Data-Driven Machine Learning Aided Fine Scale Reconstruction for Hierarchical Parameter Systems and the Geometry of Adaptive Learners

by

Chinedu Eleh

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
August 9, 2025

Keywords: conditional simulation, variational autoencoders, partial differential equations, uncertainty quantification, finite element analysis, deep learning

Approved by

Hans-Werner van Wyk, Chair, Associate Professor Mathematics and Statistics
Yanzhao Cao, Professor and Don Logan Endowed Chair of Mathematics and Statistics
Junshan Lin, Professor of Mathematics and Statistics
Roberto Molinari, Assistant Professor of Mathematics and Statistics
Yang Zhou, Associate Professor of Computer Science and Software Engineering

Abstract

While statistical uncertainty is observable in many real-world systems and data, its incorporation into numerical prediction and optimization algorithms presents numerous challenges in terms of accuracy, efficiency and storage costs. In this dissertation, we make two contributions to numerical algorithms that account for uncertainties.

In the first part, we discuss the statistical simulation of a known physical system with spatially varying random model parameters. The resulting ensembles of model outputs can be used to determine statistical properties, such as means and confidence bounds, of related quantities of interest. In these simulations, the complexity (dimensionality) of the system parameter's statistical description, which often reflects the spatial scale at which the parameters are resolved, may affect both the statistical accuracy of resulting outputs and the spatial accuracy of the model solution, especially when the parameter fields are rough. In particular, very low-complexity models, while amenable to efficient sampling methods, may introduce a statistical modeling error by underestimating the true variation in the output. On the other hand, high-dimensional parameter representations, while more detailed, may result in model responses that require a higher resolution computational mesh, thereby increasing the computational cost. We propose a method to generate high-complexity samples conditional from low-complexity ones by means of a conditional variational autoencoder.

In the second part of the dissertation, we propose GeoAdaler, a new stochastic optimization algorithm, whose step length selection is based on a scalar measure of steepness of the objective function's sample gradient, based on the angle of its normal vector with the horizontal plane. We show how the resulting family of algorithms relate to existing adaptive steplength selection techniques and establish regret bounds in the case of convex functions.

We also demonstrate numerically that GeoAdaLer is capable of being extended to nonconvex settings.

Acknowledgments

First and foremost, I thank God Almighty for good health, body, mind, and soul - throughout this PhD journey. Honestly, I wouldn't have made it without God's grace. I am deeply grateful to my advisor, Dr. Hans-Werner van Wyk, for his academic guidance and fatherly mentorship. Under your care, I've not only grown intellectually but also become a better version of myself. To my PhD committee members Drs. Yanzhao Cao, Junshan Lin, Roberto Molinari, and my University Reader Dr. Yang Zhou, thank you for generously offering your time and expertise. Your insightful feedback has improved my dissertation and will definitely shape my future publications.

A special thank you to the Department of Mathematics and Statistics for supporting me financially through a Graduate Teaching Assistantship, not just during my PhD, but also through my Master's in Data Science. To the professors who visited my class and gave feedback that shaped my teaching style, thank you! I must also thank Dr. Kimberly Mulligan for believing in me (even more than I believed in myself). You gave me the opportunity to mentor and tutor the leaders of tomorrow.

To my fellow travelers in the Parker Hall hallways - Masuzyo, Ukamaka, Hewan, Ogonna, Jordan, Yi, Eyoel (and many more) - thank you for keeping me sane and smiling through the chaos. I owe a mountain of gratitude to my family and friends: Theresa, Esther, Emeka, Chidimma, Blessing, Amarakristi, Nathaniel, Mariam, Salako, Ekene, Clint, Allison, Marie... I can't name everyone, but know this, your support made all the difference.

Now, when the academics weren't working, there were other things and people that did. To the choir at St. Michael the Archangel Catholic Church in Auburn, especially Dr. Jane Kuehne and Dr. Victor Nelson - thank you for helping me grow musically and for giving me a beautiful outlet beyond research. God bless you all!

Finally, Auburn is full of wonderful people and communities that made me feel right at home. I'll miss the Soccer Pickup group, One Cup Ministry with Marie, the F3 Nation crew, and all the joy you brought into this chapter of my life. From the bottom of my heart, thank you all!

