Probabilistic Operational Envelopes and Risk Aware Routing for Autonomous Ships

by

Nathaniel Clemett

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Naval Architecture and Marine Engineering)
University of Michigan
2026

Doctoral Committee:

Professor Matthew Collette, Chair
Professor Kevin Maki
Professor David Singer
Assistant Professor Katie Skinner

Nathaniel Clemett
nclemett@umich.edu
ORCID iD: 0009-0005-0983-003X

© Nathaniel Clemett 2026

ACKNOWLEDGMENTS

There is a saying in sailing that if you're considering reefing the sails, it is likely already too late. Throughout my years working as a professional charter captain, delivering yachts, and racing, I have unfortunately verified this statement more times than I would like to admit. The ocean's ability to rapidly build in intensity often leads to hazardous situations where damage or loss of the vessel can occur. Several times I have been forced to ride out heavy weather at a certain heading or speed. These experiences helped lead me to pursue constructing a framework to develop operational envelopes for autonomous vessels.

Before beginning the dissertation I would like to thank the people who have helped me along the very winding route to where I am today. From the very beginning, my parents have always supported my journey and helped push me in academics. Largely in credit to my father, who continued to develop my sailing skills as a child while I whined about not playing 'normal' sports like other kids. Although I may have complained nonstop as a youth about sailing, it has molded into one of my greatest passions in life and opened countless doors for me.

One of these doors was opened by my friend Shea Unti, who offered a captain job in Croatia to a young 19 year old Nate living in the Caribbean. Another is the team at University of Washington's Applied Physics Laboratory (APL), specifically Jim Thomson and Alex de Klerk. APL gave me incredible experience in a wide range of ocean engineering challenges, from a swarm of micro wave buoys being deployed into the leading edge of hurricane Irma to development of a wave energy converter that charged an underwater remotely operated vehicle. These experiences in marine engineering developed a desire to pursue an advanced degree which led me to the University of Michigan.

My experience at the University of Michigan would not have been the same without the support and guidance of my fantastic advisor, Dr. Matthew Collette. From Flywheel Energy Storage Devices to adaptive fuel predicting frameworks to lab meetings on scientific writing and presentations, I have learned so much from you this past 4.5 years. What was supposed to be a one year master program turned into a double masters, and finally a PhD because it has been such a pleasure to work with you. To Dr. Katie Skinner, thank you for allowing me to pursue directed study for a Robotics Masters within the Field Robotics Laboratory, acting as cognate on my committee, and taking the time to answer questions I have about the marine robotics field that aren't related to my PhD. Thank to you Dr. Jessica Dibelka, for supporting this work at the Office of Naval Research. Within the office of room 137, I would like to acknowledge both Brendan Sulkowski and Kyle Marlantes, who both spent significant time helping me debug code and get me up to speed on current events in the NA&ME department (I need to get better about opening department emails).

Lastly, I want to thank my wife Jess, who has told me it'll all workout more times than I can imagine. Never doubting my path, and always supporting me in whatever way she could. I would not be so close to the finish line without her help.

TABLE OF CONTENTS

ACKNOWLEDGMENTS ii

LIST OF FIGURES vii

LIST OF TABLES ix

LIST OF ACRONYMS x

ABSTRACT xii

CHAPTER

1 Introduction 1

1.1 Motivation 10
1.2 Research Scope 12
1.3 Organization of Thesis 13

2 Tools 16

2.1 Weather Data Collection 16
2.1.1 Hindcast Weather Data 16
2.1.2 Nowcast and Forecast Dataset 17
2.1.3 Weather Interpolation 18
2.1.4 Selection of Bounding Forecast Times 18
2.1.5 Spatial Bilinear Interpolation On the Grid 19
2.1.6 Temporal Interpolation of Spatially Interpolated Fields 21
2.1.7 Geodesic Distance 21
2.2 Closed-Form Solutions 22
2.2.1 Vertical Bending Moment 23
2.2.2 Heave Frequency Response Function 24
2.2.3 Pitch Frequency Response Function 25
2.2.4 Root Mean Square Values of Motion 25
2.3 Vessel Design and Hullform 27
2.4 ProteusDS Simulation Toolbox 28
2.5 Operational Envelope 30
2.6 Kullback–Leibler Divergence 31
2.7 Evidence Lower Bound Loss 32

2.8 PyTorch Lightning . . . 34
2.9 Genetic Algorithm for Hyperparameter Optimization . . . 35

3 Model Agnostic Meta-Learning for ASVs . . . 36

3.1 Algorithm and Setup . . . 36
3.2 Initial Rapid Adaptation Exploration . . . 38
3.2.1 Initial MAML Performance Exploration . . . 39
3.3 Time Domain Dataset for Increased Realism . . . 42
3.4 Model Set Up and Scoring Metrics . . . 43
3.5 Rapid Adaptation of Pre-Deployment Engineering Models to Real-World Conditions 44
3.6 Extreme Weather . . . 47
3.6.1 Prediction of Slamming Frequency . . . 48
3.7 Forgetfulness . . . 50
3.8 Operational Envelope Construction . . . 53
3.9 MAML Conclusions for ASVs . . . 58

4 Evaluating Uncertainty-based Machine Learning Models . . . 61

4.1 Bayesian Neural Network (BNN) . . . 62
4.1.1 Bayesian Neural Network Background . . . 65
4.1.2 BNN Model Setup and Training . . . 67
4.1.3 BNN Results . . . 72
4.2 Deep Evidential Regression (DER) . . . 76
4.2.1 Deep Evidential Regression Background . . . 77
4.2.2 DER Model Setup and Training . . . 79
4.2.3 DER Results . . . 83
4.3 Deep Kernel Learning (DKL) . . . 84
4.3.1 Deep Kernel Learning Background . . . 87
4.3.2 DKL Model Setup and Training . . . 89
4.3.3 DKL Results . . . 95
4.4 Comparison of Models . . . 96

5 Uncertainty Scaling . . . 102

5.1 Background and Related Work . . . 103
5.1.1 Joint Probability of Failure in Operational Envelope . . . 104
5.2 Out-of-Distribution Scaling . . . 105
5.2.1 Training Distribution and Mahalanobis Distance . . . 106
5.2.2 Smooth OOD Scaling Function . . . 107
5.2.3 Algorithm for OOD Uncertainty Scaling . . . 108
5.3 Post Inspection Damage Scaling . . . 109
5.3.1 Damage Condition Representation . . . 109
5.3.2 Feature Importance via Sensitivity Analysis . . . 110
5.3.3 Damage Proximity Measure . . . 111
5.3.4 Damage Based Scaling Factor . . . 113
5.3.5 Algorithm for Damage Based Uncertainty Scaling . . . 113
5.4 Combination of OOD and Damage Scaling . . . 114

5.5 Discussion of Design Choices . . . 116
6 Reinforcement Learning Path Routing . . . 121
6.1 Reinforcement Learning for Marine Routing . . . 121
6.1.1 Deep Deterministic Policy Gradient . . . 124
6.2 Environment . . . 128
6.2.1 Inclusion of Weather Forecast Data . . . 130
6.2.2 `reset` Function . . . 130
6.2.3 `step` Function . . . 131
6.2.4 `get_obs` Function . . . 135
6.3 Experimental Setup . . . 136
6.3.1 Geographical and Temporal Domain . . . 136
6.3.2 Grid Discretization and Observation Layout . . . 137
6.3.3 Normalization of State and Mission Geometry . . . 138
6.3.4 Observation Space Dimension . . . 140
6.3.5 Action Space and Mission Randomization . . . 141
6.3.6 Exploration Noise . . . 142
6.3.7 Training Configuration . . . 143
6.3.8 Diagnostics and Visualization . . . 144
6.3.9 Testing Protocol . . . 145
6.4 Results and Discussion . . . 146
6.4.1 Single Waypoint Mission . . . 147
6.4.2 Multiple Waypoint Mission . . . 148
7 Conclusions and Future Work . . . 152
7.1 Future Work . . . 156
7.1.1 Chapter 3: Rapid Online Adaptation with Retention . . . 156
7.1.2 Chapter 4: Probabilistic Limit State Prediction and Rare Event Responses 158
7.1.3 Chapter 5: Uncertainty Scaling Informed by Novelty and Structural Condition . . . 158
7.1.4 Chapter 6: Risk Aware Routing Under Forecast Seas . . . 159

APPENDIX . . . 160

BIBLIOGRAPHY . . . 162

LIST OF FIGURES

FIGURE

2.1 Hindcast weather data collection zone 17
2.2 The response spectra is calculated by taking at least one wave spectra and integrating them through the RAOs [Journee and Massie, 2001] 26
2.3 The U.S. Navy Vanguard launched in 2024, and was the motivation for our hull design 27
2.4 Vanguard ASV transiting in waves within the Proteus Simulation Toolbox 29
2.5 Visualization of multiple tasks within the operational envelope [Fjørtoft and Rødseth, 2020] 30
2.6 Example Operational Envelope that combines limit states for heave and pitch motion RMS, and slamming frequency for the vessel traveling at 13 knots 31

3.1 Prediction accuracy improving as number of shots increases. 41
3.2 Improvements in accuracy for initial exploration of MAML predicting VBM 42
3.3 Prediction accuracy improving as number of shots increases. 45
3.4 Performance increases rapidly before tapering off as number of shots is increased . . . 46
3.5 Prediction accuracy improving as number of shots increases 48
3.6 Performance increases rapidly before tapering off as number of shots is increased . . . 49
3.7 MAML successfully adapts to slams per hour despite initial training in small wave training with rare slamming events 50
3.8 Effectiveness of MAML is dependent on diversity of tasks 52
3.9 Effectiveness of MAML after successive calm water missions 52
3.10 MAML heave prediction differences between ground truth and models for 50 shot condition (left) and 3000 shot condition (right) 55
3.11 MAML pitch prediction differences between ground truth and models for 50 shot condition (left) and 3000 shot condition (right) 56
3.12 MAML slams per hour prediction differences between ground truth and models for 50 shot condition (left) and 3000 shot condition (right) 58
3.13 Evolution of Operational Envelope 59

4.1 Fine tuning the base model from the routine weather mission results in low variation of predictions from the models within the ensemble 63
4.2 Increasing the λ_{KL} or σ_{prior} further, results in drastic degradation of performance. . . 74
4.3 BNNs had low uncertainty that quickly collapsed after experiencing extreme waves. Forgetfullness is a serious concern. 75

4.4 Deep Evidential Regression captures both aleatoric and epistemic sources of uncertainty. For the most part, the combined uncertainty bounds encompass the ground-truth values. 85
4.5 Deep Kernel Learning 97
4.6 Of the uncertainty based ML models, the BNN is the only model to successfully adapt to slamming in an online training representation. The first column represents the models trained with 3000 shots of small wave data, while the second column is those trained models after being exposed to 3000 shots of large wave data. 100

5.1 OOD scaling inflates predictive uncertainty in rarely observed and potentially hazardous sea states, resulting in a more conservative operational envelope. 118
5.2 Post inspection damage scaling increases uncertainty in a localized neighborhood of the recorded damage condition, tightening the operational envelope near previously hazardous states. 119
5.3 Combined OOD and damage scaling yields a conservative operational envelope that reduces the risk of operating the ASV in extreme or previously damaging sea states. . 120

6.1 DDPG Flowchart [Guo et al., 2020] 125
6.2 The red box shows the geographic area for the RL demonstration 129
6.3 Subtle differences between the Operational Envelope's 750 and 3000 shot models are used to represent a vessel (3000), and its imperfect machine learning models (750) . . 132
6.4 The left column displays the end of mission plot for a single waypoint mission with randomized starting locations. The right column shows intermediary screenshots of a single mission video, which includes the failure probability nodes changing as weather develops and speeds are altered. 150

LIST OF TABLES

TABLE

2.1 Vanguard Mimic Vessel Characteristics 28

3.1 Main Particulars of initial vessel used for MAML exploration 39

4.1 DKL Hyperparameters 92
4.2 Accuracy comparison of pitch for uncertainty based machine learning models 98

5.1 Recorded damage conditions 110
5.2 Feature importance weights from perturbation analysis 111

6.1 Truncation penalties and rewards for reaching waypoints 133
6.2 Weights for reward components 133
6.3 Coordinates of RL mission waypoints 147

LIST OF ACRONYMS

ASV Autonomous Surface Vessel

BNN Bayesian Neural Network

CFD Computational Fluid Dynamics

DDPG Deep Deterministic Policy Gradient

DER Deep Evidential Regression

DKL Deep Kernel Learning

ECMWF European Centre for Medium-Range Weather Forecasts

ELBO Evidence Lower Bound

GA Genetic Algorithm

GP Gaussian Process

KL Kullback-Leibler

MAML Model-Agnostic Meta-Learning

ML Machine Learning

MSE Mean Squared Error

NIG Normal Inverse Gamma

NN Neural Network

NOAA National Oceanic Atmospheric Administration

OOD Out-of-Distribution

OU Ornstein-Uhlenbeck

RAO Response Amplitude Operator

RBF Radial Basis Function

RL Reinforcement Learning

RMS Root Mean Square

SGD Stochastic Gradient Descent

VBM Vertical Bending Moment

VI Variational Inference

ABSTRACT

As autonomous ships become more prevalent in the coming years, it is necessary to find methods to replace the human-gathered knowledge from decades of experience onboard crewed ships to vehicles without human operators and crew. Machine learning methods are extremely powerful tools, but they are often black-box methods and rigid in nature. These autonomous vessels must maintain safety margins in an environment characterized by nonlinear wave excitation and limited onboard oversight. While human crews adapt in real-time using decades of combined experience, operational envelopes based on machine learning models derived offline can become unreliable when a vessel encounters sea states that differ from predeployment assumptions or when structural capability changes during service. This dissertation develops a data assimilative, uncertainty aware modeling framework that updates motion based operational limits from sparse onboard observations and embeds those limits in mission level routing.

A notional monohull vessel based on the U.S. Navy Vanguard unmanned surface vehicle is studied using hindcast, nowcast, and forecast wave samples. Pitch, heave, and vertical bending moment responses are predicted from environmental and operational inputs using a combination of closed-form seakeeping expressions and time domain simulations in ProteusDS with nonlinear Froude-Krylov excitation. To enable onboard revision of predeployment surrogates using limited data, model agnostic meta learning is used to learn an initialization that can be adapted rapidly using small batches of time domain observations collected during operation. The objective is to reduce mismatch between predeployment predictions and observed responses while maintaining computational tractability for mission level decision making. Model agnostic meta learning enables rapid adaptation of predeployment surrogates to time domain observations using limited samples but reveals sensitivity to catastrophic forgetting when training proceeds sequentially through periods

of similar conditions. Introducing representative extreme sea state samples into each adaptation tasks mitigates this forgetting and improves retention of heavy weather knowledge across successive calm water missions.

Accurate prediction of motion limit states alone is insufficient for autonomous operation near safety boundaries, where training data is sparse and extrapolation errors can lead to catastrophic decisions. To address this, three probabilistic regression models are evaluated for uncertainty aware prediction of motion limit states: Bayesian neural networks, deep evidential regression, and deep kernel learning. These models are assessed not only on predictive accuracy, but on their ability to produce calibrated uncertainty in poorly sampled and extreme sea states that dominate operational risk. Deep kernel learning is identified as providing a favorable balance between quality, reliability, and computational efficiency. It is therefore extended with two inference time uncertainty scaling mechanisms. A Mahalanobis distance based criterion inflates uncertainty for out of distribution sea states, while locality based damage scaling increases uncertainty near conditions associated with observed damage, producing conservative probabilistic operational envelopes in regions where erroneous confidence would generate unacceptable risk.

While adaptive motion prediction and calibrated uncertainty define safe operating limits, an autonomous vessel must ultimately translate these limits into sequential control decisions over extended horizons. This motivates coupling the probabilistic motion models to a reinforcement learning routing framework that reasons over both present and forecasted environmental conditions. A Deep Deterministic Policy Gradient agent is trained to select vessel speed and heading while navigating, managing the predicted probability of exceeding pitch and heave limits while completing a mission using real nowcast and forecast wave data. By embedding probabilistic operational limits directly into the reward structure, the routing policy learns to trade transit efficiency against forecast dependent motion risk. The resulting framework demonstrates how adaptive seakeeping models, uncertainty aware limit states, and reinforcement learning can be integrated to support sustained, risk informed autonomous decision making during long duration missions.

CHAPTER 1

Introduction

Crews on traditional vessels integrate observations, instrumentation, and operational experience to maintain safety margins. Autonomous surface vehicles, abbreviated as ASVs, must provide comparable oversight without onboard personnel. Effective autonomy therefore requires predictive models that can be updated rapidly as operating conditions change and mechanisms that can integrate knowledge elicited from expert mariners.

This chapter motivates the need for onboard self-assessment and adaptive operational limit management in long duration autonomous surface missions. The literature review is organized into five topics that connect to the dissertation problem formulation: challenges unique to crewless operations, operational envelope representations and their use in mission planning, motion prediction methods that integrate physics based and learning based models, learning based adaptation methods for rapidly updating surrogates during service, and uncertainty quantification methods that support risk aware decisions under limited data. The chapter concludes by defining the research scope and summarizing the thesis organization.

Challenges in Long Duration Autonomous Operations. Long duration missions in the maritime domain commonly span weeks to months and pose challenges that differ from shorter autonomous missions. In the absence of a crew, an ASV must undertake functions normally performed by humans, including routine machinery monitoring, health monitoring, mission level planning, and continuous risk assessment. Crewed ships rely on operators who integrate heterogeneous information and apply shared experience to maintain the vessel within safe limits. The removal of

onboard personnel therefore motivates decision frameworks that can reproduce these functions autonomously. Collette et al. [2022] cataloged these challenges and noted that many design and approval tools presume a human operator is available for immediate risk assessment and ad hoc decision making. When these human centered assumptions are retained for unmanned systems, the resulting platforms may struggle to remain competitive and may not realize the benefits of autonomy. The *Needs Exploration for Long Term Autonomous Marine Systems* report identified risk categories that become central in crewless operations, including mission failure, sustainment failure, and flexibility failure. These categories highlight systemic vulnerabilities that remain insufficiently addressed by current technology.

Recent attempts at prolonged autonomous voyages highlight persistent reliability challenges. The United States Navy's Sea Hunter, a 132 foot unmanned vessel, experienced engine and generator failures during a transit from San Diego to Pearl Harbor, requiring human crews to board the vessel on three occasions for repairs [noa, c]. The IBM Mayflower, a 15 m autonomous ship intended to cross the Atlantic, suffered a generator coolant fracture on its first attempt, leading to a breakdown and a fuel spill, and a second attempt encountered additional malfunctions that led to mission diversion and early termination [Katyanna Quach, 2022]. These cases emphasize the importance of fault detection, diagnosis, and contingency management during operation. While such capabilities are critical for long duration autonomy, they are not addressed in the present dissertation, which instead focuses on managing environmental and structural risk through adaptive operational limits.

Operational limits can also be tightened after unexpected damage, illustrating a mechanism that is directly aligned with the operational envelope concepts developed in this work. Multiple United States Navy Littoral Combat Ships developed hull cracks, and an interim mitigation restricted speeds below 20 knots when wave heights exceeded approximately 2.5 m [Ziezulewicz, 2022]. This form of envelope contraction reduced exposure to further damage but imposed substantial capability losses. Together, these examples motivate the need for onboard self assessment and adaptive operational envelope management that can revise safe operating limits as evidence accumulates, complementing but not replacing onboard fault management systems.

To address these concerns, research has progressed toward onboard self assessment and adaptive mission management. Georgiadis et al. [2025] proposed a Bayesian Network framework for near real time damage assessment following a grounding. The method fuses sensor readings, inspection evidence, and probabilistic structural models to estimate hull damage and residual strength. As new evidence becomes available, the Bayesian network updates the assessment and can recommend corresponding revisions to safe operating limits. The ability to revise limits in response to damage or system degradation is essential for long duration autonomy because it provides a form of self awareness that can support conservative actions such as speed reduction or shelter seeking when structural integrity is uncertain. Despite advances in problem characterization, fully autonomous solutions for maintenance, damage control, and risk informed decision making remain immature. Extended missions continue to raise open questions on how an unmanned vessel can anticipate hazards and respond with the reliability of an experienced crew.

Operational Envelope Concepts and Mission Planning. In the absence of onboard operators, safety depends on defining the environmental and operational conditions under which an ASV can perform tasks without exceeding acceptable risk. The operational envelope provides a formal representation of these limits. An operational envelope delineates bounds for variables such as wave height, wind, speed, and heading beyond which operability or survivability is compromised. When these bounds are defined for mission tasks and critical subsystems, operators and autonomy algorithms can distinguish conditions that are permissible from conditions that require replanning or mission abort. Multiple envelopes may coexist across subsystems and failure modes, with the overall safe operating region determined by the most restrictive constraint at a given time.

Rødseth et al. [2022] extended the envelope concept beyond static geographic limits by integrating environmental conditions, technical capability, and supervisory control into a unified framework. A modular structure was used to define an approved operating domain that depends not only on location and weather, but also on system health and the availability of remote oversight. The framework was motivated by the goal of reusable approvals, in which validated envelopes can

support regulatory and safety certification across similar missions. This perspective emphasizes that autonomy introduces additional safety dimensions, including algorithm reliability and communication constraints, that should be represented explicitly within allowable operating conditions.

Dynamic adaptation of operating envelopes during a mission has also been investigated. Page et al. [2021] developed a simulation driven approach to determine and adjust safe envelopes for unmanned aerial systems in offshore operations. The framework drew inspiration from Ship Helicopter Operating Limits and used performance based cost functions to evaluate component failure thresholds under wind and turbulence. High fidelity simulations were used to identify conditions that approach these thresholds. The study showed that as subsystems degrade through wear or minor damage, the safe envelope can contract to reflect reduced capability. This enables risk informed mission planning because the mission can be adjusted immediately to remain within the revised envelope. Although the application was aerial, the underlying principle generalizes to ASVs, which can benefit from real time envelope adjustment following propulsion, sensor, or structural degradation.

Operational envelope adaptation has also been proposed for ship damage scenarios. Fjørtoft and Rødseth [2020] demonstrated that an explicit envelope representation can support safer behavior under emergency conditions by tightening operating limits in response to updated estimates of hull strength. Such integration between health assessment and mission parameters illustrates a broader requirement for autonomous vessels. Predictive models must be coupled to decision rules that translate predicted responses and associated uncertainty into constraints on future actions.

Collectively, these studies establish the operational envelope as a useful concept for autonomous mission planning and assurance. Progress includes formal definitions that extend beyond simple environmental limits and demonstrations of envelope contraction in response to degradation. However, limitations remain. Many envelope definitions rely on offline analyses and may not accommodate unanticipated conditions or novel failure modes. Simulation based approaches often require advance specification of failure criteria and extensive modeling effort. Furthermore, several proposed frameworks remain primarily conceptual and have limited validation on autonomous

ships. Methods are needed that can refine an ASV operational envelope using data collected during missions so that safety boundaries remain accurate as conditions evolve. This need motivates the integration of operational envelope concepts with learning based models that update predictions and safety limits continuously.

Motion Prediction Models and Integration of Physics Based Priors. Accurate prediction of vessel motions and loads under forecasted sea conditions is central to operational limit definition. Naval architecture traditionally relies on physics based seakeeping models, including linear transfer functions and Response Amplitude Operators, to estimate motion responses. These models are computationally efficient and grounded in first principles, but they can be challenged by nonlinear effects, multidirectional seas, and vessel specific idiosyncrasies. Data driven approaches have therefore been explored, particularly when combined with physics based information.

Schirmann et al. [2022] developed machine learning models to forecast ship motion responses using wave model outputs as inputs. Sea state descriptors, including significant wave height, peak period, and wave direction, were used to train regression models for motions. Linear ridge regression and nonlinear neural networks were evaluated using a dataset of more than 13,500 thirty-minute windows of measured vessel motions. A central contribution was the use of physics based model predictions as additional features. In this hybrid formulation, a fast seakeeping code provided approximate motion estimates for each sample, and these estimates were supplied to the learning model alongside environmental inputs. Including physics based predictions reduced mean squared error in motion amplitude forecasts. The trained models were also evaluated on a sister vessel that was not included in training, which supported the claim that physics informed features can improve robustness across related platforms. A limitation is the data volume required to achieve strong performance, since training relied on roughly 10^4 labeled examples. New ASVs may not have access to comparable historical datasets, particularly when entering novel routes or mission profiles. This motivates models that can either reduce data requirements through informative priors or adapt online as new data becomes available.

Other work has emphasized physics based priors more directly. Cetin et al. [2022] presented a near term ship motion prediction method that uses Jensen's closed form seakeeping expressions to compute Response Amplitude Operators, then combines them with wave forecast data to predict motions over hours to days. A case study on a pipe laying vessel considered time varying loading as pipe was deployed, with Response Amplitude Operators updated as loading changed. The resulting time varying motion forecasts were intended to support operational decisions such as pausing operations when predicted motions exceed limits. For long term extremes, Petranović et al. [2021] combined hindcast wave databases with analytical response models to estimate extreme wave induced loads. Closed form formulas were used to compute vertical bending moments across many historical sea states along a route. Sea states that constrained operability were identified, and annual maxima were extrapolated using a Gumbel distribution to estimate a 20 year return period bending moment. This provides a route based characterization of structural limits that can inform load based operational envelopes.

These physics informed and data driven approaches provide complementary capabilities. Data driven models can capture complex relationships present in measured data and can adapt to vessel specific behavior, but they often require large datasets and may generalize poorly outside the training domain. Physics based models encode well understood constraints and can support extrapolation into rare conditions, but they may omit nonlinear behavior and can become inaccurate when vessel characteristics change due to damage, fouling, or loading deviations. Recent work suggests that combining these perspectives can improve performance, for example by using analytical model outputs as informative features in a learning model. Nevertheless, the data requirements observed by Schirmann et al. [2022] indicate that purely data driven methods can be impractical for new ASVs. Conversely, closed form methods require accurate hydrodynamic characterization and do not update automatically when behavior shifts or prediction errors emerge and accumulate during service. These gaps motivate flexible models that can learn from modest data, incorporate physical knowledge, and update as evidence of model error accumulates. The approach developed in this dissertation uses Deep Kernel Learning within a Bayesian framework and incorporates mechanisms

to incorporate expert knowledge of hazardous conditions into the predictive uncertainty.

Learning Based Methods for Risk Aware Adaptation. Risk aware autonomy requires more than point estimates of vessel response. Models must represent confidence, support adaptation to new operating regimes, and enable conservative decision making in the presence of limited data.

Deep Kernel Learning, introduced by Wilson et al. [2015], combines deep neural network representations with Gaussian process uncertainty quantification. A neural network learns a non-linear feature embedding of the inputs, and a Gaussian process is defined over this learned feature space. The network therefore acts as a trainable kernel, enabling the Gaussian process to accommodate complex and high dimensional inputs that are difficult to address with hand designed kernels. Training is performed by maximizing the Gaussian process marginal likelihood, which jointly learns neural network parameters and Gaussian process hyperparameters. The resulting model can improve predictive accuracy relative to standard neural networks or Gaussian processes, while providing uncertainty estimates through the Gaussian process posterior. The computational burden of Gaussian processes on large datasets was addressed using sparse variational approximations and kernel structure exploitation, enabling training on very large datasets. For autonomous vessels, this combination is attractive because it can represent nonlinear relationships between environmental conditions and ship response while producing uncertainty estimates that are needed for risk informed control.

Meta learning with Gaussian processes extends these ideas by enabling rapid adaptation across related tasks or regimes. Tighineanu et al. [2023] proposed ScaML GP, a scalable method that learns across a distribution of tasks and adapts to a new task using only a small number of observations. The approach uses a Gaussian process with a multi task kernel that aggregates information across training tasks to define a structured prior. At evaluation, a small set of measurements from new task conditions the model to produce a task specific posterior with calibrated uncertainty. Reported experiments showed faster adaptation and improved uncertainty reliability compared with several meta learning baselines, with linear scalability in the number of tasks. In an autonomy

context, distinct operating regimes such as calm seas, storm seas, and varying loading states can be interpreted as tasks.

Despite these advances, additional considerations are required for long duration ASV operation. Deep Kernel Learning can still exhibit overconfidence under extrapolation if training data are sparse in extreme conditions. Meta learning methods have been validated primarily on benchmark datasets, and their integration into real engineering systems requires careful verification. Onboard autonomy also imposes constraints on simulation, optimization, and memory resources. These factors motivate approaches that preserve probabilistic caution while remaining computationally feasible for real time decision support. The present work builds on Deep Kernel Learning because it provides an uncertainty aware foundation, and it augments this foundation with efficient uncertainty scaling mechanisms informed by novelty detection and human knowledge.

Uncertainty Quantification and Out of Distribution Detection. For autonomous operation in the open ocean, the credibility of a prediction can be as important as the prediction itself. Uncertainty quantification supports risk management by identifying when conditions approach the boundary of model validity or when the operational state diverges from prior experience. Conventional maritime practice often relies on fixed safety factors, but autonomy can benefit from uncertainty estimates that vary with context and data support.

Gaussian process models are well suited for uncertainty quantification because they provide predictive distributions with a mean and variance for each query. In vessel motion prediction, this enables an autonomous system to evaluate not only expected motions but also the range of plausible motions given available data. When uncertainty is high, the controller can adopt conservative actions such as speed reduction or course alteration to reduce exposure to worst case outcomes. Standard neural networks do not provide such uncertainty by default and can assign high confidence to erroneous predictions under novel conditions. This behavior is problematic in maritime settings where extreme seas and rare combinations of wave parameters can be critical to safety.

Out of distribution detection and uncertainty calibration methods have been developed to reduce this risk. Lee et al. [2018b] proposed a Mahalanobis distance approach that models the distribution of training features in a neural network representation space. New inputs are scored by the distance of their features from the training distribution, providing an anomaly measure that can flag out of distribution conditions. In an ASV context, such a score can identify wave conditions or vessel states that are unlike those represented in training data, indicating that motion predictions should be treated cautiously. Other approaches include ensembles and Monte Carlo dropout, which approximate predictive distributions by aggregating multiple stochastic predictions. These methods can increase computational cost and may require careful calibration. In this dissertation, baseline uncertainty is obtained from the Gaussian process component of the Deep Kernel Learning model, and an efficient distance based criterion in feature space is used to inflate uncertainty when inputs appear out of distribution.

Human operational knowledge can further inform uncertainty in ways that are not captured by standard training data. Mariners accumulate experience of hazardous scenarios, for example combinations of wave height and period that previously led to damage or near capsize. When similar conditions recur, experienced operators behave more conservatively even if the precise outcome is uncertain. Encoding this type of knowledge can be viewed as a structured increase in uncertainty in regions of the operating space associated with adverse outcomes. The framework developed in this dissertation incorporates such information by increasing predicted uncertainty near conditions linked to observed incidents. This mirrors engineering practice in which safety margins are revised after unexpected failures, but it is implemented as an adaptive change to the predictive distribution that directly affects operational envelope estimation.

Gaussian processes and Deep Kernel Learning provide baseline probabilistic uncertainty. Feature space distance methods provide a mechanism for detecting novelty, with expert knowledge and incident history providing additional cues for conservatism. What remains limited is an integrated approach that combines these elements in a form suitable for onboard decision support. Prior studies often address motion prediction, novelty detection, or envelope planning in isolation.

The contribution of this dissertation is to integrate these components into a unified learning based framework for operational limit estimation. The resulting model produces accurate motion predictions with uncertainty, and it increases caution under novel inputs and near known hazardous conditions to prevent overconfidence. The next section presents the formulation of the Deep Kernel Learning model with adaptive uncertainty scaling and describes its role in estimating an operational envelope that can be refined during repeated missions.

Conclusion of Literature Review. The surveyed literature indicates a convergence between learning based prediction and risk management for autonomous marine systems. Prior work identified key challenges in long duration autonomy and emphasized the need to define safe operating limits. The operational envelope emerged as a central representation for these limits and has been used to support mission planning and envelope contraction under degradation. In parallel, motion prediction research demonstrated the value of incorporating physics knowledge into learning models while highlighting data scarcity challenges for new vessels. Recent machine learning developments, including Deep Kernel Learning, provide adaptable models with probabilistic uncertainty that aligns with risk aware autonomy requirements. The approach developed in this dissertation synthesizes these developments by combining a neural network feature extractor for expressive motion modeling, a Gaussian process layer for principled uncertainty quantification, and adaptive uncertainty modulation informed by novelty detection and expert knowledge. This synthesis enables an operational envelope that is updated as evidence accumulates, supporting conservative decisions when the vessel enters poorly supported regions of the operating space and tightening limits after adverse events. The overall aim is to advance toward autonomy that can maintain survivability margins over extended missions without relying on continuous human intervention.

1.1 Motivation

The transition from crewed ships to large autonomous surface vessels removes the onboard human capability that has traditionally handled machinery monitoring, maintenance, mission replanning,

and real time risk assessment. Recent demonstrations of long duration autonomous voyages have exposed how mechanical failures and poorly understood limits can force human intervention or premature termination of missions, even when the underlying vehicles are conceptually capable platforms. In parallel, documented cases where crewed ships operate under conservative operating restrictions after structural damage show that safe performance depends critically on knowing and enforcing appropriate operational limits as conditions change. These developments indicate that autonomy at sea is not only a control and sensing problem but also a problem of adaptive operational risk management for long duration missions.

Operational envelope concepts provide a natural language to describe safe combinations of heading, speed, environmental conditions, and system health, yet most existing envelopes are static products of offline analysis or simulations. They are difficult to update when a vessel enters conditions that differ from predeployment assumptions or when the vessel experiences damage, degradation, or biofouling. Traditional seakeeping tools based on closed form expressions or linear response operators are computationally efficient and grounded in physics, but they can misrepresent nonlinear responses in heavy seas. Purely data driven models can capture complex behavior but often require very large datasets and provide limited information about their own reliability when extrapolating to unobserved conditions.

The present dissertation is motivated by the need for models that bridge these gaps. The goal is to develop and assess learning based motion prediction frameworks that incorporate physics based priors, operate effectively with modest datasets, and provide calibrated uncertainty estimates. These models are used to construct operational envelopes that evolve as new information is collected, and that can be queried by a higher level decision making layer. The dissertation aims to demonstrate an integrated framework in which an autonomous ship can plan and adjust its mission while explicitly managing the probability of exceeding motion limit states. It does this by coupling adaptive motion prediction, uncertainty quantification, and operational envelopes with a reinforcement learning based routing agent.

This section motivates the need for operational limits that remain valid under changing sea

states, imperfect forecasts, and evolving vessel condition. The remainder of the dissertation operationalizes this motivation by constructing data driven motion surrogates, quantifying uncertainty when models are used as decision boundaries, and embedding these limits into mission level decision making.

1.2 Research Scope

The overarching objective of this dissertation is to formulate and evaluate an adaptive modeling framework that estimates motion based operational limits for autonomous surface vessels and embeds those limits into mission level routing decisions. The work focuses on a notional monohull surface vessel, loosely based on the U.S. Navy Vanguard, operating in open ocean wave environments described by hindcast, nowcast, and forecast wave data. Vessel motions and loads are studied in terms of pitch, heave, and vertical bending moment, and safety is expressed through limits on root mean square motion levels.

Within this objective, the research addresses the following questions:

1. How can predeployment engineering models be rapidly adapted to time domain simulation data that include nonlinear excitation effects, using few shot meta learning, while limiting the loss of information about rare but critical heavy weather events?

2. How do different stochastic regression frameworks, including Bayesian neural networks, deep evidential regression, and deep kernel learning, compare in their ability to predict motion based limit states from environmental and operational inputs, in terms of accuracy, adaptation speed, uncertainty calibration, and susceptibility to forgetting?

3. How can a stochastic model be augmented with lightweight uncertainty scaling procedures for out of distribution and damage conditions that encodes expert knowledge about rare or hazardous conditions and that contract the operational envelope in regions of elevated risk?

4. How can motion predictions and their scaled uncertainties be coupled to a reinforcement

learning based path routing environment that uses nowcast and forecast wave fields so that the routing policy respects a probabilistic operational envelope while still making progress toward mission waypoints?

The scope of the study is restricted to numerical and simulation based experimentation. Wave environments are represented by hindcast and forecast products with interpolation in space and time. Vessel responses are obtained either from closed form solutions or from a single commercial time domain seakeeping tool. No full-scale experiments or hardware in-the-loop tests are performed. The routing agent is trained in a custom environment that uses Deep Deterministic Policy Gradient with continuous control of speed and heading and that terminates episodes when operational limits are exceeded, when the vessel leaves a predefined geographic region, or when an allowed mission time is surpassed. Other aspects of autonomy such as perception, collision avoidance relative to traffic, or machinery health management are not treated.

1.3 Organization of Thesis

Chapter 1: introduces the motivation, background, and problem statement. It reviews challenges in long duration autonomous surface vessel operations, summarizes existing work on operational envelopes, motion prediction models, learning based adaptation, and uncertainty quantification, and identifies the gaps that motivate the proposed framework.

Chapter 2: describes the tools and data sources used throughout the study. It introduces the hindcast, nowcast, and forecast wave datasets and the interpolation strategy, presents the geodesic distance calculation used for routing, and summarizes the closed form expressions employed for vertical bending moment, heave, and pitch response. The chapter also describes the ProteusDS time domain simulations used to generate training data, defines the operational envelope representation, and reviews the evidence lower bound loss formulation used in probabilistic learning.

Chapter 3: investigates Model Agnostic Meta Learning as a mechanism for rapid adaptation of predeployment motion models. It details the regression setup, training procedure, and scoring metrics, then examines how quickly a meta learned network adapts when exposed to limited samples of time domain data in both routine and extreme waves. The chapter highlights issues of forgetting when the model is trained online on sequences of similar conditions and demonstrates how these behaviors influence the construction and evolution of operational envelopes.

Chapter 4: evaluates three uncertainty aware machine learning models for motion limit prediction. Bayesian neural networks, deep evidential regression, and deep kernel learning are formulated for the problem and compared with respect to predictive accuracy, uncertainty calibration, adaptation behavior, and robustness when training data is limited or biased. The analysis identifies deep kernel learning as a suitable basis for subsequent development.

Chapter 5: introduces additional uncertainty scaling mechanisms on top of the deep kernel learning Gaussian process posterior. An out of distribution uncertainty scaler based on a distance metric in feature space and a post inspection damage scaler that emphasizes conditions similar to observed damage events are defined. Their combined effect on the pitch and heave operational envelopes is illustrated and interpreted in terms of more conservative, risk aware limit states.

Chapter 6: presents a reinforcement learning path routing framework that leverages the adaptive operational envelopes. A custom environment is constructed that integrates nowcast and forecast wave fields, and deep kernel learning motion models. A Deep Deterministic Policy Gradient agent is trained to control vessel speed and heading to visit one or more waypoints while avoiding violations of pitch and heave limits and while managing transit time. Results for single and multiple waypoint missions are discussed, including the influence of the risk term in the reward function.

Chapter 7: summarizes the main contributions of the dissertation and outlines directions for future research, including potential extensions to additional failure modes, richer structural response

models, and field validation on operational autonomous surface vessels.

The road map establishes a coherent progression from shared tools, to rapid adaptation, to uncertainty modeling and scaling, and finally to routing under risk constraints. Chapter 2 begins this progression by defining the data, models, and algorithms that serve as the common experimental foundation for the chapters that follow.

CHAPTER 2

Tools

This chapter serves as a reference for the following experiments in the dissertation, with the goal to reduce explaining implementations several times. This allows readers to read chapters out of order if desired and be able to jump back to the Tools chapter for explanations if they have not read previous chapters. This chapter summarizes the datasets, numerical methods, and software tools used to generate the environmental inputs, vessel response datasets, and training pipelines used throughout the dissertation.

2.1 Weather Data Collection

This section defines the environmental forcing pipeline used throughout the dissertation, ensuring that simulation, learning, and routing all query consistent wave inputs across time and space. This work uses a variety of weather data, including hindcast, nowcast, and forecast weather data. This work has used weather data from both the National Oceanic and Atmospheric Administration (NOAA) and European Centre for Medium-Range Weather Forecasts (ECMWF).

2.1.1 Hindcast Weather Data

The hindcast weather data used for every section except for reinforcement learning in Chapter 6 was collected from the ECMWF model in the delineated area marked by the red box in Figure 2.1. Within the zone, significant wave height, peak period, and primary wave direction were extracted for

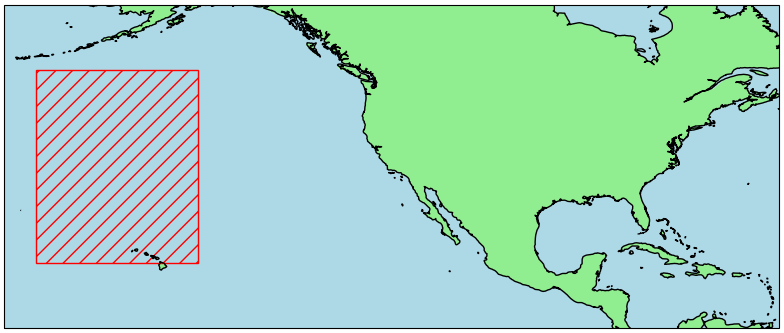


Figure 2.1: Hindcast weather data collection zone

the entire year of 2018. The hindcast fields were provided in NetCDF4 format on a regular latitude longitude grid spanning 20°N to 50°N and 175°W to 150°W, with 0.5° spacing (61 latitude points and 51 longitude points). The temporal coverage spans 1 January 2018 00:00 to 31 December 2018 21:00 with a constant 3hr sampling interval (2920 time steps). The archive also included auxiliary meteorological fields, including wind speed, wind direction, and surface pressure, although they were not included in our framework.

To assemble the collection of weather samples, random sampling occurs within the netCDF4 file that fits the criteria used for training, such as significant wave heights greater than 3.5 meters. It is important to note that because of this, training does not occur in a truly online manner. Future work plans to include temporal dependencies during training, which will pose additional challenges but be more representative of an ASV underway on-mission.

2.1.2 Nowcast and Forecast Dataset

As the University of Michigan's Marine Structures Design Laboratory continues to explore reliability of autonomy and develop machine learning models, a large dataset of weather was required for training models and evaluating multi-week missions where previous long-term autonomy missions such as the Sea Hunter have failed [noa, c].

To allow for both long duration and distance missions, the entire globe's weather data was collected from November 17th, 2024 to February 17th, 2025. This dataset of nowcast and forecast data across the globe ended up being close to 500 gigabytes of data. Because of this, a downsizer script was written to gather the relevant weather data for the proposed mission. It takes in latitude and longitude coordinates, what weather data types were needed (in our case, this was significant wave height, peak wave period, and primary wave direction), update frequency for nowcast files in intervals of three hours, and the forecast horizon for weather forecasts. This downsizer script allowed us to go from an extremely large 500 gigabyte dataset to a much smaller subset of data that could be loaded into memory for efficiency in reinforcement learning training.

The nowcast and forecast dataset supplies time evolving global wave fields needed to evaluate multi day missions and to support forecast driven routing in Chapter 6.

2.1.3 Weather Interpolation

Interpolation is introduced to reconcile gridded forecast products with the continuous state queries required by the simulation and reinforcement learning environment. An example of this is in the demonstration of Chapter 6, which had nowcast files update every six hours, but the simulation time step was every hour. The interpolation process follows the work outlined by Schirmann [2021].

Meteorological and oceanographic variables are available on a regular latitude longitude grid and at discrete forecast times stored in NetCDF files. The interpolation module provides, for any target position and forecast time, a consistent set of fields by combining spatial bilinear interpolation on the horizontal grid with linear interpolation in time.

2.1.4 Selection of Bounding Forecast Times

This step selects the two forecast snapshots that bound a query time, ensuring that subsequent interpolation remains consistent with the forecast update structure. For a given target forecast time

t the procedure first identifies the two forecast outputs that bound it. Let

$$t_1 \le t \le t_2$$

be the closest available forecast times from the same forecast cycle. The *planning* interpolation uses the latest forecast cycle issued at or before the current time, while the *nowcast* interpolation uses only files where issue time equals forecast time. These two files provide the fields $X(\varphi_i, \lambda_j, t_1)$ and $X(\varphi_i, \lambda_j, t_2)$ on the model grid, where φ_i are latitudes and λ_j are longitudes.

A nondimensional time weight $\alpha \in [0, 1]$ is then defined as

$$\alpha = \begin{cases} 0, & t_2 = t_1, \\ \dfrac{t - t_1}{t_2 - t_1}, & t_2 > t_1. \end{cases} \tag{2.1}$$

This weight controls the contribution of each forecast time in the final temporal interpolation.

