

Training Physics-Guided Neural Networks with Multiple Constraints: An Application in Lake Ecology Modeling

Aanish K. Pradhan

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

Anuj Karpatne, Chair

Cayelan C. Carey

Paul C. Hanson

Xuan Wang

April 29, 2025

Blacksburg, Virginia

Keywords: Constrained optimization, Ecosystem modeling, Multitask Learning,

Physics-guided neural networks

Copyright 2025, Aanish K. Pradhan

Training Physics-Guided Neural Networks with Multiple Constraints: An Application in Lake Ecology Modeling

Aanish K. Pradhan

(ABSTRACT)

Lakes and reservoirs are critical components of Earth’s ecosystems but are increasingly threatened by climate change and human activity, underscoring the need for reliable tools for modeling and predicting lake ecology. While machine learning has shown potential in modeling such systems, sparse environmental data often limits the ability of machine learning models to produce physically consistent predictions or generalize to novel conditions. As a result, many existing approaches rely on computationally intensive physics-biogeochemical simulations to supplement training data. Physics-Guided Neural Networks (PGNN) offer a promising alternative by embedding scientific knowledge directly into the model through physical constraints applied during training. However, training these models at scale remains challenging due to the trade-offs between satisfying physical laws and fitting the data, often leading to optimization pathologies. This thesis explores the challenge of designing, training and evaluating PGNNs with up to six constraints without relying on auxiliary simulation data. We assemble a suite of physics-based constraints grounded in limnological principles and evaluate their impact on neural network predictions by assessing within-distribution and zero-shot performance. To navigate the challenge of training with multiple constraints, we explore the use of multitask learning methods to counteract gradient pathologies that arise when training PGNNs. Our results suggest that multitask learning approaches can improve in-distribution performance in certain architectures, but they do not enhance zero-shot performance compared to unconstrained models. Our findings highlight the inherent

complexity of scaling PGNNs and emphasize the need for principled training methodologies in data-scarce modeling contexts.

Training Physics-Guided Neural Networks with Multiple Constraints: An Application in Lake Ecology Modeling

Aanish K. Pradhan

(GENERAL AUDIENCE ABSTRACT)

Lakes and reservoirs play a vital role in supporting biodiversity, providing freshwater and regulating the environment. As these ecosystems face increasing stress from climate change and human activity, it is critical to develop reliable tools for modeling lake conditions such as oxygen levels, water temperature, and algae growth. While machine learning has been shown to be a promising approach, these methods rely on large amounts of data for training. However, data from environmental systems is often sparsely available which cause models to struggle with generating physically consistent predictions or generalizing to unseen situations. While past approaches have leveraged physics-biogeochemical models to simulate the lake ecosystem and generate more data, this approach can be computationally expensive. This thesis explores the use of physics-guided neural networks (PGNN), a class of machine learning models that incorporate scientific knowledge to improve accuracy and realism which excel in data-scarce situations. However, training these models can be challenging as the model may struggle to balance between obeying the physical knowledge and fitting the data. To navigate this, we leverage multitask learning methods to assist the models during training. Our results demonstrate that applying these methods, may show promise in training PGNNs at scale albeit only with certain model architectures. Furthermore, our results show that while PGNNs trained with multiple constraints may predict better on the data they are trained on, they fail to generalize to unseen data compared to models trained without

physical knowledge. These findings highlight both the complexity of combining physics and machine learning at scale to support lake and reservoir ecosystem modeling.

Dedication

To our family's beloved German Shepherd, Momo, who brought joy, love, loyalty and peace into our lives. You are missed every day.

To my parents, whose strength, sacrifices and unconditional support carried me through one of the most difficult chapters of our family's lives. Your courage in immigrating to this country, steady guidance and support have been the foundation of all my achievements and benchmark to live up to. This work is as much yours as it is mine.

Acknowledgments

Writing a thesis as an accelerated master's student was a challenging undertaking. On several occasions, I felt like giving up because I thought I wouldn't finish on time. In those moments, I was grateful to have an enormous support system behind me. It took a village to help me get this done and I want to take the time to acknowledge the contributions of everyone who helped along the way.

First and foremost, I want to acknowledge my advisor, Dr. Anuj Karpatne. Working under his guidance over the past year has been both intellectually challenging and deeply rewarding. I embarked on this journey towards the tail-end of my undergraduate studies with little research experience but his mentorship, patience and sometimes challenging questions, even during the times that he was busy navigating his own commitments, have helped me to think deeper, communicate clearer, and push beyond surface-level understanding. His direction and feedback, were instrumental in shaping this work. This thesis stretched me in ways I didn't anticipate and there were many moments of anxiety and self-doubt; I'm grateful for Dr. Karpatne's reassurance in those times. Above all, he gave me the space to struggle but also grow, and taught me to not fixate on the destination, but to appreciate the journey of research itself, for which I am grateful.

I would like to thank my committee members for their support and guidance throughout this thesis. I am especially grateful to Dr. Cayelan C. Carey at the Virginia Tech Center for Ecosystem Forecasting and Dr. Paul C. Hanson at the University of Wisconsin-Madison Center for Limnology for their mentorship and invaluable guidance in developing the physics-guided constraints and deepening my understanding of lake ecological processes and ecosystem modeling. I'd also thank Dr. Xuan Wang for her service on my committee.

Additionally, I want to acknowledge my labmates—Amartya Dutta, Sepideh Fatemi, Mridul Khurana, Harish Babu Manogaran, M. Maruf, Kazi Sajeed Mehrab, Abhilash Neog, and Medha Sawhney—for their insights, perspectives, and teachings, but more importantly, for the fun and camaraderie we shared. A special thanks goes to Amartya, Kazi, and Medha for their steady reassurance during moments of anxiety and stress. Their reminders to step back, see the bigger picture, and re-approach challenges with a calm, composed mindset were invaluable when things didn't go as planned. I still have much to learn from them.

I am grateful to my collaborators at the University of Wisconsin-Madison Center for Limnology and the Virginia Tech Center for Ecosystem Forecasting. I'd especially like to thank Bennett McAfee for his collaboration on the LakeBeD-US manuscript and dataset, which formed the foundation for training my models, as well as Dr. Robert Ladwig for his guidance in developing the physics constraints. I'd also like to acknowledge Dr. Mary Lofton, Emma Marchisin, Sophia Skoglund and Dr. Rohit Shukla for their valuable insights and perspectives during our weekly Eco-KGML meetings. I'd like to extend a special thanks to Dr. R. Quinn Thomas and Dr. Freya Olsson for their mentorship during my undergraduate senior capstone project. Their guidance—and our trip to Lake Sunapee at the start of my master's—played a key role in sparking my interest in climate and environmental data science, ultimately shaping the direction of this thesis.

On the topic of the capstone project, I want to express my heartfelt gratitude to the Virginia Tech Computational Modeling and Data Analytics program, where I completed my undergraduate studies. My accomplishments in graduate school would not have been possible without the incredible professors who taught and advised me throughout that journey—Angie Patterson, Fred Faltin, Justin Loda, Eric Ufferman, Jason Wilson and Xinwei Deng. A special acknowledgment goes out to professors Mark Embree, Christian Lucero and Tim Warburton for encouraging me to pursue a thesis-based master's degree.

Throughout this thesis journey, I faced many moments of uncertainty and self-doubt. During those times, I was fortunate to be surrounded by friends whose encouragement and support meant the world to me. I want to thank Daniel Palamarchuk, who walked the accelerated M.S. path before me and generously guided me through every step of the process. I'm also grateful to Connor Bluestein, Henry Ham, Yunis Hussein, Noah Provenzano, and Abdullah Rizwan for their friendship and encouragement. Many thanks to Patrick Dewey, Aaron Gomez, and Amber Kummer for hosting game nights that gave me much-needed space to unwind. A very special thank you to my roommates, Bennet Humpton and Alex-Jaimes Sandoval, for their close friendship and unwavering support during my undergraduate and graduate studies.

My family has supported me through every step of my educational journey, and they were my greatest champions during this thesis. I am deeply grateful to my parents, Sonal and Kaustubh, for their unwavering support, nurturing guidance, and, when needed, their tough love. This last year-and-a-half has been a challenging time for our family, riddled with untimely passings and hospitalizations of loved ones. Despite everything, my parents remained calm and composed. Watching them navigate life with steadiness, while I often found myself overwhelmed, has shown me how much more I have yet to mature and learn—not just as a student or researcher, but as a person. Their example has been one of the most profound lessons in resilience I've received. My mother, especially, was instrumental in helping me make it through. She not only served as a source of emotional strength, listening patiently to my worries and fears during our FaceTime calls, but also as a personal inspiration having completed a master's thesis of her own—while raising me, no less. I truly couldn't have done this without her. I'd also like to thank my uncle, Rahul, for his occasional but well-timed advice. Although he has a way of sounding like a wise, old oracle—something he fully owns and jokes about—his words stuck with me more than I care to admit. I'm grateful

for his humor, perspective, and support throughout this process. I owe an immense amount of gratitude to my younger brother Aakash. Over the last year, he has diligently stayed focused on his studies without ever complaining. Knowing that the two of us were walking parallel paths, each carrying our own weight in uncertain times, helped me feel less alone. He may be my younger brother, but he's taught me more in this past year than he knows. A heartfelt and special thank you goes out to my cousin, Shreya, whose help last summer was a lifeline. Shreya held down the home front, supporting my brother, mother and I, so we could focus on helping my dad and so I could focus on my research and coursework. She did all of this while navigating a job search under immense pressure as an international student. Her efforts during this time did not go unnoticed and I'm forever grateful.

Last, but certainly not least, I want to thank Virginia Tech and the Town of Blacksburg for being my home for the last five years. I'm incredibly privileged to have learned from, and alongside, some truly bright individuals who encouraged curiosity, growth, and new ways of thinking, both inside and outside the classroom.

Contents

List of Figures	xv
List of Tables	xx
1 Introduction	1
1.1 Artificial Intelligence for Science	1
1.2 Lake and Reservoir Ecosystem Modeling	2
1.3 Physics-Guided Machine Learning	3
1.3.1 Challenges of Non-Machine Learning Approaches	3
1.3.2 Drawbacks of Pure Machine Learning Approaches	5
1.3.3 Advantages of Physics-Guided Machine Learning	6
1.4 Thesis Contributions	7
2 Literature Review	9
2.1 Mathematical Background	9
2.1.1 Constrained Optimization: Penalized Formulation	10
2.1.2 Pareto Optimization and Optimality	13
2.2 Challenges with Training Physics-Guided Neural Networks	14
2.3 Methods of Training Physics-Guided Neural Networks	16

2.3.1	Adaptive Balancing	16
2.3.2	Architecture	18
2.4	Applications of Physics-Guided Neural Networks to Lake Ecology Modeling .	19
2.4.1	Water Temperature Modeling	19
2.4.2	Phosphorus Modeling	20
2.4.3	Future Directions for Lake Ecology Modeling	20
3	Physics-Guided Constraints	22
3.1	Photosynthetic Active Radiation	22
3.1.1	Monotonicity Constraint	23
3.1.2	Beer-Lambert Constraint	24
3.2	Temperature	27
3.2.1	Temperature-Density-Depth Monotonicity Constraint	27
3.2.2	Energy Conservation Constraint	30
3.3	Dissolved Oxygen	37
3.4	Total Phosphorus	38
4	Data and Methodology	41
4.1	Data	41
4.1.1	Overview	41
4.1.2	Preprocessing	42

4.1.3	Similarity Analysis	43
4.2	Models	47
4.2.1	Multilayer Perceptron	47
4.2.2	Spatiotemporal Sequence-to-Sequence Model	49
4.3	Experiments	52
4.3.1	Experiment 1: Baselines	53
4.3.2	Experiment 2: Physics-Guided Constraints with Static Balancing	53
4.3.3	Experiment 3: Multi-Task Learning Methods	54
5	Results	55
5.1	Baselines	55
5.1.1	MLP	57
5.1.2	LSTM	58
5.2	Physics-Guided Constraints with Static Balancing	59
5.2.1	MLP	61
5.2.2	LSTM	62
5.3	Adaptive MTL Methods	65
5.3.1	MLP	66
5.3.2	LSTM-RNN	66
5.3.3	Diagnostic Experiments	68

5.3.4	Summary	72
6	Conclusions	75
6.1	Discussion	75
6.2	Limitations and Future Work	76
6.2.1	Data	76
6.2.2	Physics-Guided Constraints	77
6.2.3	Modeling	79
	Bibliography	82
	Appendices	91
	Appendix A Data: Supplementary Information	92
A.1	Missing Values Analysis	92

List of Figures

2.1	Example of the regularization effect caused by PG constraints. The blue, concentric ovals represent the contour level curves of an example, simplified, convex data cost landscape of a neural network; in reality, neural network cost function landscapes are highly nonconvex [23]. Without PG constraints, an optimizer would descend this cost function landscape down to the minima (i.e. the center of the concentric ovals). Since the PG constraints are applied over the predictions in the feature space, the nonlinear nature of the neural networks might cause the constraint region to map to an amorphous manifold in the parameter space, indicated by the gray “Physically Feasible Region”. The PG constraints push the optimizer towards solutions that minimize the data cost function while also being physically consistent as indicated by the black dot at the intersection of the physically feasible constraint region and the data cost function.	12
2.2	Visualizations of conflicting and imbalanced gradient pathologies in \mathbb{R}^2	15
2.3	Visualizations of PCGrad and IMTL methods.	17
3.1	Attenuation of PAR over 10 meters given various diffuse attenuation coefficients.	25
3.2	Density of water as a function of temperature, showing two different empirical equations for comparison.	28

3.3	Depth (measured positively in a downwards direction) versus density plot. Warmer, sparser water resides towards the surface of the lake (i.e. the top of the plot) while colder, denser water resides towards the bottom of the lake. .	29
4.1	Third-order tensor of dimension $V \times Z \times T$ representing the LakeBeD-US data. The slices represent different variables of interest, as described at the start of Chapter 3. The rows are the depths (Z) and the columns are the timesteps (T).	42
4.2	PCA biplot of lake similarity with $k = 4$, as obtained by maximizing silhouette score.	44
4.3	Silhouette scores from the lake similarity analysis. A higher silhouette score indicates a more ideal selection for the number of clusters.	45
4.4	Lake similarity, as determined by Euclidean distance, in the scaled, full feature space. The darker the color, the more similar the lakes are to each other. . .	46
4.5	Cyclical encoding visualizations.	48

4.6	Schematic of the spatiotemporal seq2seq model. The model takes an input tensor of lake variables ($V_{\text{Lake}} \times Z \times T_{\text{Lookback}}$) and a matrix of meteorological drivers ($T_{\text{Lookback}} \times V_{\text{Driver}}$). Lake data passes through a CNN-based spatial encoder and an LSTM-based temporal encoder. Meteorological drivers are concurrently encoded by an independent LSTM. The embeddings are fused via multi-headed cross-attention, which then initializes an LSTM-based temporal decoder that autoregressively predicts future lake states using a forecast of meteorological drivers while being prompted at each timestep with an embedding obtained from future driver data. A CNN-based spatial decoder reconstructs the final output tensor ($V_{\text{Lake}} \times Z \times T_{\text{Horizon}}$).	50
5.1	Overall test RMSE (i.e. RMSE across all variables when normalized) by training fraction of the model and colored by model architecture and configuration. The pair of numbers in the parenthesis indicates the lookback and horizon window lengths of the LSTM model in the form “(lookback-horizon)”.	55
5.2	Baseline, per-variable test performance for Falling Creek Reservoir by training fraction and colored by model architecture and configuration. The pair of numbers in the parenthesis indicates the lookback and horizon window lengths of the LSTM model in the form “(lookback-horizon)”.	57
5.3	Baseline, per-variable test performance for Trout Bog by training fraction and colored by model architecture and configuration. The pair of numbers in the parenthesis indicates the lookback and horizon window lengths of the LSTM model in the form “(lookback-horizon)”.	58

5.4	Baseline, per-variable test performance for Trout Lake by training fraction and colored by model architecture and configuration. The pair of numbers in the parenthesis indicates the lookback and horizon window lengths of the LSTM model in the form “(lookback-horizon)”.	59
5.5	A \log_{10} -linear plot of the learning convergence and validation error curves of the MLP model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The model trained with static PG constraint balancing is shown against the baseline model trained with no PG constraints.	63
5.6	A \log_{10} -linear plot of the learning convergence and validation error curves of the LSTM-RNN model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The model trained with static PG constraint balancing is shown against the baseline model trained with no PG constraints.	64
5.7	A \log_{10} -linear plot of the learning convergence and validation error curves of the MLP model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The various MTL methods are shown against the curves from the baseline model and model trained with static PG constraint balancing.	68

5.8	A \log_{10} -linear plot of the learning convergence and validation error curves of the LSTM-RNN model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The various MTL methods are shown against the curves from the baseline model and model trained with static PG constraint balancing.	70
5.9	A \log_{10} -linear plot of the learning convergence and validation error curves of the MLP model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The various combinations of MTL method and the applied stabilization technique are shown against the curves from the baseline model and model trained with static PG constraint balancing.	71
5.10	A \log_{10} -linear plot of the learning convergence and validation error curves of the LSTM-RNN model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The various combinations of MTL method and the applied stabilization technique are shown against the curves from the baseline model and model trained with static PG constraint balancing.	72
A.1	Missing value percentages for Falling Creek Reservoir.	93
A.2	Missing value percentages for Prairie Lake.	94
A.3	Missing value percentages for Trout Bog.	95
A.4	Missing value percentages for Trout Lake.	96

List of Tables

4.1	Acceptable flags from LakeBeD-US.	43
4.2	Loadings of first two principal components. Darker shades correspond to higher magnitude loading, indicating more dependence on the attribute. . . .	45
4.3	Selected lakes from LakeBeD-US. Prarie Lake is selected as the outlier, zero-shot evaluation lake.	47
4.4	Input features for the MLP architecture. All features except the sine and cosine cyclical encodings (CE) are Z-score normalized. The equations for generating the sine-cosine CEs are shown in Equations 4.1 and 4.2. Visualizations of the encodings are shown in Figure 4.5.	48
5.1	Baseline MLP overall, test RMSE. Blue highlight indicates the training fraction that performed the best.	57
5.2	Baseline MLP overall, zero-shot RMSE on PRLA. Distance of each lake to PRLA, according to Subsection 4.1.3, is indicated in “Distance” column. Blue highlight indicates the lake that performs the best on PRLA. Bold indicates the training fraction for each lake that performed the best.	58
5.3	LSTM baseline, test RMSE by training utilization fraction and sequence length. Blue highlight indicates the training fraction that performed the best.	59

5.4	LSTM baseline, zero-shot RMSE on PRLA by training utilization fraction and sequence length. Distance of each lake to PRLA, according to Subsection 4.1.3, is indicated in the “Dist.” column. Blue highlight indicates the lake that performs the best on PRLA. Bold indicates the training fraction for each lake that performed the best.	60
5.5	Test RMSE of MLP model trained with PG constraints using a static balancing approach. The relative RMSE against the baseline MLP model with no PG constraints is shown. Green indicates an improvement over the baseline while red indicates a worsening over the baseline.	62
5.6	Zero-shot RMSE of MLP model trained with PG constraints using a static balancing approach on PRLA. The relative RMSE against the baseline MLP model with no PG constraints is shown. Green indicates an improvement over the baseline while red indicates a worsening over the baseline. The distance of each lake to PRLA is indicated in the “Dist.” column. Bold indicates the best-performing lake.	62
5.7	Test RMSE of LSTM-RNN model trained with PG constraints using a static balancing approach. The relative RMSE against the baseline MLP model with no PG constraints is shown. Red indicates a worsening over the baseline.	64
5.8	Zero-shot RMSE of LSTM-RNN model trained with PG constraints using a static balancing approach on PRLA. The relative RMSE against the baseline MLP model with no PG constraints is shown. Red indicates a worsening over the baseline. The distance of each lake to PRLA is indicated in the “Dist.” column. Bold indicates the best-performing lake.	64

5.9	Test RMSE of the MLP model trained with PG constraints using the PCGrad and IMTL methods. The relative RMSE against the baseline MLP model with no PG constraints is shown. Red indicates a worsening over the baseline.	67
5.10	Zero-shot RMSE of the MLP model, trained with PG constraints using the PCGrad and MTL method, on PRLA. The relative RMSE against the baseline MLP model with no PG constraints is shown. Red indicates a worsening over the baseline. The distance of each lake to PRLA is indicated in the “Distance” column.	67
5.11	Test RMSE of the LSTM model trained with PG constraints using the PCGrad and IMTL methods. The relative RMSE against the baseline LSTM model with no PG constraints is shown. Green indicates an improvement over the baseline while red indicates a worsening over the baseline.	69
5.12	Zero-shot RMSE of the LSTM model, trained with PG constraints using the PCGrad and IMTL method, on PRLA. The relative RMSE against the baseline LSTM model with no PG constraints is shown. Green indicates an improvement over the baseline while red indicates a worsening over the baseline. The distance of each lake to PRLA is indicated in the “Distance” column.	69
5.13	Comparison of test RMSE values across methods for each lake and model. Relative RMSE of the best method compared to baseline is shown on the very right. Blue indicates the best performing method. Green indicates an improvement over the baseline and red indicates a worsening over the baseline. * indicates gradient clipping was applied and ** indicates LR warm-up was applied.	74

5.14 Comparison of zero-shot RMSE values across methods for each lake and model. Relative RMSE of the best method compared to baseline is shown on the very right on PRLA. The “Dist.” column indicates the distance to PRLA. **Blue** indicates the best performing method. **Green** indicates an improvement over the baseline and **red** indicates a worsening over the baseline. * indicates gradient clipping was applied and ** indicates LR warm-up was applied. **Bold** indicates the lake that performed the best on the zero-shot evaluation.

List of Abbreviations

AI Artificial Intelligence

CE Cyclical Encoding

CHL-*a* Chlorophyll *a*

DL Deep Learning

DO Dissolved Oxygen

FCR Falling Creek Reservoir

GLM General Lake Model

GradNorm Gradient Normalization

HPD Hybrid-Physics-Data

KGML Knowledge-Guided Machine Learning

LSTM-RNN Long Short-Term Memory Recurrent Neural Network

MHCA Multiheaded Cross-Attention

ML Machine Learning

MLP Multilayer Perceptron

MTL Multi-task Learning

ODE Ordinary Differential Equation

PAR Photosynthetic Active Radiation

PC Principal Component

PCA Principal Components Analysis

PCGrad Projecting Conflicting Gradients

PDE Partial Differential Equation

PG Physics-Guided

PGML Physics-Guided Machine Learning

PGNN Physics-Guided Neural Network

PIML Physics-Informed Machine Learning

PINN Physics-Informed Neural Network

PRLA Prarie Lake

RNN Recurrent Neural Network

seq2seq Sequence-to-Sequence

TB Trout Bog

TP Total Phosphorus

TR Trout Lake

w.r.t. with respect to

Chapter 1

Introduction

In this chapter, we begin with a summary on the usage of artificial intelligence (AI) for science in Section 1.1, followed by a discussion on the relevance of lake ecology modeling in Section 1.2. In Section 1.3, we introduce the field of Physics-guided machine learning and conclude by discussing the contributions of this thesis in Section 1.4.

1.1 Artificial Intelligence for Science

Since 2012, deep learning (DL) has surged in popularity and has since revolutionized numerous technical domains, achieving remarkable success across a variety of fields such as computer vision and natural language processing.

Alongside advances in DL techniques, the availability of increasingly larger scientific datasets, acquired through sensors, simulations, and experimental measurements, as well as growing levels of computing capabilities have led to the emergence of “AI for Science” as a field. The goal of AI for Science is to leverage AI to extract insights from large volumes of data to promote scientific discovery and understanding of the world. This may involve using AI to accelerate scientific simulations, design or optimize experiments, and discover patterns in large-scale datasets [33]. AI for Science initiatives have already shown immense potential and are poised to transform the way research is conducted [13, 20, 42]. In this thesis, we explore the application of AI in the domain of lake and reservoir ecological modeling, specifically

through physics-guided neural networks (PGNN).

1.2 Lake and Reservoir Ecosystem Modeling

Lakes and reservoirs play a vital role in global biogeochemical cycles, biodiversity conservation, and the provision of ecosystem services such as drinking water, fisheries, and recreation. Lakes have a large ecological and societal impact. For instance, while 70% of the planet is covered by water, 97% is saltwater that is unsuitable for drinking due to the high cost and energy requirements of desalination. Of the remaining 2.5% that is freshwater, most is locked in glaciers or underground aquifers, leaving only 0.5%, mostly in lakes, rivers, and reservoirs as accessible water for human use [34].

Accurate modeling of lake and reservoir conditions is essential for anticipating critical ecological changes and enabling timely, proactive management. Such changes include rising water temperatures, decreased dissolved oxygen levels, nutrient accumulation, and the proliferation of harmful algal blooms, each of which can degrade water quality, disrupt aquatic ecosystems, and threaten public health. For example, warmer water can accelerate algal growth, which in turn can lead to oxygen depletion and massive fish die-offs. If not predicted and mitigated early, these changes can push a lake past an ecological “tipping point”, where recovery becomes difficult or impossible.

Recent advances, including coupled physical-biogeochemical models, machine learning, and real-time sensor networks, have improved our ability to forecast such processes. Global collaborative efforts, such as the Global Lake Ecological Observatory Network, have further accelerated progress by promoting open data sharing and interdisciplinary research.

Nonetheless, significant challenges remain. Modeling must account for complex, multiscale

interactions and work with sparse data, particularly in under-monitored or manually sampled systems. Moreover, predictions must be both scientifically accurate and actionable to inform environmental policy and management decisions. Meeting these challenges is essential to safeguarding freshwater ecosystems and the communities that depend on them.

1.3 Physics-Guided Machine Learning

To motivate the need for a *physics-guided* machine learning (PGML) approach to lake and reservoir ecosystem modeling, we must first understand the challenges associated with traditional, non-machine learning approaches methods of predicting of physical systems.

