearities clearly influences the overall

²This point was chosen so that contributions from the three lowest modes are included in the reconstructed response. Note that the second mode has a node at the midpoint.

behavior of the system and any representation must accurately incorporate these components.

The results of this procedure are illustrated in Figure 5.8, comparing the simulation results using the ML representation and those obtained by direct numerical simulation. While the training was performed on the initial 4000 data points ($T \sim 2.50$) the remaining simulation for the ML system was performed with the trained network. In Figure 5.8a the force δQ is shown as a function of $(\Delta, \dot{\Delta})$, with the direct numerical simulation shown as a solid line and the predictions using the ML system shown dashed. The projections onto the various planes are also shown. The two responses are almost coincident. In Figure 5.8b the time series for δQ , Δ , and $\dot{\Delta}$ are shown for the ML system. The error between the direct numerical simulation and ML system is defined as

$$\text{Err}_{\delta Q} \equiv \delta Q_{\text{Direct numerical simulation}} - \delta Q_{\text{ML system}},$$

and the various errors are shown in Figure 5.8c.

We further quantify the magnitude of the error between the direct numerical simulation and the ML system through the Hilbert transform $\mathcal{H}[\bullet](t)$ of each signal [66]. In particular the amplitude of this complex quantity represents the magnitude of the signal. In Figure 5.9 the magnitude of $\mathcal{H}[\delta Q](t)$ is shown with the solid line, while the magnitude of the error, indicated with the thin line, is several orders of magnitude smaller.

Once trained, the network is able to be robustly applied for additional operating conditions. In the remaining numerical examples, the ML system that was trained with the initial conditions given in Eq. (5.35) is applied as various characteristics of the system are changed. For example, when the initial conditions are changed, so that

$$\begin{aligned} \dot{q}_1(0) &= 0, & \dot{q}_2(0) &= 4.4721 \times 10^{-1}, \\ \dot{q}_3(0) &= 3.1623 \times 10^{-1}. \end{aligned} \tag{5.37}$$

As illustrated in Figure 5.10 the ML representation is again able to capture the response of the

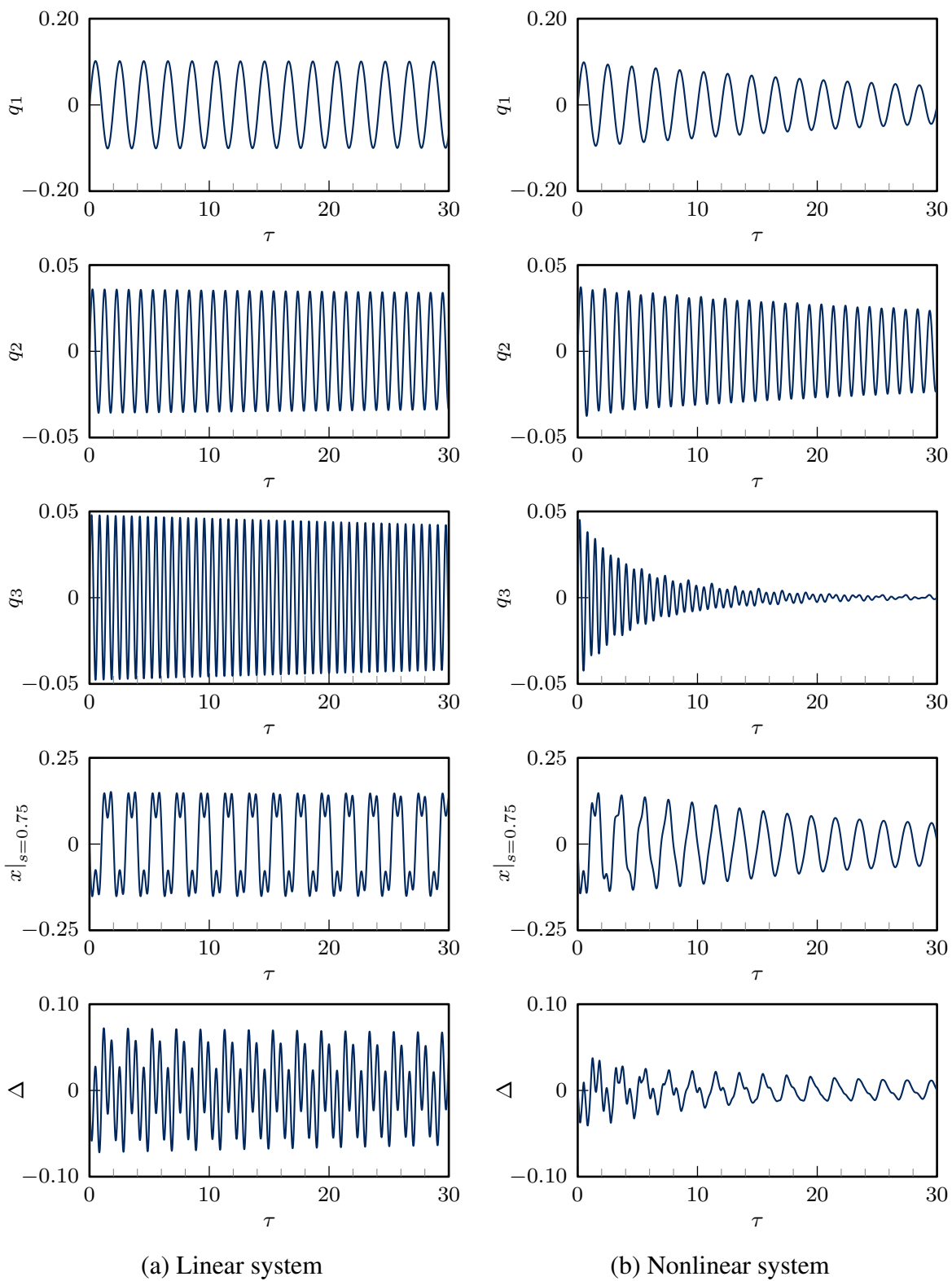
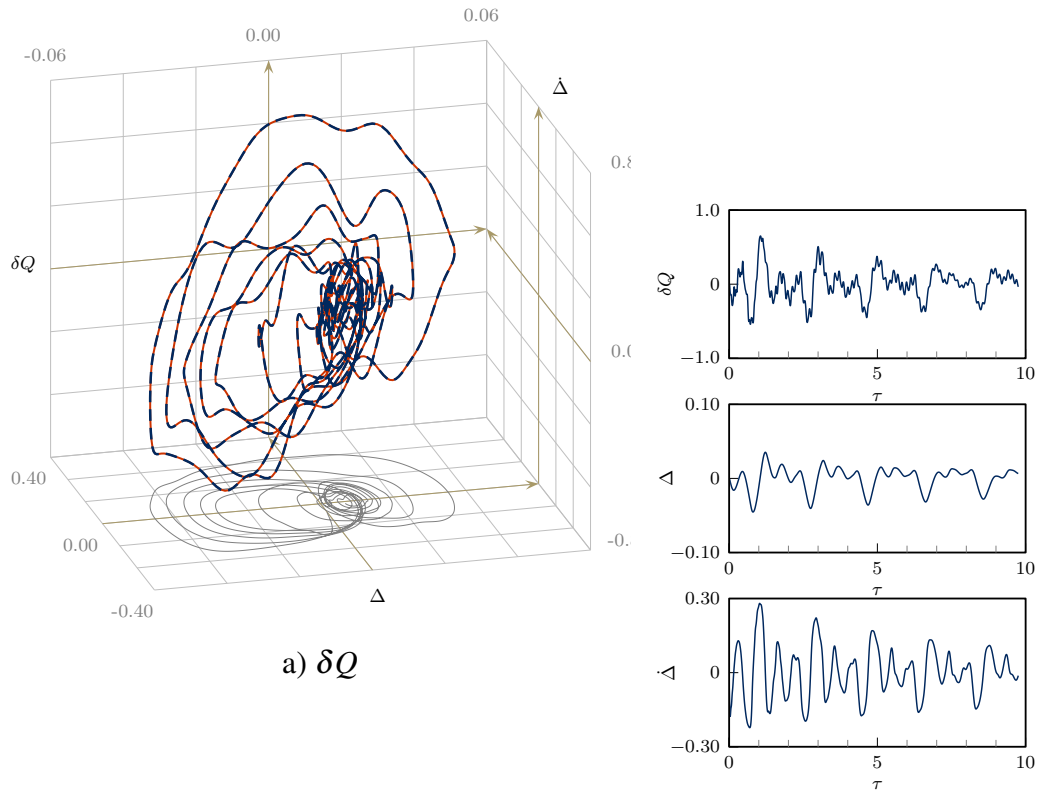
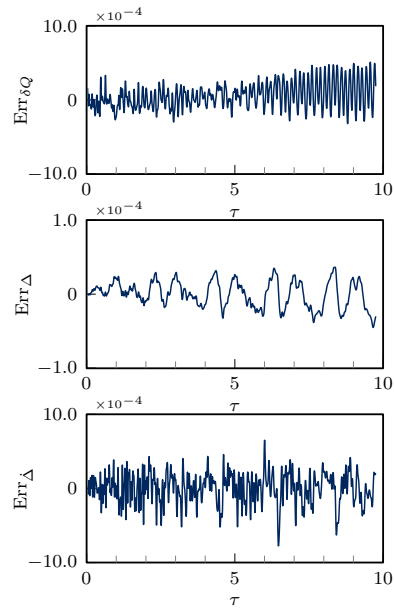


Figure 5.7. System response; direct numerical simulation



a) δQ

b) ML system



c) Error

Figure 5.8. Cubic nonlinearities and hysteretic damping

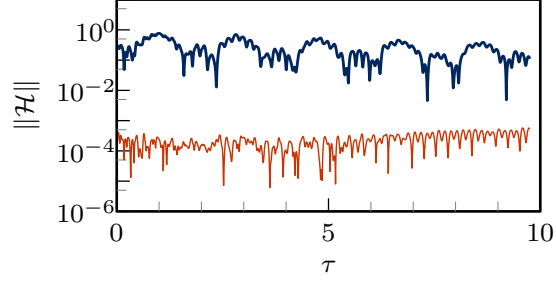


Figure 5.9. Instantaneous amplitude of the Hilbert Transform; $\|\mathcal{H}[\delta Q](t)\|$: Blue, $\|\mathcal{H}[\text{Err}_{\delta Q}](t)\|$: Red.

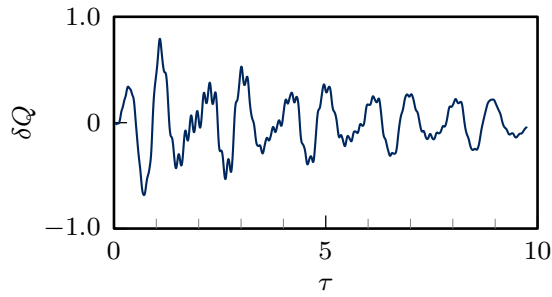
system. Note that when the system was trained the response was dominated by the first mode with minor contributions from the second and third modes. In contrast, for the initial conditions given in Eq. (5.37) the contribution from the first mode is minimal while the second mode dominates the response. Nevertheless, as observed in Figure 5.10b the error that exists in the net force δQ applied at the boundary of $\mathcal{C}_2$ is several orders of magnitude smaller than δQ .

Likewise, when the system is subject to an external excitation, the ML system is capable of accurately reproducing the response of the system even when trained from the unforced response. When a harmonic force $f(t)$ is applied to the system at $s = 0.10$ with zero initial conditions and

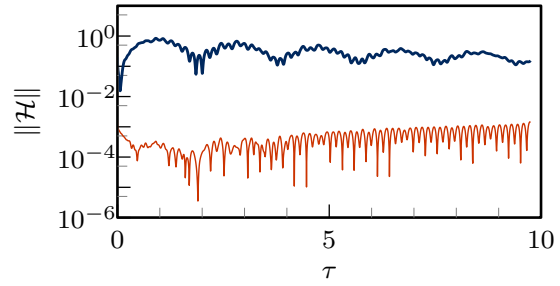
$$f(t) = F_0 \sin(\omega t), \quad F_0 = 0.20, \quad \omega = 12 \text{ rad/s}, \quad (5.38)$$

the force δQ and error are shown in Figure 5.11. For reference, the lowest modal frequency of the structure is $\Omega_1 = 3.11 \text{ rad/s}$. As before, the error between the ML system and the direct numerical simulations are several orders of magnitude smaller than the response. In addition, in Figure 5.12 the force across the isolated region is shown once the system reaches steady-state.

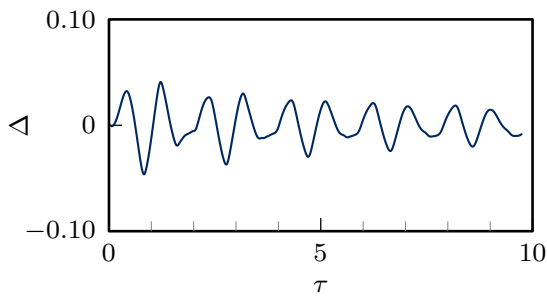
In both examples illustrated above, the ML system is responding only to the displacement and velocity across the interface. While the specific dynamics of the system arise from the forcing and initial conditions, the predictions of the ML system only depend on the response at



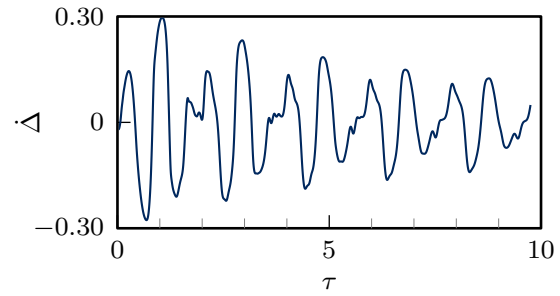
a) δQ



b) Hilbert transform; $\|\mathcal{H}[\delta Q](t)\|$: Blue,
 $\|\mathcal{H}[\text{Err}_{\delta Q}](t)\|$: Red

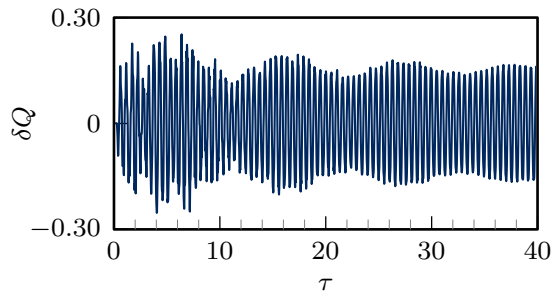


c) Δ

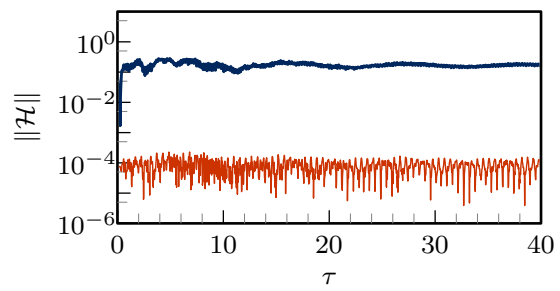


d) $\dot{\Delta}$

Figure 5.10. Cubic nonlinearities and hysteretic damping, alternative initial conditions



a) δQ



b) Hilbert transform; $\|\mathcal{H}[\delta Q](t)\|$: Blue,
 $\|\mathcal{H}[\text{Err}_{\delta Q}](t)\|$: Red

Figure 5.11. Cubic nonlinearities and hysteretic damping, forced response

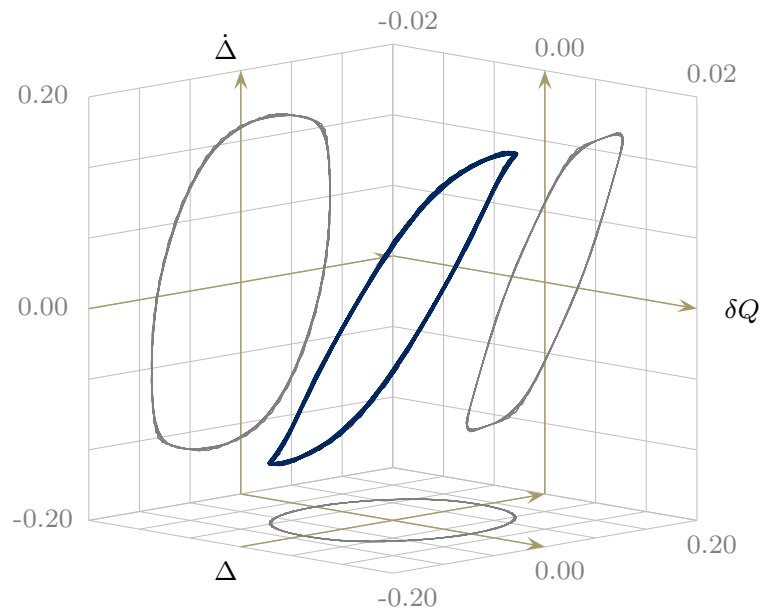


Figure 5.12. Cubic nonlinearities and hysteretic damping, forced response; steady-state response

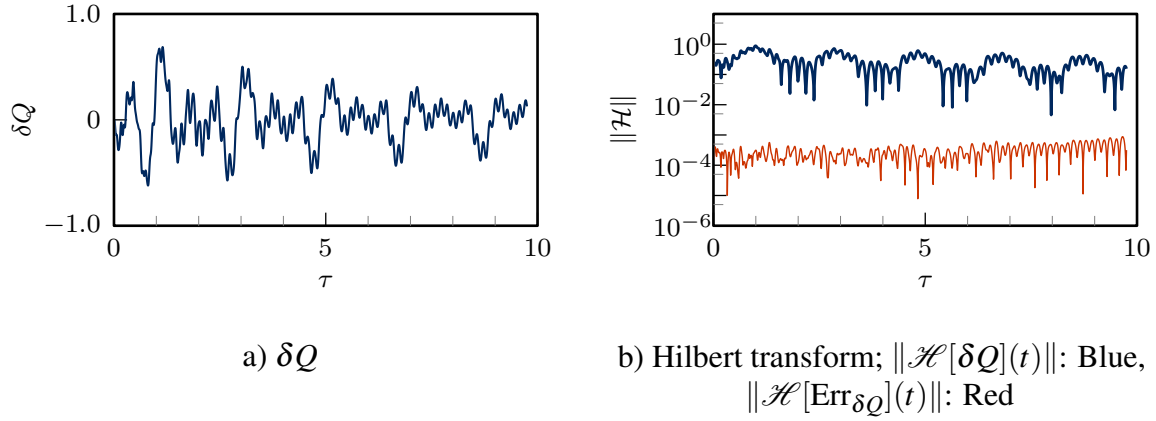
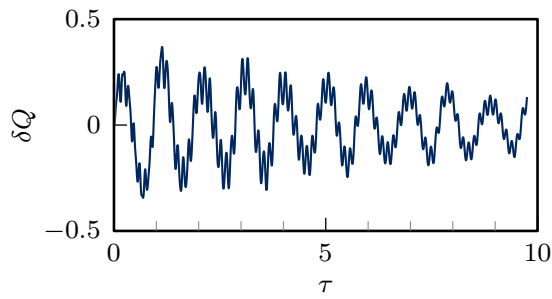


Figure 5.13. Cubic nonlinearities and hysteretic damping, altered $\mathcal{C}_1$

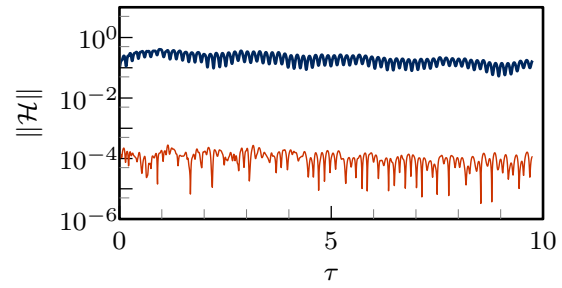
the boundary of $\mathcal{C}_2$, regardless of what external effects give rise to those dynamics. Thus it can be used as a robust, data-driven surrogate for the direct numerical simulation of the nonlinearities internal to $\mathcal{C}_2$.

In the above examples the trained ML system has been applied to varying external environments (initial conditions and external forcing) but for an identical structural system. As noted above, the trained network only responds to the dynamics at the boundary of $\mathcal{C}_2$ without regard to the dynamics of the region $\mathcal{C}_1$. To emphasize this, the ML system when trained with the original structural system is utilized as the linear system in $\mathcal{C}_1$ varies, while the system within $\mathcal{C}_2$ remains constant. For example, if the stiffness of the region within the interval $0 \leq s \leq 0.1$ is reduced by 20% the force δQ is characterized in Figure 5.13.

Finally, in this example the underlying linear system is identical through the structure. Therefore the nonlinear region $\mathcal{C}_2$ can be relocated within the structure and the original ML system can again be used to accurately describe the response of the system. As shown in Figure 5.14, if the nonlinear region is relocated from $s \in [0.25, 0.35]$ to $s \in [0.45, 0.55]$ the ML system is again able to determine δQ . Note that in this configuration the nonlinearities primarily couple with the second mode, since $\mathcal{C}_2$ is centered around a point of zero strain for the odd modes in this system. As observed in Figure 5.15, first and third modes exhibit little decay, as compared to the response when the interface was located in $s \in [0.25, 0.35]$ (c.f. Figure 5.7).

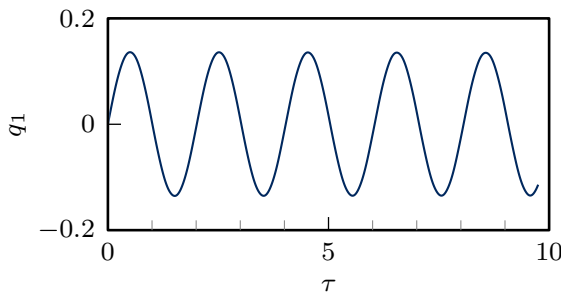


a) δQ

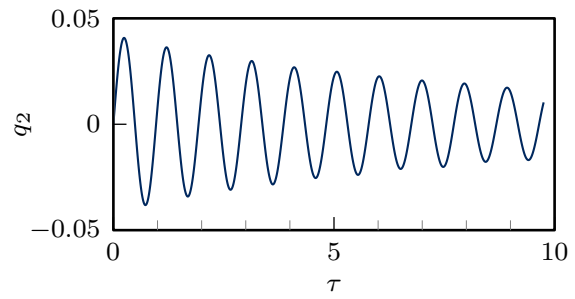


b) Hilbert transform; $\|\mathcal{H}[\delta Q](t)\|$: Blue,
 $\|\mathcal{H}[\text{Err}_{\delta Q}](t)\|$: Red

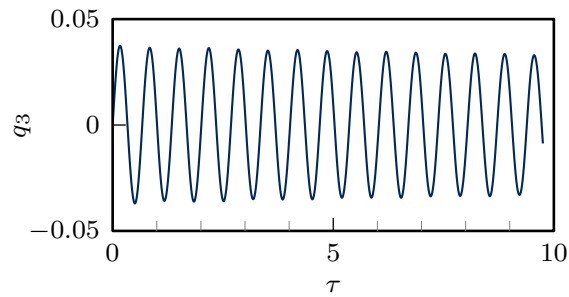
Figure 5.14. Cubic nonlinearities and hysteretic damping, $\mathcal{C}_2$ relocated to $s \in [0.45, 0.55]$



a) q_1



b) q_2



c) q_3

Figure 5.15. Cubic nonlinearities and hysteretic damping, $\mathcal{C}_2$ relocated to $s \in [0.45, 0.55]$

5.2 Conclusions

This work describes the introduction of a physics-based machine learning algorithm to reduced-order modeling of structural systems. In particular, when a structural system contains a localized region with nonlinearities, the system can be decomposed into an ideal system that describes the underlying linear system of the entire structure, including the region containing the nonlinearities, together with a second system defined over the region that spans the nonlinearities. In the structure exterior to the nonlinear region the response of the ideal system is identical to that of the original system, and only differs within the localized region. The ideal system is driven at the boundary of the nonlinear region and this forcing, referred to as the deviatoric force, represents the effect of the nonlinearities on the structure. This deviatoric force is then obtained from the second system, which is driven through the nonlinearities.

In an exact formulation this second system is solved to determine the deviatoric force acting on the ideal system, therefore describing the structural response exterior to the isolated region. This current work applies a physics-based multilayer perceptron to predict the deviatoric forces based on the response at the boundary of the localized region, therefore bypassing the direct numerical simulation of the nonlinear region. This formulation is applied to the dynamics of a rod with an internal nonlinear region spanning approximately 10% of the total length. In this work, the nonlinear region includes cubic stiffness and damping, as well as hysteretic nonlinearities that incorporate internal state variables. The ML system was trained for a single realization with randomly chosen initial conditions, and then applied to a variety of test cases. In particular the system with the trained network was able to accurately predict the structural response of the overall system when the initial conditions varied as well as when the system was subject to external time-dependent forcing. Moreover, when the isolated region was relocated within the structure the originally trained network continued to provide an accurate description of the deviatoric force required to predict the overall structural system. Therefore the network was faithfully able to describe the internal dynamics of the localized region arising from the

nonlinearities. In the current work both the ideal linear system in terms of $(\mathbf{M}, \mathbf{C}, \mathbf{K})$ must be known and the response $\mathbf{x}_\beta$ on the boundary must be measurable. Ongoing research is focused on the development of reduced-order models for the structural system in the known domain $\mathcal{C}_1$ and the development of ML systems that reduce their reliance on the observable response, both which will necessitate a reconstruction of the boundary response needed to predict the deviatoric force.

The approach described in this work could serve as the basis for the development of computationally efficient reduced-order models that provide an accurate description of the dynamics of structural systems and enable downstream tasks of relevance. Moreover, the integration of the data-driven multilayer perceptron can allow the formulation to incorporate unknown nonlinearities and possibly detect when the isolated nonlinearities begin to evolve, useful for digital twins and structural health monitoring applications. Such applications will require not only the development of order-reduction methods for the ideal linear systems, but also the optimization of the network architecture for the ML representation. Ongoing research is considering these directions as well as incorporating uncertainty quantification in the ML model.

5.3 Preview of Chapter 6

Chapter 5 presented a framework for incorporating measured data at the boundaries of isolated regions to learn corrections from a deviatoric response in a deployed structure. The work presented in Chapter 5 incorporated knowledge from an idealized model that is assumed to be known and, generally, "small"-dimensional problems were explored. Chapter 6 expands the work to consider high-dimensional problems by introducing a strategy to incorporate Lagrangian operator inference to learn a linear reduced order model. The structure-preserving ML workflow is then used to learn the nonlinear "corrections" to the linear model. This extension of the work demonstrates a way to obtain fast-running digital twin models from high-dimensional, higher fidelity models.

5.4 Remarks

Portions of this dissertation have been published or are being considered for publication. Chapter 5 is composed of the following first authored publications:

David A. Najera-Flores, D. Dane Quinn, Anthony Garland, Konstantinos Vlachas, Eleni Chatzi, and Michael D. Todd. A structure-preserving machine learning framework for accurate prediction of structural dynamics for systems with isolated nonlinearities. *Mechanical Systems and Signal Processing*, 213:111340, 2024

Chapter 6

Nonintrusive Structure-Preserving Nonlinear Model Order Reduction

Chapter 5 presented a strategy for leveraging existing knowledge about a physical system to reduce the computational burden on a learned model. Another strategy for reducing this computational burden and increasing efficiency is through model order reduction. Reduced-order models (ROMs) of nonlinear mechanical models play a key role in a variety of tasks ranging from control of soft robotics to design optimization of mechanical structures to state assessment for structural health monitoring. For structural and mechanical engineering applications, deriving a ROM that preserves the underlying Lagrangian structure is of importance because the Lagrangian structure is intimately connected to physically interpretable quantities such as momentum and energy, or in case of fluid systems, vorticity. Structure-preserving model reduction of mechanical systems was introduced in [133] where the authors showed that performing a Galerkin projection on the Euler-Lagrange equation leads to a Lagrangian system in the reduced-dimensional coordinates, thus preserving the Lagrangian structure. The work in [34] presented a computationally efficient, structure-preserving model reduction method for parametric Lagrangian systems with higher-order nonlinearities. Both of these structure-preserving model reduction approaches are intrusive in that they require access to full-order model (FOM) operators in order to derive nonlinear ROMs via the intrusive projection. This type of information, however, is typically unavailable when working with proprietary software, complicated legacy code, or experimental

data. Thus, nonintrusive methods that can learn reduced-order models directly from simulated or experimental data have become increasingly popular.

The operator inference framework [194] is a promising data-driven approach for learning low-dimensional models of FOMs with linear or low-order polynomial nonlinear terms. Using lifting transformations, this model reduction method has been extended to a broader class of FOMs with nonpolynomial nonlinear terms in [196, 197, 239, 122]. This nonintrusive approach has also been extended to the gray-box setting in [15] where the authors use knowledge about the nonpolynomial terms in analytic form to learn nonlinear ROMs from data. Recently, a variety of papers have embedded problem-specific structure in the operator inference framework to develop structure-preserving operator inference methods for Hamiltonian systems [226, 81] and Lagrangian mechanical systems [223, 68]. Similar to [15], these structure-preserving operator inference approaches assume prior knowledge about the nonlinear terms in analytic form. Such information is typically unavailable when the underlying physics is not well known or otherwise difficult to model.

As an alternative to learning interpretable polynomial dynamical systems, deep learning-based ROMs [94, 214, 74, 135, 70, 71] have been proposed for nonintrusive model reduction of nonlinear PDEs where these methods exploit deep neural networks to learn both the nonlinear trial manifold and the reduced dynamics. However, these methods are not designed to preserve the underlying geometric structure and therefore do not provide accurate predictions outside the training data regime. Discovering nonlinear dynamical systems from data was first considered in [219, 20] where the authors employed symbolic regression methods to learn the governing equations. Dictionary-based approaches based on sparse identification of nonlinear dynamical systems (SINDy) [28] have also been developed for structure-preserving sparse identification of nonlinear dynamics in [115, 48, 138]. In another research direction, the machine learning community has developed a wide variety of structure-preserving machine learning methods by endowing black-box neural networks with physics-motivated inductive biases. These structure-preserving machine learning (SpML) methods were first developed in the

context of conservative dynamical systems by preserving the geometric structure related to the underlying Hamiltonian [80, 46, 112, 43], Lagrangian [155, 51, 84, 6, 184], and conservation laws [136, 255]. Building on the work in this direction, these methods have also been generalized to nonconservative systems by preserving the metriplectic structure [137, 288]. However, a majority of these approaches are only concerned with learning nonlinear dynamical systems when the data is coming from very low-dimensional systems, e.g. 5 – 10 dimensions. Therefore, these SpML methods are ill-suited for higher-dimensional systems.

The main goal of this work is to develop a structure-preserving nonintrusive model reduction method that can learn ROMs strictly from observed data without assuming additional knowledge about the form of nonlinearity.

The chapter is structured as follows. Section 6.1 first outlines the Lagrangian mechanics formulation of nonlinear mechanical models and then presents a brief review of the intrusive structure-preserving model reduction of these mechanical FOMs. Section 6.2 presents the proposed ML-enhanced operator inference method for learning nonlinear Lagrangian ROMs from high-dimensional data. In Section 6.3, we apply the proposed method to three datasets with increasing complexity: simulated data from a conservative rod model, simulated data from a two-dimensional nonlinear membrane model with internal damping, and experimental data from a jointed structure. Section 6.4 provides concluding remarks and suggests future research directions motivated by this work.

6.1 Background

In Section 6.1.1 we introduce the Lagrangian formulation of the nonlinear mechanical FOMs considered in this work. In Section 6.1.2 we review the construction of projection-based intrusive Lagrangian ROMs.

6.1.1 Lagrangian mechanics

Consider a high-dimensional nonlinear mechanical system with a configuration manifold $Q = \mathbb{R}^n$ where n denotes the degrees of freedom in the FOM. We consider a Lagrangian viewpoint on the mechanical system where the dynamics are described by the scalar Lagrangian function

$$L(\mathbf{q}, \dot{\mathbf{q}}) = T(\dot{\mathbf{q}}) - U(\mathbf{q}), \quad (6.1)$$

where $\mathbf{q}(t)$ is the set of generalized coordinates describing the system state, $T(\dot{\mathbf{q}})$ is the scalar kinetic energy function, and $U(\mathbf{q})$ is the scalar potential energy function. The governing equations for Lagrangian mechanical systems with nonconservative forcing $\mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, t)$ can be derived via the Lagrange-d'Alembert principle, see [8] for more details. This principle seeks $\mathbf{q}(t)$ satisfying $\delta \int_{t_0}^{t_K} L(\mathbf{q}, \dot{\mathbf{q}}) dt + \int_{t_0}^{t_K} \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, t) \cdot \delta \mathbf{q} dt = 0$, where δ represents variations vanishing at initial time t_0 and final time t_K . Using integration by parts and setting the variations at the endpoints to zero yields the forced Euler-Lagrange equations

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{\mathbf{q}}} \right) - \frac{\partial L}{\partial \mathbf{q}} = \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, t). \quad (6.2)$$

For this work, we focus on FOM Lagrangians of the form

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^\top \mathbf{M} \dot{\mathbf{q}} - \frac{1}{2} \mathbf{q}^\top \mathbf{K} \mathbf{q} - U_{\text{nl}}(\mathbf{q}), \quad (6.3)$$

where $\mathbf{M} \in \mathbb{R}^{n \times n}$ denotes the symmetric positive-definite mass matrix, $\mathbf{K} \in \mathbb{R}^{n \times n}$ denotes the linear stiffness matrix, and U_{nl} is the nonlinear component of the potential energy function. We note that the assumption about the quadratic form of the system's kinetic energy in (6.3) is quite general in mechanical and structural engineering applications. In high-dimensional mechanical models obtained via semi-discretization, the linear stiffness matrix $\mathbf{K}$ is typically a symmetric positive-definite matrix.

The dissipative effects in mechanical and structural engineering applications often arise from internal damping. The nonconservative forcing $\mathbf{f}$ in (6.2) can be decomposed into linear and nonlinear components, i.e.,

$$\mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, t) = -\mathbf{C}\dot{\mathbf{q}} - \frac{\partial \mathcal{F}_{\text{nl}}(\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}}, \quad (6.4)$$

where $\mathbf{C} \in \mathbb{R}^{n \times n}$ is the linear damping matrix and $\mathcal{F}_{\text{nl}}(\dot{\mathbf{q}})$ is the scalar dissipation function for modeling nonlinear damping behavior [2]. The linear dissipation behavior in mechanical and structural engineering applications is modeled using Rayleigh damping where the linear damping matrix is proportional to the mass and stiffness matrix. As a result, the damping matrix $\mathbf{C}$ also possesses a symmetric positive-definite structure. Substituting expressions for the system Lagrangian (6.3) and the nonconservative forcing (6.4) into the forced Euler-Lagrange equations (6.2) yields the governing equations for the Lagrangian FOM

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \frac{\partial \mathcal{F}_{\text{nl}}(\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} + \mathbf{K}\mathbf{q} + \frac{\partial U_{\text{nl}}(\mathbf{q})}{\partial \mathbf{q}} = \mathbf{0}. \quad (6.5)$$

6.1.2 Structure-preserving intrusive model reduction for mechanical systems

Galerkin projection-based methods approximate the FOM configuration space $\mathbf{Q} = \mathbb{R}^n$ by a reduced-order configuration space $\mathbf{Q}_r = \mathbb{R}^r$. Using an orthonormal basis matrix $\mathbf{V}_r \in \mathbb{R}^{n \times r}$, the high-dimensional state is approximated via $\mathbf{q} \approx \mathbf{V}_r \hat{\mathbf{q}}$ where the intrusive reduced state is denoted with $\hat{\mathbf{q}} \in \mathbb{R}^r$. We follow [34] to derive governing equations for the intrusive Lagrangian ROM. The intrusive reduced Lagrangian $\hat{L}$ is defined as

$$\hat{L}(\hat{\mathbf{q}}, \dot{\hat{\mathbf{q}}}) := L(\mathbf{V}_r \hat{\mathbf{q}}, \mathbf{V}_r \dot{\hat{\mathbf{q}}}) = T(\mathbf{V}_r \dot{\hat{\mathbf{q}}}) - U(\mathbf{V}_r \hat{\mathbf{q}}). \quad (6.6)$$

Applying the Lagrange-d'Alembert principle in the reduced setting, the resulting forced Euler-Lagrange equations in r dimensions are

$$\frac{\partial \widehat{L}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}})}{\partial \widehat{\mathbf{q}}} - \frac{d}{dt} \left(\frac{\partial \widehat{L}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}})}{\partial \dot{\widehat{\mathbf{q}}}} \right) + \widehat{\mathbf{f}}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}}, t) = \mathbf{0}, \quad (6.7)$$

where $\widehat{\mathbf{f}}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}}, t) := \mathbf{V}_r^\top \mathbf{f}(\mathbf{V}_r \widehat{\mathbf{q}}, \mathbf{V}_r \dot{\widehat{\mathbf{q}}}, t)$ is the intrusive reduced nonconservative forcing.

For FOM Lagrangians of the form (6.3), the intrusive reduced Lagrangian $\widehat{L}$ is

$$\widehat{L}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}}) = \frac{1}{2} \dot{\widehat{\mathbf{q}}}^\top \left(\mathbf{V}_r^\top \mathbf{M} \mathbf{V}_r \right) \dot{\widehat{\mathbf{q}}} - \frac{1}{2} \widehat{\mathbf{q}}^\top \left(\mathbf{V}_r^\top \mathbf{K} \mathbf{V}_r \right) \widehat{\mathbf{q}} - \widehat{U}_{\text{nl}}(\mathbf{V}_r \widehat{\mathbf{q}}), \quad (6.8)$$

where $\widehat{U}_{\text{nl}} := U_{\text{nl}}(\mathbf{V}_r \widehat{\mathbf{q}})$ is the intrusive reduced nonlinear potential energy function. The intrusive reduced nonconservative forcing $\widehat{\mathbf{f}}$ corresponding to nonconservative forcing of the form (6.4) is

$$\widehat{\mathbf{f}}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}}, t) = - \left(\mathbf{V}_r^\top \mathbf{C} \mathbf{V}_r \right) \dot{\widehat{\mathbf{q}}} - \frac{\partial \widehat{\mathcal{F}}_{\text{nl}}(\dot{\widehat{\mathbf{q}}})}{\partial \dot{\widehat{\mathbf{q}}}}, \quad (6.9)$$

where $\widehat{\mathcal{F}}_{\text{nl}}(\dot{\widehat{\mathbf{q}}}) := \mathcal{F}_{\text{nl}}(\mathbf{V}_r \dot{\widehat{\mathbf{q}}})$ is the intrusive reduced dissipation function. Substituting expressions for the intrusive reduced Lagrangian (6.8) and the intrusive reduced nonconservative forcing (6.9) into the reduced forced Euler-Lagrange equation leads to

$$\widehat{\mathbf{M}} \ddot{\widehat{\mathbf{q}}} + \widehat{\mathbf{C}} \dot{\widehat{\mathbf{q}}} + \frac{\partial \widehat{\mathcal{F}}_{\text{nl}}(\dot{\widehat{\mathbf{q}}})}{\partial \dot{\widehat{\mathbf{q}}}} + \widehat{\mathbf{K}} \widehat{\mathbf{q}} + \frac{\partial \widehat{U}_{\text{nl}}(\widehat{\mathbf{q}})}{\partial \widehat{\mathbf{q}}} = \mathbf{0}, \quad (6.10)$$

where $\widehat{\mathbf{M}} := \mathbf{V}_r^\top \mathbf{M} \mathbf{V}_r$, $\widehat{\mathbf{C}} := \mathbf{V}_r^\top \mathbf{C} \mathbf{V}_r$, and $\widehat{\mathbf{K}} := \mathbf{V}_r^\top \mathbf{K} \mathbf{V}_r$ are the intrusive reduced operators. The authors in [133, 34] showed that the intrusive Lagrangian ROM (6.10) respects the symmetric positive-definite property of the system matrices and therefore, preserves the underlying Lagrangian structure. However, deriving an intrusive Lagrangian ROM requires access to FOM operators, which is often not possible when working with proprietary software or a complicated community code.

6.2 ML-enhanced structure-preserving operator inference method

Section 6.2.1 describes the projection of the high-dimensional data onto a low-dimensional subspace via proper orthogonal decomposition (POD). Section 6.2.2 formulates the structure-preserving learning problem and proposes a two-step strategy to learn a structure-preserving nonlinear ROM purely from data. Section 6.2.3 employs the Lagrangian operator inference method to infer linear reduced operators. Section 6.2.4 presents a SpML method that augments the linear Lagrangian ROM inferred in Section 6.2.3 with nonlinear terms. Section 6.2.5 summarizes the proposed two-step learning approach in Algorithm 3 and provides details about the offline training stage.

6.2.1 Data projection via proper orthogonal decomposition

We define the high-dimensional snapshot data matrix

$$\mathbf{Q} = [\mathbf{q}_1, \dots, \mathbf{q}_K] \in \mathbb{R}^{n \times K}, \quad (6.11)$$

where $\mathbf{q}_1, \dots, \mathbf{q}_K$ are the discrete states of the potentially high-dimensional system (6.5) at time $t_1, \dots, t_K$. Since it is computationally infeasible to directly learn nonlinear high-dimensional models of the form (6.5), we use the POD basis for data projection, which enables a low-dimensional representation of the high-dimensional data.

We follow [101] to build the basis matrix $\mathbf{V}_r$ from the FOM snapshot data matrix $\mathbf{Q} = [\mathbf{q}_1, \dots, \mathbf{q}_K] \in \mathbb{R}^{n \times K}$ via singular value decomposition. We compute the singular value decomposition of the snapshot data

$$\mathbf{Q} = \mathbf{V} \mathbf{\Sigma} \mathbf{W}^\top, \quad (6.12)$$

where $\mathbf{V} \in \mathbb{R}^{n \times n}$, $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$, and $\mathbf{W} \in \mathbb{R}^{K \times n}$. The columns of matrices $\mathbf{V}$ and $\mathbf{W}$ are the left and right singular vectors of $\mathbf{Q}$, respectively. We assume that the singular values $\xi_1 \geq \xi_2 \geq \dots$

in $\mathbf{\Xi}$ are in descending order. The POD basis of dimension r is denoted by $\mathbf{V}_r \in \mathbb{R}^{n \times r}$ and defined by the leading r columns of $\mathbf{V}$ that correspond to the r largest singular values. The state approximation is $\mathbf{q} \approx \mathbf{V}_r \hat{\mathbf{q}}$, where we denote the nonintrusive reduced state with $\hat{\mathbf{q}} \in \mathbb{R}^r$. Using this POD basis, we obtain reduced snapshot data via the projections onto $\mathbf{V}_r$ as

$$\hat{\mathbf{Q}} = \mathbf{V}_r^\top \mathbf{Q} = [\hat{\mathbf{q}}_1, \dots, \hat{\mathbf{q}}_K] \in \mathbb{R}^{r \times K}, \quad (6.13)$$

where the reduced snapshot $\hat{\mathbf{q}}_k$ at time t_k is $\hat{\mathbf{q}}_k = \mathbf{V}_r^\top \mathbf{q}_k$.

To learn a nonlinear ROM nonintrusively from data, we also require time-derivative data at the reduced level. We compute $\dot{\hat{\mathbf{q}}}$ and $\ddot{\hat{\mathbf{q}}}$ from the reduced trajectory data $\hat{\mathbf{q}}$ using the eighth-order central finite difference scheme

$$\dot{\hat{\mathbf{q}}}_k \approx \frac{4(\hat{\mathbf{q}}_{k+1} - \hat{\mathbf{q}}_{k-1})}{5\Delta t} - \frac{(\hat{\mathbf{q}}_{k+2} - \hat{\mathbf{q}}_{k-2})}{5\Delta t} + \frac{4(\hat{\mathbf{q}}_{k+3} - \hat{\mathbf{q}}_{k-3})}{105\Delta t} - \frac{(\hat{\mathbf{q}}_{k+4} - \hat{\mathbf{q}}_{k-4})}{280\Delta t}, \quad (6.14)$$

$$\ddot{\hat{\mathbf{q}}}_k \approx -\frac{205\hat{\mathbf{q}}_k}{72\Delta t^2} + \frac{8(\hat{\mathbf{q}}_{k+1} + \hat{\mathbf{q}}_{k-1})}{5\Delta t^2} - \frac{(\hat{\mathbf{q}}_{k+2} + \hat{\mathbf{q}}_{k-2})}{5\Delta t^2} + \frac{8(\hat{\mathbf{q}}_{k+3} + \hat{\mathbf{q}}_{k-3})}{315\Delta t^2} - \frac{(\hat{\mathbf{q}}_{k+4} + \hat{\mathbf{q}}_{k-4})}{560\Delta t^2}. \quad (6.15)$$

Using these time-derivative approximations at the reduced level, we build the snapshot matrices of the reduced first-order and second-order time-derivative data

$$\dot{\hat{\mathbf{Q}}} = [\dot{\hat{\mathbf{q}}}_1, \dots, \dot{\hat{\mathbf{q}}}_K] \in \mathbb{R}^{r \times K}, \quad \ddot{\hat{\mathbf{Q}}} = [\ddot{\hat{\mathbf{q}}}_1, \dots, \ddot{\hat{\mathbf{q}}}_K] \in \mathbb{R}^{r \times K}. \quad (6.16)$$

6.2.2 Problem formulation

The main aim of the proposed ML-enhanced structure-preserving operator inference framework is to learn a predictive nonlinear ROM from the reduced trajectory and time-derivative data while also ensuring that the learned ROM (i) is a Lagrangian system; (ii) respects the symmetric positive-definite property of system matrices; and (iii) captures the conservative and dissipative behavior accurately.

Towards this goal, we postulate the nonlinear ROM form in terms of its Lagrangian

ingredients

$$\widehat{\mathbf{M}}\ddot{\mathbf{q}} + \widehat{\mathbf{C}}\dot{\mathbf{q}} + \frac{\partial \widehat{\mathcal{F}}_{\text{nl}}(\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} + \widehat{\mathbf{K}}\mathbf{q} + \frac{\partial \widehat{U}_{\text{nl}}(\mathbf{q})}{\partial \mathbf{q}} = \mathbf{0}, \quad (6.17)$$

where $\widehat{\mathbf{M}} = \widehat{\mathbf{M}}^\top \succ \mathbf{0}$ is the nonintrusive reduced mass matrix, $\widehat{\mathbf{C}} = \widehat{\mathbf{C}}^\top \succ \mathbf{0}$ is the nonintrusive reduced linear damping matrix, $\widehat{\mathbf{K}} = \widehat{\mathbf{K}}^\top \succ \mathbf{0}$ is the nonintrusive reduced linear stiffness matrix, $\widehat{\mathcal{F}}_{\text{nl}}$ is the reduced nonlinear dissipation function, and $\widehat{U}_{\text{nl}}$ is the reduced nonlinear potential energy function. Given the reduced snapshot data $\widehat{\mathbf{Q}}$ and the reduced time-derivative data $\dot{\widehat{\mathbf{Q}}}$ and $\ddot{\widehat{\mathbf{Q}}}$, the goal is to learn the reduced Lagrangian ingredients by minimizing

$$\left\| \widehat{\mathbf{M}}\ddot{\widehat{\mathbf{Q}}} + \widehat{\mathbf{C}}\dot{\widehat{\mathbf{Q}}} + \frac{\partial \widehat{\mathcal{F}}_{\text{nl}}(\dot{\widehat{\mathbf{Q}}})}{\partial \dot{\widehat{\mathbf{q}}}} + \widehat{\mathbf{K}}\widehat{\mathbf{Q}} + \frac{\partial \widehat{U}_{\text{nl}}(\widehat{\mathbf{Q}})}{\partial \widehat{\mathbf{q}}} \right\|_F. \quad (6.18)$$

However, directly training a black-box NN to learn the nonlinear dynamics at the reduced level can be challenging from an optimization perspective, see [130] for more details. Moreover, such black-box approaches typically fail to generalize outside the training data regime, as there is no guarantee that the NN model has learned the underlying geometric structure.