2.1.5 Spatial Bilinear Interpolation On the Grid

Spatial interpolation converts discrete grid values into location specific wave conditions along candidate routes, enabling smooth environmental sampling as the vessel moves. Let (φ^*, λ^*) be the target latitude and longitude. The one dimensional arrays of grid coordinates are denoted

$$\{\varphi_i\}_{i=0}^{N_\varphi - 1}, \qquad \{\lambda_j\}_{j=0}^{N_\lambda - 1},$$

both sorted in ascending order. The routine `get_surrounding_indices` locates indices i_1, i_2 and j_1, j_2 such that

$$\varphi_{i_1} \le \varphi^* \le \varphi_{i_2}, \qquad \lambda_{j_1} \le \lambda^* \le \lambda_{j_2}. \tag{2.2}$$

These four grid points

$$C = \{(i_1, j_1), (i_1, j_2), (i_2, j_1), (i_2, j_2)\}$$

form the corners of the interpolation cell.

Define the grid spacings

$$\Delta\varphi = \varphi_{i_2} - \varphi_{i_1}, \qquad \Delta\lambda = \lambda_{j_2} - \lambda_{j_1}. \tag{2.3}$$

For each latitude index $i \in \{i_1, i_2\}$ a one dimensional latitude weight is computed as

$$w_i^{\varphi} = \max\left(0, 1 - \frac{|\varphi_i - \varphi^*|}{\Delta\varphi}\right), \tag{2.4}$$

and for each longitude index $j \in \{j_1, j_2\}$ a longitude weight

$$w_j^{\lambda} = \max\left(0, 1 - \frac{|\lambda_j - \lambda^*|}{\Delta\lambda}\right). \tag{2.5}$$

The unnormalized bilinear weight for corner $(i, j) \in C$ is then

$$\tilde{w}_{ij} = w_i^{\varphi}\, w_j^{\lambda}, \tag{2.6}$$

and the normalized interpolation weights are

$$w_{ij} = \begin{cases} \dfrac{\tilde{w}_{ij}}{\sum\limits_{(p,q)\in C} \tilde{w}_{pq}}, & \text{if } \sum\limits_{(p,q)\in C} \tilde{w}_{pq} > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{2.7}$$

For a scalar field X at time t_k the spatially interpolated value at (φ^*, λ^*) is

$$x_k(\varphi^*, \lambda^*) = \frac{\sum\limits_{(i,j)\in C} w_{ij}\, X(\varphi_i, \lambda_j, t_k)\, \chi_{ij}}{\sum\limits_{(i,j)\in C} w_{ij}\, \chi_{ij}}, \tag{2.8}$$

where χ_{ij} is equal to one when the value at (i, j) is valid and zero when the value is masked or not

a number. If all four corners are invalid the routine returns a missing value.

2.1.6 Temporal Interpolation of Spatially Interpolated Fields

Temporal interpolation aligns discrete forecast outputs with the simulation and planning time steps used later for mission level routing.

Once the spatial interpolation has been performed at the two bounding times t_1 and t_2 the module combines them linearly in time using the weight α from (2.1). For each variable, the temporally and spatially interpolated value at $(\varphi^*, \lambda^*, t)$ is

$$\hat{x}(\varphi^*, \lambda^*, t) = \begin{cases} x_1(\varphi^*, \lambda^*), & \text{if only } x_1 \text{ is valid,} \\ x_2(\varphi^*, \lambda^*), & \text{if only } x_2 \text{ is valid,} \\ (1-\alpha)\, x_1(\varphi^*, \lambda^*) + \alpha\, x_2(\varphi^*, \lambda^*), & \text{if both are valid,} \end{cases} \tag{2.9}$$

with α given by (2.1). This two stage procedure provides a continuous time and space representation of the gridded weather at arbitrary locations along the vessel trajectory.

2.1.7 Geodesic Distance

Mission level decision making requires consistent geographic distance and bearing calculations to translate latitude and longitude geometry into transit time, progress metrics, and forecast query schedules.

Geodesic distance is the length of the shortest curve constrained to a reference surface that connects two points. Under the spherical approximation, geodesics are great circle arcs. For navigation, the reference surface is more accurately represented by an oblate ellipsoid of revolution. On a surface of revolution, all meridians are geodesics, and parallels whose radius is extremal are also geodesics, which makes the equator a geodesic on an oblate ellipsoid. These special cases are closed curves; in contrast, most other geodesics are not closed and remain bounded between two

limiting parallels.

The inverse geodesic problem determines the geodesic length and endpoint azimuths from two points specified by latitude and longitude. Distances in this work are computed using the algorithm developed by Karney [2013]. It maps the ellipsoidal problem to an auxiliary sphere and evaluates the required integrals using series expansions in the flattening accurate to order $O(f^6)$. The method improves on the classical Vinccnty formula by providing accuracy at the level of floating point roundoff and by ensuring convergence for all point pairs, including configurations where Vincenty's inverse iteration may fail.

Together, these datasets and interpolation steps provide a continuous representation of wave forcing at arbitrary time and location, which is required for both learning experiments and routing demonstrations. The next section uses these environmental inputs to compute fast response estimates that define motion and load quantities later used as training targets.

2.2 Closed-Form Solutions

Closed-form structural load metrics provide an interpretable baseline for operational limit construction and enable controlled comparisons between simplified predictions and higher fidelity simulation targets.

The closed form solutions applied in this study are based on the semi-analytical expressions developed by Jensen and co-authors for rapid estimation of wave induced motions and global hull loads in monohull ships [Jensen et al., 2004, Jensen and Mansour, 2002]. The motivation for these formulations is to retain the physical structure of linear frequency domain seakeeping analysis while avoiding the computational burden associated with full hydrodynamic solvers. The required input uses principal dimensions and a small set of nondimensional coefficients together with operating conditions to enable efficient evaluation during early stage design studies, operational screening, and sensitivity analyses.

2.2.1 Vertical Bending Moment

Estimation of the wave-induced vertical bending moments (VBM) in mono-hull ships, as used in this study, is based on the closed-form solutions developed by Jensen and Mansour [2002]. These solutions start with using strip-theory to analytically derive the vertical bending moment of a homogeneously loaded box-shaped vessel. Correction factors are then introduced to modify the initial box-shaped vessel model to better represent the characteristics and dynamics of an actual ship. The VBM as a function of frequency is found through the closed-form expression in Equation 2.10.

$$\frac{VBM(\omega)}{\rho g B_0 L^2} = [1 - \cos(\frac{k_e L}{2}) - \frac{k_e L}{4}\sin(\frac{k_e L}{2})] * \kappa * \frac{1 - kT}{(k_e L)^2} * F_V(Fn)F_C(C_b) \times \sqrt[3]{|cos\mu|} \quad (2.10)$$

Where κ is the approximation of the Smith correction factor defined by Equation 2.11 [Jensen, 2001]. It is dependent on the effective wave number (Eq 2.12), and the peak period. It accounts for dynamic pressure decay as distance increases with vertical distance away from the free surface.

$$\kappa = \exp(-k_e T_p) \quad (2.11)$$

$$k_e = |kcos\mu| \quad (2.12)$$

$F_V(Fn)$ represents the speed correction factor from a homogeneously loaded box-shaped vessel to a range of ships. It accounts for speed through the incorporation of the Froude number (valid for $Fn < 0.3$) (Eq 2.13).

$$F_V(Fn) = 1 + 3Fn^2; \qquad Fn = \frac{V}{gL} \quad (2.13)$$

Due to the smaller volume of the foreward part of the vessel, ships with smaller block coefficients have lower bending moments. The block coefficient C_b is corrected through Equation 2.14.

$$\begin{aligned} F_C(C_b) &= [(1 - \varphi)^2 + 0.6\alpha(2 - \varphi)] \\ \varphi &= 2.5(1 - C_b); \quad C_b = \max(0.6, C_b) \end{aligned} \quad (2.14)$$

The equations of motion (Eq 2.15,2.16) were determined by assuming a constant sectional mass term that equaled the displaced water in addition to neglecting the coupling between heave and pitch. They are dependent on the wave number k, the frequency of the wave ω, and both the breath and depth of the box B, T.

$$2\frac{kT}{\omega^2}\ddot{w} + \frac{A^2}{kB\alpha^3\omega}\dot{w} + w = aF\cos(\varpi t) \tag{2.15}$$

$$2\frac{kT}{\omega^2}\ddot{\theta} + \frac{A^2}{kB\alpha^3\omega}\dot{\theta} + \theta = aG\sin(\varpi t) \tag{2.16}$$

The majority of this dissertation focuses on pitch, as large values generally have an increased impact on both performance and safety compared to heave.

2.2.2 Heave Frequency Response Function

To account for the frequency change due to vessel heading and speed, the frequency of encounter is used. It is dependent on the ship velocity V, and relative heading between the ship and prominent wave direction β (180° representing head seas) (Eq 2.17).

$$\omega_e = \omega - kV\cos\beta \equiv \alpha\omega \tag{2.17}$$

The parameter α (Eq 2.18), is defined by the Froude Number ($FN = \frac{V}{gL}$) where L is the length of the homogeneously loaded box.

$$\alpha = 1 - Fn\sqrt{KL}\cos(\beta) \tag{2.18}$$

The sectional hydrodynamic damping term, A (Eq 2.19), approximates the dimensionless ratio of the incoming and diffracted wave amplitudes .

$$A = 2\sin(\frac{1}{2}kB\alpha^2)\exp(-kT\alpha^2) \tag{2.19}$$

The Heave forcing function F is dependent on the effective wave number (Eq 2.12) and described by,

$$F = \kappa f \frac{2}{k_e L} \sin\left(\frac{k_e L}{2}\right) \tag{2.20}$$

$$\eta = \left(\sqrt{(1 - 2kT\alpha^2)^2 + \left(\frac{A^2}{kB\alpha^2}\right)^2} \right)^{-1} \tag{2.21}$$

The final step to calculate the frequency response function for heave is to multiply the forcing function F with η (Eq 2.22).

$$\Phi_w = \eta F \tag{2.22}$$

2.2.3 Pitch Frequency Response Function

The frequency response function is similarly found by multiplying a forcing function and η. The forcing function G, used for pitch, is described in Eq 2.23.

$$G = \kappa f \frac{24}{(k_e L)^2 L} \left[\sin\left(\frac{k_e L}{2}\right) - \frac{k_e L}{2} \cos\left(\frac{k_e L}{2}\right) \right] \tag{2.23}$$

After the forcing function has been calculated, the frequency response function is determined by combining the forcing function G with η (Eq 2.24).

$$\Phi_\theta = \eta G \tag{2.24}$$

2.2.4 Root Mean Square Values of Motion

The response spectrum is determined by applying the transfer function to the wave energy density spectrum and the frequency response function generated from Section 2.2.2 or 2.2.3, as outlined in Equation 2.26 [Journee and Massie, 2001]. A Bretschneider wave energy density spectrum was

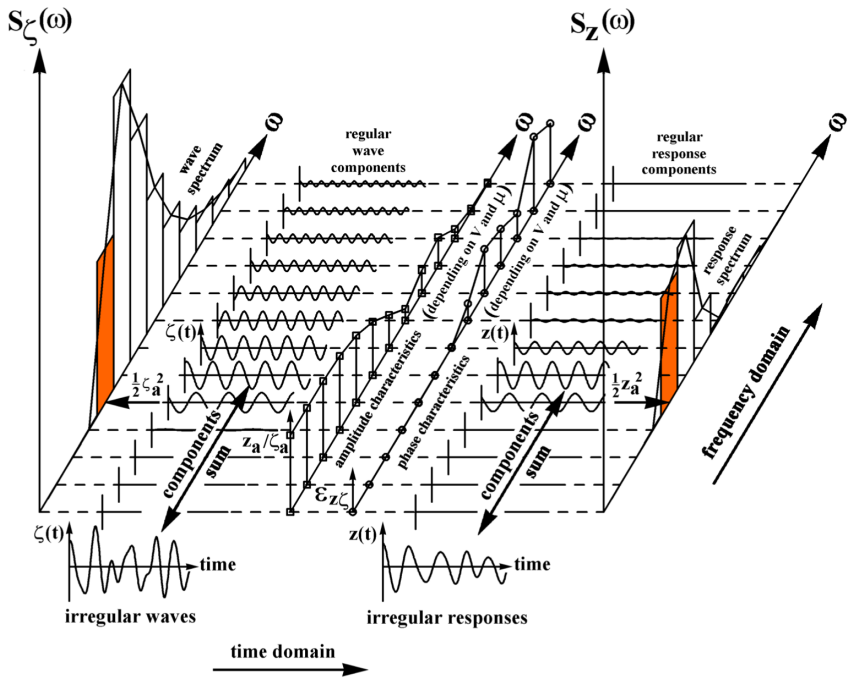


Figure 2.2: The response spectra is calculated by taking at least one wave spectra and integrating them through the RAOs [Journee and Massie, 2001]

used in this work (Eq 2.25). The Bretschneider wave energy density spectrum is parameterized by the prevailing sea state and operating condition at the vessel location. In this dissertation, significant wave height and peak period are taken from the gridded weather products described in Section 2.1 and are interpolated in space and time to the vessel position using the procedure in Section 2.1.3, after which the spectrum is constructed for each sampled condition.

$$S_\zeta(\omega) = \frac{173H_s^2}{T_p^4}\omega_e^{-5}\exp\{\frac{-692}{T_p^4}\omega_e^{-4}\} \tag{2.25}$$

$$S_\Phi(\omega) = |\Phi(\omega)|^2 \cdot S_\zeta(\omega) \tag{2.26}$$

The area under the response curve is the variance (Eq 2.27).

$$m_0 = \int_0^\infty S_\Phi(\omega)d\omega \tag{2.27}$$



Figure 2.3: The U.S. Navy Vanguard launched in 2024, and was the motivation for our hull design

Lastly, the Root-Mean Square of the respective response is determined through the square root of the response's variance (Eq 2.28).

$$\Phi_{RMS} = \sqrt{m_0} \tag{2.28}$$

The closed-form solutions define a baseline seakeeping response pipeline that maps wave spectra into motion and load statistics using closed-form assumptions. These results serve as a reference point and training target that can be contrasted with time domain nonlinear simulations from Section 2.4.

2.3 Vessel Design and Hullform

The virtual vessel used for this work is loosely based on the Navy Vanguard ASV. Vanguard is part of the Project Overlord Class of Navy unmanned surface vehicles and launched in 2024 [noa, 2024]. Project Overlord is the Navy's program designed to increase it's unmanned capabilities through modifying commercial offshore supply vessels for long-endurance operations and autonomy. The main particulars for this vessel are displayed in Table 2.1. A hullform was generated in Rhinocerous3D using the Orca3d Hull-Wizard plugin.

Table 2.1: Vanguard Mimic Vessel Characteristics

Main Particular	Notation	Value
Length BTW Perpendiculars	L_{PP}	59.0 (m)
Max Waterline Breadth	B_0	8.910 (m)
Beam	B	10.0 (m)
Draft	T	2.25 (m)
Transverse Metacentric Height	GM_T	1.237 (m)
Block Coefficient	Cb	0.61

2.4 ProteusDS Simulation Toolbox

ProteusDS is the time domain simulator used to generate response histories that reflect nonlinear excitation and transient dynamics beyond the accuracy of the closed form approximations.

To generate a machine learning time-domain dataset, DSA Ocean's ProteusDS Simulation Toolbox was used [noa, b]. To highlight discrepancies between pre-deployment models and real-world data, nonlinear Froude-Krylov was included in excitation forces. For incident waves, the Froude-Krylov force is the complex amplitude of the exiting force. This excitation force, or F_j^I, is dependent on the unit normal to the hull surface out of the fluid n_j, the incident wave potential ϕ_1, and the original relative velocity U_0. The relatively low-cost computation is done through integrating the known incident wave potential over the body surface (Eq 2.29).

$$F_j^I = -\rho \iint_S n_j (i\omega_e - U_0 \frac{d}{dx}) \phi_1 \, ds \tag{2.29}$$

After calculating radiation and diffraction terms and setting a wave spectrum to match a collected hindcast weather data sample from Section 2.1.1, 900 seconds of time-domain data was collected. After collecting this 15 minutes of simulated data, the RMS of the motions were calculated. This alternate collection method for RMS motion scales differently due to nonlinear Froude-Krylov excitation forces and was designed to represent the difference between pre-deployment engineering models and real-world data.

This 15 minutes of training data, decomposed into the RMS value of motion, is used for training. This sample is defined as a shot of data, and the terms are used interchangeably for the

duration of this dissertation. For a mission consisting of 50 shots of data, that corresponds to a 12.5 hour mission, while 3000 shots of data is a month at sea. Although this mission duration is much longer than other autonomous systems such as aerial drones, both shot levels are reasonable levels to evaluate computational efficiency for ships.

Although the time-domain simulation would generally behave, occasionally unrealistic behavior would occur that would result in extremely large motions and RMS values. To prevent this from worsening model performance, outlier data samples were removed if the pitch RMS value for the run was larger than 15 degrees.

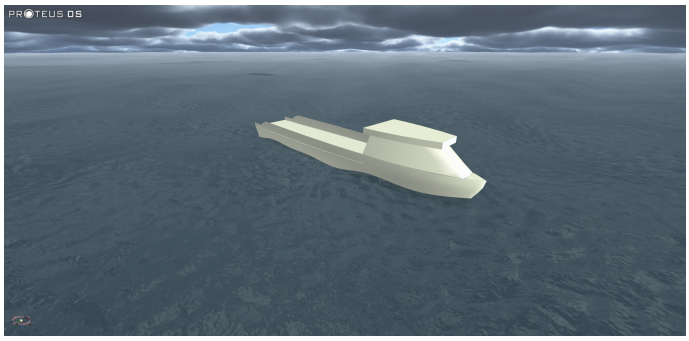


Figure 2.4: Vanguard ASV transiting in waves within the Proteus Simulation Toolbox

To calculate slamming frequency, a wave probe was placed on the hull of the model within ProteusDS. Within the 15 minute simulation, every time the probe exited and re-entered the water, it was added to a .csv file with its entry velocity as it re-entered the water. If the entry velocity exceeded 2.236 m/s, it qualified as a slamming event. This frequency was then extrapolated to an hour resulting in the commonly used metric of slams/hour.

The entry velocity was determined by the work of MK Ochi [1964]. Using a 160 meter *Mariner* class cargo ship, he determined that slamming occurred when the bow re-entered the water at speeds exceeding 3.65 m/s, regardless on the angle of entry. Based on these findings, he proposed that this entry velocity could be scaled depending on ship length for similar vessels, taking into accordance Froude Scaling ($\sqrt{L}$). This results in a slamming velocity of 2.236 m/s for our model Vanguard.

This section introduces the time domain simulation capability used to generate higher fidelity

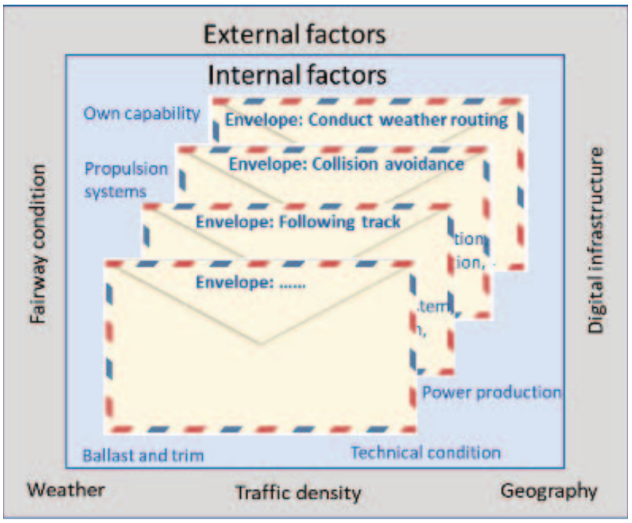


Figure 2.5: Visualization of multiple tasks within the operational envelope [Fjørtoft and Rødseth, 2020]

response data and to represent nonlinear excitation mechanisms that are not captured by closed form models. The resulting dataset underpins the later meta learning adaptation experiments and provides the response variability needed to evaluate uncertainty aware modeling choices.

2.5 Operational Envelope

The operational envelope exists as an interface between prediction and decision making, translating response limits into allowable operating conditions for a given vessel state and mission. The operational envelope is a visualization tool for human operators to visualize autonomous operational limits. It combines internal vessel factors, such as propulsion, structural loads, or loading conditions with external factors such as weather conditions and maritime traffic. By developing a collection of envelopes, an operator can ascertain both the operational capabilities and limits of an unmanned vessel.

The operational envelope is a representation of admissible operating conditions for a given vessel, defined by the intersection of environmental states and operational commands that satisfy prescribed safety criteria. In this dissertation, the envelope is expressed in terms of sea state and

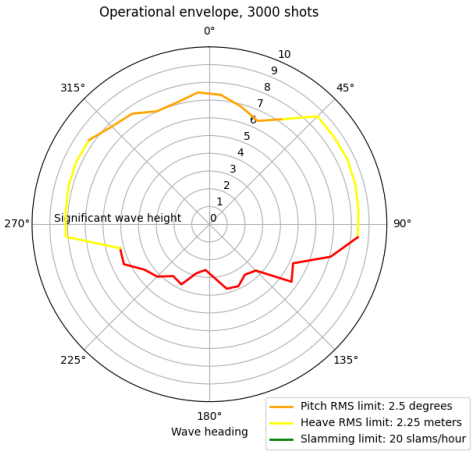


Figure 2.6: Example Operational Envelope that combines limit states for heave and pitch motion RMS, and slamming frequency for the vessel traveling at 13 knots

operating parameters such as significant wave height, wave period, relative wave heading, and vessel speed, and is bounded by limit states on root mean square motions, structural loads, and event rates such as slamming. The envelope provides an interface between response prediction models and autonomy algorithms, enabling mission planning and control to avoid combinations of heading, speed, and forecast conditions that exceed a specified probability of threshold violation. The operational envelope representation provides the decision boundary object that is updated through adaptation and uncertainty modeling in later chapters and ultimately embedded in routing decisions. Although this will be further explained in later chapters, Figure 2.6 displays an example of an envelope for the Vanguard traveling at 13 knots. A collection of these envelopes would be generated for different speeds and task limits. For example, launching or recovering a payload while underway would have stricter limits than survivability of the vessel.

2.6 Kullback–Leibler Divergence

Kullback-Leibler (KL) divergence is an information metric for comparing probability distributions, supporting later quantification of differences between training and deployment conditions. KL divergence provides a quantitative measure of the discrepancy between two probability distributions

that are defined on a common measurable space. Let $q(z)$ denote an approximating distribution and let $p(z)$ denote a reference distribution. For a discrete latent variable z taking values in a countable set $\mathcal{Z}$, the KL divergence of q from p is defined as

$$\mathrm{KL}(q(z) \,\|\, p(z)) = \sum_{z \in \mathcal{Z}} q(z) \, \log \frac{q(z)}{p(z)}. \tag{2.30}$$

For continuous latent variables with densities q and p, the definition becomes

$$\mathrm{KL}(q(z) \,\|\, p(z)) = \int q(z) \, \log \frac{q(z)}{p(z)} \, \mathrm{d}z. \tag{2.31}$$

The KL divergence can be expressed as the difference between the cross entropy and the Shannon entropy,

$$\mathrm{KL}(q(z) \,\|\, p(z)) = \mathbb{E}_{q(z)}[-\log p(z)] - \mathbb{E}_{q(z)}[-\log q(z)] \,. \tag{2.32}$$

In probabilistic machine learning it is convenient to interpret $\mathrm{KL}(q\|p)$ as a penalty that measures how far an approximating distribution q deviates from a preferred reference distribution p. By expressing distribution mismatch in a quantitative form, divergence provides a basis for identifying when learned predictors operate outside the regime supported by data. The next section builds on this probabilistic framing by introducing the evidence lower bound objective used to train uncertainty aware models.

2.7 Evidence Lower Bound Loss

The Evidence Lower Bound (ELBO) is the objective function that underpins variational methods for approximate Bayesian inference. Consider a latent variable model with joint density

$$p(x, z) = p(x \mid z) \, p(z), \tag{2.33}$$

where $p(z)$ is a prior distribution over latent variables and $p(x \mid z)$ is the likelihood of the observed data given the latent variables. The marginal likelihood or model evidence for an observation x is

$$\log p(x) = \log \int p(x \mid z)\, p(z)\, \mathrm{d}z. \tag{2.34}$$

For most models of interest this integral cannot be evaluated in closed form. Variational inference therefore introduces an auxiliary distribution $q(z \mid x)$ that approximates the true posterior $p(z \mid x)$. Using Bayes' rule and the definition of KL divergence yields

$$\begin{aligned}
\mathrm{KL}(q(z \mid x) \,\|\, p(z \mid x)) &= \mathbb{E}_{q(z|x)}\left[\log \frac{q(z \mid x)}{p(z \mid x)}\right] && (2.35)\\
&= \mathbb{E}_{q(z|x)}\left[\log \frac{q(z \mid x)}{\frac{p(x|z)\, p(z)}{p(x)}}\right] && (2.36)\\
&= \mathbb{E}_{q(z|x)}\left[\log \frac{q(z \mid x)\, p(x)}{p(x \mid z)\, p(z)}\right] && (2.37)\\
&= \mathbb{E}_{q(z|x)}\left[\log q(z \mid x) + \log p(x) - \log\big(p(x \mid z)p(z)\big)\right] && (2.38)\\
&= \log p(x) - \mathbb{E}_{q(z|x)}\left[\log\big(p(x \mid z)p(z)\big) - \log q(z \mid x)\right] && (2.39)\\
&= \log p(x) - \mathbb{E}_{q(z|x)}\left[\log p(x \mid z) + \log p(z) - \log q(z \mid x)\right]. && (2.40)
\end{aligned}$$

Rearranging terms gives the standard decomposition of the log evidence into an ELBO term and a KL divergence term,

$$\begin{aligned}
\log p(x) &= \mathbb{E}_{q(z|x)}\left[\log p(x \mid z) + \log p(z) - \log q(z \mid x)\right] + \mathrm{KL}(q(z \mid x) \,\|\, p(z \mid x)) && (2.41)\\
&\geq \mathbb{E}_{q(z|x)}\left[\log p(x \mid z) + \log p(z) - \log q(z \mid x)\right] && (2.42)\\
&=: \mathrm{ELBO}(x; q). && (2.43)
\end{aligned}$$

By non negativity of the KL divergence, the ELBO constitutes a lower bound on $\log p(x)$. Maximizing the ELBO over both model and variational parameters is therefore equivalent to maximizing

a tractable lower bound on the intractable marginal likelihood.

The ELBO can be written in a form that separates a data reconstruction term from a regularization term,

$$\mathrm{ELBO}(x; q) = \mathbb{E}_{q(z|x)}[\log p(x \mid z) + \log p(z) - \log q(z \mid x)] \tag{2.44}$$

$$= \mathbb{E}_{q(z|x)}[\log p(x \mid z)] - \mathrm{KL}(q(z \mid x) \,\|\, p(z)) \,. \tag{2.45}$$

For a dataset $\{x_i\}_{i=1}^N$ the training objective used in this work is the average negative ELBO,

$$\mathcal{L}_{\mathrm{ELBO}} = -\frac{1}{N} \sum_{i=1}^{N} \mathrm{ELBO}(x_i; q) = \frac{1}{N} \sum_{i=1}^{N} \left(-\mathbb{E}_{q(z|x_i)}[\log p(x_i \mid z)] + \mathrm{KL}(q(z \mid x_i) \,\|\, p(z)) \right) . \tag{2.46}$$

The first term corresponds to a reconstruction loss that encourages the generative model to assign high likelihood to the observed data under samples of the latent variables. The second term is a KL regularization term that penalizes deviations of the variational posterior from the prior and thereby controls the effective capacity of the latent representation. In many practical models the reconstruction term is estimated by Monte Carlo sampling.

Standard references for KL divergence and ELBO based variational inference include the original article by Kullback and Leibler [1951], the textbook treatments in Bishop [2006], the review of variational inference by Blei et al. [2017], and the application to deep generative models by Kingma and Welling [2022]. The evidence lower bound connects data fit and regularization within a unified probabilistic criterion, facilitating uncertainty representations required for conservative envelope construction. The next section records the training infrastructure used to apply these objectives consistently across multiple model families.

2.8 PyTorch Lightning

This work involves training a variety of different machine learning algorithms in different transfer learning situations with different hyperparameters, training datasets, and architectures. On top of

that, multiple machines were used for training, including the University of Michigan's Great Lakes high performance computing cluster [noa, a]. Due to this, PyTorch Lightning was used [Falcon and The PyTorch Lightning team, 2019]. Lightning handles a significant portion of the low-level ML training code, which simplifies the training process and reduces likelihood of errors. It also features a robust logging, early stopping, and checkpointing systems so you can evaluate different models easily and efficiently save and load different checkpoints for either testing or transfer learning.

Lightning was also beneficial to use for training across hardware. If allowed, it can scan your system and provide recommendations to expedite training. This can include automatically setting up and training across multiple graphical processing units (GPU) or determining the optimal number of helpers for dataloaders. This was particularly helpful when moving the training to the cluster for optimization of hyperparameters using a Genetic Algorithm as described in the following section.

2.9 Genetic Algorithm for Hyperparameter Optimization

To select the Model Agnostic Meta Learning (MAML) configuration used in Chapter 3, a genetic algorithm (GA) was employed for joint hyperparameter and architecture search. Genetic algorithms, developed by Holland [1992], are derivative free, population based optimizers that are well suited to mixed continuous and discrete design spaces. In this study, the search variables included the meta learning rate, the adaptation learning rate, the number of hidden layers, and the number of neurons per hidden layer. Each candidate configuration was evaluated by training the corresponding MAML model and computing a validation performance metric. Candidate solutions were then iteratively refined through selection and variation operations until a convergence criterion was satisfied. Because each fitness evaluation requires a full training and validation run, the optimization was executed on the University of Michigan Great Lakes high performance computing cluster to enable parallel evaluation of candidate configurations and reduce overall wall clock time [noa, a].

CHAPTER 3

Model Agnostic Meta-Learning for ASVs

Meta-Learning, often termed as "learning to learn", is a paradigm designed to enhance a machine learning algorithm's capacity to adapt to new tasks. This approach yields highly adaptable models that are well suited to be trained with limited data. Developed by Finn et al. [2017], Model Agnostic Meta-Learning (MAML) is such an algorithm, and specializes in its flexibility. MAML can be integrated with any model trained by gradient descent, and is viable for classification, regression, and reinforcement learning. This section frames Model Agnostic Meta Learning as the mechanism for few sample updating of motion surrogates, connecting predeployment knowledge to post-deployment measurements.

3.1 Algorithm and Setup

The core of MAML and similar Meta-Learning algorithms lies in their task-oriented structure. These tasks are defined by the following proposed framework. Each task ($\mathcal{T}$) encompasses several components: a loss function (L), an initial distribution over states ($q(x_1)$), a transition distribution that defines the probability of moving to the next state ($q(x_{t+1}|x_t, a_t)$), and the episode length (H). This framework allows MAML to efficiently train models by rapidly adjusting to new and varied tasks, a feature that is valuable in scenarios where data is scarce or rapidly changing, such as adapting pre-deployment engineering models to real-world data.

$$\mathcal{T} = \{L(x_1, a_1, ..., x_H, a_H), q(x_1), q(x_{t+1}|x_t, a_t), H\} \tag{3.1}$$

One feature of MAML is its ability to retain knowledge from previous tasks and use this accumulated experience to learn highly responsive parameters θ. These parameters are specifically tuned to adapt rapidly to new tasks, even when only a minimal amount of task-specific data is available. Retention of past information makes MAML a suitable candidate for structural predictions as memory of past large scale events is of utmost importance.

When MAML is applied to regression, such as in this work, the task loss is defined as the mean squared error between the model prediction and the observed response (Eq 3.2). In this chapter, the base learner is a fully connected neural network surrogate, but the MAML formulation applies more generally to any differentiable parametric model. This assists in the learning process by refining predictions based on the accuracy of previous outputs.

$$\mathcal{L}_{\mathcal{T}_i}(f_\phi) = \sum_{x^{(j)}, y^{(j)} \sim \mathcal{T}_i} \left\| f_\phi(x^{(j)}) - y^{(j)} \right\|_2^2 \tag{3.2}$$

The MAML algorithm operates through two distinct loops, each with its respective hyperparameters and application of gradient descent. Before the loops are iterated through, the initial model parameters, θ, are randomly initialized. The inner loop, or adaptation loop, focuses on individual tasks, each with k-shot sampled data points. In this loop, the model uses its current parameters θ to predict outputs for each of the input samples. The predictions are then evaluated against the actual outputs using the MSE loss function (Eq 3.2). This loss is scaled from the hyperparameter α and Stochastic Gradient Descent (SGD) is performed to generate the adapted model parameters, denoted as θ_i' for each task (Eq 3.3). The ability of the model to adjust its parameters (θ to θ_i) effectively based on immediate feedback from the loss function is key to its performance in new and varied tasks.

$$\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta) \tag{3.3}$$

After the tasks have been iterated through in the inner loop, a meta-update is performed (Eq 3.4). This step consolidates the learning from individual tasks to improve the overall model. The initial parameters θ are updated through performing SGD on the task-specific losses with

respect to the adapted parameters θ_i'. This process ensures that the model parameters (θ) are not tailored to any single task but rather optimized in a way that balances and incorporates learning from all encountered tasks. The complete MAML process is detailed in Algorithm 1.

$$\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'}) \tag{3.4}$$

Algorithm 1 Model Agnostic Meta Learning [Finn et al., 2017]

Require: $p(\mathcal{T})$ distribution over tasks
Require: α, β step size hyperparameters
1: Randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $D = \{x^{(j)}, y^{(j)}\}$ from $\mathcal{T}_i$
6: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ using D and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 3.2
7: Compute adapted parameters with gradient descent:
$\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
8: Sample datapoints $D_i' = \{x^{(j)}, y^{(j)}\}$ from $\mathcal{T}_i$ for the meta update
9: **end for**
10: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$ using each D_i' and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 2 or 3
11: **end while**

3.2 Initial Rapid Adaptation Exploration

MAML was first explored for it's suitability for ASVs to rapidly adapt pre-deployment models to match real-world conditions experienced once deployed and underway. These models can be generated from a variety of methods, such as CFD, closed-form solutions, strip theory, and others. Regardless, it is important for the vessel to undergo rapid transfer of learning to match its underlying assumptions to real-world data. In the absence or delay of this adaptation, the vessel risks operating with an operational envelope that is either nonconservative, increasing hazard exposure, or overly conservative, unnecessarily degrading mission capability.

Table 3.1: Main Particulars of initial vessel used for MAML exploration

Characteristic	Value
Length	270.0 m
Beam	32.2 m
Draft	10.85 m

3.2.1 Initial MAML Performance Exploration

To first explore MAML's adaptability, we created a dataset that composed of two separate distributions of data. The first phase of the dataset represents the pre-deployment training data, with the second part of the dataset being scaled by nonlinear factors to represent real-world data and the mismatch between engineering models and experienced conditions.

MAML was first trained using 5000 samples representing the pre-deployment training data. The large number of shots for the first phase of training is not a concern as this dataset represents the pre-deployment conditions. Each sample used a randomly selected weather sample from the ECMWF North Pacific hindcast data described in Tools Section 2.1.1. Ship velocity and heading were then randomly selected. Due to limitations of the closed-form solutions, headings within +/- 40° of beam seas (90, 270) were not selected. After weather, heading, and speed were selected, closed-form solutions were used to generate the VBM RMS values. The network was trained using this data set ($task_1$) and saved as the $task_1$ model. This model is representative of pre-deployment training, where models and operational envelopes are developed based fully on simulations or closed-form solutions.

To simulate post-deployment data, a second data set was then generated. In the absence of on-water data, a data set was initially generated in the same method as $task_1$. To emulate deployment mismatch in the absence of full scale measurements, a synthetic discrepancy was introduced by scaling the RMS response as a function of significant wave height (Eq 3.5). The scaling is inactive for H_s at or below 3.0 m, but increases linearly for H_s above 3.0 m to represent unmodeled nonlinear effects that become more pronounced in heavier seas. The resulting scaled response is treated as the $task_2$ target value and provides a controlled test of adaptation behavior.

$$f(H_s) = \begin{cases} RMS, & \text{if } H_s \leq 3.0m, \\ RMS \cdot \left(\frac{0.1H_s}{3} + 1\right), & \text{if } H_s > 3.0m \end{cases} \tag{3.5}$$

$$f(H_s) = RMS_{\text{task2}}$$

After the data set was collected, the saved model from $task_1$ was loaded and trained with a varying amount of shots from $task_2$. This problem design was constructed to evaluate a state-of-the-art machine learning algorithm for an autonomous vessel deployment and study its ability to quickly and accurately adapt its model to match real-world measurements.

The initial network used was a five layered fully-connected neural network with 32 neurons per layer. The dropout rate between layers was set to 0.1 and used to combat over-fitting. The MAML algorithm used this network to predict VBM RMS values from inputs of significant wave height H_s, wave period T_p, ship speed V, and the difference in heading between the ship and prominent wave direction μ. After experimentation, the MAML hyperparameters used were an adaptation learning rate of 0.02 and meta learning rate of 0.001.

Models were trained for up to 500 epochs per task, but given a early stopping condition of 20 epochs. If the validation loss continued to increase for 20 epochs, training would terminate early to avoid over-fitting. The model with the lowest meta-loss was then selected to be used for predictions.

Models trained with a varying amount of shots per task were evaluated against the second unseen scaled-RMS data-set from Equation 3.5. To assess performance, the standard deviation of the models predictions minus the ground-truth was used. Lower shot models may not have wave conditions over 3.0 meters; the cutoff point in which VBM RMS values are nonlinearly scaled. Because of this, an ensemble of 100 independently trained models was generated for each shot condition, using different random initializations and sampled adaptation sets. The reported performance metrics and plotted curves in Figure 3.2 use the ensemble mean prediction error across

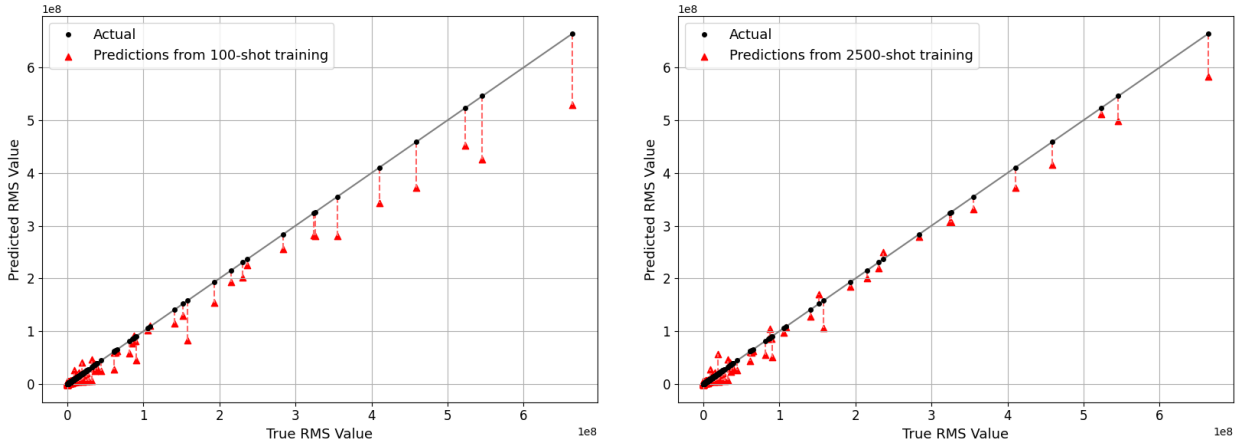


Figure 3.1: Prediction accuracy improving as number of shots increases.

these 100 models.

As the number of shots increases, prediction performance rapidly improves before beginning to taper off at the 1000 shot mark (Fig 3.2). The performance improvement of the 1500 shot increase from 1000 shots to 2500 is 8.89%. In contrast, the marginal 225 shot increase from 25 shots to 250 results in an improvement of 20.2%. The baseline condition, shown by the dashed red line in Figure 3.2, represents the model trained on pre-deployment training data generated from closed-form solutions and then evaluated on the second distribution test set which incorporates the nonlinear scaling.

Because MAML remembers prior experiences, the model tends to under-fit large RMS values. This is due to being first trained on the pre-deployment unscaled data from $task_1$. This initial MAML study characterizes the adaptation behavior of the meta learned surrogate under limited data and clarifies the operational meaning of few shot updating. The next section defines the predeployment dataset construction for the increased realism and challenge of the ProteusDS time-domain post-deployment data.

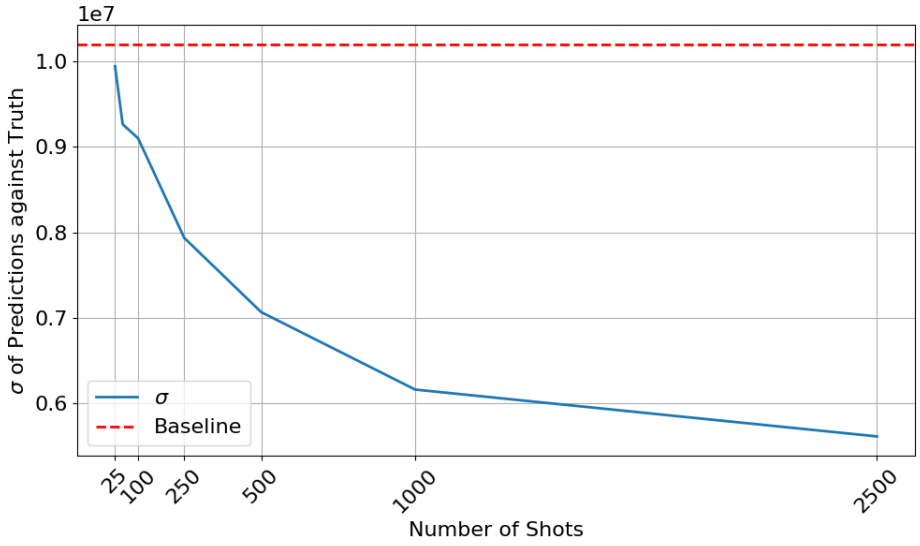


Figure 3.2: Improvements in accuracy for initial exploration of MAML predicting VBM

3.3 Time Domain Dataset for Increased Realism

After the initial exploration on MAML's suitability for ASV use in Section 3.2.1, a more realistic dataset was needed than one generated from closed-form solutions and simple nonlinear scaling to represent deployment data. The synthetic scaling study in Section 3.2.1 provides a controlled test of rapid adaptation but does not capture the modeling complexity associated with nonlinear excitation and time domain response estimation. A more realistic dataset is therefore generated using time domain simulations with nonlinear Froude Krylov excitation enabled of a model Vanguard unmanned surface vehicle, opposed to a panamax containership that Jensen used in his work. The hull model and main particulars can be found in Tools Section 2.3. The training data used for the the remaining of this chapter uses the same weather data from the previous section, and time-domain simulations from ProteusDS, also explained in the Tools Section 2.4. ProteusDS incorporated Froude-Krylov forces that vary nonlinearly as the simulation progresses. Each shot of data in the dataset represents the RMS value of motion for the vessel underway for 15 minutes. For a 50 shot mission, this corresponds to 12.5 hours underway. For a 3000 shot mission, this depicts a monthlong mission.

3.4 Model Set Up and Scoring Metrics

The underlying algorithm that MAML uses in this work is a fully connected Artificial Neural Network (ANN). The architecture of the neural network is five fully-connected layers with 32 neurons per layer. The adaptation learning rate, which is responsible for the descent rate within the inner loop of MAML was set to 0.001. The meta learning rate, which determines rate of changing initial parameters θ, was set to 0.003. These hyperparameters were selected after undergoing architecture and hyperparameter optimization using a genetic algorithm as outlined in Section 2.9.

The model features are significant wave height, wave period, ship velocity, and relative heading between the ship and incident wave. To improve performance, the relative heading was decomposed into *sine* and *cosine* components to prevent wraparound that occurs at $(360°, 0°)$. The same features are utilized to predict either pitch or heave RMS values, as well as slamming frequency if desired.

To show general trends, 30 individual models were trained at each evaluation condition. Model accuracy was determined by finding the absolute mean of the 30 models Z-scores from MAML predictions against the ground truth. This metric was chosen due to its non-dimensionality and representation of how similar the group of model's predictions were to the ground truth. As a result, accuracy of models can be compared between degrees of motion and against test sets that contain large or small RMS values.