1.3.1 Challenges of Non-Machine Learning Approaches

Traditionally, modeling of lake and reservoir ecology—and physical systems in general for that matter—requires using *process-based* models. Namely, this requires domain experts to painstakingly craft models, typically in the form of ordinary differential equations (ODE) and partial differential equations (PDE), that encode their understanding of the physical processes in the aquatic ecosystem. For example, the General Lake Model (GLM) 3.0 is a popular water balance and one-dimensional vertical stratification hydrodynamic model [18]. When paired with meteorological driver data and calibrated with observational data, GLM can account for the effect of water inflow/outflow, lake mixing, as well as lake surface heating and cooling. Such models are difficult to solve analytically into a closed-form solution (i.e. a form where the predicted state of the system can be expressed as a function of space parameters, \mathbf{z} , and a time parameter, t , such as $\mathbf{f}(\mathbf{z}, t)$) which allows the lake ecosystem's model to be evaluated at any point in spacetime. Instead, iterative numerical solvers (e.g.,

Euler’s method for ODEs and the Finite element method for PDEs) are used to predict the state \mathbf{x} of the system at each point in time. Subsequent states depend on applying the encoded dynamics to the system’s state at the current timestep (i.e. $\mathbf{x}_{t+1} = \mathbf{f}(\mathbf{x}_t)$).

Process-based models are highly explainable and are the bedrock of scientific computing. However, they have numerous shortcomings as discussed below:

1. **Computational Expense:** Numerically solving process-based models can be computationally expensive, especially for high-dimensional systems. For practical use, simplifying assumptions of the processes are required to make the models tractable or the precision of the computations are sacrificed, both of which undermine the accuracy of the predictions.
2. **Parametrization:** Such models require certain parameters to be set for the simulation to be executed. For example, GLM may require parameters such as net primary productivity, half-saturation and respiration coefficients to be set. These parameters must be estimated or measured from the system of interest and, if set incorrectly, can cause the simulation to deviate from the ground truth dynamics of the system.
3. **Equifinality:** Extending off of issues stemming from parameterization, at times, different parameterizations of a process-based model may yield equivalent predictions of the system. For example, different configurations of net primary productivity, respiration and half-saturation coefficients may yield the same outcomes for predictions of dissolved oxygen but the different parametrizations describe completely different lake ecosystems. In such cases, it can be difficult to select an appropriate model.
4. **Generalizability:** Process-based models are typically designed and calibrated for specific lakes, requiring detailed knowledge of lake geometry, inflow/outflow dynamics,

and ecological characteristics. As a result, these models often do not generalize well to new lakes without substantial re-parameterization.

5. **Modeling Accuracy:** Process-based models, as mentioned earlier, encode the understanding of the lake into systems of differential equations. If this understanding is inaccurate or incomplete, then process error will arise and then the predictions from the simulation will deviate from reality.

1.3.2 Drawbacks of Pure Machine Learning Approaches

Approaching such problems using machine learning (ML) provides a significant advantage. Instead of numerically solving a process-based model, an ML model could be trained on a large volume of data to implicitly learn the physics that dictate the dynamics of the system. While the upfront computational expense of training and validating the model is large, the inference process (i.e. forward propagation) is far more inexpensive because it bypasses the need for an iterative scheme when generating predictions. Despite its vast number of successes in scientific application, the usage of ML for predicting physical systems has drawbacks as discussed below:

1. **Generalizability:** Traditional ML relies on learning the distribution of the training data by leveraging statistical techniques. Consider for example, in the case of this thesis, a model trained to forecast water temperature at a given lake in an equatorial region. We intuit this model would perform poorly if directly applied to forecasting water temperature at a lake in an Arctic region.
2. **Data Requirements:** DL methods often rely on large volumes of training data which can be sparsely available in environmental science domains, such as lake ecology modeling.

- Physical Inconsistency:** As a result of lack of data, models can generate predictions that are inconsistent with known physical laws of the system. This can become a significant problem in fields where physical plausibility is paramount. Model predictions that contradict known physics are not only theoretically incorrect but may lead to costly errors in engineering design or policy decisions.

Naturally, the question arises: “can one combine the strengths of traditional process-based models and data-driven models to infer solutions to physical problems with speed but also physical accuracy”?

1.3.3 Advantages of Physics-Guided Machine Learning

Physics-informed machine learning (PIML) and PGML emerged as subfields to address precisely these challenges by harmonizing the predictive power of data-driven models with the theoretical rigor of physical laws. PGML explicitly incorporates physical principles, typically represented through ODEs, PDEs, or other domain-specific constraints, into the training process of models. By doing so, PGML models can leverage data, even in data-scarce scenarios, while also retaining consistency with established scientific knowledge. There are several compelling advantages to this integrated approach as discussed below:

- Enhanced Generalizability:** Incorporating physical constraints into DL models enables improved extrapolation performance, particularly in data-scarce regions of the input space. For instance, a PGML model trained to predict water temperature for lakes located in temperate climates could, in principle, perform better than a vanilla, data-only model when applied to lakes in different geographic or climatic contexts due to adherence to universal thermodynamic laws.

2. **Improved Interpretability and Explainability:** By encoding known physical laws and relationships within the model, PGML enhances interpretability. Rather than being entirely “black-box”, these models enforce insights into underlying processes and system dynamics. This interpretability not only strengthens trust in the model outcomes but also aids researchers in identifying novel insights and potential anomalies.
3. **Physical Consistency and Plausibility:** Predictions generated by PGML are inherently constrained to be physically plausible by design, thus reducing or eliminating physically impossible predictions. For ecological modeling applications, such as those addressed in this thesis, maintaining physical plausibility is essential, as predictions inform policy decisions, ecosystem management, and long-term planning efforts.
4. **Data Efficiency:** Explicitly encoding known physics reduces the dependency on large volumes of labeled data, a significant advantage in environmental applications where data collection can be expensive, sparse, or logistically challenging. Models guided by physical knowledge can leverage limited observational data more effectively, potentially reducing monitoring costs and efforts.

In summary, PGML represents a promising integration of data-driven and physics-based modeling paradigms, offering distinct advantages in interpretability, generalization, efficiency, and physical fidelity. By embedding expert domain knowledge directly into the learning process, PGML is uniquely positioned to address the complex challenges encountered in modeling lake and reservoir ecosystems.

1.4 Thesis Contributions

This thesis offers the following contributions:

1. **Scaling PGNNs to Multiple Physical Constraints:** We explore the possibility of training PGNNs with more than two physics-guided (PG) constraints. Prior studies have typically focused on one or two constraints per model; this work scales up to six and explores the implications on training dynamics and performance.
2. **Observational Data-Only Training:** Unlike prior PGNN work in lake ecology modeling, which relies on hybrid-physics-data (HPD) modeling (i.e. combining observational and simulation data) to support learning, our models are trained solely on observational data to investigate the ability of PGNNs to perform without auxiliary simulation data.
3. **Application of Multitask Learning Methods to PGNNs:** We explore the use of multitask learning methods for training PGNNs under multiple, potentially conflicting PG constraints. We also assess their interaction with training stabilization techniques like gradient clipping and learning rate warm-up.
4. **Evaluation Under Data-Scarce and Zero-Shot Scenarios:** We investigate how PGNNs and MTL methods perform in data-scarce settings, evaluating both within-distribution out-of-distribution generalization (i.e. zero-shot) across ecologically distinct lakes.

Chapter 2

Literature Review

In this chapter, we present an overview of existing literature on PGNNs. We begin with the mathematical background describing the optimization setup of PGNNs in Section 2.1, followed by a discussion of the optimization challenges when training PGNNs in Section 2.2. Next, we review existing solution approaches to training PGNNs in Section 2.3. Finally, in Section 2.4, we provide an overview of existing applications of PGNNs to lake ecology modeling.

2.1 Mathematical Background

The task of training neural networks can be expressed as an optimization problem which involves minimizing a cost function \mathcal{L} , with respect to (w.r.t.) the set of learnable parameters of the model \mathcal{P} , which measures the error of the model's predictions given some dataset \mathcal{D} . This is expressed in Equation 2.1:

$$\min_{\mathcal{P}} \mathcal{L}(\mathcal{P}; \mathcal{D}) \tag{2.1}$$

where

\mathcal{L} = Cost function (e.g., Cross-Entropy, Mean Square Error)

\mathcal{D} = Training dataset of the model [i.e., $\mathcal{D} = \{(X_i, Y_i)_{i=1}^n\} : (X_i, Y_i) \sim \mathcal{D}$]. (X_i, Y_i) is an input-observation pair sampled from the data distribution \mathcal{D} .

\mathcal{P} = Set of parameters of the model (i.e., weights, $\mathbf{W}^{(\cdot)}$, and biases, $\mathbf{b}^{(\cdot)}$, across l hidden layers: $\{\mathbf{W}^{(1)}, \mathbf{W}^{(2)}, \dots, \mathbf{W}^{(l)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}, \dots, \mathbf{b}^{(l)}\}$)

This cost function may be subject to constraints on its constituent parameters [32], as shown in Equation 2.2:

$$\min_{\mathcal{P}} \mathcal{L}(\mathcal{P}; \mathcal{D}) \quad \text{subject to} \quad \begin{cases} c_i(\mathcal{P}) = 0, i \in \mathcal{E} \\ c_i(\mathcal{P}) \geq 0, i \in \mathcal{J} \end{cases} \quad (2.2)$$

where

c_i = Constraint function

\mathcal{E} = Set of equality constraints

\mathcal{J} = Set of inequality constraints

Note that each constraint function c_i is *independent* of the training dataset of the model. Techniques like L1 and L2 regularization, which minimize the L1 and L2 norms of model parameters, respectively, to reduce overfitting, follow this constrained optimization setup.

2.1.1 Constrained Optimization: Penalized Formulation

Equation 2.2 shows the *primal* formulation of a constrained optimization task. Optimizing the primal problem can be challenging as the constraints need to be satisfied at every iteration of the numerical optimization. It is a common practice to reformulate this optimization task

into its penalized form where the constraints are “relaxed” into violable, “soft” constraints (a.k.a. penalty terms). The penalty terms are balanced with Lagrange multipliers, chosen as hyperparameters through some cross-validation scheme, that dictate the amount of emphasis to give on violations of the constraints. The Lagrangian formulation is shown in Equation 2.3:

$$\min_{\mathcal{P}} \mathcal{L}(\mathcal{P}; \mathcal{D}) + \sum_{i=1}^C \lambda_i c_i(\mathcal{P}) \quad (2.3)$$

PGNNs incorporate domain knowledge by adding soft, physics-based constraints into the cost function, as shown in Equation 2.4:

$$\min_{\mathcal{P}} \mathcal{L}(\mathcal{P}; \mathcal{D}) = \mathcal{L}_{\text{Data}}(\mathcal{P}; \mathcal{D}) + \sum_{c=1}^C \lambda_{\text{Physics}_c} \mathcal{L}_{\text{Physics}_c}(\mathcal{P}; \mathbf{X}, \hat{\mathbf{Y}}) \quad (2.4)$$

where

\mathcal{L} = Total cost function

$\mathcal{L}_{\text{Data}}$ = Supervised, data cost function (e.g. Cross-entropy, Mean Square Error)

$\mathcal{L}_{\text{Physics}_c}$ = PG constraint function

$\lambda_{\text{Physics}_c}$ = Lagrange multiplier for PG constraint c

\mathbf{X} = Model inputs

$\hat{\mathbf{Y}}$ = Model predictions

Note that the PG constraints are unsupervised as they are a function of the parameters given the input data and the model predictions, but not the observations in the training

data. The PG constraints are applied over the outputs of the model to assess the degree of physical inconsistency in the predictions against known physical laws.

In Equation 2.4, the total cost function is uniojective (i.e. $\mathcal{L} \in \mathbb{R}$) where the PG constraints can be thought of as regularizing or “pushing” the model to solutions that fit the data in a physically meaningful manner as shown in Figure 2.1.

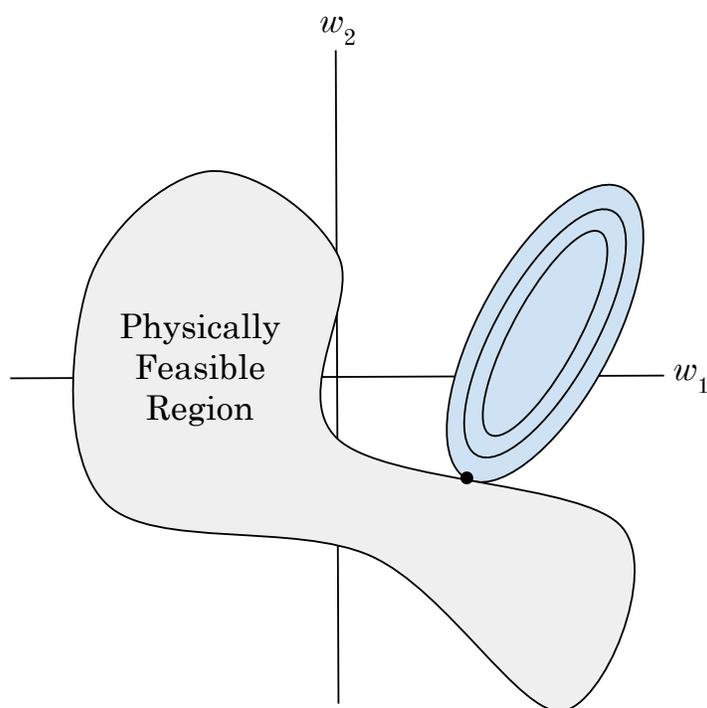


Figure 2.1: Example of the regularization effect caused by PG constraints. The blue, concentric ovals represent the contour level curves of an example, simplified, convex data cost landscape of a neural network; in reality, neural network cost function landscapes are highly nonconvex [23]. Without PG constraints, an optimizer would descend this cost function landscape down to the minima (i.e. the center of the concentric ovals). Since the PG constraints are applied over the predictions in the feature space, the nonlinear nature of the neural networks might cause the constraint region to map to an amorphous manifold in the parameter space, indicated by the gray “Physically Feasible Region”. The PG constraints push the optimizer towards solutions that minimize the data cost function while also being physically consistent as indicated by the black dot at the intersection of the physically feasible constraint region and the data cost function.

2.1.2 Pareto Optimization and Optimality

Given the setup from Equation 2.4, a model must trade-off between minimizing the supervised data cost and each of the PG constraints. This makes PGNN training closely related to multitask learning (MTL), where each cost or constraint component can be viewed as a separate task to optimize [11]. Formally, we can write the total cost as shown in Equation 2.5:

$$\min_{\mathcal{P}} \mathcal{L}(\mathcal{P}; \mathcal{D}, \hat{Y}) = \begin{bmatrix} \mathcal{L}_{\text{Data}}(\mathcal{P}; \mathcal{D}) \\ \mathcal{L}_{\text{Physics}_1}(\mathcal{P}; \mathcal{X}, \hat{Y}) \\ \mathcal{L}_{\text{Physics}_2}(\mathcal{P}; \mathcal{X}, \hat{Y}) \\ \vdots \\ \mathcal{L}_{\text{Physics}_C}(\mathcal{P}; \mathcal{X}, \hat{Y}) \end{bmatrix} \quad (2.5)$$

From a theoretical standpoint, this setup constitutes a multiobjective optimization problem (i.e. $\mathcal{L} \in \mathbb{R}^{C+1}$), in which the goal is to minimize multiple cost functions simultaneously. In such settings, no single solution may minimize all objectives at once. Instead, the aim is to find *Pareto optimal* solutions—those for which improving one objective comes at the expense of worsening other objectives. The collection of such trade-off-optimal solutions forms what is known as the *Pareto front*. However, while the concept of Pareto optimality is useful in understanding the trade-offs inherent in PGNN training, in practice, it is common to scalarize the vector of costs into a single objective using a weighted sum, as is shown in the uniojective cost function from Equation 2.4. We will adhere to that formulation.

2.2 Challenges with Training Physics-Guided Neural Networks

Training neural networks is a challenging, but well-studied problem. The primary challenge lies in the complexity of the optimization process. A typical neural network exhibits a large amount of nonconvexity in the cost function’s landscape due to the nonlinear transformations performed during the forward propagation process [23]. As a result, saddle points impede and local minima can entrap the optimizer during the training process. In PGNNs, these challenges are amplified by the introduction of PG constraints. Revisiting Equation 2.4, PGNNs optimize a combined objective: minimizing both a supervised data cost and one or more PG constraints. Balancing these competing tasks introduces a fundamental tension during training. A naive approach would be to select each $\lambda_{\text{Physics}_i}$ through hyperparameter tuning against a validation dataset. However, the relative importance of different tasks can vary during training, and using static weights can lead to poor performance. In particular, recent studies highlight three major gradient pathologies that arise in MTL scenarios [11, 25, 47]:

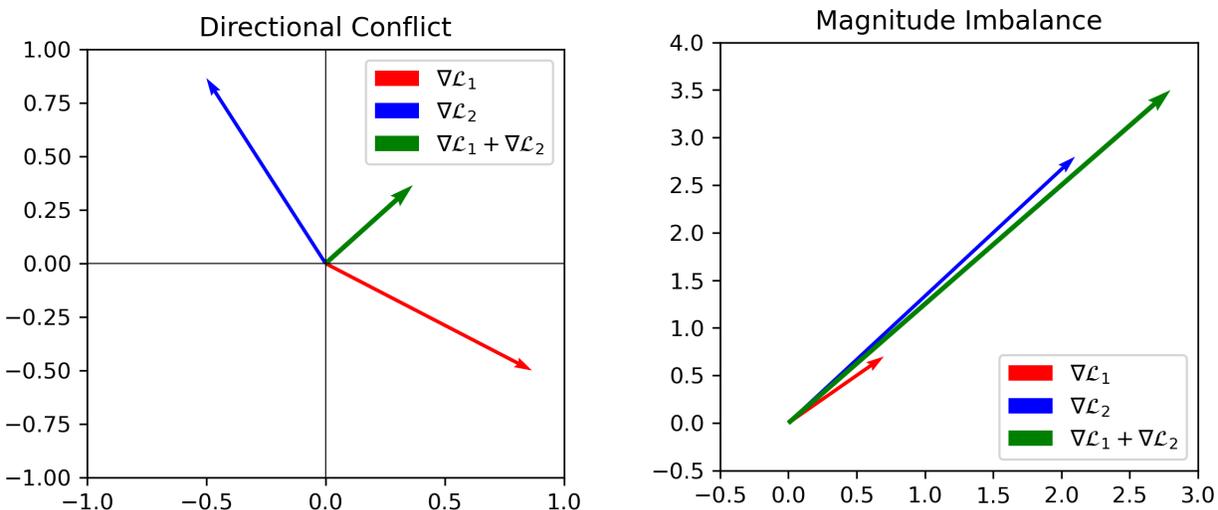
1. **Directional Conflict:** Gradients for different tasks may point in different, conflicting directions. Let $\nabla \mathcal{L}_i$ and $\nabla \mathcal{L}_j$ be the gradient of two different tasks i and j and $\cos(\theta_{i,j})$ be the cosine similarity between the two gradients, computed as shown in Equation 2.6:

$$\cos(\theta_{i,j}) = \frac{\nabla \mathcal{L}_i \cdot \nabla \mathcal{L}_j}{\|\nabla \mathcal{L}_i\| \|\nabla \mathcal{L}_j\|} \quad (2.6)$$

A *gradient conflict* occurs when $\cos(\theta_{i,j}) < 0$ (i.e. when the gradients point in opposing

directions). In this situation, an optimizer will move in the direction of a cumulative and, potentially, suboptimal gradient which neither satisfies task i nor task j . This phenomena is visualized in Figure 2.2a.

2. **Magnitude Imbalance:** PG constraint gradients may differ significantly in magnitude compared to the supervised data gradient. For example, if $\mathcal{L}_{\text{Data}}$ operates on the data in a normalized scale while $\mathcal{L}_{\text{Physics}_c}$ operates on a feature in its unnormalized scale. If left unbalanced, one task may dominate training while others are ignored. This phenomena is visualized in Figure 2.2b.



(a) Conflicting gradient pathology. $\nabla\mathcal{L}_1$ (red) and $\nabla\mathcal{L}_2$ (blue) represent the gradients for two tasks \mathcal{L}_1 and \mathcal{L}_2 , respectively, which point in opposing directions (i.e. $\cos(\theta_{\mathcal{L}_1, \mathcal{L}_2}) < 0$). $\nabla\mathcal{L}_1 + \nabla\mathcal{L}_2$ (green) represents the gradient taken by an optimizer if naively combining the per-task gradients. The overall gradient satisfies neither \mathcal{L}_1 or \mathcal{L}_2 .

(b) Imbalanced gradient pathology. $\nabla\mathcal{L}_1$ (red) and $\nabla\mathcal{L}_2$ (blue) represent the gradients for two tasks \mathcal{L}_1 and \mathcal{L}_2 , respectively, with greatly differing magnitudes (i.e. $\|\nabla\mathcal{L}_1\| \ll \|\nabla\mathcal{L}_2\|$). $\nabla\mathcal{L}_1 + \nabla\mathcal{L}_2$ (green) represents the gradient taken by an optimizer if naively combining the per-task gradients. The overall gradient heavily favors \mathcal{L}_2 .

Figure 2.2: Visualizations of conflicting and imbalanced gradient pathologies in \mathbb{R}^2 .

3. **Task Curvature Mismatch:** Some objectives (e.g., convex, smooth PG constraints) may be easier to optimize than others (e.g., highly nonconvex or stiff constraints).

This mismatch can distort the overall cost landscape, trapping the optimizer in poor regions.

Collectively, these pathologies result in suboptimal, overall gradients and parameter updates which lead to poor convergence during the training process.

2.3 Methods of Training Physics-Guided Neural Networks

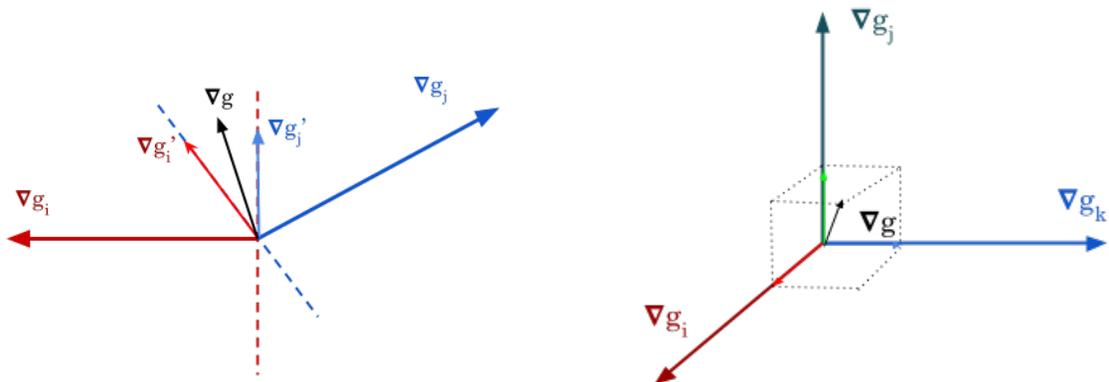
We summarize different approaches to training PGNNs below:

2.3.1 Adaptive Balancing

Adaptive balancing methods operate by directly modifying the per-task cost or gradient during each instance of backpropagation. Early methods included Gradient Normalization (GradNorm) which rescaled the cost of different tasks in proportion to how quickly each task was learning, encouraging slower tasks to “catch up” and faster tasks to slow down so as to prevent one task from dominating the training process [2]. Later methods proposed modifying the direction of gradients given to the optimizer. The Projecting Conflicting Gradients (PCGrad) method examines pairwise cosine similarity between the gradients to check for conflict [47]. If present, the gradients are projected onto each others’ normal plane, thereby eliminating conflicting components and yielding a descent direction that avoids destructive interference between tasks and a more stable optimization behavior.

More recent approaches have introduced strategies to achieve global gradient balance (i.e. without task-specific bias). Impartial Multi-Task Learning (IMTL) proposes computing a

non-conflicting, overall gradient direction whose projection onto each task-specific gradient is equal in magnitude [24]. This approach eliminates task-specific bias by treating all tasks fairly in terms of directional influence. While PCGrad resolves conflicts on a pairwise basis, the Conflict-Free Inverse Gradients (ConFIG) method seeks a globally consistent update direction that maintains a strictly positive inner product with every task gradient [25]. This ensures that all tasks either improve or remain neutral at each step. Moreover, ConFIG introduces an adaptive step-size strategy: when gradient conflicts are severe, it “softens” the update; when gradients align, it permits larger steps. This conflict-aware modulation enables greater stability, particularly when optimizing many PG constraints simultaneously. A visual comparison of PCGrad and IMTL is provided in Figure 2.3a and 2.3b, respectively.



(a) PCGrad method. $\nabla \mathbf{g}_i$ (red) and $\nabla \mathbf{g}_j$ (blue) are gradients for two conflicting tasks. PCGrad projects each pair of conflicting gradients onto each others normal planes, indicated by the dashed red and blue lines, to yield two nonconflicting gradients $\nabla \mathbf{g}'_i$ (red) and $\nabla \mathbf{g}'_j$ (blue). The overall gradient direction taken by the optimizer is $\nabla \mathbf{g}$ (black).

(b) IMTL method. $\nabla \mathbf{g}_i$ (red), $\nabla \mathbf{g}_j$ (green), and $\nabla \mathbf{g}_k$ (blue) are conflicting gradients with differing magnitudes. IMTL creates an overall gradient $\nabla \mathbf{g}$ (black) whose projections onto each of the sub-tasks are equal in magnitude, as indicated by the dotted lines projecting onto each of the sub-tasks.

Figure 2.3: Visualizations of PCGrad and IMTL methods.