We propose a two-step strategy for learning the reduced operators where we first learn linear components of (6.17) via a constrained linear least-squares problem and then learn the nonlinear reduced operators via a structure-preserving machine learning method. In contrast to the intrusive model reduction approach presented in Section 6.1.2, the proposed approach learns the nonlinear ROM (6.17) purely from the data. For learning a nonlinear reduced Lagrangian ROM of the form (6.17), we postulate a nonlinear reduced Lagrangian $\widehat{L}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}})$

$$\widehat{L}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}}) = \widehat{L}_{\text{opinf}}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}}) + \widehat{T}_{\text{NN}}(\dot{\widehat{\mathbf{q}}}) - \widehat{U}_{\text{NN}}(\widehat{\mathbf{q}}) = \frac{1}{2}\dot{\widehat{\mathbf{q}}}^\top (\mathbf{I}_r + \widehat{\mathbf{M}}_{\text{NN}}) \dot{\widehat{\mathbf{q}}} - \frac{1}{2}\widehat{\mathbf{q}}^\top \widehat{\mathbf{K}}\widehat{\mathbf{q}} - \widehat{U}_{\text{NN}}(\widehat{\mathbf{q}}), \quad (6.19)$$

where $\widehat{L}_{\text{opinf}}(\widehat{\mathbf{q}}, \dot{\widehat{\mathbf{q}}})$ is the linear L-OpInf ROM and $\widehat{T}_{\text{NN}}$ and $\widehat{U}_{\text{NN}}$ are the parametrizations of the reduced kinetic energy and nonlinear components of the reduced potential energy function,

respectively. Similarly, we postulate a nonlinear reduced dissipation function of the form

$$\widehat{\mathcal{F}}(\dot{\mathbf{q}}) = \widehat{\mathcal{F}}_{\text{opinf}}(\dot{\mathbf{q}}) + \widehat{\mathcal{F}}_{\text{NN}}(\dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^\top \widehat{\mathbf{C}} \dot{\mathbf{q}} + \widehat{\mathcal{F}}_{\text{NN}}(\dot{\mathbf{q}}). \quad (6.20)$$

where $\widehat{\mathcal{F}}_{\text{NN}}$ is the parametrization of the nonlinear components of the reduced dissipation energy function.

The first step in our two-step learning strategy learns the linear reduced stiffness matrix $\widehat{\mathbf{K}}$ in (6.19) and the linear reduced damping matrix $\widehat{\mathbf{C}}$ in (6.20). The second step employs structure-preserving neural networks to learn the reduced mass matrix $\widehat{\mathbf{M}}_{\text{NN}}$ and nonlinear components of the reduced potential energy function $\widehat{U}_{\text{NN}}$ in (6.19) and nonlinear components of the reduced dissipation energy function $\widehat{\mathcal{F}}_{\text{NN}}$ in (6.19). Since we learn the reduced potential energy function and the reduced dissipation energy function in two steps, we decompose the reduced mass matrix $\widehat{\mathbf{M}}$ in (6.17) as $(\mathbf{I}_r + \widehat{\mathbf{M}}_{\text{NN}})$, where we learn the reduced mass matrix component $\widehat{\mathbf{M}}_{\text{NN}}$ is learned during the training of structure-preserving neural networks in the second step. This specific decomposition of the reduced mass matrix $\widehat{\mathbf{M}}$ serves two purposes: it simplifies the constrained linear least-squares problem in Section 6.2.3, and it enables the learning of a correction term $\widehat{\mathbf{M}}_{\text{NN}}$ in the reduced mass matrix $\widehat{\mathbf{M}}$ that is compatible with the nonlinear components $\widehat{U}_{\text{NN}}$ and $\widehat{\mathcal{F}}_{\text{NN}}$ learned in the second learning step in Section 6.2.4.

6.2.3 Step 1: Lagrangian operator inference method for linear reduced operators

To learn linear parts of the Lagrangian ROM, we use ideas from the Lagrangian operator inference method (LOpInf) [223] which fits linear reduced operators to the reduced data in a structure-preserving way. For this portion of the learning process, we postulate a quadratic Lagrangian of the form

$$\widehat{L}_{\text{opinf}} = \frac{1}{2} \dot{\mathbf{q}}^\top \dot{\mathbf{q}} - \frac{1}{2} \dot{\mathbf{q}}^\top \widehat{\mathbf{K}} \dot{\mathbf{q}}, \quad (6.21)$$

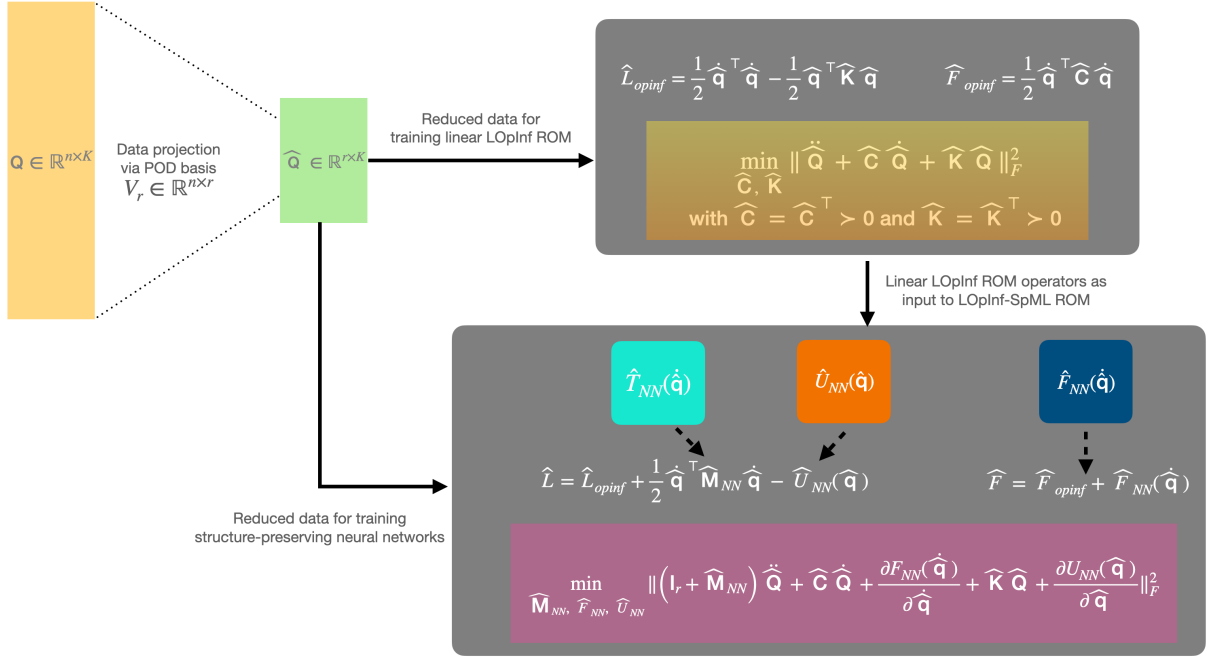


Figure 6.1. ML-enhanced structure-preserving operator inference architecture.

where $\hat{\mathbf{K}} \in \mathbb{R}^{r \times r}$ is the symmetric positive-definite reduced stiffness matrix. Similarly, we postulate a quadratic dissipation function

$$\widehat{\mathcal{F}}_{\text{opinf}} = \frac{1}{2} \dot{\hat{\mathbf{q}}}^\top \hat{\mathbf{C}} \dot{\hat{\mathbf{q}}}. \quad (6.22)$$

where $\hat{\mathbf{C}} \in \mathbb{R}^{r \times r}$ is the symmetric positive-definite reduced damping matrix.

Based on the assumed model form for the dissipation function $\widehat{\mathcal{F}}_{\text{opinf}}$ and the reduced Lagrangian $\hat{L}_{\text{opinf}}$, we derive the governing equations for the linear LOPinf ROM via the forced Euler-Lagrange equations

$$\ddot{\hat{\mathbf{q}}} + \hat{\mathbf{C}}\dot{\hat{\mathbf{q}}} + \hat{\mathbf{K}}\hat{\mathbf{q}} = \mathbf{0}. \quad (6.23)$$

We define the linear reduced operators $\hat{\mathbf{C}} \in \mathbb{R}^{r \times r}$ and $\hat{\mathbf{K}} \in \mathbb{R}^{r \times r}$ to be a solution of the constrained optimization problem

$$\min_{\hat{\mathbf{K}} = \hat{\mathbf{K}}^\top \succ \mathbf{0}, \hat{\mathbf{C}} = \hat{\mathbf{C}}^\top \succ \mathbf{0}} \left\| \ddot{\hat{\mathbf{Q}}} + \hat{\mathbf{C}}\dot{\hat{\mathbf{Q}}} + \hat{\mathbf{K}}\hat{\mathbf{Q}} \right\|_F. \quad (6.24)$$

The symmetric positive-definite constraints on the linear reduced operators ensure that the linear

LOpInf ROM (6.23) preserves the underlying Lagrangian structure. Since the snapshot data $\mathbf{Q}$ (6.11) is obtained (simulated or experimentally measured) from a nonlinear mechanical system, the linear LOpInf ROM in (6.23) may be inaccurate because it does not capture the underlying nonlinear behavior. The second step in the proposed learning approach exploits the expressive power of neural networks to augment the linear LOpInf ROM with missing nonlinear ROM operators.

6.2.4 Step 2: Structure-preserving learning for nonlinear reduced operators

In this section, we present a structure-preserving machine learning architecture to learn the nonlinear components of the reduced potential energy and the reduced dissipation function via polynomial-augmented multilayer perceptrons (MLPs). These polynomial-augmented MLPs allow us to learn neural network parametrizations of arbitrary nonlinear functions.

For learning the nonlinear components of the reduced potential energy, we define

$$\widehat{U}_{\text{NN}}(\hat{\mathbf{q}}) = \sum_{i_1, i_2, \dots, i_r}^{i_1 + i_2 + \dots + i_r = P} \alpha_{i_1, i_2, \dots, i_r} \hat{q}_1^{i_1} \hat{q}_2^{i_2} \dots \hat{q}_r^{i_r} + \sum_i^N \lambda_i \mathcal{U}^{(i)}(\hat{\mathbf{q}}), \quad (6.25)$$

where P is the degree of the polynomial term, and the function $\mathcal{U}^{(i)}$ represents an MLP. We define the NN parametrization for the reduced dissipation energy function $\widehat{\mathcal{F}}_{\text{NN}}$ in a similar fashion

$$\widehat{\mathcal{F}}_{\text{NN}}(\dot{\hat{\mathbf{q}}}) = \sum_{i_1, i_2, \dots, i_r}^{i_1 + i_2 + \dots + i_r = P} \beta_{i_1, i_2, \dots, i_r} \dot{\hat{q}}_1^{i_1} \dot{\hat{q}}_2^{i_2} \dots \dot{\hat{q}}_r^{i_r} + \sum_i^N \gamma_i \mathcal{F}^{(i)}(\dot{\hat{\mathbf{q}}}), \quad (6.26)$$

where P is the degree of the polynomial term, and the function $\mathcal{F}^{(i)}$ represents an MLP. In this work, we have used polynomial terms up to degree $P = 4$ in both $\widehat{U}_{\text{NN}}$ and $\widehat{\mathcal{F}}_{\text{NN}}$. Finally, we define the reduced kinetic energy term $\widehat{T}_{\text{NN}}$ as

$$\widehat{T}_{\text{NN}}(\dot{\hat{\mathbf{q}}}) = \frac{1}{2} \dot{\hat{\mathbf{q}}}^\top \widehat{\mathbf{M}}_{\text{NN}} \dot{\hat{\mathbf{q}}}, \quad (6.27)$$

where we constrain $\widehat{\mathbf{M}}_{\text{NN}}$ to be a symmetric positive-definite matrix, i.e., $\widehat{\mathbf{M}}_{\text{NN}} = \widehat{\mathbf{M}}_{\text{NN}}^\top \succ 0$. We define the reduced kinetic energy term $\widehat{T}_{\text{NN}}(\dot{\mathbf{q}})$ as

$$\widehat{T}_{\text{NN}}(\dot{\mathbf{q}}) = \sum_{i_1, i_2, \dots, i_r}^{i_1 + i_2 + \dots + i_r = 2} \zeta_{i_1, i_2, \dots, i_r} \dot{q}_1^{i_1} \dot{q}_2^{i_2} \dots \dot{q}_r^{i_r}, \quad (6.28)$$

noting that only up to quadratic terms are allowed and that no MLPs are used. This formulation leads to a constant reduced order mass matrix which is obtained by computing the Hessian of $\widehat{T}_{\text{NN}}(\dot{\mathbf{q}})$ as

$$\widehat{M}_{\text{NN},ij} = \frac{\partial^2 \widehat{T}_{\text{NN}}(\dot{\mathbf{q}})}{\partial \dot{q}_i \partial \dot{q}_j}, \quad (6.29)$$

where $\widehat{M}_{\text{NN},ij}$ represents the i -th row and j -th column of the $\widehat{\mathbf{M}}_{\text{NN}}$ matrix. This form of the reduced matrix ensures symmetry due to the Hessian computation.

Substituting expressions for the nonlinear reduced Lagrangian (6.19) and the nonlinear dissipation function (6.20) into the forced Euler-Lagrange equations yields the ROM model form

$$\left(\mathbf{I}_r + \widehat{\mathbf{M}}_{\text{NN}}\right) \ddot{\mathbf{q}} + \widehat{\mathbf{C}}\dot{\mathbf{q}} + \frac{\partial \widehat{\mathcal{F}}_{\text{NN}}(\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} + \widehat{\mathbf{K}}\mathbf{q} + \frac{\partial \widehat{U}(\mathbf{q})}{\partial \mathbf{q}} = \mathbf{0}. \quad (6.30)$$

To train the neural network operators in the LOpInf-SpML ROM (6.30), we consider a supervised learning setup with a loss function based on the residual of the forced Euler-Lagrange equations at the reduced level

$$\mathcal{J} = \left\| \left(\mathbf{I}_r + \widehat{\mathbf{M}}_{\text{NN}}\right) \ddot{\mathbf{Q}} + \widehat{\mathbf{C}}\dot{\mathbf{Q}} + \frac{\partial \widehat{\mathcal{F}}_{\text{NN}}(\dot{\mathbf{Q}})}{\partial \dot{\mathbf{Q}}} + \widehat{\mathbf{K}}\mathbf{Q} + \frac{\partial \widehat{U}_{\text{NN}}(\mathbf{Q})}{\partial \mathbf{Q}} \right\|_F. \quad (6.31)$$

The proposed LOpInf-SpML ROMs provide a flexible learning approach that combines the Lagrangian operator inference method [223] with the structure-preserving neural networks.

6.2.5 Training algorithm

Algorithm 3 summarizes the proposed method which learns the LOpInf-SpML ROM (6.30) in two separate stages. After projecting the high-dimensional data onto a low-dimensional subspace of appropriate dimension, we first learn the linear reduced operators by solving the constrained optimization problem (6.24). The constrained linear least-squares problems for inferring the linear reduced operators in the numerical studies have been solved using semidefinite programming mode provided by the CVX package [57]. To preserve the Lagrangian structure in the linear ROM, we declare $\widehat{\mathbf{K}}$ and $\widehat{\mathbf{C}}$ as symmetric matrix variables and impose $\widehat{\mathbf{K}} - 10^{-8} \cdot \mathbf{I}_r \succeq \mathbf{0}$ and $\widehat{\mathbf{C}} - 10^{-8} \cdot \mathbf{I}_r \succeq \mathbf{0}$. These constraints ensure that the inferred reduced operators are symmetric positive-definite matrices.

In the SpML training procedure, the projected snapshot data is split into $\widehat{\mathbf{Q}} = [\widehat{\mathbf{Q}}_{\text{train}}, \widehat{\mathbf{Q}}_{\text{val}}]$, where the training dataset $\widehat{\mathbf{Q}}_{\text{train}}$ is used for training the model parameters and the validation dataset $\widehat{\mathbf{Q}}_{\text{val}}$ is used for assessing the model's performance. The training data is divided into minibatches of size N_{mb} , which determines the number of samples that are propagated through the NN before the weights are updated. We use a minibatch size of $N_{mb} = 250$ for all the examples in Section 6.3. To aid in physical interpretability of the learned functions, the energy and dissipation terms are constrained to be positive through an added penalty to the loss function. The terms obtained through LOpInf are already constrained to be positive, and the learned terms from spML are interpreted as a perturbation about the LOpInf state. For this reason, the constraint is not enforced on the spML terms directly but on the total terms as

$$\mathcal{J}_+ = \left\| \text{ReLU}(-1(\widehat{T}(\dot{\mathbf{Q}}) + \widehat{T}_{\text{NN}}(\dot{\mathbf{Q}}))) + \text{ReLU}(-1(\widehat{U}(\widehat{\mathbf{Q}}) + \widehat{U}_{\text{NN}}(\widehat{\mathbf{Q}}))) + \right. \\ \left. \text{ReLU}(-1(\widehat{\mathcal{F}}(\dot{\mathbf{Q}}) + \widehat{\mathcal{F}}_{\text{NN}}(\dot{\mathbf{Q}}))) \right\|_F, \quad (6.32)$$

where ReLU represents the rectified linear unit which is non-zero only for positive values. Since the argument of the function is multiplied by -1 , this loss term only penalizes negative values.

The NN parametrizations in this work use the Swish activation function for all the hidden layers and the linear activation function for the final layer. All of the network parameters are collected in θ . The weights for each layer are randomly initialized using the Glorot initialization algorithm [78]. The loss function (6.31) is minimized using the ADAM optimizer [124], which is a widely used stochastic optimization method in the machine learning community. During the optimization process, weights inside the NN functions along with the coefficients in (6.25) and (6.26) are optimized simultaneously. All the LOpInf-SpML ROMs in Section 6.3 are trained using the ADAM optimizer with an initial learning rate of 10^{-4} .

6.3 Numerical results

In this section, we apply the proposed ML-enhanced structure-preserving operator inference method to three nonlinear mechanical examples with increasing complexity. In Section 6.3.1 we apply the proposed method to a one-dimensional conservative rod model with isolated elastic nonlinearities to investigate its performance for conservative nonlinear mechanical systems. In Section 6.3.2 we study a two-dimensional geometrically nonlinear membrane model with internal damping to show the effectiveness of the proposed approach for nonconservative mechanical systems. Finally, in Section 6.3.3 we consider a dataset of experimental measurements of a lap-joint beam structure to demonstrate the proposed method's ability to learn structure-preserving surrogate models from experimental data.

6.3.1 Conservative rod model with spatially isolated elastic nonlinearities

We consider the conservative structural dynamics of a one-dimensional rod with spatially isolated elastic nonlinearities. The rod is modeled with a spring-mass model consisting of a chain of masses connected with spring elements. A schematic of this nonlinear mass-spring model is shown in Figure 6.2.

Algorithm 3. ML-enhanced structure-preserving operator inference (LOpInf-SpML)

Input: High-dimensional snapshot data $\mathbf{Q} \in \mathbb{R}^{n \times K}$ and reduced dimension r . Additional hyperparameters: size of training set T_{train} and validation set T_{val} , starting learning rate η , batch size N_b , maximum number of epochs N_{epochs} , and number of minibatches $N_{mb} = (T_{train} - T_{val})/N_b$.

Output: Reduced linear operators $\hat{\mathbf{M}}_{NN}, \hat{\mathbf{C}}, \hat{\mathbf{K}} \in \mathbb{R}^{r \times r}$, neural operator for reduced nonlinear potential energy $\hat{U}_{NN}$, and neural operator for reduced nonlinear dissipation function $\hat{\mathcal{F}}_{NN}$

- 1: Build basis matrix $\mathbf{V}_r \in \mathbb{R}^{n \times r}$ from SVD of $\mathbf{Q} \in \mathbb{R}^{n \times K}$ (6.11).
 - 2: Project to obtain reduced state data $\hat{\mathbf{Q}} \in \mathbb{R}^{r \times K}$ (6.13).
 - 3: Compute reduced time-derivative data $\dot{\hat{\mathbf{Q}}}, \ddot{\hat{\mathbf{Q}}} \in \mathbb{R}^{r \times K}$ as in (6.16) from the projected data $\hat{\mathbf{Q}}$ using an appropriate finite-difference scheme.
 - 4: Solve constrained linear least-squares problem (6.24) to nonintrusively infer reduced operators $\hat{\mathbf{K}}$ and $\hat{\mathbf{C}}$.
 - 5: Split projected snapshot data into $\hat{\mathbf{Q}} = [\hat{\mathbf{Q}}_{train}, \hat{\mathbf{Q}}_{val}]$, similarly for $\dot{\hat{\mathbf{Q}}}, \ddot{\hat{\mathbf{Q}}}$.
 - 6: Randomly initialize weights $\boldsymbol{\theta}^0$.
 - 7: Initialize epoch counter $n_e = 0$.
 - 8: **while** $n_e < N_{epochs}$ **do**
 - 9: **for** $i = 1 : N_{mb}$ **do**
 - 10: Build a minibatch from the training set, $\hat{\mathbf{Q}}_{batch} \subseteq \hat{\mathbf{Q}}_{train}$, similarly for $\dot{\hat{\mathbf{Q}}}_{batch}, \ddot{\hat{\mathbf{Q}}}_{batch}$.
 - 11: Evaluate $\hat{U}_{NN}(\hat{\mathbf{Q}}_{batch})$ (6.25), $\hat{T}_{NN}(\dot{\hat{\mathbf{Q}}}_{batch})$ (6.27), and $\hat{\mathcal{F}}_{NN}(\dot{\hat{\mathbf{Q}}}_{batch})$ (6.26), which are parameterized by $\boldsymbol{\theta}^{N_{mb}n_e+i}$.
 - 12: Evaluate loss function from residual as shown in (6.31).
 - 13: Accumulate loss and compute gradient of loss function $\nabla_{\boldsymbol{\theta}} \mathcal{J}$.
 - 14: Update weights $\boldsymbol{\theta}^{N_{mb}n_e+i+1} = \text{ADAM}(\eta, \nabla_{\boldsymbol{\theta}} \mathcal{J}, \boldsymbol{\theta}^{N_{mb}n_e+i})$.
 - 15: **end for**
 - 16: Repeat **for loop** on $\hat{\mathbf{Q}}_{val}, \dot{\hat{\mathbf{Q}}}_{val}, \ddot{\hat{\mathbf{Q}}}_{val}$ with updated weights.
 - 17: Accumulate loss on $\hat{\mathbf{Q}}_{val}, \dot{\hat{\mathbf{Q}}}_{val}, \ddot{\hat{\mathbf{Q}}}_{val}$ to evaluate network convergence.
 - 18: $n_e = n_e + 1$
 - 19: **end while**
-

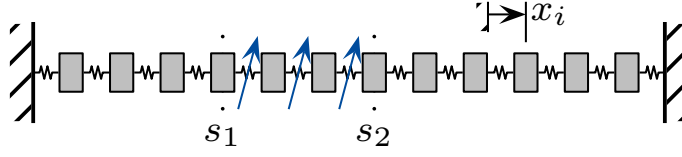


Figure 6.2. Conservative rod model. The nonlinearities are localized within the region $s \in (s_1, s_2)$, while the remainder of the rod is linear.

FOM implementation

For this numerical experiment, we consider a rod of length $\ell = 1$ and discretize it with a chain of $n = 64$ mass elements with equal mass $m = 1.56 \times 10^{-2}$ which leads to a state vector of dimension $\mathbf{q} \in \mathbb{R}^{64}$. The masses are connected with spring elements with a constant linear stiffness $\kappa = 65.0$. The region with isolated nonlinearities, i.e. $s \in (0.25, 0.35)$, is modeled with additional cubic spring elements with cubic coefficient $\eta = 2.62 \times 10^5$. As a result, the interaction force f_i between mass elements m_i and m_{i+1} is

$$f_i = \begin{cases} \kappa(q_{i+1} - q_i) + \eta(q_{i+1} - q_i)^3 & \text{if } i \in \{22, 23, 24, 25, 26, 27, 28\} \\ \kappa(q_{i+1} - q_i) & \text{otherwise} \end{cases},$$

where q_i represents the displacement of the i th element. For numerical comparison with intrusive Lagrangian ROMs, we provide details about the Lagrangian FOM for this example. The FOM Lagrangian for this conservative mechanical system can be written as

$$L(\mathbf{q}, \dot{\mathbf{q}}) = \frac{1}{2} \dot{\mathbf{q}}^\top \mathbf{M} \dot{\mathbf{q}} - \frac{1}{2} \mathbf{q}^\top \mathbf{K} \mathbf{q} - \sum_{i=22}^{28} \frac{\eta}{4} (q_{i+1} - q_i)^4, \quad (6.33)$$

where $\mathbf{M} \in \mathbb{R}^{64 \times 64}$ and $\mathbf{K} \in \mathbb{R}^{64 \times 64}$ are the symmetric positive-definite system matrices. The governing equations for the Lagrangian FOM are

$$\mathbf{M} \ddot{\mathbf{q}} - \frac{\partial L(\mathbf{q}, \dot{\mathbf{q}})}{\partial \mathbf{q}} = \mathbf{0}. \quad (6.34)$$

We specify the initial condition in terms of modal velocities $\mathbf{q}(0) = \mathbf{0}$, $\dot{\mathbf{q}}(0) = \Phi \begin{bmatrix} \alpha_1, \dots, \alpha_{64} \end{bmatrix}^\top$, where $\Phi \in \mathbb{R}^{64 \times 64}$ is the modal transformation matrix based on the linear modal analysis and α_i values are the initial modal velocity coefficients.

Learning setup

In this study, we consider a training initial condition with nonzero initial modal velocity for the first three modes

$$\alpha_1 = 4.47 \times 10^{-1}, \quad \alpha_2 = 2.24 \times 10^{-1}, \quad \alpha_3 = 3.16 \times 10^{-1}, \quad \alpha_i = 0 \text{ for } 4 \leq i \leq 64. \quad (6.35)$$

We generate the simulated snapshot data by numerically integrating the conservative FOM (6.34) for total time $T = 16$ using a Newmark integrator with a fixed time step of $\Delta t = 10^{-3}$. For the offline training, we use $K = 8000$ snapshots from $t = 0$ to $T_{\text{train}} = 8$ as training data. For the NN training, we use data from $t = 7.5$ to $T_{\text{train}} = 8$ as the validation data. The testing data consists of FOM snapshot data over the testing interval $[8, 16]$. We consider ROMs of size $r = 4$, $r = 6$, and $r = 8$ to show the increase in the accuracy of the learned ROMs with an increase in the reduced dimension.

Since the high-dimensional system is a conservative mechanical system, we learn a nonlinear ROM of the following form

$$\ddot{\hat{\mathbf{q}}} + \hat{\mathbf{K}}\hat{\mathbf{q}} + \frac{\partial \hat{U}_{\text{NN}}(\hat{\mathbf{q}})}{\partial \hat{\mathbf{q}}} = \mathbf{0},$$

where $\hat{\mathbf{K}}$ is the reduced linear stiffness matrix and $\hat{U}_{\text{NN}}$ is the nonlinear ROM operator for the nonlinear potential energy. For this numerical example, we use four hidden layers with $\{64, 30, 20, 6\}$ units in the NN parametrization for the reduced potential energy function $\hat{U}_{\text{NN}}$.

Table 6.1. Relative state error comparison between different methods.

Method	Training data			Test data		
	$r = 4$	$r = 6$	$r = 8$	$r = 4$	$r = 6$	$r = 8$
Intrusive	3.0×10^{-2}	1.3×10^{-2}	1.1×10^{-2}	8.6×10^{-2}	3.5×10^{-2}	2.6×10^{-2}
LOpInf	3.1×10^{-2}	3.1×10^{-2}	3.1×10^{-2}	8.6×10^{-2}	8.6×10^{-2}	8.6×10^{-2}
LOpInf-SpML	2.5×10^{-2}	1.1×10^{-2}	8.1×10^{-3}	7.9×10^{-2}	2.8×10^{-2}	2.6×10^{-2}
POD-SpML		3.7×10^{-2}			1.9×10^{-1}	

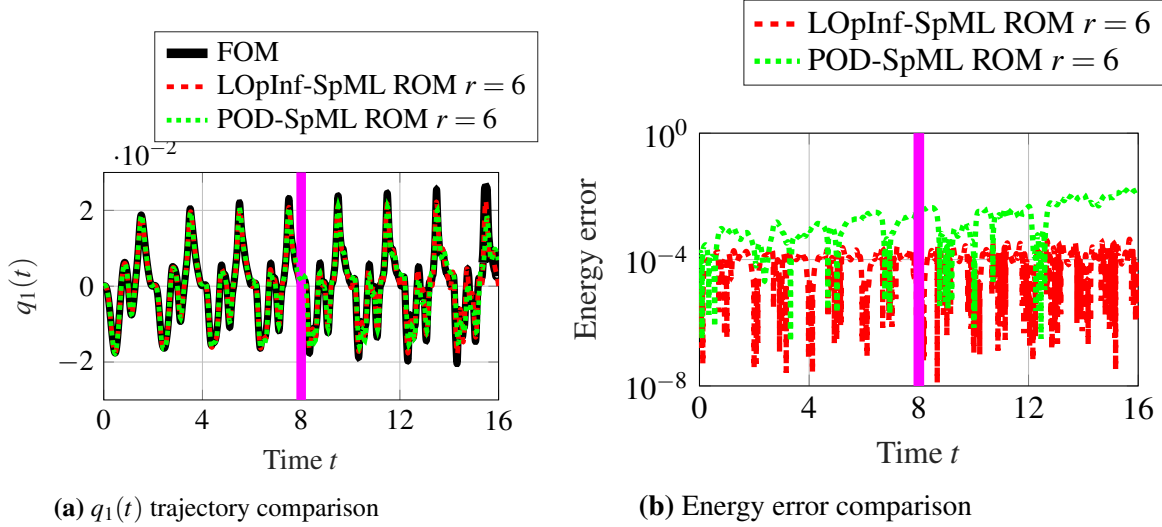


Figure 6.3. Conservative rod model. (a) The LOpInf-SpML ROM correctly predicts $q_1(t)$ trajectory 100% outside the training time interval. (b) The L-OpInf SpML ROM exhibits bounded energy error behavior. The magenta line in both plots indicates the end of the training time interval at $T_{\text{train}} = 8$.

Numerical results

Table 6.1 compares the numerical performance of four different model reduction approaches for the conservative rod model example. The relative state error values reported in Table 6.1 compute

$$\text{relative state error} = \frac{\|\mathbf{Q} - \mathbf{V}_r \tilde{\mathbf{Q}}\|_F}{\|\mathbf{Q}\|_F},$$

where $\tilde{\mathbf{Q}}$ is reduced snapshot data obtained by numerically integrating the ROM. For both training and test data regimes, we observe that the relative state error decreases with an increase in the reduced dimension for the intrusive and the proposed ML-enhanced structure-preserving

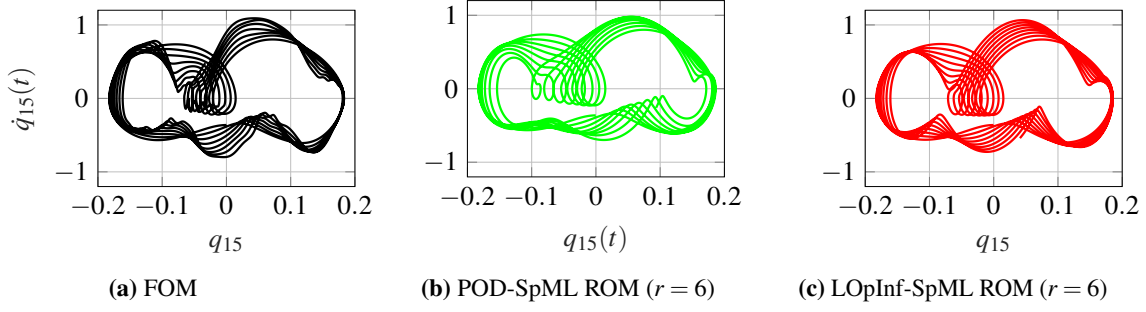


Figure 6.4. Conservative rod model. An accurate phase space portrait obtained using the LOpInf-SpML ROM demonstrates that the proposed approach has learned the underlying nonlinear dynamics.

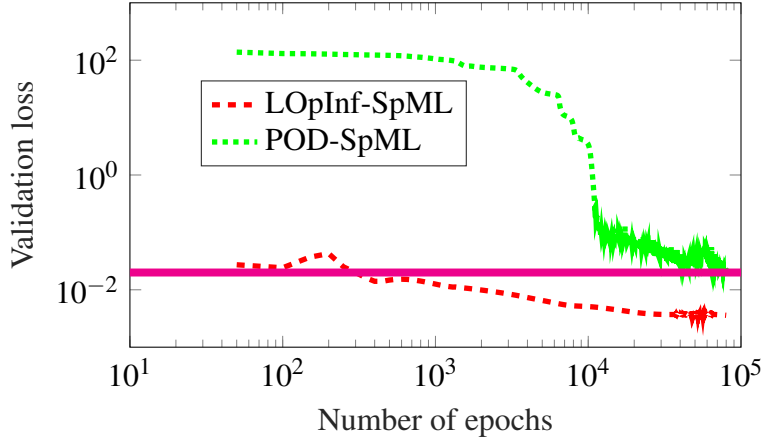


Figure 6.5. Conservative rod model. The LOpInf-SpML ROM starts at a significantly lower validation loss value and achieves the lowest error achieved by the POD-SpML ROM in $100\times$ less epochs. The dotted black line indicates the lowest validation error achieved by the POD-SpML ROM.

approach whereas the errors for the nonintrusive linear Lagrangian ROM based on L-OpInf do not show further improvement. The POD-SpML approach yields the least accurate ROM in both training and test data regimes, which shows that directly training a SpML on the reduced data is not a good strategy. Overall, the proposed ML-enhanced structure-preserving OpInf approach yields the most accurate ROMs in both training and test data regimes for all three r values.

The $q_1(t)$ trajectory plots in Figure 6.3a compare the approximate ROM solutions against the benchmark FOM solution from $t = 0$ to $T = 16$. Both the POD-SpML ROM and the LOpInf-SpML ROM solutions accurately track the FOM solution in the training data regime. In the

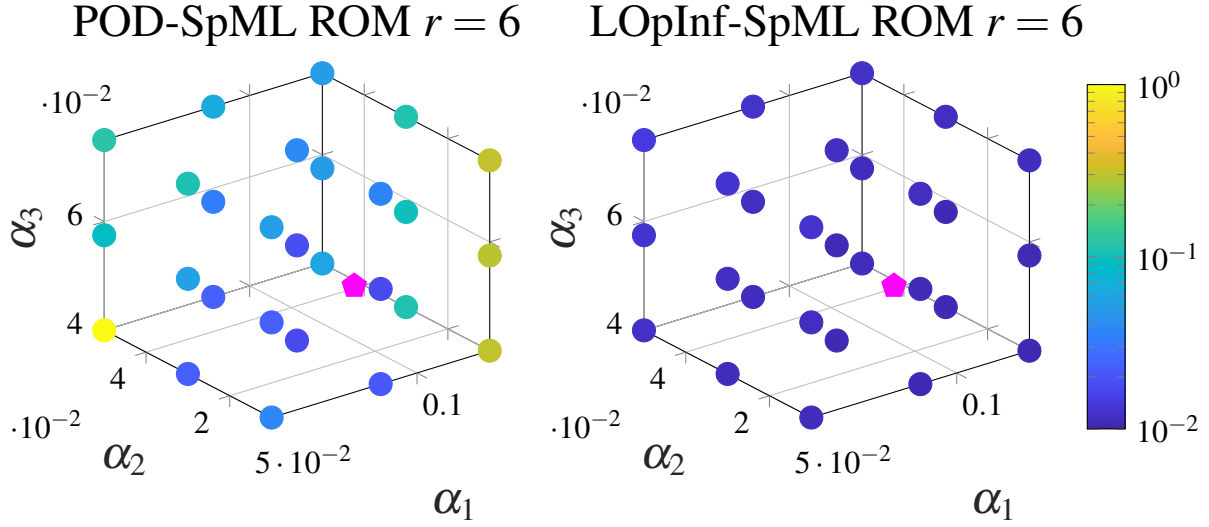


Figure 6.6. Conservative rod model. The LOpInf-SpML ROM is robust to perturbations in the training initial condition whereas the POD-SpML ROM fails to provide accurate predictions for unseen initial conditions. The magenta star in both plots represents the training initial condition.

test data regime, we observe that the LOpInf-SpML ROM solution is indistinguishable from the FOM solution whereas the POD-SpML ROM solution fails to provide accurate predictions beyond $t = 12$. The energy error plots in In Figure 6.3b, we observe that the proposed approach yields a structure-preserving ROM that exhibits bounded energy error behavior whereas the energy error for the POD-SpML ROM grows with time.

Phase space portraits are an invaluable tool for studying conservative dynamical systems. Figure 6.4 compares the accuracy of the phase-space portraits in the $(q_{15}, \dot{q}_{15})$ phase space. The phase space plot based on the ROM in Figure 6.4c accurately captures the qualitative behavior of the phase space plot based on the FOM in Figure 6.4a which demonstrates that the proposed learning approach has learned the nonlinear characteristics of the problem.

The error plots in Figure 6.5 track the decay in the validation loss for both approaches during the offline training stage. Both neural networks are trained from the reduced data using the same SpML architecture. Due to its two-step learning strategy, the proposed approach starts at a validation loss value four orders of magnitude lower than the POD-SpML method. The validation loss for the ML-enhanced structure-preserving operator inference approach continues

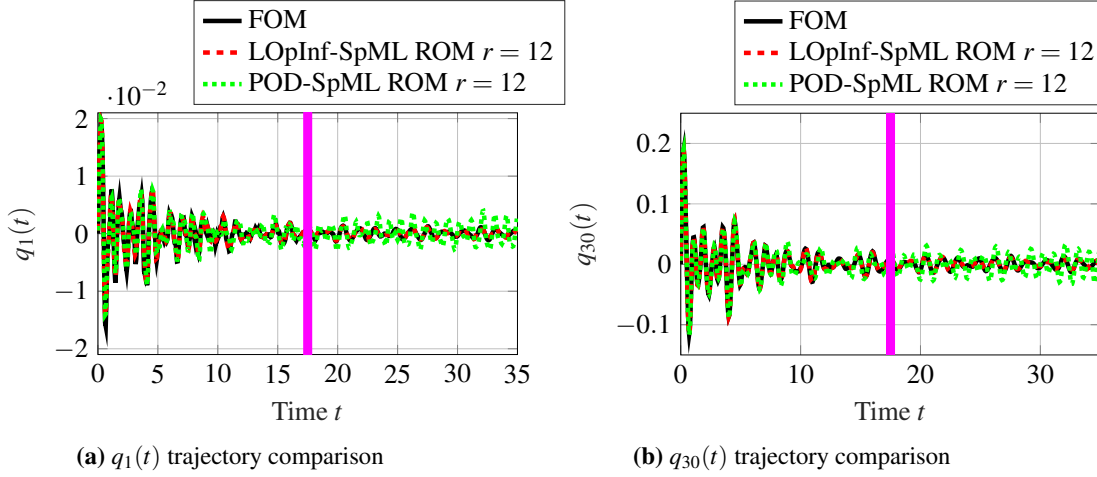


Figure 6.7. Nonconservative membrane model. Plots in (a) and (b) show that the LOpInf-SpML ROM of reduced dimension $r = 12$ provides accurate predictions 100% outside the training time interval. The magenta line indicates the end of the training time interval.

to decrease as training proceeds, remaining significantly lower than the POD-SpML ROM. The dotted horizontal black line in Figure 6.5 shows that the LOpInf-SpML achieves the lowest test error achieved by the POD-SpML ROM in $100\times$ less epochs. This study highlights that using the LOpInf ROM for learning the linear ROM components provides higher accuracy along with significant computational savings in offline training.

Finally, we study the generalizability of learned models by evaluating their accuracy for unseen initial conditions of the form (6.35). We simulate both LOpInf-SpML and POD-SpML ROMs until $T = 16$ for 27 test initial conditions based on different combinations of $(\alpha_1, \alpha_2, \alpha_3)$ values and compare the relative state error values for each of those test initial conditions. The comparison in Figure 6.6 shows that the proposed approach achieves a relative state error of approximately 10^{-2} over a wide range of initial conditions, an order of magnitude more accurate than the POD-SpML approach.

6.3.2 Two-dimensional nonlinear membrane

We consider a two-dimensional nonlinear membrane model with internal damping to study the performance of the proposed approach for nonconservative mechanical systems.

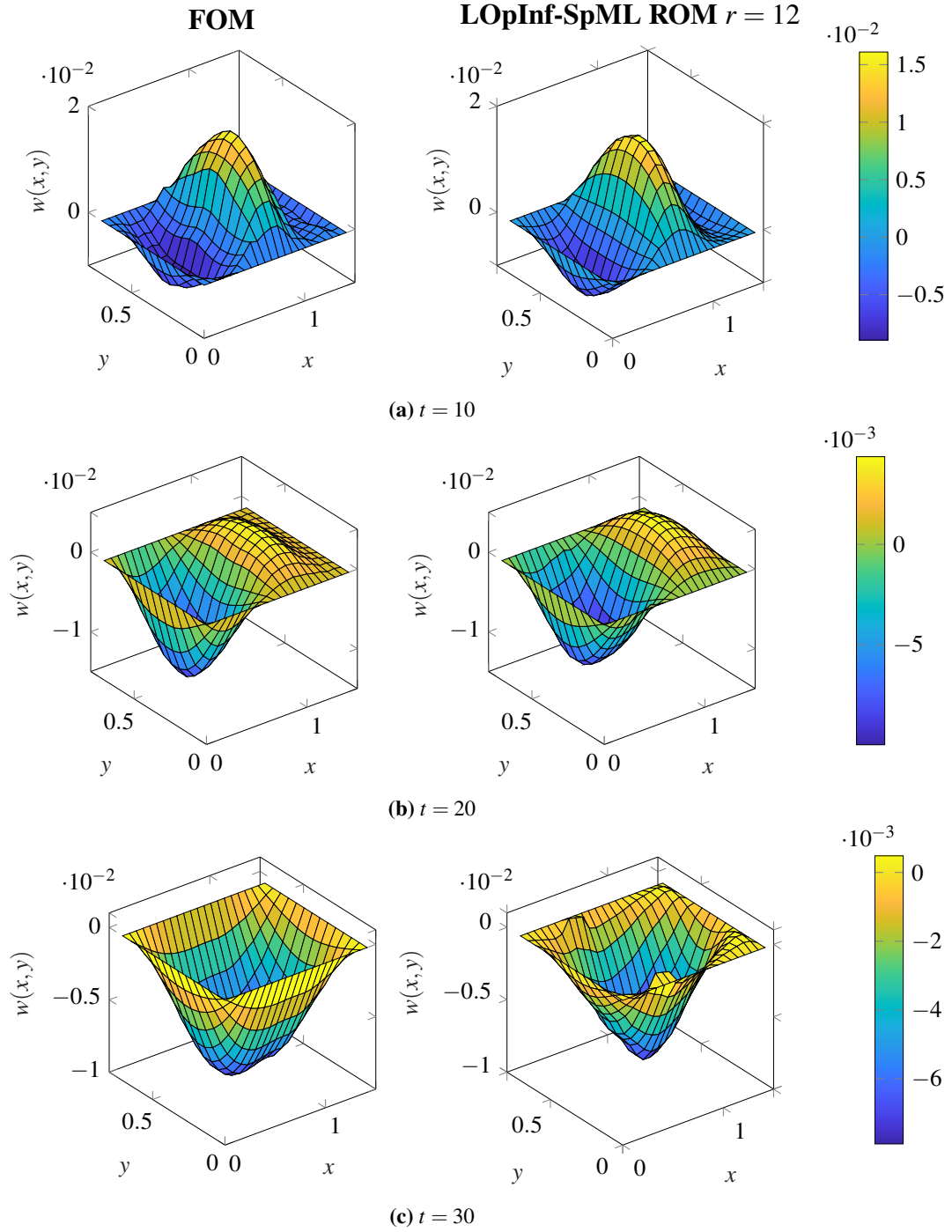


Figure 6.8. Nonconservative membrane model. The full-field solution comparison in plots (a), (b), and (c) shows that the LOpInf-SpML ROM provides accurate predictions over the entire computational domain.

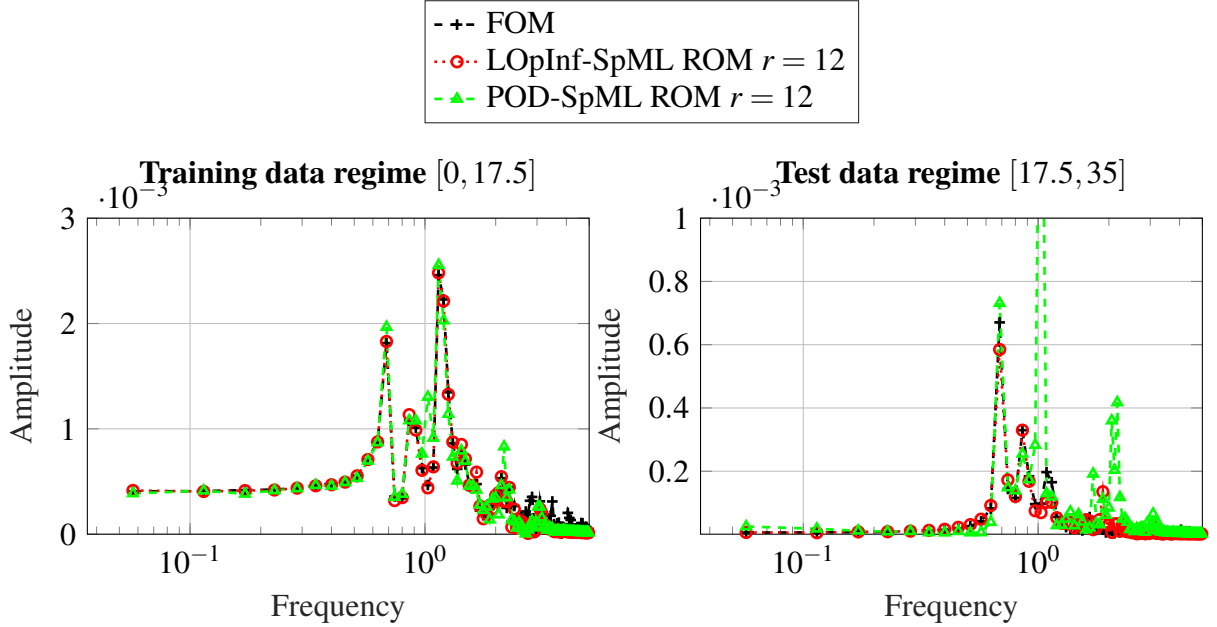


Figure 6.9. Nonconservative membrane model. In the training data regime, both ROMs capture the frequency content in the training data accurately. In the test data regime, we observe that the LOpInf-SpML ROM performs significantly better than the POD-SpML ROM.

FOM implementation

We study a two-dimensional membrane with dimensions $(l_x, l_y) = (\sqrt{2}, 1/\sqrt{2})$. The FOM for this numerical example is obtained using two-dimensional membrane elements with $N_x = 21$ nodes in the x -direction and $N_y = 13$ nodes in the y -direction. In these membrane elements, the in-plane stiffness comes from a pre-tension force. Each node in the two-dimensional membrane element has an out-of-plane displacement, which after applying the clamped boundary conditions leads to a FOM with $n = 240$ degrees of freedom.

The linear stiffness associated with each element is defined as $(k_x, k_y) = (n \cdot (l_x/N_x)^2, n \cdot (l_y/N_y)^2)$, and the linear stiffness matrix $\mathbf{K}$ is obtained by assembling elements based on their connectivity. The linear damping matrix is proportional to the stiffness matrix $\mathbf{C} = 10^{-4}\mathbf{K}$. The nonlinear behavior is modeled using cubic nonlinearities with nonlinear stiffness and nonlinear damping coefficients $k_{nl} = c_{nl} = 0.02n^{3/2} = 74.361$. Such cubic nonlinearities are commonly used for modeling complex nonlinearities exhibiting softening or hardening behavior, such as

geometric nonlinearities [120]. The governing equations for the nonconservative FOM are of the following form

$$\mathbf{M}\ddot{\mathbf{q}} + \frac{\partial \mathcal{F}(\dot{\mathbf{q}})}{\partial \dot{\mathbf{q}}} + \frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} = \mathbf{0}. \quad (6.36)$$

Similar to the conservative rod example, the proposed LOpInf-SpML approach learns a structure-preserving nonlinear ROM purely from the data, without any prior knowledge about the FOM (6.36) for the two-dimensional nonlinear membrane.