The mean of the 30 model's predictions are saved for each sample i in an n sample test set.

$$\hat{y}_i = \frac{1}{30} \sum_{m=1}^{30} \hat{y}_i^{(m)} \tag{3.6}$$

The residual between ground truth and averaged predictions is found for all test points.

$$\bar{y}_i = y_i - \hat{y}_i \tag{3.7}$$

Compute Z-scores by normalizing residuals using the standard deviation of actual values σ_y.

$$\sigma_y = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{3.8}$$

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \tag{3.9}$$

Lastly, calculate the mean absolute Z-score across all n test samples. This represents how accurate the batch of models were to ground-truth.

$$z = \frac{\bar{y}}{\sigma_y} \tag{3.10}$$

$$\mu_z = \frac{1}{n}\sum_{i=1}^{n} |z_i| \tag{3.11}$$

3.5 Rapid Adaptation of Pre-Deployment Engineering Models to Real-World Conditions

To examine MAML's suitability for rapid adaptation, a transfer-learning approach was employed. To begin, a model was trained using 5000 samples of data generated from closed-form solutions representing pre-deployment engineering models. It was trained using features described in Section 3.2.1 to predict Jensen's closed-form pitch RMS values.

After training the model on pre-deployment closed-form pitch RMS values, the model was saved. This model is referred to as the baseline model. It was then loaded and further trained, this time with ProteusDS time-domain pitch RMS values representing real-world data. The amount of ProteusDS data available for training, or shots, was increased to show how predictions against a ProteusDS test set improved as more data became available and how quickly pre-deployment bias would be phased out. As detailed in Section 3.4, 30 individual models were trained at each shot level to prevent outlier models from skewing overall trends.

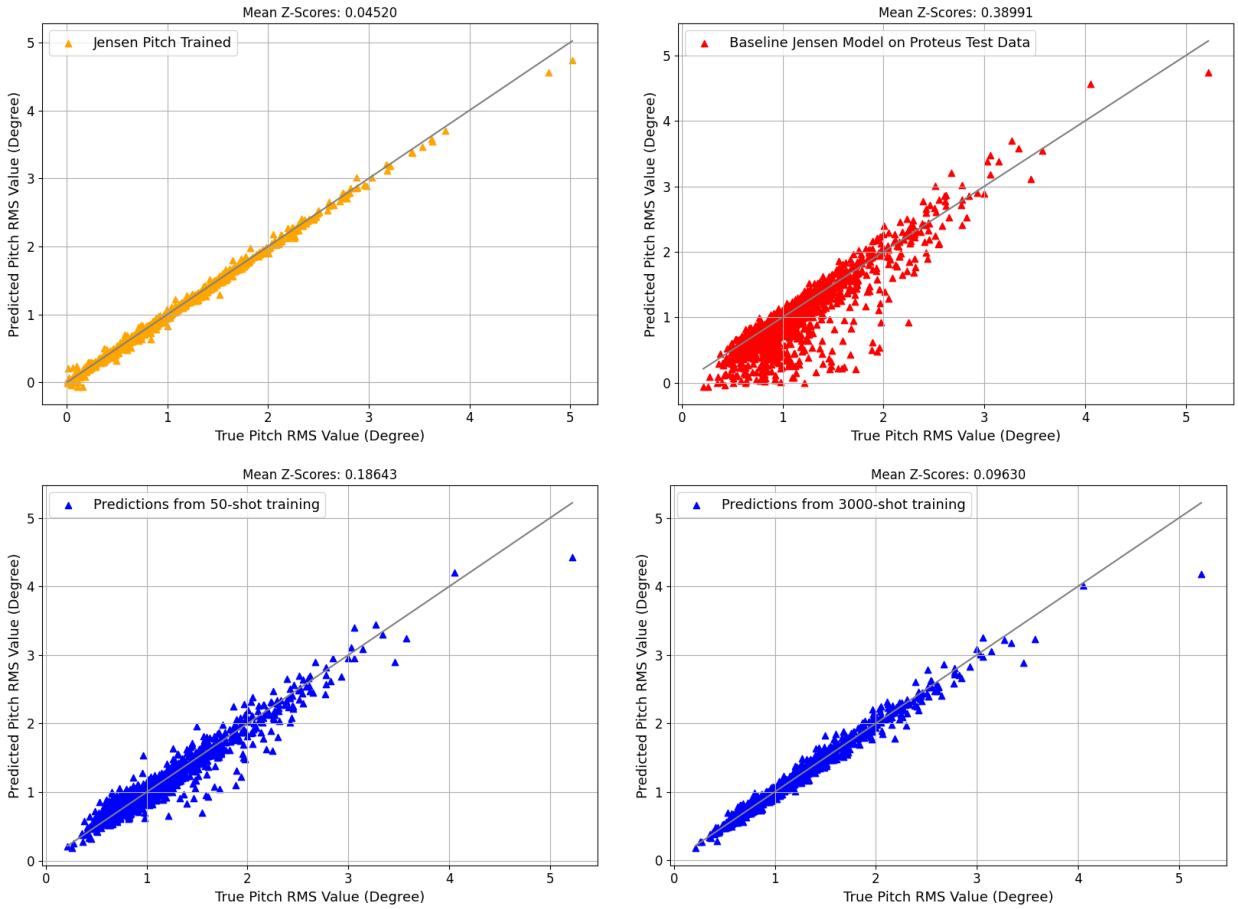


Figure 3.3: Prediction accuracy improving as number of shots increases.

When trained and evaluated on the closed-form pitch RMS values, the baseline MAML model accurately predicts against an unseen test dataset consisting of values generated by the same method (closed-form solutions). When trained and evaluated on closed form pitch RMS values, the baseline model reproduces the generating distribution, as indicated by the tight clustering about the 45 degree line in the upper left panel of Figure 3.3. When the same model is evaluated on time domain simulations that include nonlinear Froude Krylov excitation, systematic under prediction becomes apparent in the upper right subplot. The mean Z-score increases from 0.045 to 0.390, corresponding to a 762.65% increase in error relative to the closed form test set. This degradation motivates rapid adaptation of predeployment surrogates using limited real-world observations. Employing out-of-the-box methods and not updating the engineering models to match real-world data risks the vessel venturing into danger.

After 50 shots of data, or 12.5 hours of data collection, the accuracy rapidly improves and is approximately 52.1% more accurate when evaluated on data representing real-world conditions than the baseline model. Looking at the lower left subplot of Figure 3.3, predictions are increasingly distributed around the center of the ground-truth line when compared to the baseline model tested on time-domain data (red). The final predictions after 3000 shots of time-domain data are not as accurate as the initial model predicting against closed-form solution RMS values due to the variability of gathering RMS values from 900 seconds of data and the inclusion of the nonlinear Froude-Krylov excitation force.

Figure 3.4 quantifies the mean Z-score as a function of the number of adaptation shots. The curve exhibits diminishing returns beyond approximately 500 shots, indicating that most performance gains occur with limited adaptation data, while additional shots provide only marginal improvement.

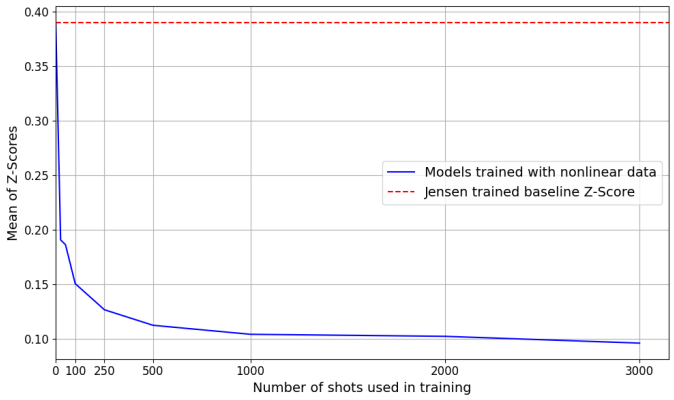


Figure 3.4: Performance increases rapidly before tapering off as number of shots is increased

Few shot updating can correct predeployment bias when higher fidelity or operational measurements deviate from engineering assumptions. The next section tests robustness in extreme seas, where nonlinear effects and safety consequences are largest.

3.6 Extreme Weather

Besides rapid adaptation of the model to real-world data after deployment, an additional few-shot learning challenge arises when experiencing large-scale wave events for the first time. The ocean has the capacity to rapidly build in intensity. Due to this, the ability to quickly learn in new, potentially hazardous, heavy weather conditions is paramount to vessel safety. Without rapid learning of these conditions, the model may assume that previous training in calm water can be accurately extrapolated to extreme conditions. As a result, the vessel may venture into hazardous wave conditions it deems safe.

To test MAML's ability to rapidly adapt to extreme weather, two datasets were generated that were split at a significant wave height of 3.5 meters. Both datasets used all heading angles and speeds ranging from 0-20 knots. The initial baseline model was trained solely with with 3000 shots of randomly selected weather with wave heights ranging from 0-3.5 meters. The targets of the small wave dataset were pitch RMS values collected from the time-domain ProteusDS simulations. The closed-form solutions were not used as the starting model's targets, representing that the vessel had already been transiting at sea for some time and pre-deployment bias had been effectively phased out of the model. The baseline model's predictions against a separate small wave test dataset is shown in the top left subplot of Figure 3.5. The top right subplot presents the baseline small wave model evaluated on a test dataset consisting of extreme weather samples. Because the nonlinear Froude-Krylov excitation force has larger effects in more extreme weather, the baseline model does a poor job predicting extreme RMS values [Lewis and Society of Naval Architects and Marine Engineers (U.S.), 1989].

After training the initial baseline model, 30 models were then cloned from the baseline and exposed to an increasing amount of data from the extreme wave dataset ($H_s > 3.5$). After 50 shots of large weather data, the distribution of predictions becomes increasingly centered around the true RMS line when compared against the baseline model, as shown in Figure 3.5. As shots increase, predictions for the scarcer extreme values in the large wave test set become increasingly refined. These improvements will be further examined in Section 3.8, where the large wave models are used

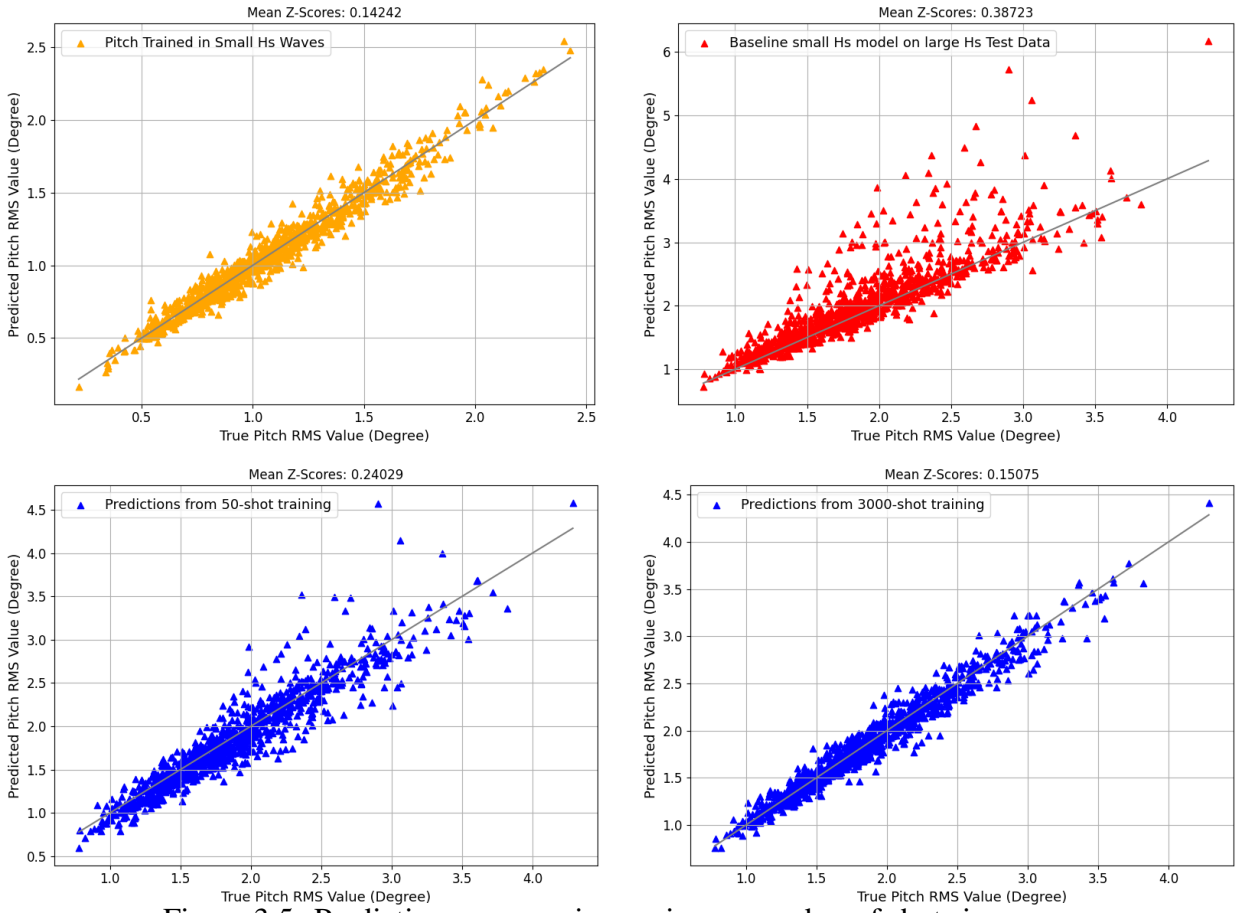


Figure 3.5: Prediction accuracy improving as number of shots increases

to generate Operational Envelopes.

As shown in Figure 3.6, performance rapidly improves until approximately 250 shots of data where improvements become more marginal with an increase of training data. Importantly, although the improvements are less extreme between higher shots of data, the predictions improving most are the extreme values in the large wave test set. These are conditions likely to be on or past the operational envelope and important to accurately predict.

3.6.1 Prediction of Slamming Frequency

After the initial research on MAML was completed, I attempted to predict slamming frequency in an online training experiment in order to increase the complexity of the created envelopes. Slamming is introduced as a discontinuous response phenomenon that challenges smooth surrogate models

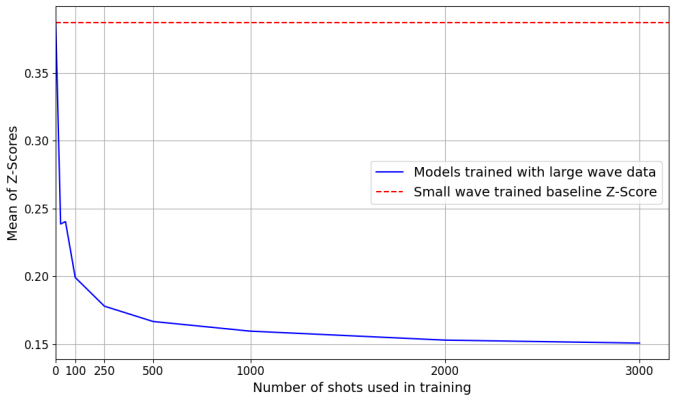


Figure 3.6: Performance increases rapidly before tapering off as number of shots is increased

and whether task based learning can capture threshold driven dynamics. To extract slamming frequency from the time domain simulation, a wave-probe was added to the hull of the model Vanguard in ProteusDS, and entry velocities of the hull reentering the water after broaching were monitored. This is further explained in Tools Section 2.4.

Based on how the Extreme Weather Section 3.6 was structured, predicting slamming frequency was a challenge because slams rarely occurred within the small-wave initial dataset. As such, the model is presented with a significant challenge of adapting its prior bias that slams per hour are trivial or happen very rarely, to large-wave conditions where they happen much more frequently and with greater intensity. This work is also explored in Chapter 4, but MAML's performance in slamming frequency prediction is vastly superior. Figure 3.7 quantifies adaptation behavior for slamming frequency prediction under extreme weather conditions. The baseline model trained exclusively with small wave data (includes rare instances of slamming), severely underpredicts slamming frequency on the operational envelope test set that includes extreme weather. This is demonstrated by almost all predictions of the baseline model falling below the ground-truth diagonal. The model trained with 50 shots of extreme weather exhibit substantial increase in prediction dispersion, but the predictions begin to fall on both sides of the ground-truth line, with an overall increase in accuracy of 31.7%. After adaptation with 750 shots, predictions cluster more tightly around the diagonal and the standard deviation of z scores decreases to approximately 0.21, with limited additional improvement at the 3000 shot level. Extreme event metrics such as slamming

represent a distinct challenge because they are sparse, nonlinear, and often outside the distribution captured during routine operation, making them central to safety-based envelope construction. These results indicate that the meta learned initialization supports rapid correction of rare event statistics despite the scarcity of slamming events in the small sea training data, and provides a concrete basis for comparing MAML against uncertainty based alternatives in Chapter 4.

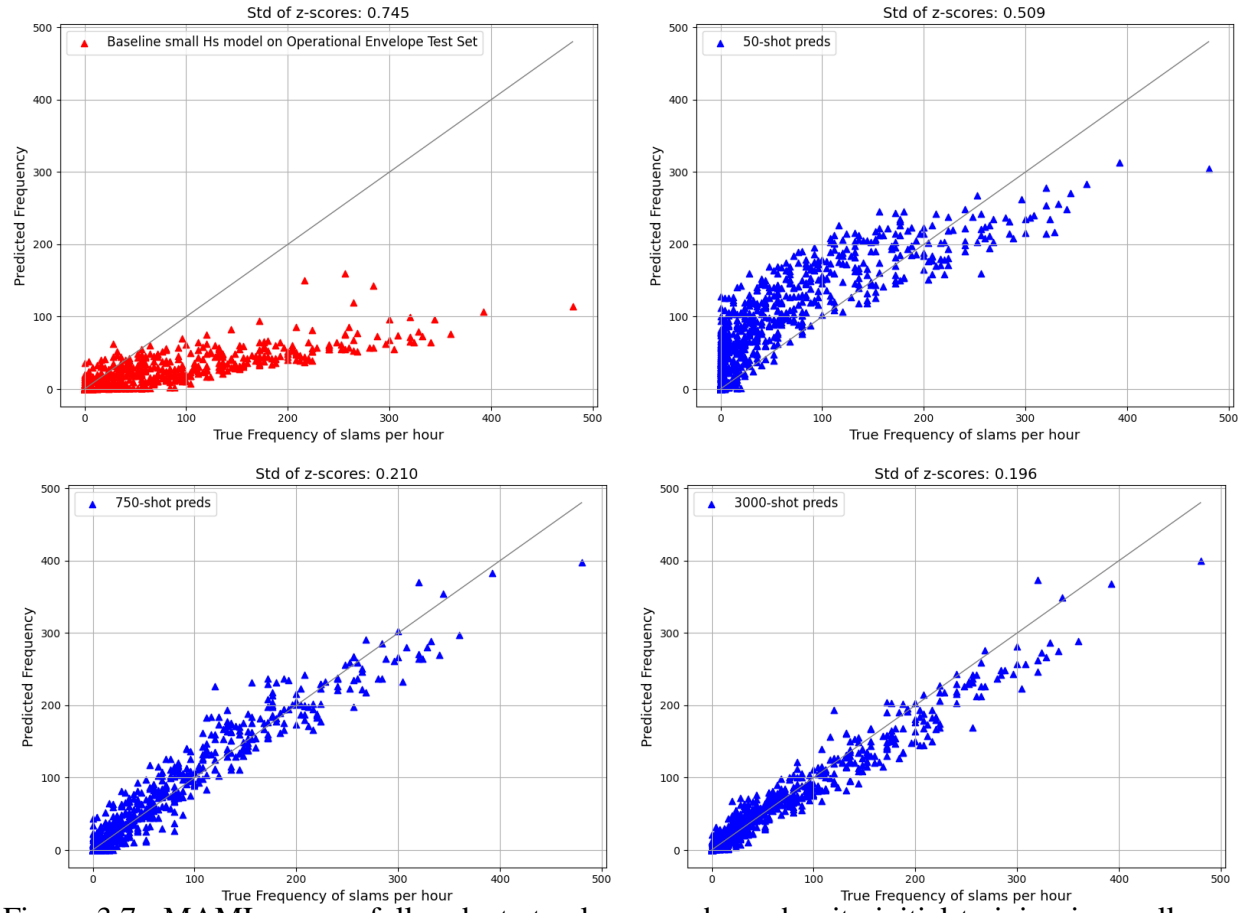


Figure 3.7: MAML successfully adapts to slams per hour despite initial training in small wave training with rare slamming events

3.7 Forgetfulness

This section evaluates catastrophic forgetting as an operational failure mode, where sequential adaptation to recent missions degrades competence on previously learned extreme conditions. When assessing the validity of MAML for Autonomous Surface Vehicles, it is important to ensure that the

model does not forget valuable information during missions in small seas. If this occurs, predictions of rarer large wave events would become more inaccurate to the detriment of a safe/unsafe classifier, and the model would have to repeat the learning of large waves detailed in Section 3.6.

To test this, an initial baseline model was trained on the extreme wave dataset from Section 3.6, ensuring that it accurately predicted large waves from the heavy weather test set. Three separate datasets of 100 samples were collected, all with significant wave heights ranging from 0-2.5 meters. The initial model trained with extreme waves was cloned thirty times and each of the models were then trained with the three small wave datasets in succession. After each model reached the early stopping criteria, it would progress to the next small wave dataset until all missions had been completed. This framework was designed to replicate a ship undergoing an extreme weather mission, followed by three much more common missions of 25 hour durations occurring in calm seas.

A core principal of MAML is that each task is representative of the distribution as a whole, represented in this work by all observable ocean conditions. This fundamentally opposes the application for online learning in a marine environment, where each shot of data is closely related to the prior shot. As MAML seeks to find sensitive initial parameters (θ) that serve as an initial starting point to adapt to new tasks, successive meta-updates with similar data result in parameters finely tuned to the similar tasks. Due to the lack of diversity in the successive calm water missions, the meta-update eventually moves the parameters through successive rounds of gradient descent towards a starting location sensitive to small water conditions (Fig 3.8). Because of this, when the model is then evaluated on the extreme wave test set, it performs poorly. Compared to the baseline model, large wave prediction accuracy decreased by 121% by completion of the three missions.

While MAML shows promise in its ability to rapidly adapt its pre-deployment engineering models to real-world conditions, the out-of-the-box algorithm is not a suitable candidate for ASV use due to forgetting of critical events. Although MAML could rapidly relearn large scale wave events, memory of conditions that produced damage could be forgotten, and the time required to re-adapt the model could result in sailing into hazardous conditions.

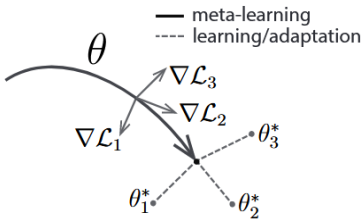


Figure 3.8: Effectiveness of MAML is dependent on diversity of tasks

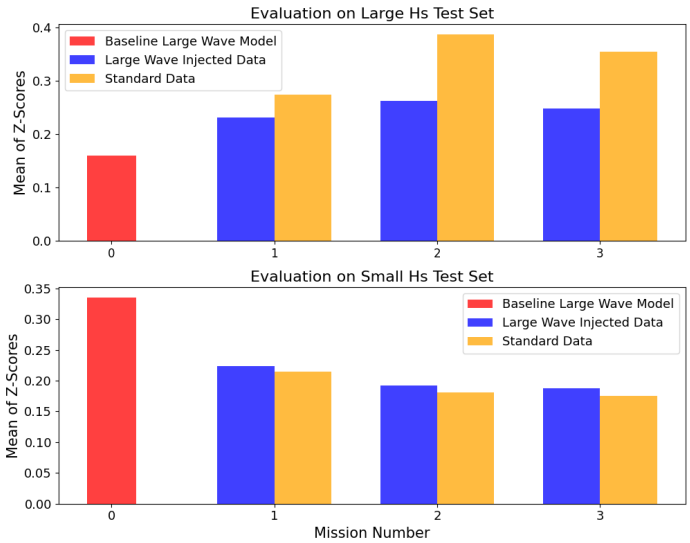


Figure 3.9: Effectiveness of MAML after successive calm water missions

To attempt to mitigate the issue of forgetfulness, important extreme wave data was interjected into each task. For each task of 30 shots, 6 samples of previously collected data from the large wave dataset used for initial training were randomly re-added into the new task. This ensures the task be more representative of the distribution as a whole and was designed to mitigate forgetting critical events while completing calm-water missions. The choice to recycle 20% of large wave data came from a balance of needing to both accurately predict current conditions and remember important critical events from large sea states.

Figure 3.9 compares model performance after each mission using both the standard MAML model and injected data model. The lower bar plot evaluates performance against a test dataset ranging from 0-2.5 meters significant wave height, which is what weather missions 1, 2, and 3 contained. The standard model does a slightly better job at predicting the conditions at hand.

However, using injected data helps mitigate forgetting of knowledge against the large-wave database. This is highlighted in the top bar plot, where the models are evaluated against the large-wave test set after successive missions. Interestingly, performance for both types of MAML improve during calm water mission 3. Compared to the standard model, the injected model is approximately 27.7% more accurate on predictions against the large wave test set after three successive missions.

The forgetting analysis connects task composition and online updates to retention of safety critical competence, motivating explicit strategies for maintaining coverage of rare extreme events. The next section translates these learning dynamics into their practical consequence through operational envelope evolution.

3.8 Operational Envelope Construction

Using predictions from the MAML-adapted networks, an Operational Envelope can be constructed to determine what heading, speed, and wave conditions are safe for a given task. To create the sea keeping envelope, the user provides an RMS limit for each degree of freedom for the task at hand. Depending on the task, this could vary widely. When it comes to critical safety of the vessel, heave is of less concern than other degrees of freedom and moments. However, for a task such as launching/recovering a drone from the deck it may become increasingly important. By defining limits for each degree of freedom, the operational limits overlap and present a clear way to understand what headings and speeds are safe to operate in based on the weather. For this demonstration, a heave RMS limit of 2.25 meters and pitch limit of 3.0 degrees were selected. These limits were arbitrarily chosen because they provide limit-states in rarer extreme weather, as well as overlapping with the other degree of freedom to showcase the Operational Envelope. For each of the subplots in Figure 3.10, the ground-truth limit-states from the ProteusDS time-domain simulations are presented by a black line. The predicted limit-states generated from the trained models is presented in blue.

To assemble the operational envelope dataset, weather samples were randomly selected until

each bin of the polar plot contained five samples. Each bin spans 10 degrees representing ship heading and 0.45 meters along the radial axis representing significant wave heights that range from 1-10 meters. The bin color is determined by a color map based on the RMS value of the bin and uses the mean of the five samples contained in each bin. It is important to note that the scale of the color map is different between pitch and heave. For this envelope, ship velocity was chosen to be 13 knots. For human crew to use the Operational Envelopes as a visualization tool to analyze ASV limits, a collection of these envelopes at varying speeds may be used.

The models selected to use for the Operational Envelope were from Section 3.6, Few-Shot Learning of Unseen, Large-Scale Wave Events. These models were designed to explore MAML's ability to adapt to extreme weather when encountered for the first time. The 30 trained models were evaluated at the 50 and 3000 shot condition for both pitch and heave. For each of the five weather samples contained in every bin, the 30 models predictions for each sample was averaged. Lastly, the mean of the 5 samples was taken from model-averaged predictions in every bin, resulting in an overall RMS prediction for that bin. Results for pitch and heave after being exposed to 50 and 3000 shots of large wave data are shown in Figures 3.10,3.11.

Model predictions for the heave limit-state are closely aligned to the ground-truth, highlighted by the overlapping blue and black lines in the top subplots of Figure 3.10. Increasing the amount of large-wave training data from 50 to 3000 shots has little improvement on changing the predicted limit-state. Compared to pitch, heave is a less challenging degree of freedom to predict, which is why pitch is used for experiments in Section 3.2-3.7. This is revealed in the polar plots by the heave ground-truth limit being more circular than pitch, and the color variation from the predictions increasing more uniformly as significant wave height increases.

When exposed to extreme weather for the first time, a process described in Section 3.6, the ASV initially under-predicts pitch RMS values at large wave heights. This is likely due to the increased contribution of the Froude-Krylov excitation forces in large waves. As the vessel starts to gain experience in heavy weather, it dangerously predicts its operational limits further than the ground-truth limit. This creates a hazardous condition where the ASV may travel beyond its

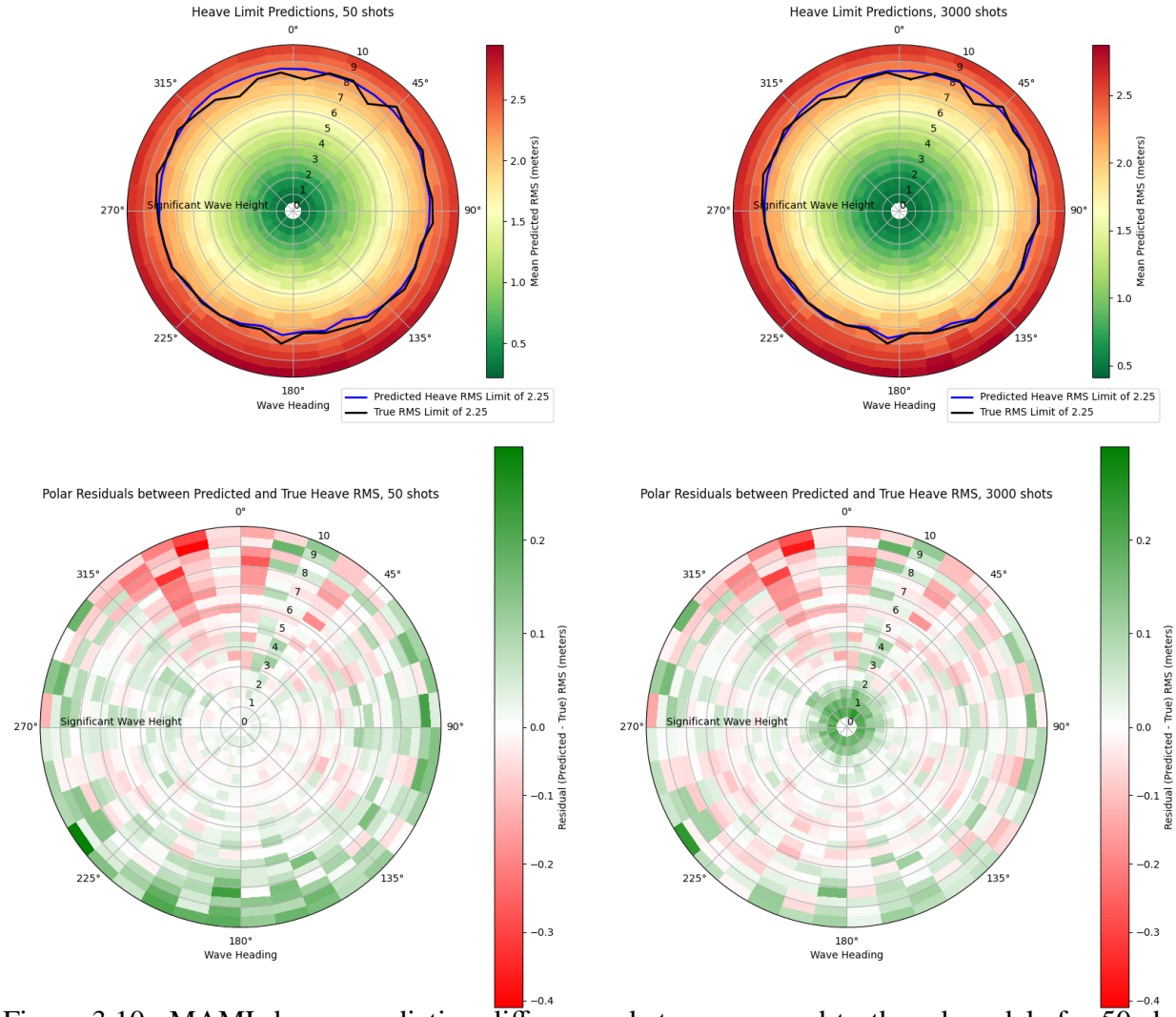


Figure 3.10: MAML heave prediction differences between ground truth and models for 50 shot condition (left) and 3000 shot condition (right)

true operational boundaries into conditions that can damage the vessel. As the amount of large wave training experience increases from 50 to 3000 shots, RMS predictions for pitch increase at larger significant wave heights. This is shown through the lower subplots in Figure 3.11 becoming increasingly red as the ASV gains experience completing missions in heavy weather. This additional training moves the predicted operational limit in pitch to be more closer aligned to the ground-truth limit state.

Performance improvements due to increasing amounts of available large wave training data were highlighted through residual plots in Figure 3.11. These subplots were created by subtracting

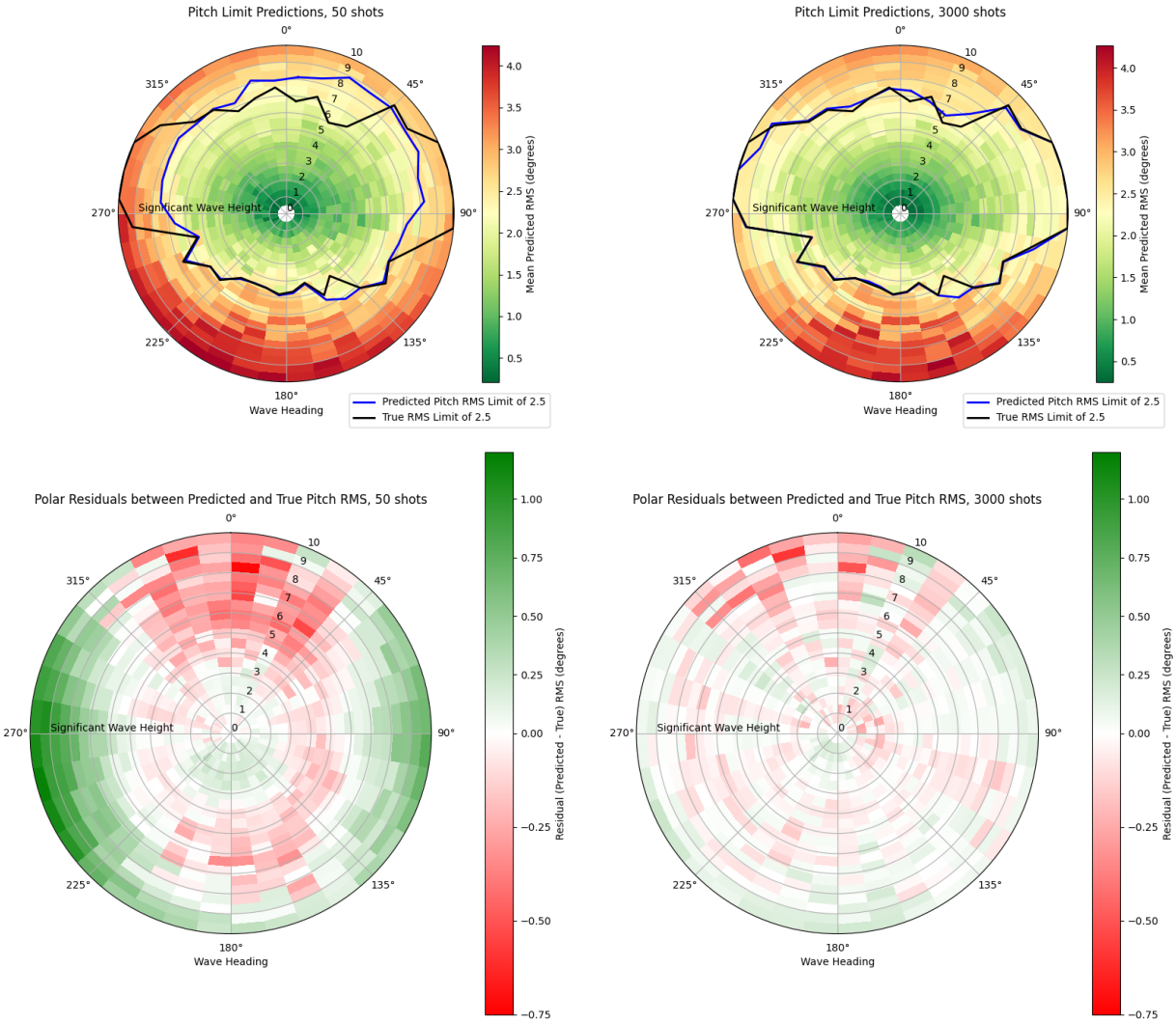


Figure 3.11: MAML pitch prediction differences between ground truth and models for 50 shot condition (left) and 3000 shot condition (right)

the predicted mean RMS bin value against the ProteusDS ground truth. The color bar was restructured to have red as the model over-predicting data, green as under-predicting data, and white showing no difference between predictions and truth.

Because the initial baseline model was trained on a dataset consisting of weather samples with significant wave heights ranging from 0-3.5 meters, the 50 shot pitch model performs well in similar conditions, but poorly in larger waves where Froude-Krylov excitation forces are more severe. This is shown in the lower subplots of Figure 3.11, where from 0-3.5 meters the color is pale green, and outside is dark red. After being exposed to additional heavy weather, the outer rings of

the 3000 shot pitch model contain a mixture of white, pale red, and pale green highlighting that the distribution of predictions has moved closer to ground-truth heavy weather RMS values. Because the second dataset does not contain weather with significant wave heights ranging from 0-3.5 meters, the issue of MAML "forgetting" occurs again, this time with MAML forgetting knowledge of small-wave events after extensive training in large waves. As a result of training data expanding from 50 to 3000 shots of extreme weather data, calm water predictions worsen and over-predict slightly. This results in the center of residual pitch subplots changing from a white/pale green to light red color. To mitigate this issue, small wave data should be injected into each task to make it more representative of the distribution as a whole. This is an underlying MAML assumption that is ignored with an online learning approach. The data injection process and reasoning is further explained in Section 3.7.

The operational envelope is constructed by combining the predicted heave, pitch, and slamming limit-states: at each heading, speed, and wave condition. The contour limit-state is determined by the first-exceeded RMS threshold of pitch, heave, and slamming.

The model adapted with 50 shots of large wave data can misidentify the governing limit state in specific heading sectors because the limited adaptation set does not sufficiently cover the joint distribution of heading, speed, and sea state. In beam seas, pitch amplitudes are low, but limited exposure to extreme wave conditions can leave residual bias in the adapted ensemble, leading to incorrect attribution of the controlling limit state in portions of the envelope. In following seas, the 50 shot adapted envelope can remain overly permissive and extend beyond the ground truth boundary, implying feasible transit into unsafe conditions. When the adaptation dataset increases to 3000 shots, the envelope contracts toward the true limit states and the governing degree of freedom transitions more consistently between pitch, heave, and slamming as heading varies.

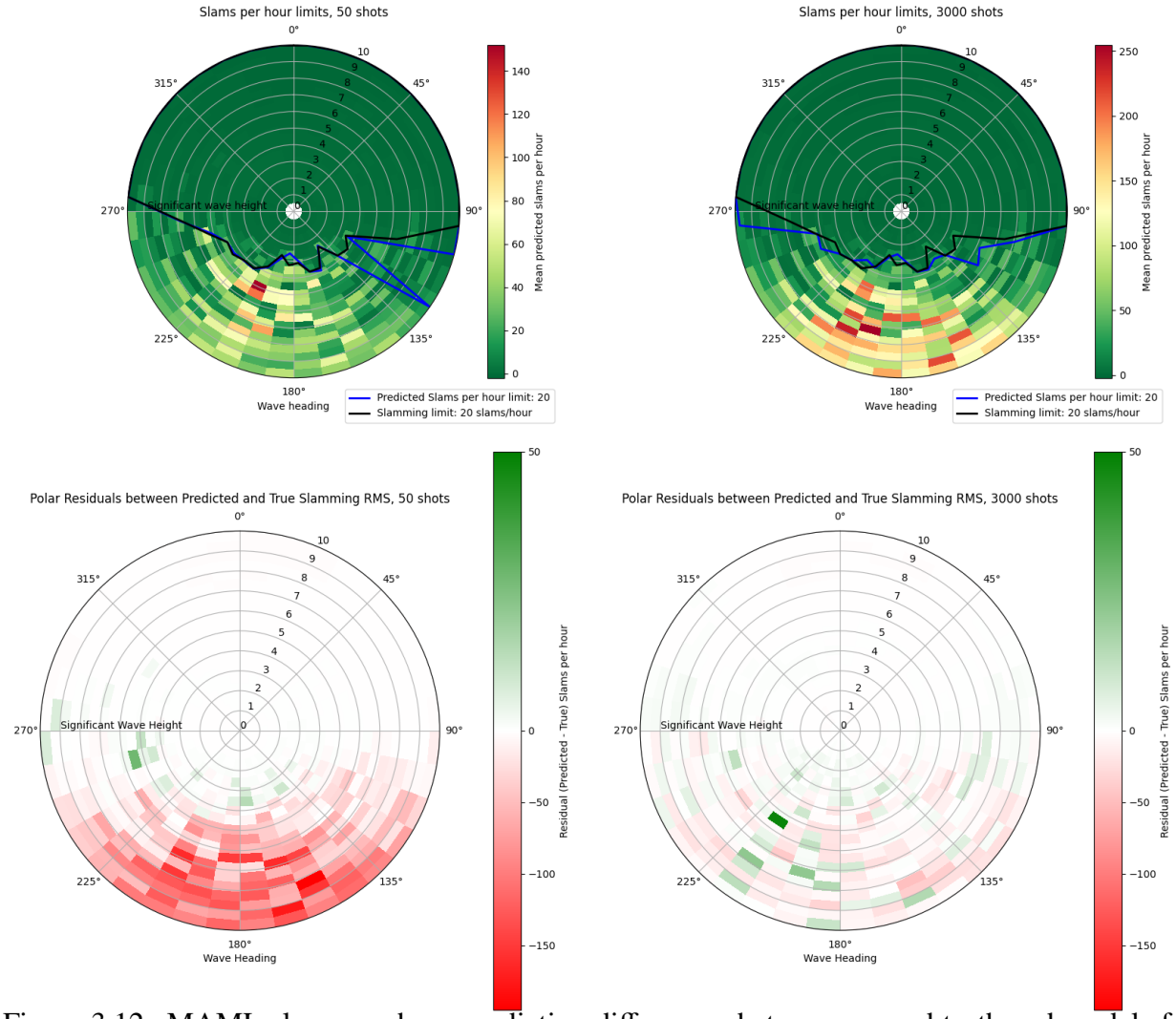


Figure 3.12: MAML slams per hour prediction differences between ground truth and models for 50 shot condition (left) and 3000 shot condition (right)

3.9 MAML Conclusions for ASVs

Large-scale autonomous surface vessels will increasingly rely on rapid online adaptation of pre-deployment models to ensure they never venture beyond their true operational limits. In this work, we demonstrate that Model-Agnostic Meta-Learning, with a simple fine-tuning modification, can transform a surrogate model trained on closed-form solution data into a model that accurately predicts both heave and pitch RMS as well as slamming frequency limit-states under previously unseen, extreme sea conditions with only a few observations. This attribute, along with the algorithm's inherent flexibility, make it suitable for use with autonomous vessels, where quick

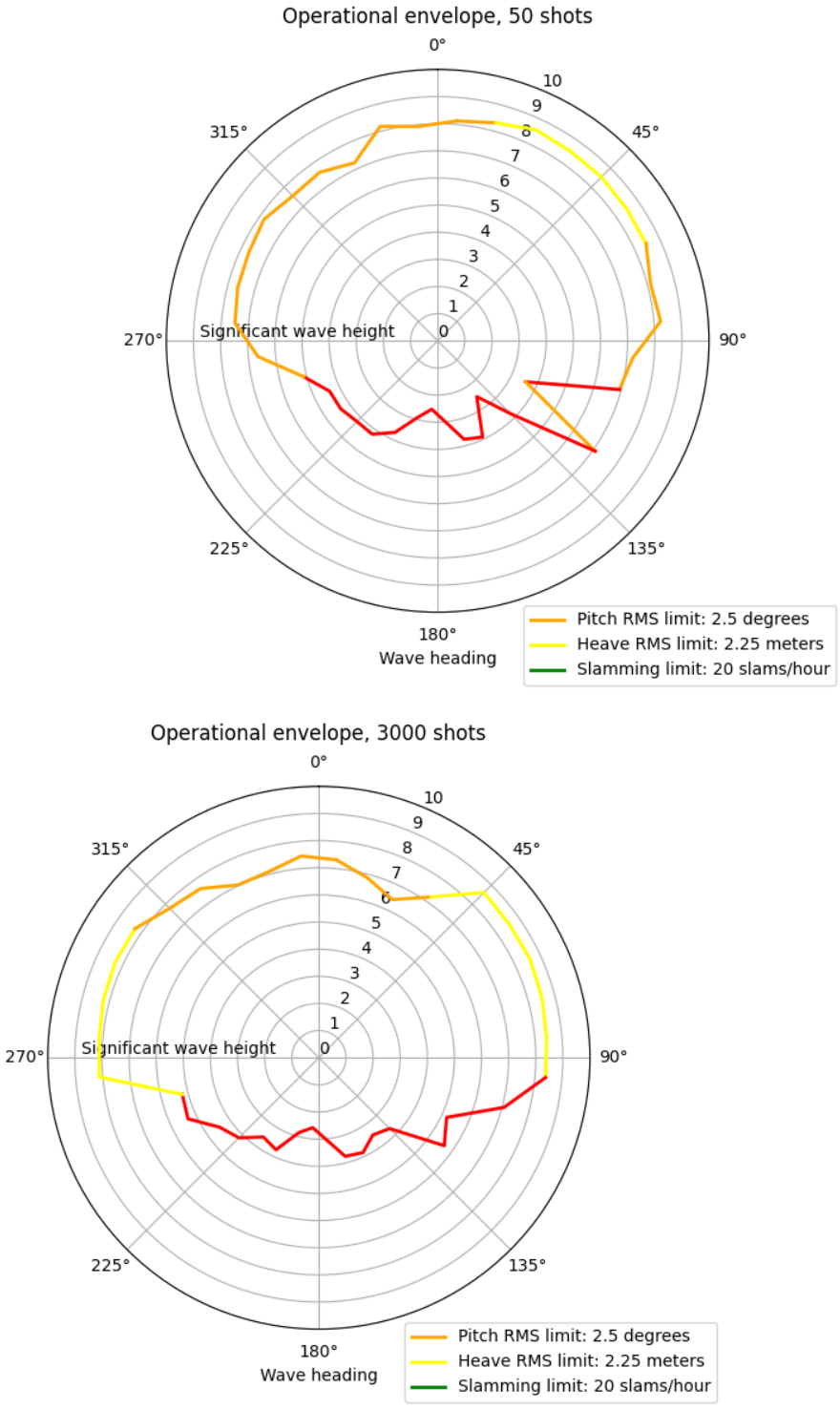


Figure 3.13: Evolution of Operational Envelope

adaptation is essential. One use case is the few-shot learning challenge from experiencing heavy weather for the first time. As shown in Section 3.6, the vessel behaves different in extreme weather due to nonlinear forces and motions compared to more routine conditions. This results in the models initially providing poor predictions for limit-states, which could result in the vessel sailing into hazardous conditions when first experienced, as demonstrated in Section 3.8. As MAML gains experience in the new conditions, predictions rapidly improve and the Operational Envelope converges to the true limit-state. However, as a result of training in an online manner, tasks consist of similar shots of data instead of a collection representing the entire distribution of all weather and mission parameters. Due to this, MAML suffers from "forgetfulness", and predictions on past experiences become worse.