An alternative perspective to these geometry-based adaptive methods proposes using cur-

riculum learning. Rather than adjusting gradients during optimization, curricular learning approaches control when and how much certain PG constraints are introduced to the overall cost function. For example, CoPhy-PGNN introduces simple constraints (i.e. those with smoother or more convex optimization landscapes) early and delays more complex or non-convex ones until later in training [11]. This approach relies on theoretical insight into the structure of each constraint and assumes that the model benefits from first building a strong internal representation before confronting more challenging objectives. While powerful, this approach is less general, as it depends on *a priori* knowledge of constraint difficulty to develop an appropriate scheduling of the PG constraints.

2.3.2 Architecture

Another approach to training PGNNs is by utilizing architectural approaches. PGA-LSTM (Physics-Guided Architecture-LSTM) was proposed as a way to ensure physically consistent predictions of lake water temperature by incorporating novel physics-informed connections in the LSTM architecture [3]. However, this approach requires physical laws to be encoded in a manner that can be integrated into the architecture which may be challenging for more complex physical laws.

While not directly related to or targeting the challenge of training PGNNs, there are other architectural developments which may be useful when training MTL neural networks or neural networks with jagged cost landscapes. For example, Residual Networks introduce “skip” connections between layers of the network which “skip” over intermediate hidden layers to allow for better propagation of gradients [17] which has shown to smoothen the overall cost landscape [23]. Other architectural modifications include Cross-Stitch neural networks [31] which leverage two model “backbones” trained on two tasks that are “stitched”

together to share some parameters. This improves generalization performance of the model.

2.4 Applications of Physics-Guided Neural Networks to Lake Ecology Modeling

The application of PGNNs to lake ecology modeling is not entirely novel. Previous studies have explored the application of PGNNs to dissolved oxygen, water temperature, and phosphorus modeling [3, 4, 15, 21]. These studies have focused on applying one to two PG constraints at a time.

2.4.1 Water Temperature Modeling

Early studies focused on predicting water temperature using simple architectures, namely multilayer perceptrons, to regress observed water temperature over a variety of input features such as time, depth, and various meteorological drivers of lake ecology (e.g., air temperature, humidity, solar radiation, etc.) [4, 5]. These models were trained against a supervised data cost function and a single PG constraint to ensure consistent water temperature predictions w.r.t. depth. This constraint leveraged a mass-conservation law that examined the corresponding density of the water given its predicted temperature. We will discuss this constraint later in Section 3.2.1. This work was later expanded to encompass a novel physics-guided architecture for LSTMs to mitigate physical inconsistency when using Monte Carlo dropout for uncertainty quantification as the randomness involved in the dropout process could result in point-estimates that violate the physical constraint [3]. These early works also examined a new paradigm of PGML termed *hybrid-physics-data* (HPD) modeling in which ML models are fed simulated data generated from a process-based model of the target system and

variable of interest. As mentioned in Section 1.3, process-based models encode known information about the physical system of interest but may not model all interactions totally. By providing simulated data to the model, physical knowledge of the system can more easily be embedded into the parameters, leaving the model to implicitly learn the residual, unmodeled physics from the process-based model.

Other studies examined the use of physics-guided recurrent neural network (RNN) for water temperature forecasting with a constraint that leveraged an energy conservation law [21]. This constrain ensured physically plausible heating and cooling of the lake over time as dictated by energy fluxes between meteorological sources and the interactions between the lake's surface and the atmosphere.

2.4.2 Phosphorus Modeling

Hanson et al. [15] demonstrated the use of a process-guided Gated Recurrent Unit-RNN to predict epilimnetic phosphorus concentration. Like previous studies, an HPD modeling approach was utilized. In this study, the GRU-RNN was trained against two physical constraints, a power-scaling cost function and a short-time Fourier transform constraint, to prevent large fluctuations of predicted phosphorus on long horizon windows.

2.4.3 Future Directions for Lake Ecology Modeling

The existing literature in the realm of Ecology Knowledge-Guided Machine Learning (Eco-KGML) shows a clear gap; PGNNs for lake ecology modeling has never been performed with more than two PG constraints at a time. Recent initiatives in the Eco-KGML realm have underscored the need to scale up. In 2024, the development of LakeGPT was announced [22, 30]. LakeGPT is an upcoming foundation model designed for multivariate forecasting of

lakes and reservoirs in the United States with the aim of enhancing understanding of aquatic ecosystems. Other recent works like “Physics-Guided Foundation Model” have explored developing physics-guided foundation models for lake ecology modeling but have not moved upwards of two PG constraints [46]. These initiatives serve as an impetus to address the need of scalable PGNNs in lake ecology modeling.

Chapter 3

Physics-Guided Constraints

In this chapter, we detail the PG constraints used to train our models. We focus on predicting five major variables that summarize lake ecology and water quality: chlorophyll *a* (CHL-*a*), dissolved oxygen (DO), photosynthetic active radiation (PAR), temperature (TEMP), and total phosphorus (TP). For DO, PAR, TEMP and TP, we borrow and expand existing PG constraints from literature and empirical modeling by limnologists, or propose novel PG constraints from known physical laws. For each constrained variable, we describe its importance in lake ecology and then detail the implementation of the constraint.

3.1 Photosynthetic Active Radiation

Photosynthetic active radiation refers to the wavelength range of visible light in the electromagnetic spectrum between 400 and 700 nanometers that can be absorbed by organisms such as plants and phytoplankton for the process of photosynthesis [40]. PAR is a fundamental controlling factor of aquatic ecosystem dynamics. From a physical perspective, the solar energy from PAR that is absorbed by the water affects thermal structure of the lake, stratification and circulation patterns of the water. From a biogeochemical perspective, PAR stimulates or inhibits the growth of organisms in the water which in turn can affect the dynamics of dissolved gasses, such as carbon and oxygen, as well as nutrient cycling [9].

There are a multitude of factors which influence the amount of incident PAR on the surface

of the lake such as atmospheric gas scattering as dictated by Rayleigh’s Law and solar zenith angle (i.e. the angle at which the sun’s rays strike the surface of the water) [40]. Once light strikes the surface of the water, the albedo of the water causes some proportion of reflectance back off of the lake. Our constraints focus on the behavior of light through the water column (i.e. after passing underneath the surface). Electromagnetic theory and Maxwell’s equations indicate that light attenuates in non-vacuous environments. Immediately, we have an understanding that the PAR predicted by a model at depth z must be less than the predicted PAR at depth $z - 1$, allowing us to impose our first constraint for PAR.

3.1.1 Monotonicity Constraint

Let $\hat{\mathbf{y}}_t \in \mathbb{R}^z$ be the predicted PAR at every depth z for time t . We can assemble a monotonicity-enforcing constraint to ensure that $\hat{y}_{z+1}^{(t)} < \hat{y}_z^{(t)}$ as follows:

$$\mathcal{L}_{\text{PAR Monotonicity}}(\hat{\mathbf{y}}_t) = \frac{1}{n_z - 1} \|\text{ReLU}(\Delta \hat{\mathbf{y}}_t)\| \quad (3.1)$$

where

$\hat{\mathbf{y}}_t \in \mathbb{R}^z =$ Predicted PAR at every depth z at time t

$n_z =$ Maximum number of depths (at 0.5 m intervals)

$\Delta =$ First-order difference in PAR (i.e. $y_{z+1}^{(t)} - y_z^{(t)} : 1 \leq z \leq n_z - 1$)

We employ a ReLU ¹ function to penalize positive changes in PAR along the depth axis. The physical inconsistency for a single timestep is determined by computing the norm of the differenced vector and then averaging over its length.

¹ReLU(x) = max(0, x)

3.1.2 Beer-Lambert Constraint

The monotonicity constraint described in Equation 3.1 might guide the model towards predicting PAR in a manner that decreases linearly w.r.t depth. However, according to the Beer-Lambert law, PAR attenuates exponentially w.r.t. depth as shown in Equation 3.2 [40].

$$\text{PAR}(z) = \text{PAR}_0 e^{-kz} \quad (3.2)$$

where

z = Depth

PAR_0 = Surface-level PAR

k = Diffuse attenuation coefficient (in m^{-1})

The diffuse attenuation coefficient k quantifies the rate of change of the exponential decay of PAR through the water. Its value is dependent on the quality of the water and can, for example, span ranges of 0.1 m^{-1} for very clear waters all the way to 18 m^{-1} for heavily muddled waters [40, 41]. However, its value does not have an upper bound.

The effect of various values of k for a surface-level PAR value of $1000 \frac{\mu\text{mol}}{\text{m}^2\text{s}}$ over 10 meters of depth is shown in Figure 3.1.

Enforcing only the monotonicity constraint might result in a model predicting PAR attenuating in a linear decay fashion (as this function is monotonic). Enforcing both constraints penalizes violations in erroneous predictions where PAR increases with depth and ensures that the predicted PAR decrease is exponential.

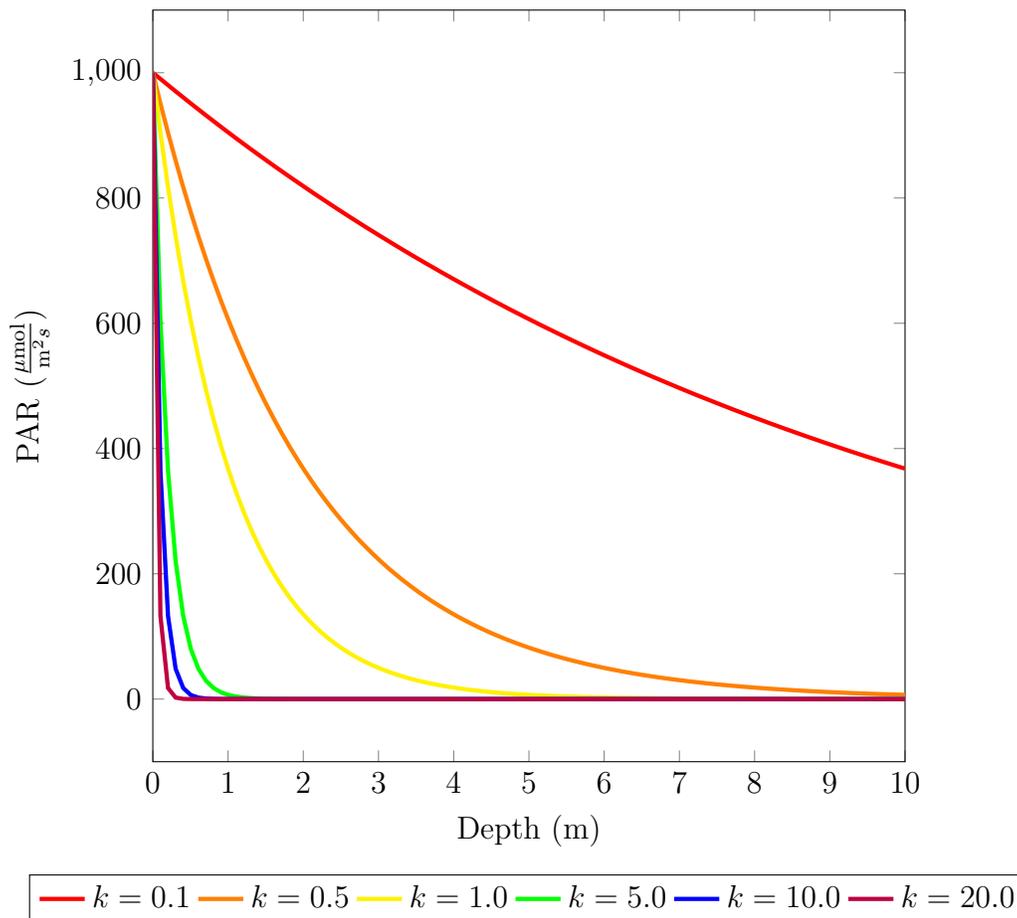


Figure 3.1: Attenuation of PAR over 10 meters given various diffuse attenuation coefficients.

While PAR decreases exponentially with depth, the attenuation coefficient, according to the standard Beer-Lambert law, remains constant. The attenuation and scattering of light through the water column is affected by a multitude of factors such as dissolved organic matter and suspended solids which might concentrate at different depths of the water column. As a result, PAR might attenuate faster at certain depths and slower at other depths. Some studies propose a modification to the standard Beer-Lambert Law where k is a function of depth z , as shown in Equation 3.3, and a multitude of other biochemical features such as chlorophyll-*a*, dissolved organic matter, total suspended solids, etc. at each depth [1]. For example, k might be a linear function of various biochemical components, as shown in

Equation 3.4.

$$\text{PAR}(z) = \text{PAR}_0 e^{-k(z)z} \quad (3.3)$$

$$k(z) = w_1 \text{CHL-}a_z + w_2 \text{DOC}_z + w_3 \text{DOM}_z \quad (3.4)$$

We do not consider this modification for our constraints as it requires data about various biochemical factors of the lake which are often sparsely available w.r.t. time and depth. This would present challenges in determining k when data is not available for a given timestep and depth of predictions. For our purposes, we focus on the Beer-Lambert law described in Equation 3.2. For each timestep of prediction, our models predict k as well as $\hat{\mathbf{y}}_t$. Given the surface-level PAR prediction (i.e. $\hat{y}_0^{(t)}$) and the predicted attenuation coefficient, we compute an ideal PAR decay $\tilde{\mathbf{y}}_t$. From there, we compute the deviation of the predictions from this idealized model as shown in Equation 3.5.

$$\mathcal{L}_{\text{Beer-Lambert}} = \frac{1}{n_z} \|\hat{\mathbf{y}}_t - \tilde{\mathbf{y}}_t\| \quad (3.5)$$

where

n_z = Maximum number of depths

$\hat{\mathbf{y}}_t$ = Predicted PAR at time t

$\tilde{\mathbf{y}}_t$ = Modeled PAR at time t as determined by Equation 3.2

3.2 Temperature

Water temperature is considered an ecological “master factor” because it governs a wide range of biological and ecological processes in lakes [26]. It directly influences the growth, survival, and reproduction of aquatic organisms which are sensitive to thermal habitat conditions [38]. As lake temperatures rise due to climate change, species distributions can shift, often enabling warm-water and invasive species to thrive at the expense of native cold-water taxa [37, 39]. These thermal shifts are also associated with increased frequency and intensity of harmful algal blooms, which degrade water quality, produce toxins, and disrupt aquatic food webs [16, 35]. Beyond biological impacts, temperature plays a pivotal role in regulating key physical and chemical processes, such as dissolved oxygen availability, stratification patterns, and the rate of biogeochemical cycling. The presence and strength of thermal stratification in lakes, which determines vertical habitat structure, are especially sensitive to temperature dynamics. Stratification limits vertical mixing, affecting nutrient distribution and the habitability of deeper waters. As such, understanding the vertical and temporal variability of temperature is essential for assessing ecological conditions, predicting species responses to climate change, and informing lake management and conservation strategies [18].

3.2.1 Temperature-Density-Depth Monotonicity Constraint

The density of pure water is related to the temperature of the water according to the following equations:

$$\rho(T) = 1000 \times \left[1 - \frac{(T + 288.9414)(T - 3.9863)^2}{508929.2(T + 68.12963)} \right] \quad (3.6)$$

$$\rho(T) = 999.974950 \times \left[1 - \frac{(T + 301.797)(T - 3.983035)^2}{522528.9(T + 69.34881)} \right] \quad (3.7)$$

where

T = Temperature (in Celsius)

ρ = Density (in kg/m³)

Equation 3.6 is given by Martin and McCutcheon [28] while Equation 3.7 is provided by Tanaka et al. [44]. A visualization of both conversion equations is given in Figure 3.2.

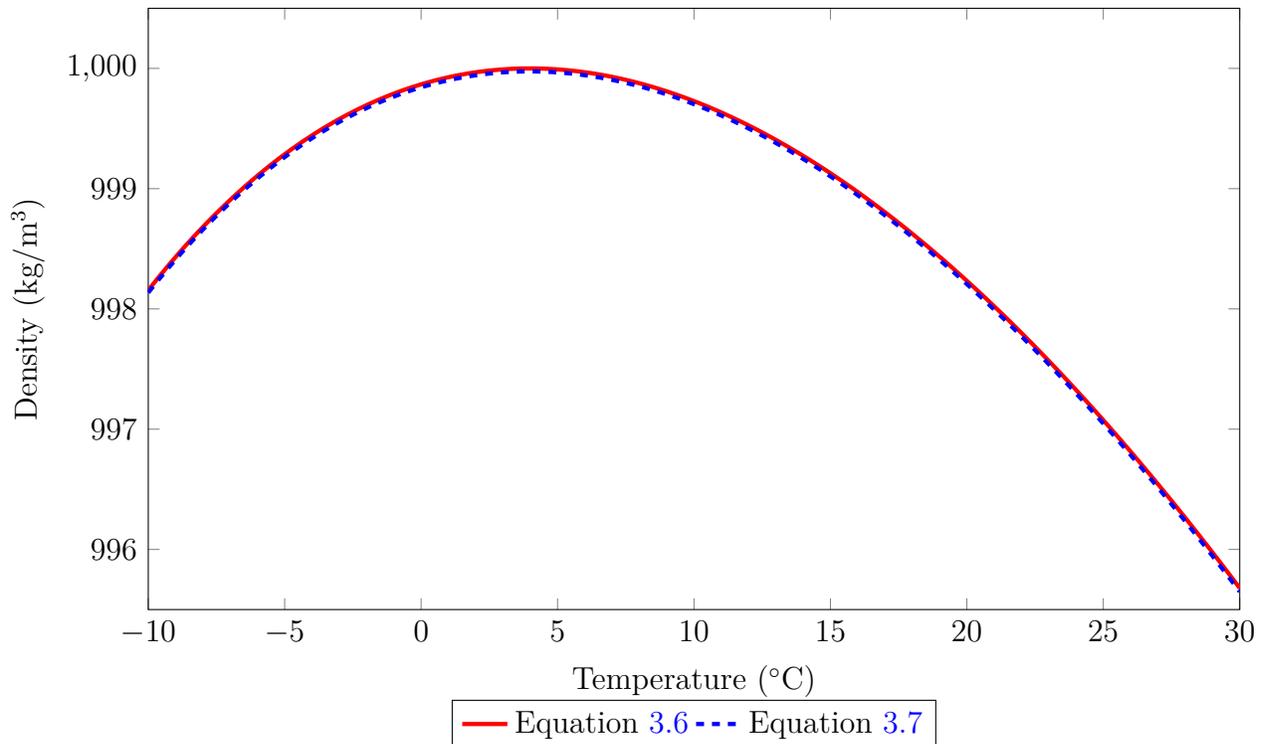


Figure 3.2: Density of water as a function of temperature, showing two different empirical equations for comparison.

Archimedes’s principle states that a denser fluid will sink below a sparser fluid [19]. This physical fact enables the development of a PG constraint that ensures predicted water density increases monotonically with depth—a behavior commonly observed during thermal stratification in lakes. Specifically, prior works have enforced that the density corresponding to predicted water temperature profiles should increase with depth, such that denser water resides beneath sparser water [4, 5, 21] as shown in Figure 3.3.

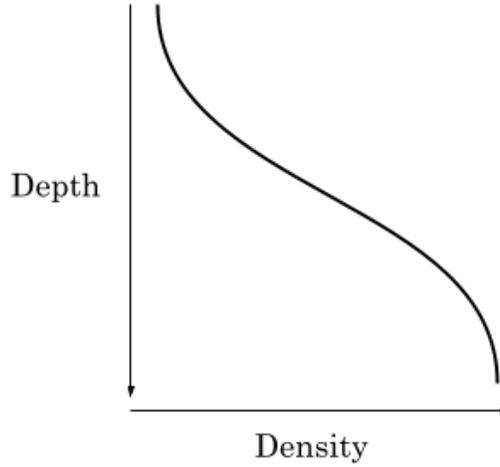


Figure 3.3: Depth (measured positively in a downwards direction) versus density plot. Warmer, sparser water resides towards the surface of the lake (i.e. the top of the plot) while colder, denser water resides towards the bottom of the lake.

Let $\hat{\mathbf{y}}_t \in \mathbb{R}^z$ be the predicted water temperature across all depths z at time t and $\hat{\boldsymbol{\rho}}_t(\hat{\mathbf{y}}_t) \in \mathbb{R}^z$ be corresponding density of the predicted water temperature across all depths z at time t .

We can enforce a monotonicity constraint such that $\hat{\rho}_{z+1}^{(t)} \geq \hat{\rho}_z^{(t)}$

$$\mathcal{L}_{\text{TEMP Monotonicity}}(\hat{\mathbf{y}}_t) = \frac{1}{n_z - 1} \|\text{ReLU}(\Delta \hat{\boldsymbol{\rho}}_t)\| \quad (3.8)$$

where

$$\hat{\boldsymbol{\rho}}_t \in \mathbb{R}^z = \text{Predicted density at every depth } z \text{ at time } t$$

n_z = Maximum number of depths

Δ = First-order difference in TEMP (i.e. $\hat{\rho}_{z+1}^{(t)} - \hat{\rho}_z^{(t)} : 1 \leq z \leq n_z - 1$)

We employ a ReLU function to penalize negative changes in density along the depth axis. The physical inconsistency for a single timestep is determined by computing the norm of the differenced vector and then averaging over its length.

3.2.2 Energy Conservation Constraint

We also include a constraint to ensure physically consistent predictions of temperature across time. The heating and cooling of a lake over time is dictated by energy fluxes from meteorological sources. Jia et al. [21] explored the challenge of enforcing physical constraints to ensure physically consistent lake heating and cooling. We detail their approach and offer an expansion of past work which adapts the physical constraint for a more diverse scenario. The following equations are given from [21].

The total amount of energy in a lake is given according to the following equation:

$$U_t = c_{\text{H}_2\text{O}} \sum_{z=0}^Z a_{z,t} \hat{\rho}_{z,t} \hat{y}_{z,t} dz \quad (3.9)$$

where

U_t = Total energy of the lake at time t (in Joules)

$c_{\text{H}_2\text{O}}$ = Specific heat capacity of water (=4186) (in Joules per kilogram Celsius)

a = Cross-sectional area of the lake (in square meters)

$\hat{\rho}$ = Predicted water density (in kg/m^3), as described in Equation 3.6

\hat{y} = Predicted water temperature

dz = Layer thickness (=0.5) (in meters)

The cross-sectional area of the lake is obtained from hypsometry data which can be cumbersome to work with. Instead, we consider a simplifying assumption where we estimate cross-sectional area using an inverted cone approximation. We consider the surface area of the lake, in square meters to be, the bottom surface of the cone and the maximum depth as the depth/height of the cone.

The day-to-day change in the total thermal energy of the lake is dictated by the net energy fluxes from meteorological inputs and thermal interactions according to the following equation:

$$\Delta U_t = R_{\text{LW, in}}(1 - \alpha_{\text{LW}}) + R_{\text{SW, in}}(1 - \alpha_{\text{SW}}) - R_{\text{LW, out}} - E - H \quad (3.10)$$

where

ΔU_t = Change in lake energy at time t

$R_{\text{LW, in}}$ = Inflowing longwave radiation, obtained from driver data (in W/m²)

$R_{\text{SW, in}}$ = Inflowing shortwave radiation, obtained from driver data (in W/m²)

α_{LW} = Albedo of surface water reflecting longwave radiation

α_{SW} = Albedo of surface water reflecting shortwave radiation

$R_{\text{LW, out}}$ = Back-radiating longwave radiation (in W/m²)

E = Latent evaporative heat flux (in W/m²)

H = Sensible heat flux (in W/m^2)

Since the temperature of the water is not absolute zero (i.e. 0 K), the lake radiates back some energy to the air. The back-radiating longwave radiation, $R_{\text{LW, out}}$, is computed as follows:

$$R_{\text{LW, out}} = \epsilon_s \delta_s T_s^4 \quad (3.11)$$

where

$R_{\text{LW, out}}$ = Back-radiating longwave radiation (in W/m^2)

ϵ_s = Emissivity of water (=0.97)

δ = Stefan-Boltzmann constant ($=5.6697 \times 10^{-8} \text{ W}/\text{m}^2\text{K}^{-4}$)

The ambient energy of the air can cause some water to vaporize. The latent evaporative heat flux, E , is computed as follows:

$$E = -\rho_{\text{Air}} C_E \nu \kappa_{10} \frac{\omega}{p} (e_s - e_a) \quad (3.12)$$

where

ρ_{Air} = Air density

C_E = Bulk aerodynamic coefficient for latent evaporative heat transfer (=0.0013 as obtained from Freitag and McFadden [12])

ν = Latent heat of vaporization

κ = Wind speed corrected for 10 meters above the lake surface ²

²Our meteorological driver data was obtained from the National Aeronautical and Space Administration's North American Land Data Assimilation System (NLDAS) which already reports wind speeds referenced to 10 meters from the surface. The correction was not applied to the driver data.