Table of Contents

Abstract ii

Acknowledgments iv

List of Figures ix

List of Tables x

1 Introduction and Problem Statement 1

1.1 Multiscale Uncertainty in Computational Science and Engineering 1

1.2 Stochastic Multiscale Methods 2

1.3 Contributions 3

1.4 Outline of the Dissertation 5

2 Background and Theoretical Framework 7

2.1 Functional Setting and Weak Formulation 7

2.2 Finite Element Discretization of Elliptic Problems 9

2.2.1 Finite element spaces 10

2.2.2 Discrete weak problems 10

2.2.3 Matrix formulation and assembly 10

2.2.4 High and Low Complexity Parameters 11

2.3 Variational Multiscale (VMS) Decomposition 12

2.3.1 Decomposition of the Solution 12

2.3.2 fine scale problem driven by the coarse residual 12

2.3.3 Discrete VMS formulation 14

2.4 Multifidelity Variance Reduction for Quantities of Interest 14

2.4.1 Control-variate (CV) Monte Carlo 14

2.4.2 Multilevel Monte Carlo (MLMC) 15

2.4.3 Bayesian Multifidelity Conditioning . . . 15
2.5 From Common Theory to Specialized Surrogates . . . 16
2.6 Low Fidelity and Correction Equations . . . 17
2.6.1 Hierarchical Parameter Enrichment . . . 18
3 CVAE-Based HPC Field Reconstruction . . . 22
3.1 Introduction . . . 22
3.2 Conditional Variational Autoencoders . . . 22
3.2.1 Autoencoders . . . 23
3.2.2 Variational Autoencoders (VAE) . . . 25
3.2.3 Conditional Variational Autoencoders . . . 27
3.2.4 CVAE Architecture and Loss . . . 28
3.2.5 Evidence Lower Bound with Physics Regularization . . . 30
3.2.6 Generation of LPC-HPC Training Pairs . . . 31
3.3 Numerical experiments . . . 32
3.3.1 One-dimensional Elastic Bar . . . 32
3.3.2 Data generation . . . 33
3.3.3 Network architecture and training protocol . . . 33
3.3.4 Test set and model evaluation . . . 34
3.3.5 Results . . . 34
4 GeoAdaLer: Geometric Insights into Adaptive Stochastic Gradient Descent Algorithms . . . 37
4.1 Introduction . . . 37
4.2 Related Work . . . 38
4.3 Mathematical Formulation . . . 40
4.3.1 Deterministic Optimization . . . 40
4.3.2 Properties of GeoAdaLer Annealing Factor . . . 43
4.3.3 Stochastic Optimization . . . 43

4.4 Relationship to the AdaGrad Family . . . 46
4.4.1 GeoAdaLer in Comparison . . . 48
4.5 Convergence Analysis . . . 49
4.5.1 Deterministic Setting . . . 49
4.5.2 Stochastic Setting . . . 50
4.6 Experiments . . . 52
4.6.1 MNIST Dataset . . . 53
4.6.2 CIFAR-10 Dataset . . . 53
4.6.3 Fashion MNIST . . . 56
4.6.4 Alternative Normal Plane Vectors . . . 56
4.7 Conclusion . . . 57
5 Summary and Future Work . . . 59
5.1 Summary . . . 59
5.2 Future Work . . . 60
Bibliography . . . 62
.1 Deterministic Convergence Proof . . . 69
.2 Stochastic Convergence Proof . . . 72
.2.1 Important Lemmas and Definitions . . . 72
.3 Proof of Theorem 4.4 . . . 75

List of Figures

3.1 An Autoencoder Neural Networks Architecture 23

3.2 CVAE Architecture for Reconstructing HPC Solutions 29

3.3 Visual comparison of CVAE and FEM solutions conditioned on one LPC solution. The right hand side is $f(x) = (x - \pi)^2$. The LPC field is generated with $m_L = 2$ KL modes, whereas the HPC field uses 100 modes. 35