By injecting the meta tasks with important shots from prior missions, MAML can mitigate the issue of forgetting experience from large wave events while undergoing missions in calmer weather. This works by ensuring that a portion of the task at hand is representative of the training distribution as a whole. However, more work needs to be done exploring this topic and how to best maintain accurate extreme motion predictions regardless of current weather.

A limitation of the MAML implementation used in this chapter is that it provides deterministic point predictions and does not quantify predictive uncertainty. This restricts risk aware operational envelope construction because the safe unsafe boundary is inferred from mean estimates rather than from probabilistic exceedance metrics.

This chapter demonstrates that rapid adaptation is feasible and operationally meaningful using MAML, but it also reveals that adaptation alone does not guarantee safe behavior under distribution shift, nonlinearities, or sequential learning. Chapter 4 therefore evaluates uncertainty aware learning models that can represent confidence and support conservative envelope construction when data is limited, biased, or novel.

CHAPTER 4

Evaluating Uncertainty-based Machine Learning Models

Operational envelopes for autonomous surface vessels may be defined by motion and load limit states over the space of sea state, heading, and speed. When learned surrogates are used to predict these limit state responses, the most safety relevant predictions occur near the envelope boundary where the probability of exceedance transitions rapidly from low to high. Training data in this boundary region is typically sparse because operations are biased toward benign conditions and extreme sea states are potentially hazardous to sample. Under these conditions, small modeling errors can shift the predicted boundary away from the true limit state, which can lead either to unsafe operations or to unnecessary conservatism. Deterministic regressors provide only point predictions and therefore cannot indicate whether a limit state estimate is supported by interpolation within the training distribution or driven by extrapolation beyond it. For safety critical autonomy, uncertainty quantification must therefore be treated as a required component of operational limit prediction and downstream decision making.

There is a wide range of stochastic models to evaluate for suitability of an autonomous vessel. Previous sections began to explore the subject by using ensemble learning and leveraging the predictions of multiple models trained from slightly different batch combinations of training data. Leveraging the mean and standard deviation of the ensemble's predictions resulted in higher confidence for limit state prediction. However, examining the spread of our predictions in the ensemble in Figure 4.1, the variation in prediction was extremely low. This was due to our

transfer learning training structure. To simulate an ASV first gaining experience in routine weather before routing in potentially hazardous conditions, a model was first trained with 3000 shots of routine weather ($H_s < 3.5m$) before being exposed to an increasing amount of extreme weather ($H_s \geq 3.5m$). Because every model in the ensemble was loaded from the initial routine weather base model and fine tuned with the same hyperparameters and weather data, the overall variation between predictions was low. The variation in the ensemble's predictions would have been greater if the fine tuning process used separate models from the start, rather than all being clones of the single model after the first stage of training. Our approach did not follow the above method because at the time we were more focused on removing outlier predictions and showing general trends of rapid adaptability than capturing model uncertainty.

The following chapter evaluates three models, each with innate uncertainty quantification, for assessing validity for use onboard an ASV. Bayesian Neural Networks, Deep Evidential Regression, and Deep Kernel Learning were all implemented and evaluated on accuracy, rapid adaptation, uncertainty quantification, and forgetfullness.

4.1 Bayesian Neural Network (BNN)

The first evaluation uses Bayesian neural networks as a baseline probabilistic approach for motion limit prediction, examining whether parameter uncertainty yields useful safety relevant confidence estimates.

Deterministic deep neural networks provide point predictions and do not represent uncertainty in model parameters. In regimes that differ from the training distribution or contain substantial noise, this limitation can yield overconfident predictions. Bayesian neural networks represent uncertainty by treating network parameters, including weights and biases, as random variables described by probability distributions rather than fixed values [Lebede and Nadarajah, 2025]. The objective is to infer a posterior distribution over parameters conditioned on the training data, which enables uncertainty quantification in addition to mean prediction.

Two uncertainty sources are relevant. Epistemic uncertainty captures uncertainty in model

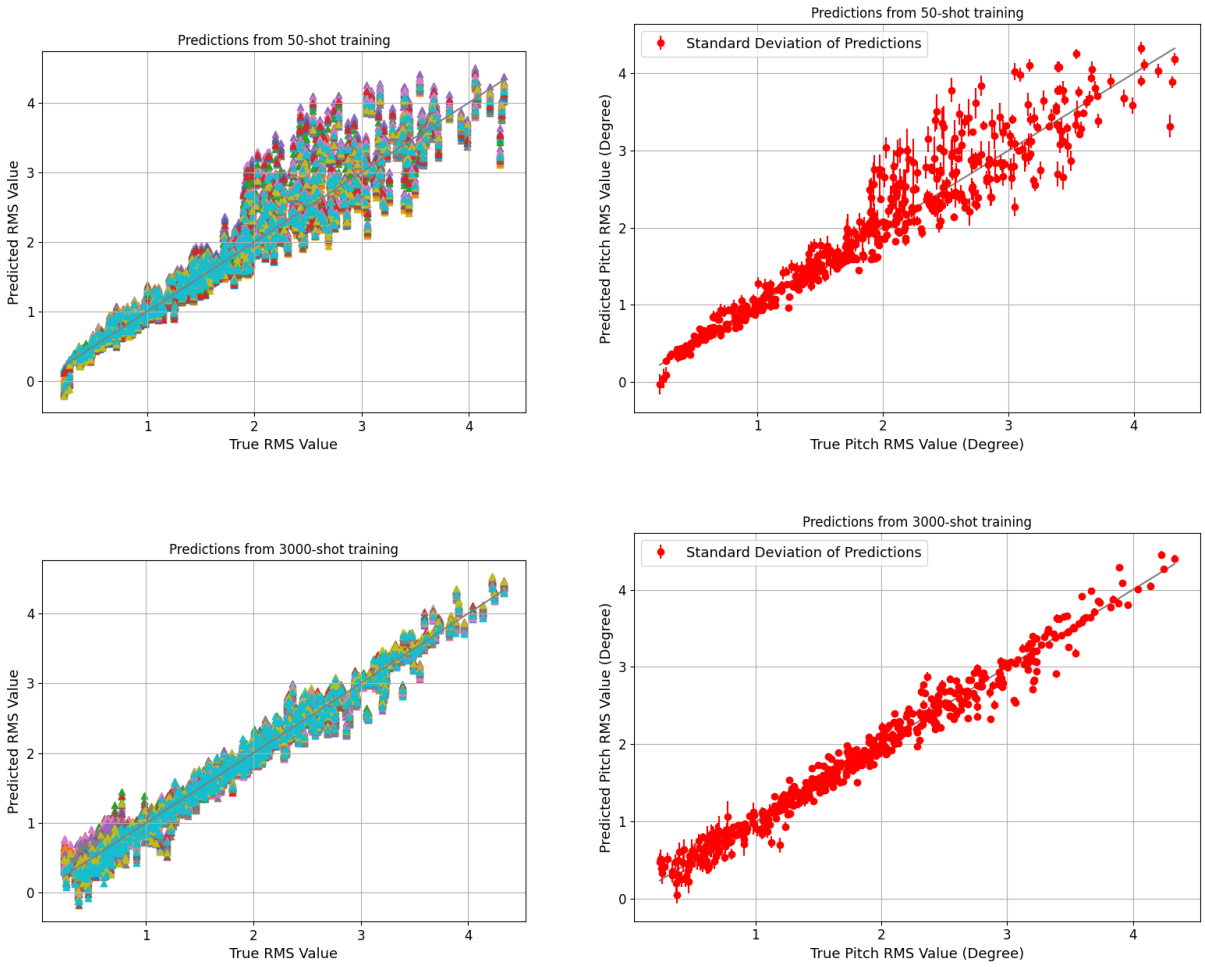


Figure 4.1: Fine tuning the base model from the routine weather mission results in low variation of predictions from the models within the ensemble

parameters due to limited or unrepresentative training data. Aleatoric uncertainty captures irreducible variability in observations due to measurement noise or intrinsic stochasticity in the data generating process. Standard neural networks primarily reflect aleatoric effects through residual error, whereas Bayesian neural networks represent epistemic uncertainty through distributions over parameters. In regression settings, this representation yields predictive distributions whose dispersion increases in regions that are weakly supported by data.

Let the dataset be $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$ and let w denote the collection of all network parameters. Bayesian inference targets the posterior $p(w \mid \mathcal{D})$ given by Bayes rule

$$p(w \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid w)\, p(w)}{p(\mathcal{D})}, \tag{4.1}$$

where $p(w)$ is a prior distribution over parameters, $p(\mathcal{D} \mid w)$ is the likelihood, and the evidence is $p(\mathcal{D}) = \int p(\mathcal{D} \mid w)\, p(w)\, dw$. For neural networks, the evidence integral is intractable in general. A classical result links Bayesian neural networks to Gaussian Process models. Neal [1996] showed that, under suitable priors, a single hidden layer Bayesian neural network converges to a Gaussian process in the infinite width limit, and subsequent work has extended and clarified this correspondence for deeper architectures and practical parameterizations. Gaussian process models provide exact Bayesian inference for certain kernels and are effective in small data regimes, but their computational cost scales poorly with dataset size and their performance depends on kernel choice. Bayesian neural networks support stochastic optimization and can be embedded in deep architectures that learn hierarchical feature representations, which makes them practical for high dimensional inputs.

Approximate inference is therefore required. Variational inference introduces an approximate posterior family $q(w; \phi)$ parameterized by variational parameters ϕ and fits q by minimizing the Kullback Leibler divergence $\mathrm{KL}(q(w; \phi) \,\|\, p(w \mid \mathcal{D}))$. This objective is equivalent to maximizing the evidence lower bound

$$\mathrm{ELBO} = \mathbb{E}_{q(w;\phi)}[\log p(\mathcal{D} \mid w)] - \mathrm{KL}(q(w; \phi) \,\|\, p(w))\,, \tag{4.2}$$

which lower bounds $\log p(\mathcal{D})$. The first term promotes data fit through the expected log likelihood. The second term penalizes departures of the approximate posterior from the prior and acts as a complexity control. Blundell et al. [2015] method of "Bayes by Backprop" optimizes this objective using stochastic gradient methods.

A mean field Gaussian posterior is assumed for each scalar parameter. For weight w_{ij},

$$q(w_{ij}; \phi_{ij}) = \mathcal{N}\left(w_{ij}; \mu_{ij}, \sigma_{ij}^2\right), \tag{4.3}$$

with trainable $\phi_{ij} = \{\mu_{ij}, \sigma_{ij}\}$. In Bayesian linear layers, the weights and biases are sampled from these variational distributions rather than treated as fixed tensors. Let $\epsilon_{ij} \sim \mathcal{N}(0, 1)$ and write

$$w_{ij} = \mu_{ij} + \sigma_{ij}\epsilon_{ij}. \tag{4.4}$$

This yields a differentiable mapping from (μ, σ) to sampled parameters and enables backpropagation through stochastic layers. At inference time, the predictive distribution for a new input x^* is

$$p(y^* \mid x^*, \mathcal{D}) = \int p(y^* \mid x^*, w)\, q(w; \phi)\, dw,$$

which is commonly approximated using Monte Carlo sampling by drawing multiple parameter samples $w^{(s)} \sim q(w; \phi)$ and aggregating the resulting predictions into a mean and dispersion estimate.

4.1.1 Bayesian Neural Network Background

Training Bayesian neural networks is more computationally intensive than training deterministic neural networks because the number of learned parameters increases and forward passes require sampling. Progress in stochastic variational inference and Bayesian deep learning software has enabled practical training on modern hardware. Alternative uncertainty estimation approaches include Monte Carlo dropout, which uses dropout at inference to approximate Bayesian model averaging,

and deep ensembles, which quantify uncertainty through diversity across independently trained deterministic models [Gal and Ghahramani, 2015]. These methods present different computational costs and calibration characteristics. Amini et al. [2020] emphasized that approximate Bayesian inference in deep networks remains sensitive to prior selection and optimization settings and that posterior integration is computationally demanding due to sampling requirements at inference.

For regression, Bayesian neural networks yield predictive distributions rather than point estimates. The posterior over parameters induces a distribution over outputs, which permits credible intervals and uncertainty aware decision thresholds. Blundell et al. [2015] reported competitive predictive performance while producing meaningful uncertainty estimates. Predictive variance is expected to increase in regions of the input space that are weakly constrained by training observations, which is important near operational limits where extrapolation risk is elevated. In maritime contexts, such uncertainty quantification is relevant because data is costly and operational conditions can change substantially across sea states.

Maritime autonomy and uncertainty at limit states. In maritime and ocean engineering, uncertainty quantification is important because failures can be of high consequence and observational coverage can be limited. Yuan et al. [2020] applied a Bayesian neural network to predict ship fuel consumption under operational changes and used the predictive distribution to quantify uncertainty in estimated mitigation effects. Their formulation accounted for uncertainty from model parameters and measurement processes and used Monte Carlo sampling to compute predictive intervals. Their comparison against a deterministic neural network and a Gaussian Process model showed that the Bayesian neural network and Gaussian Process produced calibrated predictive intervals, whereas the deterministic model did not provide interval estimates.

Bayesian layers have also been incorporated into sequence models for maritime perception and classification. Kong et al. [2022] proposed a Bayesian Transformer neural network for ship type identification from Automatic Identification System (AIS) trajectories by replacing the final classifier with a Bayesian fully connected layer. This modification enabled probability estimates

with uncertainty information under noise and ambiguity, which supports conservative decision policies when classification confidence is low.

Bayesian neural networks provide a probabilistic framework for uncertainty quantification in deep learning. The formulation places priors on network parameters and updates these priors to posteriors conditioned on data using variational objectives such as the evidence lower bound. In regression tasks, this approach produces predictive distributions whose dispersion reflects epistemic uncertainty and observation noise. These properties motivate the use of Bayesian neural networks in settings that require uncertainty aware predictions under distribution shift, including maritime autonomy.

4.1.2 BNN Model Setup and Training

The Bayesian regression model is implemented in PyTorch using the `torchbnn` package, which provides Bayesian analogues of standard layers (e.g. `BayesLinear`) and a utility loss for computing the Kullback–Leibler divergence over all Bayesian layers [Lee et al., 2022]. The training loop is wrapped inside a PyTorch Lightning module (`BayesianLightning`) that handles optimization, early stopping and checkpointing. This subsection details the model architecture, the loss used for variational training, the role of key hyperparameters, and the mechanism used to update priors for transfer learning.

Model architecture and Bayesian linear layers: The base BNN used in the experiments is a fully connected feedforward network constructed from Bayesian linear layers provided by `torchbnn`. In the `torchbnn_simple` architecture, the mapping is

$$\mathbb{R}^{d_{\text{in}}} \xrightarrow{\text{BayesLinear+ReLU}} \mathbb{R}^{32} \xrightarrow{\text{BayesLinear+ReLU}} \mathbb{R}^{32} \xrightarrow{\text{BayesLinear+ReLU}} \mathbb{R}^{32} \xrightarrow{\text{BayesLinear}} \mathbb{R}^{d_{\text{out}}}.$$

Each fully connected layer is implemented as `bnn.BayesLinear(prior_mu, prior_sigma, in_features, out_features)`, where `prior_mu` and `prior_sigma` specify the mean and stan-

dard deviation of a zero mean Gaussian prior on the weights and biases. A rectified linear unit is applied after the first three Bayesian linear layers and the output layer is left linear to produce the predictive mean for the regression target.

All Bayesian layers are mean field: for each scalar weight w_{ij} in a BayesLinear module, the layer maintains a variational posterior $q(w_{ij}) = \mathcal{N}(\mu_{ij}, \sigma_{ij}^2)$ with trainable μ_{ij} and $\log \sigma_{ij}$. In addition to the simple architecture, a more flexible `torchbnn` class is provided that allows an arbitrary list of hidden layer widths and activation functions; it also uses BayesLinear layers and optional dropout between hidden layers. The main experiments reported here use the `torchbnn_simple` configuration, which is sufficient to capture the low dimensional mapping between the selected input features and the scalar output (RMS pitch).

Training objective and use of `BKLLoss`: Training follows standard stochastic variational inference. For a mini batch $\{(x_n, y_n)\}_{n=1}^{B}$, one forward pass samples a set of weights $w^{(n)} \sim q(w)$ and produces predictions $\hat{y}_n = f(x_n; w^{(n)})$. The loss used in the Lightning training step is

$$\mathcal{L} = \underbrace{\frac{1}{B}\sum_{n=1}^{B} \|\hat{y}_n - y_n\|_2^2}_{\text{MSE data term}} + \lambda_{\text{KL}} \underbrace{\text{BKLLoss(model)}}_{\text{KL complexity term}}, \tag{4.5}$$

where λ_{KL} is the KL loss rate (denoted `kl_weight` in the implementation). The data term is a mean squared error (MSE) on normalized outputs. The KL complexity term is computed using `torchbnn.modules.loss.BKLLoss` with `reduction='mean'`, which sums the KL divergence between the variational posterior and the prior for all BayesLinear layers and divides by the number of parameters. With this configuration the training loss in Eq. (4.5) is proportional to a Monte Carlo estimate of the negative ELBO introduced in Tools Section 2.7, with the KL term rescaled by constant λ_{KL}.

Optimization uses the Adam optimizer with a fixed learning rate `lr` = 0.02. Early stopping monitors the validation loss $\mathcal{L}_{\text{val}}$ in Eq. (4.5) with patience of 15 epochs, and the best model (lowest validation loss) is saved via a checkpoint callback. The training loop is implemented with

PyTorch Lightning's `Trainer`, with a maximum of 500 epochs. In practice, early stopping usually terminates training well before this limit. During testing, the Lightning module draws 250 Monte Carlo samples of the predictive distribution for each batch, aggregates them to a predictive mean and standard deviation in physical units (using the stored normalization constants), and passes these to a plotting utility for visualization and logging. This procedure realizes the Monte Carlo approximation to the predictive distribution described earlier.

Hyperparameters and KL loss rate: The BNN introduces two hyperparameters that are specific to the Bayesian formulation in addition to the usual learning rate and architecture choices. The first is the KL loss rate $\lambda_{\mathrm{KL}} = 0.3$, which multiplies the BKLLoss term in Eq. (4.5). The second is the prior standard deviation σ_{prior}, passed as `prior_sigma` when constructing each BayesLinear layer; in the experiments reported here, $\sigma_{\mathrm{prior}} = 0.4$.

The KL constant λ_{KL} controls the trade off between fitting the data and staying close to the prior. In the idealized ELBO, the relative weighting between the data term and the KL term is determined by the dataset size and the choice of mini batch approximation. In practice many implementations introduce an explicit scalar λ_{KL} as in Eq. (4.5) to tune this trade off. If λ_{KL} is set very small, the BNN behaves similarly to a deterministic network trained by MSE: the optimizer focuses on minimizing the data term, the variational posterior can drift far from its prior, and the resulting weight uncertainty is often underestimated, which reduces epistemic uncertainty in extrapolation regions. If λ_{KL} is set too large, the optimizer is dominated by the KL term and strongly penalizes any deviation from the prior, which can cause underfitting and, in extreme cases, a form of posterior collapse where the variational posterior remains close to the prior despite the data. Similar behavior of the KL weight has been studied in variational autoencoders, where constant large KL weights can force the approximate posterior towards an uninformative prior and impede learning, while annealing the KL weight over training often improves performance [Takida et al., 2022]. The choice $\lambda_{\mathrm{KL}} = 0.3$ gives a moderate regularization strength that maintains a meaningful influence of the prior while allowing the network to adapt to the data.

Effect of the prior standard deviation σ_{prior}: In `torchbnn`, each BayesLinear layer is constructed with a Gaussian prior

$$p(w_{ij}) = \mathcal{N}(0, \sigma^2_{\text{prior}}),$$

where σ_{prior} is a global hyperparameter shared across all weights in that layer. For zero mean, isotropic Gaussian priors, the negative log prior contributes an ℓ_2 penalty of the form $\frac{1}{2\sigma^2_{\text{prior}}}\|w\|_2^2$ in the loss. The corresponding weight decay coefficient is $\lambda_{\text{wd}} = 1/(2\sigma^2_{\text{prior}})$ [MacKay, 1992]. This means that decreasing σ_{prior} increases the strength of the implicit weight decay and enforces stronger shrinkage of the weights towards zero, whereas increasing σ_{prior} weakens this regularization and allows weights to move further away from zero. Theoretical analyses of Gaussian priors in BNNs have shown that they induce well understood regularization effects and shape the distribution of hidden unit activations [Fortuin, 2022].

The prior standard deviation also appears directly in the KL divergence between the variational posterior and the prior. For a single weight with posterior $q(w) = \mathcal{N}(\mu_q, \sigma_q^2)$ and prior $p(w) = \mathcal{N}(0, \sigma^2_{\text{prior}})$, the KL divergence has the closed form

$$\text{KL}\big(q(w) \,\|\, p(w)\big) = \log \frac{\sigma_{\text{prior}}}{\sigma_q} + \frac{\sigma_q^2 + \mu_q^2}{2\sigma^2_{\text{prior}}} - \frac{1}{2}.$$

For fixed μ_q and σ_q, reducing σ_{prior} increases the second term and hence magnifies the KL penalty, which biases training towards small posterior variances and weights tightly concentrated around zero or around any updated prior mean. Conversely, increasing σ_{prior} lowers the penalty for large μ_q and σ_q and yields a broader, less informative prior, with correspondingly weaker regularization. Empirical studies have confirmed that choosing σ_{prior} too small can lead to underfitting and overly conservative predictive distributions, while overly large σ_{prior} can give overconfident predictions and poor calibration if the KL term becomes too weak [Fortuin, 2022]. In the present implementation, a value of $\sigma_{\text{prior}} = 0.4$ is used to balance these effects and provide a reasonably informative yet flexible prior.

Normalization and data handling Inputs and outputs are normalized based on statistics computed from the training datasets. A custom `LoadData` class returns data loaders together with a dictionary of normalization constants containing the mean and standard deviation of the output variable. The Lightning module stores these constants and uses them to de normalize both predictions and targets when computing test time summaries and visualizations [Falcon and The PyTorch Lightning team, 2019]. This ensures that the BNN is trained on numerically well scaled targets while evaluation and uncertainty quantification are performed in physically meaningful units (e.g. degrees of RMS pitch).

Updating priors for transfer learning A key feature of the BNN architecture is an explicit mechanism to update priors after training on a source task, so that the learned posterior over weights can serve as an informative prior for downstream tasks. Both the `torchbnn_simple` and `torchbnn` classes implement an `update_priors()` method that iterates over all BayesLinear layers and copies the current variational parameters into the corresponding prior parameters:

- for each layer, the prior mean `prior_mu` is set to the learned weight mean `weight_mu`
- the prior log standard deviation `prior_log_sigma` is set to the learned `weight_log_sigma`
- analogous updates are performed for the bias parameters when they are present

The method also ensures that the updated prior tensors reside on the correct device and updates an internal tracking structure `current_priors` for inspection and debugging.

Conceptually, calling `update_priors()` after training on a source dataset implements the Bayesian principle that "today's posterior is tomorrow's prior". When the network is subsequently fine tuned on a new but related dataset, such as data from a different operating condition of the same vessel, the KL term in Eq.4.5 now penalizes deviations from the informative prior centered at the previously learned weights rather than from a diffuse zero mean prior. This makes the BNN perform a genuine sequential Bayesian update when moving from source to target task, as opposed to simply reusing point estimates of the weights. In the transfer learning literature, several authors

have shown that using the posterior from a pre training task as a prior for downstream tasks can significantly improve data efficiency and uncertainty calibration compared to re-initializing with an uninformative prior [Fortuin, 2022].

In this implementation, an initial BNN is trained on a larger source dataset and saved as a Lightning checkpoint that contains both the variational parameters and the prior parameters. This base model is then loaded, deep copied and used as the starting point for downstream training runs with different numbers of shots, ensuring that all such runs share the same informative prior and normalization constants. The combination of an explicit prior update and a KL regularizer anchored at this updated prior allows the BNN to retain information from the source task while adapting to new data.

4.1.3 BNN Results

After training with 3000 shots of small wave data, the BNN accurately predicts motions in the small-wave training domain, shown in the plots of Figure 4.3 by the small RMS values in the lower left of each subplot. As expected, uncertainty grows in predictions outside of the training domain in extreme weather and high motion RMS events. However, the uncertainty bands in the extreme wave domain are not wide enough to account for the true values, shown by the shaded region not including the diagonal ground-truth line.

Within the training procedure, Bayesian Neural Network's have hyperparameters that can be tuned to attempt to increase the uncertainty bounds. This is done through increasing σ_{prior} and λ_{KL}, which is described in the previous Section 4.1.2. However, through experimentation, if these hyperparameters were increased further, accuracy severely dropped, eventually stopping learning from occurring (Figure 4.2). This is because increasing the λ_{KL} results in the optimizer penalizing any deviation from the prior, which results in underfitting. If σ_{prior} is increased from 0.4 to 0.5, as shown in the lowest subplot of Figure 4.2, learning does not occur and predictions become constant. This is due to the functional prior becoming larger, where weight samples produce highly variable functions.

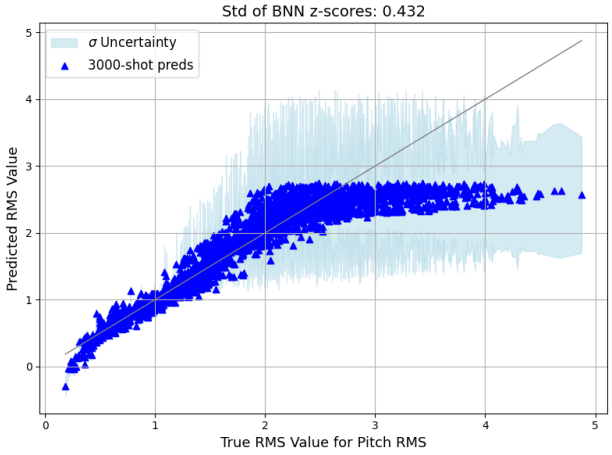
When σ_{prior} was increased from its baseline value of 0.4 to larger values (e.g. 0.45–0.5), training also collapsed. The network converged to an almost constant predictor with little dependence on the inputs, as illustrated in Figure 4.2. At first sight this seems counterintuitive, since for a single weight the KL expression

$$\mathrm{KL}\big(q(w) \,\|\, p(w)\big) = \log \frac{\sigma_{\text{prior}}}{\sigma_q} + \frac{\sigma_q^2 + \mu_q^2}{2\sigma_{\text{prior}}^2} - \tfrac{1}{2}$$

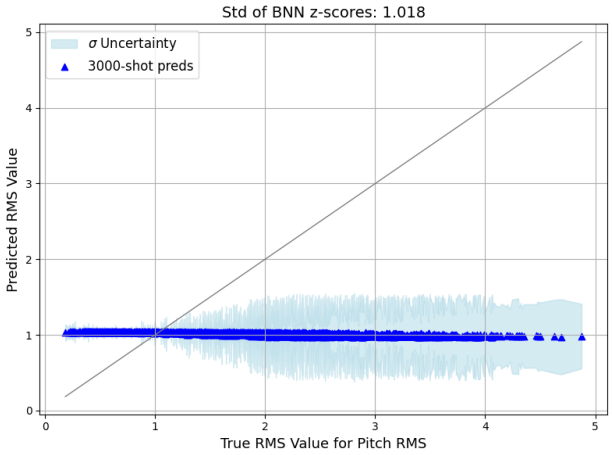
shows that increasing σ_{prior} reduces the KL penalty when (μ_q, σ_q) are held fixed, implying weaker parameter-level shrinkage. In a deep BNN, however, σ_{prior} also controls the *functional prior*, that is, the distribution of functions obtained by sampling all weights from the prior. For ReLU networks this functional prior becomes dramatically more variable with depth; even modest increases in σ_{prior} can therefore induce very rough, high-amplitude random functions. Studies of Gaussian weight priors in deep BNNs have shown that such choices can lead to heavy-tailed unit activations and unintuitive functional behavior if not carefully tuned [Fortuin, 2022].

With a fixed KL weight $\lambda_{\text{KL}} = 0.3$, the optimizer must trade off fitting the data (MSE term) against staying close to this high-variance functional prior (KL term). In the present experiments the easiest way to reduce the combined loss is to drive most input weights towards values that effectively ignore the inputs and to use mainly the biases in the final layer, yielding a nearly constant output across the input space. This solution keeps the posterior relatively close to the prior, while achieving a reasonable average MSE by predicting something close to the marginal mean of the targets, but it completely sacrifices the dependency on wave and vessel states. The resulting behavior in Figure 4.2 is therefore interpreted as a form of posterior collapse or over-pruning induced by an interaction between a too-broad Gaussian prior and a non-negligible KL weight in a deep architecture, rather than as classical overfitting. This illustrates that both excessively small and excessively large σ_{prior} can be detrimental, albeit through different mechanisms, and motivates the choice of a moderate value $\sigma_{\text{prior}} = 0.3$ in the main experiments.

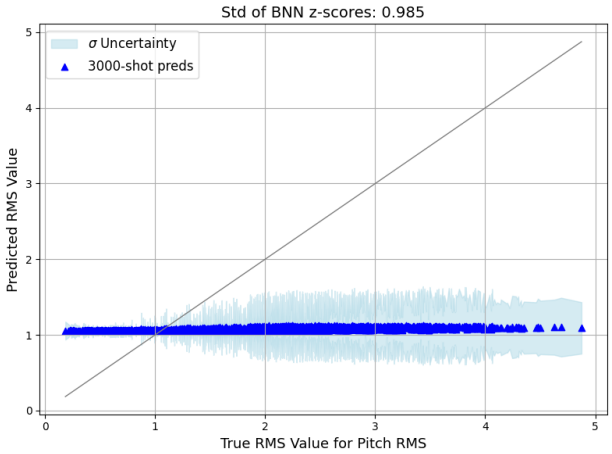
The Bayesian model establishes a reference for probabilistic prediction and highlights where



(a) Increased KL loss rate to 0.5.



(b) Increased KL loss rate to 0.6.



(c) Increased σ to 0.5.

Figure 4.2: Increasing the λ_{KL} or σ_{prior} further, results in drastic degradation of performance.

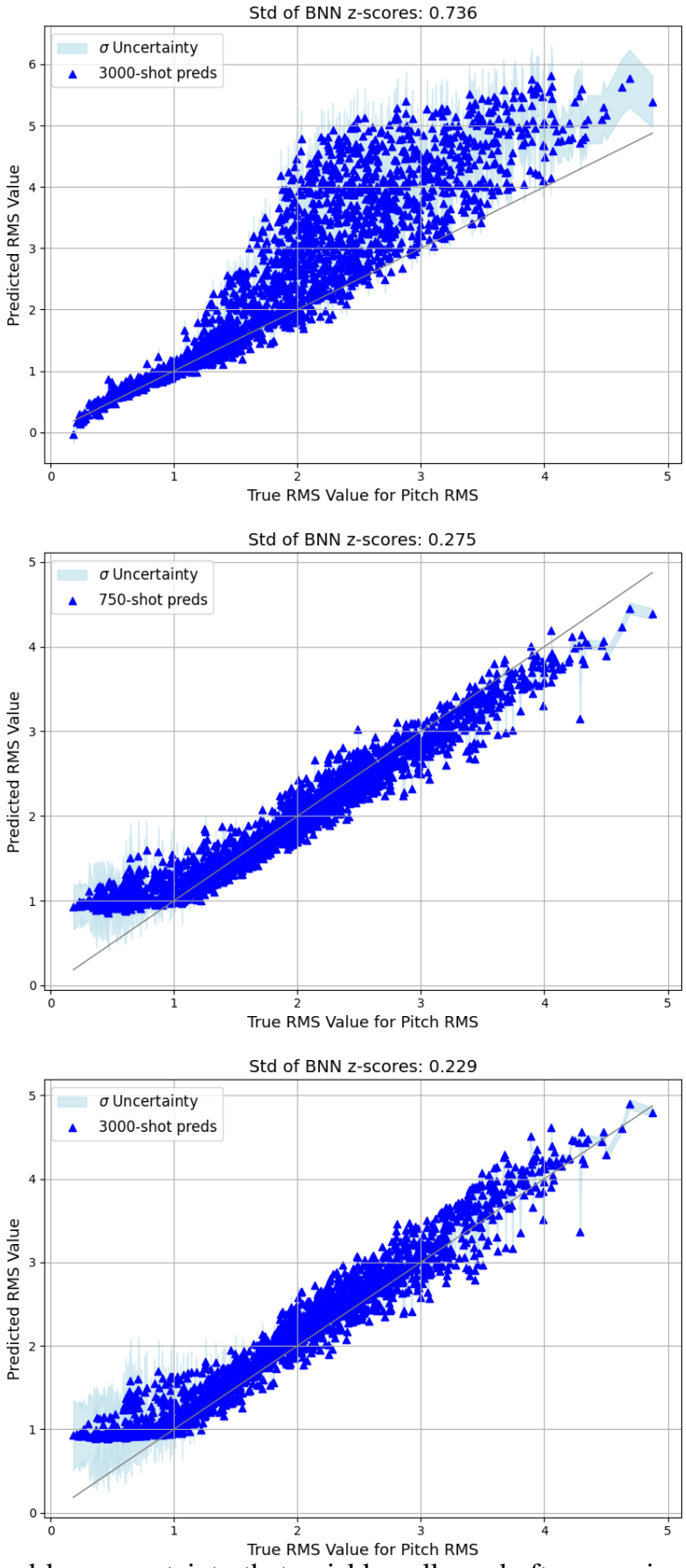


Figure 4.3: BNNs had low uncertainty that quickly collapsed after experiencing extreme waves. Forgetfullness is a serious concern.

uncertainty estimates remain informative or become unreliable under the tested adaptation regimes. These experiments indicate that the variational Bayesian neural network is highly sensitive to the interaction between the prior specification and the KL weighting, particularly under staged fine tuning intended to mimic deployment adaptation. In the present setting, predictive uncertainty tends to collapse after exposure to extreme sea states and the model exhibits pronounced catastrophic forgetting under sequential updates, reducing reliability precisely in the regimes where conservative behavior is required. These limitations motivate evaluation of alternative probabilistic regressors that provide closed form predictive distributions and improved stability for uncertainty aware limit state prediction.

4.2 Deep Evidential Regression (DER)

Deep Evidential Regression (DER), developed by Amini et al. [2020], augments a deterministic neural network with a higher order probabilistic head that outputs the parameters of a conjugate prior over the Gaussian likelihood of the target. Instead of sampling network weights as in Bayesian neural networks, DER places a Normal–Inverse Gamma (NIG) prior directly over the unknown mean and variance of the observation model and trains the network to predict the NIG hyperparameters for each input. This formulation yields a Student-t posterior predictive distribution in closed form and enables simultaneous estimation of a continuous target together with decomposed aleatoric and epistemic uncertainty, without test time sampling and without the need to train on explicit out of distribution examples.

Compared with Bayesian neural networks, evidential regression offers several practical advantages for real time systems. First, inference requires only a single forward pass through a deterministic network head, in contrast to Monte Carlo sampling over weight posteriors. Second, the NIG conjugate structure gives analytic expressions for the predictive mean and uncertainty components, and thus avoids numerical integration over weight distributions. Third, uncertainty arises from the network's "evidence" about the likelihood parameters. When the network predicts

low evidence in regions with few or no training samples, the resulting posterior predictive distribution becomes heavy tailed and exhibits large variance. Amini et al. [2020] demonstrated that this mechanism yields competitive accuracy and well calibrated uncertainties on standard tabular regression benchmarks and high dimensional tasks such as monocular depth estimation, while being faster at inference than Monte Carlo dropout and deep ensembles.

Evidential approaches have been extended and analyzed in several directions. Mixture of Normal–Inverse Gamma models capture predictive distributions in multimodal regression tasks. Meinert et al. [2023] investigated theoretical properties and limitations such as evidence contraction and the effect of different priors and regularizers on uncertainty calibration. Recent surveys provide comprehensive overviews of evidential deep learning, including connections between evidential regression, Dirichlet prior networks, and classical subjective logic [Gawlikowski et al., 2022].

4.2.1 Deep Evidential Regression Background

In a standard regression setting, let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ denote the training data. A conventional Gaussian likelihood model assumes

$$y_i \sim \mathcal{N}(\mu, \sigma^2), \qquad \theta = (\mu, \sigma^2), \tag{4.6}$$

with parameters θ treated as deterministic. Maximum likelihood estimation then chooses μ and σ^2 that maximize $\prod_i p(y_i \mid \theta)$ or, equivalently, minimize the negative log likelihood. This procedure captures aleatoric uncertainty through the residual variance σ^2 but remains oblivious to epistemic uncertainty in the prediction [Gawlikowski et al., 2022].

Deep evidential regression retains the Gaussian observation model but introduces a higher order prior on θ. Specifically, the mean μ and variance σ^2 are treated as random variables with a Normal–Inverse Gamma prior

$$\mu \sim \mathcal{N}\left(\gamma, \frac{\sigma^2}{\nu}\right), \qquad \sigma^2 \sim \Gamma^{-1}(\alpha, \beta), \tag{4.7}$$

where $\gamma \in \mathbb{R}$ is a location parameter, $\nu > 0$ and $\alpha > 1$ play the role of "virtual sample sizes", and $\beta > 0$ controls the scale of the variance [Zhang et al., 2019]. The resulting conjugate prior over (μ, σ^2) is the Normal–Inverse Gamma distribution

$$p(\mu, \sigma^2 \mid \gamma, \nu, \alpha, \beta) = \frac{\beta^\alpha \sqrt{\nu}}{\Gamma(\alpha)\sqrt{2\pi\sigma^2}} \left(\frac{1}{\sigma^2}\right)^{\alpha+1} \exp\left(-\frac{2\beta + \nu(\gamma - \mu)^2}{2\sigma^2}\right), \tag{4.8}$$

which can be interpreted in terms of virtual observations. ν acts as the number of pseudo observations supporting the mean γ, and α as the number of pseudo observations supporting the variance. The total "evidence" associated with a prediction is usually defined as

$$\Phi = 2\nu + \alpha, \tag{4.9}$$

the sum of these virtual counts.

Given the NIG prior, marginalizing (μ, σ^2) yields a Student-t posterior predictive density for y with 2α degrees of freedom, location γ, and scale parameter $\beta(1 + \nu)/(\nu\alpha)$. The first moments of the NIG provide a natural decomposition of the predictive mean and uncertainties:

$$\mathbb{E}[\mu] = \gamma, \qquad \mathbb{E}[\sigma^2] = \frac{\beta}{\alpha - 1}, \qquad \mathrm{Var}[\mu] = \frac{\beta}{\nu(\alpha - 1)}, \tag{4.10}$$

for $\alpha > 1$ [Zhang et al., 2019]. These moments are used to define the point prediction as $\hat{y} = \gamma$, aleatoric variance as $\mathbb{E}[\sigma^2]$, and epistemic variance as $\mathrm{Var}[\mu]$. The epistemic term increases when the network outputs low ν or low α, corresponding to limited evidence about the likelihood parameters.

Applications in oceanography, robotics, and maritime domains: Since its introduction, deep evidential regression has been applied in several domains that are close to maritime autonomy. Chen et al. [2023] used a deep evidential regression network combined with empirical orthogonal functions to reconstruct subsurface ocean temperature profiles from satellite sea surface data at

regional and global scales. The method delivered competitive reconstruction accuracy and provided spatially resolved uncertainty estimates, enabling practitioners to distinguish regions where the reconstruction is well constrained from regions with large epistemic uncertainty. Novitasari et al. [2024] applied evidential regression to estimate autoconversion rates in clouds from satellite observations and demonstrated calibrated uncertainty estimates that reflect both observational noise and model uncertainty.

In robotics, Uhlmann et al. [2024] proposed a deep continual evidential regression model to increase the absolute position accuracy of industrial robots. Their approach used NIG heads to model pose prediction errors and showed that evidential uncertainty provided informative confidence bounds and facilitated incremental calibration across different workspaces. For perception in mobile robotics, evidential uncertainty has also been used in semantic segmentation and trajectory prediction, where NIG based heads provide position dependent uncertainty estimates without sampling [Marvi et al., 2025].

Within maritime research, evidential deep learning has so far been used primarily in classification settings. Singh et al. [2022] combined graph based clustering of AIS trajectories with evidential deep learning classifiers to detect maritime anomalies, such as unusual vessel maneuvers and AIS blackout events, and showed that evidential uncertainty improved the detection of outliers and anomalous segments. They used evidential reasoning to fuse multiple deep learning ship classifiers and quantify classification confidence for vessel type recognition in images. These studies demonstrate that evidential models are well suited to maritime environments with sparse labels, heterogeneous sensors, and safety critical decisions, even though explicit use of DER for continuous motion or load prediction in marine vessels remains relatively unexplored.

4.2.2 DER Model Setup and Training

The evidential regression model used in the experiments adopts the same base architecture as the deterministic feedforward network described in Chapter 4. A fully connected network with rectified linear unit activations maps the normalized input feature vector $x \in \mathbb{R}^{d_{\text{in}}}$ to a low dimensional latent

representation. The final layer is replaced by a four dimensional evidential head that outputs the NIG hyperparameters for each input:

$$(\gamma, \nu, \alpha, \beta) = f(x; w),$$

where w denotes all trainable weights of the network including the evidential head. A linear activation is used for $\gamma \in \mathbb{R}$. To enforce the constraints $\nu > 0$, $\alpha > 1$, and $\beta > 0$, the raw outputs corresponding to $\tilde{\nu}, \tilde{\alpha}, \tilde{\beta}$ are transformed via a softplus nonlinearity,

$$\nu = \text{softplus}(\tilde{\nu}), \quad \alpha = \text{softplus}(\tilde{\alpha}) + 1, \quad \beta = \text{softplus}(\tilde{\beta}), \tag{4.11}$$

so that all forward passes produce valid NIG parameters [Amini et al., 2020].