ω = Ratio of the molecular mass of water (=18.0153 g/mol) to the molecular mass of dry air (=28.9634 g/mol)

p = Surface air pressure (in hectopascals)

e_s = Saturated vapor pressure

e_a = Air vapor pressure

The saturated vapor pressure is computed as follows:

$$e_s = 10^{9.28603523 \times \frac{2322.37885}{T_s + 273.15}} \quad (3.13)$$

where

e_s = Saturated vapor pressure (in millibars³)

T_s = Surface water temperature (in Celsius)

The air vapor pressure is computed as follows:

$$e_a = \left(S_{RH} \frac{RH}{100} \right) e_s \quad (3.14)$$

where

e_a = Air vapor pressure (in millibars or, equivalently, hPA)

S_{RH} = Relative humidity scaling factor (=1)

RH = Relative humidity, obtained from driver data

³1 mb = 1 hPa

e_s = Saturated vapor pressure (in millibars or, equivalently, hPA)

Since air and water are both fluids in contact with each other, there is some heat flux that occurs. The sensible heat flux, H , is computed as follows:

$$H = -\rho_{\text{Air}}c_{\text{Air}}C_H\kappa_{10}(T_s - T_a) \quad (3.15)$$

where

ρ_{Air} = Air density

c_{Air} = Specific heat capacity of air (=1005)

C_E = Bulk aerodynamic coefficient for sensible heat transfer (=0.0013 as obtained from Freitag and McFadden [12])

κ = Wind speed corrected for 10 meters above the lake surface

T_s = Surface water temperature (in Kelvin)

T_a = Air temperature (in Kelvin)

The previous study by Jia et al. [21] applies only in ice-free situations as the presence of ice can drastically affect the dynamics of the energy fluxes. We expand upon this constraint to factor in icing conditions as this would allow us to consider the full datetime range of temperature data rather than data from ice-free periods. Rather than explicitly model ice growth as is done in process-based models such as GLM 3.0 [18], which can be complicated as different kinds of ice growth need to be modeled, we make the following simplifying assumptions:

1. A surface temperature that is less than or equal to 0°C indicates ice conditions.

2. The latent evaporative and sensible heat flux terms, E and H, are negligible in icing conditions.
3. The available energy to heat the lake is affected primarily by the albedo of the ice. Specifically $\alpha_{(\cdot),\text{Ice}} \gg \alpha_{(\cdot),\text{H}_2\text{O}}$

For our constraint, we setup a binary flag to indicate icing conditions. This allows us to adaptively enable or disable certain configurations of how energy fluxes are calculated or enable/disable the calculation of certain fluxes altogether.

$$\begin{aligned}
 \widetilde{R}_{\text{LW},\text{in}} &= R_{\text{LW},\text{in}}(1 - \alpha_{\text{LW},\text{Ice}})(\text{Ice}) + R_{\text{LW},\text{in}}(1 - \alpha_{\text{LW},\text{H}_2\text{O}})(1 - \text{Ice}) \\
 \widetilde{R}_{\text{SW},\text{in}} &= R_{\text{SW},\text{in}}(1 - \alpha_{\text{SW},\text{Ice}})(\text{Ice}) + R_{\text{SW},\text{in}}(1 - \alpha_{\text{SW},\text{H}_2\text{O}})(1 - \text{Ice}) \\
 \widetilde{E} &= E(1 - \text{Ice}) \\
 \widetilde{H} &= H(1 - \text{Ice})
 \end{aligned}
 \left\{ \begin{array}{l} \text{Case 1, } T_0 \leq 0 \rightarrow \text{Ice} = 1 \\ \text{Case 2, } T_0 > 0 \rightarrow \text{Ice} = 0 \end{array} \right.$$

We do not consider heat fluxes between the the lake and the sediment/soil of the lakebed or heat flux due to groundwater or surface inflows and outflows.

From here, our new equation for computing the change in lake energy is similar to Equation 3.10.

$$\widetilde{\Delta U}_t = \widetilde{R}_{\text{LW},\text{in}} + \widetilde{R}_{\text{SW},\text{in}} - R_{\text{LW},\text{out}} - \widetilde{E} - \widetilde{H} \quad (3.16)$$

where

$\widetilde{\Delta U}_t$ = Ice-adjusted change in lake energy at time t

$\widetilde{R}_{\text{LW},\text{in}}$ = Ice-adjusted inflowing longwave radiation, obtained from driver data (in

W/m^2)

$\widetilde{R}_{\text{SW},\text{in}}$ = Ice-adjusted inflowing shortwave radiation, obtained from driver data (in W/m^2)

$R_{\text{LW},\text{out}}$ = Back-radiating longwave radiation (in W/m^2)

\widetilde{E} = Ice-adjusted latent evaporative heat flux (in W/m^2)

\widetilde{H} = Ice-adjusted sensible heat flux (in W/m^2)

$\widetilde{R}_{\text{LW},\text{in}}$ and $\widetilde{R}_{\text{SW},\text{in}}$ constitute energy influx, F_{in} , while $R_{\text{LW},\text{out}}$, \widetilde{E} and \widetilde{H} constitute energy efflux, F_{out} . From here, we can assemble our energy conservation constraint as shown in Equation 3.17.

$$\mathcal{L}_{\text{Energy Conservation}}(\widehat{\mathbf{Y}}_t) = \frac{1}{T} \sum_{t=1}^T \text{ReLU}[|\widetilde{\Delta U}_t - (F_{\text{in}} - F_{\text{out}})| - \tau_{\text{EC}}] \quad (3.17)$$

where

$\widehat{\mathbf{Y}}_t$ = Matrix of predicted water temperature across several timesteps T and depths Z (i.e. $\widehat{\mathbf{Y}}_t \in \mathbb{R}^{Z \times T}$)

T = Number of predicted timesteps

$\widetilde{\Delta U}_t$ = Ice-adjusted change in lake energy at time t from Equation 3.16

F_{in} = Ice-adjusted energy influx

F_{out} = Ice-adjusted energy efflux

τ_{EC} = Cost threshold for energy conservation added to account for process noise arising from unmodeled physics. This parameter is obtained by generating simulation data

for the given lake, computing the aforementioned fluxes and then selecting the largest value of $|\Delta U_t - (F_{\text{in}} - F_{\text{out}})|$. Since we do not generate simulation data, we utilize a value of 24, following Jia et al. [21].

3.3 Dissolved Oxygen

Dissolved oxygen (DO) dynamics in lakes are driven by a balance between oxygen sources, primarily net primary production from photosynthetic organisms, and sinks such as organismal respiration and microbial decomposition [27]. The solubility of gases in water is dictated by Henry’s Law which states that the amount of gas that can be dissolved is proportional to its partial pressure which itself is determined by factors such as atmospheric pressure, water vapor pressure, salinity and unideal gas behavior [27]. However, biological and physical processes can lead to supersaturation or depletion events. In such situations, applying Henry’s Law would result in violations of the PG constraint.

To enforce physically consistent DO dynamics in our model, we adapt a power-scaling constraint from Hanson et al. [15]. Originally developed for phosphorus, this constraint penalizes excessive short-term variability by ensuring that the magnitude of daily changes remains within a physically reasonable bound. Specifically, we constrain the daily fluctuations in the lake-integrated DO concentration such that they do not exceed the 80th percentile of forecasted daily DO changes.

We first integrate the predicted DO concentration over depth and lake area using an inverted cone approximation for hypsometry, consistent with the approach used in Section 3.2.2:

$$\hat{y}_t = \sum_{z=0}^Z \hat{y}_z^{(t)} a_z dz \quad (3.18)$$

where

\hat{y}_t = Total predicted lake DO at time t

$\hat{y}_z^{(t)}$ = Predicted DO at depth z and time t

a_z = Cross-sectional lake area

dz = Layer thickness (0.5 meter)

For our research, we predict DO at a daily timescale; this constraint penalizes any daily fluctuations that exceed the empirical 80th percentile threshold:

$$\frac{1}{T} \sum_{t=1}^T \text{ReLU} [|\hat{y}_{t+1} - \hat{y}_t| - p_{0.8}(\Delta \mathbf{y})] \quad (3.19)$$

where:

$\hat{y}_z^{(t)}$: Predicted DO concentration at depth z and time t

a_z : Cross-sectional area at depth z (via inverted cone approximation)

dz : Depth layer thickness (0.5 meters)

$p_{0.8}(\Delta \mathbf{y})$: 80th percentile of observed daily changes in integrated DO

This constraint encourages physically plausible, temporally smooth DO predictions while allowing for natural variability during ecologically active periods.

3.4 Total Phosphorus

Phosphorus is a critical nutrient in freshwater ecosystems and often the limiting factor for primary productivity. Unlike other biologically essential elements that are relatively abun-

dant, phosphorus is scarce in the hydrosphere. It plays key roles in energy transfer, genetic material synthesis, and membrane structure. This scarcity makes phosphorus a primary driver of trophic status, with excessive loading leading to eutrophication, algal blooms, and subsequent oxygen depletion [6].

Despite its importance, our modeling of total phosphorus (TP) in this work is limited by the lack of inflow and outflow data, precluding the use of steady-state mass balance models such as the Vollenweider approach shown in Equations 3.20 and 3.21 [45]. Instead, we adopt a temporal smoothness constraint, namely the power scaling constraint borrowed from Hanson et al. [15] and described earlier in Section 3.3, to regularize short-term variability in predicted phosphorus levels.

$$\frac{\Delta M}{\Delta t} = I - O - (S - R) \quad (3.20)$$

$$\frac{\bar{d}[\bar{P}]_\lambda}{dt} = \left(\frac{1}{\bar{\tau}_w}\right) [\bar{P}]_i - \left(\frac{1}{\bar{\tau}_p}\right) [\bar{P}]_\lambda \quad (3.21)$$

where

$$\frac{\Delta M}{\delta t} = \text{Change in storage of nutrient M over time } \Delta t$$

I = External nutrient load

O = Nutrient loss by outflow

S = Nutrient loss to sediments

R = Nutrient regeneration from sediments (internal loading)

$[\bar{P}]_\lambda$ = Average (total) lake concentration of dissolved and particulate phosphorus components

$[\bar{\text{P}}]_i$ = Average inflow concentration of total phosphorus

$\bar{\tau}_p$ = Average phosphorus residence time

$\bar{\tau}_w$ = Average water residence time

Hanson et al. [15] additionally employs a short-time Fourier transform constraint due to extremely long forecast horizons, namely over several weeks, but we do not consider this constraint as our predictions do not exceed more than 3 weeks into the future. This power scaling constraint penalizes unrealistically large changes in lake-integrated TP between consecutive days:

$$\hat{y}_t = \sum_{z=0}^Z \hat{y}_z^{(t)} a_z dz \quad (3.22)$$

$$\frac{1}{T} \sum_{t=1}^T \text{ReLU} [|\hat{y}_{t+1} - \hat{y}_t| - p_{0.8}(\Delta \mathbf{y})] \quad (3.23)$$

$\hat{y}_z^{(t)}$: Predicted TP concentration at depth z and time t

a_z : Cross-sectional area at depth z (via inverted cone approximation)

dz : Depth layer thickness (0.5 meters)

$p_{0.8}(\Delta \mathbf{y})$: 80th percentile of observed daily changes in integrated TP

This constraint serves as a first step toward regularizing phosphorus dynamics without requiring explicit hydrological data. While more sophisticated constraints could include sediment release rates or re-suspension dynamics, the current approach ensures that model outputs remain within empirically realistic variability ranges.

Chapter 4

Data and Methodology

In this chapter, we describe the data used for our modeling, model architectures and design considerations as well as our strategies and experimental design for answering the following research question: “Can PGNNs, trained without additional simulation-derived data, be effectively scaled to incorporate multiple physics constraints in order to improve both within-distribution prediction accuracy and out-of-distribution generalization in data-scarce settings, such as those encountered in lake ecology modeling?”

4.1 Data

We leveraged the LakeBeD-US: Computer Science Edition (CSE) dataset by Pradhan et al. [36].

4.1.1 Overview

LakeBeD-US: CSE is derived from the LakeBeD-US: Ecology Edition dataset which contains lake ecology and water quality time series and vertical profile data from 21 different lakes across the United States dating as far back as 1981. LakeBeD-US measures 17 different variables at a minutely resolution for high-frequency, sensor data and bi-weekly/daily resolutions for manually sampled, low-frequency data. The original LakeBeD-US: Ecology

Edition dataset formats data in a long format where variables were stored in a single column. LakeBeD-US: CSE converts this format into a tabular format where the various features are stored as individual columns and each unique observation taken for a given datetime and depth is a row.

It is also convenient to envision the data for a given lake as a third-order tensor as shown in Figure 4.1.

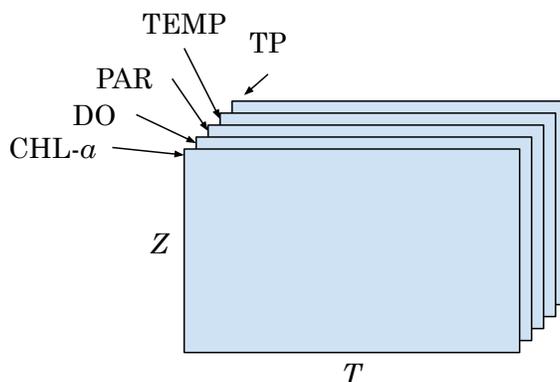


Figure 4.1: Third-order tensor of dimension $V \times Z \times T$ representing the LakeBeD-US data. The slices represent different variables of interest, as described at the start of Chapter 3. The rows are the depths (Z) and the columns are the timesteps (T).

4.1.2 Preprocessing

The LakeBeD-US: Computer Science Edition dataset was preprocessed in an identical fashion to McAfee et al. [29]’s benchmark, according to the following steps:

1. **Flag Selection:** In addition to providing datetime and depth information, LakeBeD-US also records a flag for each observation to indicate any abnormalities with the observation. We select observations marked with the flags listed in Table 4.1 as valid observations and omit other observations.

Flag	Description
0	No flag
5	Average of duplicate analyses
10	Nonstandard methods
19	Value below detection limit; set to zero
23	Negative value set to zero
25	Sensor was off during part of the averaged period
32	Date is accurate but time is inaccurate
43	Sample run using NPOC (non-purgeable organic carbon) method due to high inorganic carbon values
47	Flagged with no explanation
51	Secchi depth hit bottom (calculated for NEON Lakes only)
52	Unknown depth near surface. Labeled as 0.5m.

Table 4.1: Acceptable flags from LakeBeD-US.

- Datetime and Depth Standardization:** Since LakeBeD-US’s data is sourced by manual sampling and sensor data, irregular sampling, sensor malfunctions and off-seasons can lead to missing datetimes and depths of recorded data. For each lake’s data, we assemble a uniform range of datetimes and depths. We create a datetime scale at the minutely resolution for the high-frequency data and a daily resolution for the low-frequency data to ensure no skipped datetimes and depths.
- Merging:** We consider only the datetime range of the high-frequency data so as to not introduce more missing values in the data. We merge the low-frequency data with the high-frequency data to fill datetimes and depths with missing values and then aggregate the data to a daily median resolution.

4.1.3 Similarity Analysis

For simplicity, we consider a subset of lakes on which to train our PGNNs. Ideally, PGNNs should be able to generalize better to unseen data distributions. To test this, we consider a

set of lakes for training individual PGNN models and then another, outlier lake for zero-shot evaluation with no fine-tuning. We perform k -means clustering of the 12 lakes across the attributes listed in Table 4.3 before generating a dimensionality reduced representation with Principal Components Analysis (PCA). PCA was chosen to allow for interpretability of the low-dimensional space embeddings. The loadings for the first two principal components are provided in Table 4.2. Principal component (PC) 1 has the highest magnitude loadings on mean and maximum depth as well as residence time which may be interpreted as a lake morphometry and dynamics factor where higher values are associated with larger, deeper lakes with longer residence times. PC 2 has the highest magnitude loadings on longitude, latitude and area which could be interpreted as a geographical factor. High PC2 scores might correspond to lower elevation, eastern lakes, while low PC2 scores correspond to higher elevation, western lakes.

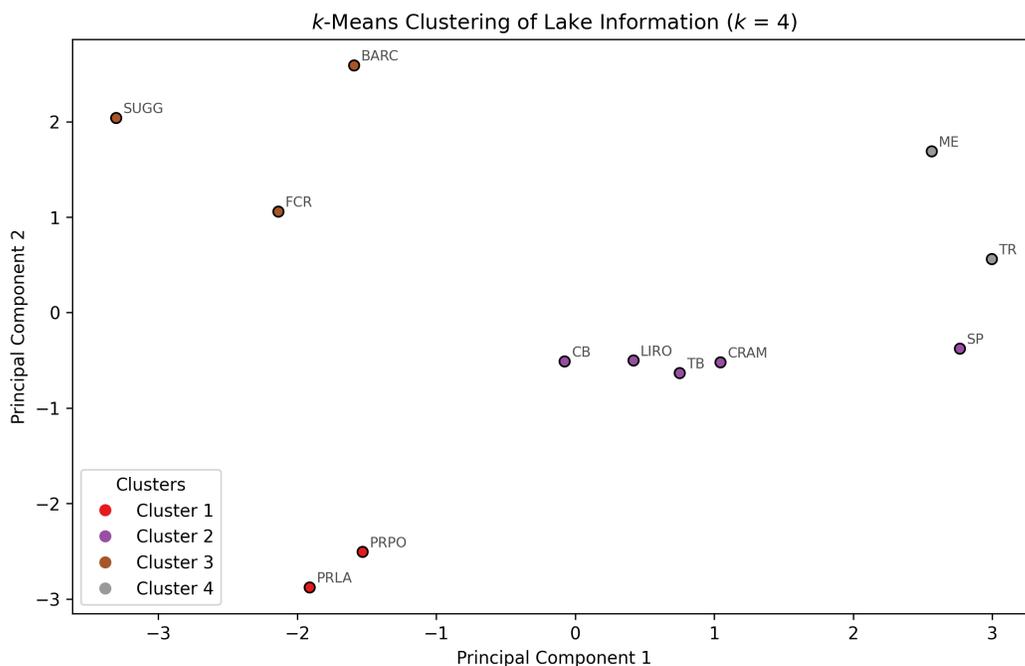


Figure 4.2: PCA biplot of lake similarity with $k = 4$, as obtained by maximizing silhouette score.

<u>Attribute</u>	<u>PC 1</u>	<u>PC 2</u>
Latitude	0.3111	-0.4758
Longitude	-0.1427	0.5323
Elevation	0.1700	-0.5055
Surface Area	0.3127	0.2512
Mean Depth	0.4639	0.1827
Max Depth	0.4488	0.1772
Residence Time	0.3543	-0.1229

Table 4.2: Loadings of first two principal components. Darker shades correspond to higher magnitude loading, indicating more dependence on the attribute.

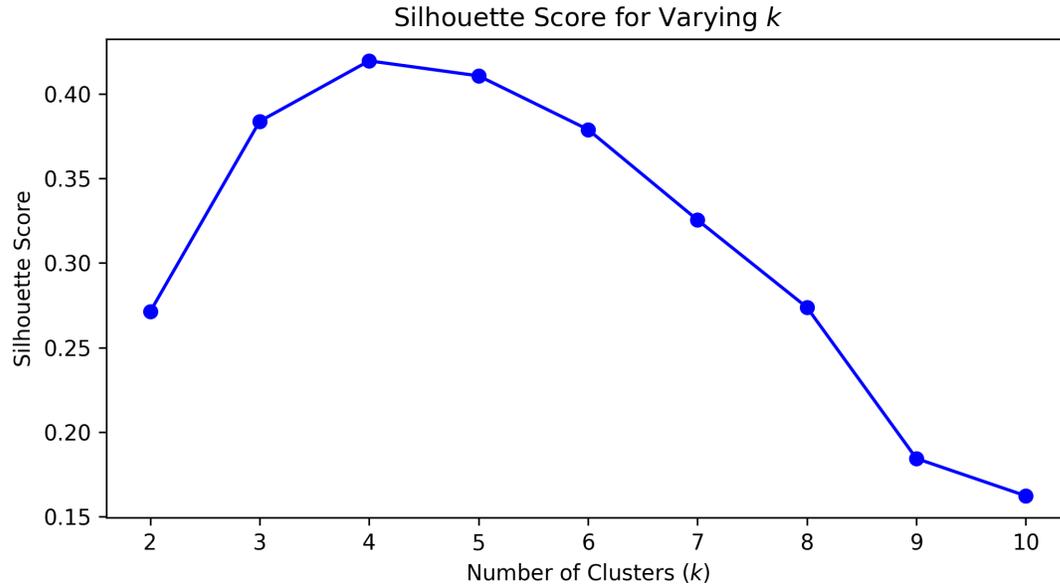


Figure 4.3: Silhouette scores from the lake similarity analysis. A higher silhouette score indicates a more ideal selection for the number of clusters.

From this analysis, we randomly select three lakes for training our models on: Falling Creek Reservoir (FCR), Trout Bog (TB) and Trout Lake (TR). We also selected Prairie Lake (PRLA) as our lake for zero-shot evaluation as it is the more outlier lake in Cluster 1 from Figure 4.2. The geographical and morphological properties used in the PCA analysis for these chosen lakes are presented in Table 4.3. We hypothesize that closer lakes may exhibit similar ecological dynamics and by extension, similar data distributions which might

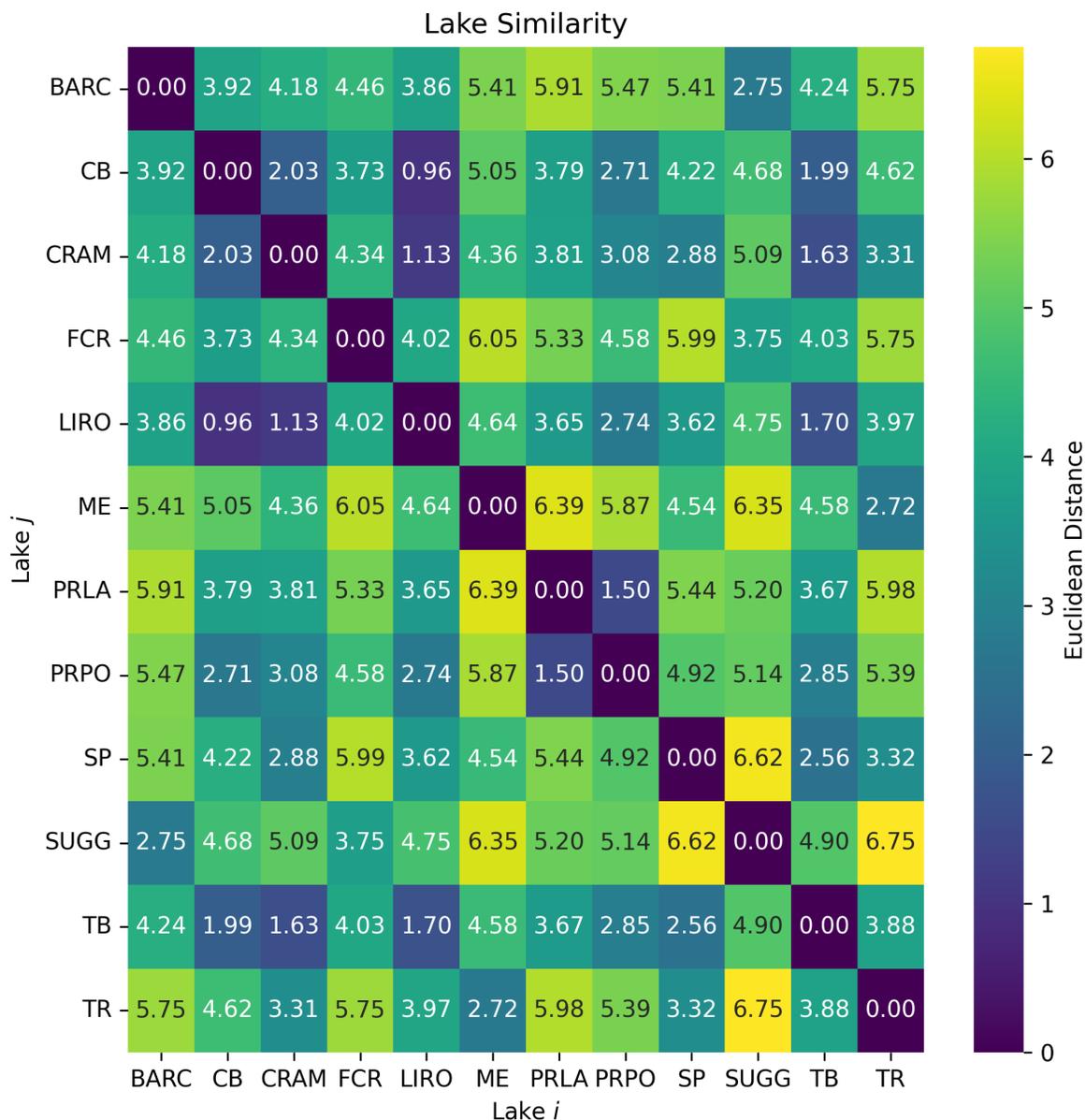


Figure 4.4: Lake similarity, as determined by Euclidean distance, in the scaled, full feature space. The darker the color, the more similar the lakes are to each other.

lead to improved zero-shot performance. Given the lake similarities from Figure 4.4, we hypothesize TB to perform the best on PRLA during zero-shot evaluation, followed by FCR and TR. However, we should note that while PCA analysis may reveal similarities between lakes with regards to physical attributes, there may be other factors at play with regards

to their ecological dynamics. For example, whether a lake is eutrophic or oligotrophic or whether or not the lake has fish or other wildlife may contradict the similarity obtained from analyzing the lakes with PCA.

Attribute	Falling Creek Reservoir	Trout Bog	Trout Lake	Prarie Lake
Abbreviation	FCR	TB	TR	PRLA
Latitude	37.3033	46.0413	46.0293	47.1591
Longitude	-79.8375	-89.6863	-89.6650	-99.1139
Elevation (m)	507.61	493.5	491.71	562.77
Surface Area (ha)	12.1371	1	1583	23
Mean Depth (m)	2.5446	5.6	14.6	1.6932
Max Depth (m)	9.3	7.9	35.7	4
Residence Time (yr)	0.68	6.4	4.5	3.8

Table 4.3: Selected lakes from LakeBeD-US. Prarie Lake is selected as the outlier, zero-shot evaluation lake.

4.2 Models

We leveraged MLP and Long Short-Term Memory Recurrent Neural Network (LSTM-RNN) architectures in our studies. We detail specific design decisions for each architecture below.

4.2.1 Multilayer Perceptron

Our MLP design closely follows that of Daw et al. [4, 5] used for the task of lake temperature modeling. The input features of the model are given in Table 4.4.

Feature	Description
1	Sine Cyclical Encoding of Month
2	Cosine Cyclical Encoding of Month
3	Sine Cyclical Encoding of Day of Year
4	Cosine Cyclical Encoding of Day of Year
5	Depth (m)
6	Air Temperature ($^{\circ}\text{C}$)
7	Long-wave Radiation (W/m^2)
8	Rain per Day (m/Day)
9	Relative Humidity
10	Short-wave Radiation (W/m^2)
11	Specific Humidity (kg/kg)
12	Surface Pressure (Pa)
13	Wind speed (m/s)

Table 4.4: Input features for the MLP architecture. All features except the sine and cosine cyclical encodings (CE) are Z-score normalized. The equations for generating the sine-cosine CEs are shown in Equations 4.1 and 4.2. Visualizations of the encodings are shown in Figure 4.5.