3.4 Statistical comparison of FEM and CVAE predictions. 36

4.1 Geometric view of θ as the GeoAdaLer step traverses the objective function. . . 41

4.2 MNIST 53

4.3 CIFAR 10 Dataset 55

4.4 Fashion MNIST Dataset 57

4.5 MNIST Dataset 58

4.6 CIFAR 10 Dataset 58

List of Tables

4.1 MNIST Final Accuracy 55

4.2 CIFAR Final Accuracy 55

4.3 Fashion MNIST Final Accuracy 56

Chapter 1

Introduction and Problem Statement

1.1 Multiscale Uncertainty in Computational Science and Engineering

Many contemporary models in materials science, geophysics, subsurface flow, and energy systems are governed by partial differential equations (PDEs) whose coefficients are only partially known. Uncertain model parameters are often modeled statistically, either as random variables in the case of scalar parameters, or as *random fields* in the case of spatially varying ones. A random field is a collection of random variables indexed over a multidimensional domain [83, 1, 74, 56]. Spatial fluctuations in permeability [3], thermal conductivity [55], or elastic moduli [2] act across several characteristic lengths. When such coefficients are modeled as hierarchical statistical parametrizations, such as Karhunen-Loève expansions or in a hierarchy of spatial scales, these can be truncated to obtain a simplified representation, giving rise to *hierarchical parameter systems*. An accepted statistical parametrization is the Karhunen-Loève (KL) expansion [74]. Truncating this expansion after the first K dominant modes produces a *low parameter complexity (LPC)* representation that is computationally convenient, yet the neglected *high parameter complexity (HPC)* tail may still drive crucial fine scale solution features.

Solving a stochastic forward problem on the full HPC field with classical finite element or finite volume method [3, 17] quickly becomes *super* expensive since (i) the mesh must resolve the smallest spatial correlation length (ii) intrusively coupled Stochastic Galerkin solvers could cost as high as $\mathcal{O}(NM^3)$ for direct solvers and $\mathcal{O}(NM^2)$ when the sparsity structures are efficiently exploited, where M is the sample size and N is the number of spatial degrees of freedom [19, 81] and (iii) each additional KL mode increases dimensionality, exacerbating the *curse of dimensionality*. Homogenization and representative element methods [39, 40]

alleviate this by working on coarser parameter scales, but inevitably smear out the HPC influence.

1.2 Stochastic Multiscale Methods

Stochastic multiscale methods are used to solve PDEs whose random coefficients have a correlation length much smaller than the domain scale. Directly resolving every microscale fluctuation for each Monte-Carlo sample needed to estimate expectations or variances or moments of quantities of interest becomes prohibitively expensive. Homogenization, multiscale finite elements, variational-multiscale closures, multilevel Monte Carlo, and emerging operator-learning surrogates all reduce this cost by exploiting scale separation while preserving accuracy in the desired statistics [6, 30, 23].

Homogenization. Classical (periodic or statistical) homogenization replaces a highly oscillatory coefficient field by a single effective tensor obtained from cell or representative-volume problems [38]. Because the effective tensor is independent of a particular realization, homogenization is accurate for ensemble averages but cannot reproduce pathwise fine-scale variability; hence no guarantee for quantification of the uncertainty induced by neglected high modes.

Multiscale finite elements (MsFEM). The MsFEM and its generalized variants construct local basis functions by solving fine-scale problems inside each coarse element, thereby adapting the space to the coefficient field and achieving high pathwise accuracy [40, 23]. However, the local problems must be recomputed for every stochastic sample, producing a deterministic surrogate with no intrinsic uncertainty quantification and an offline cost that grows linearly with the number of required realizations.

Variational multiscale (VMS) closure. Variational multiscale formulations embed unresolved scales through fine-scale Green's functions, preserving the governing physics [42]. Stochastic VMS extensions inherit this fidelity but require an explicit closure for each coefficient realization, again leading to costs proportional to the number of KL modes.

Data-driven operator learning. Neural operators such as DeepONet and the Fourier Neural Operator learn a global map $q \mapsto u$, offering mesh-independent inference [57, 49]. Conditional generative variants incorporate epistemic uncertainty [80]. Because these networks ingest the entire field, both training and inference costs scale with every additional KL mode, and no use is made of inexpensive low parameter complexity (LPC) information already available in many workflows.

Multilevel Monte Carlo (MLMC). MLMC reduces uncertainty quantification costs by sampling on a hierarchy of mesh resolutions. It performs many cheap simulations on coarse grids, progressively fewer on intermediate grids, and only a handful on the finest grid that captures the low frequency features of the medium. By carefully combining the results from each level, MLMC attains a target root mean square accuracy with a computational effort that grows roughly like the inverse square of the desired error, yet it still needs only a small number of fully resolved solves to anchor the estimate [30]. This variance reduction strategy complements model side techniques such as homogenization, MsFEM, VMS, and neural operator surrogates by exploiting scale separation purely through smart sampling in discretization space.