4.2.2.1 Problem Formulation

Given a minibatch of training data $\{(x_i, y_i)\}_{i=1}^{B}$, the network outputs a set of NIG parameters $m_i = (\gamma_i, \nu_i, \alpha_i, \beta_i) = f(x_i; w)$ for each input x_i. These parameters define an evidential distribution $p(\mu, \sigma^2 \mid m_i)$ over the Gaussian likelihood parameters for sample i. The goal of training is to find weights w such that the induced Student-t posterior predictive distribution places high probability on the observed targets y_i and the predicted evidence remains low where the regression error is large.

4.2.2.2 Training Objective

The loss function follows the formulation of Amini et al. [2020] and consists of two terms: a negative log evidence term derived from the Student-t predictive distribution and an evidence regularizer that penalizes confident but inaccurate predictions.

Type II maximum likelihood term For each sample, the marginal likelihood or model evidence is given by the Student-t density

$$p(y_i \mid m_i) = \mathrm{St}\left(y_i;\ \gamma_i,\ \frac{\beta_i(1+\nu)}{\nu\alpha},\ 2\alpha\right), \tag{4.12}$$

obtained by integrating (μ, σ^2) out of the joint distribution. Taking the negative logarithm yields the per sample negative log likelihood

$$L_i^{\mathrm{NLL}}(w) = \tfrac{1}{2}\log\left(\frac{\pi}{\nu}\right) - \alpha\log\Omega + \left(\alpha+\tfrac{1}{2}\right)\log\left(\nu(y_i-\gamma)^2+\Omega\right) + \log\frac{\Gamma(\alpha)}{\Gamma(\alpha+\frac{1}{2})}, \quad \Omega_i = 2\beta(1+\nu). \tag{4.13}$$

This term encourages the network to place high posterior predictive density on the observed targets by adjusting both the location γ_i and the evidence parameters (ν, α, β).

Evidence regularizer Without additional regularization, the network could minimize L_i^{NLL} by predicting high evidence (large ν_i, α_i) even when $|y_i - \gamma_i|$ is large, leading to overconfident but inaccurate predictions. To discourage this behavior, DER introduces an evidence penalty scaled by the prediction error and the total evidence

$$\Phi_i = 2\nu_i + \alpha_i, \qquad L_i^{\mathrm{R}}(w) = \left|y_i - \gamma_i\right| \cdot \Phi_i. \tag{4.14}$$

This regularizer forces the network to reduce evidence whenever the prediction is incorrect. Where $|y_i - \gamma_i|$ is small, large evidence is not penalized.

Total objective and hyperparameters The per sample loss combines both contributions,

$$L_i(w) \ = \ L_i^{\mathrm{NLL}}(w) \ + \ \lambda_{\mathrm{DER}}\, L_i^{\mathrm{R}}(w), \tag{4.15}$$

and the overall training loss is the average over the minibatch. The scalar coefficient $\lambda_{\mathrm{DER}} > 0$ controls the trade off between data fit and uncertainty inflation. Small values of λ_{DER} allow the

network to be overconfident, whereas very large values force the model towards high uncertainty everywhere. Amini et al. [2020] report that λ_{DER} values in the range $10^{-2} - 10^{-1}$ work well across low and high dimensional regression tasks, with the precise value tuned on a validation set.

In addition to λ_{DER}, the main hyperparameters of the DER implementation are the architecture of the base network, the choice of activation functions, the learning rate and optimizer, the batch size, and the early stopping criteria. In the present framework, these choices mirror those of the deterministic and Bayesian baselines so that differences in performance can be attributed primarily to the uncertainty modeling rather than to capacity or optimization effects.

4.2.2.3 Prediction and Uncertainty at Test Time

At test time, for an input x_* the evidential head outputs a single set of NIG parameters $m_* = (\gamma_*, \nu_*, \alpha_*, \beta_*)$. Prediction and uncertainty follow directly from the moment expressions in Eq. (4.10):

$$\hat{y}_* = \mathbb{E}[\mu \mid m_*] = \gamma_*, \tag{4.16}$$

$$\widehat{\text{Var}}_{\text{aleatoric}}(y_*) = \mathbb{E}[\sigma^2 \mid m_*] = \frac{\beta_*}{\alpha_* - 1}, \tag{4.17}$$

$$\widehat{\text{Var}}_{\text{epistemic}}(y_*) = \text{Var}[\mu \mid m_*] = \frac{\beta_*}{\nu_*(\alpha_* - 1)}. \tag{4.18}$$

4.2.2.4 Normalization and Data Handling

As in the Bayesian neural network experiments, both inputs and targets are normalized using statistics computed from the training data. A data loading utility provides PyTorch data loaders together with the mean and standard deviation of the target variable. The DER model is trained on normalized targets, while all diagnostics and plots are reported in physical units by de normalizing the predictions and uncertainty estimates. This procedure keeps the optimization numerically stable and enables direct comparison of predictive intervals with measured RMS pitch values.

4.2.2.5 Training algorithm

The training procedure for deep evidential regression can be summarized as follows.

Algorithm 2 Training a Deep Evidential Regression Model

Require: Training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$; neural network $f(x; w)$ with evidential head; regularization coefficient λ_{DER}; optimizer (e.g. Adam) with learning rate η; number of epochs T; minibatch size B.

Initialize network weights w; initialize optimizer parameters.
for epoch = $1, \ldots, T$ **do**
  for each minibatch $\{(x_i, y_i)\}_{i=1}^{B} \subset \mathcal{D}$ **do**
    Normalize inputs and targets using training statistics.
    Pass inputs through the network to obtain NIG parameters:

$$(\gamma_i, \nu_i, \alpha_i, \beta_i) = f(x_i; w), \quad i = 1, \ldots, B.$$

    Compute per sample negative log evidence L_i^{NLL} using Eq. (4.13).
    Compute per sample evidence regularizer L_i^{R} using Eq. (4.14).
    Form the minibatch loss

$$\mathcal{L} = \frac{1}{B} \sum_{i=1}^{B} \left(L_i^{\text{NLL}} + \lambda_{\text{DER}} L_i^{\text{R}} \right).$$

    Backpropagate $\nabla_w \mathcal{L}$ and update w using the chosen optimizer.
  end for
  Evaluate validation loss; apply early stopping if no improvement is observed for a specified patience.
end for

Output: Trained weights $w^\star$ defining the evidential regression model.

During inference, Algorithm 2 is reduced to a single forward pass and evaluation of the predictive moments in Eq.4.10. No sampling or ensembling is required.

4.2.3 DER Results

The subsequent subsection compares the behavior of deep evidential regression to Bayesian neural networks and deep kernel learning on the RMS pitch prediction task. The analysis focuses on predictive accuracy in the small wave training domain, extrapolation behavior in extreme sea states, the decomposition of aleatoric and epistemic uncertainty, and the degree of catastrophic forget-

ting when the model is incrementally fine tuned on data from increasingly severe environmental conditions.

The model trained on the small wave domain dataset is tightly clustered around the ground-truth samples of small wave shots, represented by the lower section of the diagonal line in the top subplot of Figure 4.4. When extrapolating its learning from the small-wave training domain to the test set that includes significant wave heights near 10.0 meters, DER's prediction distribution was centered around the ground-truth, unlike the other two algorithms.

Interestingly, for all examples of the DER model, the estimated uncertainty is completely dominated by aleatoric uncertainty, represented by the lighter uncertainty band in Figure 4.4. Future work should explore if using a sample collection time greater than 15 minutes would help decrease the aleatoric uncertainty, and if integrating sensor noise into the dataset increases epistemic uncertainty to more realistic levels.

Evidential regression provides a distinct trade space between accuracy and uncertainty inflation, offering insight into when calibration mechanisms succeed without explicit Bayesian sampling. The next section evaluates deep kernel learning, which couples learned features with a Gaussian process posterior to target structured uncertainty in function space.

4.3 Deep Kernel Learning (DKL)

Deep Kernel Learning (DKL), developed by Wilson et al. [2015], combines a deep neural network feature extractor with a Gaussian process (GP) defined in the learned feature space, thereby uniting the representation learning capabilities of deep networks with the calibrated uncertainty estimates of GPs. Wilson introduced DKL as a way to retain the nonparametric function space view of GPs while allowing the kernel to adapt through a deep, task specific embedding of the inputs. In the standard construction, a neural network $\phi(x; w)$ maps the raw input $x \in \mathbb{R}^D$ to a latent representation $h = \phi(x; w)$. Starting from any base kernel $k(x_i, x_j \mid \theta)$ with hyperparameters θ, define a deep

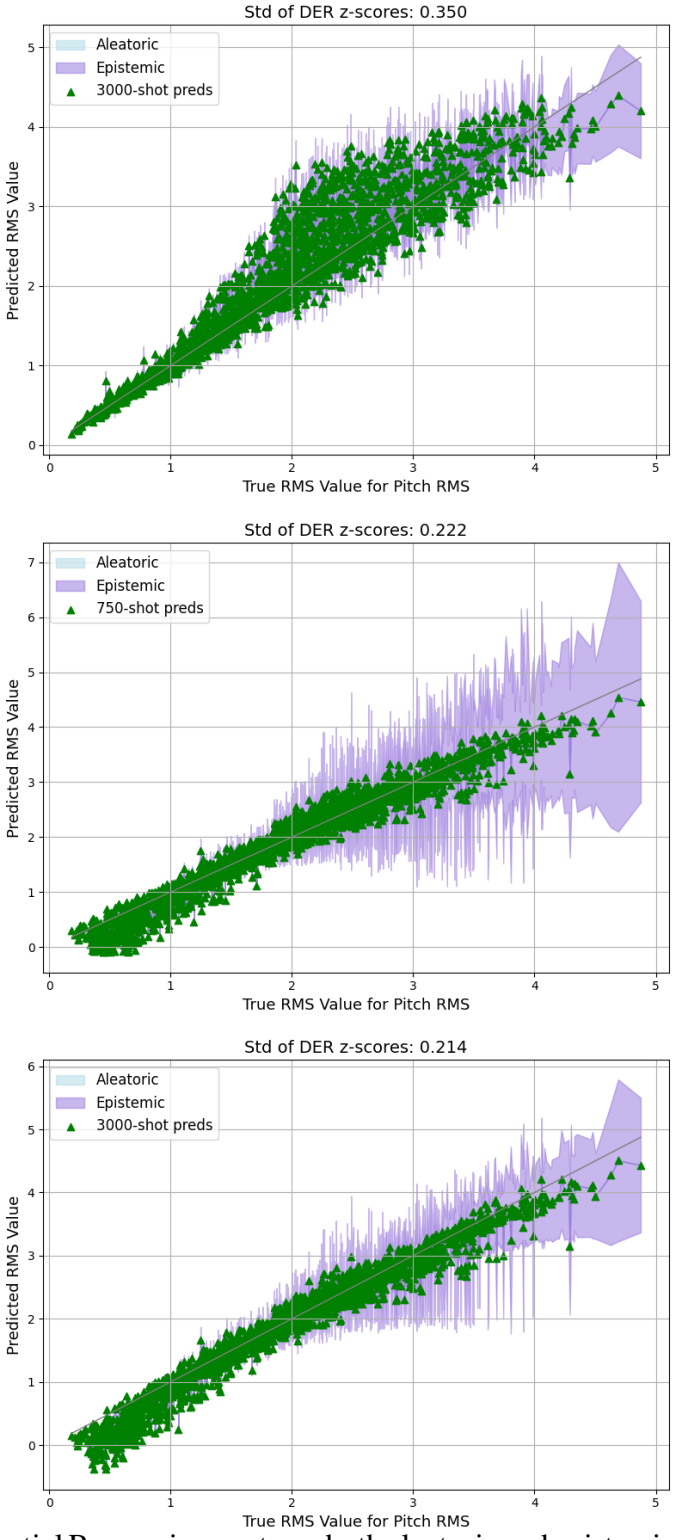


Figure 4.4: Deep Evidential Regression captures both aleatoric and epistemic sources of uncertainty. For the most part, the combined uncertainty bounds encompass the ground-truth values.

feature map $g(x, w)$ parameterized by network weights w and compose

$$k_\gamma(x_i, x_j) \;=\; k\big(g(x_i, w),\, g(x_j, w) \,\big|\, \theta\big), \qquad \gamma := \{\theta, w\}. \tag{4.19}$$

This transformation lets the GP operate in a learned representation, capturing nonstationary and hierarchical structure via g, while the base kernel controls local covariance in that space. DKL is trained end to end by maximizing the GP marginal likelihood with respect to both the network weights w and the kernel hyperparameters θ.

In the original formulation, DKL was combined with scalable KISS GP, which exploits structure in the inducing grid to reduce the cost of GP inference from $\mathcal{O}(N^3)$ to nearly linear in N while retaining exact GP semantics on the inducing grid. This framework has been shown to improve predictive performance and uncertainty calibration compared with both fixed kernel GPs and standalone deep neural networks on a wide range of regression and classification tasks, including image recognition, digit classification, and flight delay prediction [Wilson et al., 2016].

From the perspective of uncertainty quantification, DKL inherits the closed form posterior predictive distribution of GPs. GPs provide a Bayesian nonparametric prior over functions. Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ with inputs $X = \{x_i\}_{i=1}^n$, $x_i \in \mathbb{R}^D$, and scalar targets $y = (y(x_1), \ldots, y(x_n))^\top$. Assume a mean vector $\mu_i = \mu(x_i)$ and covariance matrix $(K_{X,X})_{i,j} = k_\gamma(x_i, x_j)$ with hyperparameters γ that parameterize the kernel k_γ. If $f(x)$ is a Gaussian process then the function values has a joint Gaussian distribution,

$$f(X) = [f(x_1), \ldots, f(x_n)]^\top$$

The GP prior induces a joint Gaussian distribution over function values,

$$f(X) \;\sim\; \mathcal{N}\big(\mu_X,\; K_{X,X}\big), \qquad (K_{X,X})_{ij} = k_\gamma(x_i, x_j), \qquad (\mu_X)_i = \mu(x_i). \tag{4.20}$$

With additive independent and identically distributed Gaussian observation noise $y(x)$ |

$f(x) \sim \mathcal{N}\big(f(x), \sigma^2\big)$, the posterior predictive distribution at test inputs X_* is Gaussian:

$$\mathbb{E}[f_*] = \mu_{X_*} + K_{X_*,X}\big[K_{X,X} + \sigma^2 I\big]^{-1} y, \tag{4.21}$$

$$\mathrm{cov}(f_*) = K_{X_*,X_*} - K_{X_*,X}\big[K_{X,X} + \sigma^2 I\big]^{-1} K_{X,X_*}, \tag{4.22}$$

where the covariance matrices are constructed using the deep kernel k_γ. This yields a full predictive distribution for each input, with a principled decomposition between model fit and uncertainty that is controlled by the kernel and the inducing point approximation, rather than by explicit evidence regularizers as in DER or weight priors in BNNs.

4.3.1 Deep Kernel Learning Background

Gaussian processes and base kernels. Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^n$ with $x_i \in \mathbb{R}^D$ and scalar targets y_i. A GP prior with mean function $\mu(x)$ and kernel $k_\gamma(x, x')$ defines a joint Gaussian prior over function values,

$$\mathbf{f} = [f(x_1), \ldots, f(x_n)]^\top \sim \mathcal{N}\big(\boldsymbol{\mu}, K_{X,X}\big), \qquad (K_{X,X})_{ij} = k_\gamma(x_i, x_j), \tag{4.23}$$

and, with Gaussian observation noise $y_i = f(x_i) + \varepsilon_i$, $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$, the posterior predictive distribution at test inputs X_* is given by (4.21)–(4.22). Two base kernels considered are the radial basis function (RBF)

$$k_{\mathrm{RBF}}(x, x') \;=\; \exp\!\left(-\frac{\|x - x'\|^2}{2\ell^2}\right), \tag{4.24}$$

where the lengthscale ℓ controls the rate at which correlations decay with Euclidean distance. The original DKL work also highlighted the spectral mixture (SM) kernel, which represents the kernel as a mixture of Gaussians in the spectral domain and can approximate any stationary

covariance function:

$$k_{\mathrm{SM}}(x, x' \mid \theta) \;=\; \sum_{q=1}^{Q} a_q \, \frac{|\boldsymbol{\Sigma}_q|^{1/2}}{(2\pi)^{D/2}} \, \exp\!\Big(-\tfrac{1}{2}\big\|\boldsymbol{\Sigma}_q^{1/2}(x - x')\big\|^2\Big) \, \cos\!\big(2\pi \, \langle x - x', \, \boldsymbol{\mu}_q \rangle\big), \tag{4.25}$$

with amplitudes a_q, frequency means $\boldsymbol{\mu}_q$, and positive definite bandwidth matrices $\boldsymbol{\Sigma}_q$. In DKL, such base kernels are evaluated not on the raw inputs x but on the learned features $\phi(x; w)$ as in (4.19), which induces a nonstationary and highly expressive covariance structure in the original input space.

Scalable variational Gaussian processes. Exact GP inference scales cubically with the number of training points n due to the $O(n^3)$ cost of factorizing the $n \times n$ covariance matrix. To make DKL tractable on large datasets, inducing point approximations with variational inference are widely used. Hensman et al. [2014] introduced a variational inducing point framework in which a small set of inducing inputs $Z = \{z_j\}_{j=1}^{m}$ with corresponding latent values $\mathbf{u} = f(Z)$ summarize the GP. With a Gaussian variational posterior $q(\mathbf{u}) = \mathcal{N}(\mathbf{m}, \mathbf{S})$, the evidence lower bound (ELBO) on the log marginal likelihood is

$$\log p(\mathbf{y}) \;\geq\; \sum_{i=1}^{n} \mathbb{E}_{q(f_i)}[\log p(y_i \mid f_i)] \;-\; \mathrm{KL}[q(\mathbf{u}) \,\|\, p(\mathbf{u})] \,. \tag{4.26}$$

Maximizing (4.26) jointly with respect to $\gamma = \{\theta, w\}$ yields a scalable DKL model whose training cost scales as $O(nm^2)$ with $m \ll n$. This strategy is adopted for this work.

Applications in physical and maritime adjacent domains. Beyond the original computer vision and regression benchmarks, DKL has been applied to several problems that are relevant to maritime autonomy and ocean engineering. Yu et al. [2020] used DKL to model emissions and fuel consumption of internal combustion engines, comparing against standard GPs, deep neural networks, and other surrogate models. The DKL model delivered improved predictive accuracy and well calibrated predictive intervals, demonstrating that combining deep feature learning with

GP uncertainty can outperform both purely parametric and purely kernel based approaches.

In energy systems, deep kernel variants have been used for spatio-temporal forecasting and uncertainty quantification for renewable power and green hydrogen production, where they act as flexible surrogates for expensive physics based simulations [Yu et al., 2020]. While explicit use of DKL for ship motion or structural response prediction remains limited in the open literature, the above applications illustrate that deep kernel surrogates are well suited for modeling strongly nonlinear, safety critical systems governed by complex physics, which is precisely the regime encountered in seakeeping and autonomous surface vessel operations.

Recent analyses have also emphasized that DKL's performance and uncertainty quality depend on the choice of kernel, priors, and inducing approximation. Ober et al. [2021] "promises and pitfalls" of DKL, noting for example that marginal likelihood optimization may favor overly smooth kernels in some settings and that naive choices of priors can lead to overconfident predictions. These observations motivate the careful design of priors and variational objectives in the present implementation.

4.3.2 DKL Model Setup and Training

The DKL model for RMS pitch prediction follows the same overall data pipeline as the deterministic and Bayesian baselines, but replaces the final deterministic head with a scalable GP defined in a learned feature space. Implementation uses PyTorch and GPyTorch, wrapped in a PyTorch Lightning module for training and evaluation. This subsection summarizes the feature extractor, the GP component, the variational objective, the hyperparameters, and the training algorithm.

4.3.2.1 Model architecture

The model consists of two components: a feature extractor network $\phi(x; w)$ and a variational GP defined on the learned features.

Feature extractor network. The feature extractor is implemented as a fully connected network:

$$x \in \mathbb{R}^{d_{\text{in}}} \xrightarrow{\text{Linear+ReLU}} \mathbb{R}^{64} \xrightarrow{\text{Linear+ReLU}} \mathbb{R}^{64} \xrightarrow{\text{Linear}} \mathbb{R}^{d_\phi},$$

where $d_\phi = 64$. All hidden layers use rectified linear unit activations. The input dimension d_{in} corresponds to the number of normalized environmental and vessel features used in the experiments (significant wave height, peak period, vessel speed, and trigonometric encodings of relative wave heading). The output of the feature extractor defines the latent coordinates $h = \phi(x; w)$ in which the GP operates.

Variational DKL Gaussian process The GP component is implemented by the `DKLGPModel`, which inherits from `gpytorch.models.ApproximateGP`. A mean field variational distribution over inducing variables is specified via `CholeskyVariationalDistribution` with $m = 128$ inducing points, and a `VariationalStrategy` defines their role in the approximate posterior. The inducing inputs are initialized by randomly subsampling feature vectors from the training set and are optimized jointly with the kernel hyperparameters and network weights.

The covariance is given by a scaled RBF kernel in the latent space,

$$k_\gamma(x_i, x_j) = \sigma^2_{\text{out}}\, k_{\text{RBF}}\big(\phi(x_i; w), \phi(x_j; w) \mid \ell\big)\,,$$

implemented as a `ScaleKernel` wrapped around `RBFKernel`. Hyperparameters are constrained and regularized as follows:

- Lengthscale ℓ is constrained to lie in $[10^{-3}, 2]$ and is given a Gamma$(2.0, 0.3)$ prior, with initial value $\ell_0 = 0.2$. This encourages moderately short lengthscales, promoting local sensitivity of the GP in the learned feature space.
- The output scale σ^2_{out} is constrained to be at least 0.1 and has a Gamma$(1.0, 3.0)$ prior, with initial value $\sigma^2_{\text{out},0} = 3.0$, which favors non trivial prior variances [Fortuin, 2022].

- Observation noise is modeled with a `GaussianLikelihood`

The small lengthscale and non negligible output scale priors encourage the GP to remain flexible in regions of sparse data and avoid an overly smooth, low variance prior that would underestimate uncertainty away from the inducing points. This is consistent with recent recommendations for DKL prior design in safety critical applications [Fortuin, 2022].

The GP and feature extractor are instantiated inside the `DKLRegressor` Lightning module, which also constructs a `gpytorch.mlls.VariationalELBO` object for training.

4.3.2.2 Variational objective and hyperparameters

All parameters $\gamma = \{\theta, w\}$ are learned by maximizing the log marginal likelihood $\mathcal{L}$ of the Gaussian Process. Using the chain rule,

$$\frac{\partial \mathcal{L}}{\partial \theta} = \frac{\partial \mathcal{L}}{\partial \mathbf{K}_\gamma} \frac{\partial \mathbf{K}_\gamma}{\partial \theta}, \qquad \frac{\partial \mathcal{L}}{\partial w} = \frac{\partial \mathcal{L}}{\partial \mathbf{K}_\gamma} \frac{\partial \mathbf{K}_\gamma}{\partial g(x,w)} \frac{\partial g(x,w)}{\partial w}, \tag{4.27}$$

where the implicit derivative with respect to the $n \times n$ kernel matrix is

$$\frac{\partial \mathcal{L}}{\partial \mathbf{K}_\gamma} \;=\; \tfrac{1}{2}\Big(\mathbf{K}_\gamma^{-1}\,\mathbf{y}\mathbf{y}^\top\mathbf{K}_\gamma^{-1} \;-\; \mathbf{K}_\gamma^{-1}\Big). \tag{4.28}$$

Gradients $\partial g/\partial w$ are computed by standard backpropagation through the deep network, while $\partial \mathbf{K}_\gamma/\partial \theta$ depends on the chosen base kernel. Equations 4.27–4.28 are the core ingredients for joint end-to-end training of the deep kernel.

Training uses the variational ELBO in (4.26). In the implementation, the negative ELBO is obtained via a GPyTorch `VariationalELBO` marginal log likelihood with the total number of training points n passed as `num_data`. For a given minibatch $\{(x_i, y_i)\}_{i=1}^{B}$, the Lightning training

Table 4.1: DKL Hyperparameters

Hyperparameter	Notation	Value
Number of Layers	N/A	2
Number of Neurons	N/A	64
Number of Inducing Points	m	128
Initial Lengthscale	ℓ_0	0.2
Initial Output Scale	σ^2_{out}	3.0
Optimizer	N/A	Adam
Learning Rate	η	10^{-3}
Batch Size	B	30
Max Number of Epochs	N/A	200
Early Stopping Condition	N/A	20 Epochs

step computes

$$q(\mathbf{f}_B) = p\big(\mathbf{f}_B \mid \phi(x_1; w), \ldots, \phi(x_B; w), \mathbf{u}\big), \tag{4.29}$$

$$\mathcal{L}_{\text{ELBO}} = \sum_{i=1}^{B} \mathbb{E}_{q(f_i)}[\log p(y_i \mid f_i)] - \frac{B}{n} \operatorname{KL}[q(\mathbf{u}) \,\|\, p(\mathbf{u})]\,, \tag{4.30}$$

$$\mathcal{L}_{\text{train}} = -\mathcal{L}_{\text{ELBO}}, \tag{4.31}$$

where $p(y_i \mid f_i)$ is Gaussian with variance given by the likelihood. In code, the loss is evaluated as

$$\mathcal{L}_{\text{train}} = -\texttt{mll}\big(\texttt{output}, y\big),$$

with `output` the GP latent function distribution on the batch.

Optimization uses the Adam optimizer with learning rate $\texttt{lr} = 10^{-3}$, applied jointly to the feature extractor parameters, kernel hyperparameters, inducing locations, and likelihood parameters. Model checkpoints store the best performing parameters according to the validation ELBO.

The main hyperparameters specific to DKL in this implementation are displayed in Table 4.1:

These values were chosen to yield a GP that is expressive enough to capture the nonlinear mapping from environmental conditions to RMS pitch, while remaining computationally tractable

for repeated fine tuning in the transfer learning experiments.

4.3.2.3 Prediction and uncertainty at test time

At test time, the `DKLRegressor` uses GPyTorch's `fast_pred_var` setting to obtain predictive means and variances efficiently from the variational posterior. For a batch of test inputs X_*, the model:

1. computes features $H_* = \phi(X_*; w)$,
2. evaluates the GP latent predictive distribution $q(\mathbf{f}_*)$ under the variational posterior,
3. passes this through the Gaussian likelihood to obtain $p(\mathbf{y}_* \mid X_*)$,
4. extracts the predictive mean $\hat{\mathbf{y}}_*$ and standard deviation $\hat{\boldsymbol{\sigma}}_*$.

Both $\hat{\mathbf{y}}_*$ and $\hat{\boldsymbol{\sigma}}_*$ are then de normalized using the stored output mean and standard deviation to yield predictions and uncertainties in physical units (degrees of RMS pitch).

4.3.2.4 Normalization and data handling

The DKL experiments reuse the same data loading and normalization strategy as the deterministic and Bayesian baselines. A custom `LoadData` utility provides PyTorch data loaders along with a dictionary of normalization constants for the target variable (mean and standard deviation).

All training and validation losses are computed on normalized targets to maintain numerical stability, while final plots and metrics in this chapter are reported in physical units by denormalizing the predictions and uncertainty estimates.

4.3.2.5 Training algorithm

The overall training procedure for the DKL regression model is summarized in Algorithm 3. This closely parallels the DER training algorithm but replaces the evidential head and loss with a variational GP and ELBO objective.

Algorithm 3 Training a Deep Kernel Learning Regression Model

Require: Training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$; feature extractor $\phi(x; w)$; variational GP with inducing inputs Z and variational parameters $(\mathbf{m}, \mathbf{S})$; Gaussian likelihood parameters; learning rate η; batch size B; maximum epochs $T_{\max}$.

Initialize network weights w, inducing inputs Z (e.g. by subsampling training features), variational parameters $(\mathbf{m}, \mathbf{S})$, and likelihood noise variance.
Compute normalization constants for inputs and targets; store target mean and standard deviation.
Construct a `VariationalELBO` marginal log likelihood with `num_data` $= n$.
for epoch $= 1, \ldots, T_{\max}$ **do**
    for each minibatch $\{(x_i, y_i)\}_{i=1}^{B} \subset \mathcal{D}$ **do**
        Normalize inputs and targets using training statistics.
        Compute features $h_i = \phi(x_i; w)$ for $i = 1, \ldots, B$.
        Evaluate the GP latent predictive distribution $q(\mathbf{f}_B)$ at $\{h_i\}_{i=1}^{B}$ under the current variational posterior.
        Pass $q(\mathbf{f}_B)$ through the Gaussian likelihood to obtain $p(\mathbf{y}_B \mid \mathbf{f}_B)$.
        Evaluate the minibatch ELBO $\mathcal{L}_{\mathrm{ELBO}}$ as in (4.26) and set $\mathcal{L}_{\mathrm{train}} = -\mathcal{L}_{\mathrm{ELBO}}$.
        Backpropagate $\nabla \mathcal{L}_{\mathrm{train}}$ through the GP and feature extractor and update w, Z, $(\mathbf{m}, \mathbf{S})$, and likelihood parameters using Adam.
    end for
    Evaluate validation ELBO; update best model checkpoint and apply early stopping if no improvement is observed for a prescribed patience.
end for

Output: Trained DKL model with parameters $w^{\star}$, inducing inputs $Z^{\star}$, and variational and likelihood parameters defining the approximate GP posterior.

4.3.3 DKL Results

Figure 4.5 shows DKL predictions and associated uncertainty bands at three stage. The first subplot is after training on 3000 small wave "routine weather" shots, and following subplots showing fine tuning on increasingly many extreme wave samples. The plotting layout mirrors that used for the BNN and DER experiments, enabling a direct qualitative comparison of predictive accuracy and interval width across the three approaches.

In the small wave training regime, the DKL model reproduces the observed RMS pitch with low predictive variance. As inputs move towards harsher sea states that are under represented in the training set, the GP posterior variance naturally increases, leading to wider intervals than in the training domain. This behavior is characteristic of GP based models and does not require explicit tuning of hyperparameters such as a KL weight or an evidence regularizer (as in DER). Instead, uncertainty is governed by the deep kernel, inducing approximation, and likelihood noise.

Relative to the Bayesian neural network, DKL offers a complementary trade off. Both methods optimize an approximate ELBO and introduce priors at different levels (weight priors for BNNs, kernel and inducing priors for DKL), but BNNs require Monte Carlo sampling at test time whereas DKL produces predictive means and variances in a single forward pass through the feature extractor and GP layer. In practice, the BNN in this work required careful tuning of the prior standard deviation and KL loss rate to avoid either underestimating or collapsing predictive uncertainty, whereas DKL uses a fixed variational objective whose regularization is controlled implicitly by the inducing point prior and kernel priors. This matches observations in the literature that DKL often provides well calibrated uncertainties with fewer explicit hyperparameters, provided the kernel and inducing scheme are chosen appropriately.

Compared with deep evidential regression, DKL differs in how epistemic uncertainty is represented. DER parameterizes uncertainty via a Normal Inverse Gamma prior over the observation model parameters and relies on an evidence regularizer to suppress overconfident errors, but it does not explicitly model a posterior over functions. DKL, in contrast, models a full posterior GP in the learned feature space, so epistemic uncertainty manifests directly as increased posterior variance in

regions far from inducing points or in directions where the data do not constrain the kernel strongly. As a result, DKL and DER may produce predictive intervals of similar width in sparse regions, but the underlying mechanisms are distinct: DER modulates "evidence" in parameter space, whereas DKL modulates covariance in function space.

Overall, the DKL implementation described in this section provides a third, complementary approach to uncertainty quantification for limit state prediction alongside BNNs and DER. Its GP based predictive variance offers calibrated uncertainty near and beyond the training distribution, while the deep feature extractor allows adaptation to complex, nonlinear relationships between environmental conditions and vessel motion.

4.4 Comparison of Models

Each of the evaluated models has pros and cons. However, of the three models tested, the BNN was the least promising for both transfer learning and rapid adaptation of models to extreme and potentially hazardous events. Through each stage of the training process, the BNN's Z-score representing accuracy against ground truth was higher than the other two algorithms. In addition, the BNN suffers the most from catastrophic forgetting. It is worth noting this can likely be remedied by the process of injecting samples that represent the entire training distribution as demonstrated in Section 3.7. Lastly, the uncertainty bands are too small for each stage of the training process. When looking at the top right region of Figure 4.3, the uncertainty bounds in the large wave domain do not extend to the ground truth line. Once training in the extreme wave domain begins, the uncertainty bounds collapse, and further tuning of the hyperparameters results in degraded accuracy.

Of the three models examined, Deep Evidential Regression showed the greatest promise. At each point in the training process, DER outperformed its competitors (Table 4.2). Additionally, while the BNN and DKL base models have a tendency to over-predict when extrapolating to the extreme weather domain, DER's distribution of predictions is evenly split above and below the ground truth. Additionally, DER's uncertainty bounds capture a healthy amount of uncertainty in

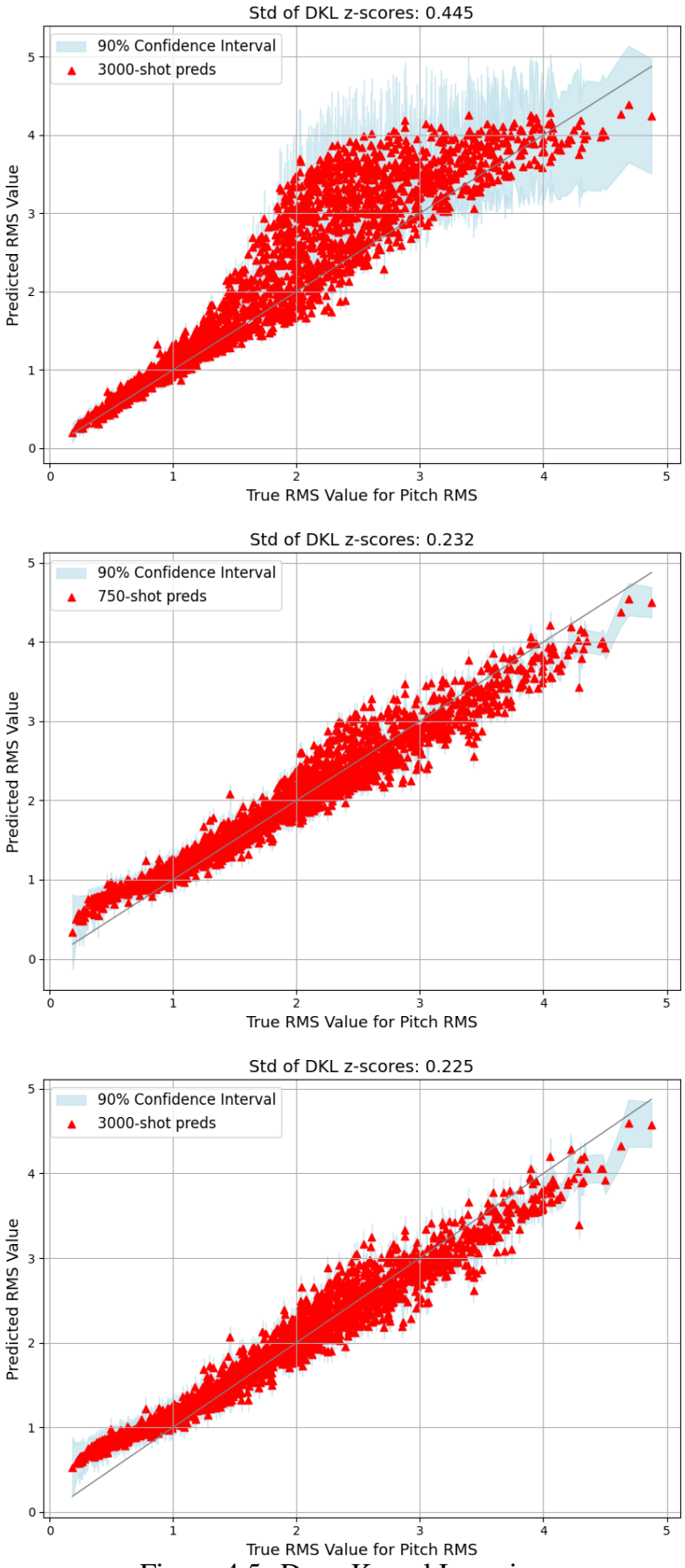


Figure 4.5: Deep Kernel Learning

Table 4.2: Accuracy comparison of pitch for uncertainty based machine learning models

Algorithm	Baseline Z-Score	750 shot Z-Score	3000 shot Z-Score
Bayesian Neural Network	0.736	0.275	0.229
Deep Evidential Regression	0.350	0.222	0.214
Deep Kernel Learning	0.445	0.232	0.225

all phases of training. It is the only model that has its uncertainty account for the ground-truth values after being trained with 3000 shots of extreme wave data, shown in the bottom subplot of Figure 4.4 with the uncertainty bounds covering both sides of the ground-truth line. It is important to reiterate that the 3000 extreme wave training samples are concentrated near the 3.5 m significant wave height threshold used to define the extreme regime, with comparatively few samples in the far tail approaching the largest sea states in the test domain (up to 10.0 m). Lastly, it also has the innate feature of being able to predict both aleatoric and epistemic uncertainty.

The main drawback of DER is that it is highly dependent on the initial model used for transfer learning. When training multiple models, occasionally the initial DER model would not 'learn'. When this occurred, the uncertainty bounds would grow to extreme and unrealistic values, with the accuracy of the predictions dropping. An extreme version of this is seen in the initial slams per hour DER model in the middle subplot of Figure 4.6. Although that figure shows no learning and extreme error bounds, occasionally this would also occur for the initial model when training on less challenging motions such as pitch or heave RMS values. This was a rare occurrence, but without hyperparameters being changed to result in the degraded performance, the unreliability of the model led us to not select it for use in future sections.

Of the three models, DKL showed most reliability in performance across initial model training when compared to DER, and had more reliable uncertainty bounds compared to BNN, highlighted by wider uncertainty bounds at extreme RMS predictions. Although the shrinking uncertainty bands after training in large waves was a concern for DKL models being suitable for out-of-the-box deployment, they served as a valuable tool to demonstrate the flexible uncertainty scaling methods presented in Chapter 5.

Interestingly, the uncertainty based ML models struggled with adapting their small-wave trained models to extreme results for slamming. Unlike nonlinear Froude-Krylov which is present in calm seas, slamming behaves like a switch, and doesn't occur until certain conditions are met. Those conditions that result in slamming for our replica Vanguard rarely occur in seas with significant wave heights equal or less than 3.5 meters (conditions of first testing of model). As a result, the models are trained in the first phase with the majority of weather conditions resulting in zero slams per hour. As such, the models have a challenging time extrapolating this learning to extreme seas, where slams will quickly become more violent and frequent. MAML exceeds in this scenario due to its task-oriented structure which helps the model rapidly adapt for few shot learning. Like MAML, the BNN struggles with forgetfullness. This is shown in the middle and lower subplot of Figure 4.3, where predictions in the small wave domain become relatively constant after training with heavy weather. This tendency of forgetfullness is most likely the reason why it is the only uncertainty based model to succeed in predicting slamming. Without holding onto prior knowledge, it focuses on the dataset at hand and adjusts to slamming frequency in large waves. Due to the challenges of predicting slamming, the rest of this dissertation focuses on Operational Envelopes that consist only of pitch and heave RMS values, with inclusion of slamming reserved for future work, where models can be tweaked further, or a voting scheme can be used to alternate between different models for different targets.

This chapter evaluated Bayesian neural networks, deep evidential regression, and deep kernel learning for uncertainty aware prediction of motion limit states under staged adaptation from routine to extreme sea states. Deep evidential regression achieved the strongest mean accuracy across the staged training protocol and maintained conservative intervals in the large wave domain yet struggled with consistent accuracy of the initial model. Bayesian neural networks were the least reliable in this setting due to sensitivity to prior and KL hyperparameters and due to uncertainty collapse and forgetting under sequential fine tuning. Deep kernel learning provided competitive accuracy together with stable and well behaved uncertainty and low inference cost, which enables cheap evaluation over operational grids and repeated risk queries during routing. For these reasons,

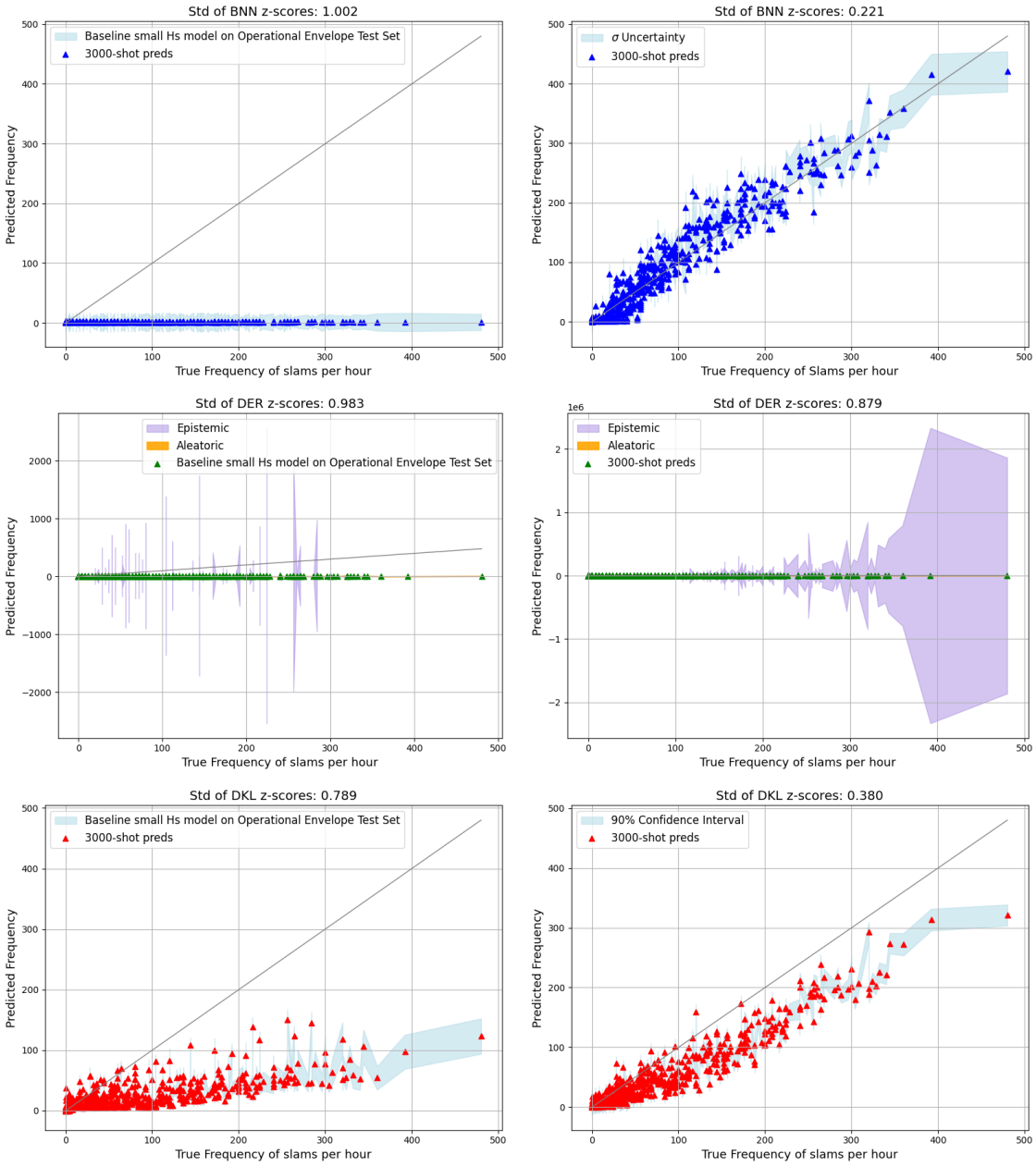


Figure 4.6: Of the uncertainty based ML models, the BNN is the only model to successfully adapt to slamming in an online training representation. The first column represents the models trained with 3000 shots of small wave data, while the second column is those trained models after being exposed to 3000 shots of large wave data.

deep kernel learning is selected as the baseline predictor for the uncertainty scaling mechanisms in Chapter 5 and the risk aware routing demonstration in Chapter 6. Chapter 5 extends this baseline by introducing explicit scaling mechanisms that inflate uncertainty in novel or damage similar conditions, addressing safety margins that are not learned directly from available data.