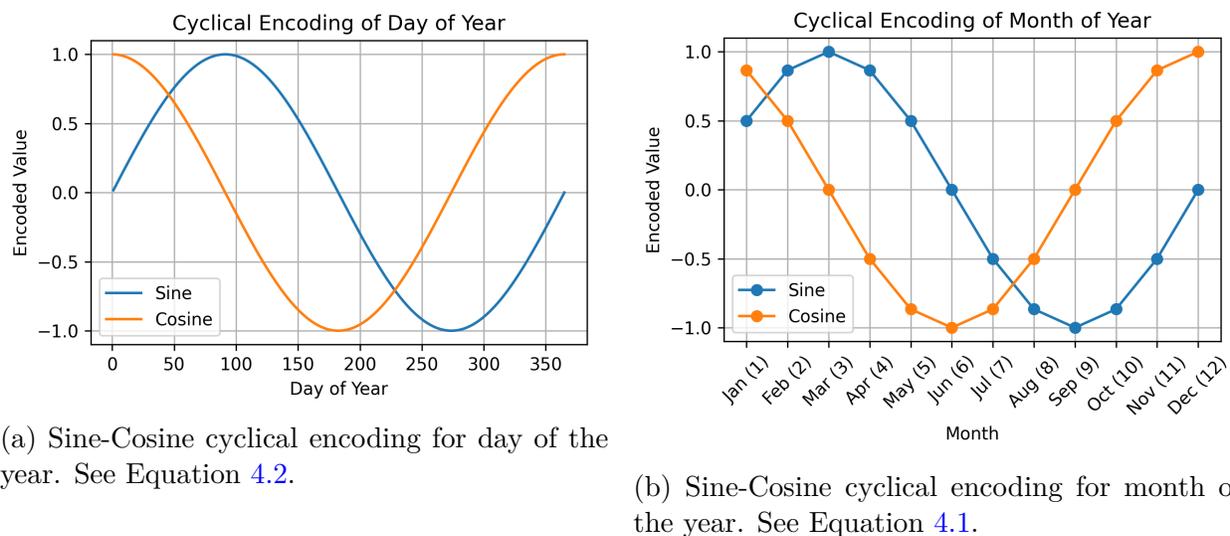


Figure 4.5: Cyclical encoding visualizations.

$$\text{CE}_{\text{Month}}(i) = \sin\left(2\pi\frac{i}{12}\right); \text{CE}_{\text{Month}}(i) = \cos\left(2\pi\frac{i}{12}\right) : 1 \leq i \leq 12 \quad (4.1)$$

$$\text{CE}_{\text{Day of Year}}(i) = \sin\left(2\pi\frac{i}{365}\right); \text{CE}_{\text{Day of Year}}(i) = \cos\left(2\pi\frac{i}{365}\right) : 1 \leq i \leq 365 \quad (4.2)$$

The MLP predicts our five features of interest as well as the PAR attenuation coefficient for each given time and depth. No imputation is required in this approach as a masked RMSE cost calculation is employed to only calculate loss between predictions and observations that are present. We fix our MLP size to contain three hidden layers with 15 neurons per layer.

4.2.2 Spatiotemporal Sequence-to-Sequence Model

We additionally consider a seq2seq model for spatiotemporal, multivariate time series prediction. A full schematic of our model is shown in [Figure 4.6](#).

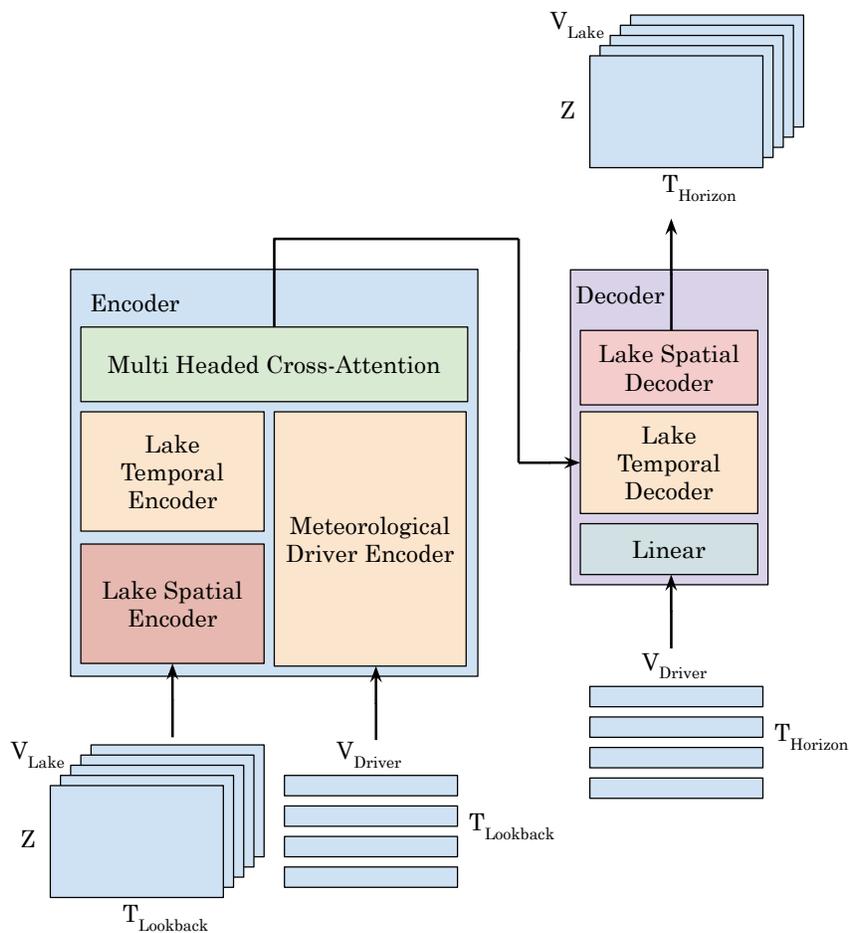


Figure 4.6: Schematic of the spatiotemporal seq2seq model. The model takes an input tensor of lake variables ($V_{\text{Lake}} \times Z \times T_{\text{Lookback}}$) and a matrix of meteorological drivers ($T_{\text{Lookback}} \times V_{\text{Driver}}$). Lake data passes through a CNN-based spatial encoder and an LSTM-based temporal encoder. Meteorological drivers are concurrently encoded by an independent LSTM. The embeddings are fused via multi-headed cross-attention, which then initializes an LSTM-based temporal decoder that autoregressively predicts future lake states using a forecast of meteorological drivers while being prompted at each timestep with an embedding obtained from future driver data. A CNN-based spatial decoder reconstructs the final output tensor ($V_{\text{Lake}} \times Z \times T_{\text{Horizon}}$).

We describe the implementation details of the architecture below.

4.2.2.1 Spatial Encoder

The spatial encoder involves applying a 1D convolution along the depth dimension for each timestep in the lookback window. Each input variable is treated as a separate channel to the 1D convolution. Mathematically the 1D convolution can be expressed as follows: $f_{\text{1D Conv.}} : \mathbb{R}^{V \times Z \times T} \rightarrow \mathbb{R}^{C \times Z \times T}$. From here, the encoding is flattened along the output channel axis into a matrix as follows: $f_{\text{Flattening}} : \mathbb{R}^{C \times Z \times T} \rightarrow \mathbb{R}^{ZC \times T}$. Because each lake varies by the number of depths reported in the data, we need to accommodate a way to handle varying-sized inputs to the temporal encoder as a result of differences in Z between lakes. To resolve this issue, we apply an Adaptive 1D Average Pooling along the rows of the output from the flattening to resize the matrix as follows: $f_{\text{Adaptive Pooling}} : \mathbb{R}^{ZC \times T} \rightarrow \mathbb{R}^{d_{\text{model}} \times T}$ where d_{model} is the embedding dimension of the model used for the seq2seq backbone.

4.2.2.2 Temporal Encoder

The temporal encoder stage consists of two parts: the lake encoder and driver encoder. Both leverage independent, but identical, LSTM architectures for encoding the data from the lookback window.

4.2.2.3 Meteorological Driver Fusion

As described earlier in Section 4.1, the lake data is also paired with meteorological driver data. We condition the lake data on the driver data by performing a cross-attention computation between the lake data encoding and the driver data encoding with the lake data embeddings used as queries and the driver data embeddings used as keys and values.

4.2.2.4 Temporal Decoder

The driver-conditioned lake encoding is passed through a temporal decoder, again using the same architecture as the temporal encoder, which autoregressively generates embeddings for the timesteps in the horizon window. To align with approaches in past works, we leverage the horizon window’s driver data to prompt the decoder at each timestep.

4.2.2.5 Spatial Decoder

The spatial decoder follows the reverse process of the spatial encoder. A 1D adaptive pooling is applied at every timestep to reshape the encodings from a $d_{\text{model}} \times T_{\text{Horizon}}$ matrix to a $ZC \times T_{\text{Horizon}}$ sized matrix. This matrix is then reshaped to a $C \times Z \times T_{\text{Horizon}}$ shaped tensor before being passed through another 1D convolution layer to generate the final predictions of shape $V_{\text{Lake}} \times Z \times T_{\text{Horizon}}$.

Since seq2seq requires historical sequences of observations, which may be missing, we use the Self-Attention Imputation for Time Series model for imputing the *lookback* sequences of the model trained with default parameters from the PyPOTS library [7, 8]. The horizon window sequences remain intact and a masked RMSE cost computation is employed to compute loss only between predictions and values that are present.

4.3 Experiments

We describe the experimental design along with any additional setup below:

4.3.1 Experiment 1: Baselines

For baselines and all experiments described hereafter, we perform a chronological training-validation-testing data split of 70%-15%-15%, respectively. For baselines, we train both model architectures on three sizes of training data: 30%, 50% and the full 70% allocation to examine the effect of training data volume on in-distribution and zero-shot performance. Models were tuned using the Tree Parzen Estimator from Optuna for 50 trials to minimize validation RMSE. All models were trained for 5 trials on data for the training lakes individually followed by zero-shot evaluation on PRLA, as described in Subsection 4.1.3.

4.3.2 Experiment 2: Physics-Guided Constraints with Static Balancing

We utilize the same data splitting scheme from Subsection 4.3.1. We only consider a 30% training utilization split in order to assess the improvement conferred by the PG constraints. In addition to computing a data cost on this split, we leverage unlabeled samples by using the input samples from the unused training data, validation and testing splits and only computing PG cost from these samples to prevent data leakage. This approach was utilized given the findings of Elhamod et al. [11] which demonstrated the benefit of large amounts of unlabeled samples, in addition to the data cost, to guide the model to physically consistent predictions. Following Daw et al. [4, 5], we perform a coarse rescaling of the PG cost terms by dividing the PG costs by the baseline physical inconsistency from training and then multiplying by the training MSE. This is done to ensure consistent units and similar scale between the PG costs and the RMSE cost function. Each $\lambda_{\text{Physics}_i}$ hyperparameter was determined through hyperparameter tuning using the same setup from Subsection 4.3.1. To prevent the hyperparameter tuning process from circumventing the physics constraints by

assigning values for each $\lambda_{\text{physics}_i}$ that are close to 0, we constrain the search space of the physics hyperparameters such that $10^{-4} \leq \lambda_{\text{physics}_i} \leq 10^4$ according to past literature in PGNNs for lakes ecology modeling

4.3.3 Experiment 3: Multi-Task Learning Methods

We follow the same experimental setup from Subsection 4.3.2 with the addition of the PC-Grad and IMTL adaptive methods implemented in the “conflictfree” Python package [25].

Chapter 5

Results

In this chapter, we discuss the results from our experiments described in Subsections 4.3.1 to 4.3.3.

5.1 Baselines

Overall, the baseline results align with conventional expectations: as the proportion of training data increases, the within-distribution test RMSE generally decreases for most of the MLP and LSTM models. We show this result in Figure 5.1.

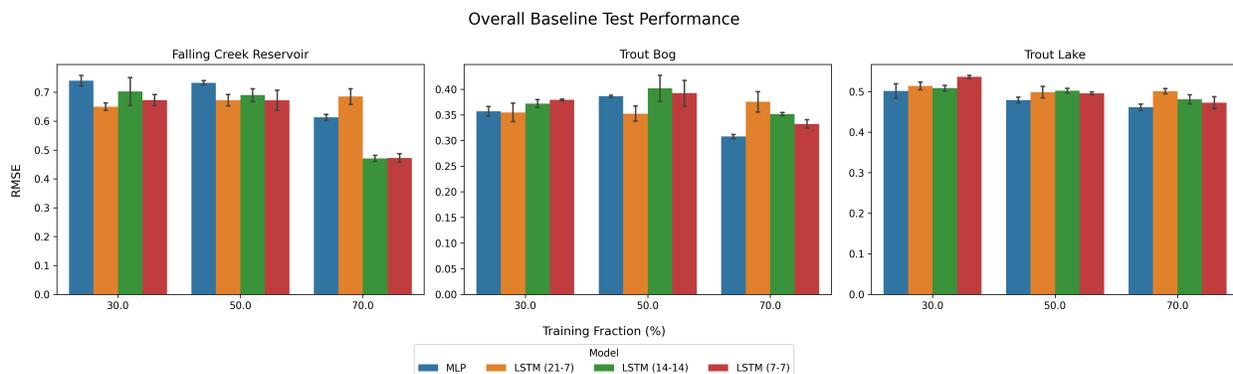


Figure 5.1: Overall test RMSE (i.e. RMSE across all variables when normalized) by training fraction of the model and colored by model architecture and configuration. The pair of numbers in the parenthesis indicates the lookback and horizon window lengths of the LSTM model in the form “(lookback-horizon)”.

This trend supports the intuition that larger training sets enable the models to learn more

robust representations. An additional observation is that there is no clear combination of lookback and horizon window (i.e., 7-7, 14-14, and 21-7) that universally outperform the others across all lakes and training fractions. For example, in FCR, LSTM (21-7), in orange, progressively gets worse RMSE while the others improve. Whereas in TR, this configuration is relatively consistent. This suggests that longer input windows may not always translate to better generalization and might introduce noise or overfitting.

We also examine, unnormalized performance for each variable across training fractions and model configurations, shown in Figures 5.2 to 5.4. Again, generally, test RMSE decreases with increased training fractions. However, for some variables, such as TP, larger amounts of training data do not seem to affect performance. TP is observed, approximately, every two weeks and yields large amounts of missing values on a daily scale. This may make it difficult for the model to learn properly, thus leading to poor performance irrespective of training size. The high RMSE values for PAR may be for similar reasons where PAR is not necessarily observed at all depths, making it difficult for the models to generalize. This may make PAR a good candidate for PG constraints. More insights into the percentage of missing values for each variable in each lake are available in Appendix A. Additionally, we observe that some variables like DO and TEMP do not benefit from more training data. Both variables are highly represented in the data (i.e. do not contain relatively large amounts of missing data), which might lead the models to learn their behaviors with minimal data.

In the zero-shot setting (see Tables 5.2 and 5.4), we observe patterns consistent with our ecological similarity hypothesis. Specifically, the lowest RMSEs are observed for TB, a lake more similar in ecological structure to the target prediction domain. In contrast, FCR and TR, which differ more substantially in their ecological dynamics, tend to yield higher errors. This suggests that transferring learned knowledge across lakes may be more effective when source-target ecological characteristics align. We also observe that the optimal training

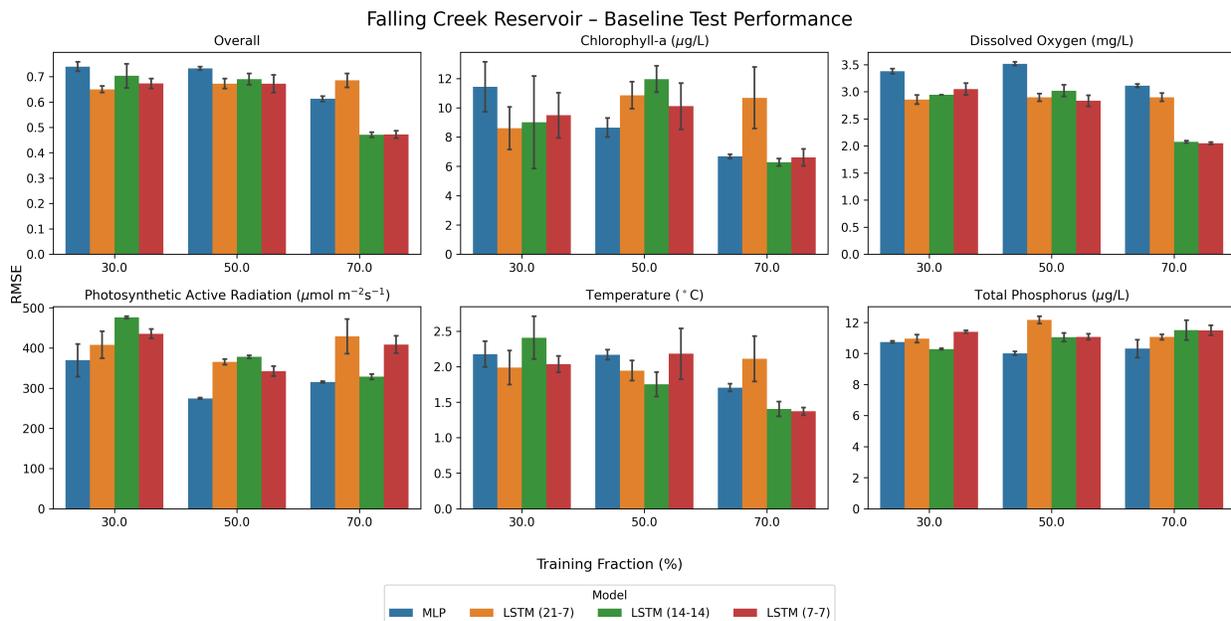


Figure 5.2: Baseline, per-variable test performance for Falling Creek Reservoir by training fraction and colored by model architecture and configuration. The pair of numbers in the parenthesis indicates the lookback and horizon window lengths of the LSTM model in the form “(lookback-horizon)”.

fraction utilization varies across training lakes and model configurations.

5.1.1 MLP

The overall test RMSEs and zero-shot RMSEs are given in Tables 5.1 and 5.2, respectively.

Lake	Training Percentage		
	30%	50%	70%
FCR	0.6180 ± 0.0280	0.5730 ± 0.0010	0.4060 ± 0.0110
TB	0.3570 ± 0.0095	0.3860 ± 0.0021	0.3080 ± 0.0041
TR	0.5014 ± 0.0177	0.4794 ± 0.0067	0.4612 ± 0.0081

Table 5.1: Baseline MLP overall, test RMSE. Blue highlight indicates the training fraction that performed the best.

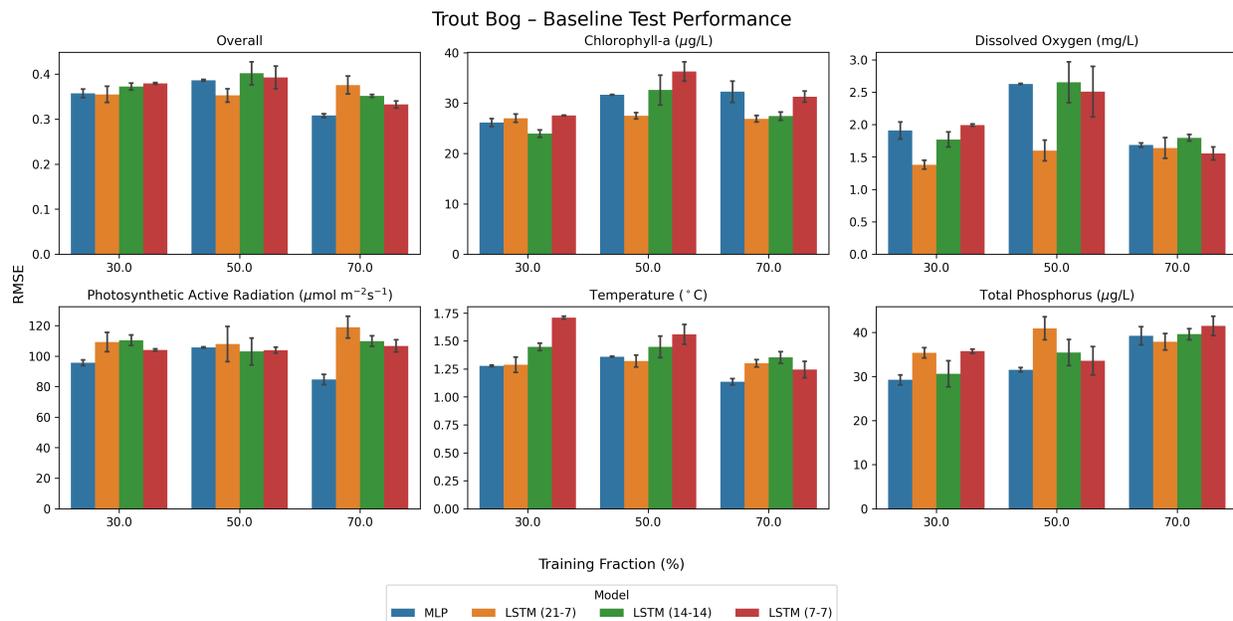


Figure 5.3: Baseline, per-variable test performance for Trout Bog by training fraction and colored by model architecture and configuration. The pair of numbers in the parenthesis indicates the lookback and horizon window lengths of the LSTM model in the form “(lookback-horizon)”.

<u>Lake</u>	<u>Distance</u>	<u>Training Percentage</u>		
		30%	50%	70%
FCR	5.33	3.1223 ± 0.0201	3.3891 ± 0.0064	2.8924 ± 0.0018
TB	3.67	1.0472 ± 0.0170	0.9188 ± 0.5423	1.1486 ± 0.0409
TR	5.98	5.0402 ± 5.3957	5.3957 ± 0.0548	5.8238 ± 0.0283

Table 5.2: Baseline MLP overall, zero-shot RMSE on PRLA. Distance of each lake to PRLA, according to Subsection 4.1.3, is indicated in “Distance” column. Blue highlight indicates the lake that performs the best on PRLA. **Bold** indicates the training fraction for each lake that performed the best.

5.1.2 LSTM

The overall test RMSEs and zero-shot RMSEs are given in Tables 5.3 and 5.4, respectively.

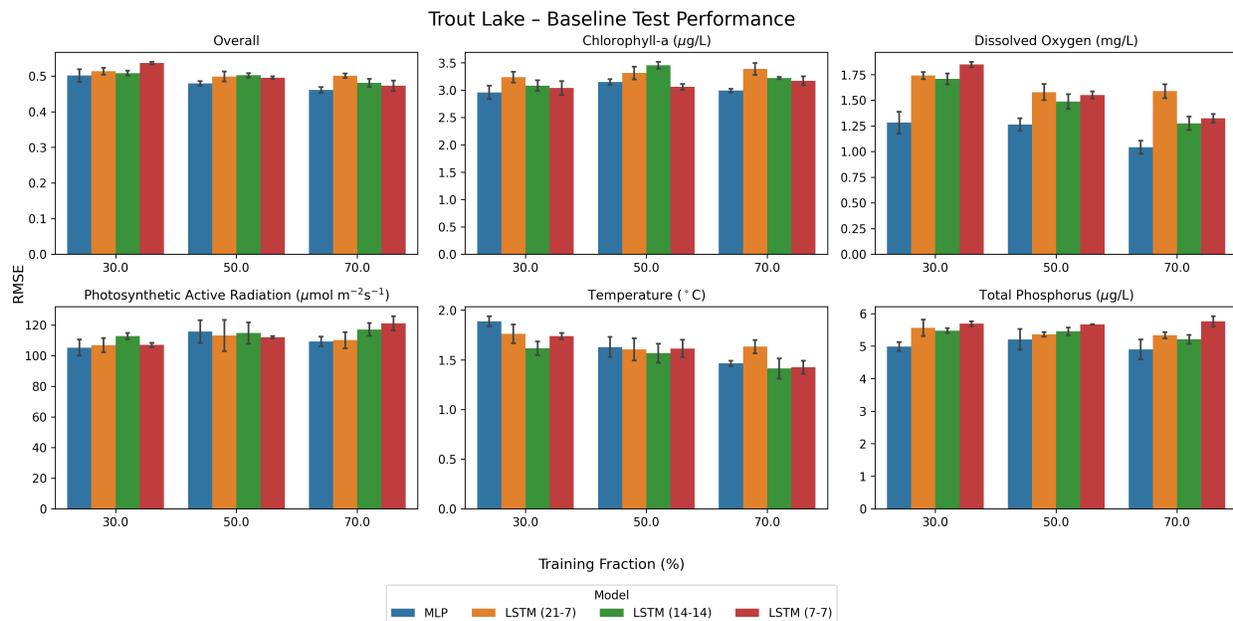


Figure 5.4: Baseline, per-variable test performance for Trout Lake by training fraction and colored by model architecture and configuration. The pair of numbers in the parenthesis indicates the lookback and horizon window lengths of the LSTM model in the form “(lookback-horizon)”.