1.3 Contributions

We develop a fully probabilistic pipeline centered on a conditional variational autoencoder (CVAE) for quantifying the influence of the high parameter complexity (HPC) tail in elliptic problems with random coefficients. We write the random field as a Karhunen-Loève series and distinguish the LPC q^L and the HPC correction q' derived from $q' = q^H - q^L$. The solution inherits the decomposition $u^H = u^L + u'$. Unlike mesh-based "coarse-fine" decompositions, this partition is purely statistical. It helps us separate the error made by truncating the KL expansion from all other numerical errors.

Using the weak form, we obtain an exact residual equation for u' whose forcing involves the inexpensive solution u^L and the correction coefficient q'. This mirrors residual formulations in variational multiscale (VMS) theory but is expressed entirely in parameter space, providing a clean analytic framework for learning u'.

Building on the CVAE framework of [73], we train a CVAE model that maps the low complexity solution u^L to a distribution over u'. Because the input to the network contains only information from the coarse modes, the network's dimensionality and its evaluation time grow with the LPC truncation level but are independent of the HPC modes. Conditioning on u^L anchors every sample to a physics-consistent baseline, giving calibrated uncertainty bands for the reconstructed fine scales.

We anticipate that there are other ways to achieve this same goal such as learning the fine scale stiffness operator associated with (2.1) in latent form and apply GMRES to obtain u' while preserving Galerkin structure. Inspired by the variational multiscale (VMS) framework of Hughes [42], one could also embed physics directly into a neural super-resolution of the Green's function, capturing nonlocal HPC effects. While the last two ideas are promising, we do not pursue them further in this work.

All HPC solves are performed once, offline, to populate the training set, an investment comparable to that required by other data-driven surrogates. During inference, each new realization needs only (i) one deterministic LPC finite-element solve and (ii) a single network evaluation. Compared with plain Monte-Carlo sampling on the full HPC mesh, this residual strategy removes repeated high fidelity solves altogether and, in our numerical experiments, cuts wall-time by one to two orders of magnitude while preserving the accuracy of quantities of interest. The saving is conceptually akin to that achieved by Multilevel Monte Carlo method, where only a handful of fine meshes are sampled to delineates hierarchy, but achieved here through learned reconstruction rather than variance reduction.

Additionally, we foresee further computational savings from expanding on hardware that cheaply evaluates our trained CVAE model during inference. Modern inference chips

designed for large language model chat bots favor exactly the dense matrix multiplications that dominate a forward pass through our trained CVAE. Authors of [44] report that a single second generation Tensor Processing Unit (TPU) sustains tens of tera operations per second while drawing only a few hundred watts, giving an order of magnitude advantage in throughput-per-watt over general purpose CPUs for the same dense workload. Running the CVAE on such hardware therefore adds almost no marginal cost once the low complexity finite element solve is finished.

Classical high resolution finite element calculations do not benefit to the same degree. Their kernels are dominated by sparse matrix assembly and irregular memory access. Careful GPU implementations still reach only a small fraction of the nominal peak performance. As shown in [12] for low order elements and in [4] for high order elements, there is GPU speed up of at most a few times over CPU baselines, largely constrained by memory bandwidth rather than arithmetic capability. In short, the inductive bias that makes the finite element method flexible on unstructured meshes also limits its ability to exploit the newest inference hardware, whereas our CVAE surrogate is naturally aligned with it.

1.4 Outline of the Dissertation

The remainder of this dissertation is organized as follows. In Chapter 2, we review functional analysis and finite element foundations, introduce the variational multiscale (VMS) decomposition, and position multifidelity variance reduction within stochastic PDE theory. In Chapter 3, we formalize a physics regularized conditional variational autoencoder (CVAE) that reconstructs high precision fine fields guided by low precision inputs and demonstrate its effectiveness through detailed numerical studies.

In Chapter 4, we present *GeoAdaLer*, a geometry aware adaptive stochastic gradient algorithm that drives deep learning training pipelines and comes with proven deterministic and stochastic convergence guarantees. In Chapter 5, we synthesize the main findings and

outline promising directions for future research. we collect technical proofs and auxiliary material in the appendices.