CHAPTER 5

Uncertainty Scaling

Chapter 4 showed that deep evidential regression achieved the strongest mean accuracy in the staged training benchmark for pitch RMS, but exhibited unreliable initial model performance. Deep kernel learning is therefore selected as the baseline for the uncertainty scaling methods in this chapter because its Gaussian process posterior provides a well behaved increase in predictive variance under extrapolation and supports deterministic, computationally efficient inference. These properties are essential for operational envelope construction and for downstream decision making, where risk must be evaluated repeatedly over large grids of headings, speeds, and sea states.

While the DKL model described in the previous chapter provides principled uncertainty quantification through its Gaussian process posterior, it is desirable to incorporate additional safety margins that encode expert knowledge about the operational domain and known failure modes. The present chapter introduces two post hoc uncertainty scaling mechanisms that act multiplicatively on the model predicted standard deviation, leaving the mean prediction unchanged. Let

$$y(x) \,\big|\, \mathcal{D} \sim \mathcal{N}\big(\mu(x), \sigma^2_{\text{total}}(x)\big) \tag{5.1}$$

denote the DKL predictive distribution at an input x, where $\mu(x)$ is the posterior mean and $\sigma_{\text{total}}(x)$ the base predictive standard deviation. The two scaling factors are defined as

$$\lambda_{\text{OOD}}(x) \geq 1, \qquad \lambda_{\text{damage}}(x) \geq 1, \tag{5.2}$$

and the final scaled uncertainty is

$$\sigma_{\text{scaled}}(x) = \sigma_{\text{total}}(x)\, \lambda_{\text{OOD}}(x)\, \lambda_{\text{damage}}(x), \tag{5.3}$$

while the mean prediction remains

$$\mu_{\text{scaled}}(x) = \mu(x). \tag{5.4}$$

The overall effect is to widen the predictive distribution whenever the query point is judged to be either far from the training distribution or close to known damaging conditions. Both mechanisms are applied at inference time without retraining the DKL model and can, in principle, be applied to any probabilistic regression model that exposes a predictive standard deviation.

5.1 Background and Related Work

Modern deep and kernel-based models often exhibit miscalibrated predictive uncertainties. Even when the mean predictions are accurate, the associated variance may underestimate or overestimate the true dispersion, which can lead to unsafe decisions in safety-critical applications. Post hoc calibration methods are therefore employed to adjust the predictive uncertainty without changing the underlying point predictions. In classification, a prominent example is temperature scaling, which rescales the logits by a single scalar "temperature" to calibrate predicted probabilities while preserving accuracy [Guo et al., 2017]. The uncertainty scaling mechanisms in this chapter adopt a similar philosophy in regression. The predictive mean from the DKL GP is retained, and only the scale of the predictive variance is adjusted.

Detecting out-of-distribution (OOD) inputs is a closely related problem. Many OOD detectors rely on distance-based scores in a learned feature space, with the Mahalanobis distance being a particularly effective choice because it accounts for feature correlations and is invariant to linear rescaling of inputs. Mahalanobis distance based scores have been successfully used for OOD detection in deep networks [Lee et al., 2018a, Podolskiy et al., 2022]. In medical imaging

and other domains, they provide a robust means for novelty detection relative to a reference training distribution [Gonzalez et al., 2022, Deraemaeker and Worden, 2014].

In safety-critical control with Gaussian process models, uncertainty plays a central role in defining safety margins and safe sets. Survey work on GP based learning control emphasizes that overly optimistic uncertainty estimates can lead to unsafe behavior and that conservative but well structured uncertainty bounds are preferable in practice [Wang and Zhang, 2024]. In maritime autonomy, the concept of an operational envelope is used to specify the combinations of environmental and system conditions under which an autonomous ship is considered safe to operate. The uncertainty scaling mechanisms introduced here directly support this envelope based view of safety by modulating the predictive variance in regions associated with increased risk.

The present chapter builds on these ideas by combining a Mahalanobis distance based OOD scaling to inflate uncertainty for rare or extrapolative sea states and a locality based damage scaling that increases uncertainty near sea states that have previously produced structural damage. Both mechanisms retain the learned mean response of the DKL model and only adjust the predictive standard deviation, ensuring that the underlying data driven model is not overridden by heuristic rules but complemented by structured safety factors.

5.1.1 Joint Probability of Failure in Operational Envelope

Example selection of a 90% survivability level for joint pitch and heave risk. A probabilistic operational envelope requires selection of an allowable exceedance probability δ. The value $\delta = 0.1$, or 90% reliability, is used throughout the illustrative envelope examples because it is sufficiently conservative to reflect safety critical decision making while remaining permissive enough to highlight how calibrated uncertainty expands or contracts the feasible operating region. The same framework supports more conservative requirements by adjusting the corresponding normal quantile $z_{1-\delta}$. For a Gaussian predictive distribution $Y(\mathbf{x}) \sim \mathcal{N}(\mu(\mathbf{x}), \sigma^2(\mathbf{x}))$, enforcing $\mathbb{P}(Y(\mathbf{x}) \leq y_{\max}) \geq 1 - \delta$ yields the one sided condition $\mu(\mathbf{x}) + z_{1-\delta}\sigma(\mathbf{x}) \leq y_{\max}$, and for $\delta = 0.1$ the standard normal quantile is $z_{0.9} \approx 1.28$.

When multiple limit states are considered simultaneously, the envelope is formed using a joint risk definition. In the routing environment, pitch and heave are treated as separate limit states with exceedance probabilities computed from their respective predictive means and calibrated standard deviations, and the joint failure probability is defined as the probability that at least one limit is exceeded.

Numerical illustration of the calculations Consider a representative operating condition $\mathbf{x}$ at which the calibrated predictors return $\Theta_{\mathrm{RMS}}(\mathbf{x}) = 3.2^\circ$ with $\sigma_\Theta(\mathbf{x}) = 0.6^\circ$ and $Z_{\mathrm{RMS}}(\mathbf{x}) = 2.1\,\mathrm{m}$ with $\sigma_Z(\mathbf{x}) = 0.4\,\mathrm{m}$. For nominal limits $\Theta_{\mathrm{lim}} = 4.0^\circ$ and $Z_{\mathrm{lim}} = 2.75\,\mathrm{m}$, the exceedance probabilities are

$$p_{\mathrm{ex},\theta} = 1 - \Phi\left(\frac{\Theta_{\mathrm{lim}} - \Theta_{\mathrm{RMS}}(\mathbf{x})}{\sigma_\Theta(\mathbf{x})}\right) = 1 - \Phi\left(\frac{4.0 - 3.2}{0.6}\right) \approx 0.092,$$

$$p_{\mathrm{ex},z} = 1 - \Phi\left(\frac{Z_{\mathrm{lim}} - Z_{\mathrm{RMS}}(\mathbf{x})}{\sigma_Z(\mathbf{x})}\right) = 1 - \Phi\left(\frac{2.75 - 2.1}{0.4}\right) \approx 0.052.$$

Using the union definition in Eq. (6.15), the joint probability of exceeding at least one limit is

$$p_{\mathrm{fail}} = 1 - (1 - p_{\mathrm{ex},\theta})(1 - p_{\mathrm{ex},z}) \approx 1 - (0.908)(0.948) \approx 0.139,$$

so the implied survivability is $1 - p_{\mathrm{fail}} \approx 0.861$. A 90% survivability contour corresponds to the set of operating conditions satisfying $p_{\mathrm{fail}} \leq 0.1$, which is directly comparable to the scalar $1 - \delta$ envelope condition when $\delta = 0.1$. Inflating σ_Θ and σ_Z via out of distribution or damage proximity scaling increases $p_{\mathrm{ex},\theta}$ and $p_{\mathrm{ex},z}$ and therefore increases p_{fail}, contracting the operational envelope in regions of elevated uncertainty or risk.

5.2 Out-of-Distribution Scaling

This section defines a novelty sensitive scaling mechanism that increases predictive variance when environmental conditions deviate from the training distribution, reducing overconfidence during

extrapolation and ensuring caution when experiencing new or unique combinations of weather and mission parameters.

5.2.1 Training Distribution and Mahalanobis Distance

The training distribution is summarized in feature space so that distance based novelty can be computed efficiently and interpreted relative to routine operating conditions.

Consider the normalized input representation

$$\tilde{x} = \left[\tilde{H}_s,\ \tilde{T}_p,\ \tilde{V},\ \widetilde{\sin\beta},\ \widetilde{\cos\beta}\right]^\top \in \mathbb{R}^p, \quad p = 5, \tag{5.5}$$

obtained by standardizing the raw features using the training set mean and standard deviation:

$$\tilde{x} = \frac{x - \mu_{\text{in}}}{\sigma_{\text{in}}}. \tag{5.6}$$

The DKL training procedure stores the empirical mean μ_{train} and the inverse covariance matrix $\Sigma_{\text{train}}^{-1}$ of the normalized training inputs, computed from a "routine weather" dataset that deliberately excludes extreme conditions to represent a vessel gaining experience before venturing into hazardous conditions. These summary statistics define an approximate Gaussian reference distribution for the inputs.

The Mahalanobis squared distance of a normalized input $\tilde{x}$ from this reference distribution is

$$d_M^2(\tilde{x}) = (\tilde{x} - \mu_{\text{train}})^\top \Sigma_{\text{train}}^{-1} (\tilde{x} - \mu_{\text{train}}). \tag{5.7}$$

For multivariate Gaussian data, $d_M^2(\tilde{x})$ follows a χ_p^2 distribution with p degrees of freedom, which means that contours of fixed Mahalanobis distance correspond to ellipsoids of constant probability density in input space [Gallego et al., 2013]. Points with small d_M^2 lie in regions densely populated by training data, whereas large values indicate potential extrapolation [Etherington, 2019]. Mahalanobis distance is particularly suitable here because it naturally accounts for correlations between

environmental variables such as H_s and T_p and is invariant to their physical units.

5.2.2 Smooth OOD Scaling Function

A smooth scaling function is introduced to convert novelty scores into controlled uncertainty inflation without creating discontinuities that would destabilize envelope boundaries.

The OOD scaling mechanism defines a multiplicative factor $\lambda_{\mathrm{OOD}}(x)$ that increases smoothly as $d_M^2(\tilde{x})$ exceeds a chosen threshold T. A logistic shaping function is used to obtain a differentiable, bounded inflation of the predictive variance:

$$\lambda_{\mathrm{OOD}}(x) = 1 + \alpha_{\mathrm{OOD}}\, \sigma_{\log}\left(\frac{d_M^2(\tilde{x}) - T}{s_{\mathrm{OOD}}}\right), \tag{5.8}$$

where $\sigma_{\log}(z) = \frac{1}{1+e^{-z}}$ is the logistic function, $\alpha_{\mathrm{OOD}} > 0$ controls the maximum additional inflation, and $s_{\mathrm{OOD}} > 0$ controls the transition steepness around T. With this choice,

$$1 \leq \lambda_{\mathrm{OOD}}(x) \leq 1 + \alpha_{\mathrm{OOD}}, \tag{5.9}$$

and inputs with $d_M^2(\tilde{x}) \ll T$ receive almost no additional uncertainty, while points far outside the training distribution asymptotically receive a total variance inflated by approximately $(1 + \alpha_{\mathrm{OOD}})^2$ relative to the base GP variance. This method was developed over using a masked on/off region to allow a human operator greater flexibility when increasing caution.

The threshold T is chosen to correspond to a high quantile of the χ_p^2 distribution so that the majority of training samples satisfy $d_M^2(\tilde{x}) \leq T$ and therefore receive negligible scaling. This choice mirrors common practice in Mahalanobis based novelty detection, where large deviations in d_M^2 are interpreted as evidence for outliers or previously unseen conditions. In the present application, the OOD scaling therefore acts as a soft safety factor that reduces confidence when encountering untested combinations of wave height, period, vessel speed and heading.

Algorithm 4 Out-of-distribution uncertainty scaling

Require: training inputs $\{\tilde{x}_n\}_{n=1}^{N}$, query input x, base standard deviation $\sigma_{\text{total}}(x)$, hyperparameters T, α_{OOD}, s_{OOD}

Ensure: OOD scaled uncertainty $\sigma_{\text{OOD}}(x)$

1: Compute training mean $\mu_{\text{train}} = \frac{1}{N}\sum_{n=1}^{N}\tilde{x}_n$

2: Compute training covariance $\Sigma_{\text{train}} = \frac{1}{N-1}\sum_{n=1}^{N}(\tilde{x}_n - \mu_{\text{train}})(\tilde{x}_n - \mu_{\text{train}})^{\top}$

3: Compute and store $\Sigma_{\text{train}}^{-1}$

4: Normalize query input: $\tilde{x} = (x - \mu_{\text{in}})/\sigma_{\text{in}}$

5: Compute Mahalanobis squared distance

$$d_M^2(\tilde{x}) = (\tilde{x} - \mu_{\text{train}})^{\top}\Sigma_{\text{train}}^{-1}(\tilde{x} - \mu_{\text{train}})$$

6: Compute OOD scaling factor

$$\lambda_{\text{OOD}}(x) = 1 + \alpha_{\text{OOD}}\,\sigma_{\log}\left(\frac{d_M^2(\tilde{x}) - T}{s_{\text{OOD}}}\right)$$

7: Return $\sigma_{\text{OOD}}(x) = \sigma_{\text{total}}(x)\,\lambda_{\text{OOD}}(x)$

The scaled predictive standard deviation associated with OOD scaling alone is

$$\sigma_{\text{OOD}}(x) = \sigma_{\text{total}}(x)\,\lambda_{\text{OOD}}(x). \tag{5.10}$$

5.2.3 Algorithm for OOD Uncertainty Scaling

Algorithm 4 summarizes the OOD scaling procedure. The algorithm integrates novelty computation and scaling directly into the prediction pipeline, making out of distribution conservatism available for both envelope plotting and routing decisions. The estimation of μ_{train} and $\Sigma_{\text{train}}^{-1}$ is performed once using the routine weather training dataset and the resulting statistics are stored within the trained model.

The main design choices are the use of Mahalanobis distance, which exploits the full covariance structure of the training inputs and has well understood statistical properties, and the use of a bounded logistic shaping function, which avoids unbounded growth of uncertainty in extremely remote regions while still providing a clear separation between familiar and unfamiliar sea states.

Figure 5.1 illustrates how $\lambda_{\text{OOD}}(x)$ inflates predictive uncertainty for large significant wave

heights and other rarely seen and potentially hazardous conditions, thereby tightening the operational envelope derived from the model.

Out of distribution scaling provides a systematic mapping from novelty to conservatism, reducing the likelihood of unsafe overconfidence in rarely observed sea states. The next section introduces damage informed scaling, which targets uncertainty inflation around conditions associated with observed or hypothesized structural degradation.

5.3 Post Inspection Damage Scaling

The second scaling mechanism incorporates post-mission feedback, specifically knowledge of any damage or extreme events the vessel has experienced, to locally adjust uncertainty. If an ASV returns from a mission with observed structural damage or system failures that can be attributed to certain sea conditions, these conditions are recorded as particularly hazardous. Subsequent predictions near these conditions should reflect a higher uncertainty even if such conditions lie within the original training distribution.

5.3.1 Damage Condition Representation

Damage conditions are represented in the same input space as operational queries, enabling direct comparison between current sea states and prior dangerous ones.

A representative "damage condition" is encoded as a raw input vector

$$x_{\text{dam}} = \left[H_s^{\text{dam}},\ T_p^{\text{dam}},\ V^{\text{dam}},\ \beta^{\text{dam}}\right]. \tag{5.11}$$

For compatibility with the trained model, this is converted to the normalized representation

$$\tilde{x}_{\text{dam}} = \frac{[H_s^{\text{dam}},\ T_p^{\text{dam}},\ V^{\text{dam}},\ \sin\beta^{\text{dam}},\ \cos\beta^{\text{dam}}] - \mu_{\text{in}}}{\sigma_{\text{in}}}, \tag{5.12}$$

so that both the query input $\tilde{x}$ and the damage condition live in the same standardized five dimen-

Table 5.1: Recorded damage conditions

H_s (m)	T_p (s)	V (m/s)	β (deg)
7.0	10.0	13	180

sional feature space. The implementation stores $\tilde{x}_{\text{dam}}$ once the damage event has been identified. Table 5.1 summarizes the damage conditions used in the present study.

5.3.2 Feature Importance via Sensitivity Analysis

Not all input dimensions contribute equally to vessel response. To reflect this, the damage scaling mechanism uses feature importance weights derived from a one-at-a-time perturbation analysis on the trained model. Sensitivity analysis is used to identify which input dimensions most strongly influence response near the damage condition, supporting weighted proximity measures. For a set of representative validation samples $x^{(k)} = [H_s^{(k)}, T_p^{(k)}, V^{(k)}, \beta^{(k)}]$, each feature is perturbed by a small relative amount (here ±10%) and the change in model output is recorded:

$$s_i = \frac{1}{K} \sum_{k=1}^{K} \left| f\left(x^{(k)} + \delta_i^{(k)} e_i\right) - f\left(x^{(k)}\right) \right|, \tag{5.13}$$

where e_i is the ith basis vector in feature space and $\delta_i^{(k)}$ denotes a small perturbation applied to feature i of sample k. The resulting scores s_i are normalized to yield feature importance weights

$$w_i = \frac{s_i}{\sum_j s_j}, \qquad \sum_i w_i = 1, \qquad w_i \geq 0. \tag{5.14}$$

These weights quantify how much variability in the predicted response can be attributed to each feature. Table 5.2 shows the resulting weights for heave and pitch. In both degrees of freedom, significant wave height is the dominant feature, followed by wave period and heading, with vessel speed having the smallest influence, consistent with physical intuition and the observed polar plots.

The feature importance computation is implemented inside the test routine by evaluating the DKL model under one at a time perturbations and storing the resulting normalized weights in a

Table 5.2: Feature importance weights from perturbation analysis

Degree of freedom	H_s (m)	T_p (s)	V (m/s)	β (deg)
Heave	0.7948	0.0695	0.0134	0.1223
Pitch	0.5242	0.2198	0.0108	0.2452

buffer that is then reused by the damage regularizer.

5.3.3 Damage Proximity Measure

A proximity score is constructed to quantify similarity to damage associated conditions in a way that reflects feature importance and correlation structure. The proximity measure is needed to localize scaling to neighborhoods of operating conditions that resemble the recorded damage inducing state.

Given a normalized query input

$$\tilde{x} = \left[\tilde{H}_s,\ \tilde{T}_p,\ \tilde{V},\ \widetilde{\sin\beta},\ \widetilde{\cos\beta}\right] \tag{5.15}$$

and a stored normalized damage condition $\tilde{x}_{\text{dam}}$, a feature wise distance vector $\Delta(\tilde{x}) \in \mathbb{R}^4$ is defined as

$$\Delta_1(\tilde{x}) = \left|\tilde{H}_s - \tilde{H}_s^{\text{dam}}\right|, \tag{5.16}$$

$$\Delta_2(\tilde{x}) = \left|\tilde{T}_p - \tilde{T}_p^{\text{dam}}\right|, \tag{5.17}$$

$$\Delta_3(\tilde{x}) = \left|\tilde{V} - \tilde{V}^{\text{dam}}\right|. \tag{5.18}$$

For the heading, a periodic distance is required to respect the $0^\circ/360^\circ$ wrap around. Let

$$\beta(\tilde{x}) = \text{atan2}\left(\widetilde{\sin\beta}, \widetilde{\cos\beta}\right), \tag{5.19}$$

$$\beta(\tilde{x}_{\text{dam}}) = \text{atan2}\left(\widetilde{\sin\beta}^{\text{dam}}, \widetilde{\cos\beta}^{\text{dam}}\right), \tag{5.20}$$

both in radians. The smallest angular separation normalized to $[0, 1]$ is then

$$\Delta_4(\tilde{x}) = \frac{1}{\pi} \left| \left((\beta(\tilde{x}) - \beta(\tilde{x}_{\text{dam}}) + \pi) \bmod 2\pi \right) - \pi \right|. \tag{5.21}$$

The full distance vector is

$$\Delta(\tilde{x}) = \big(\Delta_1,\ \Delta_2,\ \Delta_3,\ \Delta_4\big). \tag{5.22}$$

A locality threshold $\tau > 0$ is introduced to restrict the influence of the damage event to a neighborhood around $\tilde{x}_{\text{dam}}$. Only dimensions with $\Delta_i(\tilde{x}) < \tau$ are considered "close" for the purpose of scaling.

For each feature, a proximity contribution is defined as

$$p_i(\tilde{x}) = \begin{cases} 1 - \Delta_i(\tilde{x}), & \text{if } \Delta_i(\tilde{x}) < \tau, \\ 0, & \text{otherwise,} \end{cases} \tag{5.23}$$

so that smaller offsets from the damage condition produce larger contributions, while points outside the locality threshold for a given feature contribute nothing.

The raw proximity score is a weighted sum

$$s_{\text{raw}}(\tilde{x}) = \sum_{i=1}^{4} w_i\, p_i(\tilde{x}), \tag{5.24}$$

where the weights w_i are the feature importance weights described above. In batch evaluation, s_{raw} is normalized to lie in $[0, 1]$ by dividing by its maximum over the batch,

$$s(\tilde{x}) = \frac{s_{\text{raw}}(\tilde{x})}{\max_{\tilde{x}'} s_{\text{raw}}(\tilde{x}') + \varepsilon}, \qquad 0 \le s(\tilde{x}) \le 1, \tag{5.25}$$

where a small constant ε avoids division by zero when no points are close to the damage condition. In practice, if no feature distance satisfies $\Delta_i(\tilde{x}) < \tau$ for any i, the damage scaling reverts to one.

5.3.4 Damage Based Scaling Factor

A scaling factor is defined to translate damage proximity into conservative variance inflation while maintaining numerical stability and interpretability. The scaling factor formalizes how uncertainty is inflated near the damage condition.

The damage based scaling factor is defined as

$$\lambda_{\text{damage}}(x) = \begin{cases} 1 + \lambda_d \, s(\tilde{x}), & \text{if } \min_i \Delta_i(\tilde{x}) < \tau, \\ 1, & \text{otherwise,} \end{cases} \tag{5.26}$$

where $\lambda_d \geq 0$ controls the maximum additional inflation in the vicinity of the damage condition. This definition ensures that:

- the scaling applies only in a neighborhood of the recorded damage condition
- proximity in more influential features (larger w_i) contributes more strongly to the scaling
- the scaling transitions smoothly as the query point moves away from the damage point

The scaled predictive standard deviation due only to the damage mechanism is

$$\sigma_{\text{damage}}(x) = \sigma_{\text{total}}(x) \, \lambda_{\text{damage}}(x). \tag{5.27}$$

Because the damage scaling operates purely on the predictive variance, it can be updated whenever a new damage event is observed without retraining the ML model. The damage condition vector and feature importance weights are simply updated, and the subsequent predictions automatically reflect the revised safety margin.

5.3.5 Algorithm for Damage Based Uncertainty Scaling

A computationally tractable procedure enables damage scaling to be evaluated repeatedly during mission planning and route execution without retraining the underlying surrogate.

Algorithm 5 Post inspection damage based uncertainty scaling

Require: stored normalized damage condition $\tilde{x}_{\text{dam}}$, feature weights $w \in \mathbb{R}^4$, locality threshold τ, scaling amplitude λ_d, query input x, base standard deviation $\sigma_{\text{total}}(x)$
Ensure: damage scaled uncertainty $\sigma_{\text{damage}}(x)$

Convert raw query input x to normalized representation $\tilde{x} = [\tilde{H}_s, \tilde{T}_p, \tilde{V}, \widetilde{\sin\beta}, \widetilde{\cos\beta}]$
Compute feature wise distances $\Delta_1, \Delta_2, \Delta_3$ and angular distance Δ_4 as defined above
if $\min_i \Delta_i \geq \tau$ **then**
    return $\sigma_{\text{damage}}(x) = \sigma_{\text{total}}(x)$
end if
For each i set $p_i = 1 - \Delta_i$ if $\Delta_i < \tau$, else $p_i = 0$
Compute raw proximity score $s_{\text{raw}} = \sum_{i=1}^{4} w_i p_i$
Normalize to obtain $s = s_{\text{raw}} / (\max_{\tilde{x}'} s'_{\text{raw}} + \varepsilon)$
Compute scaling factor $\lambda_{\text{damage}}(x) = 1 + \lambda_d\, s$
return $\sigma_{\text{damage}}(x) = \sigma_{\text{total}}(x)\, \lambda_{\text{damage}}(x)$

Algorithm 5 summarizes the damage based scaling procedure for a single stored damage condition. Extension to multiple damage events would typically involve taking the maximum or another aggregate of the individual scaling factors over all recorded incidents.

Figure 5.2 shows the effect of the damage based scaling for the recorded damage condition in Table 5.1. As a visual reference, the heave polar plot displays a widened uncertainty band in head seas near the 180° heading where the damage condition has been placed. Damage scaling grounds uncertainty inflation in operational experience, concentrating conservatism where prior evidence suggests an elevated likelihood of limit exceedance or degradation.

5.4 Combination of OOD and Damage Scaling

This section consolidates statistical novelty based scaling and damage based scaling into a single framework that can be evaluated efficiently during operation. In deployment, both scaling mechanisms are applied multiplicatively to the base DKL uncertainty according to Equation 5.3. Given a query input x, the prediction pipeline is as follows:

1. Normalize x to obtain $\tilde{x}$.

2. Evaluate the DKL GP posterior to obtain $\mu(x)$ and $\sigma_{\text{total}}(x)$.

3. Compute $\lambda_{\text{OOD}}(x)$ from the Mahalanobis distance $d_M^2(\tilde{x})$ using (5.8).

4. Compute $\lambda_{\text{damage}}(x)$ from the damage proximity score $s(\tilde{x})$ using (5.26).

5. Form the scaled uncertainty $\sigma_{\text{scaled}}(x) = \sigma_{\text{total}}(x)\lambda_{\text{OOD}}(x)\lambda_{\text{damage}}(x)$.

This procedure is implemented in the `predict` method of the `DKLRegressor`, which first obtains the base GP posterior, then applies the damage regularizer and the Mahalanobis based OOD scaling to the standard deviation.

The scaled uncertainty enters the operational envelope by determining which sea states satisfy given motion or load limits at a specified safety probability. For example, suppose a scalar response $Y(x)$ is approximated as Gaussian with mean $\mu(x)$ and scaled standard deviation $\sigma_{\text{scaled}}(x)$ and the operator wishes to enforce $Y(x) \leq y_{\max}$ with probability at least $1 - \delta$. A one sided Gaussian bound yields the constraint

$$\mu(x) + z_{1-\delta}\, \sigma_{\text{scaled}}(x) \leq y_{\max}, \tag{5.28}$$

where $z_{1-\delta}$ is the $(1 - \delta)$ quantile of the standard normal distribution (for example $z_{0.9} \approx 1.28$). Inflating $\sigma_{\text{scaled}}(x)$ via either $\lambda_{\text{OOD}}(x)$ or $\lambda_{\text{damage}}(x)$ therefore shrinks the set of x that satisfy this inequality, resulting in a more conservative operational envelope in regions with higher inferred risk. This behavior is desirable since it narrows allowable operations in rare extreme seas and in conditions similar to those that previously produced damage, while leaving operations in well explored and benign conditions largely unchanged.

Figure 5.3 illustrates the combined effect of OOD and damage scaling on pitch, heave and the resulting operational envelope. Large wave heights that are far from the training distribution and headings near the recorded damage state exhibit substantially inflated uncertainty, leading to a conservative but interpretable safety margin around these high risk regions.

The combined scaling mechanism yields an envelope that contracts selectively in novel or damage proximate regimes while remaining comparatively permissive in well characterized conditions.

5.5 Discussion of Design Choices

The two scaling mechanisms are deliberately constructed as multiplicative adjustments on the predictive standard deviation rather than as modifications to the mean. This design preserves the data driven mean response learned by the DKL GP while modulating only the confidence in that prediction. This is conceptually similar to post hoc calibration methods in classification that adjust predicted probabilities while leaving decision boundaries unchanged.

The choice of Mahalanobis distance for OOD scaling reflects its long history as a robust measure of multivariate distance that accounts for input correlations and scales. By estimating the training covariance from routine weather data, the model gains a statistically meaningful notion of what constitutes a "typical" sea state. The logistic shaping of $\lambda_{\mathrm{OOD}}(x)$ ensures that the scaling varies smoothly and saturates at a finite maximum, avoiding overly pessimistic uncertainty in very remote regions that may be irrelevant for planned operations.

The damage based scaling, in contrast, is local and asymmetric in feature space. It is tailored to encode specific post mission observations of structural damage, with the size and shape of the neighborhood of influence controlled by the locality threshold τ and the feature importance weights w_i. The use of a periodic angular distance in heading and a feature sensitivity based weighting scheme ensures that the scaling respects both the physics of ship motion and the learned response of the model. The mechanism thereby bridges structural health monitoring concepts, where Mahalanobis type distances and local novelty detectors are used to flag abnormal behavior, with operational envelope based safety management in autonomous marine systems [Deraemaeker and Worden, 2014].

Together, the OOD and damage scaling mechanisms provide a flexible framework for integrating expert knowledge and operational feedback into probabilistic motion prediction models. The approach remains computationally efficient, as both scalings can be evaluated in closed form given stored training statistics and damage conditions, and it avoids the need for retraining when updating safety margins. This makes the proposed uncertainty scaling particularly suitable for long lived autonomous systems that must adapt their safety envelopes over time as more data and

experience become available.

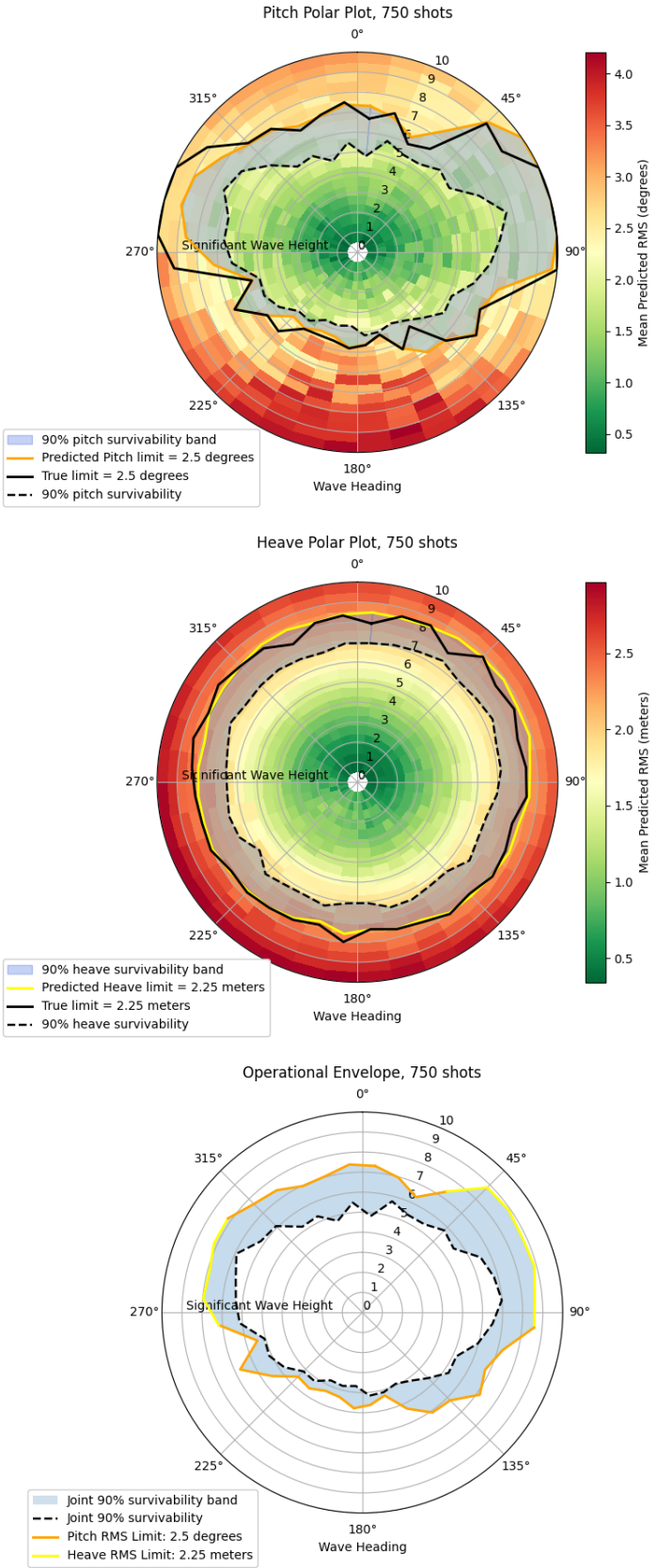


Figure 5.1: OOD scaling inflates predictive uncertainty in rarely observed and potentially hazardous sea states, resulting in a more conservative operational envelope.

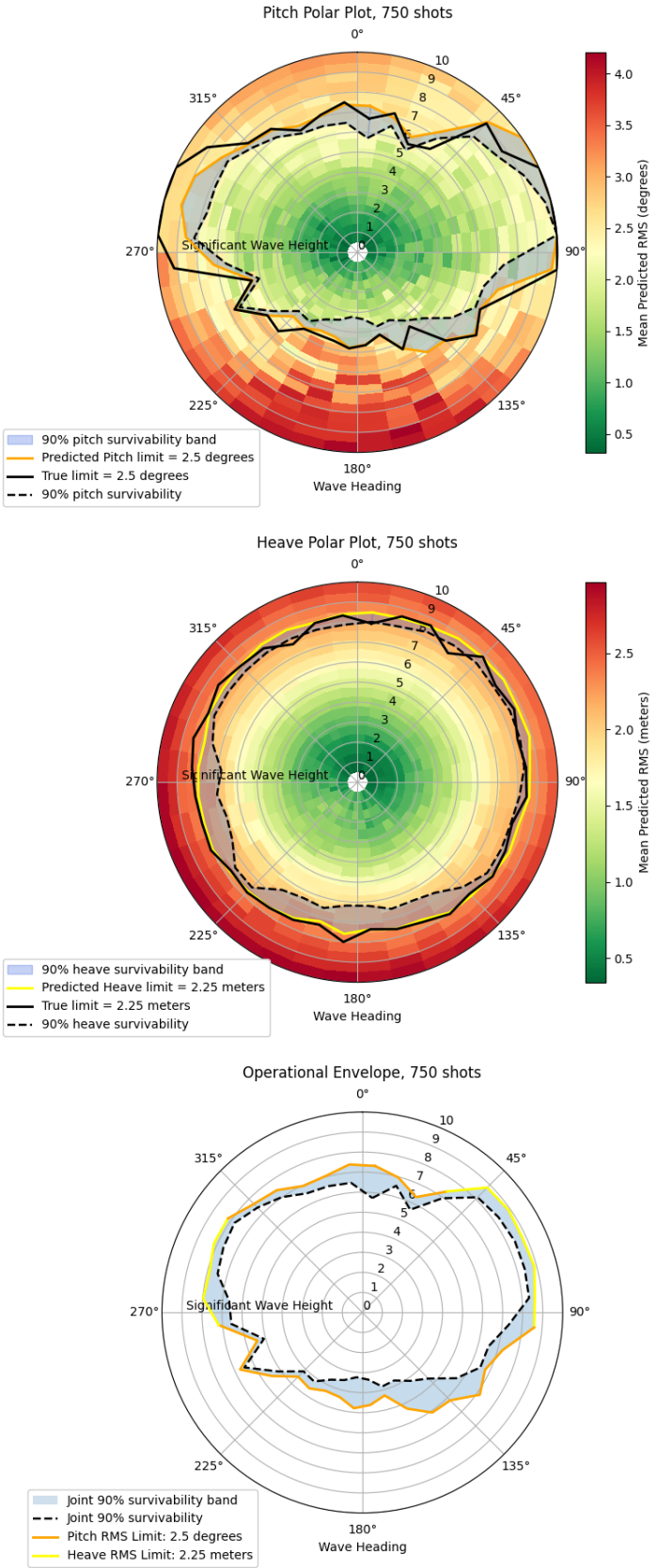


Figure 5.2: Post inspection damage scaling increases uncertainty in a localized neighborhood of the recorded damage condition, tightening the operational envelope near previously hazardous states.

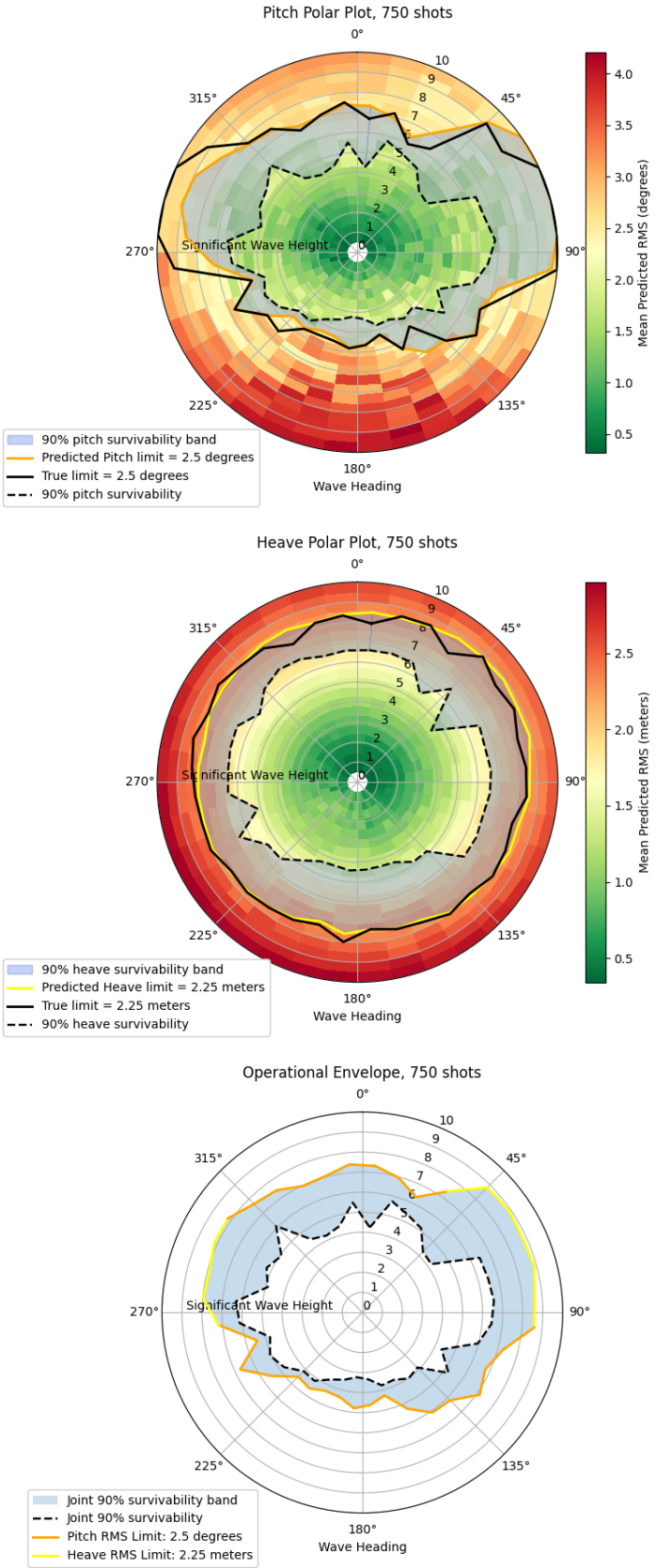


Figure 5.3: Combined OOD and damage scaling yields a conservative operational envelope that reduces the risk of operating the ASV in extreme or previously damaging sea states.

CHAPTER 6

Reinforcement Learning Path Routing

This work finishes with a demonstration using uncertainty aware models to safely route a simulated autonomous vessel through single and multi-waypoint missions, while being aware of forecasted weather to make informed decisions to efficiently and safely complete the mission. To do this, a Reinforcement Learning (RL) agent is trained using the Deep Deterministic Policy Gradient (DDPG) algorithm.

6.1 Reinforcement Learning for Marine Routing

Over the last decade, RL has been increasingly applied to maritime path planning, obstacle avoidance, and weather routing. Classical Q learning and Deep Q Network (DQN) methods have been adopted for grid based route finding, while actor critic algorithms such as Deep Deterministic Policy Gradient and Proximal Policy Optimization (PPO) have enabled control in continuous heading and speed spaces, often with the aim of reducing fuel consumption or transit time.

One line of work focuses on local path planning for smart ships in restricted waters. Chen et al. [2019] formulated ship path planning as a Q learning problem in a discretized environment. The state representation encodes grid location and heading, and the action space consists of discrete rudder commands. The reward function combines progress towards the goal with penalties for approaching obstacles, so that the learned policy realizes collision free routes that respect approximate turning constraints of the vessel. Environmental forcing and seakeeping responses are

not modeled, and safety is expressed only through geometric separations from obstacles and the shoreline.

Lin et al. [2025] extended Q learning based approaches to unmanned surface vehicles by combining electronic chart display and information system (ECDIS) data with wind and current fields in a multimodal fusion DQN architecture. Their system constructs a raster map with both static charted obstacles and dynamic circular risk zones and augments this with interpolated wind and ocean current features. The reinforcement learning agent operates on this multimodal state to generate discrete heading adjustments, optimizing a reward that encourages reaching the target, short path length, and collision avoidance. Although meteorological data are included, the effect of the environment on the vessel is represented implicitly through drift and collision risk, rather than through an explicit model of ship motions or structural loads.

Reinforcement learning has also been employed for open ocean weather routing and voyage optimization. Moradi et al. [2022] considered an Atlantic crossing case study and trained DQN, DDPG, and PPO agents to deviate from a great circle route in response to time varying wave and wind fields. In their framework, the agent controls both heading and speed, and weather information is supplied along the route from a dynamic forecast database. The reward is formulated to realize different optimization strategies, including pure fuel minimization subject to a maximum voyage duration and a multi objective case that balances fuel consumption with a simple safety term based on significant wave height. Ship safety is therefore quantified by penalizing large wave heights, without an explicit assessment of the vessel motion response or failure probabilities for specific tasks or structural components.

Latinopoulos et al. [2025] developed a deep reinforcement learning framework for voyage optimization that combines Double DQN and DDPG with high resolution ECMWF forecast products. The environment is defined on a latitude longitude grid aligned with the meteorological data, and the state vector contains ship position and speed together with the interpolated wind and current conditions along the prospective route. In the continuous control variant a DDPG agent selects speed and course at each step, with exploration driven by Ornstein Uhlenbeck noise. The reward

integrates distance, fuel consumption, and arrival time objectives, and the authors demonstrate reductions in fuel oil consumption and improved just in time arrival performance relative to a fixed speed baseline. Safety is implicitly addressed through avoidance of land and restricted zones, but the method does not incorporate a physics based model of vessel motions or an operational envelope defined in terms of motion or load limit states.

The reinforcement learning path routing framework developed in this dissertation differs from the studies above in several important respects. First, the environment state is augmented with predictions from Deep Kernel Learning (DKL) motion models that map local sea state parameters, vessel heading, and speed to distributions of heave and pitch response, including predictive uncertainty. These models are trained and used to construct an operational envelope in the space of heading, speed, and wave conditions. The DKL motion predictors and uncertainty scaling parameters are trained prior to reinforcement learning and remain fixed during policy optimization. During RL training, only the actor and critic networks are updated, ensuring that changes in routing behavior are attributable to policy learning rather than to changes in the motion prediction model. Second, the reward function includes a dedicated risk term R_{risk} defined as the joint probability of failure of the motion limit states implied by the operational envelope. This term penalizes states where the predicted probability that heave or pitch exceed their allowable RMS limits is large. It is assigned a comparatively high weight in the overall reward so that the agent prioritizes routes that maintain low failure probability even when this increases path length or transit time. Episodes are also terminated with a large negative reward if the operational limits are crossed.

Finally, the routing agent is trained in a setting where both nowcast and forecast weather fields are available. As a result the learned policy is not only sensitive to instantaneous sea state at the vessel location, but also to the forecast evolution of conditions along candidate trajectories and to the confidence of the motion predictions in those conditions. The resulting controller therefore uses reinforcement learning to trade progress, time, and speed objectives against a probabilistic quantification of motion risk, in contrast to existing routing schemes that operate without an explicit

probability of failure representation.

6.1.1 Deep Deterministic Policy Gradient

The routing problem involves continuous action choices in heading and speed, motivating an actor critic method capable of learning continuous control policies from forecast conditioned state representations.