<u>Lookback-Horizon</u>	<u>Lake</u>	<u>Training Percentage</u>		
		30%	50%	70%
7-7	FCR	0.6728 ± 0.0194	0.6719 ± 0.0348	0.4720 ± 0.0149
	TB	0.3792 ± 0.0016	0.3924 ± 0.0250	0.3324 ± 0.0081
	TR	0.5365 ± 0.0037	0.4956 ± 0.0038	0.4728 ± 0.0146
14-14	FCR	0.7030 ± 0.0481	0.6900 ± 0.0222	0.4709 ± 0.0104
	TB	0.3722 ± 0.0076	0.4017 ± 0.0256	0.3517 ± 0.0032
	TR	0.5365 ± 0.0037	0.4956 ± 0.0038	0.4728 ± 0.0146
21-7	FCR	0.6500 ± 0.0129	0.6724 ± 0.0196	0.6852 ± 0.0271
	TB	0.3549 ± 0.0178	0.3525 ± 0.0148	0.3756 ± 0.0198
	TR	0.5140 ± 0.0094	0.4988 ± 0.0140	0.5010 ± 0.0067

Table 5.3: LSTM baseline, test RMSE by training utilization fraction and sequence length. Blue highlight indicates the training fraction that performed the best.

5.2 Physics-Guided Constraints with Static Balancing

In this experiment, we evaluated the effect of balancing all PG constraints using static hyperparameters.

<u>Lookback -Horizon</u>	<u>Lake</u>	<u>Dist.</u>	<u>Training Percentage</u>		
			30%	50%	70%
7-7	FCR	5.33	3.1124 ± 0.0175	3.3693 ± 0.0182	2.8565 ± 0.0205
	TB	3.67	1.6409 ± 0.0045	1.4518 ± 0.0623	1.4448 ± 0.0751
	TR	5.98	4.9898 ± 0.0088	5.2759 ± 0.0619	5.7151 ± 0.0274
14-14	FCR	5.33	3.0963 ± 0.0196	3.3854 ± 0.0057	2.8337 ± 0.0216
	TB	3.67	1.6322 ± 0.0570	1.5657 ± 0.0827	1.5123 ± 0.0700
	TR	5.98	5.0563 ± 0.0480	5.3389 ± 0.0622	5.7383 ± 0.0323
21-7	FCR	5.33	3.1316 ± 0.0363	3.3663 ± 0.0179	3.1068 ± 0.0118
	TB	3.67	1.6561 ± 0.0585	1.4999 ± 0.0835	1.6503 ± 0.0612
	TR	5.98	5.0371 ± 0.0381	5.3637 ± 0.0228	5.0802 ± 0.0478

Table 5.4: LSTM baseline, zero-shot RMSE on PRLA by training utilization fraction and sequence length. Distance of each lake to PRLA, according to Subsection 4.1.3, is indicated in the “Dist.” column. Blue highlight indicates the lake that performs the best on PRLA. **Bold** indicates the training fraction for each lake that performed the best.

Applying a static balancing scheme revealed differences in performance and training dynamics between the MLP and LSTM models. For the MLP, the impact of static balancing was inconsistent across both test and zero-shot evaluations. In some cases, models saw declines in test RMSE and zero-shot RMSE while other saw declines in the former but not the latter. In contrast, the LSTM model exhibited more consistent trends. Across all lakes, the application of static PG constraints led to a decline in test and zero-shot performance. Examination of the convergence curves showed that static constraint balancing often demonstrated instability, with jagged convergence behavior, suggesting the presence of gradient pathologies such as conflicting gradients or imbalanced magnitudes as discussed earlier.

Despite these challenges, the experiments reaffirmed a generalizable trend: lakes that were more similar in ecological structure to the target, zero-shot evaluation lake yielded better zero-shot performance, supporting our ecological similarity hypothesis.

Overall, static PG constraint balancing provided insight incorporating physical knowledge

into neural networks via fixed terms. While there may be potential in guiding predictions using this approach, the results encourage further exploration. In cases where performance worsened, the findings are consistent with prior literature suggesting that multiple PG constraints can impede learning. Conversely, in instances where performance unexpectedly improved, the results encourage experimentation as to whether MTL approaches can potentially enhance these improvements further.

The results for the MLP and LSTM models are shown in Subsections 5.2.1 and 5.2.2, respectively. In addition to RMSE, we also report relative RMSE to assess the improvement or worsening over the baseline, computed as shown in Equation 5.1.

$$\text{RRMSE} = \frac{\text{RMSE}_{\text{PGNN}}}{\text{RMSE}_{\text{Baseline}}} \quad (5.1)$$

5.2.1 MLP

Tables 5.5 and 5.6 present the effects of statically balanced PG constraints on the MLP model’s performance. The impact is inconsistent across within-distribution and out-of-distribution scenarios. For FCR, this approach led to a reduction in test and zero-shot RMSE. Conversely, TR witnessed a reduction in test RMSE while making improvements in zero-shot RMSE. TB performs the best during zero-shot evaluation which is consistent with our lake similarity hypothesis. At times the RMSE is hampered while at others it is improved. While we initially hypothesized that static constraint balancing, due to its increased training complexity, would uniformly degrade performance, the results suggest that static balancing may not inherently disadvantage the model. However, there may be room for improvement by leveraging the MTL strategies.

If we examine the convergence behavior the the MLP model in Figure 5.5, we observe that,

with static balancing of the PG constraints, the model inconsistently converges to higher or lower training and validation costs across the lakes. In FCR and TB, we observe that the statically balanced model exhibits a jagged convergence curve, potentially affirming the gradient pathologies that might arise during PGNN training, as discussed in Section 2.2. Curiously, we do not observe this pattern in TR; the statically balanced model demonstrates higher convergence, but a smooth curve for the validation cost compared to the baseline model despite both being trained with LR scheduling.

<u>Lake</u>	<u>RMSE</u>	<u>RRMSE</u>
FCR	1.1406 ± 0.0011	1.8456
TB	0.3322 ± 0.0073	0.9305
TR	0.6216 ± 0.0738	1.2397

Table 5.5: Test RMSE of MLP model trained with PG constraints using a static balancing approach. The relative RMSE against the baseline MLP model with no PG constraints is shown. **Green** indicates an improvement over the baseline while **red** indicates a worsening over the baseline.

<u>Lake</u>	<u>Dist.</u>	<u>RMSE</u>	<u>RRMSE</u>
FCR	5.33	3.1673 ± 0.0006	1.0144
TB	3.67	1.0237 ± 0.0093	0.9776
TR	5.98	4.9608 ± 0.0208	0.9842

Table 5.6: Zero-shot RMSE of MLP model trained with PG constraints using a static balancing approach on PRLA. The relative RMSE against the baseline MLP model with no PG constraints is shown. **Green** indicates an improvement over the baseline while **red** indicates a worsening over the baseline. The distance of each lake to PRLA is indicated in the “Dist.” column. **Bold** indicates the best-performing lake.

5.2.2 LSTM

Tables 5.7 and 5.8 present the effects of statically balanced PG constraints on the LSTM-RNN model’s performance. The model demonstrates more consistent results than the MLP model. Static balancing of the PG constraints consistently reduces the test and zero-shot performance of the model across all lakes. In the zero-shot setting, the model trained on TB performs the best on PRLA, in-line with our similarity hypothesis. The diminished performance affirms our hypothesis that training PGNNs with multiple constraints presents a

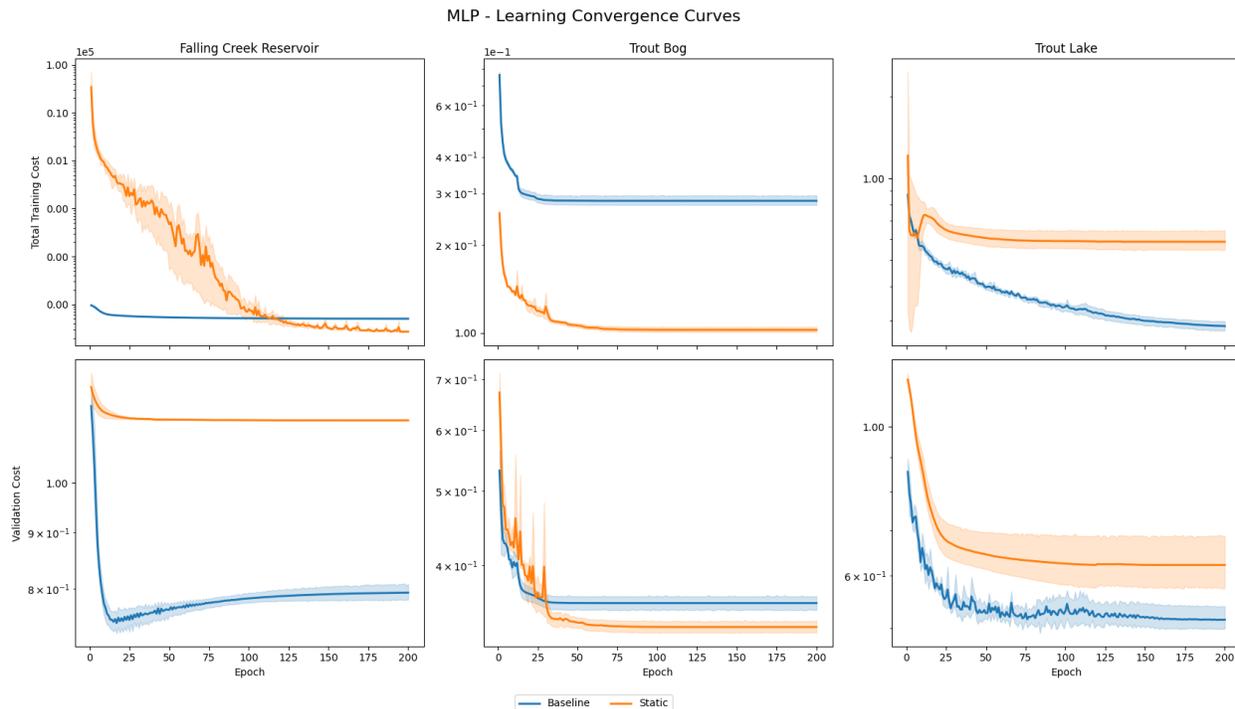


Figure 5.5: A \log_{10} -linear plot of the learning convergence and validation error curves of the MLP model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The model trained with static PG constraint balancing is shown against the baseline model trained with no PG constraints.

challenging optimization problem, thus warranting further investigation with MTL methods.

If we examine the convergence behavior of the LSTM-RNN model in Figure 5.6, we observe that the model trained with static balancing converged to lower training costs than the baselines but converged higher on validation cost. We also observe that the statically balanced model exhibits a jagged convergence curve, again, potentially affirming the gradient pathologies that might arise during PGNN training. Unlike the MLP, the baseline model demonstrates smooth convergence across all lakes.

Lake	RMSE	RRMSE
FCR	0.6877 ± 0.0328	1.0221
TB	0.4283 ± 0.0053	1.1295
TR	0.5952 ± 0.0538	1.1094

Table 5.7: Test RMSE of LSTM-RNN model trained with PG constraints using a static balancing approach. The relative RMSE against the baseline MLP model with no PG constraints is shown. **Red** indicates a worsening over the baseline.

Lake	Dist.	RMSE	RRMSE
FCR	5.33	3.2191 ± 0.0468	1.0343
TB	3.67	1.7604 ± 0.0970	1.0728
TR	5.98	5.0594 ± 0.0379	1.0139

Table 5.8: Zero-shot RMSE of LSTM-RNN model trained with PG constraints using a static balancing approach on PRLA. The relative RMSE against the baseline MLP model with no PG constraints is shown. **Red** indicates a worsening over the baseline. The distance of each lake to PRLA is indicated in the “Dist.” column. **Bold** indicates the best-performing lake.

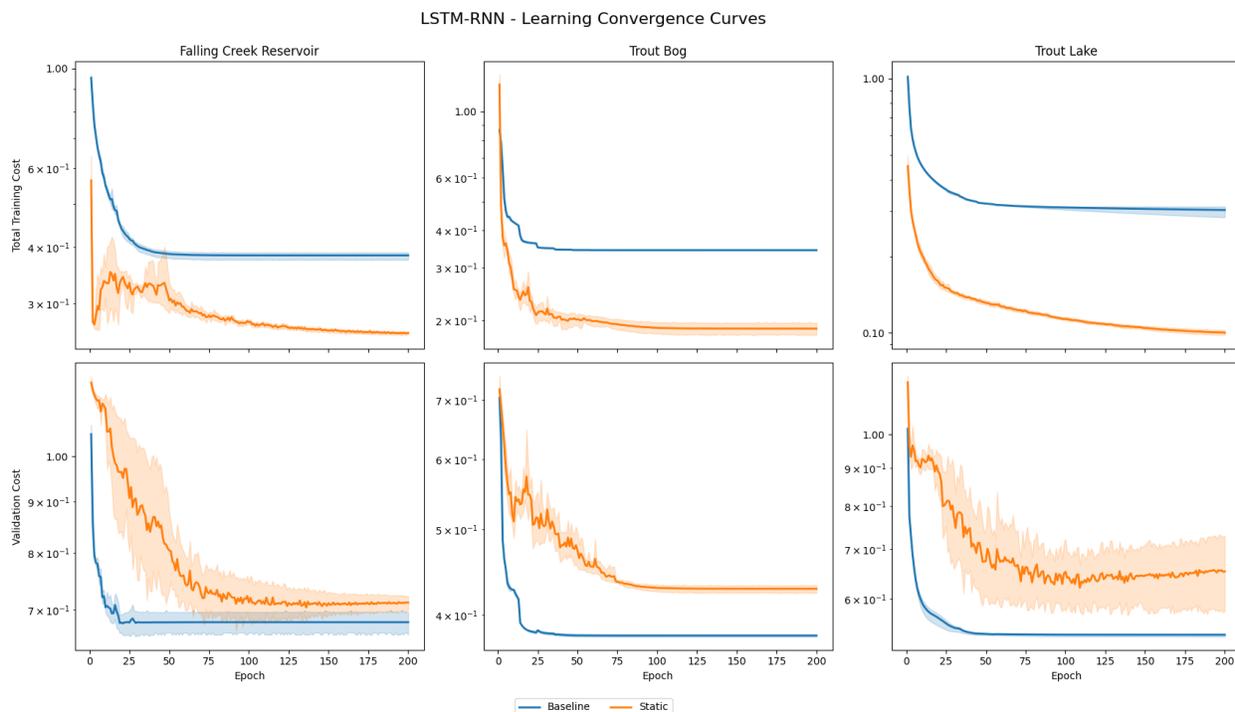


Figure 5.6: A \log_{10} -linear plot of the learning convergence and validation error curves of the LSTM-RNN model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The model trained with static PG constraint balancing is shown against the baseline model trained with no PG constraints.

5.3 Adaptive MTL Methods

In this section, we evaluated the effectiveness of MTL methods for training PGNNs with multiple constraints.

Our findings reveal a nuanced picture. For LSTM models, MTL strategies, particularly PCGrad, outperformed both static constraint balancing and the baseline in terms of within-distribution test RMSE. In contrast, MLP models did not consistently benefit from MTL methods, with both PCGrad and IMTL often failing to outperform simpler approaches.

In zero-shot settings, MTL methods did not yield consistent improvements. In fact, in many cases, these methods worsened generalization performance, suggesting a potential overfitting to the training lake’s data distribution. This may reflect a tension between achieving physical consistency on within-distribution evaluation and maintaining generalizability to unseen distributions.

While MTL methods show some promise, they are not a universal solution. Directly applying MTL methods can cause erratic training dynamics, especially in MLP models. We conducted diagnostic experiments with stabilization techniques including gradient clipping and learning rate warm-up which improved training stability and improved results. However, there is no one-size-fits all strategy, indicating that the choice of MTL method and supplemental optimization techniques must be experimented with and tuned for the specific application.

The results for the MLP and LSTM models are shown in Subsections [5.3.1](#) and [5.3.2](#), respectively.

5.3.1 MLP

Tables 5.9 and 5.10 present the effects of the PCGrad and IMTL methods on the MLP model’s performance. The impact is consistent across within-distribution and out-of-distribution scenarios. All lakes demonstrated a reduction in test and zero-shot RMSE. However, our similarity hypothesis remains intact as TB performs the best during zero-shot evaluation.

Examining the convergence behavior, we observe that the MTL methods display erratic convergence behavior. For training cost, IMTL displays spiking behavior and ultimately fails to converge across all lakes. PCGrad’s training convergence behavior is dependent on the training lake. In FCR, the convergence is jagged, like the static balancing, but converges faster and at a higher error, while in TB and TR the error displays diverging behavior and a failure to converge. For the validation cost, both methods converge at or higher than the statically balanced model in all lakes. The validation curves also demonstrate jagged, spiking behavior.

This result comes as a surprise as we initially hypothesized that MTL methods would confer an improvement over the baseline’s RMSEs. These results may suggest some underlying training pathology. We conduct diagnostic experiments in Subsection 5.3.3 to try and improve the training dynamics of the MTL methods.

5.3.2 LSTM-RNN

Tables 5.11 and 5.12 present the effects of PCGrad and IMTL methods on the LSTM model’s performance. Unlike the results from Section 5.2, the LSTM demonstrates some conflicting results. Generally, the test and zero-shot RMSEs are worsened when directly applying PCGrad and IMTL. Not all lakes observed reductions in test RMSE with TB and TR

<u>MTL Method</u>	<u>Lake</u>	<u>RMSE</u>	<u>RRMSE</u>
PCGrad	FCR	1.0837 ± 0.0040	1.7536
	TB	0.5379 ± 0.0057	1.5067
	TR	0.9717 ± 0.1518	1.9380
IMTL	FCR	1.1098 ± 0.0467	1.7958
	TB	0.6938 ± 0.0180	1.9434
	TR	1.0904 ± 0.0116	2.1747

Table 5.9: Test RMSE of the MLP model trained with PG constraints using the PCGrad and IMTL methods. The relative RMSE against the baseline MLP model with no PG constraints is shown. **Red** indicates a worsening over the baseline.

<u>MTL Method</u>	<u>Lake</u>	<u>Distance</u>	<u>RMSE</u>	<u>RRMSE</u>
PCGrad	FCR	5.33	3.6436 ± 0.3943	1.1670
	TB	3.67	1.4074 ± 0.0055	1.3440
	TR	5.98	5.0563 ± 0.0617	1.0032
IMTL	FCR	5.33	3.4307 ± 0.1005	1.0988
	TB	3.67	1.7118 ± 0.1127	1.6346
	TR	5.98	5.2214 ± 0.0516	1.0369

Table 5.10: Zero-shot RMSE of the MLP model, trained with PG constraints using the PCGrad and MTL method, on PRLA. The relative RMSE against the baseline MLP model with no PG constraints is shown. **Red** indicates a worsening over the baseline. The distance of each lake to PRLA is indicated in the “Distance” column.

showing improvements. In the zero-shot setting, TB still performs the best and only TR demonstrates an improvement in zero-shot RMSE.

Examining the convergence behavior in Figure 5.8, we observe that the MTL methods display a more stable behavior compared to the MLP model. Although there is jagged convergence behavior that is observed in both the training and validation curves for the baselines, IMTL and, at times, PCGrad, the convergence is decreasing in a comparable fashion to the baselines, albeit at a slower speed. This could potentially be attributed to slower learning rate as determined by the hyperparameter tuning process.

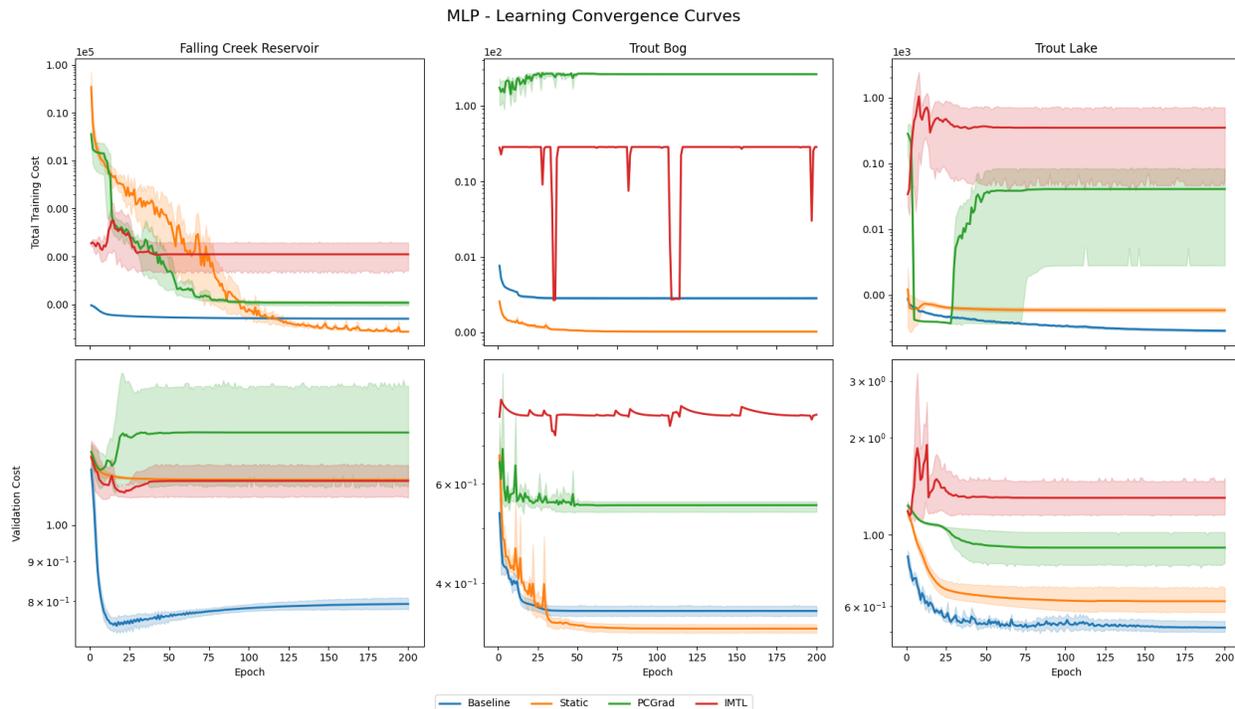


Figure 5.7: A \log_{10} -linear plot of the learning convergence and validation error curves of the MLP model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The various MTL methods are shown against the curves from the baseline model and model trained with static PG constraint balancing.

However, this result again comes as a surprise as we initially hypothesized that MTL methods would improve the model performance. As with the MLP model, we conduct further diagnostic experiments in Subsection 5.3.3 to try and seek improvements from the MTL methods.

5.3.3 Diagnostic Experiments

We investigated the erratic convergence behavior of the MTL methods by leveraging gradient clipping and learning rate (LR) warm-up to try and stabilize the training process of the model. Gradient clipping limits (i.e. “clips”) the magnitude of the gradients in the model

<u>MTL Method</u>	<u>Lake</u>	<u>RMSE</u>	<u>RRMSE</u>
PCGrad	FCR	1.1584 ± 0.0021	1.7218
	TB	0.3184 ± 0.0189	0.8397
	TR	0.5088 ± 0.0155	0.9484
IMTL	FCR	0.6759 ± 0.0170	1.0046
	TB	0.3858 ± 0.0278	1.0174
	TR	0.5616 ± 0.0247	1.0468

Table 5.11: Test RMSE of the LSTM model trained with PG constraints using the PCGrad and IMTL methods. The relative RMSE against the baseline LSTM model with no PG constraints is shown. **Green** indicates an improvement over the baseline while **red** indicates a worsening over the baseline.

<u>MTL Method</u>	<u>Lake</u>	<u>Distance</u>	<u>RMSE</u>	<u>RRMSE</u>
PCGrad	FCR	5.33	3.1592 ± 0.0004	1.0150
	TB	3.67	1.7132 ± 0.0616	1.0441
	TR	5.98	5.0041 ± 0.0613	1.0029
IMTL	FCR	5.33	3.1373 ± 0.0949	1.0080
	TB	3.67	1.6371 ± 0.0504	1.0030
	TR	5.98	5.0234 ± 0.0444	0.9973

Table 5.12: Zero-shot RMSE of the LSTM model, trained with PG constraints using the PCGrad and IMTL method, on PRLA. The relative RMSE against the baseline LSTM model with no PG constraints is shown. **Green** indicates an improvement over the baseline while **red** indicates a worsening over the baseline. The distance of each lake to PRLA is indicated in the “Distance” column.

during backpropagation to prevent exploding gradients, a common problem in RNN models [48]. Specifically, we employ a global gradient clipping where we constrained the norm of the overall, treated gradient (i.e. the gradient without conflict or magnitude pathologies) returned from the MTL method to a maximum norm of 1.0. LR warm-up assists and stabilizes model convergence by progressively increasing the LR over several iterations during the initial epochs of model training before holding the LR constant or returning to a previously defined LR schedule. This assists the model’s optimizer by allowing it to escape from local minima that it may encounter during the early parts of the training process. Specifically, we

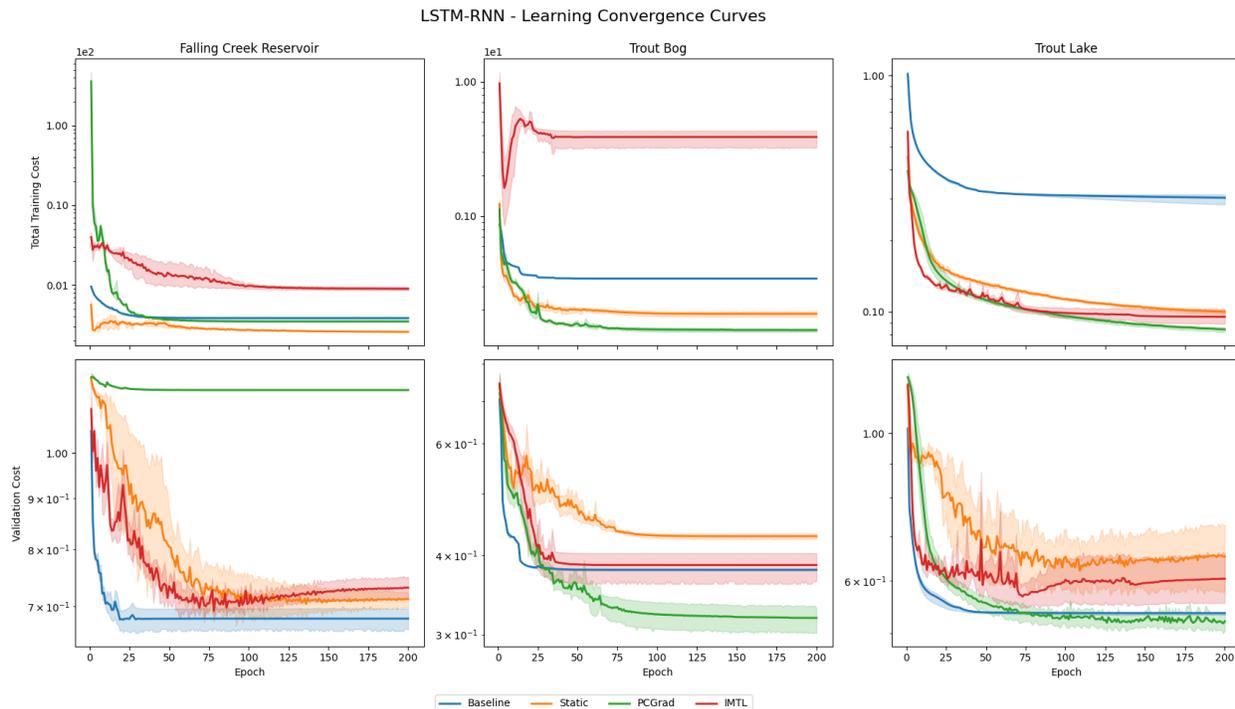


Figure 5.8: A \log_{10} -linear plot of the learning convergence and validation error curves of the LSTM-RNN model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The various MTL methods are shown against the curves from the baseline model and model trained with static PG constraint balancing.

utilized a linear warm-up schedule to increase the LR to 1.0 over the first 20 epochs before returning to a “Reduce LR on plateau” LR schedule. Some studies suggest that when the neural network is randomly initialized, the gradients from the different tasks can rapidly change directions which may make it difficult for methods like IMTL to find good overall gradient directions [25]. We applied these techniques separately to the model configurations described in Section 5.3 and examined their effect on the learning convergence curves from $\mathcal{L}_{\text{Train}}$ and $\mathcal{L}_{\text{Validation}}$. The curves from the MLP and LSTM-RNN models are shown in Figures 5.9 and 5.10, respectively.