Learning setup

In this numerical example, we generate simulated data for the training and testing datasets by integrating the nonconservative FOM (6.36) with the Newmark integrator until the final time $T = 35$. We use a fixed time step of $\Delta t = 5 \times 10^{-3}$. We consider $K = 3500$ snapshots over the training time interval $[0, 17.5]$ as training data. For the NN training, we use data from $t = 15$ to $T_{\text{train}} = 17.5$ as the validation data. Similar to the conservative rod example, we study the time extrapolation capability of the LOpInf-SpML ROM over the testing interval $[17.5, 35]$. We consider ROMs of size $r = 12$ for comparison between the POD-SpML approach and the proposed LOpInf-SpML approach.

Due to the nonconservative nature of the problem, we consider a nonconservative nonlinear ROM of the following form

$$\ddot{\hat{\mathbf{q}}} + \hat{\mathbf{C}}\dot{\hat{\mathbf{q}}} + \frac{\partial \widehat{\mathcal{F}}_{\text{NN}}(\dot{\hat{\mathbf{q}}})}{\partial \dot{\hat{\mathbf{q}}}} + \hat{\mathbf{K}}\hat{\mathbf{q}} + \frac{\partial \widehat{U}_{\text{NN}}(\hat{\mathbf{q}})}{\partial \hat{\mathbf{q}}} = \mathbf{0}, \quad (6.37)$$

where $\hat{\mathbf{K}}$ and $\hat{\mathbf{C}}$ are the linear reduced operators, and $\widehat{\mathcal{F}}_{\text{NN}}$ and $\widehat{U}_{\text{NN}}$ are the neural network parametrizations for the nonlinear dissipation function and the nonlinear potential energy function, respectively. Both $\widehat{\mathcal{F}}_{\text{NN}}$ and $\widehat{U}_{\text{NN}}$ are trained using a five-layer neural network with $\{128, 64, 30, 20, 12\}$ units.

Numerical results

In Figure 6.7, we compare the ROM solutions with the FOM solution for two different degrees of freedom. The $q_1(t)$ trajectory plots in Figure 6.7a show that the LOpInf-SpML ROM captures the decay in the FOM solution accurately whereas the POD-SpML ROM provides inaccurate predictions outside the training data regime. For the $q_{30}(t)$ trajectory plots in Figure 6.7b, we observe that the LOpInf-SpML ROM solution is indistinguishable from the FOM solution which demonstrates the proposed approach's ability to learn the internal nonlinear damping of the high-dimensional system. The POD-SpML ROM solution, on the other hand, fails to learn the internal damping which leads to substantially high errors in the test data regime.

Figure 6.8 compares the FOM solution with the ROM solution at different time instances. In Figure 6.8a, the LOpInf-SpML ROM provides an approximate full-field solution that agrees with the FOM solution. In Figure 6.8b, we observe that the proposed approach accurately predicts the membrane displacements over the entire two-dimensional domain. Even though the solution at $t = 30$ in Figure 6.8c has been significantly damped out, the ROM solution predicts the displacement amplitude over the two-dimensional domain with reasonable accuracy.

Finally, we study the frequency characteristics of this nonconservative problem by analyzing the time-series data for the first node. We study the spectral information in the training data and the test data separately. All the models are numerically integrated with a time step of $\Delta t = 5 \times 10^{-3}$, so the sampling frequency for this setting is 200Hz . We apply the standard fast Fourier transform to obtain the frequency-amplitude curves in Figure 6.9. We observe that both LOpInf-SpML and POD-SpML capture the frequency content in the training data regime accurately with the LOpInf-SpML ROM performing marginally better. In the test data regime, we observe that the amplitude-frequency curve for the LOpInf-SpML ROM agrees with the FOM amplitude-frequency curve whereas the POD-SpML ROM only performs well for frequencies below 0.5 Hz . This spectral analysis shows that the ROM learned with the proposed ML-enhanced approach has learned the damped dynamics accurately.

6.3.3 Half Brake-Reuß beam

Accurate prediction of the dynamics of jointed structures remains a challenging problem due to the strong nonlinearities at the frictional interfaces found in joints. The main purpose of a joint in an engineering structure is to usually connect two separate substructures in a stiff manner. One of the principal roadblocks to accurate high-fidelity modeling is the lack of understanding of the physics behind the energy dissipation mechanisms at the joints. In part, this lack of understanding is due to the multiscale nature of interfacial mechanics for jointed structures where the influencing sources can range from macro-scale geometry and loading to nano-scale grain boundaries, see [25] for more details.

Due to the problem-specific and dynamic nature of interfacial mechanics, it is unrealistic to expect a universal law of friction that is predictive across most types of interfaces. For such problems, the structural and mechanical engineering community is mainly interested in developing a predictive model for different types of joints such as lap joints, dove-tail and fir-tree joints, and tape joints. In this section, we apply the proposed learning approach to learn a predictive model of a beam structure with a lap joint from a dataset consisting of experimental measurements. The dataset considered in this numerical example is obtained from the experimental setup in [45, 110] where the authors used high-speed cameras combined with digital image correlation (DIC) to obtain the full-field response of the structure known as the "Half Brake-Reuß beam". The data was downloaded from obtained from the authors' repository

¹.

Experimental data collection using DIC measurements

The half Brake-Reuß beam is a modification of the Brake-Reuß beam [26], which is a widely used benchmark structure for studying nonlinear dynamics in jointed systems. The half Brake-Reuß beam consists of two beams that are joined through a three-bolt lap joint as shown in Figure 6.10. The lap joint is the most common type of interface found in built-up structures

¹<https://github.com/mattiacedese/BRBtesting>

and is defined as the mating of two components through a bolted connection.

The free-response data considered in this study was obtained by first exciting the beam at a specific frequency of interest (i.e., at the natural frequency of the mode of interest) with a fixed sinusoidal signal and then detaching the shaker once resonance is identified. Even though this procedure is meant to isolate a mode of interest, the full-field response includes multiple modes due to modal coupling resulting from the nonlinearity. The nonlinearity in this structure arises due to energy dissipation caused by slipping and possible intermittent contact in the joint.

For the free-response experiments, the beam structure was suspended horizontally using bungee cords to simulate the free-free boundary conditions. Using the DIC technique, the vibrational motion of $n = 206$ measurement points was tracked for $T = 3.06$ s with a sampling frequency of $f_{\text{sampling}} = 5000$ Hz. The recorded images from this experiment were post-processed to build a dataset containing the time history of the displacements of the measurement points.

Learning setup

For this numerical example, we use the first $K = 5086$ snapshots over the training time interval $[0, 1.02]$ s to train the LOpInf-SpML ROM. For the NN training, we use data from $t = 0.92$ s to $T_{\text{train}} = 1.02$ s as the validation data. The testing data consists of experimental measurements over the testing time interval $[1.02, 3.06]$ s. Joints introduce two qualitatively important features to a structure: amplitude-dependent stiffness and amplitude-dependent damping. The aim of this study is to use the LOpInf-SpML ROM as a surrogate model for the jointed beam structure and provide accurate predictions of these qualitative properties outside the training data regime. To accurately capture the amplitude-dependent frequency and damping characteristics of the jointed structure, we consider the most expressive ROM form (6.30) in this study. In addition to learning the neural differential operators for the dissipation and potential energy functions, we also learn a reduced kinetic energy term $\hat{T}_{\text{NN}}$ with a symmetric positive-definite reduced mass matrix $\hat{\mathbf{M}}_{\text{NN}}$ in the offline training phase. Both $\hat{\mathcal{F}}_{\text{NN}}$ and $\hat{U}_{\text{NN}}$ are trained using a four-layer neural network with $\{16, 16, 16, 16\}$ units.

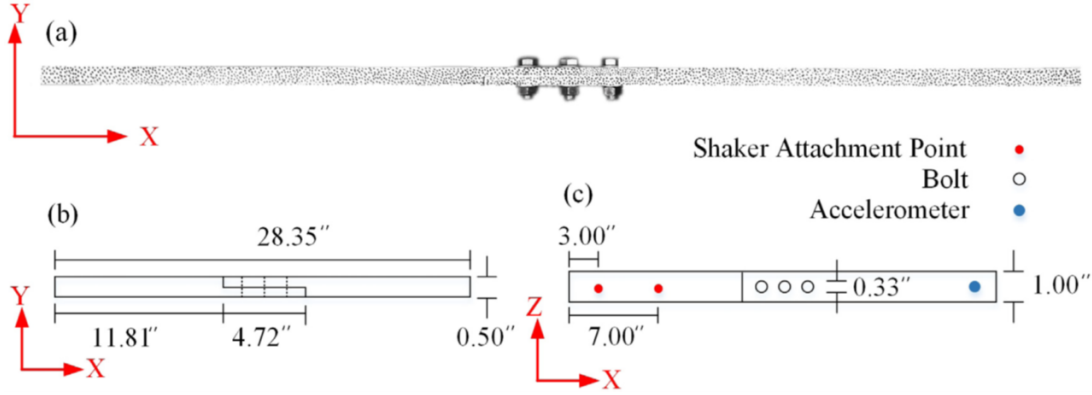


Figure 6.10. Half Brake-Reuß beam. Geometry of the jointed structure including a photo with DIC speckle pattern shown (a), top view (b), and front view (c). Figure reprinted from [45], Copyright (2022), with permission from Elsevier and the authors.

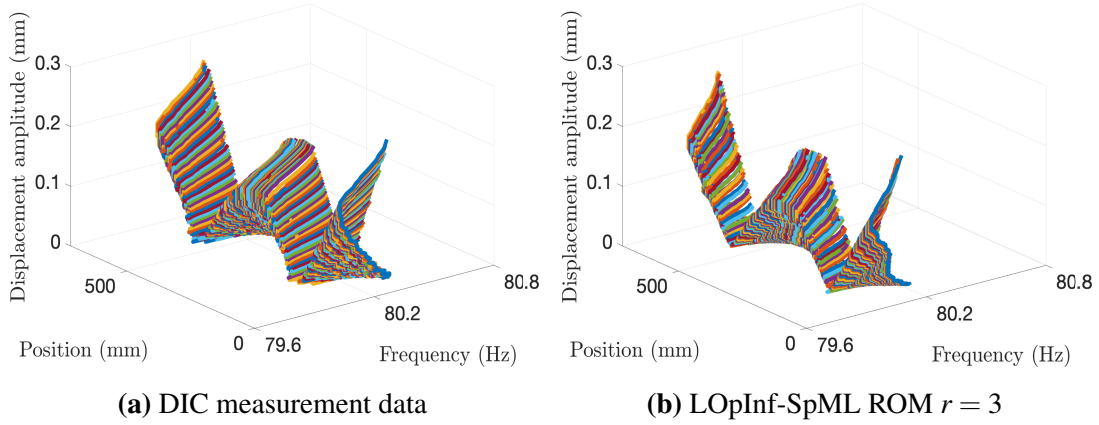


Figure 6.11. Half Brake-Reuß beam. The LOpInf-SpML ROM of size $r = 3$ accurately predicts the amplitude-dependent frequency characteristics and yields backbone curves that appear to agree with the backbone curves obtained directly from the DIC measurements.

Numerical results

Figure 6.11 shows the amplitude-dependent frequency curves of the jointed structure along different measurement points along the length of the beam. In Figure 6.11b, we observe that the LOpInf-SpML ROM accurately predicts the frequency range with less than 0.5% error. More importantly, the amplitude-dependent frequency curve for the learned ROM has the same shape as the one obtained from the experimental data which demonstrates that the proposed approach has learned the nonlinear characteristics of the problem.

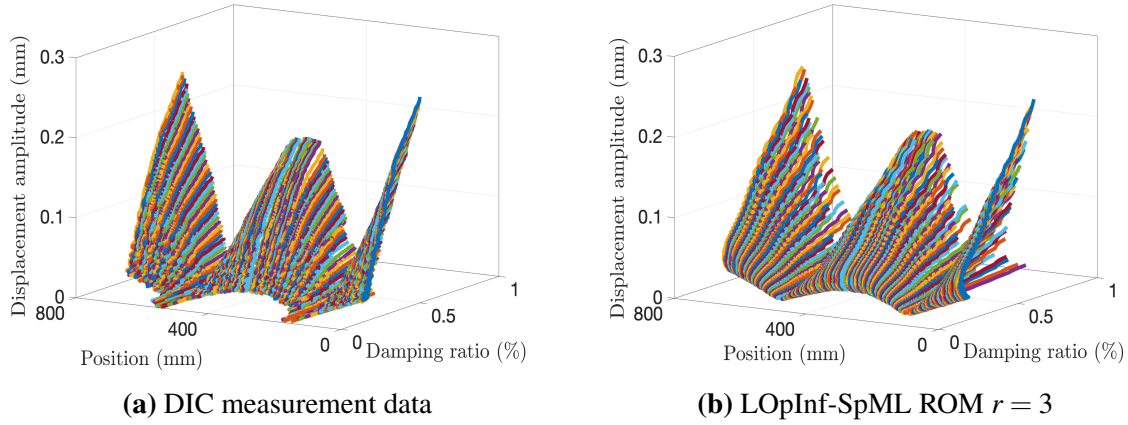


Figure 6.12. Half Brake-Reuß beam. The amplitude-dependent damping plots based on the LOpInf-SpML ROM of dimension $r = 3$ are reasonably similar to the plots obtained from the DIC measurements.

In Figure 6.12, we obtain the amplitude-dependent damping plots using the LOpInf-SpML ROM and compare them with the plots based on the DIC measurements. Even though the proposed ROM predicts damping ratio values higher than the ones based on experimental data for some points, the amplitude-damping curves obtained using the ROM in Figure 6.12b are reasonably similar to the plots directly obtained from experimental measurements in Figure 6.12a.

As an alternative to displacement-based plots, Figure 6.13 shows the nonlinear characteristics of the jointed structure through energy-dependent plots. The energy of the system at time t is approximated by discretizing the beam into lumped mass elements

$$E(t) = \sum \frac{1}{2} m \dot{q}_i^2(t)$$

where $\dot{q}_i$ is the velocity of the i th measurement point, and $m = M/n$ is the lumped mass of each measurement point. Since the energy calculation utilizes information about the motion at all the points across the length of the beam, the energy-dependent plots provide a more global perspective on the mechanical system under analysis. The frequency-energy plots in Figure 6.13a show that the LOpInf-SpML ROM of reduced dimension $r = 3$ accurately predicts the time-evolution of the frequency-energy relationship. In Figure 6.13b we observe that the learned

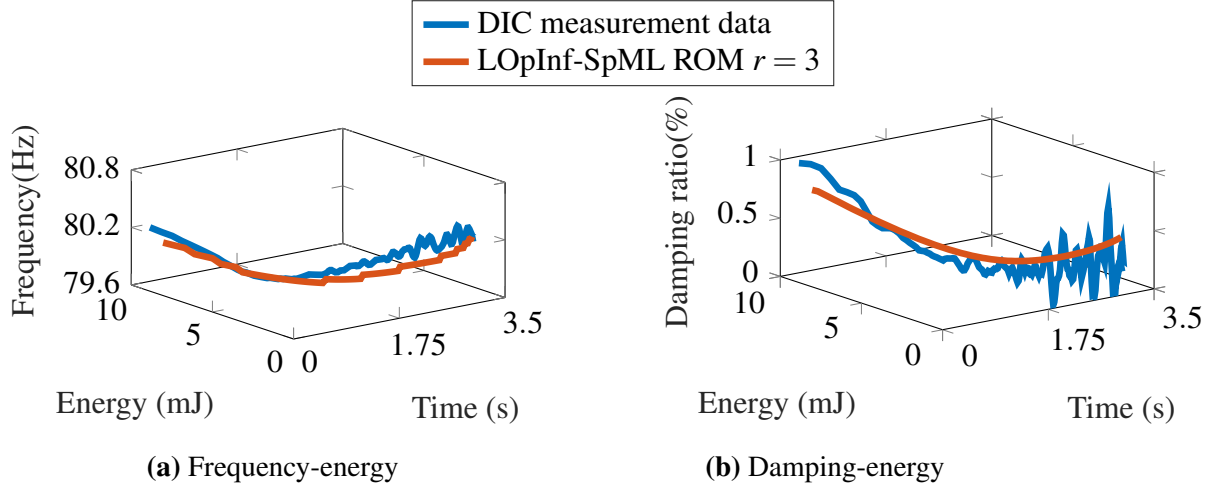


Figure 6.13. Half Brake-Reuß beam. The proposed approach accurately predicts the frequency-energy curve in (a) and captures the overall trend of the challenging damping-energy curve in (b).

ROM accurately captures the overall energy-damping trend but predicts damping ratio values higher than the ones obtained from the DIC measurements. We note that the energy-dependent plots based on DIC measurements are known to demonstrate increased variability towards small amplitudes due to the degradation of the signal-to-noise ratio in DIC measurements.

6.4 Conclusions

We have presented an ML-enhanced model reduction method that learns structure-preserving ROMs of nonlinear mechanical systems from data in a nonintrusive manner. The presented method parametrizes the ROM in terms of the mechanical problem's Lagrangian structure and then learns the reduced operators in a structure-preserving way to ensure that the learned operators are Lagrangian. The proposed learning algorithm uses a POD basis to project the high-dimensional data onto a low-dimensional subspace and then derives the Lagrangian ROM by (i) inferring the linear reduced operators using a structure-preserving nonintrusive operator inference method, and then (ii) learning nonlinear reduced operators using a structure-preserving machine learning method. The proposed ML-enhanced operator inference framework is well-suited for engineering applications where the data are generated from complicated

high-fidelity computational models or from experimental measurements.

The numerical experiments for conservative and nonconservative mechanical systems demonstrate the wide applicability of the proposed approach. The nonlinear conservative rod model shows that the ML-enhanced approach learns long-time stable ROMs with bounded energy error, while also providing accurate predictions for unseen initial conditions. Compared to the POD-SpML approach, the proposed ML-enhanced structure-preserving operator inference approach achieves significant computational savings in the offline training phase which highlights the key role played by the linear Lagrangian ROM in our two-step learning algorithm. The two-dimensional nonlinear membrane example demonstrates the proposed method’s ability to learn accurate ROMs for nonconservative mechanical systems with nonlinear damping. For the half Brake-Reuß Beam example, the learned ROM accurately captures the amplitude-dependent frequency and damping characteristics of the jointed structure from a dataset consisting of experimental measurements. This example highlights the ability of the proposed method to learn a computationally efficient predictive model from experimental observations for applications where the underlying physics is not well understood.

This work has motivated a number of future research directions. The proposed method uses linear basis approximations, which can make offline training prohibitively expensive if the problem requires a large basis size. For such problems, the proposed approach could be combined with autoencoders or structure-preserving nonlinear manifolds to reduce the computational costs in the offline phase. In another direction, the proposed ML-enhanced approach could be extended to mechanical problems with geometric nonlinearities. Finally, we would like to apply our framework to learn models from noisy and sparse data.

6.5 Preview of Chapter 7

Chapter 6 extended the structure-preserving, data-driven ML framework to incorporate a reduced-order linear model. However, all of the previous work up to this point has assumed that

noise is not impacting the predictions. In the intended applications of this work, measurement noise may have a noticeable impact on the performance of these models. To this end, Chapter 7 introduces a framework to obtain predictive intervals when observational noise is present in the data ingested by the networks.

6.6 Remarks

Portions of this dissertation have been published or are being considered for publication. Chapter 6 is composed of the following first co-authored publications:

Harsh Sharma, David A. Najera-Flores, Michael D. Todd, and Boris Kramer. Lagrangian operator inference enhanced with structure-preserving machine learning for nonintrusive model reduction of mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 423:116865, 2024

Chapter 7

Uncertainty Quantification of a Machine Learning Model for Identification of Isolated Nonlinearities with Conformal Prediction

Chapters 3 through 6 have presented a framework for development of ML models that can function as digital twins in an SHM system. However, to provide information that is actionable, SHM systems need to deal with the uncertainty of the predictions provided by these digital twin systems in the presence of noise coming from measurements that serve as inputs to the model. The previous work by the authors [180] considered only noise-free data. However, in real experimental applications, the data are expected to contain noise which motivates the current study on its effect on the ML system predictions. The authors of [185] summarized a variety of ways to perform uncertainty quantification in ML systems, including Monte Carlo dropout [73], Bayesian neural networks [157], neural network ensembles [190], spectral-normalized neural Gaussian processes (SNGP) [144], among others. However, all of these methods require special treatment of the neural network architecture through the inclusion of specific layer types or modified training schemes. For cases where a deterministic neural network is available, an approach to characterizing the observational uncertainty (e.g., from measurement noise in the inputs) in an ML system is through conformal predictions [222].

While traditional estimators such as the jackknife estimator capture measurement un-

certainty, conformal prediction methods are able to additionally capture model bias [13]. Here, the jackknife+ estimator is used to construct measures of uncertainty for the damage parameter at each snapshot. In this work, we illustrate a general structure with an isolated nonlinearity formulation, present a structure-preserving machine learning methodology for estimating the deviatoric force at the boundary, and apply conformal methods to measure the uncertainties about the estimated deviatoric forces. We demonstrate how these uncertainty measures can be leveraged to identify changes in the structure in the presence of noise through hypothesis testing. The chapter is organized as follows. Section 7.1 presents a summary of previous work and the enhancements to existing methods in this chapter. Section 7.2 illustrates the application of the proposed approach on a numerical example while Section 7.3 provides conclusions and discussion of future work opportunities.

7.1 Background and Methodology

This work leverages a structure-preserving machine learning method developed by the authors in [180] and presented in Chapter 5. We briefly describe the methodology here. Section 7.1.1 and 7.1.2 summarize previous work that forms the foundation for this chapter while Section 7.1.3 describes the statistical inference methods used to quantify uncertainty in the model.

7.1.1 Isolated Nonlinearity Formulation

This work considers a general structure that contains an isolated region where nonlinearities can occur. This structure can be decomposed into two adjacent regions $\mathcal{C}_1$ and $\mathcal{C}_2$ as illustrated in Fig. 7.1. Note that region $\mathcal{C}_1$ is described by the internal variables $\mathbf{x}_1 \in \mathbb{R}^{N_1}$, while $\mathbf{x}_2 \in \mathbb{R}^{N_2}$ are the corresponding internal variables for $\mathcal{C}_2$. The equations of motion within $\mathcal{C}_1$ are assumed to be linear and known since the nonlinearities are localized in $\mathcal{C}_2$. As a result, the

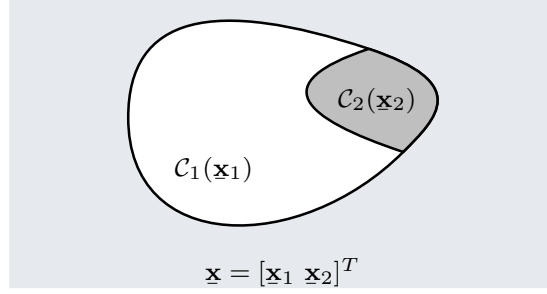


Figure 7.1. General Structure; the nonlinearities are localized within the region $\mathcal{C}_2$.

equations of motion for this structure can be written as

$$\begin{aligned}
 \mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} + \mathbf{N}_\star &= \mathbf{f}(t), \\
 \mathbf{N}_\star &= [\mathbf{0} \quad \mathbf{N}(\mathbf{x}_2)], \quad \mathbf{N} \in \mathbb{R}^{N_2}, \\
 \mathbf{x} &= [\mathbf{x}_1, \mathbf{x}_2] \in \mathbb{R}^{(N_1+N_2)}
 \end{aligned} \tag{7.1}$$

The isolated nonlinearities $\mathcal{C}_2$ are represented by $\mathbf{N} \in \mathbb{R}^{N_2}$. Examples of isolated nonlinearities include joints, nonlinear attachments, or localized damage. Based on the work in [199, 258], the equations of motion can be partitioned into two systems of equation terms as

$$\begin{aligned}
 \mathbf{M}\ddot{\mathbf{w}} + \mathbf{C}\dot{\mathbf{w}} + \mathbf{K}\mathbf{w} &= \mathbf{f}(t) - \mathbf{Q}_\star, \\
 \mathbf{M}_2\ddot{\mathbf{z}} + \mathbf{C}_2\dot{\mathbf{z}} + \mathbf{K}_2\mathbf{z} + \mathbf{N}(\mathbf{y} + \mathbf{z}) &= \mathbf{0}, \\
 \mathbf{Q}_\star &= \begin{bmatrix} \mathbf{0}_c & \mathbf{Q} & \mathbf{0}_2 \end{bmatrix}^T, \\
 \mathbf{Q} &\equiv \mathbf{M}_{\alpha\beta}\ddot{\mathbf{z}}_\beta + \mathbf{C}_{\alpha\beta}\dot{\mathbf{z}}_\beta + \mathbf{K}_{\alpha\beta}\mathbf{z}_\beta,
 \end{aligned} \tag{7.2}$$

where $\mathbf{w}$ is a mixed displacement vector can be defined as

$$\mathbf{w} \equiv \begin{bmatrix} \mathbf{x}_c & \mathbf{x}_\alpha & \mathbf{y}_\beta & \mathbf{y}_n \end{bmatrix}^T = \begin{bmatrix} \mathbf{x}_1 & \mathbf{y} \end{bmatrix}^T, \tag{7.3}$$

where $\mathbf{y} \in \mathbb{R}^{(N_1+N_2)}$ represents the response of the ideal system (i.e., in the absence of a deviatoric force), and $\mathbf{z} \in \mathbb{R}^{N_2}$ is the deviatoric response defined as $\mathbf{z} = \mathbf{x}_2 - \mathbf{y}$. Furthermore, the subscripts represent the following regions:

- $\mathbf{c}$: DOFs in $\mathcal{C}_1$ that are not coupled to the interface DOFs
- α : DOFs in $\mathcal{C}_1$ that are coupled to the interface DOFs
- β : DOFs in $\mathcal{C}_2$ that are coupled to the interface DOFs
- $\mathbf{n}$: DOFs in $\mathcal{C}_2$ that are not coupled to the interface DOFs

The advantage of this formulation is that the deviatoric force $\mathbf{Q}_\star$ is expressed as a function of $\mathbf{z}_\beta$ which is available at the boundary of the isolated region. This formulation avoids the need to have access to the interior of the isolated region $\mathcal{C}_2$ which may not be physically accessible (e.g., a joint). This work assumes that $\mathbf{x}_\beta$ is obtained by measuring the response at the boundary (e.g., with sensors) and that these measurements may be noisy.

7.1.2 Structure-Preserving Machine Learning

The structure-preserving ML formulation is based on the work presented in [184, 180]. We start by defining the kinetic energy $\mathcal{T}$, the potential energy $\mathcal{V}$, and the Rayleigh damping $\mathcal{J}$ terms of the known system as

$$\mathcal{T}(\dot{\mathbf{w}}) = \frac{1}{2} \dot{\mathbf{w}}^\top \mathbf{M} \dot{\mathbf{w}}, \quad (7.4)$$

$$\mathcal{V}(\mathbf{w}) = \frac{1}{2} \mathbf{w}^\top \mathbf{K} \mathbf{w}, \quad (7.5)$$

$$\mathcal{J}(\dot{\mathbf{w}}) = \frac{1}{2} \dot{\mathbf{w}}^\top \mathbf{C} \dot{\mathbf{w}}. \quad (7.6)$$

Following the definition of the Lagrangian, this is obtained as

$$\mathcal{L}(\mathbf{w}, \dot{\mathbf{w}}) = \mathcal{T}(\dot{\mathbf{w}}) - \mathcal{V}(\mathbf{w}). \quad (7.7)$$

from which the generalized momentum can be obtained as

$$p_k = \frac{\partial \mathcal{L}(\mathbf{w}, \dot{\mathbf{w}})}{\partial \dot{w}_k}. \quad (7.8)$$

We then used these quantities to compute the Hamiltonian as

$$H(\mathbf{w}, \dot{\mathbf{w}}) = \mathbf{p} \cdot \dot{\mathbf{w}} - \mathcal{L}(\mathbf{w}, \dot{\mathbf{w}}). \quad (7.9)$$

Next, the momentum rate (i.e., the time derivative of the generalized momentum) is required to relate the derived quantities back to quantities that can be measured (i.e., acceleration). The momentum rate is computed as

$$\dot{p}_k = -\frac{\partial H(\mathbf{w}, \dot{\mathbf{w}})}{\partial w_k} + Q_k^{\text{nc}} - \frac{\partial \mathcal{J}(\dot{\mathbf{w}})}{\partial \dot{w}_k}, \quad (7.10)$$

where Q_k^{nc} is the k^{th} term of the generalized force vector, and includes both the external forces and the deviatoric term,

$$\mathbf{Q}^{\text{nc}} = \mathbf{f}(t) + \mathbf{Q}_\star. \quad (7.11)$$

Following the work presented in [180], the deviatoric term is modeled with a multilayer perceptron (MLP). In this work, Swish activation functions are used in all hidden layers, and a linear activation is used for the output layer. The MLP is augmented with a dictionary of polynomial terms, which includes linear and higher order terms. Polynomial terms are chosen because they are a common way of modeling nonlinear forces in structural systems [120]. The

MLP takes in the boundary terms as input,

$$\begin{aligned} \mathbf{Q}_\star = & \mathcal{G}(\mathbf{z}_\beta, \dot{\mathbf{z}}_\beta) + \\ & \sum_{i,j,\dots,p}^{i+j+\dots,p=K} \lambda_{i,j,\dots,p} z_{\beta 1}^i z_{\beta 2}^j \dots z_{\beta k}^p + \\ & \gamma_{i,j,\dots,p} \dot{z}_{\beta 1}^i \dot{z}_{\beta 2}^j \dots \dot{z}_{\beta k}^p, \end{aligned} \quad (7.12)$$

where $\mathbf{z}_\beta$ corresponds to the difference between the measured and ideal response at the boundary of the isolated region. The coefficients λ and γ are learned during training, K is the polynomial degree chosen, and the function $\mathcal{G}(\cdot)$ represents the MLP. More details are provided in the original articles [184, 180]. A diagram illustrating the neural network architecture is shown in Fig. 7.2.

Once the vector of generalized forces, including the contribution from the deviatoric force term, is defined, the accelerations can be predicted as

$$\ddot{\mathbf{w}}_n = \mathbf{M}_{ni}^{-1}(\dot{\mathbf{p}}_i - \mathbf{M}_{ij,k} \dot{\mathbf{w}}_k \dot{\mathbf{w}}_j), \quad (7.13)$$

where $\mathbf{M}$ is the known mass matrix of the idealized system, and $\dot{\mathbf{p}}$ is the momenta rate (obtained from Eqn. (7.10)). As $\mathbf{w}$ is a vector of known quantities, the learning can be supervised by minimizing the squared error between the predicted and known accelerations,

$$\mathcal{L}_{\text{recon}} = \|\ddot{\mathbf{w}} - \ddot{\mathbf{w}}\|_2^2, \quad (7.14)$$

Once trained, the deviatoric force can be computed from the discrepancy term, $\mathbf{z}_\beta = \mathbf{x}_\beta - \mathbf{y}_\beta$, as

$$\mathbf{Q}_\star = \mathcal{N}(\mathbf{z}_\beta, \dot{\mathbf{z}}_\beta), \quad (7.15)$$

where $\mathcal{N}$ is a short-hand form for Eqn. (7.12), for any set of initial conditions or forcing function. It should be noted that the deviatoric force is not directly used for training. Instead, the residual

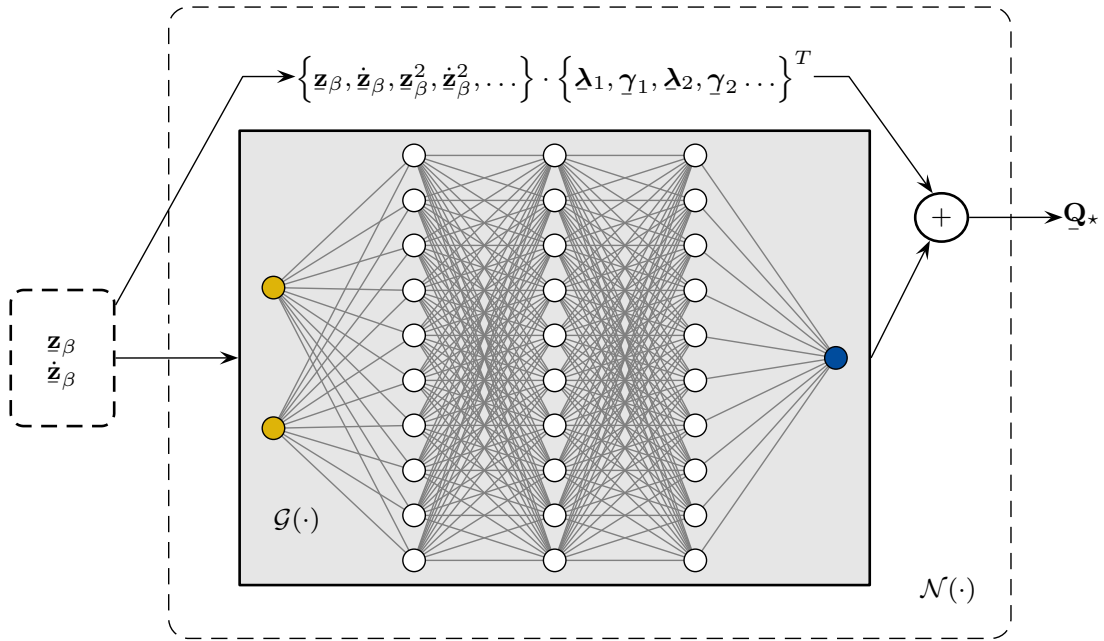


Figure 7.2. Neural network architecture used to model the deviatoric force term. The inputs to the network and to the polynomial terms are $\mathbf{z}_\beta$ and $\dot{\mathbf{z}}_\beta$. The outputs from the network and the polynomial terms are added together to obtain the deviatoric force.

of the equation of motion of the system is used to train the network. This formulation enforces preservation of the underlying geometric structure of the dynamic system.

7.1.3 Statistical Inference on Deviatoric Forces

In practice, the responses $\mathbf{z}$ are sampled from some non-bifurcated (unimodal) distribution that induces a distribution on $\mathbf{Q}_\star$, denoted as $\mathcal{D}(\mathbf{Q}_\star)$. Characterizing $\mathcal{D}(\mathbf{Q}_\star)$ allows for building prediction intervals on functions of samples from $\mathcal{D}(\mathbf{Q}_\star)$. If the distribution on $\mathbf{z}$ is known, it is possible to obtain the distribution of $\mathbf{Q}_\star$ through the transformation defined in Eqn. (7.15). Commonly, the distribution on $\mathbf{z}$ is unknown, leading to a nonparametric characterization of $\mathcal{D}(\mathbf{Q}_\star)$.

A common method for characterizing uncertainty under nonparametric assumptions is through the jackknife estimator. The jackknife estimator, f_{jack} , of a parameter, θ , is an average of functions of systematic subsamples of a collection of data of size n . Explicitly, for an estimator, f_n , the jackknife estimator is defined [268] as

$$f_{\text{jack}} = f_n - (n-1) \left(\frac{1}{n} \sum_{i=1}^n f_{(-i)} - f_n \right), \quad (7.16)$$

where $f_{(-i)}$ is the same computation of f_n but for the subsample with the i^{th} data point left out. The construction of Eqn. (7.16) reduces in-sample bias [268]. While f_n has a bias

$$\text{bias}(f_n) = \frac{a}{n} + \frac{b}{n^2} + \mathcal{O}\left(\frac{1}{n^3}\right), \quad (7.17)$$

the jackknife estimator, f_{jack} , has a reduced bias [268] of

$$\text{bias}(f_{\text{jack}}) = \frac{b}{(n-1)n} + \mathcal{O}\left(\frac{1}{n^2}\right), \quad (7.18)$$

for some $a, b \in \mathbb{R}$. While the bias reduction in Eqn. (7.18) from Eqn. (7.17) lends Eqn. (7.16) to be a desirable estimator.

The variance of the f_{jack} can be obtained through a jackknife estimate of its own by writing

$$\tilde{f}_i = nf_n - (n-1)f_{(-i)}. \quad (7.19)$$

As f_{jack} is the average of (7.19), the variance of f_n can be computed as

$$v_{\text{jack}} = \frac{1}{n(n-1)} \sum_{i=1}^n \left(\tilde{f}_i - \frac{1}{n} \sum_{j=1}^n \tilde{f}_j \right)^2. \quad (7.20)$$

Under stable conditions on f , (7.20) is a consistent estimator for the variance of f_{jack} [268].

Defining $f_n = \mathbf{Q}_\star$ in Eqn. (7.16), the estimation process follows as responses of the neural network in Eqn. (7.15). To this end, Eqn. (7.16) assumes that the true response of $\mathcal{N}(\mathbf{z}_\beta, \dot{\mathbf{z}}_\beta)$ is the parameter θ . For different training sets, or cross-validating data subsets, the value of θ is changing from sample to sample. This induces a machine-learning model bias into the estimation process. Therefore, a conformal prediction method is applied to capture-and-correct for this bias as the jackknife estimator similarly captures the bias in the traditional nonparametric setting [13].

The jackknife+ estimator is a conformal prediction method that extends the jackknife estimator to account for the machine learning model bias by minimizing the absolute deviation of each leave-one-out estimator, $f_{(-i)}$. As the variance of the jackknife estimator is computed using a sample of the jackknife pseudo-values (7.19), an estimation methodology such as local absolute deviations may be used to compute prediction sets. Given absolute deviations, $d_i = |f_i - f_{(-i)}|$, an estimator of f_{n+1} for a new observation, $\mathbf{x}_{n+1}$, can be computed as

$$f_{\text{jk}+}(\mathbf{x}_{n+1}) = \text{Median} \left(\left\{ f_{(-i)}(\mathbf{x}_{n+1}) \right\}_{i=1}^n \right). \quad (7.21)$$

To account for the model bias in Eqn. (7.21), local absolute deviations are calculated as

$$\begin{aligned} a_i &= f_{(-i)}(\mathbf{x}_{n+1}) - d_i \\ b_i &= f_{(-i)}(\mathbf{x}_{n+1}) + d_i, \end{aligned} \quad (7.22)$$

forming a collection of upper and lower bounds for each of the i models. A conformal prediction set is then given by

$$C_{n,\alpha}^{\text{jk}+} = [q_{n,\alpha/2}(\{a_i\}_{i=1}^n), q_{n,1-\alpha/2}(\{b_i\}_{i=1}^n)], \quad (7.23)$$

where $q_{n,\alpha}$ is the α -percentile of a sample of n observations [123]. The model bias is accounted for in Eqn. (7.23) through the local absolute deviations given in Eqn. (7.22). A jackknife+ prediction interval is then

$$P\left\{f_{n+1} \in C_{n,\alpha}^{\text{jk}+}\right\} \geq 1 - \alpha. \quad (7.24)$$

Recalling that setting $f_n = \mathbf{Q}_*$, Eqn. (7.24) yields a $100(1 - \alpha)\%$ conformal prediction interval for the deviatoric force. This general approach will be used to obtain estimates of the prediction intervals in the following section.

7.2 Results

The following subsections describe an example problem and demonstrate the proposed approach to define predictive intervals and detect shifts in the domain through hypothesis testing.

7.2.1 Example description

To demonstrate the proposed methodology a numerical example representing a one-dimensional rod with isolated nonlinearities is used. This example is described in detail in [180] but a general description is provided here. The rod is discretized with 64 masses that are connected through springs with mass $m = 1$ and linear stiffness $k = 1$. Proportional damping was defined as $\mathbf{C} \equiv \xi \mathbf{K}$. The nonlinear region $\mathcal{C}_2$ is located in the interval $(s_1, s_2) = (0.25, 0.35)$, so

that the isolated region exists between elements $\ell \equiv 16$ and $r \equiv 22$. The deviatoric force across the isolated region is identified as

$$\delta Q \equiv Q_{23} - Q_{15}, \quad (7.25)$$

and the displacement and velocity across the interval are defined as

$$\Delta \equiv x_{22} - x_{16}, \quad \dot{\Delta} \equiv \dot{x}_{22} - \dot{x}_{16}. \quad (7.26)$$

The initial conditions of the system are specified in terms of the modal displacements and velocities $(\mathbf{q}(0), \dot{\mathbf{q}}(0))$, where

$$\mathbf{x}(0) = \Phi \mathbf{q}(0), \quad \dot{\mathbf{x}}(0) = \Phi \dot{\mathbf{q}}(0), \quad (7.27)$$

where Φ is the modal transformation matrix.

The nonlinearities in the isolated region are assumed to contain linear mistuning and cubic nonlinearities in terms of both stiffness and damping. Finally, hysteretic nonlinear damping is also present between the elements within $\mathcal{C}_2$, so that the nonlinear force is represented as

$$\begin{aligned} f_{\text{nl},i} = & \kappa_i (x_{i+1} - x_i) + \chi_i (\dot{x}_{i+1} - \dot{x}_i) \\ & + \eta_i (x_{i+1} - x_i)^3 \\ & + \zeta_i (\dot{x}_{i+1} - \dot{x}_i)^3 + g_{\text{hys},i}, \end{aligned} \quad (7.28)$$

where the hysteretic force is defined as

$$g_{\text{hys},i} = k_{\text{h}} ((x_{i+1} - x_i) - r_i), \quad (7.29)$$

following the regularized formulation presented in [180].

Unless noted otherwise the parameters of the nonlinear region are chosen as

$$\begin{aligned}\kappa_i &= 0, \quad \chi_i = 0, \quad \eta_i = 2.621 \times 10^5, \\ \zeta_i &= 2.621 \times 10^3, \quad k_{h,i} = \rho \cdot 6.50 \times 10^{-1}, \\ \mu_i &= 1.00 \times 10^{-1}, \quad \varepsilon_i = 1.00 \times 10^{-2},\end{aligned}\tag{7.30}$$

with $\xi = 10^{-4}$, and ρ represents the level of hysteresis included. For the data generated for training the network, $\rho = 0.0$.

7.2.2 Noise model description

In some of the previous chapters, it had been assumed that the measurements were noise-free. In contrast, in this chapter, it is assumed that the measurements taken at the boundary ($\mathbf{x}_\beta$) are noisy. To this end, random Gaussian noise is added to the simulated data as

$$\mathbf{x}_\beta(t) = \bar{\mathbf{x}}_\beta(t) + \boldsymbol{\delta},\tag{7.31}$$

where $\bar{\mathbf{x}}_\beta(t)$ is the mean process and $\boldsymbol{\delta}$ is the random noise which is assumed to be independent and identically distributed (IID) modeled as

$$\boldsymbol{\delta} = \mathbf{v} \cdot \sqrt{\frac{\sum_i^N \bar{\mathbf{x}}_\beta(t_i)^2}{N-1}} \cdot \text{Normal}(0, 1),\tag{7.32}$$

where $\mathbf{v}$ is the noise factor considered which can also be interpreted as the reciprocal of the signal-to-noise ratio (SNR) since it represents the ratio between the noise variance and the signal variance.

As a result, the deviatoric response inherits the same additive noise as

$$\mathbf{z}_\beta(t) = \bar{\mathbf{z}}_\beta(t) + \boldsymbol{\delta},\tag{7.33}$$

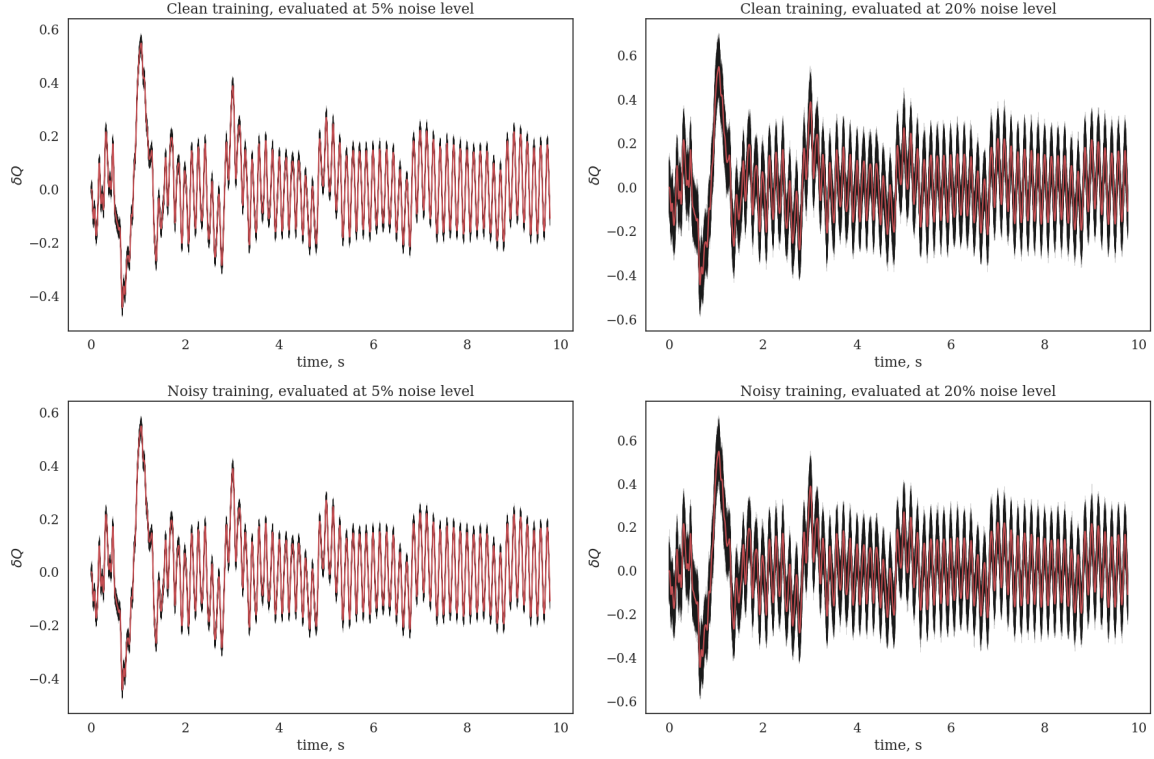


Figure 7.3. Predictions produced with neural network trained with noisy and clean data for input data with noise levels of 5% and 20%. Network predictions: black, actual: red.

and similarly for the deviatoric velocities, $\dot{\mathbf{z}}_{\beta}(t)$.

7.2.3 Effect of noise on model performance

The noise observed in the inputs is propagated through the trained neural network when evaluating the deviatoric force term. As a result, the outputs are noisy as well. As the neural network does not yield a closed-form transformation from the input distribution to the response distribution, we leverage the use of conformal sets to build an illustration of sensitivity which enables the definition of predictive confidence intervals. We start by assessing the network's performance when trained with noisy and clean (i.e., noise-free) data. To this end, the network was trained with one simulated realization using the rod model. Random noise was added to the response as described in Eqn. (7.31) with $\nu = 0.05$.

For the following results, the trained networks were evaluated with a different realization

(i.e., with different initial conditions than those used for training). The deviatoric force was modeled as described in Eqn. (7.28) where only linear terms were included in the polynomial terms (based on prior knowledge of how the interface displacements relate to the forces), and the MLP consisted of 5 hidden layers with Swish activation functions and the following number of units: $\{8, 12, 12, 10, 10\}$, which were determined based on a hyperparameter grid search. The output layer had a linear activation function. The model was trained using the Adam optimizer [124] with a learning rate of 1×10^{-4} for 65,000 epochs, which took around two hours to train on a single GPU. The neural network was implemented using the Jax [23] and Flax [91] packages. Fig. 7.3 illustrates the predicted deviatoric force across the interface when given noisy inputs. Both the network trained with noisy data and the network trained with clean data predict really similar outputs when confronted with noisy data during inference. This result is illustrated in Fig. 7.4 and 7.5. As shown, the distributions predicted by both networks are indistinguishable from each other which indicates that the network is robust to the presence of noise (at least additive white noise). This result may be due to the structure-preserving constraints in the network that have a regularization effect. It should be noted that the network trained with noisy data required twice the amount of training epochs to reach the same level of accuracy as the network trained with clean data.