Deep Deterministic Policy Gradient (DDPG), developed by Lillicrap et al. [2019], is an off policy, model free, actor critic algorithm formulated for continuous state and action spaces. In the DDPG framework an actor network $\mu(s \mid \theta_\mu)$ maps a state vector s to a deterministic control action a, while a critic network $Q(s, a \mid \theta_Q)$ estimates the expected discounted return associated with executing action a in state s and subsequently following the actor policy. The critic provides a differentiable approximation of the action value function that is used both for policy evaluation and for improving the actor by deterministic policy gradient updates. The deterministic policy gradient theorem shows that, for a parameterized deterministic policy, the gradient of the performance objective can be written as an expectation of $\nabla_a Q(s, a \mid \theta_Q) \nabla_{\theta_\mu} \mu(s \mid \theta_\mu)$ evaluated at $a = \mu(s)$, which allows direct policy optimization in continuous action spaces without sampling from a stochastic policy.

In practice, DDPG employs several mechanisms that improve stability during learning. Experience tuples (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer, and mini batches are sampled uniformly when updating the networks. This decorrelates successive samples and improves data efficiency. Target networks $Q'(s, a \mid \theta_{Q'})$ and $\mu'(s \mid \theta_{\mu'})$ are maintained as slowly updated copies of the critic and actor. During each gradient step the critic target y_j is computed using the target networks, and the target parameters are updated by a soft update rule

$$\theta_{Q'} \leftarrow \tau \theta_Q + (1 - \tau)\theta_{Q'}, \qquad \theta_{\mu'} \leftarrow \tau \theta_\mu + (1 - \tau)\theta_{\mu'}, \tag{6.1}$$

where $0 < \tau \ll 1$. These features follow the standard DDPG implementation and are summarized

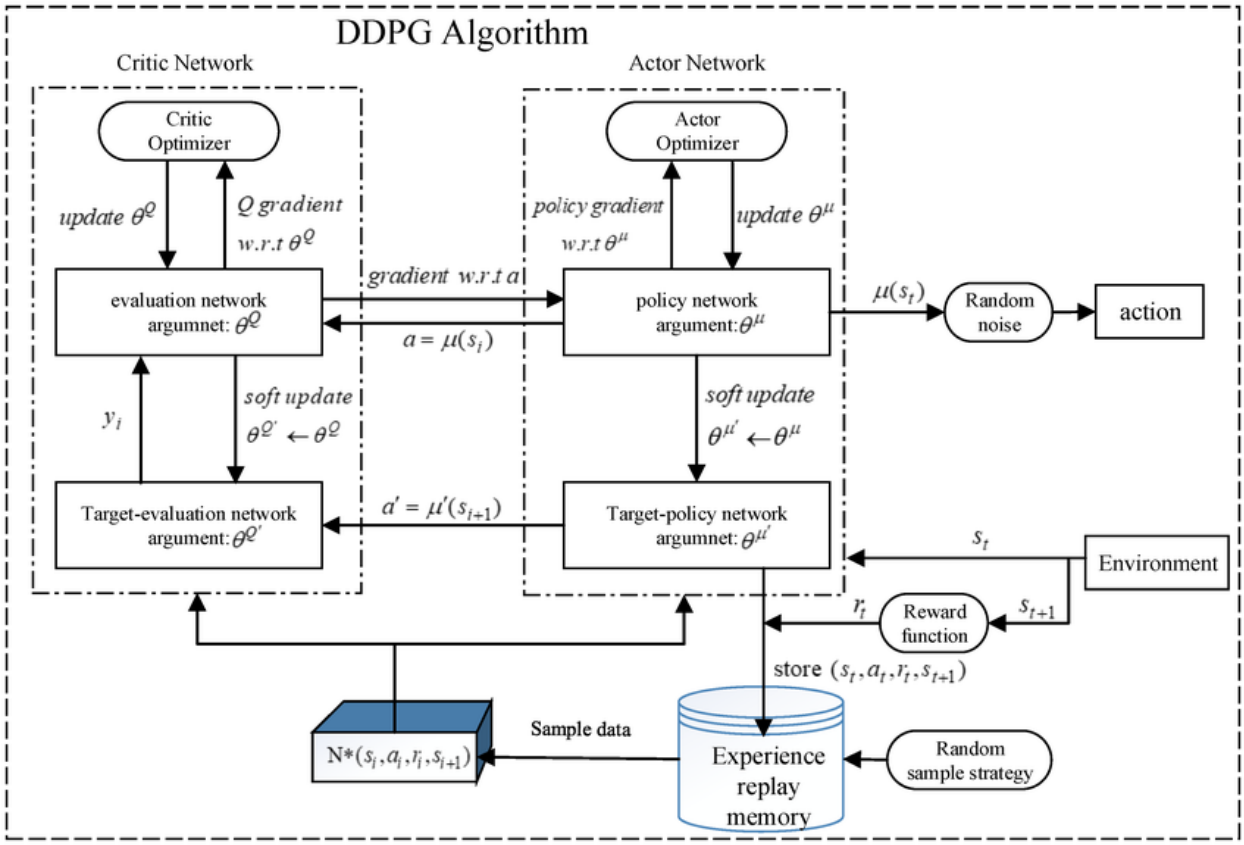


Figure 6.1: DDPG Flowchart [Guo et al., 2020]

in Algorithm 6.

For the reinforcement learning path routing problem the DDPG algorithm is used with a continuous two dimensional action vector that controls the vessel speed through water and course. The state vector contains the vessel position, speed, and heading together with a compact representation of the nowcast and forecast environmental conditions at the present and future locations. The critic therefore learns a value function $Q(s, a)$ that jointly reflects expected progress towards mission waypoints, transit time, and the risk of exceeding motion limits through the reward function described in Section 6.2.3.1. The deterministic actor provides a direct mapping from state to continuous control, avoiding the need to discretize speed and heading as in DQN or Q learning based weather routing methods, and allowing the policy to exploit small adjustments in speed and bearing that can have a large effect on accumulated risk and travel time.

Algorithm 6 Deep Deterministic Policy Gradient (DDPG) [Lillicrap et al., 2019]

Randomly initialize critic network $Q(s, a \mid \theta^Q)$ and actor $\mu(s \mid \theta^\mu)$ with weights θ^Q and θ^μ
Initialize target networks Q' and μ' with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$
Initialize replay buffer $\mathcal{R}$
for episode = 1 to M **do**
    Initialize a noise process $\mathcal{N}$ for action exploration (Ornstein Uhlenbeck)
    Receive initial state s_1 (vessel position, course, speed, current, and forecasted weather conditions)
    for $t = 1$ to T **do**
        Select continuous action $a_t = \mu(s_t \mid \theta^\mu) + \mathcal{N}_t$ (vector of vessel STW and bearing)
        Execute action a_t in the environment and observe reward r_t and next state s_{t+1}
        Store transition (s_t, a_t, r_t, s_{t+1}) in replay buffer $\mathcal{R}$
        Sample minibatch of N transitions (s_j, a_j, r_j, s_{j+1}) from $\mathcal{R}$
        Compute target value

$$y_j = r_j + \gamma\, Q'\Big(s_{j+1},\ \mu'(s_{j+1} \mid \theta^{\mu'}) \,\Big|\, \theta^{Q'}\Big)$$

        Update critic by minimizing the loss

$$L = \frac{1}{N} \sum_j \Big(y_j - Q(s_j, a_j \mid \theta^Q)\Big)^2$$

        Update actor policy using the sampled policy gradient

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_j \nabla_a Q(s, a \mid \theta^Q)\big|_{s=s_j,\, a=\mu(s_j)}\, \nabla_{\theta^\mu} \mu(s \mid \theta^\mu)\big|_{s_j}$$

        Update target networks with soft updates

$$\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}, \qquad \theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

    end for
end for

6.1.1.1 Action Noise and Ornstein-Uhlenbeck Exploration

Within the DDPG algorithm, the deterministic actor would converge rapidly to a greedy strategy if actions were executed exactly as $\mu(s_t)$ during training. In continuous control problems such as ship routing, this lack of exploration can lead to premature convergence to suboptimal policies and poor coverage of the state action space. To promote exploration the DDPG formulation perturbs the actor output with an additive noise process N_t, so that the action applied to the environment is $a_t = \mu(s_t) + N_t$ as in Algorithm 6.

A simple choice for N_t is independent Gaussian noise, but this can produce unrealistically rapid changes in heading and speed from one time step to the next. For systems with inertia, such as ship maneuvering, it is desirable that exploration noise is temporally correlated, so that consecutive actions vary smoothly. To achieve this, an Ornstein-Uhlenbeck (OU) process is used for the action noise [Uhlenbeck and Ornstein, 1930]. The OU process is a mean reverting Gaussian process that satisfies the stochastic differential equation

$$\mathrm{d}\eta_t = \theta \left(\mu_\eta - \eta_t\right) \mathrm{d}t + \sigma \, \mathrm{d}W_t, \tag{6.2}$$

where θ is the mean reversion rate, μ_η is the long term mean, σ controls the instantaneous volatility, and W_t is a standard Wiener process. In discrete time with step size Δt this can be approximated as

$$\eta_{t+1} = \eta_t + \theta \left(\mu_\eta - \eta_t\right) \Delta t + \sigma \sqrt{\Delta t} \, \epsilon_t, \tag{6.3}$$

where $\epsilon_t \sim \mathcal{N}(0, 1)$. The noise term N_t in Algorithm 6 is obtained by evaluating this process separately for the speed and course components of the action vector.

Using OU noise has two important effects in the routing environment. First, the correlated perturbations produce smooth variations in speed and heading across successive time steps, which better reflect the physical limitations of the vessel and avoid the highly oscillatory behavior that can arise with uncorrelated noise. Second, the noise amplitude can be scheduled to decay over the

course of training, which allows broad exploration during early episodes and gradually focuses the policy on exploitation of the best trajectories once the value function has stabilized [Hollenstein et al.]. In this study the OU noise parameters are chosen so that the initial exploration produces moderate deviations in both speed and heading relative to the nominal policy.

By combining OU exploration with the risk aware reward function, the agent is encouraged to sample trajectories that approach the boundaries of the operational envelope while receiving strong negative rewards when the predicted probability of failure is high or the envelope is exceeded. Over training, this allows the actor to learn control strategies that remain within low risk regions of the envelope, even in the presence of severe forecasted weather.

To demonstrate how an autonomous vessel can use it's operational envelope to safely route through a mission, this dissertation concludes with a demonstration using the trained DKL models from Chapter 5 to predict motion limits using nowcast and forecast data to visit mission waypoints while not surpassing operational limits. To do this, we developed a custom reinforcement learning environment, and leverage the Deep Deterministic Policy Gradient (DDPG) algorithm to develop a trained agent that uses forecast weather to predict future probability of failure, and has the ability to continuously modify its heading and speed, which in turn changes the motion response, allowing it to safely route itself through severe weather and accomplish it's mission without surpassing operational limits that can damage the vessel.

6.2 Environment

Forecast conditioned routing requires a state representation that links mission geometry to future sea states at reachable locations, so that the agent can anticipate risk rather than reacting only to local conditions. The routing environment uses the nowcast and forecast wave fields described in Tools Section 2.1.2, which are distinct from the ECMWF hindcast data used for motion model training in Section 2.1.1. Interpolation between discrete nowcast and forecast updates follows the procedure summarized in Section 2.1.3.

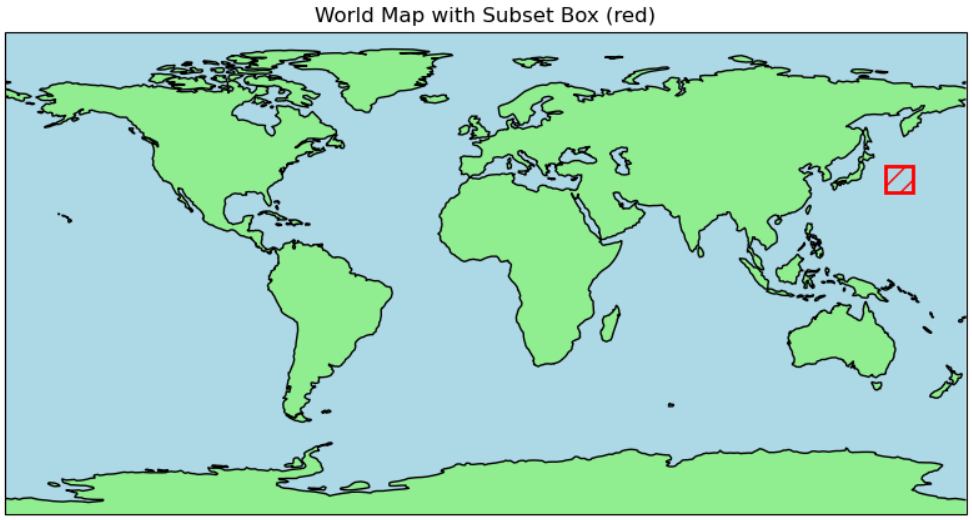


Figure 6.2: The red box shows the geographic area for the RL demonstration

The custom environment developed for this task uses Gymnasium (gym) and Stable Baselines3 for inclusion of RL algorithms [Towers et al., Raffin et al., 2021]. The weather data collected for this demonstration includes nowcast and forecast weather data from December 18th through December 28th of 2024 in the geographic area delineated in Figure 6.2. A forecast horizon of 350 hours was used, allowing the ASV the ability to look over two weeks into the future. A weather update of 6 hours was used, at which point nowcast and forecast files were updated. Between the weather updates, interpolation was used as explained in Section 2.1.3.

The operational limits that were placed on the vessel were 4.0 degrees for pitch and 2.5 meters for heave. If the limits were surpassed, the environment was reset and the task the ASV was attempting failed. As a reminder, the operational envelope uses a variety of tasks, such as station keeping, launching drones from deck, or following track. Although RMS limits of 4.0 degrees for pitch and 2.5 meters for heave are unlikely to result in total loss of the vessel, they trigger routinely in the collected weather if operating into head seas at high speed, resulting in an interesting and challenging simulation environment.

To leverage pre-built packages to assist with the demonstration, a custom gym environment

was developed. Gym is a standard API for reinforcement learning training. To make a custom gym environment, several functions need to integrated beyond inheriting from the abstract class gymnasium.Env. The mandatory functions are `reset`, `step`, and the recommended `get_obs`.

6.2.1 Inclusion of Weather Forecast Data

When creating the environment, the first thing done is to discretize the operational bounds into a grid by 1 degree increments in latitude and longitude. These node locations are used by the agent to incorporate weather data forecasts.

To do this, the distance is calculated between the vessel's current position and each of the grid node locations by solving the inverse geodesic problem as described in the Tools Section 2.1.7.

After the distance is measured from the vessel to each of the grid nodes, the current speed is used with the distance to determine the travel time to that node location. The transit time is then added to the current simulation time to determine the projected arrival time of the vessel to that grid point. The location of the grid node, current time, and arrival time are used to lookup the forecasted weather data. For each node in the grid, the significant wave height, peak wave period, and primary wave direction are returned. If the time of arrival is past the horizon window of the forecast data, such as when the vessel is transiting at speeds close to 0 knots, the weather data at the end of the current forecast horizon is used.

6.2.2 `reset` Function

Robust policy learning depends on exposing the agent to diverse initial conditions and mission instances, which are controlled through environment initialization and episode reset logic.

`Reset` is responsible for resetting the environment to its initial state and marks the initiation of a new episode. In this demonstration, a more robust agent was trained by having the starting position of the vessel randomly selected within the operational bounds. One restriction placed on the location of the randomized start location was that it had to be at least 25 nautical miles away from a waypoint.

After reset is called, the episode counter is incremented and the ship heading, speed, and reward are reset to 0.0.

6.2.3 `step` Function

The environment's `step` function is where vessel motion, forecast queries, and operational limit based termination criteria are coupled into a single decision process. The step function is responsible for taking the commanded action, applying that action, and progressing the simulation environment due to that action and the natural progression of the weather. This entails applying the continuous heading (0-360 degrees) and speed [0-20 knots] commands. Using the heading and speed, the kinematics are updated and the vessel moves within the environment.

To simulate an ASV using a machine learning model during operation that is not perfectly accurate, a higher fidelity model was used to predict the motions with the nowcast data. Regardless of simulation or in situ measurements, the trained model will always be less accurate than the actual motions and loads onboard the vessel. In lieu of real-world sensor data, this multi-fidelity approach models the disparity between sensor data onboard a vessel and machine learning predictions. The higher fidelity model was trained with 3000 shots of large wave data, while the lower fidelity model that was used for forecast data was trained with 750 shots of extreme weather data. The 3000 shot model predicts an operational envelope that is much more aligned with the ground truth model than the 750 shot model. Because of this, we can simulate the ASV vessel using the 3000 shot model, with the lower fidelity 750 shot model as the machine learning method an ASV may employ.

The `step` function is also responsible for advancing the weather data by the time step, which in our case is 1 hour. Because the frequency of steps is higher than the nowcast updates, interpolation is used to provide intermediary nowcast weather data. Every six hours, the nowcast weather and its corresponding forecasts are updated. This method of rolling forecasts is similar to the work of marine routing with RL from Latinopoulos et al. [2025].

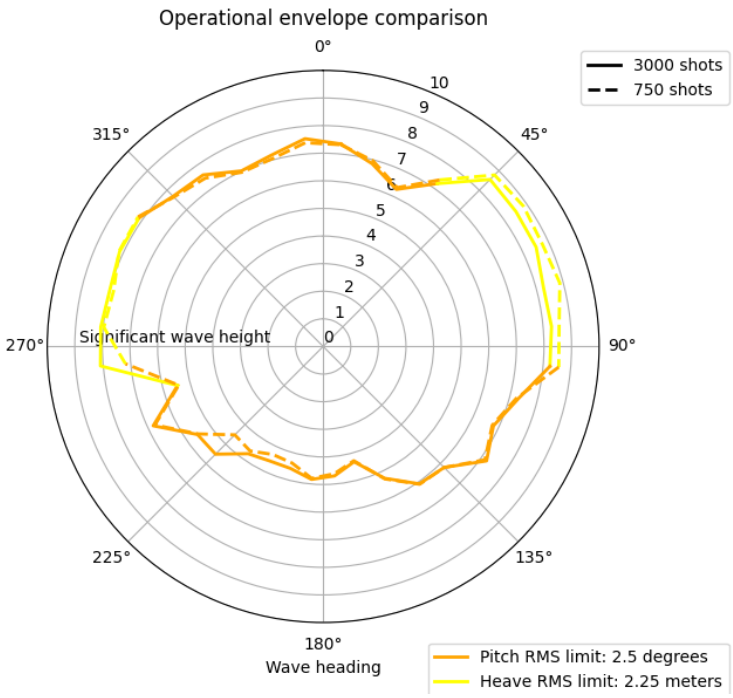


Figure 6.3: Subtle differences between the Operational Envelope's 750 and 3000 shot models are used to represent a vessel (3000), and its imperfect machine learning models (750)

6.2.3.1 Reward Function

After the `step` has been applied, several checks take place to ensure that the simulation can continue and does not need to be reset. The first check is that the vessel did not transit outside of operational bounds. If this check fails, truncation takes place in the environment, resulting in the episode terminating early with a penalty of -1000 being applied. If the vessel takes longer to complete the mission than we have nowcast weather data for, truncation and termination occurs, and a penalty of -1500 is applied. Lastly, if the vessel surpasses its operational limits, termination occurs and a penalty of -1000 is applied. The pitch and heave RMS values of the vessel are calculated using the nowcast weather at the vessel's position with the high fidelity DKL models for pitch and heave that are trained with 3000 shots of extreme weather data to represent the ASV's motions.

Intermediary rewards are given to the agent every time it reaches a waypoint. This is determined by the vessel arriving within 10 nautical miles of the waypoint. Every waypoint it reaches rewards the agent with +1000. Finally, if all waypoints are reached, an additional reward

Result of Step	Reward/Penalty
Transit Out of Bounds	-1000
Time Exceedance	-1500
Surpass Operational Limits	-1000
Reach Waypoint	+1000
Finish Mission Safely	+5000

Table 6.1: Truncation penalties and rewards for reaching waypoints

Weights	Association	Value
w_1	Optimal Speed	25.0
w_2	Risk (Failure Probability)	20.0
w_3	Progress Towards Waypoint	3.0
w_4	Heading Aligned to Waypoint	10.0
w_5	Time Penalty	10.0

Table 6.2: Weights for reward components

of +5000 is added.

The `step` function is also responsible for applying the reward function, which occurs every step, unlike the truncation penalties and waypoint bonuses. The reward function and its weights were found by experimentation and testing different combinations.

$$Reward = w_1 * R_{speed} + w_2 * R_{risk} + w_3 * R_{progress} + w_4 * R_{heading} + w_5 * R_{time} \tag{6.4}$$

Where Eq. 6.5 provides a quadratic reward if the agent's speed is close to the provided optimal speed. In this demonstration, the maximum speed of the vessel is 20 knots, with the optimal speed selected as 18 knots.

$$r_{speed} = \left(\frac{V - V_{opt}}{V_{maax}}\right)^2 \tag{6.5}$$

R_{risk} is defined as the joint probability of failure used for the operational envelope. This penalizes the vessel for placing itself in dangerous scenarios, and prioritizes safe passage and operations.

$R_{progress}$ evaluates the current position's distance to the next waypoint and subtracts it from the previous step's distance to the waypoint. This allows for a positive reward going towards the waypoint, and penalizes the agent if it moves away from the waypoint. Additionally, the reward is larger for bigger steps towards its goal, encouraging faster speeds at headings directed at the waypoint.

$R_{heading}$ provides a reward for steering at the next waypoint, and a penalty for heading away from it. It is a similar reward and penalty to $R_{progress}$ but is focused on pointing at the next waypoint. Even if the speed is zero knots or similar low speed, the heading reward helps the agent be in the right orientation when it is safe to transit again.

R_{time} is responsible for penalizing both slow transit and meandering paths. To do this, after each reset of the environment the route is selected using a greedy approach. The selection process determines routes by selecting whatever waypoint is closest to the vessel, and then repeating this process at the new location until all of the waypoints have been selected. Once the route has been set, the distance is calculated using the geodesic distance (Tools Section 2.1.7), from the start waypoint to the next waypoint, for every leg in the mission. Once the distance of the direct route has been determined, the reasonable minimum time is found by multiplying the distance by the optimum speed of 18 knots. Notably, it is not multiplied by the maximum speed, allowing for some wiggle room in course alteration for the agent that can be made up by traveling faster than the optimal speed. The optimal mission time is added to the start time T_0, and changes every mission due to the randomization of the starting location and waypoint order. Once the datetime in the simulation surpasses the datetime of $T_0 + T_{optimal}$ a penalty of -10.0 is tacked on each step/hour until the mission completes. Prior to the time penalty, a direct route with a far away waypoint would often score worse than an episode with a meandering course where the start waypoint was close to the first waypoint.

6.2.4 `get_obs` Function

The observation construction encodes the risk relevant information the agent can exploit, making it the critical interface where uncertainty aware operational envelopes become actionable for routing.

Although not a mandatory function, `get_obs` or `get_observation` is a function strongly recommended by gym. After the `step` has been implemented, `get_obs` is responsible for gathering information on how the environment changes and providing the agent with the observation information. The observation space array is how the agent knows its current state as well as forecasted weather.

Within the function, the vessel's position, speed, and heading are used to find the distance and decomposed bearing to each of the waypoints in the mission. To incorporate the weather forecast data, the weather grid node locations are iterated through. For each of these locations, the transit time is determined by using the distance between the grid point and the vessel and the speed that has been chosen by the agent during the `step` function. Using the simulation's current time and adding the transit time to the grid point allows for a forecast weather lookup at that location if the vessel moved towards it at a constant speed. If the vessel's speed is zero, or the transit time surpasses the forecast horizon, the weather for that point at the forecast horizon is used. The relative heading between the ship's course to the grid point and the primary wave heading forecast is then calculated. This relative heading, along with the speed determined by `step` and the forecasted significant wave height and peak wave period are used by the DKL models for pitch and heave. The models predict motion RMS as well as uncertainty in the predictions. These predictions and uncertainty are used to generate a overall probability of failure for the vessel in the forecasted weather if it traveled

directly to that grid point at its current speed. The failure probabilities used at each grid node are

$$z_{\Theta,n} = \frac{\Theta_{\lim} - \Theta_n^{\mathrm{RMS}}}{\sigma_{\Theta,n}}, \qquad z_{Z,n} = \frac{Z_{\lim} - Z_n^{\mathrm{RMS}}}{\sigma_{Z,n}}, \tag{6.6}$$

$$\Phi(z) = \tfrac{1}{2}\left[1 + \operatorname{erf}\left(\frac{z}{\sqrt{2}}\right)\right], \tag{6.7}$$

$$p_{\mathrm{ex},\Theta,n} = 1 - \Phi\left(z_{\Theta,n}\right), \tag{6.8}$$

$$p_{\mathrm{ex},Z,n} = 1 - \Phi\left(z_{Z,n}\right), \tag{6.9}$$

$$p_{\mathrm{fail},n} = 1 - \left(1 - p_{\mathrm{ex},\Theta,n}\right)\left(1 - p_{\mathrm{ex},Z,n}\right), \tag{6.10}$$

where $\Theta_{\lim}$ and $Z_{\lim}$ are the pitch and heave safety limits, respectively, and $\Phi(\cdot)$ is the standard normal cumulative distribution function implemented via the error function.

This process is repeated for each of the grid points and then concatenated with the vessel's current condition. This process is further explained in the Appendix, under the pseudocode algorithms 7-8.

6.3 Experimental Setup

The reinforcement learning routing experiments are implemented using the Deep Deterministic Policy Gradient (DDPG) algorithm from Stable Baselines3 together with the custom `VesselEnv` environment described in Section 6.2. The environment couples a kinematic ship motion model, pre trained Deep Kernel Learning (DKL) predictors for pitch and heave, and a nowcast plus forecast wave field constructed with the `Wave_Data_Calc` interface.

6.3.1 Geographical and Temporal Domain

The underlying meteorological dataset provides nowcast and forecast fields on a regular latitude–longitude grid over the larger data domain

$$30^\circ\,\mathrm{N} \le \varphi \le 40^\circ\,\mathrm{N}, \qquad 150^\circ\,\mathrm{E} \le \lambda \le 160^\circ\,\mathrm{E}. \tag{6.11}$$

To avoid edge effects in interpolation, the vessel is restricted to a slightly smaller operational box

$$30.5^\circ\,\mathrm{N} \le \varphi \le 39.5^\circ\,\mathrm{N}, \qquad 150.5^\circ\,\mathrm{E} \le \lambda \le 159.5^\circ\,\mathrm{E}. \tag{6.12}$$

The usable time interval of the dataset is

$$t \in [t_0, t_1], \qquad t_0 = 18\ \text{December 2024 00:00 UTC}, \quad t_1 = 28\ \text{December 2024 18:00 UTC}. \tag{6.13}$$

The simulation uses a fixed integration step of

$$\Delta t = 1\ \text{hour}, \tag{6.14}$$

so that an episode of length N_{steps} corresponds to N_{steps} simulated hours. An upper bound on episode length is enforced as

$$N_{\text{steps}} \le N_{\max}, \qquad N_{\max} = 500, \tag{6.15}$$

which corresponds to at most 500 simulated hours per episode.

6.3.2 Grid Discretization and Observation Layout

Discretization choices control the fidelity of forecast look ahead information and the computational cost of repeated risk evaluation across candidate future positions. Two spatial grids are used within the environment. The first grid is a coarse one degree grid used to construct the risk based observation features. Its spacing is

$$\Delta\varphi_{\text{grid}} = 1^\circ, \qquad \Delta\lambda_{\text{grid}} = 1^\circ, \tag{6.16}$$

which yields a set of node positions

$$\{(\varphi_n, \lambda_n)\}_{n=1}^{N_{\text{grid}}} \tag{6.17}$$

with

$$N_{\text{grid}} = \left(\frac{39.5 - 30.5}{1} + 1\right)\left(\frac{159.5 - 150.5}{1} + 1\right) = 10 \times 10 = 100. \tag{6.18}$$

These N_{grid} nodes are used exclusively to evaluate the forecast based probability of failure that enters the observation vector.

A second, finer plotting grid is used only for visualization of the wave and risk fields. This grid has spacing

$$\Delta\varphi_{\text{plot}} = 0.25^\circ, \qquad \Delta\lambda_{\text{plot}} = 0.25^\circ, \tag{6.19}$$

and is used to construct two dimensional fields of significant wave height and wave direction for the summary plots and videos shown in the results section.

6.3.3 Normalization of State and Mission Geometry

Normalization is required to balance heterogeneous state components so that learning dynamics are not dominated by arbitrary unit scales in time, distance, and kinematic variables. `get_obs` is normalized to a compact numerical range before being passed to the actor and critic. The center of the operational domain is

$$\varphi_c = \frac{\varphi_{\min}^{\text{op}} + \varphi_{\max}^{\text{op}}}{2}, \qquad \lambda_c = \frac{\lambda_{\min}^{\text{op}} + \lambda_{\max}^{\text{op}}}{2}, \tag{6.20}$$

with half ranges

$$\Delta\varphi = \varphi_{\max}^{\text{op}} - \varphi_{\min}^{\text{op}}, \qquad \Delta\lambda = \lambda_{\max}^{\text{op}} - \lambda_{\min}^{\text{op}}. \tag{6.21}$$

The diagonal distance of the operational box in nautical miles,

$$d_{\max} = d\big((\varphi_{\min}^{\text{op}}, \lambda_{\min}^{\text{op}}), (\varphi_{\max}^{\text{op}}, \lambda_{\max}^{\text{op}})\big), \tag{6.22}$$

is used as the normalization scale for waypoint distances. The time normalization horizon is chosen as

$$T_{\text{norm}} = \min\left(168\ \text{h}, \frac{t_1 - t_0}{3600\ \text{s/h}}\right), \tag{6.23}$$

so that the normalized time variable

$$t_{\text{norm}} = \text{clip}\left(\frac{t - t_0}{T_{\text{norm}}}, 0, 1\right) \tag{6.24}$$

remains in $[0, 1]$ for the entire dataset. Latitude and longitude of the vessel are normalized as

$$\varphi_{\text{norm}} = \text{clip}\left(\frac{\varphi - \varphi_c}{\Delta\varphi/2}, 0, 1\right), \qquad \lambda_{\text{norm}} = \text{clip}\left(\frac{\lambda - \lambda_c}{\Delta\lambda/2}, 0, 1\right), \tag{6.25}$$

and the speed is normalized as

$$V_{\text{norm}} = \text{clip}\left(\frac{V}{V_{\max}}, 0, 1\right), \qquad V_{\max} = 20\ \text{knots}. \tag{6.26}$$

At most K mission waypoints are explicitly encoded in the observation. In the implementation

$$K = 4. \tag{6.27}$$

For each encoded waypoint j, the normalized distance and bearing features are

$$d_j^* = \text{clip}\left(\frac{d_j}{d_{\max}}, 0, 1\right), \qquad \alpha_j^{\text{rad}} = \alpha_j \frac{\pi}{180}, \tag{6.28}$$

and the waypoint feature vector is

$$\mathbf{w}_j = \begin{bmatrix} d_j^* \\ \sin\alpha_j^{\text{rad}} \\ \cos\alpha_j^{\text{rad}} \\ \mathbb{1}_{\text{visited},j} \end{bmatrix}. \tag{6.29}$$

6.3.4 Observation Space Dimension

The observation dimension formalizes the information bandwidth available to the policy and clarifies how much of the forecast risk field is exposed for decision making. The observation vector is constructed by concatenating base kinematic features, waypoint features, and a scalar risk feature per grid node. With the base length equal to six,

$$n_{\text{base}} = 6, \tag{6.30}$$

the number of waypoint features equal to

$$n_{\text{wp}} = 4K = 16, \tag{6.31}$$

and one scalar feature per grid node

$$n_{\text{node}} = N_{\text{grid}} = 100, \tag{6.32}$$

the total observation dimension is

$$n_{\text{obs}} = n_{\text{base}} + n_{\text{wp}} + n_{\text{node}} = 6 + 16 + 100 = 122. \tag{6.33}$$

The observation space is therefore a box

$$\mathcal{S} = \{s \in \mathbb{R}^{n_{\text{obs}}} : \ell_i \le s_i \le u_i,\ i = 1, \ldots, n_{\text{obs}}\}, \tag{6.34}$$

where the lower and upper bounds ℓ_i and u_i follow the normalization scheme in Section 6.2.4. Time and speed components, latitude and longitude components, waypoint flags, and node risk features all lie in $[0, 1]$.

6.3.5 Action Space and Mission Randomization

Action definitions and mission randomization jointly determine what behaviors the agent can express and what variability it must handle to be considered robust. The continuous action space of the DDPG agent is a three dimensional box

$$\mathcal{A} = [-1, 1]^3. \tag{6.35}$$

At time t, the actor outputs

$$a_t = \begin{bmatrix} a_t^{(0)} \\ a_t^{(1)} \\ a_t^{(2)} \end{bmatrix} \in \mathcal{A}. \tag{6.36}$$

The first two components are interpreted as unnormalized sine and cosine of the vessel heading. After normalization

$$s_t = \frac{a_t^{(0)}}{\sqrt{(a_t^{(0)})^2 + (a_t^{(1)})^2 + \varepsilon}}, \qquad c_t = \frac{a_t^{(1)}}{\sqrt{(a_t^{(0)})^2 + (a_t^{(1)})^2 + \varepsilon}}, \qquad \varepsilon = 10^{-8}, \tag{6.37}$$

the commanded heading in degrees is

$$\psi_t = \left(\operatorname{atan2}(s_t, c_t) \, \frac{180}{\pi} \right) \bmod 360. \tag{6.38}$$

At each reset the starting position of the vessel is sampled uniformly within the operational bounds. Denoting the operational latitude and longitude intervals by $[\varphi_{\min}^{\text{op}}, \varphi_{\max}^{\text{op}}]$ and $[\lambda_{\min}^{\text{op}}, \lambda_{\max}^{\text{op}}]$, the starting position is

$$\varphi_0 \sim \mathcal{U}(\varphi_{\min}^{\text{op}}, \varphi_{\max}^{\text{op}}), \qquad \lambda_0 \sim \mathcal{U}(\lambda_{\min}^{\text{op}}, \lambda_{\max}^{\text{op}}). \tag{6.39}$$

The final destination waypoint is kept fixed, while intermediate waypoint order determined by a greedy nearest neighbor heuristic used to construct a nominal route for the time penalty, as described

in Section 6.2.3.1. A minimum start–goal separation is enforced when the goal is randomized so that

$$d\big((\varphi_0, \lambda_0), (\varphi_{\text{goal}}, \lambda_{\text{goal}})\big) \geq d_{\min}^{\text{goal}}, \tag{6.40}$$

with $d_{\min}^{\text{goal}} = 25$ nautical miles in the implementation. Waypoint arrival is detected when the distance between the vessel and waypoint j satisfies

$$d_j(t) \leq R_{\text{goal}}, \qquad R_{\text{goal}} = 10 \text{ nautical miles}. \tag{6.41}$$

Upon satisfaction of this condition, the waypoint is marked as visited and the waypoint bonus is added to the reward.

6.3.6 Exploration Noise

Exploration strategy influences whether the agent discovers safer alternative routes that may initially appear suboptimal under sparse rewards. During training, exploration in action space is produced by an Ornstein–Uhlenbeck process applied to each component of the action. Let

$$N_t = \begin{bmatrix} N_t^{(0)} \\ N_t^{(1)} \\ N_t^{(2)} \end{bmatrix} \tag{6.42}$$

be the noise vector. Its discrete time dynamics are given by

$$N_{t+1} = N_t + \theta\,(\mu_N - N_t)\,\Delta t + \sigma\sqrt{\Delta t}\,\xi_t, \tag{6.43}$$

where $\xi_t \in \mathbb{R}^3$ has independent standard normal components, θ is the mean reversion rate, μ_N is the long term mean, and σ controls the instantaneous standard deviation [Uhlenbeck and Ornstein,

1930]. In this study

$$\theta = 0.15, \qquad \mu_N = \mathbf{0}, \qquad \sigma = \begin{bmatrix} 0.4 \\ 0.4 \\ 0.4 \end{bmatrix}. \tag{6.44}$$

The action applied to the environment during training is

$$a_t^{\text{applied}} = a_t + N_t. \tag{6.45}$$

6.3.7 Training Configuration

Training configuration parameters define sample efficiency and convergence behavior for continuous control in a stochastic, forecast driven environment. Training is performed using the `DDPG` implementation in Stable Baselines3 with a multilayer perceptron policy `MlpPolicy`. The main hyperparameters are

$$\alpha = 5 \times 10^{-4}, \qquad N_{\text{buffer}} = 2 \times 10^5, \qquad N_{\text{batch}} = 256, \tag{6.46}$$

where α is the learning rate, N_{buffer} is the replay buffer capacity, and N_{batch} is the minibatch size. Learning begins after the first transition is collected and proceeds for a total of

$$T_{\text{train}} = 3 \times 10^5 \tag{6.47}$$

environment time steps.

The environment used for training is wrapped in a `Monitor` to record episodic statistics and to allow Stable Baselines3 to log rewards and episode lengths. Both environments share the same meteorological dataset and DKL models but evolve independently with different random seeds.

A checkpoint callback saves the full agent state every 10^3 time steps, including the replay buffer. An evaluation callback is executed every $2 * 10^3$ time steps. At each evaluation, several

episodes are rolled out with the deterministic policy and the mean episodic return

$$J_m^{\text{eval}} \tag{6.48}$$

is computed. The model with the highest observed value of J_m^{eval} is stored as the current best model.

An early stopping mechanism based on the evaluation return terminates training when no improvement is observed over a prescribed number of evaluations. Denoting by J_k^{best} the best evaluation return up to evaluation index k, training is stopped after ten valuations without improvement.

6.3.8 Diagnostics and Visualization

Diagnostics connect learned policy behavior back to physical interpretation by revealing whether routing decisions align with forecast risk gradients and operational envelope constraints. To aid interpretation of the learned policies, the environment records the full trajectory of each episode, including vessel latitude, longitude, heading, speed and the simulation time. After each episode, a summary plot can be generated that overlays:

1. The operational bounds
2. The significant wave height field $H_s(\varphi, \lambda, t)$ at the final time
3. The vessel route $(\varphi(t), \lambda(t))$ with hourly markers
4. Heading vectors at each recorded time step
5. Mission waypoints and the starting position.

In addition, for selected episodes an animation is produced in MP4 format. For frame index i, at time t_i, the animation overlays:

1. The nowcast wave field $H_s(\varphi, \lambda, t_i)$ on the plotting grid
2. The vessel route up to time t_i

3. The instantaneous heading vector at the vessel position

4. The spatial distribution of the node failure probabilities

$$p_{\text{fail},n}(t_i), \qquad n = 1, \ldots, N_{\text{grid}}, \tag{6.49}$$

   rendered as colored markers on the forecast node grid

5. A textual panel summarizing the current latitude and longitude, local wave conditions (H_s, T_p), relative wave heading, and vessel speed

The node failure probabilities used in the animation are computed consistently with the forecast based node features in the observation. For each frame time t_i and for each grid node n, the estimated time of arrival t_n^{ETA} is calculated, the forecast wave parameters are queried, the DKL models are evaluated, and the probability of failure $p_{\text{fail},n}(t_i)$ is obtained through the same Gaussian exceedance formulation as in Section 6.2.4.

6.3.9 Testing Protocol

A testing protocol is defined to ensure that reported performance reflects policy quality under controlled evaluation conditions rather than sampling variability. For testing, multiple missions are executed using the trained policy, with the number of mission waypoints ranging from one to three. Each mission uses the same meteorological dataset and operational bounds as during training. The starting position of the vessel in each mission is randomized within the operational box according to the uniform sampling scheme described above, and the order of intermediate waypoints is determined by the nearest neighbor heuristic used in the time penalty calculation.

For each test mission, the environment logs whether the vessel:

1. Completes the mission within the dataset time horizon without exceeding pitch or heave limits

2. Exits the operational bounds before mission completion

3. Exceeds the available nowcast time horizon

4. Violates the operational envelope limits

These trajectories and termination outcomes are used to assess the robustness of the learned DDPG policy to variations in initial condition and mission geometry, and to quantify how effectively the agent exploits the forecast risk field to maintain low predicted probability of failure while completing its routing objectives.

6.4 Results and Discussion

The preceding sections defined a forecast conditioned routing environment, the associated observation and action parameterizations, and a reward structure that couples mission progress to probabilistic motion risk. This section reports the resulting policy behavior for both a single waypoint task and a multiple waypoint task, with emphasis placed on interpretable connections between the forecast derived risk representation provided to the agent and the resulting route selection strategies. Because the reinforcement learning component is intended as an end to end demonstration of the dissertation framework, the discussion is structured to highlight not only task success, but also the extent to which learned behaviors align with seakeeping based intuition and the operational envelope concepts developed in earlier chapters.

Training was performed on a campus workstation equipped with an NVIDIA RTX 2000 Ada GPU. Both the single waypoint and multiple waypoint agents were trained to completion in under 24 hours. This runtime includes repeated environment rollouts with randomized initial conditions, periodic evaluation of the deterministic policy, and early stopping based on the evaluation return. The reported routes are generated using the best checkpoint as identified by the evaluation callback, ensuring that qualitative conclusions are drawn from the highest performing policy observed during training rather than from the final training iterate.

Waypoint Number	Latitude (deg)	Longitude (deg)
Waypoint 1	38.0	158.0
Waypoint 2	34.0	155.0
Waypoint 3	32.5	152.0

Table 6.3: Coordinates of RL mission waypoints

The mission geometries are defined by the three waypoints listed in Table 6.3. In all experiments, success is defined operationally as completing the required waypoint visitation objective within the available meteorological time horizon while remaining inside the geographic operational bounds and avoiding exceedance of the pitch and heave operational limits enforced by the environment.

6.4.1 Single Waypoint Mission

The single waypoint mission uses Waypoint 1 as a fixed terminal objective located at $(38.0, 158.0)$. The left column of Figure 6.4 summarizes representative end of mission trajectories for the best performing policy under multiple randomized starting locations. Using the best model, ten independent missions were executed and all completed safely; three representative trajectories are included in the figure.

Several consistent behaviors emerge across these rollouts. First, the policy exhibits an early phase in which progress is deliberately structured to exploit benign conditions at the beginning of the available weather window. Rather than attempting to minimize geodesic distance at every step, the agent preferentially repositions toward a corridor near the diagonal of the operational domain. This behavior is consistent with the forecast conditioned state representation. The observation includes a spatial field of risk values computed for reachable forecast nodes, which exposes to the policy a forward looking indication of where high risk is expected to develop as sea state severity increases.

Once positioned near the central corridor, the policy tends to maintain relative headings that correspond to beam seas as the wave field builds later in the mission. This preference is physically

interpretable. For the considered hull and operational limits, beam seas reduce pitch response relative to head seas, and therefore reduce the joint probability of exceeding pitch and heave limits. The result is a route that is slightly less direct early in the mission but that remains inside a low risk region of the operational envelope as conditions intensify. The policy is not merely reacting to local wave height, but is using forecasted weather with heading dependent motion response predictions to avoid entering states associated with a high probability of failure.

A third feature is the manner in which speed is modulated. The reward structure includes explicit incentives for efficient transit, including a speed shaping term and progress related terms, while simultaneously penalizing risk. The resulting policy uses speed as a secondary control lever. Speed is increased in segments where the forecast risk field indicates acceptable margins, and reduced when the projected probability of failure rises due to an unfavorable combination of wave height and relative wave direction. This behavior supports the intended design goal of the reward formulation, namely that the agent should learn to trade progress and efficiency against a probabilistic representation of motion limit state risk rather than defaulting to a conservative low speed policy everywhere.

6.4.2 Multiple Waypoint Mission

Multiple waypoint missions test whether the policy can manage longer horizon planning and cumulative risk while satisfying sequential visitation constraints. The multiple waypoint mission uses all three waypoint locations from Table 6.3. In contrast to the single waypoint case, the agent must balance immediate progress toward the next waypoint against the downstream impact of route choices on later legs of the mission. This structure is designed to probe whether the forecast based risk representation can support anticipatory decisions that remain beneficial when evaluated over a longer planning horizon. Additionally, the longer mission duration means that the agent has to route through larger weather systems that develops later in the collected dataset.

While the left column of the subplots in Figure 6.4 shows three individual single waypoint missions, the right column shows a single multi-waypoint mission at different within the mission.