Overall, the LSTM-RNN models demonstrated more stable convergence in training cost compared to the MLP models. Across all three lakes, the LSTM-RNN was less prone to

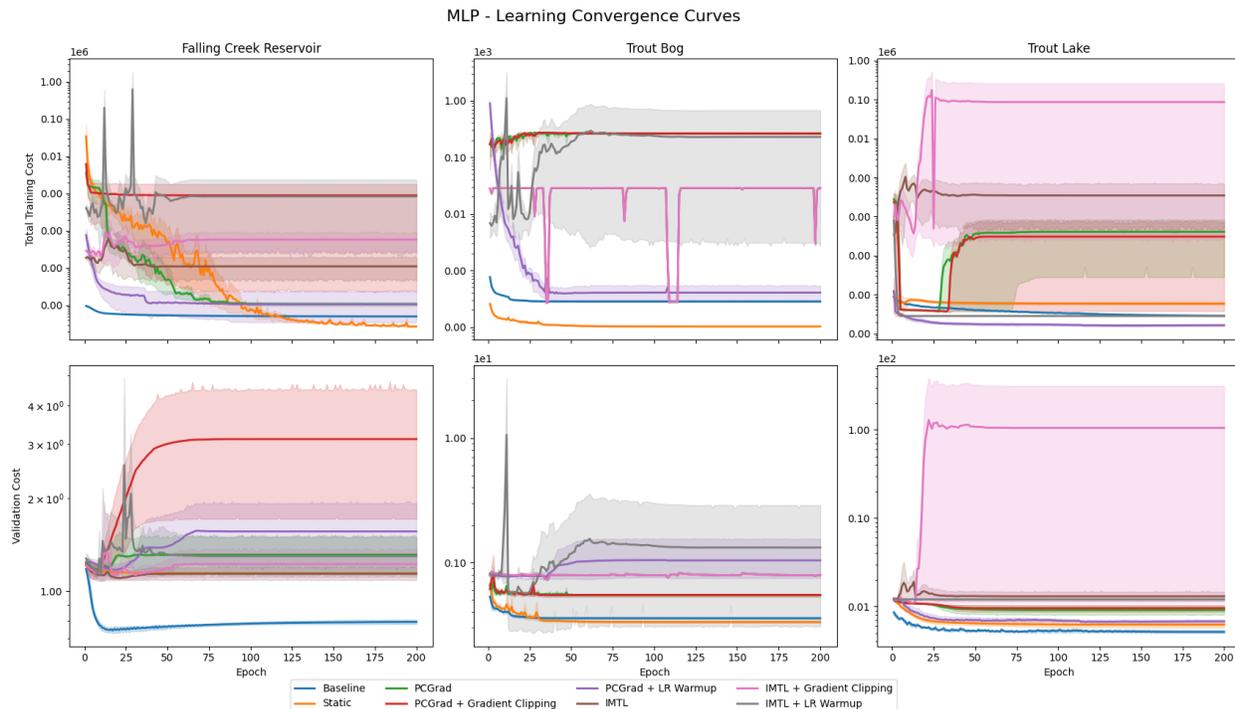


Figure 5.9: A \log_{10} -linear plot of the learning convergence and validation error curves of the MLP model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The various combinations of MTL method and the applied stabilization technique are shown against the curves from the baseline model and model trained with static PG constraint balancing.

divergence even when PG constraints were active. In contrast, the MLP model showed more erratic behavior particularly when IMTL was combined with LR warm-up, which frequently caused spiking in training cost. While the LSTM-RNN models demonstrated a more typical decay of the training and validation costs, their curves still demonstrated jagged, non-smooth convergence, compared to the baseline when MTL methods were applied.

These stabilization techniques can help smoothen training cost but they do not *consistently* translate to improved validation error across all lakes and training methods as the benefits appear to be lake- and model-specific.

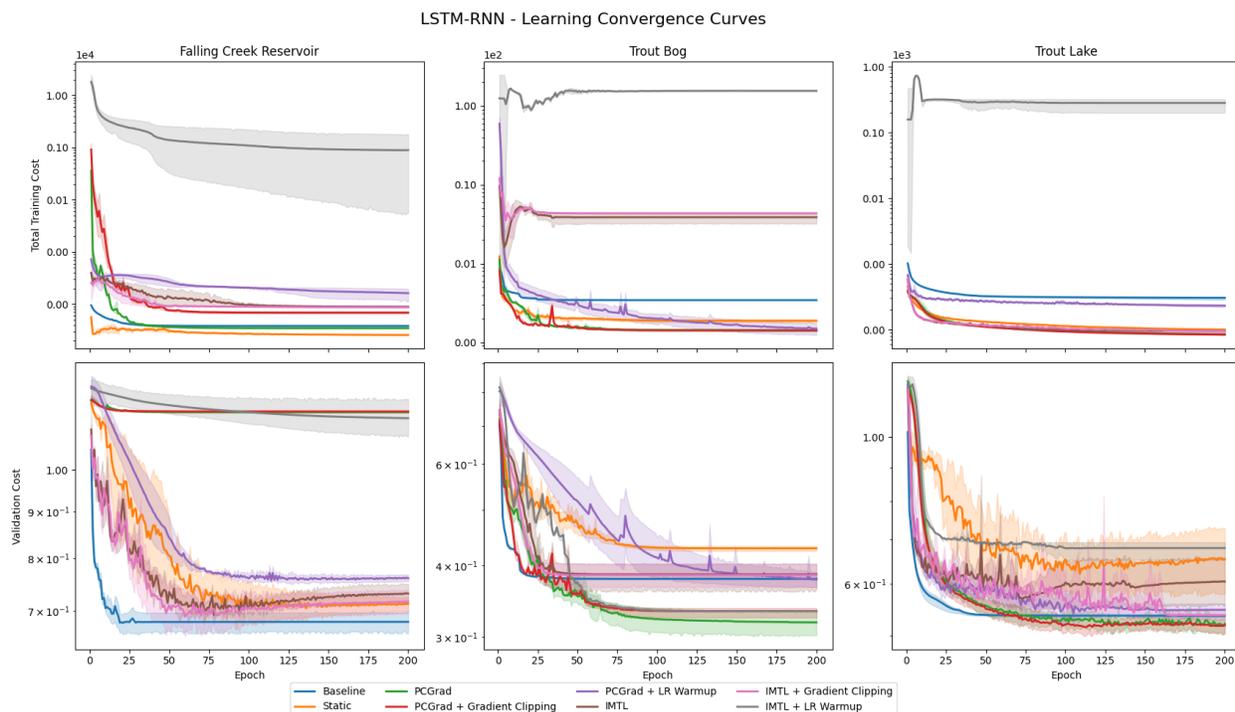


Figure 5.10: A \log_{10} -linear plot of the learning convergence and validation error curves of the LSTM-RNN model with a 95% confidence interval (shaded) region shown for all three lakes. A \log_{10} scale was chosen to account for remnant scale imbalances introduced by the PG constraints. The various combinations of MTL method and the applied stabilization technique are shown against the curves from the baseline model and model trained with static PG constraint balancing.

5.3.4 Summary

Tables 5.13 and 5.14 present a consolidated comparison of the test and zero-shot RMSEs for both MLP and LSTM models trained with different strategies: baseline (no PG constraints), static PG constraint balancing, and the MTL methods PCGrad and IMTL. Where stabilization techniques (discussed in Subsection 5.3.3) yielded improved performance for a given MTL method, the best is reported.

Overall, the LSTM models appear to benefit more consistently from MTL approaches in terms of within-distribution test RMSE. In particular, PCGrad combined with stabilization

techniques achieved the lowest test RMSE for TB and TR, outperforming both the baseline and static constraint balancing. Conversely, the MLP models showed limited or no benefit from the MTL strategies. In several cases, MTL methods actually worsened performance compared to both the baseline and static approaches, despite the application of stabilization techniques.

In the zero-shot generalization setting, results were less encouraging. Across both MLP and LSTM models, MTL methods did not provide consistent improvements over the baseline or static balancing approaches. In some cases, such as for TB in the LSTM model, adaptive MTL methods slightly improved zero-shot RMSE. Notably, TB remained the top-performing lake in most zero-shot evaluations, reaffirming the similarity hypothesis.

One interpretation of these findings is that the LSTM architecture may be more naturally suited to benefit from gradient-based MTL methods. However, the improved within-distribution performance observed for LSTM models may also signal increased model fit to the training distribution at the cost of generalization, as indicated by the lack of improvement or even degradation in the zero-shot setting. This may suggest an overfitting tendency when strong constraints are applied through MTL in data-sparse environments.

The mixed success of MTL methods, especially for the MLP models, raises questions about the compatibility of PG constraint optimization with shallow feedforward architectures.

<u>Model</u>	<u>Lake</u>	<u>Method</u>				<u>Best RRMSE</u>
		Baseline	Static	PCGrad	IMTL	
MLP	FCR	0.6180	1.1406	1.0837	1.1098	1.7531
	TB	0.3570	0.3322	0.5379	0.6938	0.9305
	TR	0.5014	0.6216	0.6374**	1.0899*	1.2410
LSTM	FCR	0.6728	0.6877	0.7522**	0.6709 *	0.9972
	TB	0.3792	0.4283	0.3184	0.3330**	0.8395
	TR	0.5365	0.5952	0.5088	0.5345*	0.9484

Table 5.13: Comparison of test RMSE values across methods for each lake and model. Relative RMSE of the best method compared to baseline is shown on the very right. **Blue** indicates the best performing method. **Green** indicates an improvement over the baseline and **red** indicates a worsening over the baseline. * indicates gradient clipping was applied and ** indicates LR warm-up was applied.

<u>Model</u>	<u>Lake</u>	<u>Dist.</u>	<u>Method</u>				<u>Best RRMSE</u>
			Baseline	Static	PCGrad	IMTL	
MLP	FCR	5.33	3.1223	3.1673	3.5811**	3.3947**	1.0144
	TB	3.67	1.0472	1.0237	1.3751*	1.6264*	0.9776
	TR	5.98	5.0402	4.9608	4.9618**	5.0067**	0.9842
LSTM	FCR	5.33	3.1124	3.2191	3.1409**	3.1373	1.0080
	TB	3.67	1.6409	1.7604	1.6747**	1.6358**	0.9969
	TR	5.98	4.9898	5.0594	5.0041	5.0189**	1.0029

Table 5.14: Comparison of zero-shot RMSE values across methods for each lake and model. Relative RMSE of the best method compared to baseline is shown on the very right on PRLA. The “Dist.” column indicates the distance to PRLA. **Blue** indicates the best performing method. **Green** indicates an improvement over the baseline and **red** indicates a worsening over the baseline. * indicates gradient clipping was applied and ** indicates LR warm-up was applied. **Bold** indicates the lake that performed the best on the zero-shot evaluation.

Chapter 6

Conclusions

We conclude this thesis by discussing our findings in Section 6.1, followed by a discussion on limitations and avenues for future work in Section 6.2.

6.1 Discussion

This thesis explored the challenge of training PGNNs with multiple constraints under data-scarce scenarios, without generating additional simulation data, for the task of lake ecology modeling. We assembled a suite of PG constraints informed by limnological theory by creating new PG constraints or borrowing and expanding on PG constraints from past literature. We assessed their integration into MLP and LSTM architectures by examining within-distribution test RMSE and out-of-distribution zero-shot RMSE.

First, we observed that naively incorporating multiple PG constraints through static balancing may occasionally confer additional performance but, generally, breaks down as indicated by reduced RMSE. MLP and LSTM models trained with static constraint balancing often exhibited non-smooth convergence behavior which aligns with known gradient pathologies in MTL.

Second, to address these challenges, we explored the application of MTL methods such as PCGrad and IMTL. While these methods are theoretically motivated to reduce gradient

interference, their practical effectiveness varies by model architecture. We demonstrate that by directly applying MTL methods, we do not necessarily observe consistent improvements in RMSE. However, when combined with training stabilization techniques, we show that MTL methods consistently improve LSTM, but not MLP, test RMSE. However, irrespective of model architecture, MTL methods fail to show consistent gains in zero-shot RMSE, suggesting that MTL strategies may cause the model to overfit to training distributions when data is limited.

In summary, MTL methods show some promise, but are not a silver bullet. While these findings highlight an important lesson that adding more constraints does not necessarily yield better-performing models, it provides an initial exploration into training PGNNs with multiple constraints.

6.2 Limitations and Future Work

We discuss the limitations of this work, connecting it to potential avenues of improvement or exploration for future works.

6.2.1 Data

The LakeBeD-US dataset contains large amounts of missing values, as is common in environmental data. Past works applying PGNNs for lake ecology modeling leveraged an HPD scheme where simulation data was generated from the GLM model and supplemented with the observational data, offering a somewhat “physics-based” imputation. Future work could explore the effect of adding incremental amounts of simulation data to assess the amount of simulation data needed to reach desired RMSE levels when using PG constraints. In other

words, we can attempt to answer the question “how much additional data is really needed so as to be computationally efficient?”

An additional avenue of exploration is by examining the affect of data missingness on performance and physical inconsistency. A better understanding might be obtained by generating simulation data to impute all missing values and then performing an ablation study to examine how masking different amounts of data in different manners affects the baseline physical inconsistency. For example, we might experiment with masking large percentages of the data randomly or masking small contiguous portions of the data.

6.2.2 Physics-Guided Constraints

Unlike PINNs, which are typically designed to solve well-established and rigorously defined PDEs—such as the Navier-Stokes, Burgers’, or Allen-Cahn equations—the constraints used in PGNNs might be based on incomplete, simplified, or heuristic representations of physical processes. The scientific community generally agrees on the form and structure of the equations used in PINNs, as these equations have been extensively validated and widely adopted across their respective domains. In contrast, PGNN constraints may be adapted, redefined, or augmented depending on the availability of data, the specific ecological processes of interest, or the need to approximate complex interactions with simplified formulations. For example, our Beer-Lambert constraint for PAR, described in Subsection 3.1.2, utilized a static diffuse attenuation coefficient k which might not necessarily hold true because the water quality can change through the water column. More data would be required to expand upon this constraint in future works. Similarly, the TEMP energy conservation constraint, described in Subsection 3.2.2, doesn’t factor in heat fluxes between the sediment and soil. More detailed equations would be required to account for this process.

Our PG constraints for DO and TP are currently more heuristic in nature, relying on general empirical laws rather than mechanistic representations of the underlying physical and biochemical processes. While ODEs and PDEs for these variables of interest exist, such as shown in Equations 6.1 and 6.2 given by Hanson et al. [14], utilizing these equations would require predefining, or learning and predicting the additional constant terms in the equations (i.e., Entrainment, NPP, Inflow, etc.) which themselves would need to be physically plausible.

$$\frac{dDO}{dt} = \text{Entrainment} + \text{NPP} + F_{\text{atm}} - R_{\text{tot}} \quad (6.1)$$

$$\frac{dP}{dt} = \text{Inflow} + \text{Entrainment} - \text{Settling} - \text{Binding} - \text{Outflow} \quad (6.2)$$

where

Entrainment = Movement and mixing of water from one lake layer to another

NPP = Net Primary Production

F_{atm} = Atmospheric oxygen flux

R_{tot} = Total respiration

Inflow = External input of phosphorus

Settling = Phosphorus loss due to particles settling out

Binding = Phosphorus loss due to chemical bonding with other substances

Outflow = Phosphorus loss due to water flowing out of the system

While more descriptive and rigorous PG constraints are an avenue for future work, such expansions raise a fundamental trade-off concern: while PGNNs aim to leverage ML to model complex systems without explicitly defining every physical process—as is done in process-based models—enhancing constraints to this degree risks shifting the modeling paradigm closer to that of a traditional hydrodynamic model. In simple words, “too much physics may be equally as bad as too little physics”. Future work will need to strike a balance between maintaining the flexibility of PGNNs and the desire for physically grounded, interpretable constraint formulations.

6.2.3 Modeling

There are several modeling strategies that future work could explore to improve both the training process and the physical consistency of PGNNs.

One potential avenue is to explore different training regimes, such as pretraining on data followed by PG fine-tuning. This approach could allow the model to first capture patterns in the data in a flexible, unconstrained fashion, and subsequently learn to refine those patterns in accordance with known physical principles. This staged training process may help mitigate the competition between learning from data and adhering to physical constraints, particularly in cases where the constraints are approximate or incomplete. However, care must be taken to determine the appropriate amount of fine-tuning necessary so as to avoid catastrophic forgetting of patterns learned from the pretraining stage.

Another possible direction is to apply a constraint annealing schedule, as described by Elhamod et al. [11]. This technique involves gradually introducing PG constraints during training, which could prevent early domination by difficult or poorly conditioned constraints. However, designing an effective annealing schedule would require insight into the complex-

ity and smoothness of each constraint. One possibility is to train separate models using individual constraints and analyze convergence behavior and optimization dynamics (e.g., through metrics such as convergence speed or the curvature of the cost landscape). More sophisticated methods such as NeuroVisualizer could also be employed to visualize the cost landscapes via an autoencoder [10]. This may offer deeper insights into which constraints are easier to optimize. Alternatively, it may be possible to develop an online scheme that adaptively weighs constraint contributions based on backpropagation feedback at each training iteration.

Transfer learning is another strategy worth exploring. Training PGNNs on multiple lakes, particularly lakes with diverse environmental regimes, might enable the model to generalize underlying physical relationships that are shared across systems. By learning from data across lakes, the model might require less explicit guidance from PG constraints during training, leading to both reduced physical inconsistency and improved generalization performance on unseen systems.

Future work could investigate whether training a model on a small set of comprehensive and well-understood physical constraints, such as those governing PAR, TEMP, and DO, can lead to the emergence of physically consistent predictions for other variables, such as TP, even when no explicit constraint is applied. This idea echoes principles in systems and self-organizing behavior, where a small number of well-enforced rules can give rise to more global order (e.g., Conway’s Game of Life). If such emergent consistency can be observed, it would suggest that PGNNs may be capable of implicitly learning or approximating latent physical relationships, reducing the need for extensive hand-crafted constraint development across all variables. Systematically evaluating this hypothesis, potentially through ablation studies or *post hoc* consistency checks, could offer insight into the inductive biases introduced by PG training and whether those biases are sufficient to support broader generalization in

under-constrained regimes.

Lastly, emerging work in the field of equation discovery may provide opportunities to complement PGNNs with symbolic knowledge extraction. For instance, recent approaches like LLM-SR highlight the possibility of using LLMs to derive symbolic expressions that describe physical laws directly from data [43]. While the integration of such techniques into PGNN frameworks remains an open question, it is conceivable that they could assist in either generating candidate constraints or verifying that the learned behavior of PGNNs is consistent with interpretable physical relationships.

Bibliography

- [1] Peter J. Alsip, Mark D. Rowe, Alexander Kain, and Casey Godwin. Modeling attenuation of photosynthetically active radiation across the optical gradient in the Laurentian Great Lakes with application to Lake Erie. *Journal of Great Lakes Research*, 50(4):102364, Aug 2024. ISSN 03801330. doi: 10.1016/j.jglr.2024.102364. URL <https://linkinghub.elsevier.com/retrieve/pii/S0380133024001126>.
- [2] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. In *Proceedings of the 35th International Conference on Machine Learning*, 2018.
- [3] Arka Daw, R. Quinn Thomas, Cayelan C. Carey, Jordan S. Read, Alison P. Appling, and Anuj Karpatne. *Physics-Guided Architecture (PGA) of Neural Networks for Quantifying Uncertainty in Lake Temperature Modeling*, pages 532–540. Society for Industrial and Applied Mathematics, 2020. doi: 10.1137/1.9781611976236.60. URL <https://epubs.siam.org/doi/abs/10.1137/1.9781611976236.60>.
- [4] Arka Daw, Anuj Karpatne, William Watkins, Jordan Read, and Vipin Kumar. Physics-guided neural networks (pgnn): An application in lake temperature modeling. arXiv:1710.11431 [cs], Sep 2021. URL <http://arxiv.org/abs/1710.11431>.
- [5] Arka Daw, Anuj Karpatne, Williams Watkins, Jordan Read, and Kumar Vipin. *Physics-guided neural networks (PGNN): An application in lake temperature modeling*, chapter 15, pages 253–372. Taylor & Francis, 2022. doi: 10.1201/9781003143376-15.
- [6] Peter J. Dillon and Lewis A. Molot. *The Phosphorus Cycle*, page 359–425. Elsevier, 2024.

- ISBN 978-0-12-822701-5. doi: 10.1016/B978-0-12-822701-5.00015-X. URL <https://linkinghub.elsevier.com/retrieve/pii/B978012822701500015X>.
- [7] Wenjie Du. PyPOTS: a Python toolbox for data mining on Partially-Observed Time Series, 2023.
- [8] Wenjie Du, David Côté, and Yan Liu. Saits: Self-attention-based imputation for time series. *Expert Systems with Applications*, 219:119619, 2023. ISSN 0957-4174. doi: <https://doi.org/10.1016/j.eswa.2023.119619>. URL <https://www.sciencedirect.com/science/article/pii/S0957417423001203>.
- [9] NASA Earthdata. Photosynthetically active radiation, Feb 2025. URL <https://www.earthdata.nasa.gov/topics/biosphere/photosynthetically-active-radiation>.
- [10] Mohannad Elhamod and Anuj Karpatne. Neuro-visualizer: A novel auto-encoder-based loss landscape visualization method with an application in knowledge-guided machine learning. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 12429–12447. PMLR, 21–27 Jul 2024. URL <https://proceedings.mlr.press/v235/elhamod24a.html>.
- [11] Mohannad Elhamod, Jie Bu, Christopher Singh, Matthew Redell, Abantika Ghosh, Viktor Podolskiy, Wei-Cheng Lee, and Anuj Karpatne. Cophy-pgmn: Learning physics-guided neural networks with competing loss functions for solving eigenvalue problems. *Association for Computing Machinery*, 13(6), Dec 2022. ISSN 2157-6904. doi: 10.1145/3530911. URL <https://doi.org/10.1145/3530911>.
- [12] Dean R. Freitag and Terry T. McFadden. *Introduction to Cold Regions Engineering*. American Society of Civil Engineers, New York, NY, February 1997. ISBN 978-0-7844-

- 0006-7. doi: 10.1061/9780784400067. URL <https://ascelibrary.org/doi/book/10.1061/9780784400067>.
- [13] Varun Gulshan, Lily Peng, Marc Coram, Martin C. Stumpe, Derek Wu, Arunachalam Narayanaswamy, Subhashini Venugopalan, Kasumi Widner, Tom Madams, Jorge Cuadros, Ramasamy Kim, Rajiv Raman, Philip C. Nelson, Jessica L. Mega, and Dale R. Webster. Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA*, 316(22):2402–2410, 12 2016. ISSN 0098-7484. doi: 10.1001/jama.2016.17216. URL <https://doi.org/10.1001/jama.2016.17216>.
- [14] P. C. Hanson, R. Ladwig, C. Buelo, E. A. Albright, A. D. Delany, and C. C. Carey. Legacy phosphorus and ecosystem memory control future water quality in a eutrophic lake. *Journal of Geophysical Research: Biogeosciences*, 128(12):e2023JG007620, 2023. doi: <https://doi.org/10.1029/2023JG007620>. URL <https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2023JG007620>. e2023JG007620 2023JG007620.
- [15] Paul C. Hanson, Aviah B. Stillman, Xiaowei Jia, Anuj Karpatne, Hilary A. Dugan, Cayelan C. Carey, Jemma Stachelek, Nicole K. Ward, Yu Zhang, Jordan S. Read, and Vipin Kumar. Predicting lake surface water phosphorus dynamics using process-guided machine learning. *Ecological Modelling*, 430:109136, August 2020. ISSN 03043800. doi: 10.1016/j.ecolmodel.2020.109136.
- [16] Ted D. Harris and Jennifer L. Graham. Predicting cyanobacterial abundance, microcystin, and geosmin in a eutrophic drinking-water reservoir using a 14-year dataset. *Lake and Reservoir Management*, 33(1):32–48, 2017. doi: 10.1080/10402381.2016.1263694. URL <https://doi.org/10.1080/10402381.2016.1263694>.
- [17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning

- for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 770–778. IEEE, June 2016. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.90. URL <http://ieeexplore.ieee.org/document/7780459/>.
- [18] Matthew R. Hipsey, Louise C. Bruce, Casper Boon, Brendan Busch, Cayelan C. Carey, David P. Hamilton, Paul C. Hanson, Jordan S. Read, Eduardo De Sousa, Michael Weber, and Luke A. Winslow. A General Lake Model (GLM 3.0) for linking with high-frequency sensor data from the Global Lake Ecological Observatory Network (GLEON). *Geoscientific Model Development*, 12(1):473–523, January 2019. ISSN 1991-9603. doi: 10.5194/gmd-12-473-2019. URL <https://gmd.copernicus.org/articles/12/473/2019/>.
- [19] Carol Hodanbosi and Jonathan G. Fairman. Buoyancy: Archimedes principle, Aug 1996. URL https://www.grc.nasa.gov/WWW/k-12/WindTunnel/Activities/buoy_Archimedes.html.
- [20] Christopher J. Shallue and Andrew Vanderburg. Identifying exoplanets with deep learning: A five-planet resonant chain around kepler-80 and an eighth planet around kepler-90. *The Astronomical Journal*, 155(94), Jan 2018. doi: 10.3847/1538-3881/aa9e09. URL <https://iopscience.iop.org/article/10.3847/1538-3881/aa9e09/pdf>.
- [21] Xiaowei Jia, Jared Willard, Anuj Karpatne, Jordan S. Read, Jacob A. Zwart, Michael Steinbach, and Vipin Kumar. Physics-guided machine learning for scientific discovery: An application in simulating lake temperature profiles. *ACM/IMS Transactions on Data Science*, 2(3):1–26, August 2021. ISSN 2691-1922. doi: 10.1145/3447814.
- [22] Barbara L. Micale. Virginia tech faculty receive national artificial intelligence research resource pilot awards, Jun 2024. URL <https://news.vt.edu/articles/2024/06/three-virginia-tech-faculty-receive-nairr-pilot-awards-for-proje.html>.