To further evaluate the robustness of the network to noise, the network trained with clean data was used inside a time integration loop and random noise (at 1% level) was added at every integration step. Fig. 7.6 illustrates the effect of the noise on the integrated response. As shown, the mean response can still be recovered even when noise is added during integration. While the effect of error accumulation is evident in these plots, the response did not diverge for the time range that was considered.

7.2.4 Prediction intervals and damage detection

The next step is to define predictive intervals for the network using the conformal regression approach outlined in Section 7.1.3. To this end, the Python package MAPIE [159]

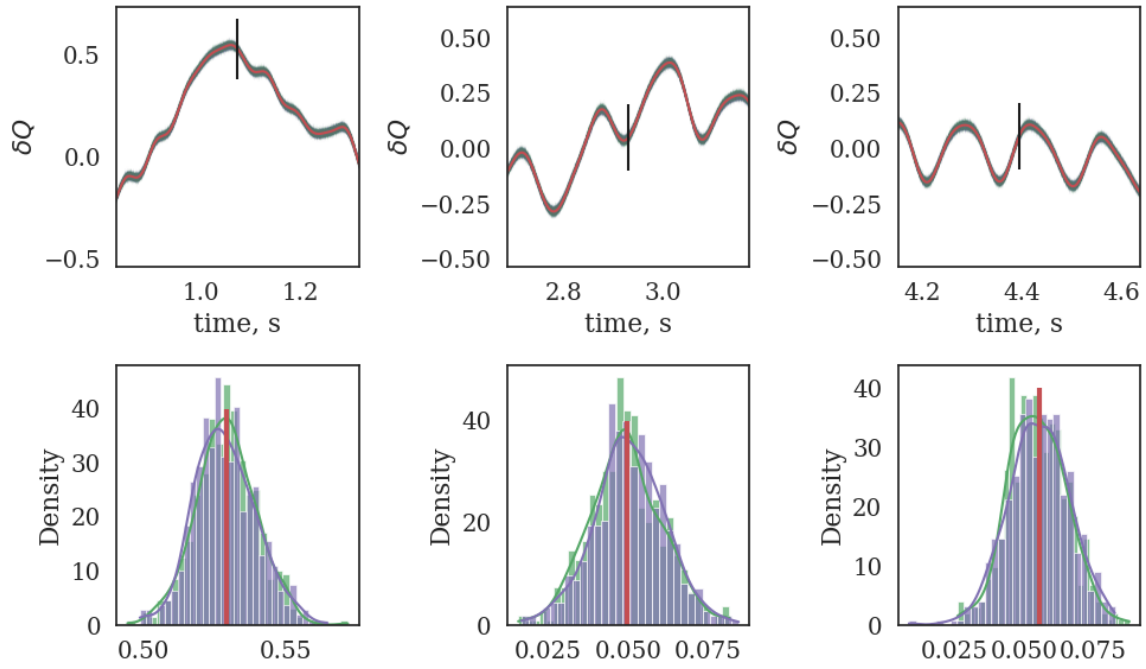


Figure 7.4. Close-up of predictions for noisy input data (5%) and corresponding histograms at three points in time. Clean: green, noisy: magenta, actual: red.

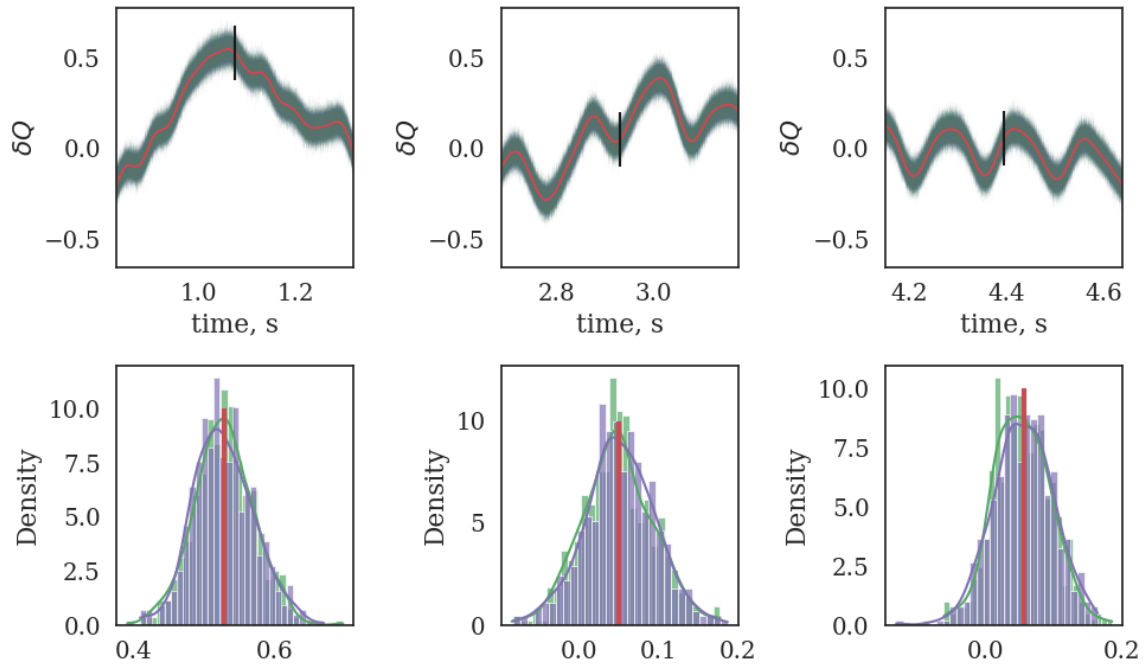


Figure 7.5. Close-up of predictions for noisy input data (20%) and corresponding histograms at three points in time. Clean: green, noisy: magenta, actual: red.

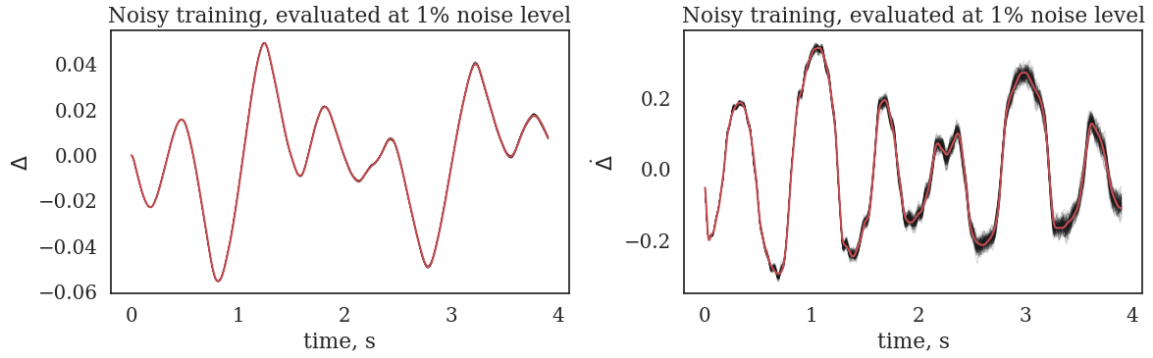


Figure 7.6. Response obtained from time integration with trained neural network in the loop. Network predictions: black, actual: red.

was used to define 95% confidence intervals for the trained models using the jackknife+ method. The intervals are obtained by fitting a conformal regression model to a "calibration" set but the results presented here are for a different realization (i.e., different initial conditions). Fig. 7.7, 7.8, and 7.9 illustrate the confidence intervals obtained for noise levels 5%, 10%, and 20%, respectively. As illustrated, the predicted confidence intervals provide reasonable coverage of the data (i.e., roughly 95% of the data is covered by the interval).

Now we test whether these predictive intervals can be informative in determining whether damage is present in the structure based on response measurements at the boundary of the isolated region. To simulate damage, the parameter ρ that controls the level of hysteresis in the structure is set to either 0, 1, 10, and 20. These levels define the baseline, low damage, medium damage, and high damage levels, respectively. These cases were simulated and compared to the predicted response from the neural network that had been trained with noisy data from the baseline model. This process was repeated for all three levels of damage and the three noise levels previously considered. For example, Fig. 7.10 corresponds to low level damage and 5% noise. In this case, it is evident that there has been a shift in the system response as the response of the damaged system (shown in red) is now offset from the mean of the distribution. However, when more noise is added, as illustrated in Fig. 7.11, this shift becomes less apparent. In contrast, a high level damage is apparent even in the presence of 20% noise, as shown in Fig. 7.12. These

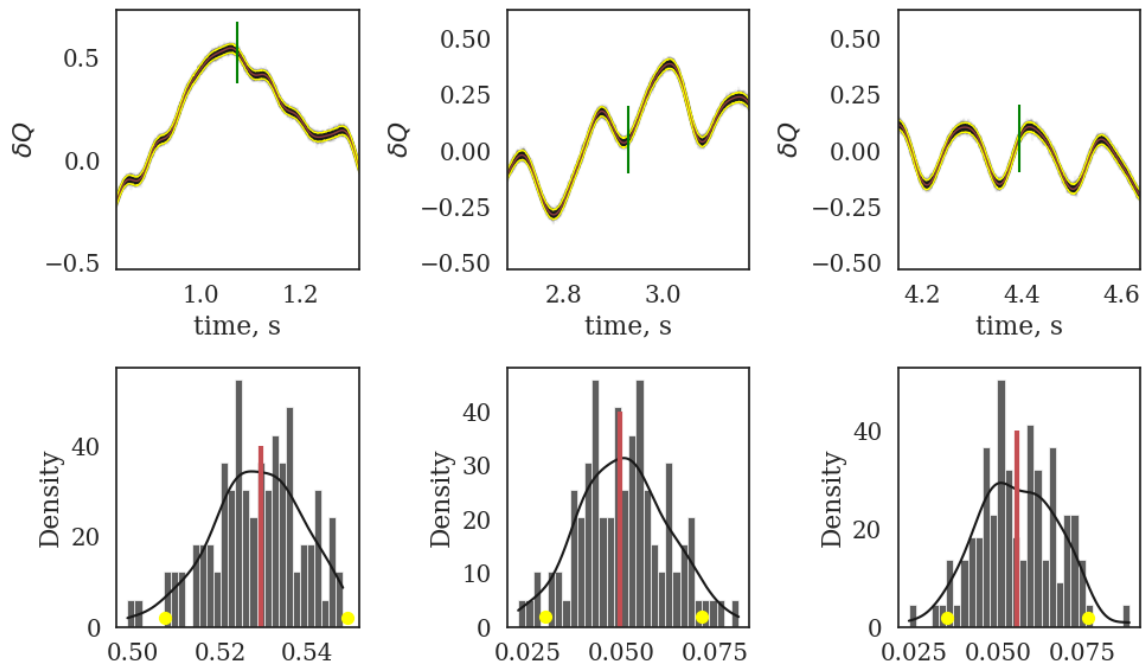


Figure 7.7. Deviatoric force predictions from noisy inputs (at 5% noise level) at three points in time alongside predictive confidence intervals. Actual response shown in red, 95% predictive confidence intervals shown in yellow. Green line indicates time point at which the histograms are computed.

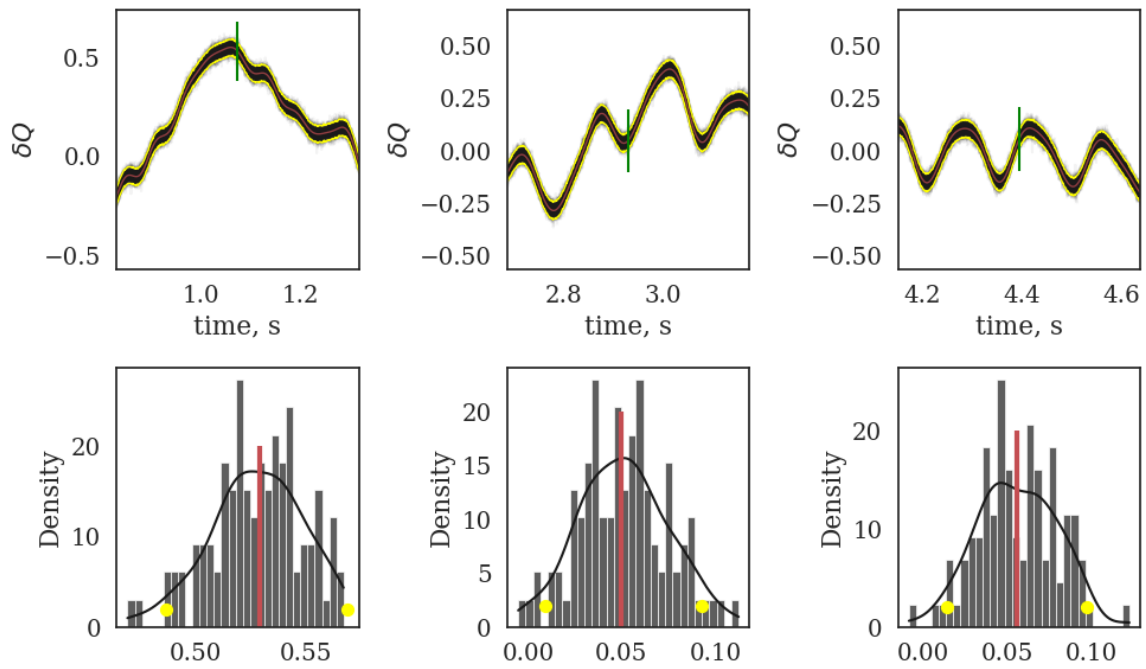


Figure 7.8. Deviatoric force predictions from noisy inputs (at 10% noise level) at three points in time alongside predictive confidence intervals. Actual response shown in red, 95% predictive confidence intervals shown in yellow. Green line indicates time point at which the histograms are computed.

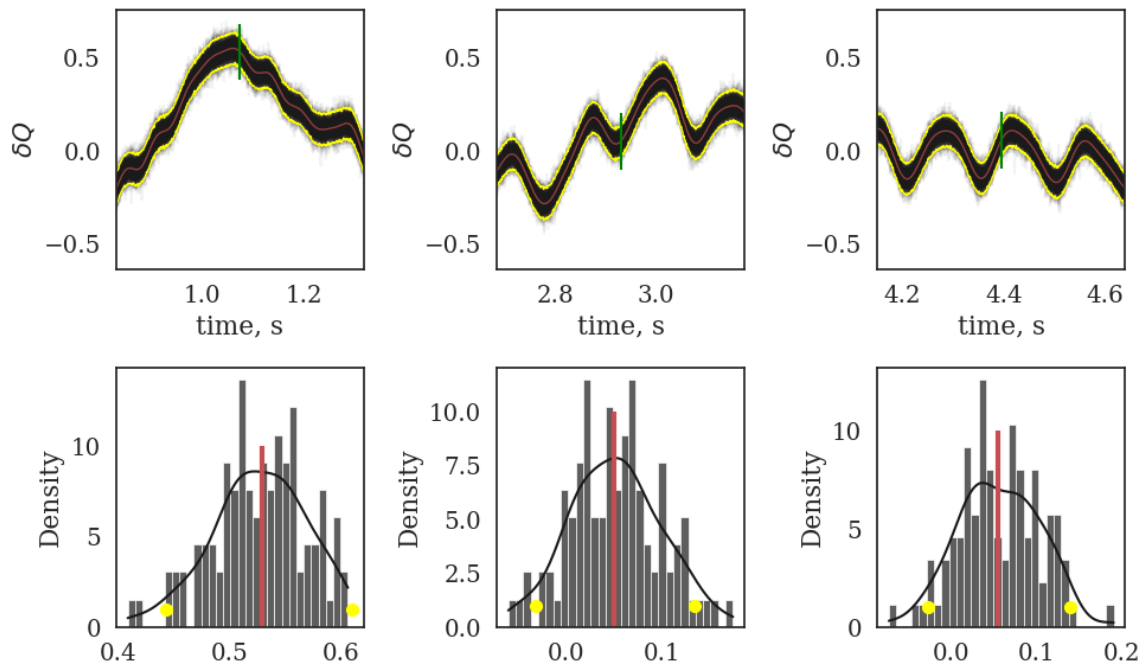


Figure 7.9. Deviatoric force predictions from noisy inputs (at 20% noise level) at three points in time alongside predictive confidence intervals. Actual response shown in red, 95% predictive confidence intervals shown in yellow. Green line indicates time point at which the histograms are computed.

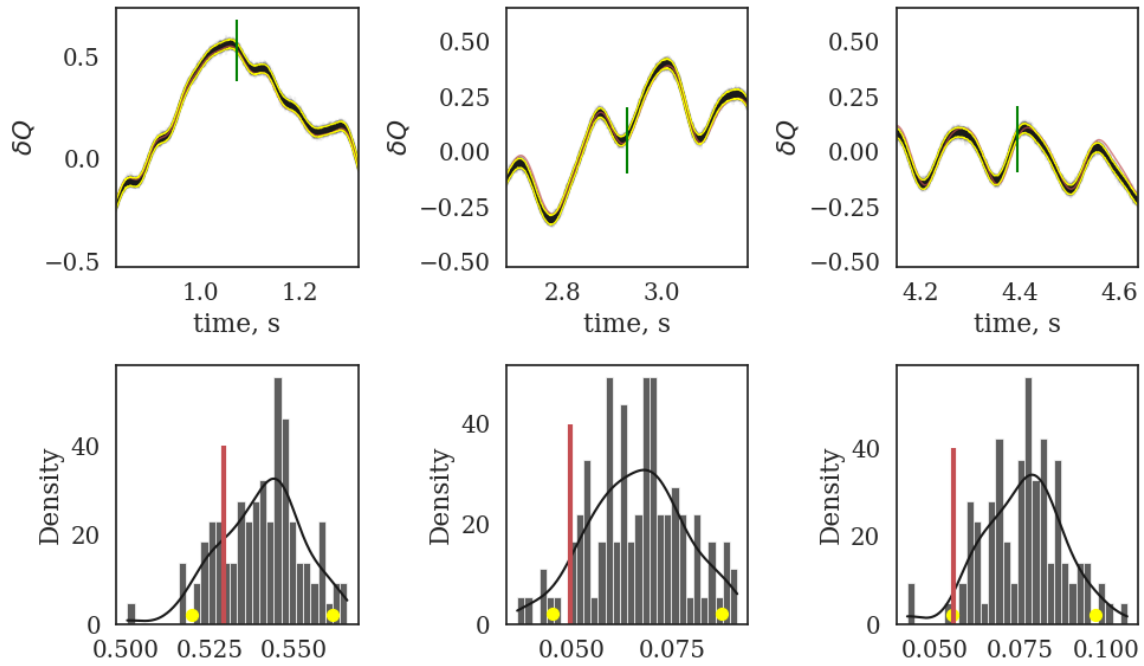


Figure 7.10. Deviatoric force predictions from noisy inputs (at 5% noise level) at three points in time alongside predictive confidence intervals. Low level damaged response shown in red, 95% predictive confidence intervals shown in yellow. Green line indicates time point at which the histograms are computed.

results illustrate the challenge of distinguishing meaningful domain shifts in the presence of noise.

The next step is to propagate these intervals through the dynamic process via time integration to obtain the response across the isolated region. For the following examples, a noise factor of 1% was used in order to avoid numerical convergence problems during time integration. For conciseness, only the velocity across the isolated region is shown. Fig. 7.13 shows the time histories of the velocity and their corresponding power spectral density (PSD). The three levels of damage (blue) are plotted alongside the baseline response (black). As shown, it is hard to distinguish between the cases from visual inspection which motivates a more quantitative assessment. Another way to inspect the data is to look at the histograms of the samples at different time points or frequency points. Fig. 7.14 illustrate the sample distributions at different points in time from the time histories, and Fig. 7.15 show the histograms at different frequencies

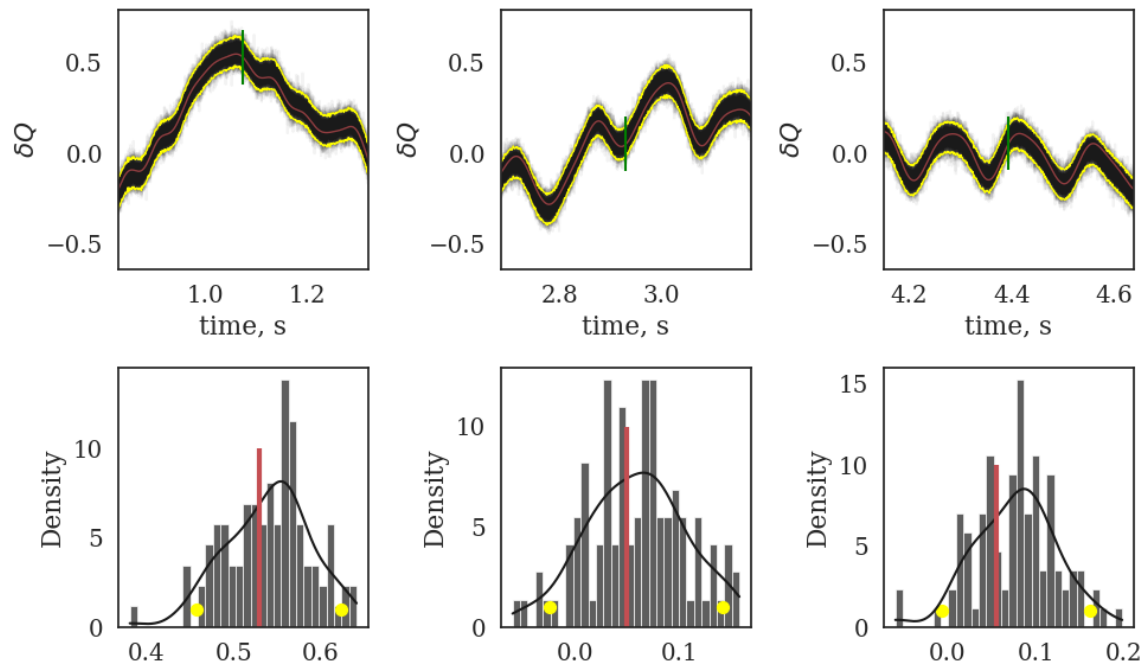


Figure 7.11. Deviatoric force predictions from noisy inputs (at 20% noise level) at three points in time alongside predictive confidence intervals. Low level damaged response shown in red, 95% predictive confidence intervals shown in yellow. Green line is time point at which the histograms are computed.

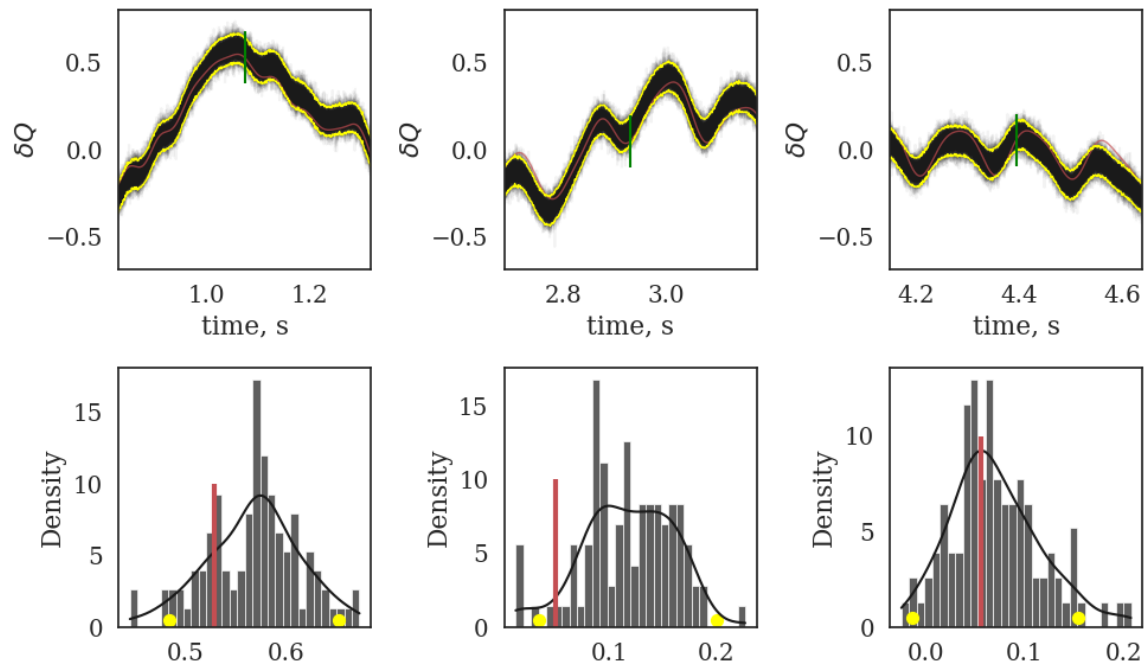


Figure 7.12. Deviatoric force predictions from noisy inputs (at 20% noise level) at three points in time alongside predictive confidence intervals. High level damaged response shown in red, 95% predictive confidence intervals shown in yellow. Green line is time point at which the histograms are computed.

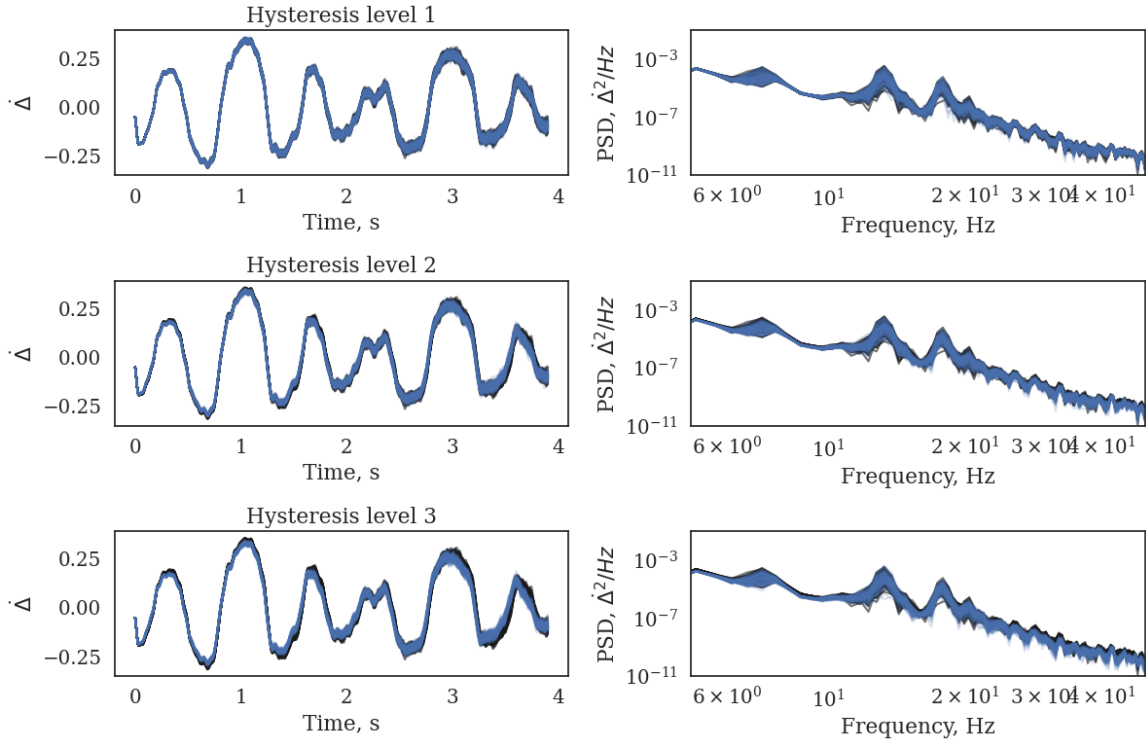


Figure 7.13. Velocity time history (left) and power spectral density (PSD) across isolated region boundary. The baseline response is shown in black and the response for the damaged cases is shown in blue.

from the PSDs. As shown, the distributions exhibit heteroscedastic behavior and, as a result, the trends observed in the histograms are not constant through time or frequency.

To further understand how the distributions vary through time and frequency, a Kolmogorov Smirnov (KS) two-sample test was performed to compare the damaged cases to the baseline using the implementation in the SciPy package [257]. In this test, the null hypothesis (H_0) is that the two samples (i.e., baseline and damaged) are from the same continuous distribution while the alternative hypothesis (H_1) is that they are from two different continuous distributions. The physical interpretation of failing to reject the null hypothesis is that there is no damage present (i.e., the two datasets are from a "healthy" baseline state). In this case we use a significance level of 5% which implies that p-values that are larger than 0.05 imply that the response is from the same distribution as the baseline state. In other words, p-values smaller than 0.05 will indicate the presence of damage. Fig. 7.16 illustrates the p-values obtained from the

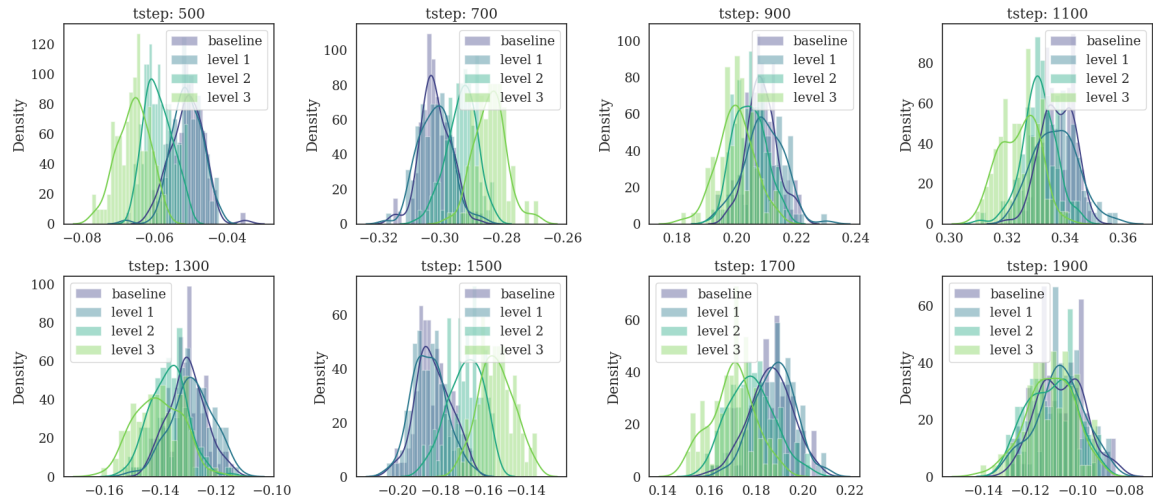


Figure 7.14. Histograms of velocity time history samples corresponding to different damage levels.

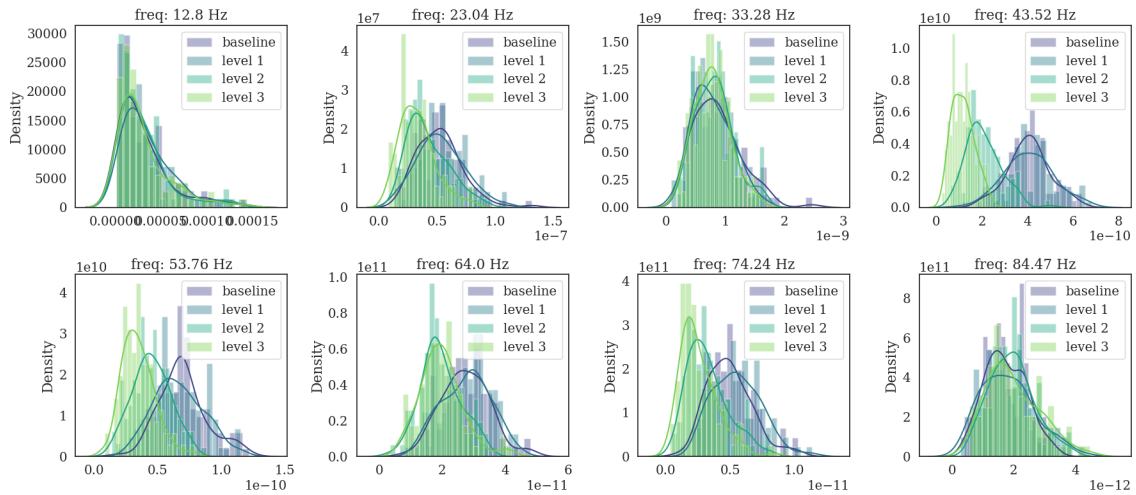


Figure 7.15. Histograms of velocity PSD samples corresponding to different damage levels.

KS test as a function of time (left) and frequency (right). As shown, the statistical significance of the test depends on the damage level which is not surprising. As shown in the histograms in Fig. 7.14 and 7.15, the shift in the empirical distributions is more evident as the damage level increases. Moreover the peaks in the p-values from the time history seem to correspond to zero-crossings where the differences in response may not be evident. On the frequency side, the test seems to be more effective at detecting the damaged cases at higher frequencies. This result implies that the type of damage modeled (i.e., hysteresis) may be having a more pronounced effect on the higher frequency content. The time history p-values are more consistent through time because the high-frequency response is distributed across time. Finally, to combine the two concepts, the Short Time Fourier Transform (STFT) was used to compute the change in frequency as a function of time and the KS test was performed at each frequency and time cell. These results are plotted in Fig. 7.17 where the upper limit of the color bar was set to $\log(0.05)$ so that yellow cells indicate that the null hypothesis was not rejected (i.e., no different from the baseline). These results seem consistent with the PSD and time history results as they show that the level 1 (low) damage is hard to detect with the KS test and that damage is not evident at the lower frequency response. A similar analysis was performed with the Cramer-von Mises (CVM) [50, 52] to verify that similar trends were observed. The p-value as a function of time and frequency is plotted in Fig. 7.18. The CVM test exhibits lower sensitivity to the domain shift in the STFT as illustrated by the fact that there are large yellow regions in Fig. 7.18 which indicates that the data were sampled from the same distribution. However, the number of cells for which the CVM rejects the null hypothesis (i.e., blue regions) increases as the level damage increases.

7.3 Conclusions

This chapter presented a framework for uncertainty quantification of neural network response predictions through the definition of predictive intervals using conformal regression.

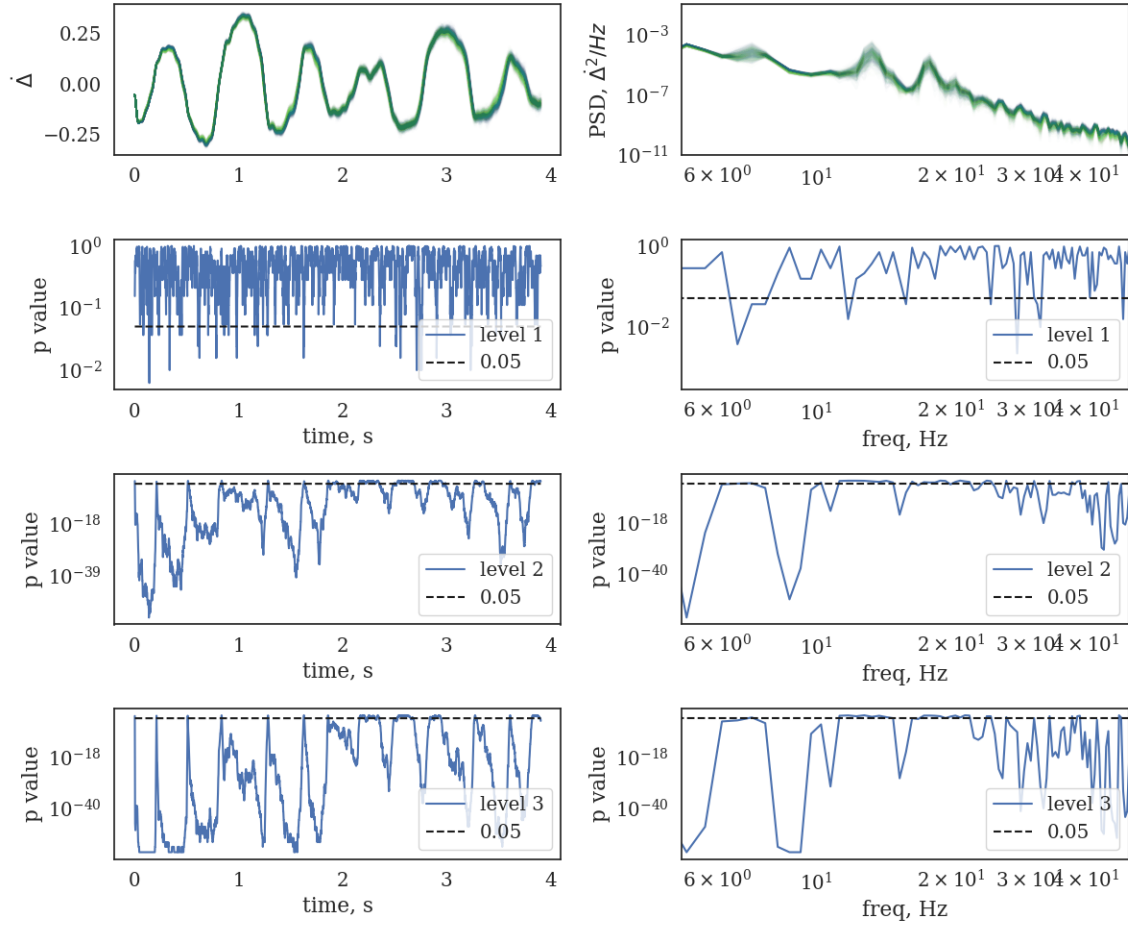


Figure 7.16. Velocity time histories (top left) and PSDs (top right) and p-values as a function of time (left) and frequency (right) with $p = 0.05$ shown with a dashed black line.

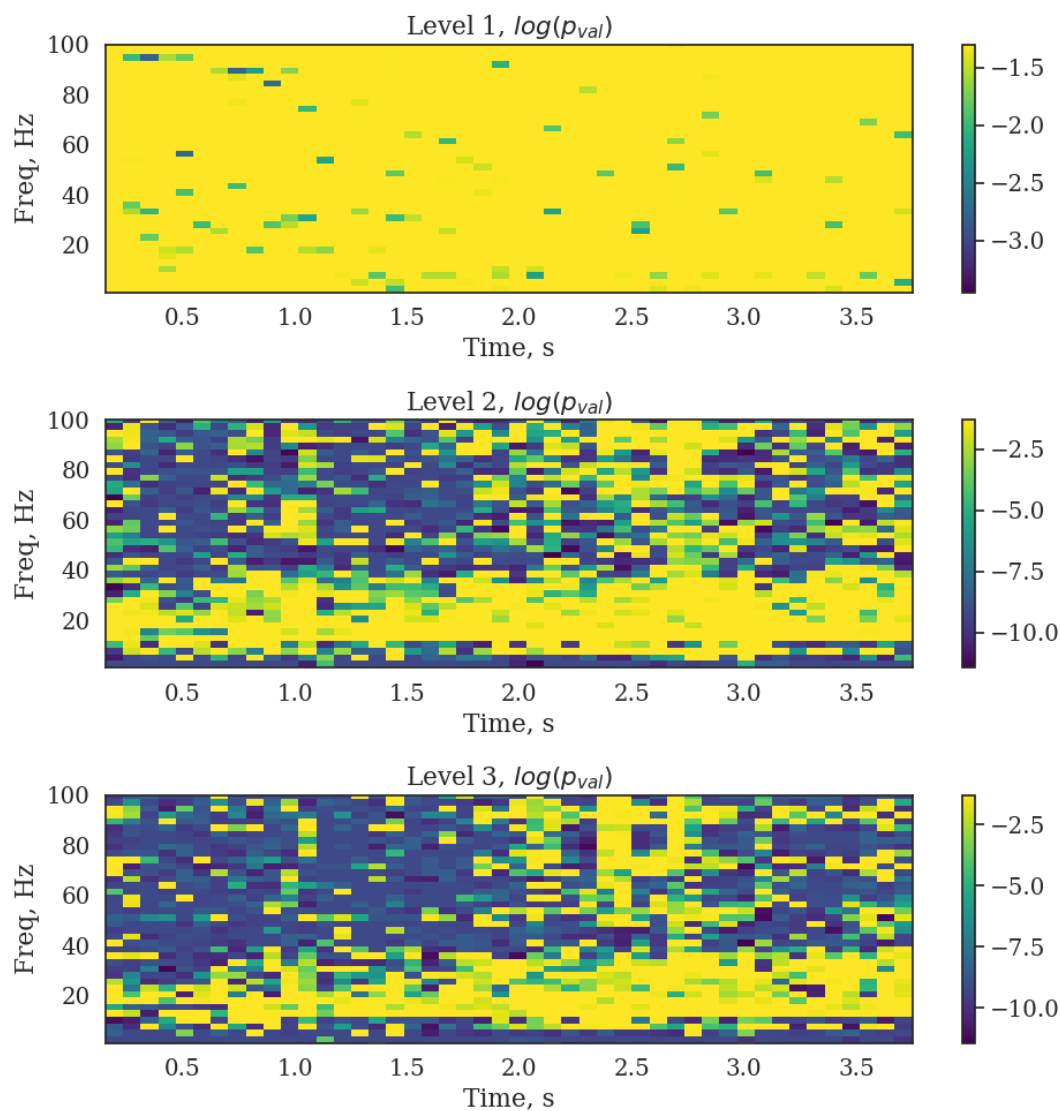


Figure 7.17. P-values as a function of time and frequency computed from STFT of data computed with the KS test. Upper color bar limit is set to $\log(0.05)$ so that anything that is yellow exceeds $p = 0.05$.

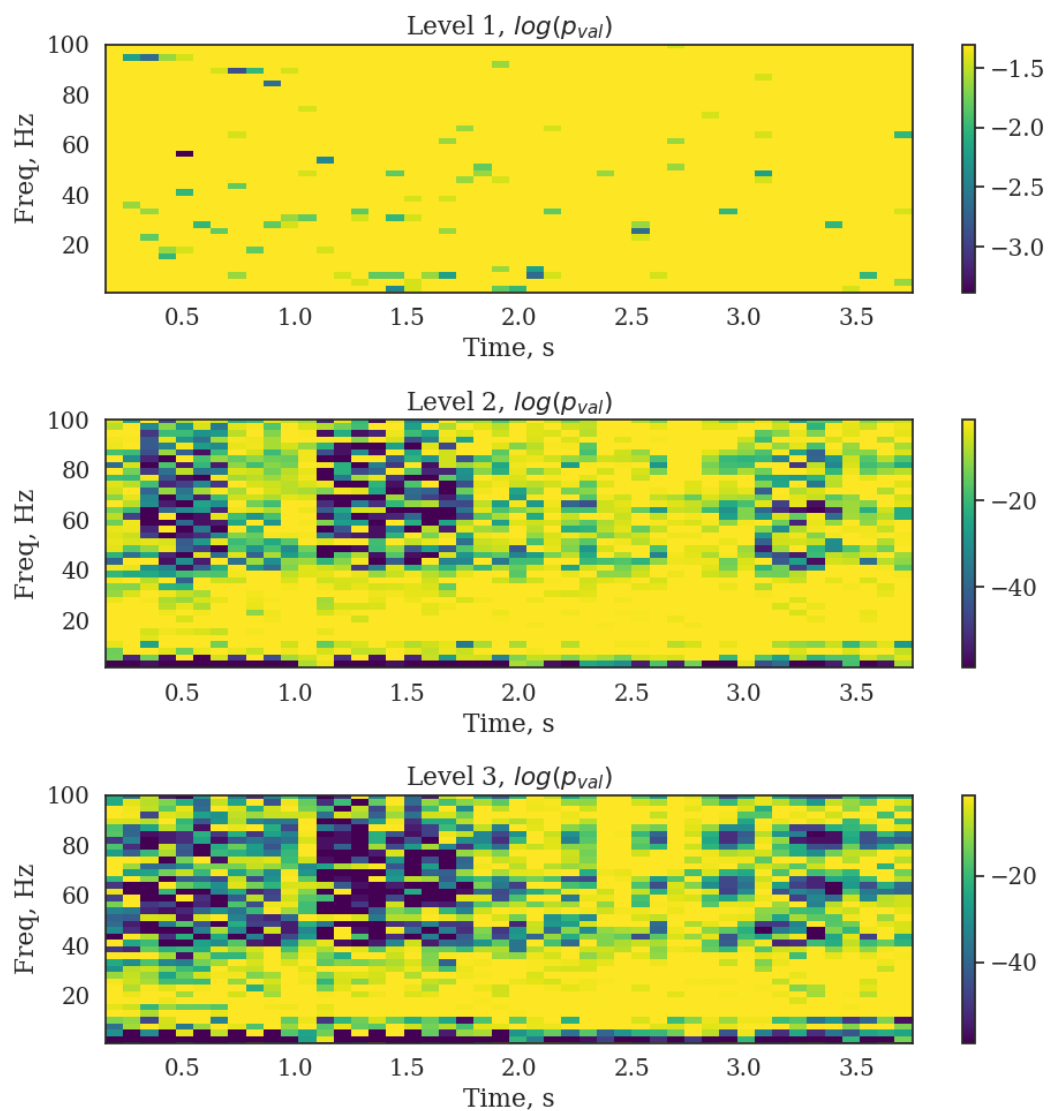


Figure 7.18. P-values as a function of time and frequency computed from STFT of data computed with the CVM test. Upper color bar limit is set to $\log(0.05)$ so that anything that is yellow exceeds $p = 0.05$.

The proposed approach was developed in the context of a structure-preserving neural network that has an embedded isolated nonlinearity formulation to provide physical constraints to the problem. As this model is intended to be used in an online structural health monitoring system, the performance of the models in the presence of measurement noise was evaluated. It was shown that the trained models were not significantly affected by noise in the inputs. This noise was propagated through the machine learning model and through the time integration of the dynamic system. Conformal regression was used to define a predictive interval of the integrated response that was used to assess the presence of damage (in the form of hysteresis) through hypothesis testing. Future work will explore other sources of uncertainty (such as epistemic uncertainty) and will apply the proposed approach to experimental cases.

7.4 Preview of Chapter 8

Chapter 7 introduced a strategy based on conformal predictions to handle observational noise present in measurement data. This chapter demonstrated the robustness of the structure-preserving digital twin to random noise in the data. However, epistemic uncertainty was not considered in Chapter 7. To this end, Chapter 8 introduces a framework for treatment of epistemic uncertainty in the ML system by embedding distance awareness in the feature space through a specialized neural network architecture. Chapter 8 demonstrates the ways in which the ML model can be used to detect domain shift systematically through binary hypothesis testing based on features extracted from an uncertainty-aware digital twin.

7.5 Remarks

Portions of this dissertation have been published or are being considered for publication. Chapter 7 is composed of the following first authored publications:

David A. Najera-Flores, Justin Jacobs, D. Dane Quinn, Anthony Garland, and Michael D. Todd. Uncertainty Quantification of a Machine Learning Model for Identification of Isolated

Nonlinearities With Conformal Prediction. *Journal of Verification, Validation and Uncertainty Quantification*, 9(2):021005, 06 2024

Chapter 8

Uncertainty-Aware Structure-Preserving Digital Twin for Domain Shift Detection from Nonlinear Dynamic Response of Structural Systems

A structure-preserving digital twin approach that leveraged the isolated formulation was introduced by the authors in [180] and presented in Chapter 5. That work was focused on learning a neural network ensemble embedded into the Euler-Lagrange formulation of the system to predict deviatoric forces from measurements at the boundary of the isolated nonlinear regions. The authors also addressed the challenge of observational uncertainty (e.g., from measurement noise) in [175] by employing conformal predictions to build prediction intervals based on noise observed in the inputs to the system. However, dealing with changes to the underlying physical system after the digital twin has been trained and deployed remains a challenge. Because the digital twin relies on continuous data streams, there may be a time when the measured inputs are inconsistent with the original physical system, making the digital twin predictions unreliable. This inconsistency between the data that were originally employed for the training of the model and new data coming in at a future time is known as “domain shift” (or distributional shift) [201]. The problem of domain shift is considered a major challenge in current AI/ML research [7], with many papers focused on developing domain adaptation strategies [238, 232, 140, 100, 220, 162] for robust ML model deployment. Most of these

methods are applied to purely data-driven approaches where physically interpretable models are not available. Although some researchers have attempted to extract physical interpretations from domain shift, such as changes in the underlying structural system [161], most of these methods rely on computing a shift metric one way or another, including a measure of uncertainty, such as Monte Carlo dropout [73, 161], to inform domain shift detection. Uncertainty-aware ML algorithms are gaining popularity primarily in health science applications where reliability and trustworthiness is critical [150, 160, 55] but according to [150], most of the research in this area employs techniques meant to quantify model epistemic uncertainty. A recent algorithm proposed in [144] known as the spectral-normalized neural Gaussian process (SNGP) introduces a simple way of ensuring distance preservation and awareness in the hidden feature space. SNGPs combine the expressive power of neural networks with the distance-aware properties of Gaussian processes while incorporating distance preservation constraints. SNGPs have been shown to be an effective way of performing uncertainty quantification in SHM [185]. However, the applications thus far have been focused on purely data-driven methods.