Chapter 2

Background and Theoretical Framework

2.1 Functional Setting and Weak Formulation

Let $D \subset \mathbb{R}^d$ ($d \in \{1, 2, 3\}$) be a bounded Lipschitz domain with outward unit normal $\mathbf{n}$. Throughout, we employ standard notation: $\mathbb{R}$ for the reals, bold symbols for vectors, $(\cdot, \cdot)$ for $L^2(D)$ inner products, and $\|\cdot\|_{H^1(D)}$ for Sobolev norms. Let $q(x, \xi)$ be a lognormal diffusion coefficient on a domain $D \subset \mathbb{R}^d$, expanded as

$$\log q(x, \xi) = \mu(x) + \sum_{k=1}^{\infty} \sqrt{\lambda_k}\, \phi_k(x)\, \xi_k, \qquad \xi_k \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, 1).$$

For a prescribed cut off K we write

$$q^{\mathrm{L}} := q\big(x, \xi_1, \ldots, \xi_K\big), \qquad q^{\mathrm{H}} := q - q^{\mathrm{L}},$$

and interpret q^{L} as the *coarse* statistical description and q^{H} as a fine stochastic correction. The forward model of interest is the steady state elliptic boundary value problem

$$-\nabla \cdot \big(q(x, \xi)\, \nabla u(x, \xi)\big) = f(x) \quad \text{in } D, \qquad u = 0 \text{ on } \partial D, \tag{2.1}$$

whose quantity of interest (QoI) may be a linear functional $\mathcal{Q}[u]$ or pointwise evaluations. Directly resolving (2.1) for many ξ samples remains out of reach on modern high performance computing clusters when $K \gg 10$; yet neglecting q^{H} produces biased estimates of $\mathbb{E}[\mathcal{Q}[u]]$ and distorted uncertainty bands.

Following the steady elliptic model introduced in Chapter 1, we consider

$$\begin{cases} -\nabla\cdot\big(q(x,\xi)\,\nabla u(x,\xi)\big) \;=\; f(x) & \text{in } D, \\ u(x,\xi) \;=\; 0 & \text{on } \partial D, \end{cases} \tag{2.2}$$

where ξ is a finite or countably infinite collection of random variables on a complete probability space $(\Omega, \mathcal{F}, \mathbb{P})$; $q(x,\xi)$ denotes the *high parameter complexity* (HPC) random field, modeled as a truncated Karhunen-Loève (KL) expansion with N_{H} modes; $f \in L^2(D)$ is a deterministic forcing term.

Sobolev setting. We adopt the usual Sobolev space notation

$$V \;:=\; H_0^1(D) \quad \text{with norm} \quad \|v\|_V^2 := \int_D |\nabla v|^2 \,\mathrm{d}x.$$

For ease of exposition we assume that the random field satisfies

$$0 < q_{\min} \;\le\; q(x,\xi) \;\le\; q_{\max} < \infty \quad \text{a.s. for a.e. } x \in D, \tag{2.3}$$

which guarantees uniform ellipticity and boundedness [28, 76].

Weak formulation. Multiplying (2.2) by a test function $v \in V$, integrating by parts, and invoking the homogeneous Dirichlet boundary condition yields the *pathwise* weak problem:

Find $u(\cdot,\xi) \in V$ such that, for almost every $\xi \in \Omega$,

$$a_\xi\big(u(\xi), v\big) \;=\; \ell(v) \quad \forall\, v \in V, \tag{2.4}$$

with bilinear and linear forms

$$a_\xi(w,v) := \int_D q(x,\xi)\,\nabla w\cdot\nabla v \,\mathrm{d}x, \qquad \ell(v) := \int_D f\, v \,\mathrm{d}x.$$

Assumption (2.3) implies that $a_\xi(\cdot,\cdot)$ is coercive and continuous on $V \times V$; hence the Lax-Milgram theorem guarantees existence and uniqueness of $u(\cdot,\xi)$ for each realization of ξ [27, 65].

Low parameter complexity surrogate. Retaining only the first $N_{\mathrm{L}} \ll N_{\mathrm{H}}$ KL modes of q defines a *low parameter complexity* (LPC) coefficient $q^L(x,\xi)$ and correspondingly, a coarse solution $u^L(\cdot,\xi) \in V$ satisfying

$$a_\xi^{\mathrm{L}}\big(u^L(\xi), v\big) = \ell(v) \quad \forall v \in V, \qquad \text{where } a_\xi^{\mathrm{L}}(w,v) := \int_D q^L(x,\xi)\nabla w \cdot \nabla v \,\mathrm{d}x. \tag{2.5}$$

We interpret the difference $u - u^L$ as a fine scale correction that encapsulates the modeling error induced by KL truncation. In Section 2.3 we show how this correction can be characterized analytically via a variational multiscale decomposition, forming the cornerstone of the data-driven surrogates developed in the next chapter.