- [23] Hao Li, Zheng Xu, Gavin Taylor, Christoph Studer, and Tom Goldstein. Visualizing the loss landscape of neural nets. In *32nd Conference on Neural Information Processing Systems*, 2018.
- [24] Liyang Liu, Yi Li, Zhanghui Kuang, Jing-Hao Xue, Yimin Chen, Wenming Yang, Qingmin Liao, and Wayne Zhang. Towards impartial multi-task learning. In *9th International Conference on Learning Representations*, 2021.
- [25] Qiang Liu, Mengyu Chu, and Nils Thuerey. Config: Towards conflict-free training of physics informed neural networks. In *13th International Conference on Learning Representations*, 2024. URL <https://arxiv.org/abs/2408.11104>.
- [26] John J. Magnuson, Larry B. Crowder, and Patricia A. Medvick. Temperature as an ecological resource. *American Zoologist*, 19(1):331–343, Feb 1979. ISSN 0003-1569. doi: 10.1093/icb/19.1.331. URL <https://doi.org/10.1093/icb/19.1.331>.
- [27] Rafael Marcé, Lluís Gómez-Gener, and Cayelan C. Carey. *Oxygen*, page 237–274. Elsevier, 2024. ISBN 9780128227015. doi: 10.1016/B978-0-12-822701-5.00011-2. URL <https://linkinghub.elsevier.com/retrieve/pii/B9780128227015000112>.
- [28] James L. Martin and Steven C. McCutcheon. *Hydrodynamics and Transport for Water Quality Modeling*. CRC Press, 1999.
- [29] Bennett J. McAfee, Aanish Pradhan, Abhilash Neog, Sepideh Fatemi, Robert T. Hensley, Mary E. Lofton, Anuj Karpatne, Cayelan. C. Carey, and Paul. C. Hanson. Lakebed-us: a benchmark dataset for lake water quality time series and vertical profiles. *Earth System Science Data Discussions*, 2025:1–43, 2025. doi: 10.5194/essd-2025-27. URL <https://essd.copernicus.org/preprints/essd-2025-27/>.
- [30] Jeffrey Mervis. New u.s. ai network aims to make supercomputers available to

- more researchers: Pilot grants will help scientists train software to tackle societal problems, May 2024. URL <https://www.science.org/content/article/new-u-s-ai-network-aims-make-supercomputers-available-more-researchers>.
- [31] Ishan Misra, Abhinav Shrivastava, Abhinav Gupta, and Martial Hebert. Cross-stitch networks for multi-task learning. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 3994–4003. IEEE, June 2016. ISBN 978-1-4673-8851-1. doi: 10.1109/CVPR.2016.433. URL <http://ieeexplore.ieee.org/document/7780802/>.
- [32] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer Series in Operations Research. Springer, 2nd edition, 2006. ISBN 9780387303031.
- [33] U.S. Department of Energy. Artificial intelligence for science, 2025. URL <https://www.energy.gov/topics/artificial-intelligence-science>.
- [34] Bureau of Reclamation California-Great Basin. Water facts - worldwide water supply, 2020. URL <https://www.usbr.gov/mp/arwec/water-facts-ww-water-sup.html>.
- [35] Hans W. Paerl and jef Huisman. Blooms like it hot. *Science*, 320(5872):57–58, 2008. doi: 10.1126/science.1155398. URL <https://www.science.org/doi/abs/10.1126/science.1155398>.
- [36] Aanish Pradhan, Bennett J. McAfee, Abhilash Neog, Sepideh Fatemi, Mary E. Lofton, Cayelan C. Carey, Anuj Karpatne, and Paul C. Hanson. LakeBeD-US: Computer Science Edition - a benchmark dataset for lake water quality time series and vertical profiles, 2024. URL <https://huggingface.co/datasets/eco-kgml/LakeBeD-US-CSE>.
- [37] Frank J. Rahel and Julian D. Olden. Assessing the effects of climate change on aquatic invasive species. *Conservation Biology*, 22(3):521–533, 2008. doi: <https://doi.org/10.1111/j.1523-1739.2008.01211.x>.

- //doi.org/10.1111/j.1523-1739.2008.00950.x. URL <https://conbio.onlinelibrary.wiley.com/doi/abs/10.1111/j.1523-1739.2008.00950.x>.
- [38] James J. Roberts, Kurt D. Fausch, Douglas P. Peterson, and Mevin B. Hooten. Fragmentation and thermal risks from climate change interact to affect persistence of native trout in the colorado river basin. *Global Change Biology*, 19(5):1383–1398, May 2013. ISSN 1354-1013, 1365-2486. doi: 10.1111/gcb.12136.
- [39] James J. Roberts, Kurt D. Fausch, Mevin B. Hooten, and Douglas P. Peterson. Non-native trout invasions combined with climate change threaten persistence of isolated cutthroat trout populations in the southern rocky mountains. *North American Journal of Fisheries Management*, 37(2):314–325, 02 2017. ISSN 0275-5947. doi: 10.1080/02755947.2016.1264507. URL <https://doi.org/10.1080/02755947.2016.1264507>.
- [40] Kevin C. Rose. *Light in Inland Waters*, pages 75–94. Elsevier, 2024. ISBN 978-0-12-822701-5. doi: 10.1016/B978-0-12-822701-5.00006-9. URL <https://linkinghub.elsevier.com/retrieve/pii/B9780128227015000069>.
- [41] Conrado M. Rudorff, John M. Melack, Sally MacIntyre, Cláudio C. F. Barbosa, and Evlyn M. L. M. Novo. Seasonal and spatial variability of co2 emission from a large floodplain lake in the lower amazon. *Journal of Geophysical Research: Biogeosciences*, 116(G4), 2011. doi: <https://doi.org/10.1029/2011JG001699>. URL <https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2011JG001699>.
- [42] Chris Shallue and Andrew Vanderburg. Earth to exoplanet: Hunting for planets with machine learning, Dec 2017. URL <https://blog.google/technology/ai/hunting-planets-machine-learning/>.
- [43] Parshin Shojaee, Kazem Meidani, Shashank Gupta, Amir Barati Farimani, and Chan-

- dan K Reddy. Llm-sr: Scientific equation discovery via programming with large language models, 2025. URL <https://arxiv.org/abs/2404.18400>.
- [44] M Tanaka, G Girard, R Davis, A Peuto, and N Bignell. Recommended table for the density of water between 0 °c and 40 °c based on recent experimental reports. *Metrologia*, 38(4):301, aug 2001. doi: 10.1088/0026-1394/38/4/3. URL <https://dx.doi.org/10.1088/0026-1394/38/4/3>.
- [45] R.A. Vollenweider and J. Kerekes. The loading concept as basis for controlling eutrophication philosophy and preliminary results of the oecd programme on eutrophication. In S.H. Jenkins, editor, *Eutrophication of Deep Lakes*, pages 5–38. Pergamon, 1980. ISBN 978-0-08-026024-2. doi: <https://doi.org/10.1016/B978-0-08-026024-2.50005-5>. URL <https://www.sciencedirect.com/science/article/pii/B9780080260242500055>.
- [46] Runlong Yu, Chonghao Qiu, Robert Ladwig, Paul Hanson, Yiqun Xie, and Xiaowei Jia. Physics-guided foundation model for scientific discovery: An application to aquatic science. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 28548–28556, Apr 2025. doi: 10.1609/aaai.v39i27.35078. URL <https://ojs.aaai.org/index.php/AAAI/article/view/35078>.
- [47] Tianhe Yu, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. Gradient surgery for multi-task learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/3fe78a8acf5fda99de95303940a2420c-Paper.pdf.
- [48] Nicolas Zucchet and Antonio Orvieto. Recurrent neural networks: vanishing and exploding gradients are not the end of the story. In A. Globerson, L. Mackey, D. Bel-

grave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 139402–139443. Curran Associates, Inc., 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/fbb07254ef01868967dc891ea3fa6c13-Paper-Conference.pdf.

Appendices

Appendix A

Data: Supplementary Information

A.1 Missing Values Analysis

Figures [A.1](#) to [A.4](#) show heatmaps that visualize the percentage of missing values for our ecological variables of interest. If we envision the data as a third-order tensor where the rows are depths, the columns are variables and the slices (i.e. third dimension) are datetimes at a daily median resolution, these heatmaps show the percentage of missing values for each variable at a given depth. White spaces indicate depth-variable combinations for which no observations were recorded over the entire time period.

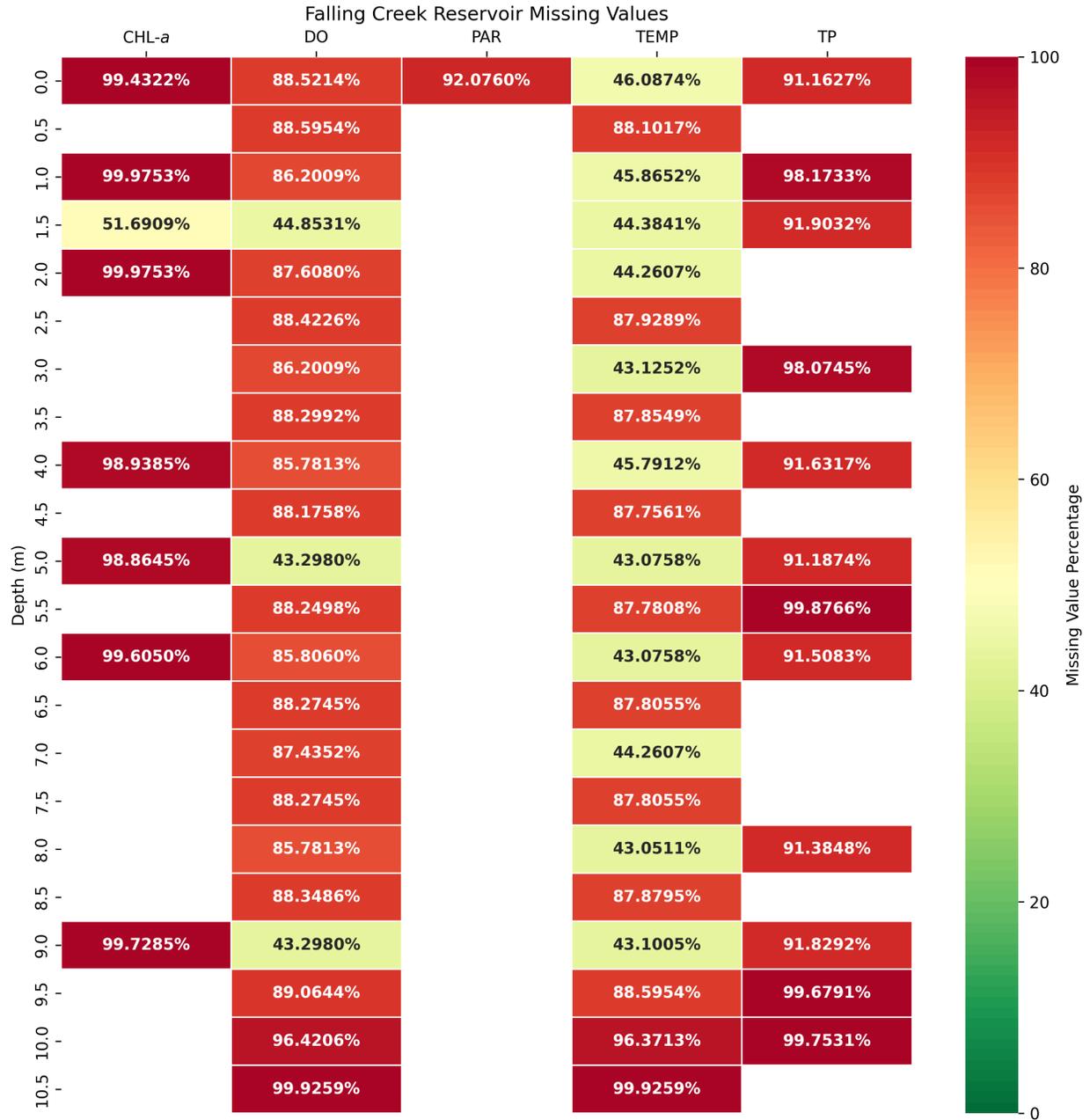


Figure A.1: Missing value percentages for Falling Creek Reservoir.

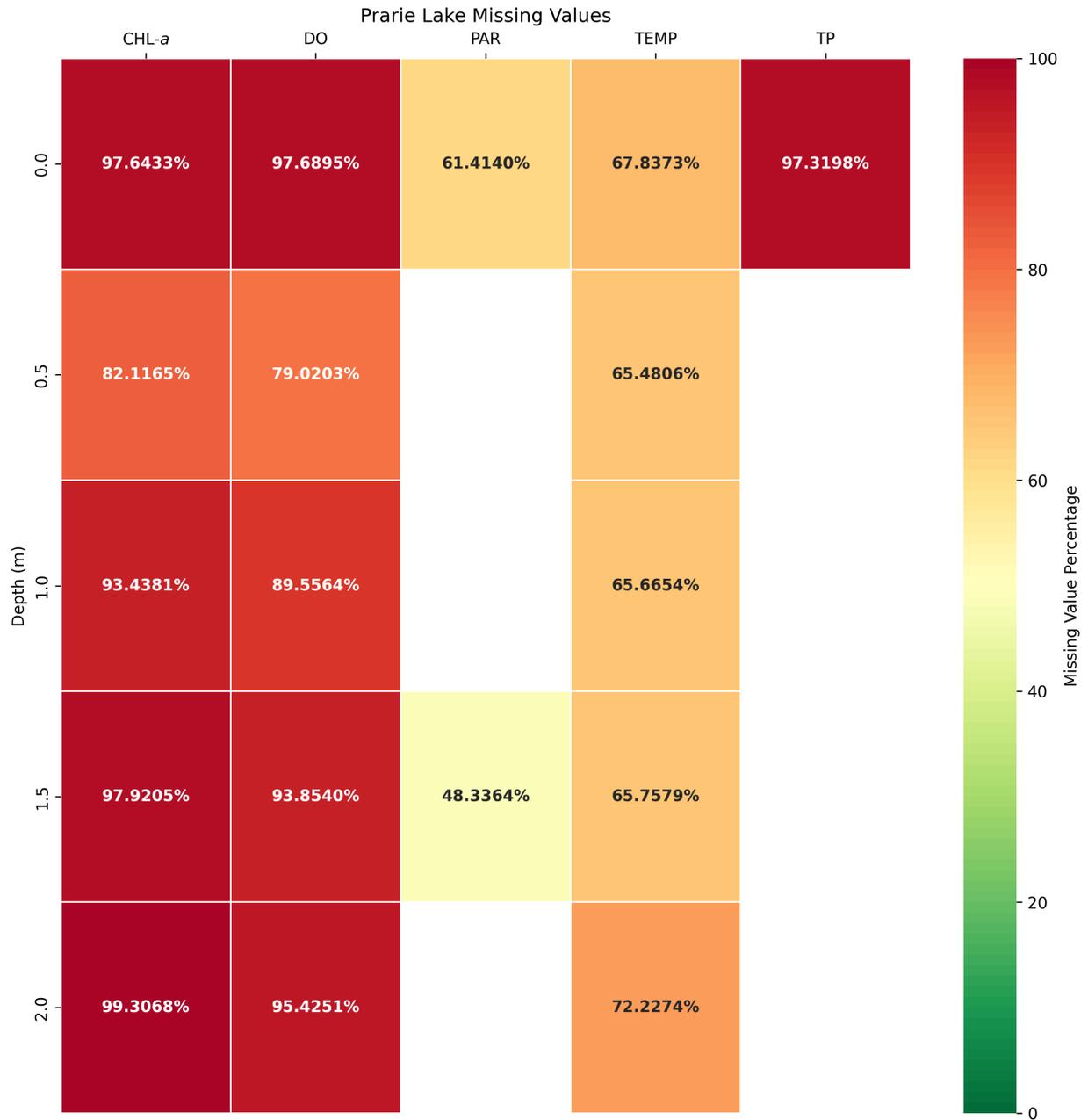


Figure A.2: Missing value percentages for Prairie Lake.

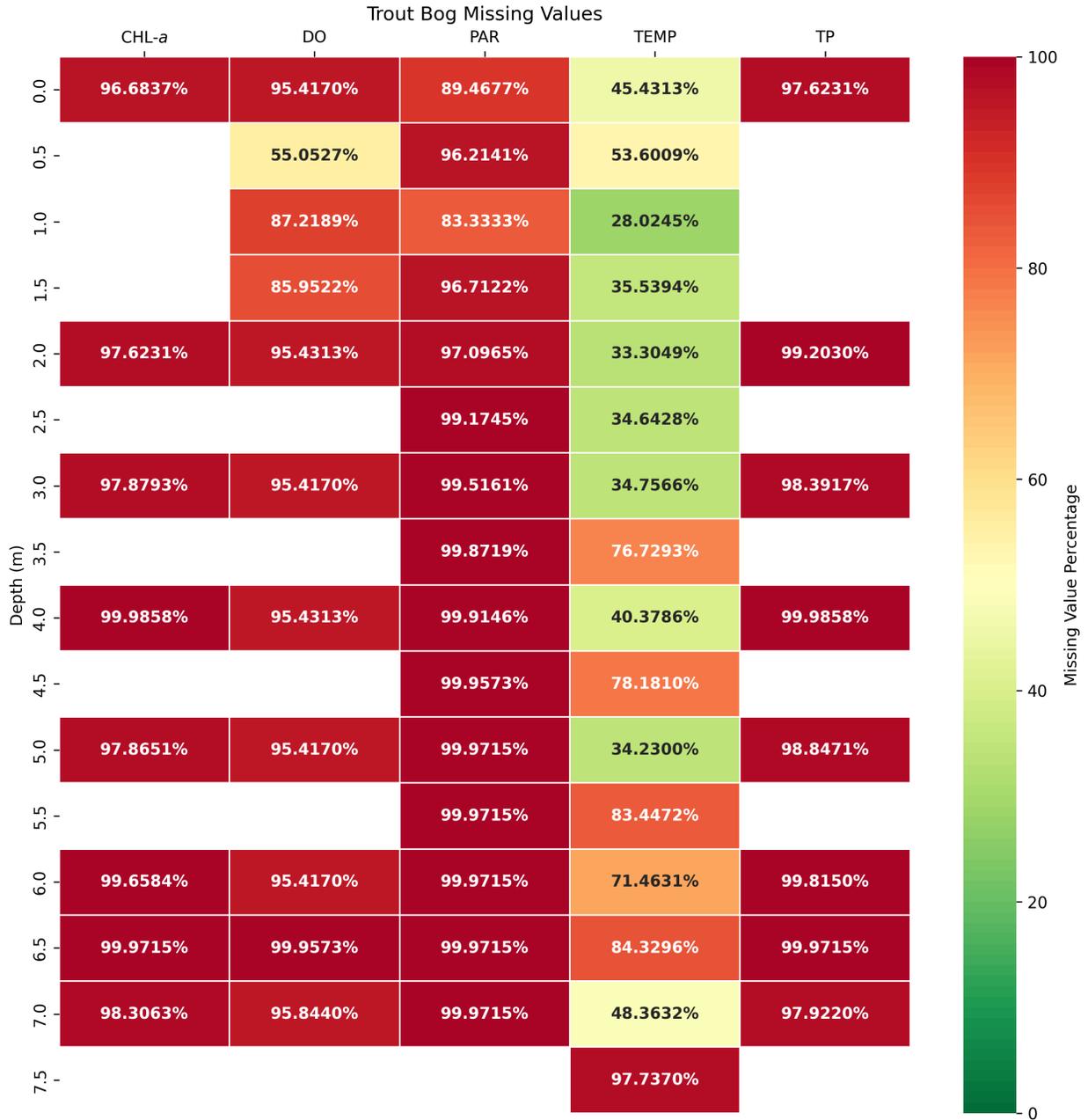


Figure A.3: Missing value percentages for Trout Bog.

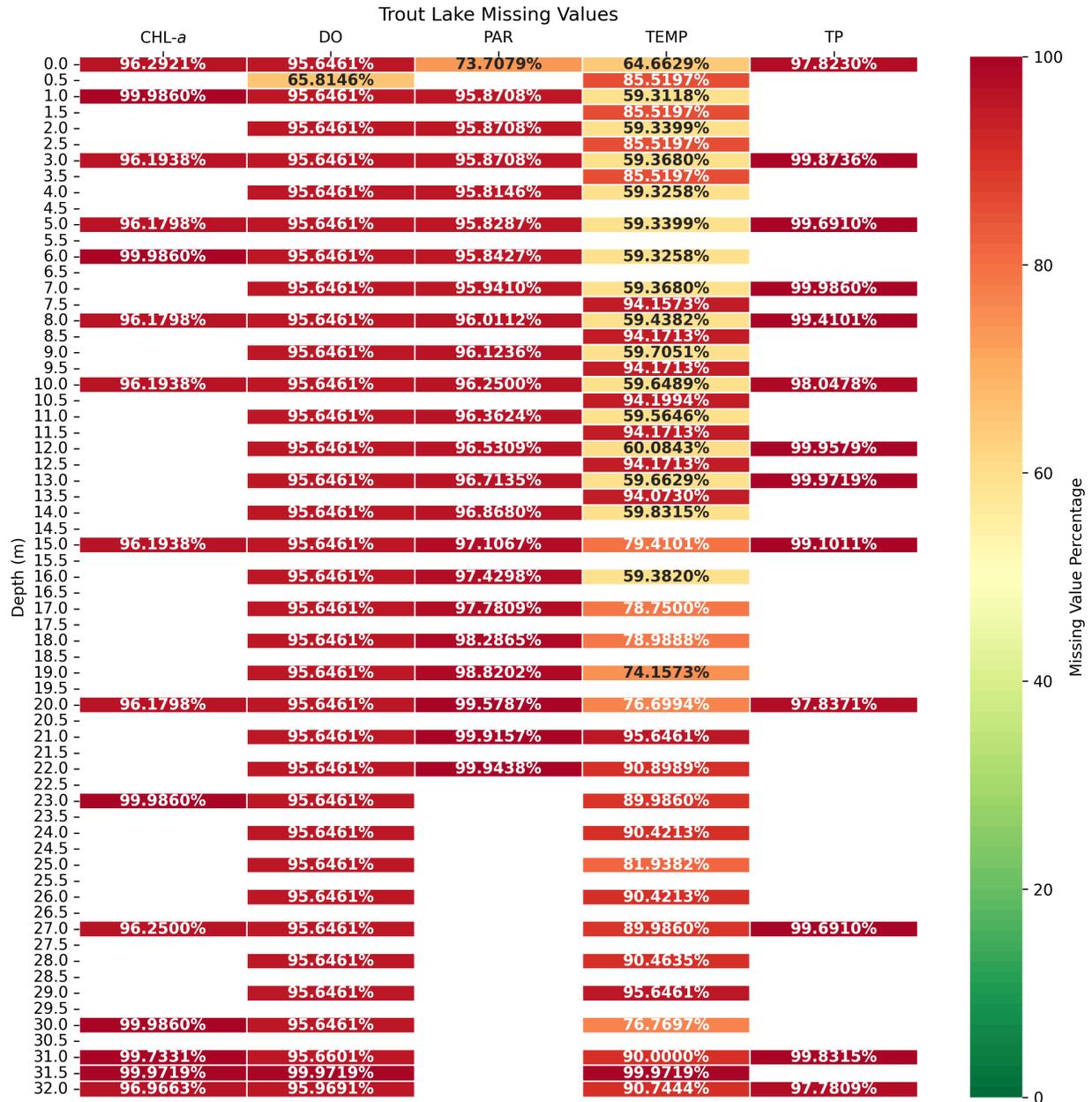


Figure A.4: Missing value percentages for Trout Lake.