In this work, we propose to integrate the concepts of SNGP into the existing structure-preserving neural network framework [180] to detect domain shifts observable in the system response due to underlying structural changes. Furthermore, we propose a workflow for detection based on binary hypothesis testing of features extracted from the statistical distributions of prediction variance over a fixed time window. The proposed approach combines physics-constrained, structure-preserving ML with a distance-preserving, uncertainty-aware formulation to provide a way to detect domain shift when the digital twin is confronted with new data. The objective of this work is to put forward a method for systematic domain shift detection from measurable dynamic responses to inform the possibility of damage or unexpected changes in the underlying structural system, and prevent SHM systems from relying on an outdated digital twin. The proposed approach is demonstrated on a computational example with relevant structural nonlinearities. It is noted that while the proposed workflow is general, specific details such as the choice of features and specific hypothesis testing strategy may be problem-dependent.

The chapter is organized as follows. Section 8.1 presents a summary of previous work and the enhancements to existing methods in this chapter. Section 8.2 illustrates the application of the proposed approach on a numerical example while Section 8.3 provides conclusions and discussion of future work opportunities.

8.1 Background and Methodology

This chapter builds upon a structure-preserving machine learning method developed by the authors in [180] and presented in Chapter 5. The theoretical background for the isolated formulation is briefly summarized in 8.1.1. Section 8.1.2 described the structure-preserving formulation. Section 8.1.3 presents the theoretical and implementation background for SNGP, and Section 8.1.4 describe the statistical techniques employed for domain shift detection.

8.1.1 Isolated Nonlinearity Formulation

This work considers general structures (linear or nonlinear) containing isolated regions in which nonlinearities may be present. Examples of isolated nonlinearities include joints, nonlinear attachments, or localized damage. Following the work in [199, 180, 258], the structure is decomposed into two regions $\mathcal{C}_1$ and $\mathcal{C}_2$, where the equations of motion within $\mathcal{C}_1$ are assumed to be linear and known. In contrast the nonlinearities within the system are localized to $\mathcal{C}_2$. Note that these regions need not be continuous, allowing for the possibility of multiple isolated domains described by $\mathcal{C}_2$. Region $\mathcal{C}_1$ is described by the internal variables $\mathbf{x}_1 \in \mathbb{R}^{N_1}$, while $\mathbf{x}_2 \in \mathbb{R}^{N_2}$ are the corresponding internal variables for $\mathcal{C}_2$. As a result, the equations of motion for this structure can be written as

$$\begin{aligned} \mathbf{M}\ddot{\mathbf{x}} + \mathbf{C}\dot{\mathbf{x}} + \mathbf{K}\mathbf{x} + \mathbf{N}_\star &= \mathbf{f}(t), \\ \mathbf{N}_\star &= [\mathbf{0} \quad \mathbf{N}(\mathbf{x}_2)], \quad \mathbf{N} \in \mathbb{R}^{N_2}, \\ \mathbf{x} &= [\mathbf{x}_1 \quad \mathbf{x}_2] \in \mathbb{R}^{(N_1+N_2)}. \end{aligned} \tag{8.1}$$

where $\mathbf{N} \in \mathbb{R}^{N_2}$ represents the vector for isolated nonlinear forces, $\mathbf{M}, \mathbf{K}, \mathbf{C}$ represent the known mass, stiffness, and damping matrices, respectively. Furthermore, the equation of motion can be decomposed as

$$\begin{aligned}\mathbf{M}\ddot{\mathbf{w}} + \mathbf{C}\dot{\mathbf{w}} + \mathbf{K}\mathbf{w} &= \mathbf{f}(t) - \mathbf{Q}_\star, \\ \mathbf{M}_2\ddot{\mathbf{z}} + \mathbf{C}_2\dot{\mathbf{z}} + \mathbf{K}_2\mathbf{z} + \mathbf{N}(\mathbf{y} + \mathbf{z}) &= \mathbf{0}, \\ \mathbf{Q}_\star &= \begin{bmatrix} \mathbf{0}_c & \mathbf{Q} & \mathbf{0}_2 \end{bmatrix}^T, \\ \mathbf{Q} &\equiv \mathbf{M}_{\alpha\beta}\ddot{\mathbf{z}}_\beta + \mathbf{C}_{\alpha\beta}\dot{\mathbf{z}}_\beta + \mathbf{K}_{\alpha\beta}\mathbf{z}_\beta.\end{aligned}\tag{8.2}$$

In this reformulation of the system, $\mathbf{w} \in \mathbb{R}^{(N_1+N_2)}$ is defined as a mixed displacement vector

$$\mathbf{w} \equiv \begin{bmatrix} \mathbf{x}_1 & \mathbf{y} \end{bmatrix}^T = \begin{bmatrix} \mathbf{x}_c & \mathbf{x}_\alpha & \mathbf{y}_\beta & \mathbf{y}_n \end{bmatrix}^T, \tag{8.3}$$

combining $\mathbf{x}_1 \in \mathbb{R}^{N_1}$, the *exact* response of the system in $\mathcal{C}_1$, with $\mathbf{y} \in \mathbb{R}^{N_2}$, the response of the ideal system, that is, the system in the absence of the nonlinearities, but subject to the force $\mathbf{Q}$ applied at the boundary of the two regions, and referred to as the deviatoric force. Finally, within $\mathcal{C}_2$ the deviatoric response is defined as $\mathbf{z} \equiv \mathbf{x}_2 - \mathbf{y} \in \mathbb{R}^{N_2}$. These coordinates are further decomposed, so that the subscript c represents the DOFs in $\mathcal{C}_1$ that are not coupled to the interface DOFs, while α represents the DOFs in $\mathcal{C}_1$ that are coupled to the interface. Likewise β represents the DOFs in $\mathcal{C}_2$ that are coupled to the interface (i.e., the only DOFs in $\mathcal{C}_2$ that are observable), and n represents the DOFs in $\mathcal{C}_2$ that are not coupled to the interface (i.e., not observable).

The advantage of this formulation is that the deviatoric force $\mathbf{Q}$ is expressed as a function of $\mathbf{z}_\beta$ which is defined at the boundary of the isolated region, and recovers the *exact* solution in the region $\mathcal{C}_1$. When directly simulating Eqn. (8.2) the influence of the nonlinearities are within $\mathbf{z} \in \mathbb{R}^{N_2}$, while $\mathbf{w}$ is determined from a linear equation, subject to $\mathbf{Q}$. More generally, if the deviatoric force $\mathbf{Q}$ can be determined (e.g. through ML as is done below), then formulation avoids

the need to have access to the interior of the isolated region $\mathcal{C}_2$ which may not be physically accessible (e.g., a joint). This work assumes that $\mathbf{x}_\beta$ is obtained by measuring the response at the boundary (e.g., with sensors).

8.1.2 Structure-Preserving Machine Learning

The foundation for the structure-preserving ML formulation has been presented in previous work by the authors [184, 180, 175]. In that work, the problem is formulated in terms of the Euler-Lagrange equations of motion. To compute the Lagrangian, the kinetic energy $\mathcal{T}$, the potential energy $\mathcal{V}$, and the Rayleigh damping $\mathcal{J}$ terms of the known system are obtained as

$$T(\dot{\mathbf{w}}) = \frac{1}{2} \dot{\mathbf{w}}^\top \mathbf{M} \dot{\mathbf{w}}, \quad (8.4)$$

$$V(\mathbf{w}) = \frac{1}{2} \mathbf{w}^\top \mathbf{K} \mathbf{w}, \quad (8.5)$$

$$\mathcal{J}(\dot{\mathbf{w}}) = \frac{1}{2} \dot{\mathbf{w}}^\top \mathbf{C} \dot{\mathbf{w}}. \quad (8.6)$$

The Lagrangian is obtained by simply taking the difference of the kinetic and potential energy as

$$\mathcal{L}(\mathbf{w}, \dot{\mathbf{w}}) = T(\dot{\mathbf{w}}) - V(\mathbf{w}). \quad (8.7)$$

from which the generalized momentum can be obtained as

$$p_k = \frac{\partial \mathcal{L}(\mathbf{w}, \dot{\mathbf{w}})}{\partial \dot{w}_k}. \quad (8.8)$$

We then used these quantities to compute the Hamiltonian as

$$H(\mathbf{w}, \dot{\mathbf{w}}) = \mathbf{p} \cdot \dot{\mathbf{w}} - \mathcal{L}(\mathbf{w}, \dot{\mathbf{w}}). \quad (8.9)$$

The Hamiltonian is introduced here in order to relate these quantities to a measurable state such as acceleration. To this end, the momentum rate (i.e., the time derivative of the generalized

momentum) is also computed as

$$\dot{p}_k = Q_k^{\text{nc}} - \frac{\partial H(\mathbf{w}, \dot{\mathbf{w}})}{\partial w_k} - \frac{\partial \mathcal{J}(\dot{\mathbf{w}})}{\partial \dot{w}_k}, \quad (8.10)$$

where Q_k^{nc} is the k^{th} term of the generalized force vector, and includes both the external forces and the deviatoric term,

$$\mathbf{Q}^{\text{nc}} = \mathbf{f}(t) + \mathbf{Q}_\star. \quad (8.11)$$

In contrast to the work presented in [180], in this work the deviatoric force is derived from a nonlinear potential function $V_{\text{nl}}(\cdot)$ and a nonlinear damping function $\mathcal{J}_{\text{nl}}(\cdot)$. This formulation is an extension of the work previously presented in [184, 225] where neural networks have been used to learn scalar fields corresponding to the nonlinear potential and nonlinear damping terms. It should be noted that in this isolated formulation, these are functions of the boundary degrees-of-freedom as described in Section 8.1.1 which are assumed to be measurable and therefore do not represent the nonlinearity directly in terms of the internal state variables. Therefore, the deviatoric forces are obtained as

$$\mathbf{Q}_\star = \frac{\partial V_{\text{nl}}(\mathbf{z}_\beta)}{\partial \mathbf{z}_\beta} + \frac{\partial \mathcal{J}_{\text{nl}}(\dot{\mathbf{z}}_\beta)}{\partial \dot{\mathbf{z}}_\beta}, \quad (8.12)$$

where $\mathbf{z}_\beta$ and $\dot{\mathbf{z}}_\beta$ corresponds to the difference between the measured and ideal displacements and velocities at the boundary of the isolated region, and $V_{\text{nl}}(\mathbf{z}_\beta)$, $\mathcal{J}_{\text{nl}}(\dot{\mathbf{z}}_\beta)$ represent a nonlinear potential and a nonlinear damping function that will be modeled with neural networks (as described in Section 8.1.3). A diagram illustrating the general workflow is shown in Fig. 8.1.

With the nonlinear forces now computed, the accelerations can then be predicted from the equation of motion as

$$\hat{\dot{w}}_n = M_{ni}^{-1}(\dot{p}_i - M_{ij,k}\dot{w}_k\dot{w}_j), \quad (8.13)$$

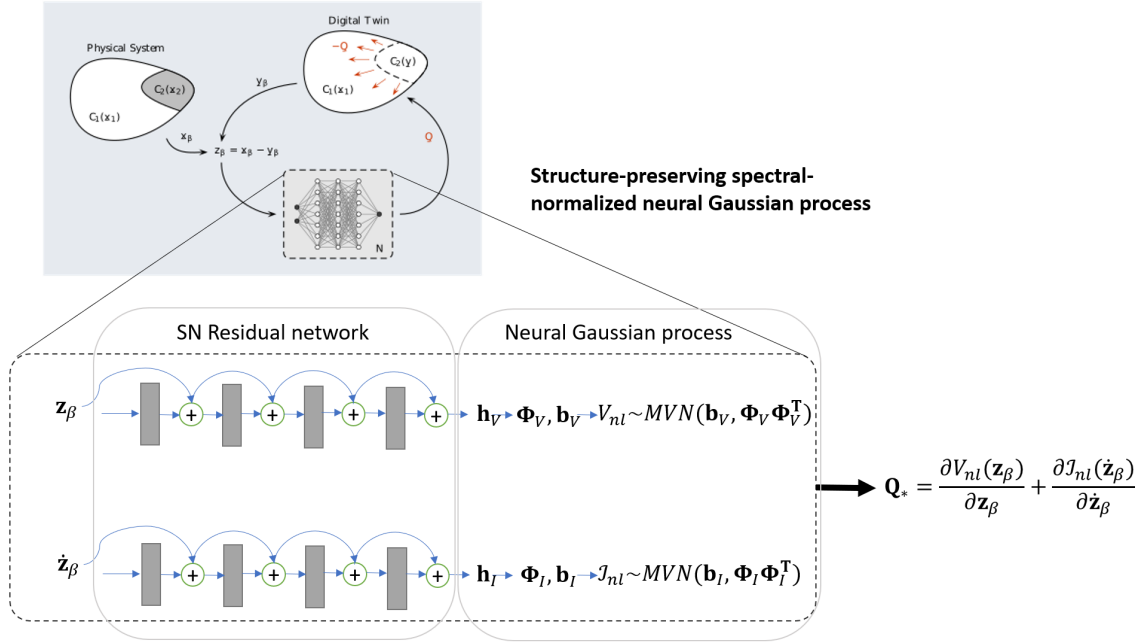


Figure 8.1. Neural network architecture used to model the deviatoric force term. The inputs to the network and to the polynomial terms are $\mathbf{z}_\beta$ and $\dot{\mathbf{z}}_\beta$. The outputs from the network and the polynomial terms are added together to obtain the deviatoric force.

where $\mathbf{M}$ is the known mass matrix of the idealized system, and $\dot{\mathbf{p}}$ is the momenta rate (obtained from Eqn. (8.10)). As $\mathbf{w}$ is a vector of known quantities, the learning can be supervised by minimizing the squared error between the predicted and known accelerations,

$$\mathcal{L}_{\text{recon}} = \|\hat{\mathbf{w}} - \ddot{\mathbf{w}}\|_2^2, \quad (8.14)$$

Once trained, the deviatoric force can be computed from the discrepancy term, $\mathbf{z}_\beta = \mathbf{x}_\beta - \mathbf{y}_\beta$, as

$$\mathbf{Q}_* = \mathcal{N}(\mathbf{z}_\beta, \dot{\mathbf{z}}_\beta), \quad (8.15)$$

where $\mathcal{N}$ is a short-hand form for Eqn. (8.12), for any set of initial conditions or forcing function. As shown, the prediction of the deviatoric force involves an input that is measured at the boundary ($\mathbf{x}_\beta$) and a predicted quantity ($\mathbf{y}_\beta$) so this is how the influence from measured data is propagated into the prediction. As illustrated in Fig. 8.1, the response prediction occurs recursively by

feeding back the network output at every time step. It should be noted that the deviatoric force is assumed to be unknown and therefore not directly used for training. Instead, the residual of the equation of motion of the system (Eqn. 8.13) is used to train the network in an unsupervised manner. This formulation enforces preservation of the underlying geometric structure of the dynamic system.

8.1.3 Spectral-Normalized Neural Gaussian Process

Following the work in [144], the neural network architectures used in this work consist of residual connections with spectral-normalized weights. This choice of architecture ensures distance-preservation as described in [144] by enforcing the lower bound of the Lipschitz constant through its residual connections, and the upper bound by using spectral normalization. This construction represents a bi-Lipschitz constraint that prevents disproportional contraction (also known as feature collapse) or expansion of the input space when it is transformed to the hidden space [144]. There are two separate networks and they both follow the exact same architecture. The input to the network is either $\mathbf{z}_\beta$ or $\dot{\mathbf{z}}_\beta$, which are N_β -dimensional vectors, where N_β is the number of degrees-of-freedom in the boundary of the isolated region. The input layer maps the input to a the hidden space of dimension d with a linear dense layer. The next L hidden layers follow a residual architecture where the input is added to the output after each dense layer, as illustrated in Fig. 8.1. A swish activation function is used at each dense layer. The output layer uses a Gaussian process (GP) with kernel matrix, $\mathbf{K}_{\text{GP}}$. The hidden representation, $\mathbf{h}_i \in \mathbb{R}^{d_{L-1} \times 1}$, of the previous penultimate layer of dimension d_{L-1} serves as input to the GP layer.

As the output Gaussian process is intractable, a low-rank approximation of the kernel matrix is applied via

$$\mathbf{K}_{\text{GP}} = \Phi\Phi^T, \quad (8.16)$$

where

$$\Phi_i = \sqrt{2\sigma^2/d_L} \cos(-\mathbf{W}_L \mathbf{h}_i + \mathbf{b}_L), \quad (8.17)$$

are random features that form a random sampling of a square-integrable basis [202] for the output layer. Here $\mathbf{W}_L \in \mathbb{R}^{d_L \times d_L}$ is a fixed weight matrix with entries sampled i.i.d. from a standard normal distribution, $\mathbf{b}_L \in \mathbb{R}^{d_L \times 1}$ is a fixed bias vector with entries sampled i.i.d. from a uniform distribution in the range $(0, 2\pi)$, and σ^2 is the GP kernel variance and is set to 1.0.

The output of the SNGP is then obtained as

$$g(h_i) = \Phi_i^T \mathbf{u} + c, \quad (8.18)$$

where $\mathbf{u} \in \mathbb{R}^{d_L \times 1}$ is the vector of learnable weights in the last layer and is initialized by randomly sampling from a zero-mean normal distribution with variance $\tau^2 = 3.0$ (tuned manually for numerical stability) and $c \in \mathbb{R}^{1 \times 1}$ is the learnable constant bias. The output $g(h_i) \in \mathbb{R}^{1 \times 1}$ is a scalar field corresponding to either the nonlinear potential $V_{\text{nl}}(\mathbf{z}_i)$ or nonlinear damping $\mathcal{J}_{\text{nl}}(\dot{\mathbf{z}}_i)$. As described in [144], this formulation results in the following random-feature GPs

$$\begin{aligned} V_{\text{nl}}(\mathbf{z}) &= g_V(\mathbf{h}_V) \sim N(c_V, \Phi_V^T \Phi_V) \\ \mathcal{J}_{\text{nl}}(\dot{\mathbf{z}}) &= g_I(\mathbf{h}_I) \sim N(c_I, \Phi_I^T \Phi_I), \end{aligned} \quad (8.19)$$

where $N(\cdot)$ represents the normal distribution and the subscripts V, I are associated with the potential and damping function parameters, respectively. The covariance or GP kernel (Eqn. 8.16) is computed across all training examples, such that

$$\mathbf{K}_{\text{GP},ij} = \exp(-\|\mathbf{h}_i - \mathbf{h}_j\|_2^2/2), \quad (8.20)$$

or expressed in words, the uncertainty is based on the distance of the hidden features, $\mathbf{h}$, which are a nonlinear transformation of the input state space. Given the distance-preservation constraints,

the measure of uncertainty computed by the SNGP is then related to the distance in the state space manifold.

$$\min_{\Psi_V, \Psi_I} \mathcal{J}(\Psi_V, \Psi_I) \quad (8.21)$$

where Ψ_V and Ψ_I are the collection of neural network weights corresponding to the nonlinear potential and nonlinear damping networks, respectively, and $\mathcal{J}(\Psi_V, \Psi_I)$ is the total loss function composed of the reconstruction and regularization terms

$$\mathcal{J}(\Psi_V, \Psi_I) = \mathcal{L}_{\text{recon}} + \lambda_V \|\Psi_V\|_2^2 + \lambda_I \|\Psi_I\|_2^2, \quad (8.22)$$

where λ_V and λ_I are weights used to control the regularization effect. In this work both were set to 10^{-4} . The minimization problem is solved using the Adam optimizer with a learning rate set to 10^{-4} . Other hyperparameters used in the networks include: 4 hidden layers with 16 units each, and a Swish activation function for all of them.

To obtain the prediction variance, we need to construct the model Hessian matrix, $\mathbf{H} \in \mathbb{R}^{d_L \times d_L}$, of the log posterior likelihood at the maximum a posteriori (MAP) estimate. This MAP estimate is induced as the Gaussian process posterior approximated in (8.16) and (8.17). Per [144], the model Hessian is defined as

$$\mathbf{H} = \sum_{i=1}^{N_{\text{train}}} \Phi_i \Phi_i^T + \mathbf{I}, \quad (8.23)$$

where N_{train} is the number of training examples. The Hessian matrix corresponds to the precision matrix, or the inverse of the covariance matrix $\Sigma \in \mathbb{R}^{d_L \times d_L}$. Hence, the prediction variance of a new example can be computed as

$$\text{var}(\mathbf{z}_k) = \Phi_k^T \Sigma \Phi_k \quad (8.24)$$

where Σ is fixed after training but Φ_k is a function of the new input $\mathbf{z}_k$. The entirety of the

training algorithm is presented in Algorithm 4. All of the code was implemented using the Python packages Jax [23] and Flax [91].

Algorithm 4. SpSNGP Training

Input: K snapshots of $\mathbf{w}$, $\dot{\mathbf{w}}$, $\ddot{\mathbf{w}}$, and the corresponding measurements at the boundary, $\mathbf{x}_\beta$, $\dot{\mathbf{x}}_\beta$, $\ddot{\mathbf{x}}_\beta$. Additional hyperparameters: size of training set N_{train} and validation set N_{val} , starting learning rate l_r , batch size N_b , maximum number of epochs N_{epochs} , and number of minibatches $N_{mb} = (N_{\text{train}} - N_{\text{val}})/N_b$.

Output: Learned nonlinear potential V_{nl} and nonlinear damping $\mathcal{J}_{\text{nl}}$ neural GPs with trained weights $\Psi = [\Psi_V, \Psi_I]$.

- 1: Split training data into $\mathbf{w}^{\text{train}} = [\mathbf{w}^{\text{subtrain}}, \mathbf{w}^{\text{val}}]$ and $\mathbf{x}_\beta^{\text{train}} = [\mathbf{x}_\beta^{\text{subtrain}}, \mathbf{x}_\beta^{\text{val}}]$.
 - 2: Randomly initialize weights $\Psi^0 = [\Psi_V^0, \Psi_I^0]$. Specifically, $\mathbf{W}_L \sim N(0, 1)$, $\mathbf{b}_L \sim U(0, 2\pi)$.
 - 3: Initialize epoch counter $n_e = 0$.
 - 4: **while** $n_e < N_{\text{epochs}}$ **do**
 - 5: **for** $i = 1 : N_{mb}$ **do**
 - 6: Get a minibatch from the sub-training set, $\mathbf{w}^{\text{batch}} \subseteq \mathbf{w}^{\text{subtrain}}$ and $\mathbf{x}_\beta^{\text{batch}} \subseteq \mathbf{x}_\beta^{\text{subtrain}}$.
 Similarly for $\dot{\mathbf{x}}_\beta$, $\ddot{\mathbf{x}}_\beta$.
 - 7: Extract $\mathbf{y}_\beta^{\text{batch}}$ from $\mathbf{w}^{\text{batch}}$
 - 8: Compute $\mathbf{z}_\beta^{\text{batch}} = \mathbf{x}_\beta^{\text{batch}} - \mathbf{y}_\beta^{\text{batch}}$
 - 9: Compute $V_{\text{nl}}(\mathbf{z}_\beta^{\text{batch}})$, $\mathcal{J}_{\text{nl}}(\dot{\mathbf{z}}_\beta^{\text{batch}})$
 - 10: Compute the deviatoric force $\mathbf{Q}_\star^{\text{batch}}$ (Eqn. 8.12)
 - 11: Assemble the generalized force vector from external and deviatoric forces (Eqn. 8.11)
 - 12: Compute accelerations from equation of motion (Eqn. 8.13)
 - 13: Evaluate loss function terms to construct total loss (Eqn. 8.22)
 - 14: Accumulate loss and compute gradient of loss function $\nabla_\Psi \mathcal{J}$.
 - 15: Update weights $\Psi^{N_{mb}n_e+i+1} = \text{ADAM}(l_r, \nabla_\Psi \mathcal{J}, \Psi^{N_{mb}n_e+i})$.
 - 16: **end for**
 - 17: Repeat **for loop** on $\mathbf{w}_{\text{val}}$ with updated weights.
 - 18: Accumulate loss on $\mathbf{w}_{\text{val}}$ to evaluate network convergence.
 - 19: $n_e = n_e + 1$
 - 20: **end while**
 - 21: Construct Hessian matrix (Eqn. 8.23).
 - 22: Invert Hessian matrix to obtain posterior covariance matrix, Σ .
-

8.1.4 Hypothesis Testing for Domain Shift

For a given structural system, the model (8.15) exhibits an expected structural response that enforces preservation of the underlying geometry induced by the physical model. In turn, the physical model may be represented by the parameters associated with the nonlinearities in

stiffness, damping, and hysteretic nonlinear damping. Any change in parameter may yield a change in the response, notated here as a domain shift. Therefore the existence of a domain shift indicates there is a change in the parameter set for the physical model.

Detection of a domain shift leads to the development of a hypothesis test that is both sensitive to changes within the parameter set yet robust to changes in the initial conditions. As knowledge of the parameters are unknown, changes in the output trajectories of (8.15) may be used as a symptom of the presence of domain shift. Similarly, due to the harmonics of the resulting trajectories, a hypothesis test that is robust to windowing is necessary. Therefore tests that involve location parameters such as the mean and median may lead to an increase in false positives in the test. For shape parameters such as the variance, heavy tails and multi-modality within the sample may also lead to high false positives than expected. Therefore, a test of hypotheses is developed using the interquartile range.

Following the work of [169], for a given sample size, n , or trajectory samples over a sampling frame, $\{t_1, t_2, \dots, t_n\}$, we can define X_i to represent the value of the trajectory, $X(t_i)$, at time t_i . Then $X_{n:1}, X_{n:2}, \dots, X_{n:n}$ form an ordered sample of the trajectories with $X_{n:1}$ and $X_{n:n}$ being the minimum and maximum of the unordered sample, respectively. While the joint distribution of the ordered sample is given by

$$f_{X_{n:1}, \dots, X_{n:n}}(y_1, \dots, y_n) = n! \prod_{k=1}^n f(y_k), \quad (8.25)$$

where $-\infty < y_1 \leq y_2 \leq \dots \leq y_n < \infty$ represents the space of possible orderings, $y_k := X_{n:k}$. From (8.25), the joint distribution of two order statistics is given by

$$\begin{aligned}
f_{X_{n:i}, X_{n:j}}(y_i, y_j) &= \frac{n!}{(i-1)!(j-i-1)!(n-j)!} \\
&\times [F_X(y_i)]^{j-1} \\
&\times [F_X(y_j) - F_X(y_i)]^{j-i-1} \\
&\times [1 - F_X(y_j)]^{n-j} \\
&\times f_X(y_i) f_X(y_j),
\end{aligned} \tag{8.26}$$

where $-\infty < y_i \leq y_j < \infty$, F_X and f_X are the cumulative distribution function and the probability distribution function of the unordered sample, X . A hypothesis test involving the interquartile range then depends on the convolution distribution, $\text{IQR} := X_{n:\hat{j}} - X_{n:\hat{i}}$, where $\hat{i}$ and $\hat{j}$ are the indices that represent the first and third quartiles, respectively.

Despite having a framework for the convolution distribution for IQR, computation of (8.26) requires knowledge of the unknown unordered sampling distribution, f_X . Furthermore, the potential multi-modality and heavy tails in estimating f_X may lead to mischaracterization of the variance of the convolution distribution. Therefore, the mean and variance of the convolution distribution is estimated through a bootstrap. Hence a comparison test of two interquartile ranges can be computed through a Wald-type test.

Given two trajectories, $X_1, X_2, \dots, X_n$, and $Z_1, Z_2, \dots, Z_n$, a comparison of the interquartile ranges can be defined as

$$\begin{aligned}
H_0 : \quad & \text{IQR}_X - \text{IQR}_Z = 0 \\
H_1 : \quad & \text{IQR}_X - \text{IQR}_Z \neq 0
\end{aligned} \tag{8.27}$$

The means and variances of the convolution distributions derived from (8.26) are obtained through a bootstrap of K samples. The resulting Wald-type test is carried out explicitly by a Welch's t -test defined as

$$T = \frac{\bar{X} - \bar{Z}}{\sqrt{S_X^2 + S_Z^2}} \quad (8.28)$$

The test statistic (8.28) then follows a t -distribution with degrees of freedom

$$\text{df} = \frac{\left(\frac{S_Z^2}{K} + \frac{S_X^2}{K}\right)^2}{\frac{1}{K-1} \left(\frac{S_Z^2}{K}\right)^2 + \frac{1}{K-1} \left(\frac{S_X^2}{K}\right)^2} \quad (8.29)$$

obtained through a Welch–Satterthwaite approximation [269]. A large absolute value in (8.28) suggests that the interquartile ranges of the unordered trajectories are different. With one of the two trajectories being a baseline trajectory, a rejection of (8.27) indicates that the second trajectory exhibits a potential domain shift in the system.

8.2 Results

The following subsections describe an example problem and demonstrate the proposed approach to define predictive intervals and detect shifts in the domain through hypothesis testing.

8.2.1 Example description

A simple one-dimensional rod previously introduced in [180] was used to demonstrate the proposed approach. A general description is provided here for convenience. Proportional damping was defined as $\mathbf{C} \equiv \xi \mathbf{K}$. The nonlinear region $\mathcal{C}_2$ is located in the interval $(s_1, s_2) = (0.25, 0.35)$, so that the isolated region exists between elements $\ell \equiv 16$ and $r \equiv 22$. The deviatoric force across the isolated region is identified as

$$\delta Q \equiv Q_{23} - Q_{15}, \quad (8.30)$$

The initial conditions of the system are specified in terms of the modal displacements

and velocities $(\mathbf{q}(0), \dot{\mathbf{q}}(0))$, where

$$\mathbf{x}(0) = \Phi \mathbf{q}(0), \quad \dot{\mathbf{x}}(0) = \Phi \dot{\mathbf{q}}(0), \quad (8.31)$$

where Φ is the modal transformation matrix.

The nonlinearities in the isolated region are assumed to contain linear mistuning and cubic nonlinearities in terms of both stiffness and damping. Finally, hysteretic nonlinear damping is also present between the elements within $\mathcal{C}_2$, so that the nonlinear force is represented as

$$\begin{aligned} f_{\text{nl},i} = & \kappa_i (x_{i+1} - x_i) + \chi_i (\dot{x}_{i+1} - \dot{x}_i) \\ & + \eta_i (x_{i+1} - x_i)^3 \\ & + \zeta_i (\dot{x}_{i+1} - \dot{x}_i)^3 + g_{\text{hys},i}, \end{aligned} \quad (8.32)$$

where the hysteretic force is defined as

$$\begin{aligned} g_{\text{hys},i} = & k_h ((x_{i+1} - x_i) - r_i), \\ \dot{r}_i = & \varepsilon_i \tan \left(\frac{k_h}{\mu} ((x_{i+1} - x_i) - r_i) \right), \end{aligned} \quad (8.33)$$

following the regularized formulation presented in [180].

Unless noted otherwise the parameters of the nonlinear region are chosen as

$$\begin{aligned} \kappa_i = 0, \quad \chi_i = 0, \quad \eta_i = \rho_\eta \cdot 2.621 \times 10^5, \\ \zeta_i = 2.621 \times 10^3, \quad k_{h,i} = \rho_{k_h} \cdot 6.50 \times 10^1, \\ \mu_i = 1.00 \times 10^{-1}, \quad \varepsilon_i = 1.00 \times 10^{-2}, \end{aligned} \quad (8.34)$$

with $\xi = 10^{-4}$, and ρ_η and ρ_{k_h} represent the level of cubic nonlinearity and hysteresis, respectively. For the data generated for training the network, $\rho_\eta = 1.0$, and $\rho_{k_h} = 0.0$.

8.2.2 Definition of domain shift

To demonstrate the proposed approach, several structural changes were considered. The purpose of each of these changes is to demonstrate how the methodology can be used to detect domain shifts from data. In this work, these structural changes were designed to shift the domain of the data with respect to the source domain represented by the training data (referred to as the Baseline). These changes are modeled by modifying certain parameters of the underlying physical model, namely the parameters associated with the nonlinearities: cubic stiffness and hysteresis. The motivation for choosing these parameters is that they both may be representative of damage. For example, hysteresis arises in the presence of plasticity or frictional dissipation due to slip (e.g., due to a loose bolt). On the other hand, a change in the stiffness may represent localized structural damage. However, a change in initial conditions does not represent a domain shift because the underlying physics remain the same. The full list of cases that were considered is presented in Table 8.1. There are four distinct groups that were modeled. The first group belonging to the Baseline cases represents the cases where the data is expected to be in-distribution with respect to the source domain. Baseline 1 and 2 are only different from Baseline 0 in that the initial conditions have been changed by deforming the structure in a shape composed of contributions from the first three elastic modes as described in Section 8.2.1. It is noted that the change of initial conditions prescribed here do not change the basin of attraction of the dynamic system. A bifurcation in the dynamic system would be considered domain shift but it was not studied in this work. The Hysteresis cases contain a non-zero hysteretic force modeled as described in Section 8.1.1. The Cubic Stiffness and Gradual Cubic Stiffness cases both contain changes to the cubic stiffness in the isolated region. However, in the Gradual Cubic Stiffness case, the cubic stiffness is changed gradually with a quadratic ramp starting at 4 seconds and reaching its maximum value at 10 seconds. The quadratic ramp was chosen because it mimics a self-accelerating damage mechanism. The specific parameters that are modified for each of these cases are ρ_η and ρ_{k_h} which were introduced in Eqn. (8.34).

Table 8.1. Domain shift cases. For the gradual stiffness increase, the max value is provided. The initial condition (IC) coefficients represent the contribution of each mode shape to the initial deformation.

Case name	ρ_η	ρ_{k_h}	IC
Baseline 0 (train)	1	0	(1.0,0.5,2.0)
Baseline 1 (test)	1	0	(0.5,1.0,0.5)
Baseline 2 (test)	1	0	(1.0,1.0,1.0)
Hysteresis Lvl. 1	1	1	(1.0,0.5,2.0)
Hysteresis Lvl. 2	1	10	(1.0,0.5,2.0)
Hysteresis Lvl. 3	1	20	(1.0,0.5,2.0)
Hysteresis Lvl. 4	1	50	(1.0,0.5,2.0)
Hysteresis Lvl. 5	1	100	(1.0,0.5,2.0)
Cubic Stiff. Lvl. 1	2	0	(1.0,0.5,2.0)
Cubic Stiff. Lvl. 2	5	0	(1.0,0.5,2.0)
Cubic Stiff. Lvl. 3	20	0	(1.0,0.5,2.0)
Cubic Stiff. Lvl. 4	100	0	(1.0,0.5,2.0)
Grad. Cubic Stiff. Lvl. 1	5	0	(1.0,0.5,2.0)
Grad. Cubic Stiff. Lvl. 2	20	0	(1.0,0.5,2.0)
Grad. Cubic Stiff. Lvl. 3	100	0	(1.0,0.5,2.0)

8.2.3 Effect of domain shift on baseline response

Numerical simulations were run for all the cases listed in Table 8.1 and the responses at the boundary ($\mathbf{x}_\beta$, $\dot{\mathbf{x}}_\beta$) were used as the reference representing the monitored boundary. At each time step, the neural networks were evaluated to obtain an updated response ($\mathbf{w}$, $\dot{\mathbf{w}}$) from which an updated deviatoric force was obtained recursively. In other words, the error induced by the neural networks is propagated through time integration at each time step. For this reason, the mismatch between the system that the network is evaluating (learned from the Baseline 0 data) and the new data coming in corresponding to a different system should induce error on the predictions. This error is represented in Fig. 8.2 which illustrates the phase portrait of a representative DOF obtained for all the different cases, with the corresponding mean percent error (MPE) obtained for all 64 DOF displacements. The representative DOF was chosen to be outside the isolated region but close enough to be sensitive to the nonlinearities. For ease of

visualization, the different groups have different border colors. As illustrated, the Baseline cases have the lowest MPE, as expected, since these cases are all within the training domain, even if Baseline 1 and 2 have different initial conditions. Next, the Hysteresis cases have increasing MPE as the hysteresis level (corresponding to the magnitude of ρ_{k_h}) goes up. It should be noted that the MPE does not increase linearly with the magnitude of ρ_{k_h} because the hysteresis force is mathematically saturated at a certain level. It is observed that even though significant hysteresis was introduced, the trained system was still able to predict the response with an MPE below 1% even after 10,000 steps. The fact that the MPE remains low is evident in Fig. 8.2 where the solid and dashed lines are indistinguishable from each other in most cases. This result highlights one of the challenges of the detection problem because these types of errors may be easy to miss since they are relatively small. This observation will play a role in the choice of algorithms used for detection in Section 8.2.4. The Cubic Stiffness cases induce a larger error on the response with the very extreme level 4 (where the original cubic stiffness was increased by 100 times) having an MPE of up to 8.33%. In contrast, the Gradual Cubic Stiffness cases where the cubic stiffness is gradually increased, the error is still relatively low, though it is possible that it would continue to grow as time progresses.

Based on this discussion, the MPE seems to be an indicator that something in the system has changed. Although we are able to compute the error in this controlled setting, in a real application the true response may not be known. For this reason, we seek to compute another output that can still be used as an indicator of domain shift in the absence of the true response. As discussed in Section 8.1.3, in this work we have used an SNGP to compute the variance associated with each prediction from the neural network. Because the neural networks predict the nonlinear potential and nonlinear damping functions directly, these are the functions that will be inspected going forward. Furthermore, it was found that the nonlinear damping function was more sensitive to domain shifts, so special attention will be granted to the nonlinear damping. To illustrate how the variance changes when the domain shifts due to structural changes, Fig. 8.3 shows the nonlinear damping ($\mathcal{J}_{nl}$) as a function of time with an interval corresponding to

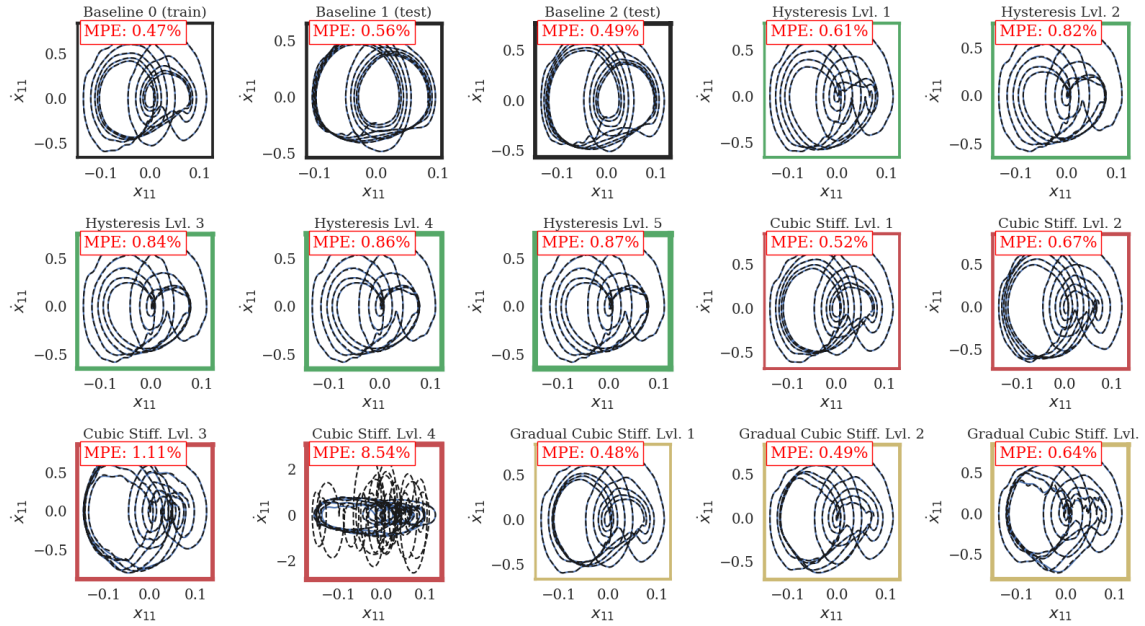


Figure 8.2. Phase portraits of a representative DOF (DOF = 11). Mean percent error annotated for each case. Actual response shown in a solid blue line, predicted response shown in a black dashed line.

one standard deviation. The bottom plot shows the prediction variance as a function of time. For conciseness, only the Gradual Cubic Stiffness cases are shown. As observed, the prediction variance grows as the severity of the change increases, with level 3 having a very clear increase. It is also noted that the gradual stiffness change starts at 4 seconds, hence the variance and mean prediction agree perfectly below 4 seconds. This result shows how the predicted variance can be interpreted by the user as a warning regarding the fidelity of the output in the shifted domain. In other words, the network exhibits uncertainty awareness. It is interesting to note that the MPE for level 3 was still under 1% but the network still predicts high uncertainty, which is a desirable behavior because the network has no way of knowing whether or not the prediction is accurate and therefore, it is appropriate for it to be uncertain about its prediction. The statistics of these signals will be explored in more detail in the next section to explore how they relate to domain shift.

Fig. 8.3 also illustrates another challenge when comparing variance outputs across

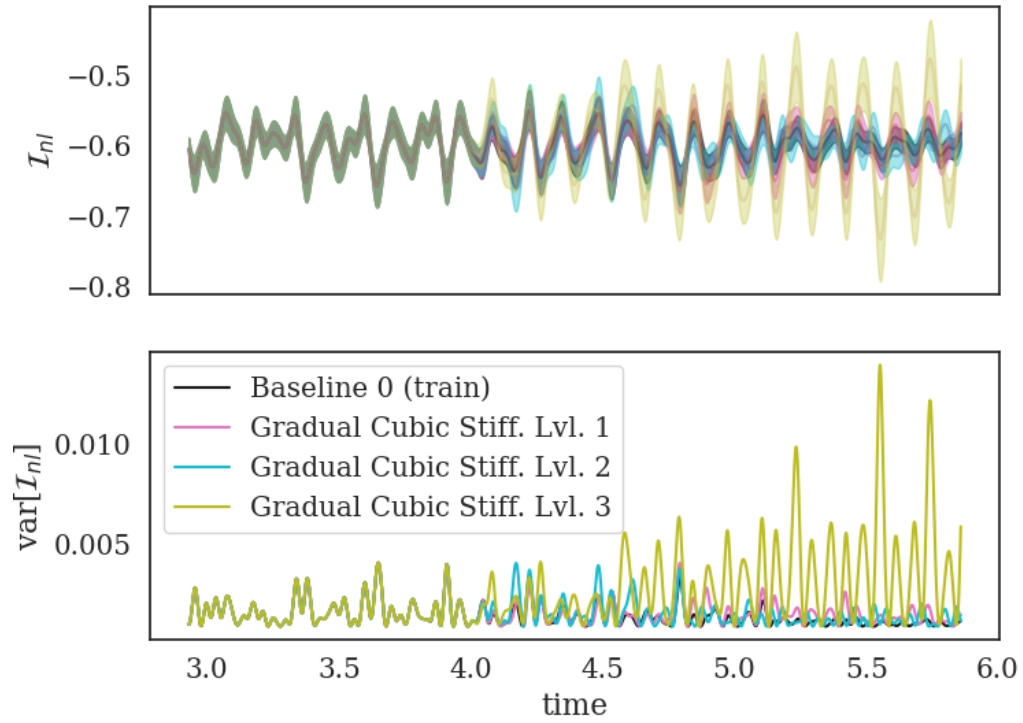


Figure 8.3. Top: nonlinear damping mean ($\mathcal{I}_{nl}$) as a function of time with one standard deviation shown around it. Bottom: variance of the nonlinear damping as a function of time.

different cases and that is the fact that time histories cannot be compared directly because they may take different trajectories, or they may not even have the same initial conditions. For this reason, it cannot be assumed that the time histories will be time-consistent. Hence, instead of comparing the prediction variance on a time step by time step basis, statistical properties of the variance time histories collected over 10,000 time steps are used. Fig. 8.4 shows the boxplots corresponding to all the cases considered for both the nonlinear potential (top) and nonlinear damping (bottom). Note that the natural log of the variance is used for convenience. As shown, the statistical distributions of the log-variance change as structural changes are introduced. Qualitatively, both the dispersion of the distribution and the central tendency change to different degrees as a function of the underlying structural changes. Therefore, the discussion going forward will focus on the statistical features computed from the distributions, namely the mean and median, the variance (i.e., the variance of the log-variance distributions), and the interquartile range (IQR), which is defined as the range between the first and third quartile. Because these distributions are not Gaussian, the IQR and the variance have different properties which will be explored in Section 8.2.4.

Lastly, to illustrate how these statistical features (i.e., mean, variance, and IQR) correlate to the MPE, Fig. 8.5 plots three statistical properties extracted from the log-variance for all the different cases in Table 8.1 vs. their corresponding MPE. As shown, there is a high correlation, as measured by the Spearman rank coefficient [99], between the MPE and the different statistical properties of the log-variance, which was over 83% for all features. This result suggests that these features can be used to detect the underlying domain shift as will be explored in Section 8.2.4.

8.2.4 Domain shift detection

The results presented in Section 8.2.3 suggest that the statistical properties of the log-variance distributions may be used as indicators of an underlying domain shift. However, the qualitative assessment presented in Section 8.2.3 is not enough to perform domain shift detection systematically. To this end, binary hypothesis testing is used to determine whether domain

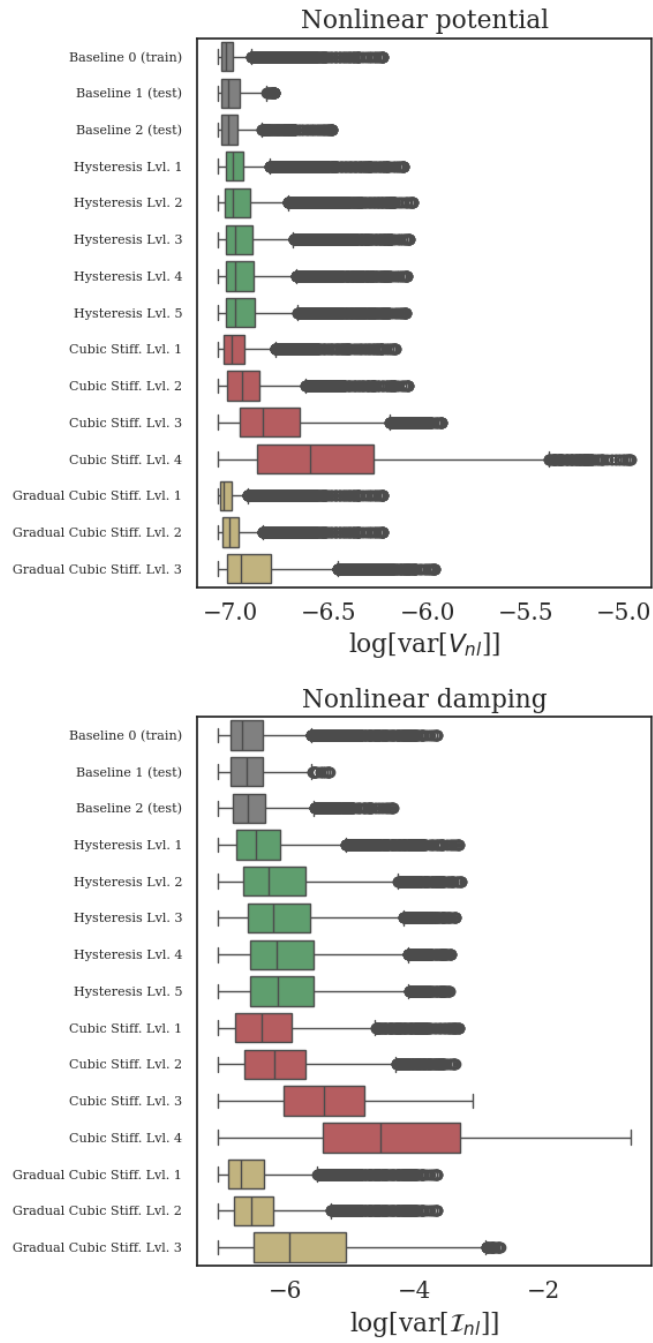


Figure 8.4. Top: box plots for each cases considered of the log-variance of the nonlinear potential (top) and the nonlinear damping prediction.