Ensemble perspective. For quantities of interest (QoIs) $Q : V \to \mathbb{R}$ e.g. pointwise pressure, average flux, or risk indicators, we are ultimately interested in statistics such as $\mathbb{E}\big[Q\big(u(\xi)\big)\big]$, where the expectation is over the ξ distribution. For computational purposes, this entails solving (2.4) for many realizations, motivating the surrogate modeling method proposed in Chapter 3.

With the functional analytic framework firmly in place, we next summarize the finite element machinery that converts the weak problems (2.4)–(2.5) into algebraic systems suitable for both high and low parameter complexity simulations.

2.2 Finite Element Discretization of Elliptic Problems

In this section we recall the conforming Galerkin discretization of the weak problems (2.4)-(2.5) introduced in Section 2.1. Throughout, the permeability coefficient is either the *high parameter complexity* field q^{H} or its upscaled form q^{L}.

2.2.1 Finite element spaces

Let $\{\mathcal{T}_h\}_{h\downarrow 0}$ be a family of shape-regular, conforming, simplicial partitions of D indexed by the mesh-size parameter $h := \max_{K\in\mathcal{T}_h} \operatorname{diam}(K)$ [17, 10]. For each h we define the classical continuous piecewise linear subspace

$$V_h \;:=\; \big\{ v_h \in C^0(\overline{D}) : v_h|_K \in \mathbb{P}_1(K)\ \forall K \in \mathcal{T}_h,\ v_h = 0 \text{ on } \partial D \big\} \;\subset\; V.$$

The nodal basis $\{\varphi_i\}_{i=1}^{n_h}$ of V_h is indexed by the interior vertices of $\mathcal{T}_h$, so that every $v_h \in V_h$ can be written as $v_h = \sum_{i=1}^{n_h} v_i \varphi_i$ with coefficient vector $\mathbf{v} \in \mathbb{R}^{n_h}$.

2.2.2 Discrete weak problems

For each realization of ξ and for each coefficient choice $\mathfrak{c} \in \{\mathrm{H}, \mathrm{L}\}$, the finite element approximation $u_h^{\mathfrak{c}}(\xi) \in V_h$ is defined by

$$a_{\xi,h}^{\mathfrak{c}}\big(u_h^{\mathfrak{c}}(\xi), v_h\big) \;=\; \ell(v_h) \quad \forall v_h \in V_h, \tag{2.6}$$

where

$$a_{\xi,h}^{\mathfrak{c}}(w_h, v_h) := \sum_{K\in\mathcal{T}_h} \int_K q^{\mathfrak{c}}(x,\xi)\, \nabla w_h \cdot \nabla v_h \,\mathrm{d}x.$$

Because q^{H} and q^{L} satisfy the uniform ellipticity assumption (2.3), the discrete bilinear form inherits coercivity and continuity. Hence (2.6) is well posed for every ξ [10].

2.2.3 Matrix formulation and assembly

Expanding $u_h^{\mathfrak{c}}(\xi)$ and the test functions in the nodal basis leads to the linear system

$$\mathbf{A}^{\mathfrak{c}}(\xi)\, \mathbf{u}^{\mathfrak{c}}(\xi) \;=\; \mathbf{f}, \tag{2.7}$$

where the stiffness matrix $\mathbf{A}^{\mathfrak{c}}(\xi) \in \mathbb{R}^{n_h \times n_h}$ and load vector $\mathbf{f} \in \mathbb{R}^{n_h}$ have components

$$A_{ij}^{\mathfrak{c}}(\xi) \;:=\; \int_D q^{\mathfrak{c}}(x,\xi)\, \nabla\varphi_j \cdot \nabla\varphi_i \, \mathrm{d}x, \quad f_i \;:=\; \int_D f\, \varphi_i \, \mathrm{d}x.$$

The matrix is symmetric positive-definite for every ξ, so (2.7) can be solved by the Conjugate-Gradient method, multigrid, or sparse direct solvers; see [26, 79].

2.2.4 High and Low Complexity Parameters

For q^{H} the integrals in $A_{ij}^{\mathrm{H}}(\xi)$ must be evaluated with quadrature rules that resolve the fine scale oscillations induced by the high KL modes. In contrast, q^{L} is smoother, enabling coarser quadrature and thus cheaper assembly.

The same mesh $\mathcal{T}_h$ may be used for both problems when performing a *pure parameter study*. In multiscale settings, however, it is common to employ a refined mesh for the HPC problem and reuse the coarse matrix as a *preconditioner* when solving (2.7) with $\mathfrak{c} = \mathrm{H}$ [58, 24, 29].