This was done to highlight the two methods of visualization developed. The right column of Figure 6.4 presents a representative multiple waypoint rollout at several times during the mission and highlights the visualization strategy used to interpret policy behavior. Two distinct time scales are presented in these screenshots. The background wave field corresponds to the nowcast conditions at the current simulation time, whereas the overlaid grid nodes represent forecast conditioned risk values. Each node color is computed from the joint probability of failure that would be incurred if the vessel were to maintain its current speed and transit to that node, with the forecast queried at the estimated time of arrival. As a result, the node field can reveal elevated risk even in locations where the instantaneous nowcast appears moderate, because the forecast at the time of arrival may be significantly more severe.

This distinction becomes particularly important near the second visited waypoint in the demonstrated multiple waypoint mission. In the middle screenshot of Figure 6.4, nodes with a higher likelihood of failure cluster near the vicinity of the second waypoint. The underlying physical driver is a change in the primary wave direction, which clocks from a southerly direction toward a northeasterly direction as the mission progresses. If the vessel were to reverse course and revisit that region, the combination of travel time and evolving wave direction would cause the vessel to encounter head seas upon arrival, rather than the more benign relative headings experienced earlier. In addition, the significant wave height builds as the mission progresses, with the final screenshot showing that significant wave height increases over the same period. Under head seas in larger waves and at a transit speed of 12.48 knots, some nodes predict a probability of failure of approximately 70%. This forecast derived risk gradient provides a clear rationale for why the policy avoids certain spatial corridors even when they appear feasible under the instantaneous nowcast.

A comparison between the single waypoint and multiple waypoint policies suggests there is not a unique mechanism for achieving safe mission completion within the environment constraints. Both policies begin by steering into head seas early in the mission when sea states are calm. However, the single waypoint policy tends to proceed more directly at lower speeds, using speed reduction as the primary means to manage pitch response when relative heading becomes unfavorable. By

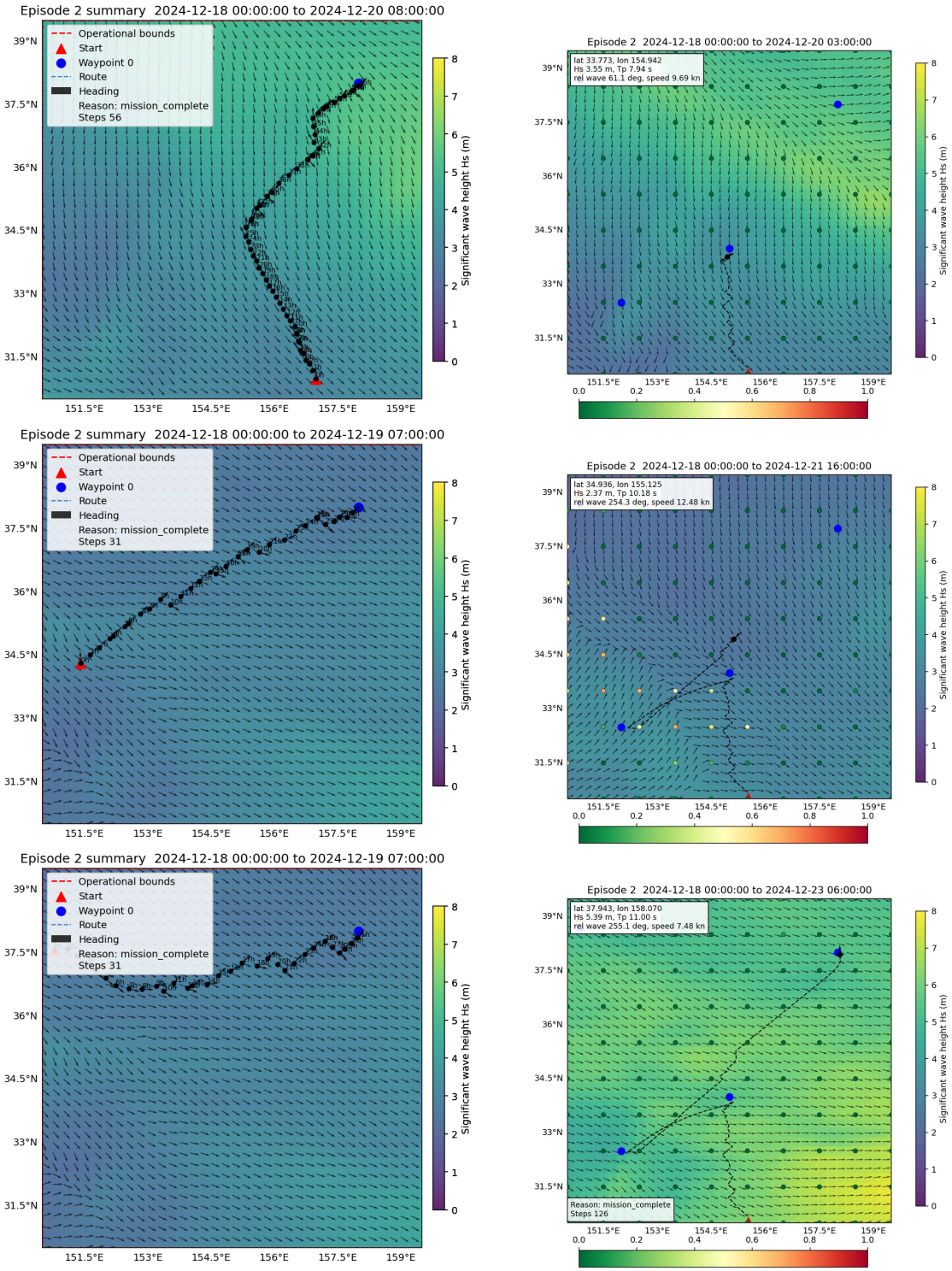


Figure 6.4: The left column displays the end of mission plot for a single waypoint mission with randomized starting locations. The right column shows intermediary screenshots of a single mission video, which includes the failure probability nodes changing as weather develops and speeds are altered.

contrast, the multiple waypoint policy tends to maintain higher transit speeds while using course modulation, tacking through head seas to avoid sustained exposure to wave conditions that result in high pitch motions. The learned policy is shaped by both the incentives for efficiency and the strong penalties associated with elevated risk and operational limit violation. The multiple waypoint mission, which has a longer horizon and sequential constraints, provides more opportunities for the agent to realize that modest increases in path length can be offset by higher speed in low risk segments, provided that risk is controlled through heading selection.

The results above demonstrate that a deep reinforcement learning policy can successfully incorporate probabilistic operational envelope information into routing decisions utilizing weather forecasts. The reinforcement learning demonstration in this chapter provides a mission level mechanism for using the envelopes developed in Chapter 4. The routing agent does not require explicit enumeration of safe and unsafe headings and speeds. Instead, it receives a forecast conditioned risk field derived from the same probabilistic limit state framework used to define the operational envelope, and it learns a continuous control policy that selects headings and speeds to trade mission objectives against predicted failure probability. The routing behavior observed here suggests that risk aware reinforcement learning can provide a practical interface between physics informed operational envelopes and mission execution.

CHAPTER 7

Conclusions and Future Work

Autonomous surface vessels are increasingly expected to execute long duration missions with limited or no human supervision. In this operating regime, safety depends on the ability to quantify and manage the risk of exceeding limit states under changing sea states, imperfect forecasts, and evolving vessel condition. Operational envelopes derived solely from predeployment analysis can become unreliable when the true vessel response differs from engineering assumptions, when nonlinear effects become important in heavy weather, or when the vessel has accumulated damage that changes its effective safety margins. The central objective of this dissertation was to develop an adaptive framework that updates motion based limits during operation and to demonstrate how these limits can be embedded in mission level decision making.

The work began by establishing a consistent modeling and simulation foundation for a notional monohull based on the U.S. Navy Vanguard unmanned surface vehicle and by generating response data across a wide range of headings, speeds, and sea states. Motions and loads were evaluated using both closed form seakeeping methods and time domain simulation with nonlinear Froude Krylov excitation, enabling a controlled study of how modeling fidelity affects predicted operational limits. This foundation supported the development of learning based surrogates that can be updated in situ and that expose uncertainty in their predictions, which is essential when operational limits are used as decision boundaries. This objective aligns with the research scope defined in Chapter 1, by addressing adaptive surrogate modeling for seakeeping limit states, uncertainty quantification and management near operational boundaries, and integration of these elements into autonomous

planning and routing under realistic forecast driven environmental forcing.

A first contribution was the use of Model Agnostic Meta Learning, abbreviated as MAML, to rapidly adapt surrogate motion models from predeployment predictions to post-deployment measurements. The results showed that a model trained on simplified closed form targets can exhibit substantial bias when evaluated against higher fidelity time domain responses, with a severe degradation in accuracy when nonlinear excitation is present. Few shot adaptation reduced this mismatch quickly, with marked improvements after limited data collection. This demonstrated that rapid transfer of learning is a viable mechanism for phasing out predeployment bias and for updating operational envelopes as the vessel encounters conditions that deviate from prior assumptions.

A second contribution was the identification and characterization of catastrophic forgetting as a key failure mode for online learning in marine operations. When successive missions are dominated by similar calm water conditions, MAML's meta updates drift toward parameters that are well tuned for the recent tasks but that lose competency on rare and safety critical heavy weather cases. In the presented experiments, performance on large wave conditions degraded substantially after repeated calm missions, with large wave prediction error increasing 121% relative to the baseline by completion of three missions. A simple experience injection strategy, in which a small fraction of extreme wave samples is reintroduced during calm water tasks, mitigated this behavior and improved retention, yielding approximately 27.7% higher accuracy on the large wave test set than the standard approach after three successive missions. These findings clarify that adaptation speed alone is insufficient for safety critical autonomy.

The dissertation then focused on uncertainty aware motion modeling, motivated by the need to represent confidence when translating predictions into operational limits. Bayesian neural networks, deep evidential regression, and deep kernel learning were evaluated as candidates for predicting response distributions rather than only point estimates. The comparative study showed that these approaches differ not only in mean accuracy but also in calibration and stability when exposed to shifting conditions. DKL emerged as a practical choice due to favorable calibration, competitive error metrics across shot counts, and computational efficiency that supports repeated

evaluation across large state grids. This selection was consequential because the downstream operational envelope and routing components depend on reliable predictive dispersion, not only on mean response.

A further contribution was the development of post hoc uncertainty scaling mechanisms that allow a probabilistic operational envelope to remain conservative when the model is asked to extrapolate to rarely seen conditions or when operational history suggests heightened risk. Out of distribution scaling was implemented using Mahalanobis distance in the standardized input space derived from routine weather training data. This mechanism inflates predictive standard deviation smoothly for rare or extrapolative combinations of wave height, period, speed, and heading while leaving familiar regions largely unchanged. Complementing this, damage based scaling was introduced to incorporate post mission feedback by locally increasing uncertainty near recorded damaging conditions. In the case study, a representative damage condition was specified by a significant wave height of 7.0 m, a peak period of 10.0 s, a vessel speed of 13 m per s, and a wave heading of 180 degrees, and feature sensitivity analysis showed that significant wave height was the dominant driver of both heave and pitch response. Critically, both scaling mechanisms preserve the learned mean response and modify only the predictive variance, allowing safety margins to be adjusted without retraining. The combined effect is an operational envelope that contracts in extreme seas and in neighborhoods of known hazardous conditions, producing an interpretable and computationally efficient safety factor that is aligned with both data driven learning and operational risk management.

Finally, the dissertation demonstrated how a probabilistic operational envelope can inform mission level autonomy through reinforcement learning based routing. A custom environment coupled vessel kinematics, nowcast and forecast wave fields, and DKL based motion predictions for pitch and heave. Deep Deterministic Policy Gradient was used to control continuous speed and heading, and the reward function included a dedicated risk term defined as the joint probability of exceeding motion limits implied by the operational envelope. This formulation shifts routing from heuristic avoidance of high waves to explicit minimization of predicted failure probability while

maintaining progress toward mission waypoints. The demonstration used a nowcast and forecast dataset spanning 18 December 2024 through 28 December 2024, with operational limits set to a pitch RMS threshold of 4.0 degrees and a heave RMS threshold of 2.75 m. By augmenting the agent observation with forecasted node wise failure probabilities, the learned policy became sensitive to both instantaneous conditions and forecast evolution along candidate trajectories, enabling risk aware tradeoffs between transit time, speed, and safety.

Overall, this dissertation contributes an integrated framework for adaptive operational envelope management in autonomous surface vessels. The framework links physics based simulation, rapid online adaptation, uncertainty aware regression, structured uncertainty scaling informed by distribution shift and damage history, and risk aware routing that uses probability of failure directly in decision making. The results support a practical view of autonomy in which the vessel continuously updates not only its estimate of the environment but also its estimate of its own limitations and confidence, and then uses this information to select actions that remain within defensible safety margins.

Several limitations and assumptions were made while developing this work. The analysis is based primarily on simulated response data, and the operational feedback mechanism is represented through recorded damage conditions rather than direct structural health measurements. The probabilistic envelope construction relies on distributional assumptions for response modeling and on simplified combination of limit states when computing joint failure probability. In addition, the routing demonstration focuses on motion based constraints in open ocean conditions and does not include the full set of real world considerations such as traffic, collision avoidance, or actuator level constraints beyond smoothness of control. Within these bounds, the dissertation establishes a cohesive methodology for embedding adaptive, uncertainty aware operational envelopes into autonomous decision making and provides evidence that such integration can improve the safety relevance of learning enabled maritime autonomy. The present operational envelopes focus on RMS pitch and heave. Slamming frequency, represented in Chapter 3 as slams per hour, exhibited substantially greater prediction difficulty and was therefore excluded from envelope construction

in future chapters, making slamming and associated impact load limit states a natural priority for future work.

7.1 Future Work

Future work builds on the contributions of Chapters 3 through 6, with the goal of improving retention during continual adaptation, expanding operationally relevant limit states, maintaining calibrated uncertainty under distribution shift and degradation, and enabling risk aware routing over longer missions.

One significant extension of this work is validation in full scale or hardware relevant experiments. Field trials with an instrumented vessel would then allow testing of online adaptation, uncertainty scaling, and routing under realistic sensor noise, evolving sea states, and operational constraints. This constrains would present an increased challenge for rapid adaptation, but likely provide more realistic uncertainty bands for all of the stochastic methods explored in this work. To test this, the rapid adaptation experimentation from Chapter 3 would replace the second phase of training (simulation) with real-world data.

The operational envelope would be more expressive with additional topics. One such use topic would be roll, which was left out of our dataset because our hullform used in the simulation did not include the necessary appendages for a realistic value. Another possibility would be the vertical bending moment of the vessel, which was initially explored with Jensen's closed-from solutions, but not used afterwards due to it being absent from the simulation toolkit.

7.1.1 Chapter 3: Rapid Online Adaptation with Retention

Chapter 3 showed that online adaptation can correct predeployment model mismatch, but sequential updates can erode performance in rare, safety critical heavy weather conditions. A near term priority is to incorporate continual learning mechanisms that preserve competence in extreme sea states while still enabling rapid adaptation.

The most practical extensions are replay strategies that deliberately retain rare and high consequence conditions, coupled with lightweight regularization that limits drift in safety critical regions of the input space. In parallel, update scheduling should be made explicit through policies that control when updates occur and how strongly they are applied as conditions evolve. Operational relevance should be strengthened through validation on field data, including sensor quality changes, loading variation, and gradual performance degradation from fouling or component aging.

Another extension is geographic transfer learning across distinct wave climates, operating areas, and seasons. The current framework is demonstrated using a specific set of hindcast, nowcast, and forecast products in a defined region, and the learned surrogates and uncertainty scaling strategies are conditioned on the resulting distribution of sea states. In operational settings, an autonomous surface vessel may transition between regions whose wave spectra, storm statistics, current driven effects, and prevailing headings differ substantially. A practical path forward is to treat each region or season as a related task family and to learn transferable representations that can be adapted rapidly when the vessel enters a new environment. Meta learning across region specific task distributions should be explored. Additionally, the training data is not temporally correlated, but randomly sampled using the netCDF4 file. Temporally correlated shots of data will prove an increased challenge for model forgetfullness, but be more representative of an actual ASV mission learning in an online manner.

Combating forgetfullness can be reworked to further prioritize unique samples and rare data that should not be forgotten. To do this, I propose integrating the Mahalanobis distance metric with the training dataset wand exploring two new methods. The first would include a cutoff threshold τ, with samples smaller than the threshold not being recycled and injected into tasks. For samples greater than the cutoff, the probability of recycling into current tasks would be dependent on the Mahalanobis d^2 value, with larger and more unique samples having an increased chance in being retrained on.

The second route to combat forgetfullness would use the Mahalanobis based distance metric in a binned histogram approach. When a new sample comes in, it replaces the old sample in that

bin. This would preserve unique and large d^2 values that rarely get updated, and swap out routine weather samples that the vessel sees often.

7.1.2 Chapter 4: Probabilistic Limit State Prediction and Rare Event Responses

Chapter 4 evaluated uncertainty aware models for RMS motion limit states. The most impactful extension is to predict additional responses that directly influence damage risk and abort decisions, including slamming related indicators, acceleration criteria, and key load measures such as vertical bending moment.

Since operational decisions depend on concurrent risk across responses, multioutput probabilistic modeling should be pursued to support joint exceedance estimation and dependence representation. Evaluation should emphasize performance under realistic distribution shifts, including geography and season changes and configuration variation, and should include sequential learning across missions rather than isolated train test splits.

Of all the models explored, MAML was the most that 'just worked'. It's flexibility and adaptability showed the greatest promise, but its deterministic nature made it unsuitable for risk-aware routing. Because of this, future work should explore ways to include uncertainty with MAML. Because it is model agnostic, this could entail the base structure being a gaussian process or Bayesian neural network.

7.1.3 Chapter 5: Uncertainty Scaling Informed by Novelty and Structural Condition

Chapter 5 introduced uncertainty scaling based on novelty and damage history. However, this chapter assumed that it was known what conditions damaged the vessel. In reality, connecting sensor data to determine faults is significantly more challenging, especially without human crew onboard. The most feasible and operationally useful next step is to connect damage informed scaling to

structural health monitoring outputs, enabling automated margins driven by measured degradation indicators such as strain features, vibration signatures, and fatigue accumulation metrics.

To support realistic operational histories, scaling should accommodate multiple incidents through simple aggregation and time decay so that inflation reflects persistence under repairs and inspections. The impact of scaling should be quantified within an end-to-end decision pipeline by measuring resulting conservatism, mission completion, and violation frequency under plausible hyperparameter settings.

7.1.4 Chapter 6: Risk Aware Routing Under Forecast Seas

Chapter 6 demonstrated risk aware routing with forecast driven operational envelopes and continuous control. The most important extension is to scale to longer missions with richer objectives by incorporating energy costs alongside time and risk, using propulsion and load models with fuel constraints.

Policy reliability should be improved by comparing continuous control methods with stronger stability and constraint handling, and by tuning reward weights using systematic search to avoid unrealistic behaviors. Forecast uncertainty should be treated explicitly by combining higher uncertainty to predictions further out temporally.

Other continuous action policies should be explored, and the resulting weights and penalties for the reward function should be optimized through a process such as a genetic algorithm, which was not performed here due to computational cost.

The online mission routing can be used as a framework for true online learning. Previously, the training involved random sampling from training distributions of hindcast data, rather than temporally correlated shots that are less varied. The developed framework sets the groundwork for this training, by integrating nowcast weather and vessel kinematics, with the entire globes marine weather available for several months.

APPENDIX A

Reinforcement Learning Observation Pseudocode

Algorithm 7 `get_obs`: Construction of the observation vector

Read simulation time t_{cur}, start time t_0, vessel latitude φ, longitude λ, speed V, and heading ψ.

Compute elapsed time $\Delta t = (t_{\text{cur}} - t_0)$ and normalized kinematics using (**??**) to (**??**).

Compute $\psi^{\text{rad}} \leftarrow \big((\psi + 360^\circ) \bmod 360^\circ\big)\pi/180$.

for each destination waypoint i **do**

    Compute geodesic distance d_i and bearing α_i from (φ, λ).

    Wrap bearing $\alpha_i \leftarrow (\alpha_i + 360^\circ) \bmod 360^\circ$.

end for

Select the persistent active waypoint index i^* from $\{d_i\}$ as described in Section 6.2.4.

Construct an ordered index list $\mathcal{I}$ of length K:

    start with i^* (if defined), then append remaining unvisited indices by increasing d_i, and pad with `None`.

Initialize waypoint feature list $\mathbf{f}^{\text{wp}} \leftarrow [\ ]$.

for each $j \in \mathcal{I}$ **do**

    if j is `None` **then**

        Append $[0, 0, 0, 0]$ to $\mathbf{f}^{\text{wp}}$.

    else

        $d_j^* \leftarrow \text{clip}(d_j/d_{\max}, 0, 1)$; if waypoint j is visited then set $d_j^* \leftarrow 0$.

        $\alpha_j^{\text{rad}} \leftarrow \alpha_j \pi/180$ and $v_j \leftarrow \mathbb{I}\{\text{waypoint } j \text{ visited}\}$.

        Append $\left[d_j^*, \sin \alpha_j^{\text{rad}}, \cos \alpha_j^{\text{rad}}, v_j\right]$ to $\mathbf{f}^{\text{wp}}$.

    end if

end for

Choose ETA speed $V_\eta \leftarrow \max(V, 0)$ if the current speed is used for ETA, otherwise $V_\eta \leftarrow V_{\text{ETA}}$.

Compute remaining forecast horizon $T_{\text{horizon}} \leftarrow \max(0, (t_1 - t_{\text{cur}})/3600)$ hours, where t_1 is the last reliable forecast time.

Initialize node feature array $\mathbf{f}^{\text{node}} \in \mathbb{R}^{N_{\text{grid}}}$ with zeros.

Algorithm 8 `get_obs` (continued)

for each grid node $n = 1, \ldots, N_{\text{grid}}$ at (φ_n, λ_n) **do**

  Compute geodesic distance d_n and course to node bearing β_n.

  if $V_\eta \le 0$ **then**

    Set $t_n^{\text{ETA}} \leftarrow t_1$.

  else

    $\tau_n \leftarrow d_n / \max(V_\eta, V_{\min})$.

    Set $t_n^{\text{ETA}} \leftarrow t_1$ if τ_n is nonfinite or $\tau_n > T_{\text{horizon}}$, otherwise set $t_n^{\text{ETA}} \leftarrow t_{\text{cur}} + \tau_n$.

  end if

  Query the wave model to obtain $(H_{s,n}, \beta_n^{\text{dir}}, T_{p,n}) \leftarrow \textsc{WaveModel}(\varphi_n, \lambda_n, t_{\text{cur}}, t_n^{\text{ETA}})$.

  Convert to wave heading $\beta_n^{\text{wave}} \leftarrow (\beta_n^{\text{dir}} + 180^\circ) \bmod 360^\circ$.

  Compute relative heading $\phi_n^{\text{rel}} \leftarrow (\beta_n - \beta_n^{\text{wave}}) \bmod 360^\circ$.

  Predict motions in forecast mode, $\left(\Theta_n^{\text{RMS}}, Z_n^{\text{RMS}}, \sigma_{\Theta,n}, \sigma_{Z,n}\right) \leftarrow \text{DKL_}\textsc{Predict}(H_{s,n}, T_{p,n}, V_\eta, \phi_n^{\text{rel}})$.

  Compute $p_{\text{fail},n}$ using (6.8) to (6.10) and set $f_n^{\text{node}} \leftarrow p_{\text{fail},n}$.

end for

Form base features $\mathbf{b} = \left[t_{\text{norm}}, \varphi_{\text{norm}}, \lambda_{\text{norm}}, V_{\text{norm}}, \sin\psi^{\text{rad}}, \cos\psi^{\text{rad}}\right]^{\mathsf{T}}$.

Flatten $\mathbf{f}^{\text{wp}}$ and $\mathbf{f}^{\text{node}}$ and set $\mathbf{obs} \leftarrow \text{concat}(\mathbf{b}, \mathbf{f}^{\text{wp}}, \mathbf{f}^{\text{node}})$.

return obs.

BIBLIOGRAPHY

Advanced Research Computing Core Facility, a.

ProteusDS, b. URL https://proteusds.com/.

Sea Hunter, c. URL https://en.wikipedia.org/wiki/Sea_Hunter.

U.S. Navy Announces Launch of Vanguard Unmanned Surface Vessel. *Naval Sea Systems Command*, January 2024. URL https://www.navsea.navy.mil/Media/News/Article-View/Article/3642890/us-navy-announces-launch-of-vanguard-unmanned-surface-vessel/.

Alexander Amini, Wilko Schwarting, Ava Soleimany, and Daniela Rus. Deep Evidential Regression, November 2020. URL http://arxiv.org/abs/1910.02600. arXiv:1910.02600 [cs].

Christopher M. Bishop. *Pattern recognition and machine learning*. Information science and statistics. Springer, New York, 2006. ISBN 978-0-387-31073-2.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*, 112(518):859–877, April 2017. ISSN 0162-1459, 1537-274X. doi: 10.1080/01621459.2017.1285773. URL http://arxiv.org/abs/1601.00670. arXiv:1601.00670 [stat].

Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight Uncertainty in Neural Networks, May 2015. URL http://arxiv.org/abs/1505.05424. arXiv:1505.05424 [stat].

Ali Cetin, Vegard R. Solum, and Cristina M. Evans. Machine Learning Based Prediction of Wave-Induced Vessel Response. In *Volume 5A: Ocean Engineering*, page V05AT06A034, Hamburg, Germany, June 2022. American Society of Mechanical Engineers. ISBN 978-0-7918-8589-5. doi: 10.1115/OMAE2022-78261. URL https://asmedigitalcollection.asme.org/OMAE/proceedings/OMAE2022/85895/V05AT06A034/1147799.

Chen Chen, Xian-Qiao Chen, Feng Ma, Xiao-Jun Zeng, and Jin Wang. A knowledge-free path planning approach for smart ships based on reinforcement learning. *Ocean Engineering*, 189: 106299, October 2019. ISSN 0029-8018. doi: 10.1016/j.oceaneng.2019.106299. URL https://www.sciencedirect.com/science/article/pii/S0029801819304706.

Zuoyi Chen, Jun Wu, Chao Deng, Xiaoqi Wang, and Yuanhang Wang. Deep Attention Relation Network: A Zero-Shot Learning Method for Bearing Fault Diagnosis Under Unknown Domains. *IEEE Transactions on Reliability*, 72(1):79–89, March 2023. ISSN 0018-9529, 1558-1721. doi: 10.1109/TR.2022.3177930. URL https://ieeexplore.ieee.org/document/9792444/.

Matthew Collette, Rachel Bielski, Peter Rohrer, Adam Magistro, Brendan Sulkowski, and Joseph Van Houten. Needs Exploration for Long-Term Autonomous Marine Systems: Working Report. August 2022. doi: 10.7302/4894. URL http://deepblue.lib.umich.edu/handle/2027.42/173163. Publisher: My University.

A. Deraemaeker and K. Worden. On the use of the Mahalanobis squared-distance to filter out environmental effects in structural health monitoring. *MATEC Web of Conferences*, 16:02004, 2014. ISSN 2261-236X. doi: 10.1051/matecconf/20141602004. URL https://www.matec-conferences.org/articles/matecconf/abs/2014/07/matecconf_csndd2014_02004/matecconf_csndd2014_02004.html. Publisher: EDP Sciences.

Thomas R. Etherington. Mahalanobis distances and ecological niche modelling: correcting a chi-squared probability error. *PeerJ*, 7:e6678, April 2019. ISSN 2167-8359. doi: 10.7717/peerj.6678. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC6450376/.

William Falcon and The PyTorch Lightning team. PyTorch Lightning, March 2019. URL https://github.com/Lightning-AI/lightning.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks, July 2017. URL http://arxiv.org/abs/1703.03400. arXiv:1703.03400 [cs].

K. E. Fjørtoft and Ø. J. Rødseth. Using the Operational Envelope to Make Autonomous Ships Safer. In *Proceedings of the 30th European Safety and Reliability Conference and 15th Probabilistic Safety Assessment and Management Conference*, pages 76–83. Research Publishing Services, 2020. ISBN 978-981-14-8593-0. doi: 10.3850/978-981-14-8593-0_3496-cd. URL http://rpsonline.com.sg/proceedings/9789811485930/html/3496.xml.

Vincent Fortuin. Priors in Bayesian Deep Learning: A Review, March 2022. URL http://arxiv.org/abs/2105.06868. arXiv:2105.06868 [stat].

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning, 2015. URL https://arxiv.org/abs/1506.02142. Version Number: 6.

Guillermo Gallego, Carlos Cuevas, Raúl Mohedano, and Narciso Garcia. On the Mahalanobis Distance Classification Criterion for Multidimensional Normal Distributions. *IEEE Transactions on Signal Processing*, 61:4387–4396, September 2013. doi: 10.1109/TSP.2013.2269047.

Jakob Gawlikowski, Cedrique Rovile Njieutcheu Tassi, Mohsin Ali, Jongseok Lee, Matthias Humt, Jianxiang Feng, Anna Kruspe, Rudolph Triebel, Peter Jung, Ribana Roscher, Muhammad

Shahzad, Wen Yang, Richard Bamler, and Xiao Xiang Zhu. A Survey of Uncertainty in Deep Neural Networks, January 2022. URL http://arxiv.org/abs/2107.03342. arXiv:2107.03342 [cs].

Dimitris G. Georgiadis, Manolis S. Samuelides, and Daniel Straub. Near-real-time ship grounding damage assessment using Bayesian networks, June 2025. URL http://arxiv.org/abs/2506.06493. arXiv:2506.06493 [stat].

Camila Gonzalez, Karol Gotkowski, Moritz Fuchs, Andreas Bucher, Armin Dadras, Ricarda Fischbach, Isabel Kaltenborn, and Anirban Mukhopadhyay. Distance-based detection of out-of-distribution silent failures for Covid-19 lung lesion segmentation. *Medical Image Analysis*, 82: 102596, August 2022. doi: 10.1016/j.media.2022.102596.

Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1321–1330. PMLR, July 2017. URL https://proceedings.mlr.press/v70/guo17a.html. ISSN: 2640-3498.

Siyu Guo, Xiuguo Zhang, Yisong Zheng, and Yiquan Du. An Autonomous Path Planning Model for Unmanned Ships Based on Deep Reinforcement Learning. *Sensors*, 20(2):426, January 2020. ISSN 1424-8220. doi: 10.3390/s20020426. URL https://www.mdpi.com/1424-8220/20/2/426.

James Hensman, Alex Matthews, and Zoubin Ghahramani. Scalable Variational Gaussian Process Classification, November 2014. URL http://arxiv.org/abs/1411.2005. arXiv:1411.2005 [stat].

John H. Holland. Genetic Algorithms. *Scientific American*, 267(1):66–73, 1992. ISSN 0036-8733. URL https://www.jstor.org/stable/24939139. Publisher: Scientific American, a division of Nature America, Inc.

Jakob Hollenstein, Sayantan Auddy, Matteo Saveriano, Erwan Renaudo, and Justus Piater. Action Noise in Off-Policy Deep Reinforcement Learning: Impact on Exploration and Performance.

J. J. Jensen. 4 - Wave Loads on Ships. In J. J. Jensen, editor, *Load and Global Response of Ships*, volume 4 of *Elsevier Ocean Engineering Series*, pages 123–178. Elsevier, 2001. doi: https://doi.org/10.1016/S1571-9952(01)80006-4. URL https://www.sciencedirect.com/science/article/pii/S1571995201800064. ISSN: 1571-9952.

Jørgen Juncher Jensen and A. E. Mansour. Estimation of Ship Long-term Wave-induced Bending Moment using Closed-Form Expressions. *International Journal of Maritime Engineering*, pages 41–55, 2002. ISSN 1479-8751. Publisher: The Royal Institution of Naval Architects.

Jørgen Juncher Jensen, Alaa E. Mansour, and Anders Smærup Olsen. Estimation of ship motions using closed-form expressions. *Ocean Engineering*, 31(1):61–85, January 2004. ISSN 00298018. doi: 10.1016/S0029-8018(03)00108-2. URL https://linkinghub.elsevier.com/retrieve/pii/S0029801803001082.

J.M.J Journee and W.W Massie. *Offshore Hydromechanics*. Delft University of Technology, 2001.

Charles F. F. Karney. Algorithms for geodesics. *Journal of Geodesy*, 87(1):43–55, January 2013. ISSN 0949-7714, 1432-1394. doi: 10.1007/s00190-012-0578-z. URL http://arxiv.org/abs/1109.4448. arXiv:1109.4448 [physics].

Katyanna Quach. IBM's autonomous Mayflower ship breaks down in second transatlantic attempt. *The Register*, May 2022. URL https://www.theregister.com/2022/05/11/mayflower_autonomous_ship/.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes, December 2022. URL http://arxiv.org/abs/1312.6114. arXiv:1312.6114 [stat].

Zhan Kong, Yaqi Cui, Wei Xiong, Fucheng Yang, Zhenyu Xiong, and Pingliang Xu. Ship Target Identification via Bayesian-Transformer Neural Network. *Journal of Marine Science and Engineering*, 10(5):577, May 2022. ISSN 2077-1312. doi: 10.3390/jmse10050577. URL https://www.mdpi.com/2077-1312/10/5/577. Publisher: Multidisciplinary Digital Publishing Institute.

S. Kullback and R. A. Leibler. On Information and Sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, March 1951. ISSN 0003-4851, 2168-8990. doi: 10.1214/aoms/1177729694. URL https://projecteuclid.org/journals/annals-of-mathematical-statistics/volume-22/issue-1/On-Information-and-Sufficiency/10.1214/aoms/1177729694.full. Publisher: Institute of Mathematical Statistics.

Charilaos Latinopoulos, Efstathios Zavvos, Dimitrios Kaklis, Veerle Leemen, and Aristides Halatsis. Marine Voyage Optimization and Weather Routing with Deep Reinforcement Learning. *Journal of Marine Science and Engineering*, 13(5):902, May 2025. ISSN 2077-1312. doi: 10.3390/jmse13050902. URL https://www.mdpi.com/2077-1312/13/5/902. Publisher: Multidisciplinary Digital Publishing Institute.

Ngartera Lebede and Saralees Nadarajah. Enhancing autonomous systems with bayesian neural networks: a probabilistic framework for navigation and decision-making. *Frontiers in Built Environment*, 11, May 2025. ISSN 2297-3362. doi: 10.3389/fbuil.2025.1597255. URL https://www.frontiersin.org/journals/built-environment/articles/10.3389/fbuil.2025.1597255/full. Publisher: Frontiers.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S. Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep Neural Networks as Gaussian Processes, March 2018a. URL http://arxiv.org/abs/1711.00165. arXiv:1711.00165 [stat].

Kimin Lee, Kibok Lee, Honglak Lee, and Jinwoo Shin. A Simple Unified Framework for Detecting Out-of-Distribution Samples and Adversarial Attacks, 2018b. URL https://arxiv.org/abs/1807.03888. Version Number: 2.

Sungyoon Lee, Hoki Kim, and Jaewook Lee. Graddiv: Adversarial robustness of randomized neural networks via gradient diversity regularization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. Publisher: IEEE.

Edward V. Lewis and Society of Naval Architects and Marine Engineers (U.S.), editors. *Principles of naval architecture. Volume III. Motions in waves and controllability*. Society of Naval Architects and Marine Engineers, Jersey City, NJ, 2nd revision edition, 1989. ISBN 978-0-939773-02-2 978-1-61583-299-6.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning, July 2019. URL http://arxiv.org/abs/1509.02971. arXiv:1509.02971 [cs].

Quanbao Lin, Huaxing Gou, Peidong Tian, Tian-Yu Zuo, Hanzhong Zhang, Xin Wang, and Poly Z.H. Sun. RL-Based USV Path Planning Under the Marine Multimodal Features Considerations. *IEEE Internet of Things Journal*, 12(11):15274–15287, June 2025. ISSN 2327-4662. doi: 10.1109/JIOT.2025.3530534. URL https://ieeexplore.ieee.org/document/10843741/.

David J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 4(3):448–472, May 1992. ISSN 0899-7667. doi: 10.1162/neco.1992.4.3.448. URL https://ieeexplore.ieee.org/document/6796869.

Sajad Marvi, Christoph Rist, Julian Schmidt, Julian Jordan, and Abhinav Valada. Evidential Uncertainty Estimation for Multi-Modal Trajectory Prediction, March 2025. URL http://arxiv.org/abs/2503.05274. arXiv:2503.05274 [cs].

Nis Meinert, Jakob Gawlikowski, and Alexander Lavin. The Unreasonable Effectiveness of Deep Evidential Regression. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(8):9134–9142, June 2023. ISSN 2374-3468, 2159-5399. doi: 10.1609/aaai.v37i8.26096. URL http://arxiv.org/abs/2205.10060. arXiv:2205.10060 [cs].

MK Ochi. Prediction of Occurrence and Severity of Ship Slamming at Sea. *Proceedings 5th Symposium on Naval Hydrodynamics*, 1964.

Mohammad Hossein Moradi, Martin Brutsche, Markus Wenig, Uwe Wagner, and Thomas Koch. Marine route optimization using reinforcement learning approach to reduce fuel consumption and consequently minimize CO2 emissions. *Ocean Engineering*, 259:111882, September 2022. ISSN 0029-8018. doi: 10.1016/j.oceaneng.2022.111882. URL https://www.sciencedirect.com/science/article/pii/S0029801822012239.

Radford M. Neal. Priors for Infinite Networks. In Radford M. Neal, editor, *Bayesian Learning for Neural Networks*, pages 29–53. Springer, New York, NY, 1996. ISBN 978-1-4612-0745-0. doi: 10.1007/978-1-4612-0745-0_2. URL https://doi.org/10.1007/978-1-4612-0745-0_2.

Maria Carolina Novitasari, Johannes Quaas, and Miguel R. D. Rodrigues. Cloudy with a chance of uncertainty: autoconversion rates forecasting via evidential regression from satellite data. *Environmental Data Science*, 3:e45, January 2024. ISSN 2634-4602. doi: 10.1017/eds.2024.37.

Sebastian W. Ober, Carl E. Rasmussen, and Mark van der Wilk. The Promises and Pitfalls of Deep Kernel Learning, July 2021. URL http://arxiv.org/abs/2102.12108. arXiv:2102.12108 [stat].

Vincent Page, Christopher Dadswell, Matt Webster, Mike Jump, and Michael Fisher. Towards the Determination of Safe Operating Envelopes for Autonomous UAS in Offshore Inspection Missions. *Robotics*, 10(3):97, September 2021. ISSN 2218-6581. doi: 10.3390/robotics10030097. URL https://www.mdpi.com/2218-6581/10/3/97. Number: 3 Publisher: Multidisciplinary Digital Publishing Institute.

Tamara Petranović, Antonio Mikulić, Marko Katalinić, Maro Ćorak, and Joško Parunov. Method for Prediction of Extreme Wave Loads Based on Ship Operability Analysis Using Hindcast Wave Database. *Journal of Marine Science and Engineering*, 9(9):1002, September 2021. ISSN 2077-1312. doi: 10.3390/jmse9091002. URL https://www.mdpi.com/2077-1312/9/9/1002.

Alexander Podolskiy, Dmitry Lipin, Andrey Bout, Ekaterina Artemova, and Irina Piontkovskaya. Revisiting Mahalanobis Distance for Transformer-Based Out-of-Domain Detection, May 2022. URL http://arxiv.org/abs/2101.03778. arXiv:2101.03778 [cs].

Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-Baselines3: Reliable Reinforcement Learning Implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021. URL http://jmlr.org/papers/v22/20-1364.html.

Ørnulf Jan Rødseth, Lars Andreas Lien Wennersberg, and Håvard Nordahl. Towards approval of autonomous ship systems by their operational envelope. *Journal of Marine Science and Technology*, 27(1):67–76, March 2022. ISSN 1437-8213. doi: 10.1007/s00773-021-00815-z. URL https://doi.org/10.1007/s00773-021-00815-z.

Matthew Schirmann. *Physics-Informed Data-Driven Models for Ship Response Prediction using Global Wave Data*. PhD thesis, University of Michigan, 2021. URL http://deepblue.lib.umich.edu/handle/2027.42/169863.

Matthew L. Schirmann, Matthew D. Collette, and James W. Gose. Data-driven models for vessel motion prediction and the benefits of physics-based information. *Applied Ocean Research*, 120:102916, March 2022. ISSN 01411187. doi: 10.1016/j.apor.2021.102916. URL https://linkinghub.elsevier.com/retrieve/pii/S0141118721003850.

Sandeep Kumar Singh, Jaya Shradha Fowdur, Jakob Gawlikowski, and Daniel Medina. Leveraging Graph and Deep Learning Uncertainties to Detect Anomalous Trajectories, March 2022. URL http://arxiv.org/abs/2107.01557. arXiv:2107.01557 [cs].

Yuhta Takida, Wei-Hsiang Liao, Chieh-Hsin Lai, Toshimitsu Uesaka, Shusuke Takahashi, and Yuki Mitsufuji. Preventing Oversmoothing in VAE via Generalized Variance Parameterization, August 2022. URL http://arxiv.org/abs/2102.08663. arXiv:2102.08663 [cs].

Petru Tighineanu, Lukas Grossberger, Paul Baireuther, Kathrin Skubch, Stefan Falkner, Julia Vinogradska, and Felix Berkenkamp. Scalable Meta-Learning with Gaussian Processes, December 2023. URL http://arxiv.org/abs/2312.00742. arXiv:2312.00742 [stat] version: 1.

Mark Towers, Ariel Kwiatkowski, Jordan K Terry, John U. Balis, Gianluca de Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente,

Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Jin Shen Tan, and Omar G. Younis. Gymnasium: A Standard Interface for Reinforcement Learning Environments. URL https://github.com/Farama-Foundation/Gymnasium.

G. E. Uhlenbeck and L. S. Ornstein. On the Theory of the Brownian Motion. *Physical Review*, 36 (5):823–841, September 1930. doi: 10.1103/PhysRev.36.823. URL https://link.aps.org/doi/10.1103/PhysRev.36.823. Publisher: American Physical Society.

Eckart Uhlmann, Mitchel Polte, Julian Blumberg, Sheng Yin, and Gang Wang. Increasing the Absolute Position Accuracy of Industrial Robots by Means of a Deep Continual Evidential Regression Model *. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4774–4780, May 2024. doi: 10.1109/ICRA57147.2024.10611053. URL https://ieeexplore.ieee.org/document/10611053.

T Vinccnty. DIRECT AND INVERSE SOLUTIONS OF GEODESICS 0 THE ELLIPSOID WLTH APPLICATION OF ESTED EQUATIO S.

Jie Wang and Youmin Zhang. A Tutorial on Gaussian Process Learning-based Model Predictive Control, April 2024. URL http://arxiv.org/abs/2404.03689. arXiv:2404.03689 [cs].

Andrew G Wilson, Zhiting Hu, Russ R Salakhutdinov, and Eric P Xing. Stochastic Variational Deep Kernel Learning. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/hash/bcc0d400288793e8bdcd7c19a8ac0c2b-Abstract.html.

Andrew Gordon Wilson, Zhiting Hu, Ruslan Salakhutdinov, and Eric P. Xing. Deep Kernel Learning, November 2015. URL http://arxiv.org/abs/1511.02222. arXiv:1511.02222 [cs].

Changmin Yu, Marko Seslija, George Brownbridge, Sebastian Mosbach, Markus Kraft, Mohammad Parsi, Mark Davis, Vivian Page, and Amit Bhave. Deep kernel learning approach to engine emissions modeling. *Data-Centric Engineering*, 1:e4, January 2020. ISSN 2632-6736. doi: 10.1017/dce.2020.4. URL https://www.cambridge.org/core/journals/data-centric-engineering/article/deep-kernel-learning-approach-to-engine-emissions-modeling/638332CFF08E18BCD55D6ACC75BB3B28.

Jun Yuan, Jiang Zhu, and Victor Nian. Neural Network Modeling Based on the Bayesian Method for Evaluating Shipping Mitigation Measures. *Sustainability*, 12(24):10486, January 2020. ISSN 2071-1050. doi: 10.3390/su122410486. URL https://www.mdpi.com/2071-1050/12/24/10486. Publisher: Multidisciplinary Digital Publishing Institute.

Ying-Ying Zhang, Teng-Zhong Rong, and Man-Man Li. The empirical Bayes estimators of the mean and variance parameters of the normal distribution with a conjugate normal-inverse-gamma prior by the moment method and the MLE method. *Communications in Statistics - Theory and Methods*, 48(9):2286–2304, May 2019. ISSN 0361-0926. doi: 10.1080/03610926.2018.1465081. URL https://doi.org/10.1080/03610926.2018.1465081. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/03610926.2018.1465081.

Geoff Ziezulewicz. The littoral combat ship's latest problem: Class-wide structural defects leading to hull cracks. *Navy Times*, May 2022.