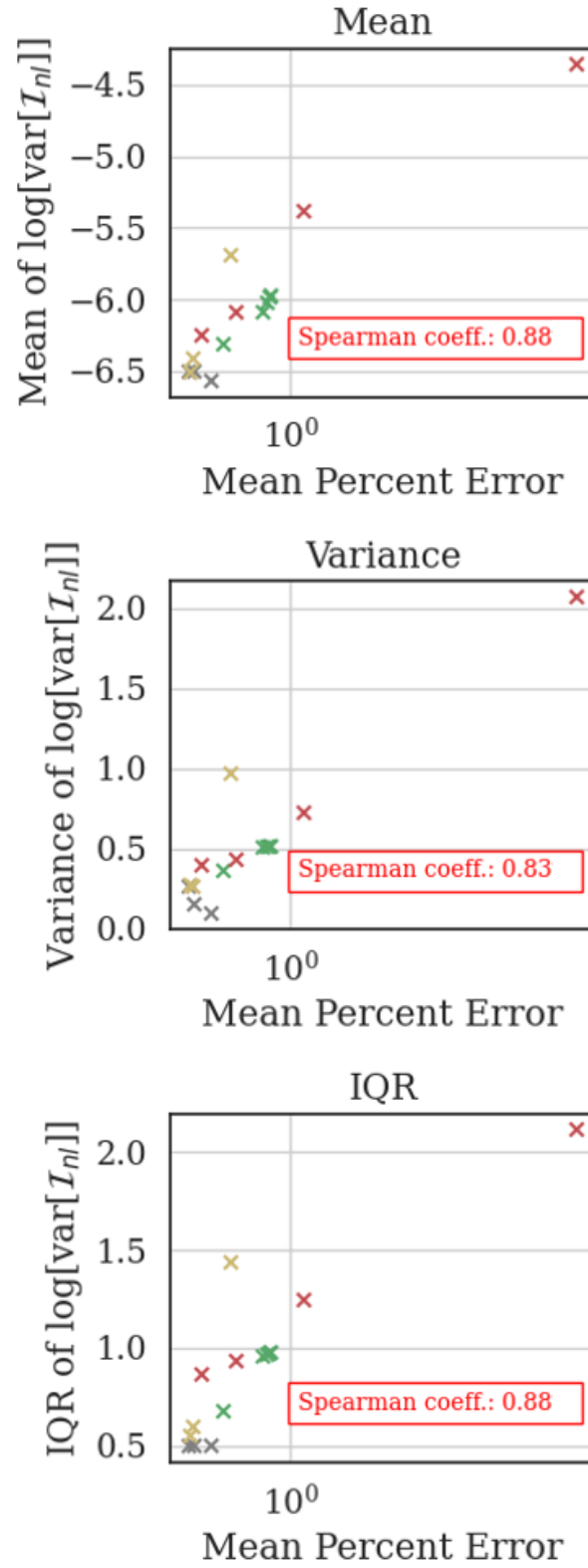


Figure 8.5. Top: nonlinear damping mean ($\mathcal{I}_{nl}$) as a function of time with one standard deviation shown around it. Bottom: variance of the nonlinear damping as a function of time.

shift has occurred with respect to the original Baseline 0 case. Based on the results presented in Section 8.2.3 suggesting that the log-variance distribution can function as an indicator, the following tests were studied to find the most appropriate test for this application: Kolmogorov-Smirnov test to compare statistical distributions, Bartlett test of distribution variance to test for homoscedasticity across different cases, a Levene test of variance equality using both the mean and median methods, and lastly a Wald test of the IQR (as described in Section 8.1.4). For each of these tests, the null hypothesis (H_0) represents the case in which the data are in-distribution (or within the domain) with respect to the Baseline 0 case. The alternative hypothesis (H_1) then represents the case in which the data is out-of-distribution (or in a shifted domain) with respect to Baseline 0. The most significant challenge with performing hypothesis testing for these data is finding a test that is robust to initial condition changes.

After performing all the tests listed above, it was found that all of them, except for the Wald test of IQR, rejected the null hypothesis for all cases. These results imply that all the tests, except for the IQR test, failed to be robust to normative changes (i.e., initial conditions). To better understand the intuition behind these results, we can study the box plots in Fig. 8.4. If we focus on the nonlinear damping distributions, it can be observed that, at least visually, the dispersion of the distribution may be a good indicator of domain shift since the dispersion is roughly equal for the gray boxes (Baseline) but different for all the other cases. It can also be observed that the median or the mean may not be great indicators because they shift even within the gray boxes (Baseline). This observation leads to the consideration of distribution dispersion as a candidate indicator of domain shift. Fig. 8.6 shows the variance and IQR for each of the cases, with a horizontal dashed line indicating the Baseline 0 case. As observed, the variance of the distribution is not consistent across the Baseline cases, which means that the variance feature is not robust to changes in initial conditions. On the other hand, the IQR has the desired properties of the indicator feature: the IQR is very consistent across the Baseline cases and it grows as the level of domain shift is increased. For this reason, the IQR is the preferred domain shift indicator feature.

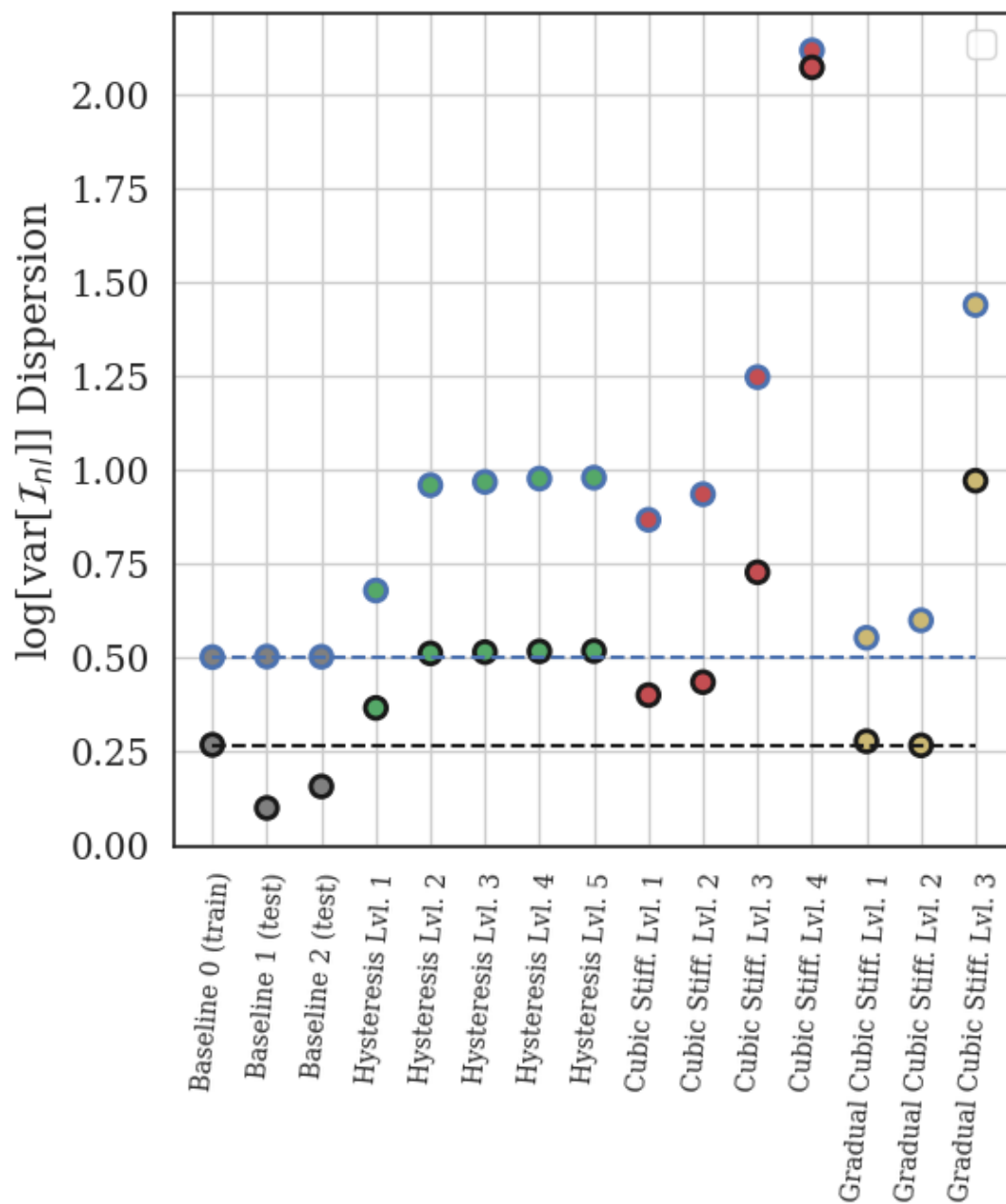


Figure 8.6. Comparison of dispersion metrics as a function of domain shift. Blue: IQR, Black: Variance. Dashed lines represent the baseline dispersion level.

The IQR of the distributions is a single scalar value so it cannot be used in hypothesis testing directly. Instead, 500 samples are obtained via bootstrapping of the original 10,000 samples in the log-variance time history to obtain the convolutional distribution for the IQR (as described in Section 8.1.4). Choosing more than 500 samples did not change the results. The bootstrapping approach results in a non-parametric distribution which limits the types of hypothesis tests that can be performed. It is noted that a parametric formulation of the IQR distribution was attempted by first performing kernel density estimation of the original log-variance distribution but this approach failed because the log-variance distributions exhibit multi-modal behavior and heavy tails, as described in Section 8.1.4. Other likelihood-free approaches such as fitting a normalizing flow [279] could be explored in future work. For this work, the classical Wald test was used to perform hypothesis testing, as described in Section 8.1.4. As reported before, the Wald test of the IQR distributions resulted in the desired behavior of failing to reject the null hypothesis for all the Baseline cases while rejecting it for all other cases. The test statistics obtained for each binary hypothesis test are presented in Fig. 8.7. Note that the test statistic is provided instead of the p-value because some of the p-values are smaller than machine precision and therefore not very informative. The test statistic is compared to the 95% confidence level (corresponding to a test statistic value of 1.96 for the t -distribution as shown in Eqn.(8.28)). Fig. 8.7 can be interpreted as anything below the dashed representing “fail to reject H_0 ” cases while anything above representing “reject H_0 ” cases. In other words, the cases above the dashed line are considered to be in a shifted domain. The IQR distributions are also plotted in Fig. 8.7 to illustrate how these vary with each case. As shown, the mean of the distributions roughly follow the level of severity associated with the structural change but the relationship between these two has not been thoroughly explored in this work.

Lastly, to understand how the distance-preserving mechanism embedded in the networks is operating, we applied multidimensional scaling (MDS) to both the original state space data ($\mathbf{w}, \dot{\mathbf{w}}$) and the transformed hidden representation obtained by the networks ($\mathbf{h}_V, \mathbf{h}_I$). The MDS algorithm is a dimensionality reduction technique that maps high-dimensional data to a lower-

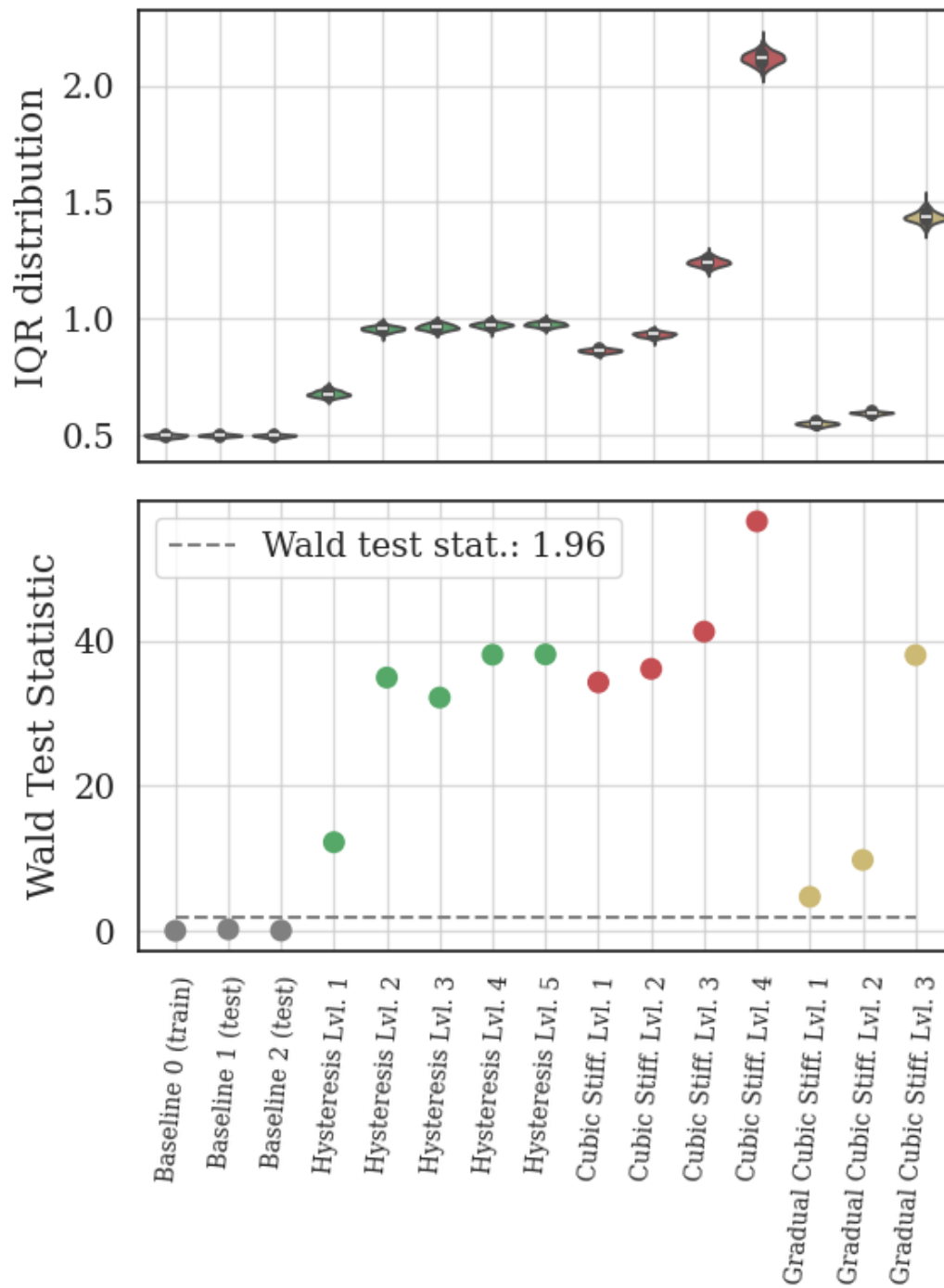


Figure 8.7. Top: violin plots of IQR distributions. Bottom: Wald test statistic for each case vs. Baseline 0. Dashed line represents 95% confidence interval (or 5% p-value).

dimensional space while preserving pairwise distances between the data points as best as possible, given the reduced dimension constraints [22]. For this reason, MDS is useful to understand relationships and patterns in data that may be hard to visualize in their original dimension. In this work, MDS is used solely for visualization purposes. In this case, both sets of data are mapped down to a 2-dimensional space with MDS. The state space data are transformed from their original 128 dimensions (64 DOFs, both displacements and velocities) and the hidden features from their original 32 dimensions (16 hidden units per network). The data are plotted in the transformed 2-dimensional feature space in Fig. 8.8. As shown, the Baseline points (shown in gray) are clustered together in the MDS representation of the hidden feature space while they are more separated in the original state space. These patterns show how the network transforms the data from their original physical space to a hidden space that, while preserving their distance and preventing feature collapse (as described in Section 8.1.3), also clusters the in-distribution points together when compared to the original state space. Also, neighbors, such as Baseline 0 and the Gradual Stiffness case remain close neighbors in the transformed space (which is also why a robust statistical test was developed instead of relying on distances). This property is a result of the enforced distance preservation via spectral normalization and, in combination with the GP kernel in the last layer, it is what enables the uncertainty-aware properties of the SNGP. It is noted that MDS is only a convenient way to visualize the high-dimensional data, but the strict distance preservation properties occur in the original 32-dimensional hidden space.

8.3 Conclusions

We presented an uncertainty-aware ML-enhanced digital twin with structure-preserving constraints to detect domain shifts in data. The proposed approach leveraged and extended previous work by the authors that combined a data-driven approach with physical constraints to learn the deviatoric behavior from isolated regions with nonlinearities. As the model is intended to be used in an online SHM system, the digital twin may be confronted with data

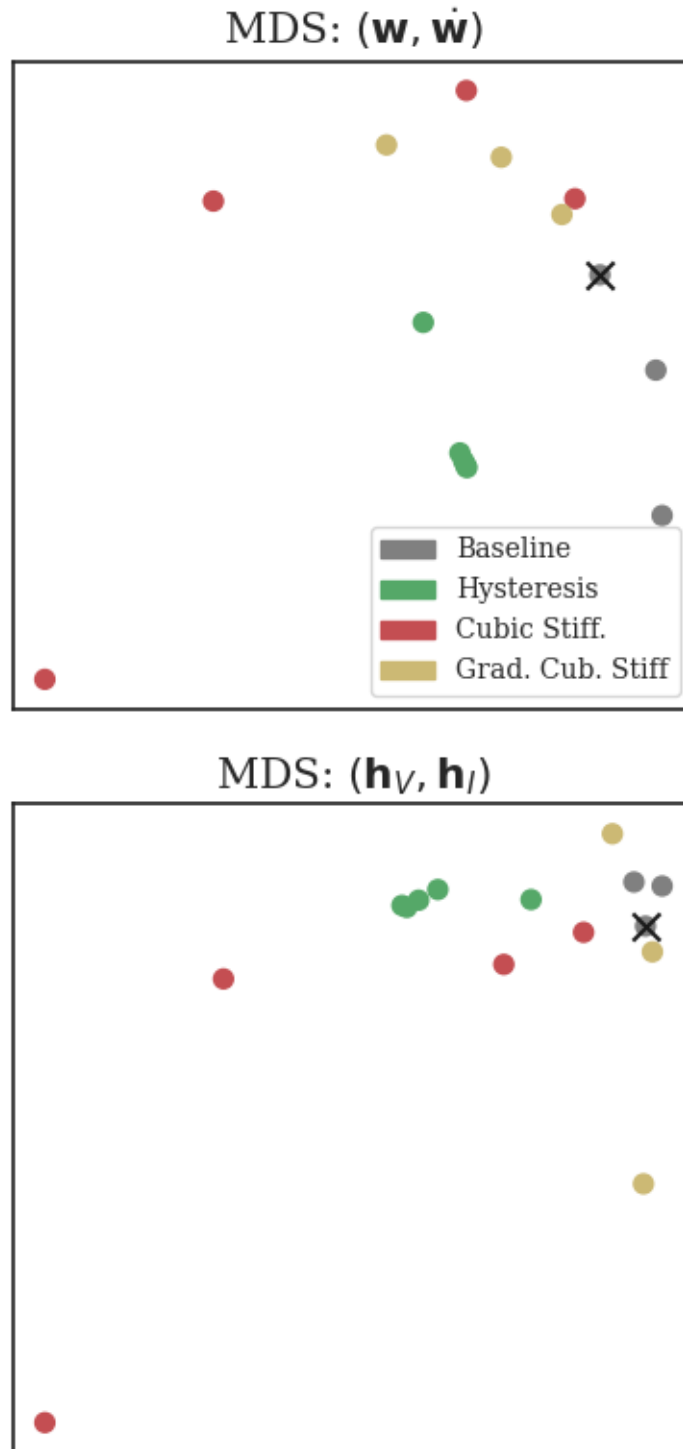


Figure 8.8. Top: MDS of 128-dimensional state space. Bottom: MDS of 32-dimensional hidden space. Color groups are consistent with Fig. 8.4. The marker “ $\times$ ” indicates the Baseline 0 case.

from a monitored system that has experienced structural changes (e.g., due to damage) leading to a domain shift with respect to the original data used for training. A computational example was used to demonstrate how the trained model can be used to detect domain shifts robustly with no prior knowledge about the underlying structural changes. This work showed how the order statistics of the predicted variance can be used to perform binary hypothesis testing for systematic detection of domain shift. Future work will explore applications to other systems, incorporating noise, and alternative architectures.

8.4 Preview of Chapter 9

Chapter 7 and 8 introduced a strategy for uncertainty quantification of the deterministic ML model. While Chapter 7 focused on treatment of aleatory uncertainty (e.g., measurement noise), Chapter 8 presented an approach for characterization of epistemic uncertainty. Furthermore, it was demonstrated how this measure of uncertainty can be used to detect domain shift in the data being ingested by the ML model. Chapter 8 showed how the proposed approach is robust to normative changes (such as in-distribution change of initial conditions) and sensitive to structural changes that should be flagged as domain shifts. Chapter 9 builds upon this work to demonstrate the design of a domain shift detector by considering the trade-offs between sensitivity and specificity or, in other words, precision and recall.

8.5 Remarks

Portions of this dissertation have been published or are being considered for publication. Chapter 8 is composed of the following first authored publications:

David Najera-Flores, Justin Jacobs, D. Dane Quinn, Anthony Garland, and Michael Todd. Uncertainty-aware, structure-preserving machine learning approach for domain shift detection from nonlinear dynamic responses of structural systems. Manuscript accepted for publication in the Journal of Risk and Uncertainty in Engineering Systems Part B: Mechanical Engineering,

2024

Chapter 9

A Probabilistic Reasoner Based on Bayes Risk for Damage Detection in Structural Systems

9.1 Background and Motivation

A structural health monitoring system should enable robust operation of the monitored system by providing actionable information to stakeholders so they can perform informed decisions about the structure's maintenance. This task involves the mapping from data to knowledge via an inference process. This inference process is also known as *reasoning*. More precisely, a probabilistic reasoner takes into account the various probability distributions from uncertain events to propose a course of action given the cost of decision consequences (risk). This task implicitly involves performing a diagnosis or detection of an underlying change in the system via indirect observable symptoms [253]. The work by the author in [172] introduced a strategy for domain shift detection based on symptoms related to the dynamic response of a system. That work presented a method based on a distance-preserving, uncertainty-aware machine learning model that acted as a digital twin. The proposed model was able to ingest data from an external source (such as sensors) and compute the potential energy and the damping of the underlying system, as well as the corresponding predictive variance. The work in [172] demonstrated how a Wald-type hypothesis test based on the order statistics of the distribution

of the logarithm of the variance over time was useful for robust domain shift detection. The present work builds upon this past work to demonstrate how this workflow can be used to provide actionable information based on the desired sensitivity and specificity of the detector.

9.2 Uncertainty-aware ML model

The model introduced in Chapter 9 is briefly summarized in this section. The neural network architectures used in this work consist of residual connections with spectral-normalized weights. This choice of architecture ensures distance-preservation as described in [144] by enforcing the lower bound of the Lipschitz constant through its residual connections, and the upper bound by using spectral normalization. This construction represents a bi-Lipschitz constraint that prevents disproportional contraction (also known as feature collapse) or expansion of the input space when it is transformed to the hidden space [144]. There are two separate networks and they both follow the exact same architecture. The input to the network is either $\mathbf{z}_\beta$ or $\dot{\mathbf{z}}_\beta$, which are N_β -dimensional vectors, where N_β is the number of degrees-of-freedom in the boundary of the isolated region. The input layer maps the input to a the hidden space of dimension d with a linear dense layer. The next L hidden layers follow a residual architecture where the input is added to the output after each dense layer. A swish activation function is used at each dense layer. The output layer uses a Gaussian process (GP) with kernel matrix, $\mathbf{K}_{\text{GP}}$. The hidden representation, $\mathbf{h}_i \in \mathbb{R}^{d_{L-1} \times 1}$, of the previous penultimate layer of dimension d_{L-1} serves as input to the GP layer.

As the output Gaussian process is intractable, a low-rank approximation of the kernel matrix is applied via

$$\mathbf{K}_{\text{GP}} = \Phi\Phi^T, \quad (9.1)$$

where

$$\Phi_i = \sqrt{2\sigma^2/d_L} \cos(-\mathbf{W}_L \mathbf{h}_i + \mathbf{b}_L), \quad (9.2)$$

are random features that form a random sampling of a square-integrable basis [202] for the output layer. Here $\mathbf{W}_L \in \mathbb{R}^{d_L \times d_L}$ is a fixed weight matrix with entries sampled i.i.d. from a standard normal distribution, $\mathbf{b}_L \in \mathbb{R}^{d_L \times 1}$ is a fixed bias vector with entries sampled i.i.d. from a uniform distribution in the range $(0, 2\pi)$, and σ^2 is the GP kernel variance and is set to 1.0.

The output of the SNGP is then obtained as

$$g(h_i) = \Phi_i^T \mathbf{u} + c, \quad (9.3)$$

where $\mathbf{u} \in \mathbb{R}^{d_L \times 1}$ is the vector of learnable weights in the last layer and is initialized by randomly sampling from a zero-mean normal distribution with variance $\tau^2 = 3.0$ (tuned manually for numerical stability) and $c \in \mathbb{R}^{1 \times 1}$ is the learnable constant bias. The output $g(h_i) \in \mathbb{R}^{1 \times 1}$ is a scalar field corresponding to either the nonlinear potential $V_{\text{nl}}(\mathbf{z}_i)$ or nonlinear damping $\mathcal{J}_{\text{nl}}(\dot{\mathbf{z}}_i)$.

To obtain the prediction variance, we need to construct the model Hessian matrix, $\mathbf{H} \in \mathbb{R}^{d_L \times d_L}$, of the log posterior likelihood at the maximum a posteriori (MAP) estimate. This MAP estimate is induced as the Gaussian process posterior approximated in (9.1) and (9.2). Per [144], the model Hessian is defined as

$$\mathbf{H} = \sum_{i=1}^{N_{\text{train}}} \Phi_i \Phi_i^T + \mathbf{I}, \quad (9.4)$$

where N_{train} is the number of training examples. The Hessian matrix corresponds to the precision matrix, or the inverse of the covariance matrix $\Sigma \in \mathbb{R}^{d_L \times d_L}$. Hence, the prediction variance of a new example can be computed as

$$\text{var}(\mathbf{z}_k) = \Phi_k^T \Sigma \Phi_k \quad (9.5)$$

where Σ is fixed after training but Φ_k is a function of the new input $\mathbf{z}_k$. All of the code was

implemented using the Python packages Jax [23] and Flax [91].

9.3 Damage detector definition

The approach taken in this work is based on statistical detection theory [117]. However, unlike classical detection theory which deals with known signals, the present problem deals with unknown processes, e.g., cases in which the underlying statistical process is not completely known *a priori*. In this work, the probability density function and its corresponding parameters are unknown since they are being generated by a neural network. This statistical process corresponds to the predictive variance associated with the damping and potential energy of the structural system. These signals are used as observable symptoms in lieu of direct observation of damage. To this end, the problem statement in this work consists of the following binary hypothesis test:

$$\begin{aligned} H_0 &: \text{no damage} \\ H_1 &: \text{damage is present} \end{aligned} \tag{9.6}$$

Following the hypothesis test framework developed in [172], the two hypotheses rely on the statistical dispersion of the signal being considered as

$$\begin{aligned} H_0 &: \text{IQR}_X - \text{IQR}_Z = 0 \\ H_1 &: \text{IQR}_X - \text{IQR}_Z \neq 0, \end{aligned} \tag{9.7}$$

where IQR is defined as the interquartile range (IQR) of the random signals X and Z , which correspond to a vector of unordered samples. This vector is defined as the logarithm of the predictive neural network variance within a predefined time window. In this case, the signal X corresponds to the predictive variance associated with a baseline case, which is representative of

the healthy state of the structure, while Z represents other time signals to be tested.

An empirical distribution of the random signals, X and Z , is obtained through a bootstrap of K samples drawn within a predefined time window. Consistent with [172], a Wald-type test is carried out explicitly by a Welch's t -test defined as

$$T = \frac{\bar{X} - \bar{Z}}{\sqrt{S_X^2 + S_Z^2}}. \quad (9.8)$$

The test statistic (9.8) then follows a t -distribution with degrees of freedom

$$\text{df} = \frac{\left(\frac{S_Z^2}{K} + \frac{S_X^2}{K}\right)^2}{\frac{1}{K-1} \left(\frac{S_Z^2}{K}\right)^2 + \frac{1}{K-1} \left(\frac{S_X^2}{K}\right)^2}. \quad (9.9)$$

The two-tailed p-value associated with the test statistic in (9.8) can then be computed as

$$p = 2(1 - F(T)) \quad (9.10)$$

where $F(T)$ represents the cumulative density function (CDF) of the test statistic. The hypothesis test can therefore be expressed as

$$\begin{aligned} p &\geq \alpha : \text{fail to reject } H_0 \\ p &< \alpha : \text{we reject } H_0 \end{aligned} \quad (9.11)$$

where α represents the significance level of the hypothesis test, also interpreted here as the probability of false alarm (P_{FA}). In other words, the significance level α defines the probability $P_{FA} = P(H_1|H_0)$, or the probability of deciding that the structure is damaged when it is not, also known as a type I error.

Furthermore, this binary hypothesis test also allows us to compute the associated power of the test (denoted as β) which defines the probability that the test correctly rejects H_0 when H_1

is true. This correct inference is known as the probability of detection, $P_D = P(H_1|H_1)$. For most detectors, there exists a trade-off between the probability of false alarms and the probability of detection. This trade-off is the same as the trade-off between the specificity and sensitivity of the test (or the probability of false alarms vs. misses). Therefore, designing a detector is a task that must consider this trade-off in a systematic way. One approach to accomplish this is to rely on the concept of Bayes risk, $\mathcal{R}(\theta)$, which combines the weighted likelihood of each possible decision as

$$\mathcal{R}(\theta) = E(C, \theta) = \sum_{i=0}^1 \sum_{j=0}^1 C_{ij} P(H_i|H_j, \theta) P(H_j) \quad (9.12)$$

where the coefficients C_{ij} represent the cost associated with deciding H_i when H_j is true, θ are the fixed choice parameters associated with the detector or decision function, the likelihood terms $P(H_i|H_j, \theta)$ can be computed from the hypothesis test formulation as described above, and the prior terms $P(H_j)$ can be defined based on prior knowledge (for example, from historical data). The Bayes risk can be interpreted as the smallest integrated risk over all decision functions [210]. These decision functions are parameterized by θ which enables minimization of the risk function by considering all the possible θ .

9.4 Demonstration of damage detector performance

This section will demonstrate the approach described in Section 9.3 by applying the proposed workflow to a structural rod with isolated deviatoric behavior representative of damage.

9.4.1 Problem description

As described in Section 9.3, a Wald-type hypothesis test was used to detect damage in the rod structure described in Chapter 8. The same damage cases were considered in this work as well (listed in Table 8.1). In addition, 5% synthetic measurement noise was added to the input data. As demonstrated in Chapter 8, the trained neural networks are robust to measurement noise

even when added at every iteration during time integration. This noise was added with the intent of approximating a more realistic sensor measurement. The additive noise was modeled as

$$\mathbf{x}_\beta(t) = \bar{\mathbf{x}}_\beta(t) + \delta, \quad (9.13)$$

where $\bar{\mathbf{x}}_\beta(t)$ is the mean process and δ is the random noise which is assumed to be independent and identically distributed (IID) modeled as

$$\delta = v \cdot \sqrt{\frac{\sum_i^N \bar{x}_\beta(t_i)^2}{N-1}} \cdot \text{Normal}(0, 1), \quad (9.14)$$

where $v = 0.05$ and it represents the noise factor which can also be interpreted as the reciprocal of the signal-to-noise ratio (SNR) since it represents the ratio between the noise variance and the signal variance.

The deviatoric response inherits the same additive noise as

$$\mathbf{z}_\beta(t) = \bar{\mathbf{z}}_\beta(t) + \delta, \quad (9.15)$$

and similarly for the deviatoric velocities, $\dot{\mathbf{z}}_\beta(t)$. These are then used to compute the deviatoric forces as

$$\mathbf{Q}_\star = \frac{\partial V_{\text{nl}}(\mathbf{z}_\beta)}{\partial \mathbf{z}_\beta} + \frac{\partial \mathcal{J}_{\text{nl}}(\dot{\mathbf{z}}_\beta)}{\partial \dot{\mathbf{z}}_\beta}, \quad (9.16)$$

where $\mathbf{z}_\beta$ and $\dot{\mathbf{z}}_\beta$ are the deviatoric displacements and velocities at the measured boundary (as described in more detail in Chapter 9). The $V_{\text{nl}}(\cdot)$ and $\mathcal{J}_{\text{nl}}(\cdot)$ terms represent the nonlinear potential and damping functions, which are modeled with a structure-preserving, spectral-normalized neural Gaussian process (sp-SNGP) as described in Chapter 9. Consistent with previous work, this work focuses on the study of the damping term variance predicted by

sp-SNGP for domain shift detection as an indicator for damage.

9.4.2 Optimizing for probability of detection

Designing an optimal detector depends on the specific objectives of the SHM system. One reasonable objective is to maximize the probability of detection or, in other words, design a highly sensitive detector (i.e., a high recall rate). We can use the receiver operating characteristic (ROC) curve to study this objective. Moreover, the ROC curve also shows the trade-off between recall and precision. In other words, a highly-sensitive test will also result in a larger false alarm rate. To study how this trade-off, we consider the case where progressive damage is introduced via a gradual change to the cubic stiffness term. Referring to the nonlinearity described in Eq. 8.32, the cubic stiffness term is modified as

$$f_{nl,i} = \eta_i(x_{i+1} - x_i)^3 + \eta_{d,i}(t - t_d)^2(x_{i+1} - x_i)^3, \quad (9.17)$$

where

$$\begin{cases} \eta_d = 0.6944, & \text{if } t \geq t_d = 4 \\ 0 & \text{otherwise,} \end{cases} \quad (9.18)$$

where η is the cubic stiffness coefficient, η_d is the damage coefficient, and t_d is the time at the onset of damage. As shown, the damage growth is modeled as a quadratic function to represent a self-accelerating degradation process (i.e., the second derivative is non-zero).

Given this progressive damage model, we compute the ROC curve as a function of window size and time. We accomplish this by defining a window with a starting point fixed at $t = 4$ and variable window size. Figure 9.1 shows the ROC curves for 100 WS values corresponding to $WS = \{100, 129, 158 \dots 3000\}$, with the color of each line representing time evolution (from blue to red). As shown, as the lines get more red, the ROC curves approach an ideal detector (i.e., the area under the curve (AUC) approaches 1 and the p-value approaches 0). This result exemplifies how damage is easier to detect as time progresses which is consistent with

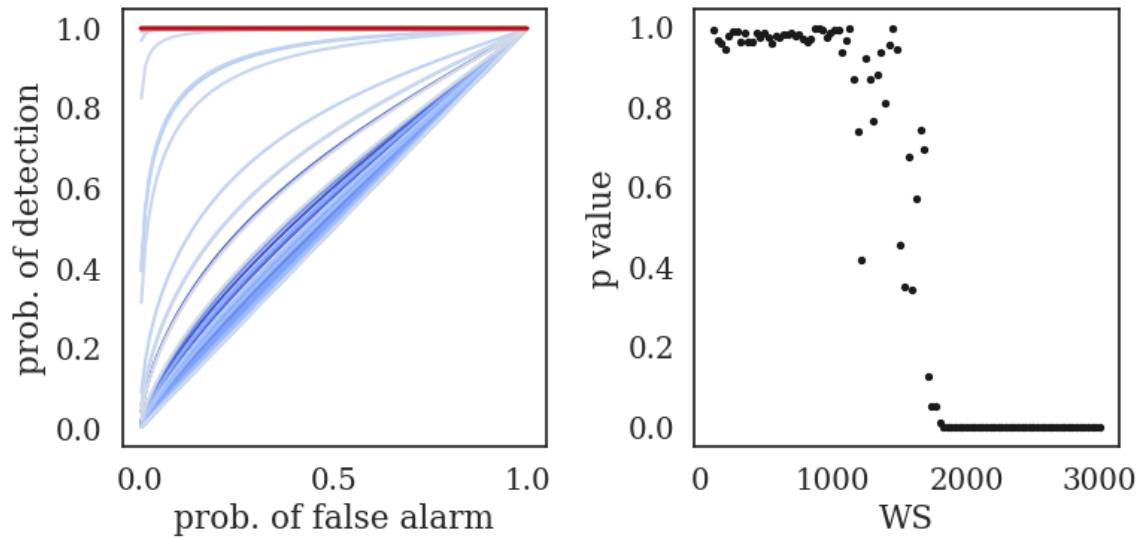


Figure 9.1. Left: ROC curves colored by WS value (blue to red: short to large). Right: P-value as a function of WS.

the self-accelerating damage evolution model. This effect is also evident in the IQR histograms shown in Figure 9.2 where it can be seen that the distributions are hard to distinguish when small window sizes are used while they become easier to distinguish as more data is collected (also consistent with the small p-values in Figure 9.1). As a result, one may choose the detector that maximizes the probability of detection, which corresponds to the detector that observes data in the most damaged state. However, this behavior may be unacceptable for a realistic SHM system where decisions need to be taken in a time-critical manner. To this end, Section 9.4.3 will consider the cost associated with each decision as well as the consequence of a delayed decision.

9.4.3 Considering cost of wrong decisions

As described in Section 9.4.2, a detector that only considers probability of detection tends to favor features corresponding to an extended monitoring period. However, waiting for data may have consequences (e.g., risk of structural failure) and/or additional monitoring cost associated with data collection. For this reason, an additional term to model this penalty is added to the Bayes Risk expression as

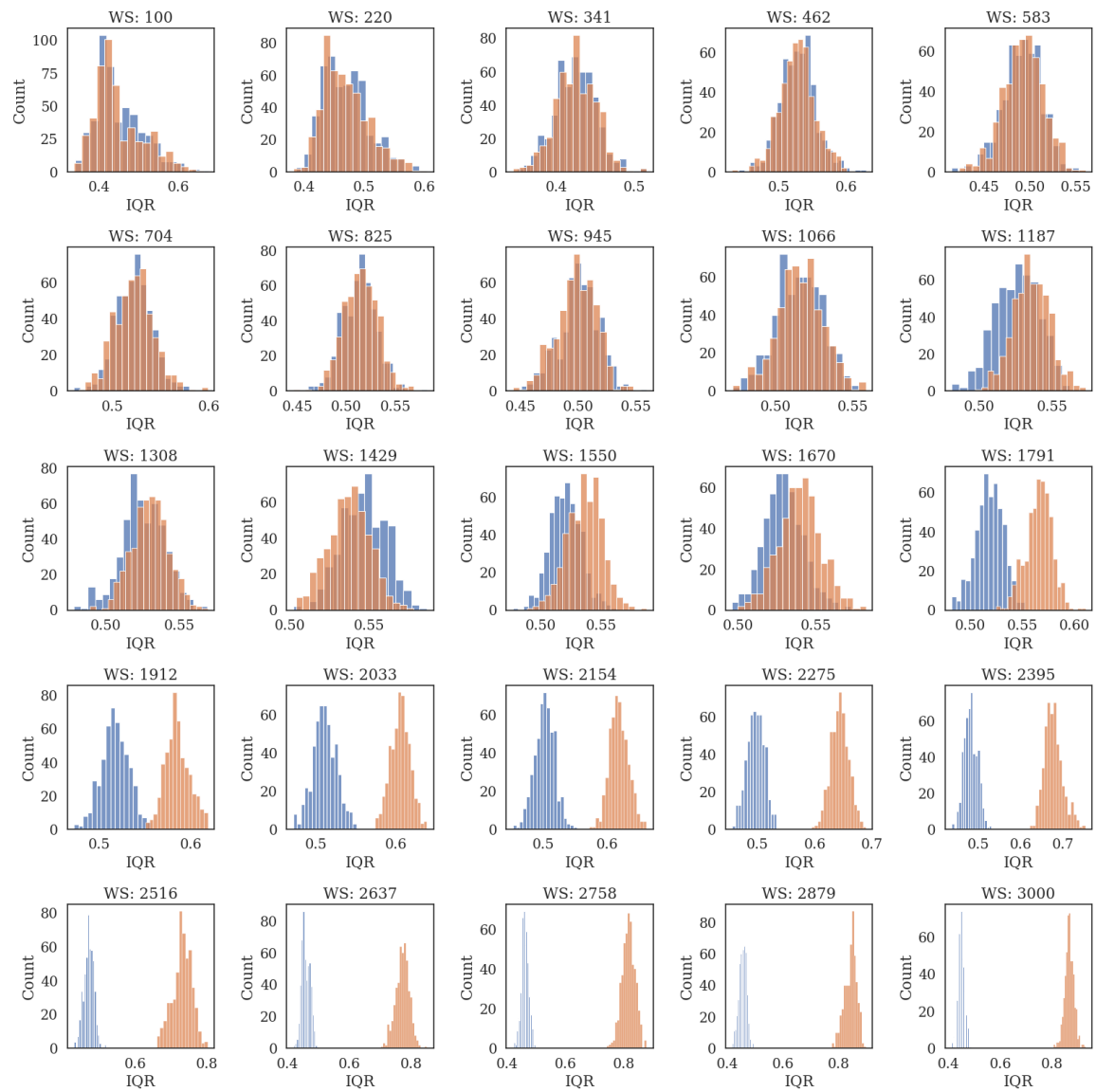


Figure 9.2. IQR histograms for varying window sizes.

$$\mathcal{R}(\theta) = E(C, \theta) = \sum_{i=0}^1 \sum_{j=0}^1 C_{ij} P(H_i | H_j, \theta) P(H_j) + \mu \theta^2, \quad (9.19)$$

where $\mu = 1.0 \times 10^{-7}$ is a normalizing coefficient. A quadratic term was chosen for the penalty term after the quadratic penalty that comes out of assuming a Gaussian prior for θ (see ridge regression in [88]). Furthermore, normalizing the expression by the cost of a correct detection (C_{11}), and setting the cost of normal operations to $C_{00} = 0$, substituting in the probability definitions, and setting $\theta = WS$, we obtain

$$E(C, WS) = \lambda(1 - P_D)P_{damage} + \gamma P_{FA}(1 - P_{damage}) + P_D P_{damage} + \mu WS^2, \quad (9.20)$$

where $\gamma = C_{10}/C_{11}$ and $\lambda = C_{01}/C_{11}$.

With this Bayes Risk formulation as our cost model, we can explore the effect of the cost parameters, WS , our threshold for false alarms and misses, and the prior probability. Figure 9.3 illustrates the effect of the prior probability of damage, the coefficients λ and γ , and WS . The cost has been normalized to be 1.0 at $WS = 0$ for these plots for ease of visualization. The top row of plots considers the variation of γ which represents the ratio of the cost of a false alarm with respect to the cost of a correct detection. So γ essentially models the cost of unnecessary downtime due to a false alarm. For these plots, λ is fixed at 2. As shown in the top row of plots, the cost functions are relatively insensitive to γ given a fixed prior probability of damage. Most of the variation is coming from varying P_{damage} . When $P_{damage} = 0.1$, which means that the prior probability of damage is low (e.g., based on historical trends), the best detector is the one with the smallest WS because, as described above, there is a quadratic cost associated with WS . However, as P_{damage} increases, the minimum cost moves to the right of the x-axis, indicating that more samples are preferable, even if an additional cost is incurred, because the probability of finding damage is higher. The second row of plots shows the sensitivity to λ which

represents the cost of failure. A large λ is associated with significant consequences of failure, such as it would be the case in a primary structural component. All the plots in the second row favor a non-zero WS except when λ is smallest. There are two valleys in all of the plots and the absolute minimum varies depending on the value of λ . When $\lambda = 0.5$, in other words the consequence of missing to detect damage is half of the cost of correctly detecting it, the optimal detector is that with the smallest WS. This scenario is clearly a very unlikely one since the cost of an error is typically higher than the cost of detecting failure before it happens. As λ increases, the minimum cost moves from $WS = 0$ to the first valley and then to the second valley as P_{damage} also increases. This result is reasonable as the cost associated with the consequences of undetected failure overcome the cost of data collection to a certain extent. As we know from Section 9.4.2, there are diminishing returns beyond a certain WS value (i.e., the p-value plateaus in Figure 9.1). This result stands in contrast to the conclusion reached from Figure 9.1 where cost/risk was not considered and the best detector corresponded to the one that collected the most data. To understand why there is variability between $WS = 1000$ and $WS = 2000$, we can look at the trends of the IQR distributions in Figure 9.4 (as a reminder, these were obtained through bootstrapping as described in Section 9.3). As shown, the IQRs only start to deviate at around the 1000 WS mark and they exhibit an oscillatory pattern because the variances are themselves oscillatory (due to their dependence on the state space response). The first minimum in the cost function corresponds to the first evident deviation of the IQRs around $WS = 1200$. After that, the IQRs tend to be hard to distinguish up until the $WS = 1600$ mark where we find the second minimum of our cost. It is expected that this transition region will vary based on the type of damage and the specific trajectories being considered so a robust detector should consider the second minimum after the oscillatory behavior has settled.