Standard Céa-type arguments yield, for $\mathfrak{c} \in \{\mathrm{H}, \mathrm{L}\}$ and each ξ,

$$\|u^{\mathfrak{c}}(\xi) - u_h^{\mathfrak{c}}(\xi)\|_V \;\le\; C_{\mathrm{FE}}\, h\, \|u^{\mathfrak{c}}(\xi)\|_{H^2(D)},$$

assuming D is convex and $u^{\mathfrak{c}}(\xi) \in H^2(D)$. The constant C_{FE} is independent of h and ξ [10]. Combined with the analysis of the fine scale correction in Section 2.3, this estimate underpins our surrogate modeling error budget in Chapter 3.

The finite element method described above furnishes the discrete solutions u_h^{H} and u_h^{L} that act as *ground truth* and *conditioning data*, respectively, for the data driven surrogates of Chapter 3. In the next section we revisit the variational multiscale decomposition that links these two levels of resolution.

2.3 Variational Multiscale (VMS) Decomposition

The classical variational multiscale (VMS) framework of Hughes [41, 42] partitions the *solution space* into "coarse" and "fine" scales to stabilize or enrich Galerkin schemes, usually with mesh resolution in mind. In the present work we re-purpose that idea to separate the *parameter space*. Coarse scales correspond to the low-parameter-complexity coefficient q^{L}, whereas fine scales represent the modeling error introduced by truncation of the Karhunen-Loève (KL) expansion. This viewpoint follows the localization principles developed for coefficient-contrast multiscale problems by Målqvist & Peterseim [58].

2.3.1 Decomposition of the Solution

We write the high fidelity solution $u^{\mathrm{H}}(\cdot,\xi) \in V$ (associated with q^{H}) as the *direct sum*

$$u^{\mathrm{H}}(\xi) \;=\; u^{\mathrm{L}}(\xi) \;+\; u'(\xi), \qquad \text{with } u^{\mathrm{L}} \in V_{\mathrm{c}},\; u' \in V_{\mathrm{f}},$$

where $V_{\mathrm{c}} := V$ plays the role of the coarse space and V_{f} is the (abstract) fine scale subspace characterizing parameter induced corrections. For Galerkin FE computations V_{c} will be realized by the same discretization space used for (2.5), but equipped with q^{L}. For our purpose, this discretization space maybe fine or coarse in mesh resolution.

2.3.2 fine scale problem driven by the coarse residual

Insert $u^{\mathrm{H}} = u^{\mathrm{L}} + u'$ into the pathwise weak form (2.4) and subtract the low-complexity problem (2.5):

$$a^{\mathrm{H}}_{\xi}(u', v) \;=\; \big(f - a^{\mathrm{L}}_{\xi}(u^{\mathrm{L}}, \cdot)\big)(v) \quad \forall\, v \in V_{\mathrm{f}}, \tag{2.8}$$

where $a^{\mathrm{H}}_{\xi}(\cdot,\cdot)$ is the bilinear form with q^{H} and the term on the right is the *coarse residual*. Choosing V_{f} as the kernel of a suitable projector $\Pi_{\mathrm{c}} : V \to V_{\mathrm{c}}$ (typically the L^2-orthogonal projection) guarantees $a^{\mathrm{L}}_{\xi}(u^{\mathrm{L}}, v) = \ell(v)$ for all $v \in V_{\mathrm{f}}$; hence the residual reduces to $r_{\xi}(v) := -\big(a^{\mathrm{L}}_{\xi} - a^{\mathrm{H}}_{\xi}\big)\big(u^{\mathrm{L}}, v\big) = \int_D (q^{\mathrm{H}} - q^{\mathrm{L}})\, \nabla u^{\mathrm{L}} \cdot \nabla v \, \mathrm{d}x$.

Provided q^{H} is piecewise H^1 and $V_{\mathrm{f}} \subset H_0^1(D)$, (2.8) is equivalent to

$$-\nabla\cdot\big(q^{\mathrm{H}}\,\nabla u'\big) \;=\; \nabla\cdot\big((q^{\mathrm{H}}-q^{\mathrm{L}})\,\nabla u^{\mathrm{L}}\big) \quad \text{in } D, \qquad u'|_{\partial D} = 0. \tag{2.9}$$

Equation (2.9) reveals three features that are central to our data-driven strategy: (i) the driving term involves the *contrast* $q^{\mathrm{H}}-q^{\mathrm{L}}$ only; (ii) boundary conditions remain homogeneous, so u' is localized around regions where the contrast is large; (iii) no additional dependence on the forcing f appears, making u' reusable across many right-hand sides.