The same data are plotted in Figure 9.5 for a fixed $WS = 1330$ to study the effect of γ, λ , and the probability of false alarm. The value of γ increases as we move from top to bottom, and the value of λ increases as we go from left to right. In all cases, $\gamma > 1.0, \lambda > 1.0$ to study the case where the cost of an error is higher than the cost of a correct detection. In all of these plots

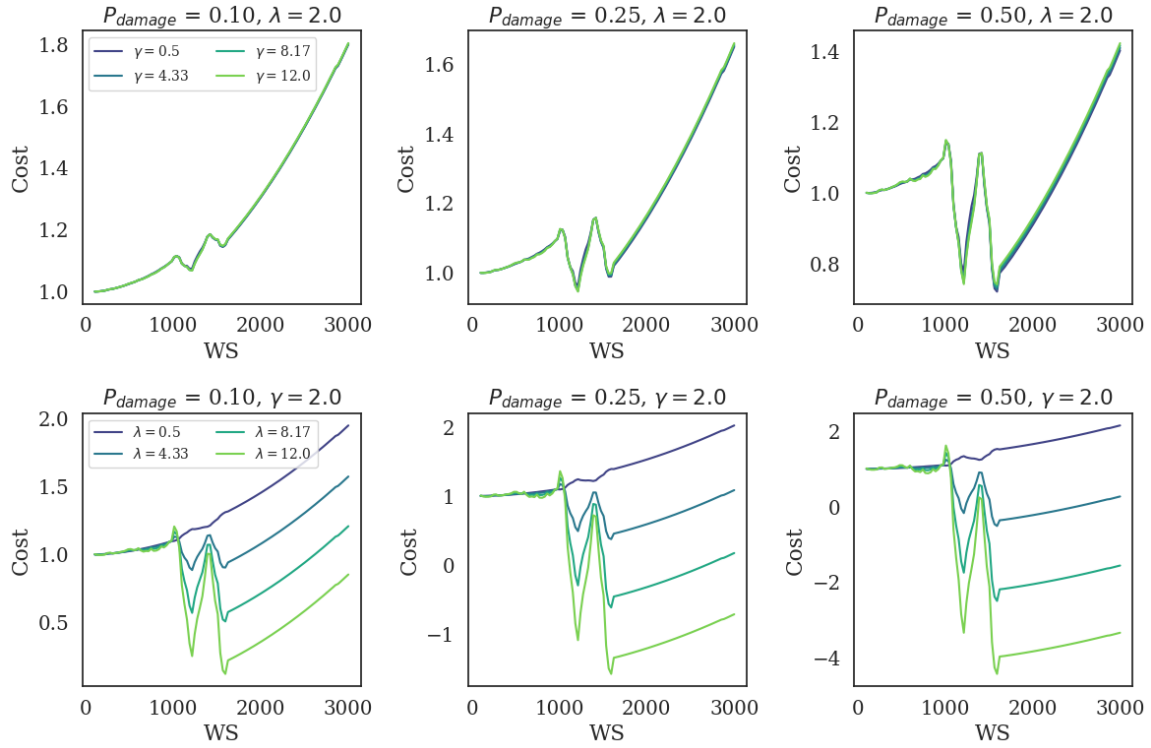


Figure 9.3. Cost vs. WS as a function of prior probability of damage and λ .

the color of the curves represents the prior probability of damage and the minimum cost of each curve is marked with $\times$. If we start by looking at the top plot, we can see that the minimum cost changes as we vary P_{damage} . In other words, the admissible rate of false alarms grows as P_{damage} increases. The minimums are clustered around low false alarm rates when λ is smaller while they quickly move to larger false alarm rates when we increase λ . This trend is due to the cost of an undetected failure growing. In other words, knowing that damage is likely, it is preferable to have a large rate of false alarms since undetected damage is very costly. As we move from top to bottom, the cost of unnecessary downtime increases. One extreme case is the lower left corner where $\gamma = 20.0, \lambda = 1.5$. This case represents an extreme where the cost of unnecessary downtime far exceeds the cost of undetected failure (also knowing that the prior probability of damage is low). In this case, having a very low rate of false alarms is preferable. However, as the probability of damage increases, the plots on the right hand column show that even when the cost of making mistakes is high, a high rate of false alarms is still preferable. When there is a

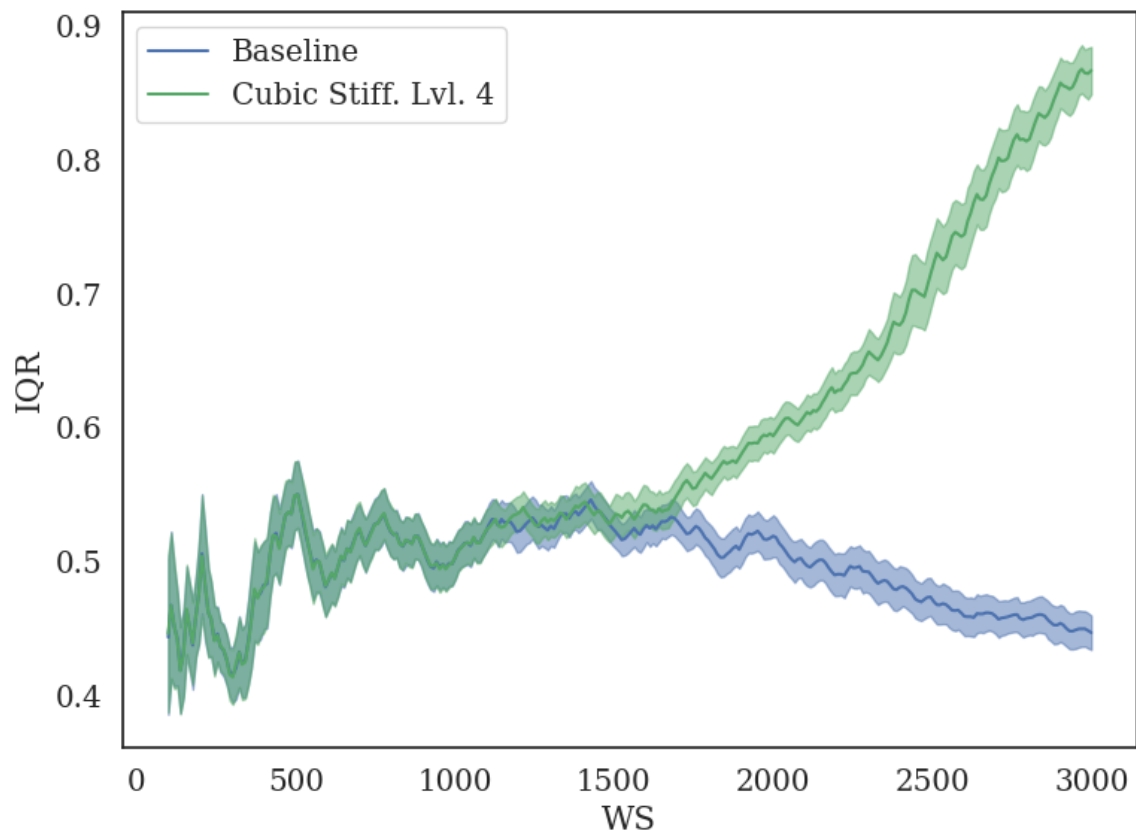


Figure 9.4. IQR mean and one standard deviation band around it for the baseline (blue) and the damaged case (green).

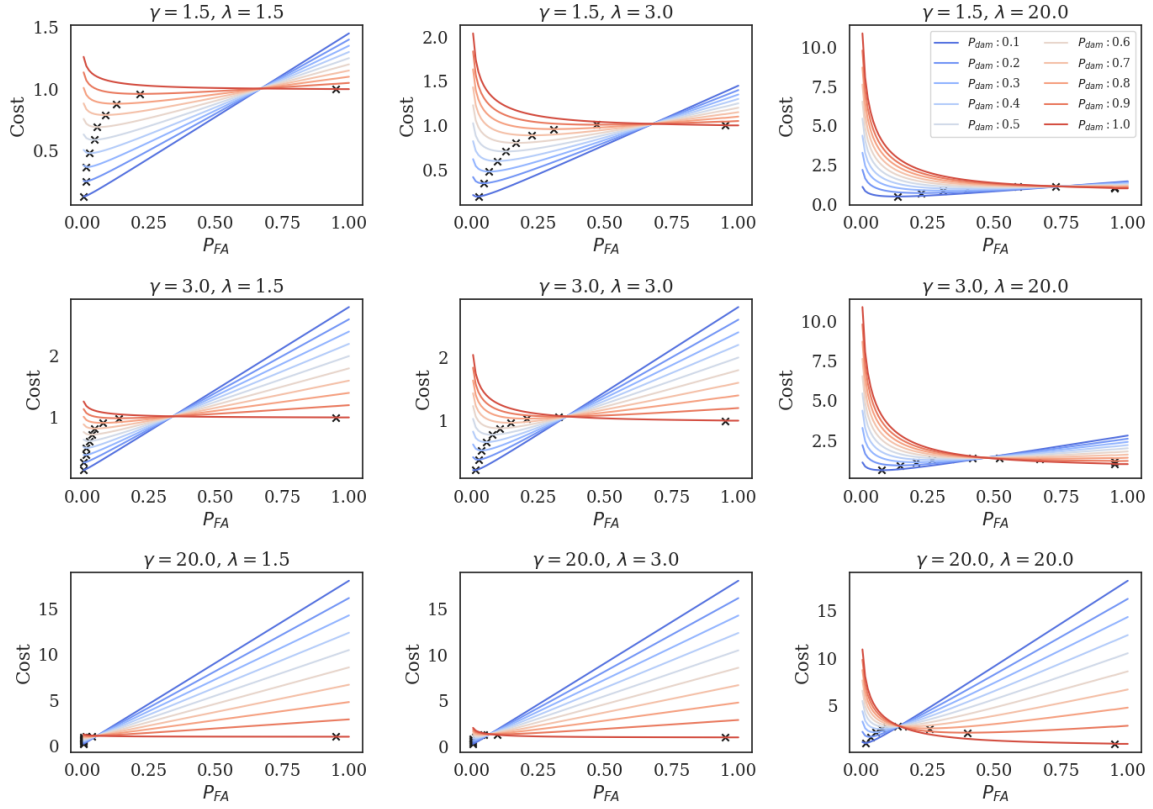


Figure 9.5. Cost vs. probability of false alarm as a function of prior probability of damage, λ , and γ .

large cost associated with unnecessary downtime, the cost is proportional to the probability of false alarm, except for the case where $P_{damage} = 1$ which is when damage is certain.

9.4.4 Detector performance when considering multiple damage types

Sections 9.4.2 and 9.4.2 only considered the progressive damage case (gradual cubic stiffness change) as an exemplar. In this section we consider all of the damage cases described in Chapter 8. We still apply the same Wald-type hypothesis test previously described. However, the power of the test cannot be easily computed across multiple damage cases so we evaluate the detector performance by fixing $P_{FA} = 0.05$ and generating Monte Carlo samples and then counting the true positives (TP), true negatives (TN), false positives (FP) and false negatives (FN), where TP is equivalent to P_D . The results across all cases are presented in the confusion

matrices shown in Figure 9.6 as a function of WS. This information is further summarized by computing the F1 score as a function of WS, as shown in Figure 9.7. As illustrated, the F1 score improves with increased WS, leading to the same conclusion drawn in Section 9.4.2 that the best detector is the one that collects the most data. However, as shown in Section 9.4.3, this result changes when the cost/risk information is considered.

9.5 Model updating

Section 9.4 demonstrated an approach for designing a detector based on desired properties defined by cost or risk of making certain types of mistakes. Once damage or domain shift has been detected, a logical next step is to update the model to reflect the current state of the monitored structure. To demonstrate the strategy for model updating, we will focus on the example with a large change to the cubic stiffness coefficient (Cubic Stiff Lvl. 4 in Chapter 8). This example was selected because it induced the largest error out of all the cases considered. Once damage has been detected using the approach previously presented, the existing sp-SNGP model is re-trained for a few iterations using the new data being observed. For this training, a window size of 3500 time steps from the new data being measured at the boundary of the isolated region was used to update the model. The model updates are performed by the classical backpropagation technique which involves stochastic gradient descent (SGD). The authors in [156] provide an interpretation of SGD with momentum as a posterior sampler in a Bayesian inference context. In this way, the model updating strategy employed in this work can be interpreted as a form of Bayesian inference with adaptive parameters via the ADAM optimizer.

The backpropagation algorithm was run for 10,000 epochs (it took 15 minutes on a single CPU) to update the network weights. As described in Chapter 8, there are two separate networks that model the nonlinear potential energy (V_{nl}) and the nonlinear damping ($\mathcal{J}_{nl}$) respectively. Figure 9.8 shows a close-up of the nonlinear potential energy and damping time histories and their respective predictive intervals obtained by applying different initial conditions than those

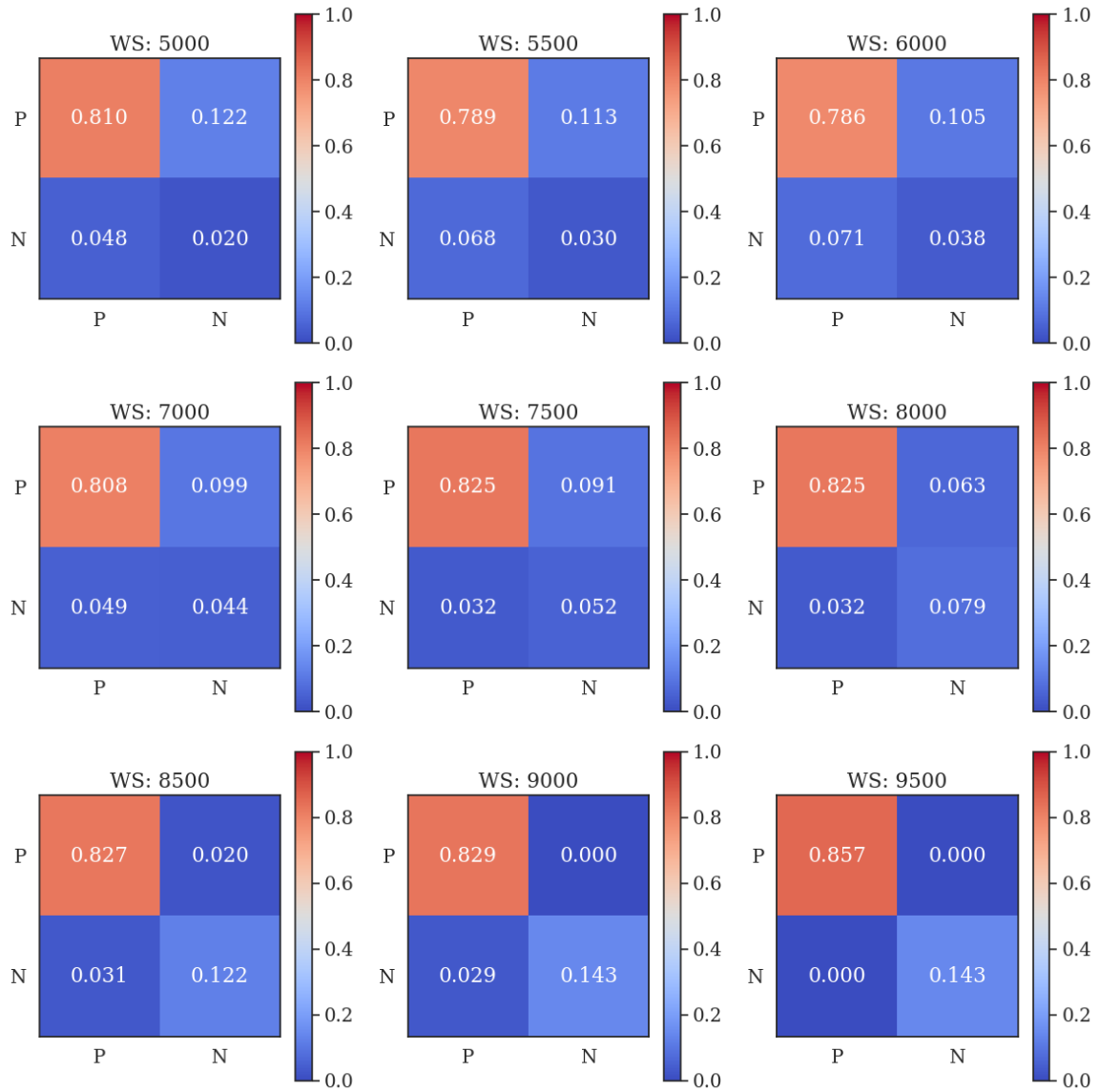


Figure 9.6. Confusion matrix across all damage cases as a function of window size. Predicted condition shown across horizontal axis and actual conditions shown across vertical axis.

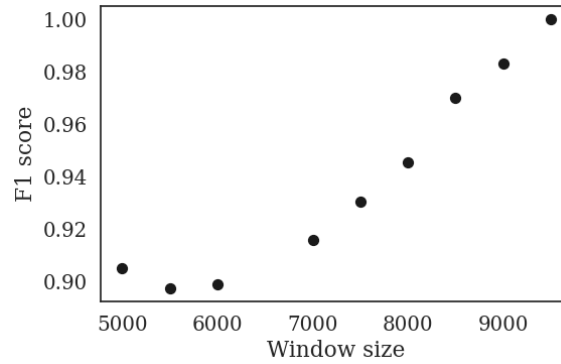


Figure 9.7. F1 score as a function of window size.

used for training. As shown, the model’s uncertainty is significantly reduced after the update. Figure 9.9 illustrates the distributions of log-variances over 10,000 time steps. As shown, the log-variance is reduced several orders of magnitude in both cases. This result is consistent with the previous findings when running sp-SNGP which are that reduced uncertainty is associated with in-distribution data. This update has effectively shifted the domain of the ML model so that the updated data is now in-distribution.

As mentioned, there are two separate networks that need to be updated. In the absence of prior knowledge about the type of nonlinearity induced by the damage, both networks would need to be updated. The distribution of non-zero log-transformed weights for both networks are presented in Figure 9.10 before and after the model update. As shown, the distributions before and after are fairly close to each other. To evaluate whether or not they are the same distribution, we employed a Kolmogorov-Smirnov (KS) two-sample test. The p-value for the nonlinear potential weight distributions was 0.030 while for the nonlinear damping weight distributions it was 0.626. Considering a threshold of $p = 0.05$, we reject the null hypothesis of the KS test that the nonlinear potential weights were drawn from the same distribution, while we fail to reject the null hypothesis for the nonlinear damping, leading to the conclusion that the two samples were drawn from different distributions. In other words, the difference between the nonlinear potential weight distributions is statistically significant while it is not for the nonlinear damping. This result suggests that the domain shift observed in the Cubic Stiff. Lvl. 4 was caused by a

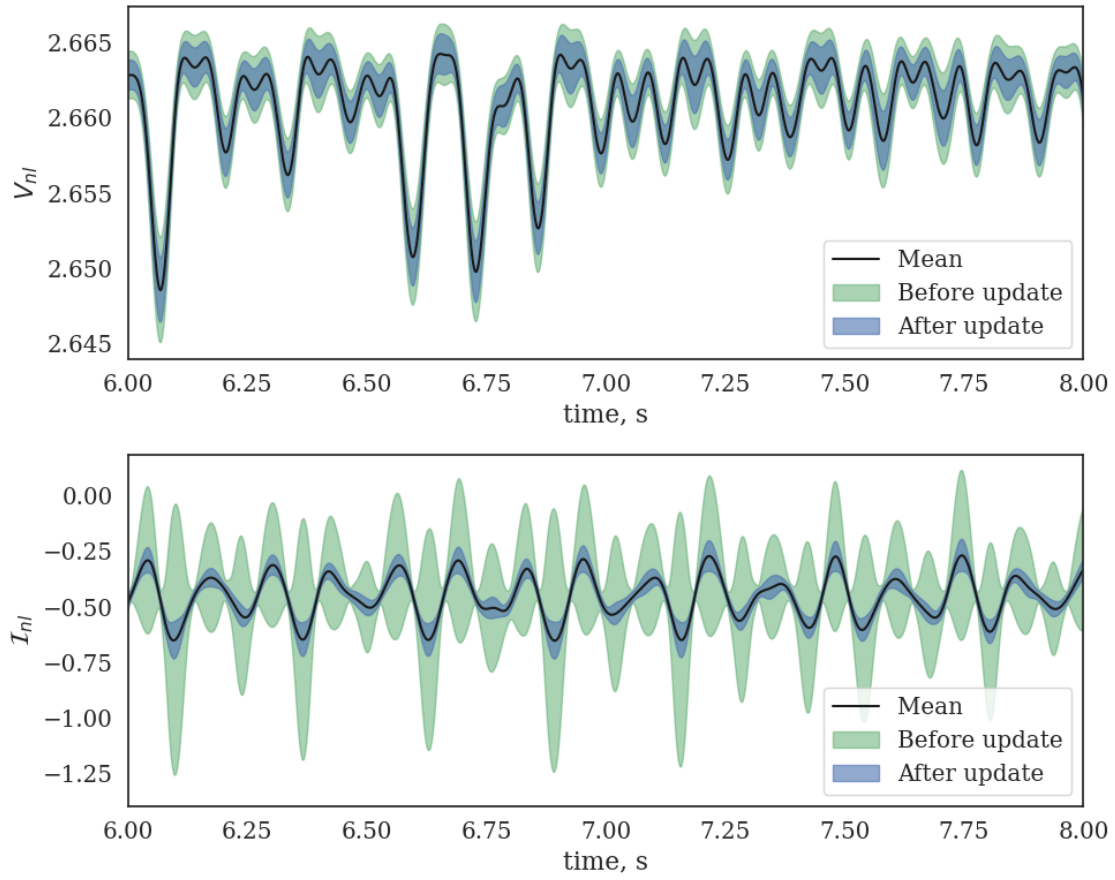


Figure 9.8. Nonlinear potential (top) and damping (bottom) time histories. One standard deviation intervals shown.

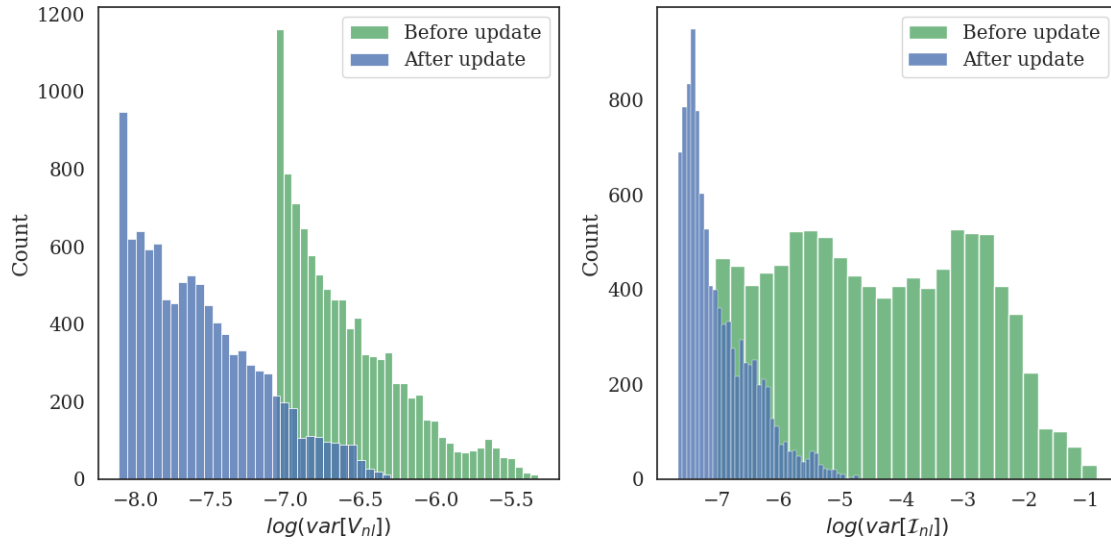


Figure 9.9. Histograms of log-variance of nonlinear potential (left) and damping (right) network weights.

change to the nonlinear potential, which is consistent with the physical change (an update to the cubic stiffness term), which only contributes to the nonlinear potential function. This result can be used to infer the type of change to the system from the data in the absence of any knowledge informing the physical mechanism that is causing the damage symptoms. Future work could extend this work to fully parametric formulation that enables parameter identification.

9.6 Conclusion

This chapter presented a demonstration of a strategy for cost/risk-informed decision-making based on features extracted from an uncertainty-aware ML model. The ML model predictive variance was used as the basis for defining a feature to detect damage. Based on the work presented in Chapter 9 and the discussion in the present chapter, it was shown that the order statistics of the log-transformed variance of the damping time histories could be used to detect domain shift which in turn was used as an indicator of damage. While Chapter 9 was mostly focused on maximizing probability of detection, the present chapter demonstrated a more practical case where cost and risk were considered via a Bayes risk formulation, establishing a

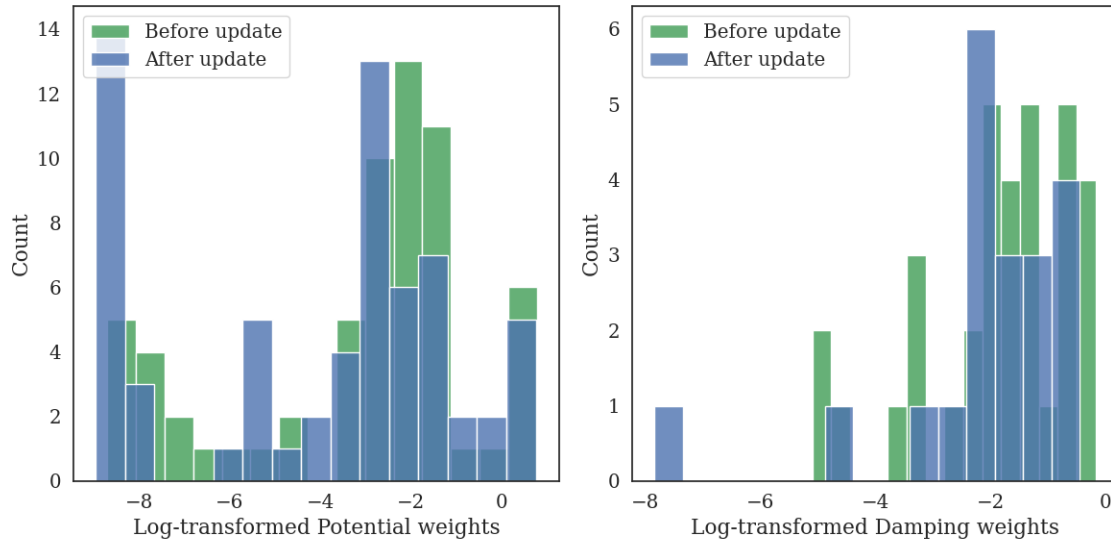


Figure 9.10. Histograms of log-variance of nonlinear potential (left) and damping (right).

workflow for probabilistic reasoning. As shown, the optimal number of samples to be collected depends on the cost associated with each decision taken. While this work focused on a single example with multiple damage cases, the workflow developed is extensible and applicable to other structures, and other cost paradigms. Moreover, this chapter demonstrated how the model can be updated to new data, and how that updates reduce the predictive uncertainty significantly. Furthermore, the statistics of the updated weights were used to infer an understanding of the physical mechanism driving the change. Future work will focus on extensions to understand the physics of the damage mechanism with more detail.

Chapter 10

Conclusions and Future Research

The research presented in this dissertation established a foundation for a group of physics-constrained and structure-preserving ML algorithms for use in system identification and SHM applications. Although the majority of the examples demonstrated in this dissertation were concerned with nonlinear structural dynamics, there were a couple of applications demonstrating the applicability of the general family of algorithms to domains outside this field. This work considered a subset of possible applications with a focus towards enabling the use of this technology in practical applications. To this end, practical considerations were addressed, such as the challenge of partial observability, the common family of problems with isolated nonlinearities, as well as model order reduction. Deployment of these systems in real applications where reliability and trustworthiness are paramount led to a focus on uncertainty quantification through various methods. Both epistemic and aleatoric uncertainty were considered, relying on different types of strategies to deal with these different sources. Lastly, a demonstration of a damage detection framework was presented to motivate the adoption of the proposed approach in SHM systems.

Chapter 2 introduced the use of physics-constrained ML models for forecasting of pit corrosion growth and lithium-ion battery discharge capacity capability. In both cases, a Bayesian neural network was used to model both random and epistemic uncertainty in the models to enable an implicit embedding of uncertainty associated with unmodeled physics (e.g., microstructural

effects). Chapter 3 introduced the concept of a Hamiltonian digital twin to learn ML models from data originating from nonlinear systems. This chapter provides the foundation for the subsequent chapters. Chapter 4 addressed a common challenge with structure-preserving and physics-constrained ML models that require access to a complete set of generalized coordinates to enforce physics by learning a diffeomorphic mapping from time-delayed embeddings of partial measurements. Chapter 5 leveraged an isolated formulation for systems with localized nonlinearities (such as joints or localized damage) to increase the efficiency of the ML models by having them focus on learning a deviatoric term. Chapter 6 addressed the efficiency challenge by first performing a linear projection and leveraging Lagrangian operator inference to allow the ML model to focus on learning a closure term in a reduced space. This approach was demonstrated to be advantageous over trying to learn the nonlinearity directly without a linear model baseline. Chapter 7 dealt with observational noise coming from sensors by demonstrating robustness to additive white noise and presenting a strategy for defining predictive intervals from noisy predictions. Chapter 8 extended uncertainty quantification to deal with epistemic uncertainty by embedding distance preservation of the state space in the ML model's latent space and adding a GP layer that provides distance awareness. Lastly, Chapter 9 demonstrated the use of this uncertainty-aware ML model for damage detection through the application of a Bayes Risk workflow that considers cost and/or risk to inform decision-making.

Future work will focus on extensions of this work to other types of structural nonlinearities as well as improvements to the reduced-order modeling techniques. There are many families of problems that have not been considered yet, such as dynamic systems with multiple basins of attraction, including chaotic problems and other systems with stable bifurcations. Beyond structural dynamics, the proposed framework can be applied to other Hamiltonian systems, such as material modeling or thermo-mechanical problems. All of these areas can be explored in future work. And as the field of ML is rapidly evolving, new techniques should be adopted if they provide significant advantages over the algorithms currently being used.

A few conclusions and lessons learned are summarized below:

- Inclusion of physical constraints and structure-preservation leads to more robust ML models but has a non-trivial computational cost, often requiring more resources for the computation of gradients. As hardware technology evolves, special focus should be given to this family of problems to speed up the computation and adoption of these types of ML models.
- Most of the nonlinear dynamics problems considered in this research dealt with systems with relatively low underlying dimensionality, being excited by harmonic or band-limited random vibration inputs. Problems where many more modes are excited (such as shock) may need special consideration and future work could focus on these types of problems.
- Throughout this work, the amount of data used for training was kept relatively small (for most problems, only a single realization was used). This result was made possible by the physical constraints but may vary depending on the problem complexity. In this work, it was demonstrated that the trained ML models were able to extrapolate within their in-distribution domain but this may not be the case in problems with multiple basins of attraction (such as isolas).
- A general workflow was presented for the treatment of uncertainty quantification in ML models through either conformal regression or uncertainty-awareness constraints. However, there is more work to be done in this area to provide a completely generalizable approach for trustworthiness in ML systems, but we believe that the foundations presented in this research provide a path forward in this area.

Appendix A

Chapter 2 Appendix

A.1 Hamiltonian network: autoencoder hyperparameters

The autoencoder hyperparameters were obtained by running a grid search using the platform Weights and Biases [18]. The validation loss plotted as a function of epochs is shown in Figure A.1. For all of these runs, the nonlinear activation function was the Swish function [204]. Table A.1 summarizes the results for all the iterations considered. The names of the iterations were automatically generated by Weights and Biases. Based on these results, the autoencoder used had 6 layers and 14 nodes per layer.



Figure A.1. Validation loss as a function of epochs during training.

Table A.1. Summary of hyperparameter grid search results.

Name	State	Runtime (s)	Layers	Nodes	epoch	test_loss
vague-firebrand-2	finished	4393	4	14	20000	0.050
helpful-fire-8	finished	4745	6	22	20000	0.021
crisp-snow-1	finished	4302	4	10	20000	0.013
swept-meadow-7	finished	4724	6	18	20000	0.010
fast-donkey-4	finished	4427	4	22	20000	0.006
robust-cherry-3	finished	4390	4	18	20000	0.004
wise-bee-5	finished	4724	6	10	20000	0.004
wobbly-snow-6	finished	4733	6	14	20000	0.003

Appendix B

Chapter 3 Appendix

B.1 Invertible neural network: hyperparameters

The hyperparameters used for the the autoencoder, either the INN (i.e., s_A, s_B, t_A, t_B) or the MLP, are listed in Table B.1 and those used for the energy networks (i.e., T_{nn}, U_{nn}, I_{nn}) are listed in Table B.2. For all networks, a linear activation function was used for the output layer. The hyperparameters were found by performing a grid search and choosing the hyperparameters that resulted in the lowest validation loss. For all examples, the total number of epochs was 150,000. The rest of the training settings are provided in Table B.3.

Table B.1. Hyperparameters for autoencoder

Example	Units per layer	Activation	AE type
Oscillator A	{2, 2, 2, 2}	tanh	INN
Oscillator B	{6, 6, 6, 6}	tanh	INN
Oscillator C	{6, 6, 6, 6}	tanh	INN
Membrane	{3, 3, 3, 3, 3, 3}	tanh	INN
HBRB	{1, 1, 1, 1, 1, 1}	tanh	INN
S4 Beam	{6, 6, 6, 6}	tanh	MLP

B.2 Comparison of MLP and INN autoencoders

Based on the findings in section 4.5.1, the sensitivity of the decoding operation to the latent space was explored. To this end, random noise was added to the latent space predictions

Table B.2. Hyperparameters for energy networks

Example	Units per layer	Activation
Oscillator A	{8, 8, 8, 8}	swish
Oscillator B	{8, 8, 8, 8}	swish
Oscillator C	{8, 8, 8, 8}	swish
Membrane	{80, 64, 32, 18}	swish
HBRB	{18, 18, 18, 18}	swish
S4 Beam	{8, 8, 8, 8}	tanh

Table B.3. Training settings

Example	T_{train}	T_{val}	T_{test}	η	N_b
Oscillator A	6200	500	8300	1×10^{-4}	250
Oscillator B	6200	500	8300	1×10^{-4}	250
Oscillator C	6200	500	8300	1×10^{-4}	250
Membrane	3000	500	3900	1×10^{-4}	250
HBRB	6500	500	8500	1×10^{-4}	250
S4 Beam	5000	500	8500	1×10^{-4}	250

for case C. The signal to noise ratio was 50. These noisy latent space time histories were passed to either the INN or the MLP decoders to recover the physical space measurements. The results are shown in Figure B.1. As illustrated, the INN reconstruction is more noisy than the MLP reconstruction. The MSE for the MLP predictions when the latent space was perturbed was 1.24 times larger than the MSE of the original latent space, while it was 5.90 times larger for the INN. This result is consistent with the findings in section 4.5.1 where small errors in the time integration in latent space were amplified by the INN decoder. A possible explanation for this result may be the fact that the MLP decoder is trained using the predicted data, not the reference data, so that it learns to be less sensitive to inaccuracies in the latent space predictions. On the other hand, the INN is only trained using the forward transformation and so it does not learn this inherent robustness that the MLP inherits. Improved training strategies for the INN may be explored in future research.

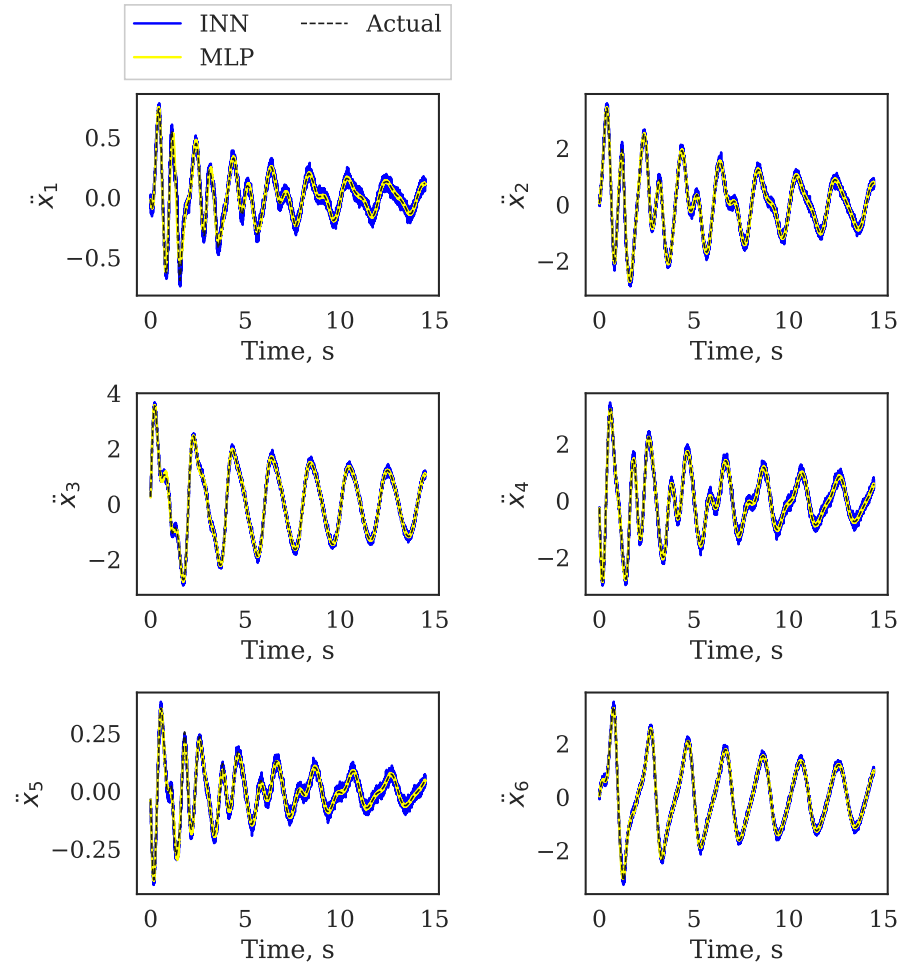


Figure B.1. Decoded oscillator time histories. Dashed black is the actual response, blue is the INN decoding, and yellow is the MLP decoding.

Bibliography

- [1] David E Acuña and Marcos E Orchard. Particle-filtering-based failure prognosis via sigma-points: Application to lithium-ion battery state-of-charge monitoring. *Mechanical Systems and Signal Processing*, 85:827–848, 2017.
- [2] Sondipon Adhikari. *Damping models for structural vibration*. University of Cambridge, 2001.
- [3] Toufik Aggab, Manuel Avila, Pascal Vrignat, and Frédéric Kratz. Unifying model-based prognosis with learning-based time-series prediction methods: application to li-ion battery. *IEEE Systems Journal*, 15(4):5245–5254, 2021.
- [4] KV Akpanyung and RT Loto. Pitting corrosion evaluation: a review. In *Journal of Physics: Conference Series*, volume 1378, page 022088. IOP Publishing, 2019.
- [5] Matthew S. Allen, Daniel Rixen, Maarten van der Seijs, Paolo Tiso, Thomas Abrahamsson, and Randall L. Mayes. *Substructuring in Engineering Dynamics*, volume 594. Springer International Publishing, 2020.
- [6] Christine Allen-Blanchette, Sushant Veer, Anirudha Majumdar, and Naomi Ehrich Leonard. LagNetVip: A Lagrangian neural network for video prediction. In *Proceedings of the AAAI Conference on Physics-Guided AI*, 2020.
- [7] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety, 2016.
- [8] Vladimir Igorevich Arnol’d. *Mathematical Methods of Classical Mechanics*, volume 60. Springer Science & Business Media, 2013.
- [9] Joar Axås, Mattia Cenedese, and George Haller. Correction to: Fast data-driven model reduction for nonlinear dynamical systems. *Nonlinear Dynamics*, 111(9):7959–7959, February 2023.
- [10] Jonghyuk Baek, Jiun-Shyan Chen, and Kristen Susuki. A neural network-enhanced reproducing kernel particle method for modeling strain localization. *International Journal for Numerical Methods in Engineering*, 123(18):4422–4454, 2022.

- [11] Guangxing Bai, Yunsheng Su, Maliha Maisha Rahman, and Zequn Wang. Prognostics of lithium-ion batteries using knowledge-constrained machine learning and kalman filtering. *Reliability Engineering & System Safety*, 231:108944, 2023.
- [12] Joseph Bakarji, Kathleen Champion, J Nathan Kutz, and Steven L Brunton. Discovering governing equations from partial measurements with deep delay autoencoders. *Proc. Math. Phys. Eng. Sci.*, 479(2276), August 2023.
- [13] Rina Foygel Barber, Emmanuel J. Candes, Aaditya Ramdas, and Ryan J. Tibshirani. Predictive inference with the jackknife+, 2020.
- [14] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.
- [15] Peter Benner, Pawan Goyal, Boris Kramer, Benjamin Peherstorfer, and Karen Willcox. Operator inference for non-intrusive model reduction of systems with non-polynomial nonlinear terms. *Computer Methods in Applied Mechanics and Engineering*, 372:113433, 2020.
- [16] Peter Benner, Serkan Gugercin, and Karen Willcox. A survey of projection-based model reduction methods for parametric dynamical systems. *SIAM review*, 57(4):483–531, 2015.
- [17] Peter Benner, Mario Ohlberger, Albert Cohen, and Karen Willcox. *Model reduction and approximation: theory and algorithms*. Computational Science and Engineering. Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017.
- [18] Lukas Biewald. Experiment tracking with weights and biases, 2020. Software available from wandb.com.
- [19] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks, 2015.
- [20] Josh Bongard and Hod Lipson. Automated reverse engineering of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 104(24):9943–9948, 2007.
- [21] Matthew S. Bonney and David Wagg. Historical perspective of the development of digital twins. pages 15–20. Springer, 2022.
- [22] I Borg and P J F Groenen. *Modern multidimensional scaling*. Springer Series in Statistics. Springer, New York, NY, 2 edition, August 2005.
- [23] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018.
- [24] G. Bradski. The OpenCV Library. *Dr. Dobb’s Journal of Software Tools*, 2000.

- [25] Matthew RW Brake. *The mechanics of jointed structures: Recent research and open challenges for developing predictive models for structural dynamics*. Springer, 2017.
- [26] MRW Brake, CW Schwingshackl, and P Reuß. Observations of variability and repeatability in jointed structures. *Mechanical Systems and Signal Processing*, 129:282–307, 2019.
- [27] Adam R. Brink, David A. Najera-Flores, and Cari Martinez. The neural network collocation method for solving partial differential equations. *Neural Comput. Appl.*, 33(11):5591–5608, jun 2021.
- [28] Steven L. Brunton, Joshua L. Proctor, and J. Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.
- [29] Jonathan Buck, Eric Jayson, and David Najera. Using optimization to improve the quasi-isotropic status quo. In *Proceedings of the American Society for Composites: Thirty-First Technical Conference*, 2016.
- [30] Jonathan Buck, David Najera, Doug Melville, and Eric Jayson. Prediction of fiber-reinforced composite material properties using collaborative filtering techniques. In *Proceedings of the American Society for Composites: Thirty-third Technical Conference*, 2018.
- [31] Jacques Buffle, Zeshi Zhang, and Konstantin Startchev. Metal flux and dynamic speciation at (bio) interfaces. part i: Critical evaluation and compilation of physicochemical parameters for complexes with simple ligands and fulvic/humic substances. *Environmental science & technology*, 41(22):7609–7620, 2007.
- [32] Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β -vae, 2018.
- [33] Francesco Cadini, Claudio Sbarufatti, Francesco Cancelliere, and Marco Giglio. State-of-life prognosis and diagnosis of lithium-ion batteries by data-driven particle filters. *Applied energy*, 235:661–672, 2019.
- [34] Kevin Carlberg, Ray Tuminaro, and Paul Boggs. Preserving Lagrangian structure in nonlinear model reduction with application to structural dynamics. *SIAM Journal on Scientific Computing*, 37(2):B153–B184, 2015.
- [35] P Castaldo, B Palazzo, and A Mariniello. Effects of the axial force eccentricity on the time-variant structural reliability of aging rc cross-sections subjected to chloride-induced corrosion. *Engineering Structures*, 130:261–274, 2017.
- [36] Marcantonio Catelani, Lorenzo Ciani, Romano Fantacci, Gabriele Patrizi, and Benedetta Picano. Remaining useful life estimation for prognostics of lithium-ion batteries based on recurrent neural network. *IEEE Transactions on Instrumentation and Measurement*, 70:1–11, 2021.

- [37] M. Cenedese, J. Axås, H. Yang, M. Eriten, and G. Haller. Data-driven nonlinear model reduction to spectral submanifolds in mechanical systems. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 380(2229), June 2022.
- [38] Mattia Cenedese, Joar Axås, Bastian Bäuerlein, Kerstin Avila, and George Haller. Data-driven modeling and prediction of non-linearizable dynamics via spectral submanifolds. *Nature Communications*, 13(1), February 2022.
- [39] Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink. Fusing physics-based and deep learning models for prognostics. *Reliability Engineering & System Safety*, 217:107961, 2022.
- [40] Chaochao Chen and Michael Pecht. Prognostics of lithium-ion batteries using model-based and data-driven methods. In *Proceedings of the IEEE 2012 Prognostics and System Health Management Conference (PHM-2012 Beijing)*, pages 1–6. IEEE, 2012.
- [41] Lei Chen, Hao Wei Zhang, Lin Yun Liang, Zhe Liu, Yue Qi, Peng Lu, James Chen, and Long-Qing Chen. Modulation of dendritic patterns during electrodeposition: A nonlinear phase-field model. *Journal of Power Sources*, 300:376–385, 2015.
- [42] Long-Qing Chen. Phase-field models for microstructure evolution. *Annual review of materials research*, 32(1):113–140, 2002.
- [43] Renyi Chen and Molei Tao. Data-driven prediction of general hamiltonian dynamics via learning exactly-symplectic maps. In *International Conference on Machine Learning*, pages 1717–1727. PMLR, 2021.
- [44] Tianping Chen and Hong Chen. Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems. *IEEE Transactions on Neural Networks*, 6(4):911–917, 1995.
- [45] Wei Chen, Debasish Jana, Aryan Singh, Mengshi Jin, Mattia Cenedese, Giancarlo Kosova, Matthew R.W. Brake, Christoph W. Schwingshackl, Satish Nagarajaiah, Keegan J. Moore, and Jean-Philippe Noël. Measurement and identification of the nonlinear dynamics of a jointed structure using full-field data, part i: Measurement of nonlinear dynamics. *Mechanical Systems and Signal Processing*, 166:108401, 2022.
- [46] Zhengdao Chen, Jianyu Zhang, Martin Arjovsky, and Léon Bottou. Symplectic recurrent neural networks. In *International Conference on Learning Representations*, 2019.
- [47] Francisco Chinesta, Elias Cueto, Emmanuelle Abisset-Chavanne, Jean Louis Duval, and Fouad El Khaldi. Virtual, digital and hybrid twins: A new paradigm in data-based engineering and engineered data. *Archives of Computational Methods in Engineering*, 27:105–134, 1 2020.

- [48] Hoang K Chu and Mitsuhiro Hayashibe. Discovering interpretable dynamics by sparsity promotion on energy and the lagrangian. *IEEE Robotics and Automation Letters*, 5(2):2154–2160, 2020.
- [49] Roy R. Craig and Mervyn C.C. Bampton. Coupling of substructures for dynamic analyses. *AIAA Journal*, 6:1313–1319, 1968.
- [50] Harald Cramér. On the composition of elementary errors. *Scandinavian Actuarial Journal*, 1928(1):13–74, 1928.
- [51] Miles Cranmer, Sam Greydanus, Stephan Hoyer, Peter Battaglia, David Spergel, and Shirley Ho. Lagrangian neural networks. In *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*, 2020.
- [52] Sándor CSöRgő and Julian J. Faraway. The Exact and Asymptotic Distributions of Cramér-Von Mises Statistics. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1):221–234, 12 2018.
- [53] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314, December 1989.
- [54] Aymen Danoun, Etienne Prulière, and Yves Chemisky. Thermodynamically consistent recurrent neural networks to predict non linear behaviors of dissipative materials subjected to non-proportional loading paths. *Mechanics of Materials*, 173:104436, 2022.
- [55] Tareen Dawood, Chen Chen, Baldeep S. Sidhu, Bram Ruijsink, Justin Gould, Bradley Porter, Mark K. Elliott, Vishal Mehta, Christopher A. Rinaldi, Esther Puyol-Antón, Reza Razavi, and Andrew P. King. Uncertainty aware training to improve deep learning model calibration for classification of cardiac MR images. *Medical Image Analysis*, 88:102861, 2023.
- [56] Eduardo A Barros de Moraes, Hadi Salehi, and Mohsen Zayernouri. Data-driven failure prediction in brittle materials: A phase field-based machine learning framework. *Journal of Machine Learning for Modeling and Computing*, 2(1), 2021.
- [57] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *The Journal of Machine Learning Research*, 17(1):2909–2913, 2016.
- [58] Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matthew D. Hoffman, and Rif A. Saurous. Tensorflow distributions. *CoRR*, abs/1711.10604, 2017.
- [59] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp, 2017.

- [60] Guangzhong Dong and Jingwen Wei. A physics-based aging model for lithium-ion battery with coupled chemical/mechanical degradation mechanisms. *Electrochimica Acta*, 395:139133, 2021.
- [61] J.R. Dormand and P.J. Prince. A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980.
- [62] Austin Downey, Yu-Hui Lui, Chao Hu, Simon Laflamme, and Shan Hu. Physics-based prognostics of lithium-ion battery using non-linear least squares with dynamic bounds. *Reliability Engineering & System Safety*, 182:1–12, 2019.
- [63] Qian Fang, Xuankang Mou, and Shibin Li. A physics-informed neural network based on mixed data sampling for solving modified diffusion equations. *Scientific Reports*, 13(1), February 2023.
- [64] Charles R Farrar and Keith Worden. An introduction to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 365:303–315, 2 2007.
- [65] Charles R Farrar and Keith Worden. *Structural Health Monitoring*. John Wiley & Sons, Nashville, TN, November 2012.
- [66] Michael Feldman. Hilbert transform in vibration analysis. *Mechanical Systems and Signal Processing*, 25:735–802, 4 2011.
- [67] Hugo Festjens, Gaël Chevallier, and Jean luc Dion. A numerical tool for the design of assembled structures under dynamic loads. *International Journal of Mechanical Sciences*, 75:170–177, 10 2013.
- [68] Yevgeniya Filanova, Igor Pontes Duff, Pawan Goyal, and Peter Benner. An operator inference oriented approach for linear mechanical systems. *Mechanical Systems and Signal Processing*, 200:110620, 2023.
- [69] Vincent Fortuin, Mark Collier, Florian Wenzel, James Allingham, Jeremiah Liu, Dustin Tran, Balaji Lakshminarayanan, Jesse Berent, Rodolphe Jenatton, and Effrosyni Kokiopoulou. Deep classifiers with label noise modeling and distance awareness, 2022.
- [70] Stefania Fresca, Luca Dede’, and Andrea Manzoni. A comprehensive deep learning-based approach to reduced order modeling of nonlinear time-dependent parametrized PDEs. *Journal of Scientific Computing*, 87:1–36, 2021.
- [71] Stefania Fresca and Andrea Manzoni. POD-DL-ROM: Enhancing deep learning-based reduced order models for nonlinear parametrized PDEs by proper orthogonal decomposition. *Computer Methods in Applied Mechanics and Engineering*, 388:114181, 2022.
- [72] M. I. Friswell, J. E. T. Penny, and S. D. Garvey. Using linear model reduction to investigate the dynamics of structures with local non-linearities. *Mechanical Systems and Signal Processing*, 9:317–328, 1995.

- [73] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning, 2016.
- [74] Han Gao, Jian-Xun Wang, and Matthew J Zahr. Non-intrusive model reduction of large-scale, nonlinear dynamical systems using deep learning. *Physica D: Nonlinear Phenomena*, 412:132614, 2020.
- [75] Paul Gardner, Mattia Dal Borgo, Valentina Ruffini, Aidan J. Hughes, Yichen Zhu, and David J. Wagg. Towards the development of an operational digital twin. *Vibration*, 3, 2020.
- [76] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995.
- [77] Andrew Gelman, Daniel Simpson, and Michael Betancourt. The prior can often only be understood in the context of the likelihood. *Entropy*, 19(10):555, 2017.
- [78] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterton, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.
- [79] Alex Graves. Practical variational inference for neural networks. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.
- [80] Samuel Greydanus, Misko Dzamba, and Jason Yosinski. Hamiltonian neural networks. *Advances in neural information processing systems*, 32, 2019.
- [81] Anthony Gruber and Irina Tezaur. Canonical and noncanonical Hamiltonian operator inference. *Computer Methods in Applied Mechanics and Engineering*, 416:116334, 2023.
- [82] Fabian M. Gruber and Daniel J. Rixen. Dual craig-bampton component mode synthesis method for model order reduction of nonclassically damped linear systems. *Mechanical Systems and Signal Processing*, 111:678–698, 10 2018.
- [83] Jian Guo, Zhaojun Li, and Michael Pecht. A bayesian approach for li-ion battery capacity fade modeling and cycles to failure prognostics. *Journal of Power Sources*, 281:173–184, 2015.
- [84] Jayesh K Gupta, Kunal Menda, Zachary Manchester, and Mykel Kochenderfer. Structured mechanical models for robot learning and control. In *Learning for Dynamics and Control*, pages 328–337. PMLR, 2020.
- [85] George Haller and Sten Ponsioen. Nonlinear normal modes and spectral submanifolds: existence, uniqueness and use in model reduction. *Nonlinear Dynamics*, 86(3):1493–1534, August 2016.

- [86] George Haller and Sten Ponsioen. Nonlinear normal modes and spectral submanifolds: existence, uniqueness and use in model reduction. *Nonlinear Dynamics*, 86:1493–1534, 11 2016.
- [87] CM Hansson. The impact of corrosion on society. *Metallurgical and Materials Transactions A*, 42:2952–2962, 2011.
- [88] Trevor Hastie, Robert Tibshirani, and J H Friedman. *The elements of statistical learning*. Springer series in statistics. Springer, New York, NY, 2 edition, December 2009.
- [89] Xiaolong He and Jiun-Shyan Chen. Thermodynamically consistent machine-learned internal state variable approach for data-driven modeling of path-dependent materials. *Computer Methods in Applied Mechanics and Engineering*, 402:115348, 2022. A Special Issue in Honor of the Lifetime Achievements of J. Tinsley Oden.
- [90] Yangzi He and Shabnam J. Semnani. Machine learning based modeling of path-dependent materials for finite element analysis. *Computers and Geotechnics*, 156:105254, 2023.
- [91] Jonathan Heek, Anselm Levskaya, Avital Oliver, Marvin Ritter, Bertrand Rondepierre, Andreas Steiner, and Marc van Zee. Flax: A neural network library and ecosystem for JAX, 2023.
- [92] Rosemarie Helmerich and Astrid Zunkel. Partial collapse of the berlin congress hall on may 21st, 1980. *Engineering failure analysis*, 43:107–119, 2014.
- [93] Oliver Hennigh, Susheela Narasimhan, Mohammad Amin Nabian, Akshay Subramaniam, Kaustubh Tangsali, Zhiwei Fang, Max Rietmann, Wonmin Byeon, and Sanjay Choudhry. Nvidia simnet™: An ai-accelerated multi-physics simulation framework. In *International Conference on Computational Science*, pages 447–461. Springer, 2021.
- [94] Jan S Hesthaven and Stefano Ubbiali. Non-intrusive reduced order modeling of nonlinear problems using neural networks. *Journal of Computational Physics*, 363:55–78, 2018.
- [95] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.
- [96] T. L. Hill, A. Cammarano, S. A. Neild, and D. A. W. Barton. Identifying the significance of nonlinear normal modes. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473:20160789, 3 2017.
- [97] Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory - COLT '93*. ACM Press, 1993.

- [98] Yoshito Hirata and Kevin Judd. Constructing dynamical systems with specified symbolic dynamics. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 15(3), July 2005.
- [99] Lior Hirschfeld, Kyle Swanson, Kevin Yang, Regina Barzilay, and Connor W. Coley. Uncertainty quantification using neural networks for molecular property prediction. *Journal of Chemical Information and Modeling*, 60(8):3770–3780, 2020. PMID: 32702986.
- [100] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1989–1998. PMLR, 10–15 Jul 2018.
- [101] Philip Holmes, John L Lumley, Gahl Berkooz, and Clarence W Rowley. *Turbulence, Coherent Structures, Dynamical Systems and Symmetry*. Cambridge University Press, 2012.
- [102] Joonki Hong, Dongheon Lee, Eui-Rim Jeong, and Yung Yi. Towards the swift prediction of the remaining useful life of lithium-ion batteries with end-to-end deep learning. *Applied energy*, 278:115646, 2020.
- [103] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.
- [104] Chao Hu, Byeng D Youn, and Jaesik Chung. A multiscale framework with extended kalman filter for lithium-ion battery soc and capacity estimation. *Applied Energy*, 92:694–704, 2012.
- [105] Zhen Hu and Sankaran Mahadevan. Global sensitivity analysis-enhanced surrogate (gsas) modeling for reliability analysis. *Structural and Multidisciplinary Optimization*, 53:501–521, 2016.
- [106] Zhen Hu and Sankaran Mahadevan. Uncertainty quantification and management in additive manufacturing: current status, needs, and opportunities. *The International Journal of Advanced Manufacturing Technology*, 93:2855–2874, 2017.
- [107] Justin Jacobs, David Najera-Flores, Adam R. Brink, and Tatiana Flanagan. Parameter estimation for dynamical systems under continuous and discontinuous gaussian noise using data assimilation techniques. In Ramin Madarshahian and Francois Hemez, editors, *Data Science in Engineering, Volume 9*, pages 135–143, Cham, 2022. Springer International Publishing.
- [108] Emily Jewell, Matthew S. Allen, Iman Zare, and Mitchell Wall. Application of quasi-static modal analysis to a finite element model and experimental correlation. *Journal of Sound and Vibration*, 479:115376, 2020.

- [109] Chen Jiang, Manuel A Vega, Michael D Todd, and Zhen Hu. Model correction and updating of a stochastic degradation model for failure prognostics of miter gates. *Reliability Engineering & System Safety*, 218:108203, 2022.
- [110] Mengshi Jin, Giancarlo Kosova, Mattia Cenedese, Wei Chen, Aryan Singh, Debasish Jana, Matthew R.W. Brake, Christoph W. Schwingshackl, Satish Nagarajaiah, Keegan J. Moore, and Jean-Philippe Noël. Measurement and identification of the nonlinear dynamics of a jointed structure using full-field data; part ii - nonlinear system identification. *Mechanical Systems and Signal Processing*, 166:108402, 2022.
- [111] Pengzhan Jin, Zhen Zhang, Aiqing Zhu, Yifa Tang, and George Em Karniadakis. Symp-nets: Intrinsic structure-preserving symplectic networks for identifying hamiltonian systems. *Neural Networks*, 132:166–179, 2020.
- [112] Pengzhan Jin, Zhen Zhang, Aiqing Zhu, Yifa Tang, and George Em Karniadakis. Symp-Nets: Intrinsic structure-preserving symplectic networks for identifying Hamiltonian systems. *Neural Networks*, 132:166–179, 2020.
- [113] Xiaowei Jin, Shengze Cai, Hui Li, and George Em Karniadakis. Nsfnets (navier-stokes flow nets): Physics-informed neural networks for the incompressible navier-stokes equations. *Journal of Computational Physics*, 426:109951, 2021.
- [114] Rong Jing, Zhimin Xi, Xiao Guang Yang, and Ed Decker. A systematic framework for battery performance estimation considering model and parameter uncertainties. *International Journal of Prognostics and Health Management*, 5(2), 2014.
- [115] Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Discovering conservation laws from data for control. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 6415–6421. IEEE, 2018.
- [116] Dmitriy Kats, Zhidong Wang, Zhengtao Gan, Wing Kam Liu, Gregory J Wagner, and Yanping Lian. A physics-informed machine learning method for predicting grain structure characteristics in directed energy deposition. *Computational Materials Science*, 202:110958, 2022.
- [117] Steven M Kay. *Fundamentals of statistical signal processing, volume II*. Prentice Hall, Philadelphia, PA, January 1998.
- [118] Jack Raymond Kayser. *The effects of corrosion on the reliability of steel girder bridges*. University of Michigan, 1988.
- [119] Gaetan Kerschen, Jean-claude Golinval, Alexander F Vakakis, and Lawrence A Bergman. The method of proper orthogonal decomposition for dynamical characterization and order reduction of mechanical systems: an overview. *Nonlinear dynamics*, 41(1):147–169, 2005.

- [120] Gaetan Kerschen, Keith Worden, Alexander F. Vakakis, and Jean-Claude Golinval. Past, present and future of nonlinear system identification in structural dynamics. *Mechanical Systems and Signal Processing*, 20:505–592, 2006.
- [121] Ehsan Kharazmi, Zhongqiang Zhang, and George E.M. Karniadakis. hp-vpinns: Variational physics-informed neural networks with domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 374:113547, 2021.
- [122] Parisa Khodabakhshi and Karen E Willcox. Non-intrusive data-driven model reduction for differential–algebraic equations derived from lifting transformations. *Computer Methods in Applied Mechanics and Engineering*, 389:114296, 2022.
- [123] Byol Kim, Chen Xu, and Rina Foygel Barber. Predictive inference is free with the jackknife+-after-bootstrap, 2020.
- [124] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.
- [125] Sara Kohtz, Yanwen Xu, Zhuoyuan Zheng, and Pingfeng Wang. Physics-informed machine learning model for battery state of health prognostics using partial charging segments. *Mechanical Systems and Signal Processing*, 172:109002, 2022.
- [126] Y Kondo. Prediction of fatigue crack initiation life based on pit growth. *Corrosion*, 45(1):7–11, 1989.
- [127] K H Kraemer, G Datseris, J Kurths, I Z Kiss, J L Ocampo-Espindola, and N Marwan. A unified and automated approach to attractor reconstruction. *New Journal of Physics*, 23(3):033017, mar 2021.
- [128] Vladimir M. Krasnopolsky and Michael S. Fox-Rabinovitz. Complex hybrid models combining deterministic and machine learning components for numerical climate modeling and weather prediction. *Neural Networks*, 19(2):122–134, 2006. Earth Sciences and Environmental Applications of Computational Intelligence.
- [129] Dimitri Krattiger, Long Wu, Martin Zacharczuk, Martin Buck, Robert J. Kuether, Matthew S. Allen, Paolo Tiso, and Matthew R.W. Brake. Interface reduction for hurty/craig-bampton substructured models: Review and improvements. *Mechanical Systems and Signal Processing*, 114:579–603, 1 2019.
- [130] Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael W Mahoney. Characterizing possible failure modes in physics-informed neural networks. *Advances in Neural Information Processing Systems*, 34:26548–26560, 2021.
- [131] Robert J. Kuether and Matthew R. W. Brake. Instantaneous frequency and damping from transient ring-down data. In Matt Allen, Randall L. Mayes, and Daniel Rixen, editors, *Dynamics of Coupled Structures, Volume 4*, pages 253–263, Cham, 2016. Springer International Publishing.

- [132] Robert J. Kuether and David A. Najera. Parameter estimation of joint models using global optimization. In Matthew S. Allen, Randall L. Mayes, and Daniel Jean Rixen, editors, *Dynamics of Coupled Structures, Volume 4*, pages 29–39, Cham, 2017. Springer International Publishing.
- [133] Sanjay Lall, Petr Krysl, and Jerrold E Marsden. Structure-preserving model reduction for mechanical systems. *Physica D: Nonlinear Phenomena*, 184(1-4):304–318, 2003.
- [134] Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes, 2017.
- [135] Kookjin Lee and Kevin T Carlberg. Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders. *Journal of Computational Physics*, 404:108973, 2020.
- [136] Kookjin Lee and Kevin T Carlberg. Deep conservation: A latent-dynamics model for exact satisfaction of physical conservation laws. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 277–285, 2021.
- [137] Kookjin Lee, Nathaniel Trask, and Panos Stinis. Machine learning structure preserving brackets for forecasting irreversible processes. *Advances in Neural Information Processing Systems*, 34:5696–5707, 2021.
- [138] Kookjin Lee, Nathaniel Trask, and Panos Stinis. Structure-preserving sparse identification of nonlinear dynamics for data-driven modeling. In *Mathematical and Scientific Machine Learning*, pages 65–80. PMLR, 2022.
- [139] Penghua Li, Zijian Zhang, Qingyu Xiong, Baocang Ding, Jie Hou, Dechao Luo, Yujun Rong, and Shuaiyong Li. State-of-health estimation and remaining useful life prediction for the lithium-ion battery based on a variant long short term memory neural network. *Journal of power sources*, 459:228069, 2020.
- [140] Xiang Li, Wei Zhang, and Qian Ding. Cross-domain fault diagnosis of rolling element bearings using deep generative neural networks. *IEEE Transactions on Industrial Electronics*, 66(7):5525–5534, 2019.
- [141] Linxia Liao and Felix Köttig. Review of hybrid prognostics approaches for remaining useful life prediction of engineered systems, and an application to battery life prediction. *IEEE Transactions on Reliability*, 63(1):191–207, 2014.
- [142] Chen Lin, Haihui Ruan, and San-Qiang Shi. Phase field study of mechanico-electrochemical corrosion. *Electrochimica Acta*, 310:240–255, 2019.
- [143] Chun-Pang Lin, Javier Cabrera, Fangfang Yang, Man-Ho Ling, Kwok-Leung Tsui, and Suk-Joo Bae. Battery state of health modeling and remaining useful life prediction through time series model. *Applied Energy*, 275:115338, 2020.

- [144] Jeremiah Zhe Liu, Zi Lin, Shreyas Padhy, Dustin Tran, Tania Bedrax-Weiss, and Balaji Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness, 2020.
- [145] Jian Liu and Ziqiang Chen. Remaining useful life prediction of lithium-ion batteries based on health indicator and gaussian process regression model. *Ieee Access*, 7:39474–39484, 2019.
- [146] Kailong Liu, Yunlong Shang, Quan Ouyang, and Widanalage Dhammika Widanage. A data-driven approach with uncertainty quantification for predicting future capacities and remaining useful life of lithium-ion battery. *IEEE Transactions on Industrial Electronics*, 68(4):3170–3180, 2021.
- [147] Yixuan Liu, Zhen Hu, Michael Todd, and Chao Hu. Data-driven remaining useful life estimation using gaussian mixture models. In *AIAA Scitech 2021 Forum*, page 1487, 2021.
- [148] Yuefeng Liu, Guangquan Zhao, and Xiyuan Peng. Deep learning prognostics for lithium-ion battery based on ensembled long short-term memory networks. *IEEE Access*, 7:155130–155142, 2019.
- [149] Yuefeng Liu, Guangquan Zhao, Xiyuan Peng, and Cong Hu. Lithium-ion battery remaining useful life prediction with long short-term memory recurrent neural network. In *Annual Conference of the PHM Society*, volume 9, 2017.
- [150] Tyler J Loftus, Benjamin Shickel, Matthew M Ruppert, Jeremy A Balch, Tezcan Ozrazgat-Baslanti, Patrick J Tighe, Philip A Efron, William R Hogan, Parisa Rashidi, Gilbert R Upchurch, Jr, and Azra Bihorac. Uncertainty-aware deep learning in healthcare: A scoping review. *PLOS Digit. Health*, 1(8):e0000085, August 2022.
- [151] Roland T Loto and Cleophas A Loto. Corrosion and protection of facilities and infrastructures in telecommunications industry-a review. In *IOP Conference Series: Materials Science and Engineering*, volume 1107, page 012014. IOP Publishing, 2021.
- [152] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deepnet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3):218–229, 2021.
- [153] Ye Lu, Hengyang Li, Lei Zhang, Chanwook Park, Satyajit Mojumder, Stefan Knapik, Zhongsheng Sang, Shaoqiang Tang, Daniel W. Apley, Gregory J. Wagner, and Wing Kam Liu. Convolution hierarchical deep-learning neural networks (c-HiDeNN): finite elements, isogeometric analysis, tensor decomposition, and beyond. *Computational Mechanics*, 72(2):333–362, May 2023.
- [154] Yu Hui Lui, Meng Li, Austin Downey, Sheng Shen, Venkat Pavan Nemani, Hui Ye, Collette VanElzen, Gaurav Jain, Shan Hu, Simon Laflamme, and Chao Hu. Physics-based prognostics of implantable-grade lithium-ion battery for remaining useful life prediction. *Journal of Power Sources*, 485:229327, 2021.

- [155] Michael Lutter, Christian Ritter, and Jan Peters. Deep Lagrangian networks: Using physics as model prior for deep learning. In *International Conference on Learning Representations*, 2018.
- [156] Stephan M. t, Matthew D. Hoffman, and David M. Blei. Stochastic gradient descent as approximate bayesian inference. *Journal of Machine Learning Research*, 18(134):1–35, 2017.
- [157] David J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, 05 1992.
- [158] Weijie Mai, Soheil Soghrati, and Rudolph G Buchheit. A phase field model for simulating the pitting corrosion. *Corrosion Science*, 110:157–166, 2016.
- [159] MAPIE Development Team. MAPIE - model agnostic prediction interval estimator, 2023.
- [160] Celia Martín Vicario, Dalia Rodríguez Salas, Andreas Maier, Stefan Hock, Joji Kuramatsu, Bernd Kallmuenzer, Florian Thamm, Oliver Taubmann, Hendrik Ditt, Stefan Schwab, Arnd Dörfler, and Iris Muehlen. Uncertainty-aware deep learning for trustworthy prediction of long-term outcome after endovascular thrombectomy. *Sci. Rep.*, 14(1), March 2024.
- [161] Carianne Martinez, David A. Najera-Flores, Adam R. Brink, D. Dane Quinn, Eleni Chatzi, and Stephanie Forrest. Confronting domain shift in trained neural networks. In Luca Bertinetto, João F. Henriques, Samuel Albanie, Michela Paganini, and Gül Varol, editors, *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, volume 148 of *Proceedings of Machine Learning Research*, pages 176–192. PMLR, 11 Dec 2021.
- [162] Carianne Martinez, Kevin M. Potter, Matthew D. Smith, Emily A. Donahue, Lincoln Collins, John P. Korbin, and Scott A. Roberts. Segmentation certainty through uncertainty: Uncertainty-refined binary volumetric segmentation under multifactor domain shift. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 484–486, 2019.
- [163] Allen T. Mathis, Nidish N. Balaji, Robert J. Kuether, Adam R. Brink, Matthew R. W. Brake, and D. Dane Quinn. A Review of Damping Models for Structures With Mechanical Joints¹. *Applied Mechanics Reviews*, 72(4):040802, 07 2020.
- [164] Marios Mattheakis, David Sondak, Akshunna S. Dogra, and Pavlos Protopapas. Hamiltonian neural networks for solving equations of motion. *Phys. Rev. E*, 105:065305, Jun 2022.
- [165] Andrew McClellan, Joseph Lorenzetti, Marco Pavone, and Charbel Farhat. A physics-based digital twin for model predictive control of autonomous unmanned aerial vehicle landing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 380, 8 2022.

- [166] Madhav Mishra, Jesper Martinsson, Matti Rantatalo, and Kai Goebel. Bayesian hierarchical model-based prognostics for lithium-ion batteries. *Reliability Engineering & System Safety*, 172:25–35, 2018.
- [167] David Montes de Oca Zapiain, James A Stewart, and Rémi Dingreville. Accelerating phase-field-based microstructure evolution predictions via surrogate models trained by machine learning methods. *npj Computational Materials*, 7(1):3, 2021.
- [168] Griffin Mooers, Michael Pritchard, Tom Beucler, Jordan Ott, Galen Yacalis, Pierre Baldi, and Pierre Gentine. Assessing the potential of deep learning for emulating cloud superparameterization in climate models with real-geography boundary conditions. *Journal of Advances in Modeling Earth Systems*, 13(5):e2020MS002385, 2021.
- [169] Nitis Mukhopadhyay. *Probability and Statistical Inference*. CRC Press, 2000.
- [170] David Najera, Eric Jayson, Thomas Board, and Doug Melville. A framework for bayesian calibration of fiber-reinforced composite material properties. In *Proceedings of the American Society for Composites: Thirty-fifth Technical Conference*, 2020.
- [171] David A. Najera, Christopher I. Van Damme, and Victoria Harris. Model reduction of geometrically nonlinear structures viaphysics-informed autoencoders. In Ramin Madarshahian and Francois Hemez, editors, *Data Science in Engineering, Volume 9*, pages 145–148, Cham, 2022. Springer International Publishing.
- [172] David Najera-Flores, Justin Jacobs, D. Dane Quinn, Anthony Garland, and Michael Todd. Uncertainty-aware, structure-preserving machine learning approach for domain shift detection from nonlinear dynamic responses of structural systems. Manuscript accepted for publication in the *Journal of Risk and Uncertainty in Engineering Systems Part B: Mechanical Engineering*, 2024.
- [173] David Najera-Flores and Michael Todd. State-space reconstruction from partial observables using an invertible neural network with structure-preserving properties for nonlinear structural dynamics. Manuscript accepted for publication in *Nonlinear Dynamics*, 2024.
- [174] David A. Najera-Flores, Zhen Hu, Mayank Chadha, and Michael D. Todd. A physics-constrained bayesian neural network for battery remaining useful life prediction. *Applied Mathematical Modelling*, 122:42–59, 2023.
- [175] David A. Najera-Flores, Justin Jacobs, D. Dane Quinn, Anthony Garland, and Michael D. Todd. Uncertainty Quantification of a Machine Learning Model for Identification of Isolated Nonlinearities With Conformal Prediction. *Journal of Verification, Validation and Uncertainty Quantification*, 9(2):021005, 06 2024.
- [176] David A. Najera-Flores and Robert J. Kuether. A Study of Whole Joint Model Calibration Using Quasi-Static Modal Analysis. *Journal of Vibration and Acoustics*, 142(5):051109, 06 2020.

- [177] David A. Najera-Flores and Robert J. Kuether. Approximating nonlinear response to harmonic base-excitation using fixed-interface quasi-static modal analysis. *Journal of Sound and Vibration*, 544:117381, 2023.
- [178] David A. Najera-Flores, Jonel Ortiz, Moheimin Y. Khan, Robert J. Kuether, and Paul R. Miles. A Bayesian Multi-Fidelity Neural Network to Predict Nonlinear Frequency Backbone Curves. *Journal of Verification, Validation and Uncertainty Quantification*, 9(2):021003, 06 2024.
- [179] David A. Najera-Flores, Guofeng Qian, Zhen Hu, and Michael D. Todd. Corrosion morphology prediction of civil infrastructure using a physics-constrained machine learning method. *Mechanical Systems and Signal Processing*, 200:110515, 2023.
- [180] David A. Najera-Flores, D. Dane Quinn, Anthony Garland, Konstantinos Vlachas, Eleni Chatzi, and Michael D. Todd. A structure-preserving machine learning framework for accurate prediction of structural dynamics for systems with isolated nonlinearities. *Mechanical Systems and Signal Processing*, 213:111340, 2024.
- [181] David A. Najera-Flores and Michael D. Todd. Ensemble of numerics-informed neural networks with embedded hamiltonian constraints for modeling nonlinear structural dynamics. In Matthew R.W. Brake, Ludovic Renson, Robert J. Kuether, and Paolo Tiso, editors, *Nonlinear Structures & Systems, Volume 1*, pages 27–30, Cham, 2023. Springer International Publishing.
- [182] David A. Najera-Flores and Michael D. Todd. State space reconstruction from embeddings of partial observables in structural dynamic systems for structure-preserving data-driven methods. In Ramin Madarshahian and François Hemez, editors, *Data Science in Engineering, Volume 10*, pages 139–141, Cham, 2023. Springer Nature Switzerland.
- [183] David A. Najera-Flores and Michael D. Todd. A structure-preserving neural differential operator with embedded Hamiltonian constraints for modeling structural dynamics. *Computational Mechanics*, 72:241–252, March 2023.
- [184] David A. Najera-Flores and Michael D. Todd. A structure-preserving neural differential operator with embedded hamiltonian constraints for modeling structural dynamics. *Computational Mechanics*, 3 2023.
- [185] Venkat Nemani, Luca Biggio, Xun Huan, Zhen Hu, Olga Fink, Anh Tran, Yan Wang, Xiaoping Du, Xiaoge Zhang, and Chao Hu. Uncertainty quantification in machine learning for engineering design and health prognostics: A tutorial, 2023.
- [186] Venkat Nemani, Adam Thelen, Chao Hu, and Steve Daining. Degradation-aware ensemble of diverse predictors for remaining useful life prediction. *Journal of Mechanical Design*, 145(3):031706, 2023.
- [187] Selina SY Ng, Yinjiao Xing, and Kwok L Tsui. A naive bayes model for robust remaining useful life prediction of lithium-ion battery. *Applied Energy*, 118:114–123, 2014.

- [188] Gang Ning, Ralph E White, and Branko N Popov. A generalized cycle life model of rechargeable li-ion batteries. *Electrochimica acta*, 51(10):2012–2022, 2006.
- [189] J.P. Noël and G. Kerschen. Nonlinear system identification in structural dynamics: 10 more years of progress. *Mechanical Systems and Signal Processing*, 83:2–35, 2017.
- [190] David Opitz and Richard Maclin. Popular ensemble methods: An empirical study. *J. Artif. Int. Res.*, 11(1):169–198, jul 1999.
- [191] Niall M O’Dowd, Ramin Madarshahian, Michael Siu Hey Leung, Joseph Corcoran, and Michael D Todd. A probabilistic estimation approach for the failure forecast method using bayesian inference. *International Journal of Fatigue*, 142:105943, 2021.
- [192] Louis M. Pecora, Linda Moniz, Jonathan Nichols, and Thomas L. Carroll. A unified approach to attractor reconstruction. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 17(1):013110, 2007.
- [193] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [194] Benjamin Peherstorfer and Karen Willcox. Data-driven operator inference for nonintrusive projection-based model reduction. *Computer Methods in Applied Mechanics and Engineering*, 306:196–215, 2016.
- [195] E. P. Petrov and D. J. Ewins. Analytical Formulation of Friction Interface Elements for Analysis of Nonlinear Multi-Harmonic Vibrations of Bladed Disks . *Journal of Turbomachinery*, 125(2):364–371, 04 2003.
- [196] Elizabeth Qian, Boris Kramer, Alexandre N Marques, and Karen E Willcox. Transform & learn: A data-driven approach to nonlinear model reduction. In *AIAA Aviation Forum*, page 3707, 2019.
- [197] Elizabeth Qian, Boris Kramer, Benjamin Peherstorfer, and Karen Willcox. Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems. *Physica D: Nonlinear Phenomena*, 406:132401, 2020.
- [198] Guofeng Qian, Karnpiwat Tantratian, Lei Chen, Zhen Hu, and Michael D Todd. A probabilistic computational framework for the prediction of corrosion-induced cracking in large structures. *Nature Scientific Reports*, 12(1):20898, 2022.
- [199] D. Dane Quinn and Adam R. Brink. Global system reduction order modeling for localized feature inclusion. *Journal of Vibration and Acoustics, Transactions of the ASME*, 143:041006, 2021.

- [200] D. Dane Quinn, David A. Najera-Flores, Anthony Garland, Konstantinos Vlachas, Eleni Chatzi, and Michael D. Todd. An order reduction framework for structural systems with isolated nonlinearities. *International Design Engineering Technical Conferences and Computers and Information Engineering Conference*, 2023.
- [201] Joaquin Quionero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D. Lawrence. *Dataset Shift in Machine Learning*. The MIT Press, 2009.
- [202] A. Rahimi and B. Recht. Random features for large-scale kernel machines. *Advances in Neural Information Processing Systems 20*, pages 1177–1184, 2008.
- [203] M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.
- [204] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017.
- [205] Lei Ren, Li Zhao, Sheng Hong, Shiqiang Zhao, Hao Wang, and Lin Zhang. Remaining useful life prediction for lithium-ion battery: A deep learning approach. *Ieee Access*, 6:50587–50598, 2018.
- [206] Kevin L Rens, Terry J Wipf, and F Wayne Klaiber. Review of nondestructive evaluation techniques of civil infrastructure. *Journal of performance of constructed facilities*, 11(4):152–160, 1997.
- [207] L. Renson, G. Kerschen, and B. Cochelin. Numerical computation of nonlinear normal modes in mechanical engineering. *Journal of Sound and Vibration*, 364:177–206, 3 2016.
- [208] H. Ribera, S. Shirman, A. V. Nguyen, and N. M. Mangan. Model selection of chaotic systems from data with hidden variables using sparse data assimilation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 32(6):063101, 06 2022.
- [209] Daniel J. Rixen. A dual craig-bampton method for dynamic substructuring. *Journal of Computational and Applied Mathematics*, 168:383–391, 7 2004.
- [210] Christian Robert. *The Bayesian choice*. Springer Texts in Statistics. Springer, New York, NY, 2 edition, August 2007.
- [211] Michael Ross, David Najera-Flores, and James Freymiller. Including Mean Stress in the Effective Equivalent Stress of the ASME BPVC Elastic–Plastic Fatigue Method. *Journal of Nondestructive Evaluation, Diagnostics and Prognostics of Engineering Systems*, 5(3):031006, 05 2022.
- [212] T. Konstantin Rusch and Siddhartha Mishra. Unicornn: A recurrent model for learning very long time dependencies. *CoRR*, abs/2103.05487, 2021.

- [213] Bhaskar Saha, Kai Goebel, and Jon Christophersen. Comparison of prognostic algorithms for estimating remaining useful life of batteries. *Transactions of the Institute of Measurement and Control*, 31(3-4):293–308, 2009.
- [214] Omer San and Romit Maulik. Neural network closures for nonlinear model order reduction. *Advances in Computational Mathematics*, 44:1717–1750, 2018.
- [215] Abhinav Saxena, Jose Celaya, Edward Balaban, Kai Goebel, Bhaskar Saha, Sankalita Saha, and Mark Schwabacher. Metrics for evaluating performance of prognostic techniques. In *2008 International Conference on Prognostics and Health Management*, pages 1–17, 2008.
- [216] Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 International Conference on Prognostics and Health Management*, pages 1–9, 2008.
- [217] Alexander Schein, Kevin T Carlberg, and Matthew J Zahr. Preserving general physical properties in model reduction of dynamical systems via constrained-optimization projection. *International Journal for Numerical Methods in Engineering*, 122(14):3368–3399, 2021.
- [218] Stefan Scheiner and Christian Hellmich. Stable pitting corrosion of stainless steel as diffusion-controlled dissolution process with a sharp moving electrode boundary. *Corrosion science*, 49(2):319–346, 2007.
- [219] Michael Schmidt and Hod Lipson. Distilling free-form natural laws from experimental data. *Science*, 324(5923):81–85, 2009.
- [220] Ozan Sener, Hyun Oh Song, Ashutosh Saxena, and Silvio Savarese. Learning transferrable representations for unsupervised domain adaptation. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.
- [221] Kristen A. Severson, Peter M. Attia, Norman Jin, Nicholas Perkins, Benben Jiang, Zi Yang, Michael H. Chen, Muratahan Aykol, Patrick K. Herring, Dimitrios Fraggdakis, Martin Z. Bazant, Stephen J. Harris, William C. Chueh, and Richard D. Braatz. Data-driven prediction of battery cycle life before capacity degradation. *Nature Energy*, 4(5):383–391, March 2019.
- [222] Glenn Shafer and Vladimir Vovk. A tutorial on conformal prediction, 2007.
- [223] Harsh Sharma and Boris Kramer. Preserving lagrangian structure in data-driven reduced-order modeling of large-scale dynamical systems. *Physica D: Nonlinear Phenomena*, 462:134128, 2024.
- [224] Harsh Sharma, David A. Najera-Flores, Michael D. Todd, and Boris Kramer. Lagrangian operator inference enhanced with structure-preserving machine learning for nonintrusive

- model reduction of mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 423:116865, 2024.
- [225] Harsh Sharma, David A. Najera-Flores, Michael D. Todd, and Boris Kramer. Lagrangian operator inference enhanced with structure-preserving machine learning for nonintrusive model reduction of mechanical systems. *Computer Methods in Applied Mechanics and Engineering*, 423:116865, 2024.
 - [226] Harsh Sharma, Zhu Wang, and Boris Kramer. Hamiltonian operator inference: Physics-preserving learning of reduced-order models for canonical hamiltonian systems. *Physica D: Nonlinear Phenomena*, 431:133122, 2022.
 - [227] Sheng Shen, Mohammadkazem Sadoughi, Meng Li, Zhengdao Wang, and Chao Hu. Deep convolutional neural networks with ensemble learning and transfer learning for capacity estimation of lithium-ion batteries. *Applied Energy*, 260:114296, 2020.
 - [228] Zhong-Hui Shen, Jian-Jun Wang, Jian-Yong Jiang, Sharon X Huang, Yuan-Hua Lin, Ce-Wen Nan, Long-Qing Chen, and Yang Shen. Phase-field modeling and machine learning of electric-thermal-mechanical breakdown of polymer-based dielectrics. *Nature communications*, 10(1):1843, 2019.
 - [229] Junchuan Shi, Alexis Rivera, and Dazhong Wu. Battery health management using physics-informed machine learning: Online degradation modeling and remaining useful life prediction. *Mechanical Systems and Signal Processing*, 179:109347, 2022.
 - [230] Aabhas Singh, Mitchell Wall, Matthew S. Allen, and Robert J. Kuether. Spider configurations for models with discrete iwan elements. In Gaetan Kerschen, M. R. W. Brake, and Ludovic Renson, editors, *Nonlinear Structures and Systems, Volume I*, pages 25–38, Cham, 2020. Springer International Publishing.
 - [231] Seho Son, Siheon Jeong, Eunji Kwak, Jun-hyeong Kim, and Ki-Yong Oh. Integrated framework for soh estimation of lithium-ion batteries using multiphysics features. *Energy*, 238:121712, 2022.
 - [232] Karin Stacke, Gabriel Eilertsen, Jonas Unger, and Claes Lundstrom. Measuring domain shift for deep learning in histopathology. *IEEE Journal of Biomedical and Health Informatics*, 25(2):325–336, February 2021.
 - [233] Janusz Stafiej, Dung Di Caprio, and Łukasz Bartosik. Corrosion-passivation processes in a cellular automata based simulation study. *The Journal of Supercomputing*, 65:697–709, 2013.
 - [234] Per Ståhle and Eskil Hansen. Phase field modelling of stress corrosion. *Engineering Failure Analysis*, 47:241–251, 2015.
 - [235] Mark G Stewart and Ali Al-Harthy. Pitting corrosion and structural reliability of corroding rc structures: Experimental data and probabilistic analysis. *Reliability engineering & system safety*, 93(3):373–382, 2008.

- [236] Marcus Stoffel, Franz Bamer, and Bernd Markert. Deep convolutional neural networks in structural dynamics under consideration of viscoplastic material behaviour. *Mechanics Research Communications*, 108:103565, 2020.
- [237] N. Sukumar and Ankit Srivastava. Exact imposition of boundary conditions with distance functions in physics-informed deep neural networks. *Computer Methods in Applied Mechanics and Engineering*, 389:114333, 2022.
- [238] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI’16, page 2058–2065. AAAI Press, 2016.
- [239] R. Swischuk, B. Kramer, C. Huang, and K. Willcox. Learning physics-based reduced-order models for a single-injector combustion process. *AIAA Journal*, 58:6:2658–2672, 2020.
- [240] Robert Szalai, David Ehrhardt, and George Haller. Nonlinear model identification and spectral submanifolds for multi-degree-of-freedom mechanical vibrations. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 473, 2017.
- [241] Piyush Tagade, Krishnan S Hariharan, Sanoop Ramachandran, Ashish Khandelwal, Arunava Naha, Subramanya Mayya Kolake, and Seong Ho Han. Deep gaussian process regression for lithium-ion battery health prognosis and degradation mode diagnosis. *Journal of Power Sources*, 445:227281, 2020.
- [242] Floris Takens. Detecting strange attractors in turbulence. In David Rand and Lai-Sang Young, editors, *Dynamical Systems and Turbulence, Warwick 1980*, pages 366–381, Berlin, Heidelberg, 1981. Springer Berlin Heidelberg.
- [243] K. E. Tatsis, V. K. Dertimanis, C. Papadimitriou, E. Lourens, and E. N. Chatzi. A general substructure-based framework for input-state estimation using limited output measurements. *Mechanical Systems and Signal Processing*, 150, 3 2021.
- [244] Adam Thelen, Meng Li, Chao Hu, Elena Bekyarova, Sergey Kalinin, and Mohan Sanghadasa. Augmented model-based framework for battery remaining useful life prediction. *Applied Energy*, 324:119624, 2022.
- [245] Adam Thelen, Yu Hui Lui, Sheng Shen, Simon Laflamme, Shan Hu, Hui Ye, and Chao Hu. Integrating physics-based modeling and machine learning for degradation diagnostics of lithium-ion batteries. *Energy Storage Materials*, 2022.
- [246] Adam Thelen, Xiaoge Zhang, Olga Fink, Yan Lu, Sayan Ghosh, Byeng D. Youn, Michael D. Todd, Sankaran Mahadevan, Chao Hu, and Zhen Hu. A comprehensive review of digital twin — part 1: modeling and twinning enabling technologies. *Structural and Multidisciplinary Optimization*, 65:354, 12 2022.

- [247] Adam Thelen, Xiaoge Zhang, Olga Fink, Yan Lu, Sayan Ghosh, Byeng D. Youn, Michael D. Todd, Sankaran Mahadevan, Chao Hu, and Zhen Hu. A comprehensive review of digital twin—part 2: roles of uncertainty quantification and optimization, a battery digital twin, and perspectives. *Structural and Multidisciplinary Optimization*, 66:1, 1 2023.
- [248] MD Todd, M Leung, and J Corcoran. A probability density function for uncertainty quantification in the failure forecast method. In *Proc. 9th European workshop on structural health monitoring*. BINDT, 2018.
- [249] Zheming Tong, Jiazhi Miao, Shuiguang Tong, and Yingying Lu. Early prediction of remaining useful life for lithium-ion batteries based on a hybrid machine learning method. *Journal of Cleaner Production*, 317:128265, 2021.
- [250] George Tsialiamanis, David J. Wagg, Nikolaos Dervilis, and Keith Worden. On generative models as the basis for digital twins. *Data-Centric Engineering*, 2, 2021.
- [251] L. C. Uzal, G. L. Grinblat, and P. F. Verdes. Optimal reconstruction of dynamical systems: A noise amplification approach. *Phys. Rev. E*, 84:016223, Jul 2011.
- [252] C. Van Damme, D.A. Najera-Flores, and R. Kuether. Application of JAX to compute gradients within harmonic balance solvers. IMAC XLI, 2023.
- [253] Linda C. van der Gaag and Silja Renooij. On the sensitivity of probabilistic networks to reliability characteristics. In Bernadette Bouchon-Meunier, Giulianella Coletti, and Ronald R. Yager, editors, *Modern Information Processing*, pages 395–405. Elsevier Science, Amsterdam, 2006.
- [254] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008.
- [255] Toby van Gastelen, Wouter Edeling, and Benjamin Sanderse. Energy-conserving neural network for turbulence closure modeling. *arXiv preprint arXiv:2301.13770*, 2023.
- [256] Leopold Vehovar, V Kuhar, and A Vehovar. Hydrogen-assisted stress-corrosion of pre-stressing wires in a motorway viaduct. *Engineering Failure Analysis*, 5(1):21–27, 1998.
- [257] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

- [258] Konstantinos Vlachas, Anthony Garland, D.Dane Quinn, and Eleni Chatzi. Parametric reduced-order modeling for component-oriented treatment and localized nonlinear feature inclusion. *Nonlinear Dynamics*, 112(5):3399–3420, January 2024.
- [259] Konstantinos Vlachas, David Najera-Flores, Carianne Martinez, Adam R. Brink, and Eleni Chatzi. A physics-based reduced order model with machine learning-boosted hyper-reduction. In Brandon J. Dilworth, Timothy Marinone, and Michael Mains, editors, *Topics in Modal Analysis & Parameter Identification, Volume 8*, pages 131–139, Cham, 2023. Springer International Publishing.
- [260] Barry Voight. A method for prediction of volcanic eruptions. *Nature*, 332(6160):125–130, 1988.
- [261] Barry Voight. A relation to describe rate-dependent material failure. *Science*, 243(4888):200–203, 1989.
- [262] D. J. Wagg, K. Worden, R. J. Barthorpe, and P. Gardner. Digital twins: State-of-the-art and future directions for modeling and simulation in engineering dynamics applications. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical Engineering*, 6, 9 2020.
- [263] Eric Walker, Sean Rayman, and Ralph E White. Comparison of a particle filter and other state estimation methods for prognostics of lithium-ion batteries. *Journal of Power Sources*, 287:1–12, 2015.
- [264] Mitchell Wall, Matthew S Allen, and Robert J Kuether. Observations of modal coupling due to bolted joints in an experimental benchmark structure. *Mech. Syst. Signal Process.*, 162(107968):107968, January 2022.
- [265] Mitchell Wall, Matthew S. Allen, and Iman Zare. Predicting s4 beam joint nonlinearity using quasi-static modal analysis. In Gaetan Kerschen, M. R. W. Brake, and Ludovic Renson, editors, *Nonlinear Structures and Systems, Volume 1*, pages 39–51, Cham, 2020. Springer International Publishing.
- [266] Kun Wang and WaiChing Sun. A multiscale multi-permeability poroplasticity model linked by recursive homogenizations and deep learning. *Computer Methods in Applied Mechanics and Engineering*, 334:337–380, 2018.
- [267] Tianyi Wang, Jianbo Yu, David Siegel, and Jay Lee. A similarity-based prognostics approach for remaining useful life estimation of engineered systems. In *2008 international conference on prognostics and health management*, pages 1–6. IEEE, 2008.
- [268] Larry Wasserman. *All of Nonparametric Statistics (Springer Texts in Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.
- [269] B.L. Welch. The generalization of Student’s problem when several different population variances are involved. *Biometrika*, pages 28–35, 1947.

- [270] Shun Weng, Hongping Zhu, Yong Xia, Jiajing Li, and Wei Tian. A review on dynamic substructuring methods for model updating and damage detection of large-scale structures. *Advances in Structural Engineering*, 23:584–600, 2 2020.
- [271] Jarmila Woodtli and Rolf Kieselbach. Damage due to hydrogen embrittlement and stress corrosion cracking. *Engineering failure analysis*, 7(6):427–450, 2000.
- [272] Louise Wright and Stuart Davidson. How to tell the difference between a model and a digital twin. *Advanced Modeling and Simulation in Engineering Sciences*, 7, 12 2020.
- [273] Long Wu and Paolo Tiso. Nonlinear model order reduction for flexible multibody dynamics: a modal derivatives approach. *Multibody System Dynamics*, 36:405–425, 4 2016.
- [274] Rih-Teng Wu and Mohammad Reza Jahanshahi. Deep convolutional neural network for structural dynamic response estimation and system identification. *Journal of Engineering Mechanics*, 2019.
- [275] Zhimin Xi, Modjtaba Dahmardeh, Bing Xia, Yuhong Fu, and Chris Mi. Learning of battery model bias for effective state of charge estimation of lithium-ion batteries. *IEEE Transactions on Vehicular Technology*, 68(9):8613–8628, 2019.
- [276] Xiaoyu Xie, Jennifer Bennett, Sourav Saha, Ye Lu, Jian Cao, Wing Kam Liu, and Zhengtao Gan. Mechanistic data-driven prediction of as-built mechanical properties in metal additive manufacturing. *npj Computational Materials*, 7(1):1–12, 2021.
- [277] Liu Yang, Xuhui Meng, and George Em Karniadakis. B-pinns: Bayesian physics-informed neural networks for forward and inverse pde problems with noisy data. *Journal of Computational Physics*, 425:109913, 2021.
- [278] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. *CoRR*, abs/2001.06782, 2020.
- [279] Jice Zeng, Michael D Todd, and Zhen Hu. Probabilistic damage detection using a new likelihood-free bayesian inference method. *J. Civ. Struct. Health Monit.*, 13(2-3):319–341, March 2023.
- [280] Qi Zeng, Yash Kothari, Spencer H. Bryngelson, and Florian Schäfer. Competitive physics informed networks, 2022.
- [281] Enrui Zhang, Ming Dao, George Em Karniadakis, and Subra Suresh. Analyses of internal structures and defects in materials using physics-informed neural networks. *Science Advances*, 8(7):eabk0644, 2022.
- [282] Guanfeng Zhang, George A Gazonas, and Florin Bobaru. Supershear damage propagation and sub-rayleigh crack growth from edge-on impact: A peridynamic analysis. *International Journal of Impact Engineering*, 113:73–87, 2018.

- [283] Jingliang Zhang and Jay Lee. A review on prognostics and health monitoring of li-ion battery. *Journal of power sources*, 196(15):6007–6014, 2011.
- [284] Lei Zhang, Chanwook Park, Ye Lu, Hengyang Li, Satyajit Mojumder, Sourav Saha, Jiachen Guo, Yangfan Li, Trevor Abbott, Gregory J. Wagner, Shaoqiang Tang, and Wing Kam Liu. Isogeometric convolution hierarchical deep-learning neural network: Isogeometric analysis with versatile adaptivity. *Computer Methods in Applied Mechanics and Engineering*, page 116356, 2023.
- [285] Weiping Zhang, Binbin Zhou, Xianglin Gu, and Hongchao Dai. Probability distribution model for cross-sectional area of corroded reinforcing steel bars. *Journal of Materials in Civil Engineering*, 26(5):822–832, 2014.
- [286] Yongzhi Zhang, Rui Xiong, Hongwen He, and Michael G. Pecht. Long short-term memory recurrent neural network for remaining useful life prediction of lithium-ion batteries. *IEEE Transactions on Vehicular Technology*, 67(7):5695–5705, 2018.
- [287] YongZhi Zhang, Rui Xiong, HongWen He, Xiaobo Qu, and Michael Pecht. Aging characteristics-based health diagnosis and remaining useful life prognostics for lithium-ion batteries. *ETransportation*, 1:100004, 2019.
- [288] Zhen Zhang, Yeonjong Shin, and George Em Karniadakis. Gfinns: GENERIC formalism informed neural networks for deterministic and stochastic dynamical systems. *Philosophical Transactions of the Royal Society A*, 380(2229):20210207, 2022.
- [289] Ying Zhao, Chen Jiang, Manuel A Vega, Michael D Todd, and Zhen Hu. Surrogate modeling of nonlinear dynamic systems: a comparative study. *Journal of Computing and Information Science in Engineering*, 23(1):011001, 2023.
- [290] Qiang Zheng, Lingzao Zeng, and George Em Karniadakis. Physics-informed semantic inpainting: Application to geostatistical modeling. *Journal of Computational Physics*, 419:109676, 2020.