2.3.3 Discrete VMS formulation

Let $V_h \subset V$ be the standard P_1 finite-element space on a mesh $\mathcal{T}_h$, and set $V_{\mathrm{c},h} := V_h$ with coefficients q^{L}. One way to realise the fine space is the orthogonal complement $V_{\mathrm{f},h} := (\mathrm{I} - \Pi_{\mathrm{c},h})V_h$, where $\Pi_{\mathrm{c},h}$ is the q^{L}-weighted H^1 projection. Following the algebraic VMS recipe of [42], elimination of u'_h from the coupled system leads to an effective coarse scale matrix

$$K_{\mathrm{VMS}} \;=\; K^{\mathrm{L}} \;-\; K^{\mathrm{LH}}\,(K^{\mathrm{H}}_{\mathrm{ff}})^{-1}\,K^{\mathrm{HL}},$$

where K^{L} is the standard stiffness matrix with q^{L}, $K^{\mathrm{H}}_{\mathrm{ff}}$ is the restriction of the high fidelity stiffness to fine-fine couplings, and the mixed blocks $K^{\mathrm{LH}}, K^{\mathrm{HL}}$ encode the parameter contrast. This Schur-complement-type operator motivates our data-driven *coarse-aware* reconstruction algorithms in Chapters 4–5.

2.4 Multifidelity Variance Reduction for Quantities of Interest

Let $Q : V \to \mathbb{R}$ be a square-integrable quantity of interest (QoI) and recall the *high-parameter-complexity* (HPC) and *low-parameter-complexity* (LPC) solutions $u^{\mathrm{H}}, u^{\mathrm{L}} \in V$ introduced in section 2.1. Our goal is to estimate the intractable expectation $\mu_H := \mathbb{E}\big[Q(u^{\mathrm{H}})\big]$ with minimal calls to the costly solver for u^{H}. Three well-established families of *multifidelity* estimators exploit the cheap surrogate information contained in u^{L} or $Q(u^{\mathrm{L}})$.

2.4.1 Control-variate (CV) Monte Carlo

Define $\Delta Q := Q(u^{\mathrm{H}}) - Q(u^{\mathrm{L}})$ and write

$$\mu_H \;=\; \underbrace{\mathbb{E}\big[Q(u^{\mathrm{L}})\big]}_{\text{cheap}} \;+\; \underbrace{\mathbb{E}\big[\Delta Q\big]}_{\text{low variance}}\,.$$

Because $\delta(\Delta Q) \ll \delta\big(Q(u^{\mathrm{H}})\big)$ whenever the surrogate is accurate, an i.i.d. Monte-Carlo estimator based on $M_H \ll M_L$ high fidelity samples and M_L cheap samples achieves a mean-square

error of order $\frac{\delta(\Delta Q)}{M_H} + \frac{\delta\left(Q(u^{\mathrm{L}})\right)}{M_L}$ [45]. Optimal allocation of (M_H, M_L) and the coefficient in a generalised CV estimator is analysed in [62].

2.4.2 Multilevel Monte Carlo (MLMC)

MLMC [31, 18] builds a hierarchy of KL truncations $N_{\mathrm{L}} = N_0 < N_1 < \cdots < N_L = N_{\mathrm{H}}$ and telescopes the expectation,

$$\mu_H \;=\; \mathbb{E}\big[Q(u^{(0)})\big] \;+\; \sum_{\ell=1}^{L} \mathbb{E}\big[Q\big(u^{(\ell)}\big) - Q\big(u^{(\ell-1)}\big)\big].$$

Coupled sampling of successive levels (same ξ for the pair in the difference) yields variance decay $\delta\big[Q(u^{(\ell)}) - Q(u^{(\ell-1)})\big] = \mathcal{O}(N_\ell^{-\beta})$ with $\beta > 1$ for sufficiently smooth Q; thus the work to attain RMSE ε drops from $\mathcal{O}(\varepsilon^{-3})$ (plain MC) to $\mathcal{O}(\varepsilon^{-2})$ or better.

2.4.3 Bayesian Multifidelity Conditioning

In the co-kriging framework of [45], $\big(Q(u^{\mathrm{L}}), Q(u^{\mathrm{H}})\big)$ is modeled as a joint Gaussian process with a linear model of co-regionalization prior. Posterior conditioning on cheap evaluations of $Q(u^{\mathrm{L}})$ yields a closed form Gaussian predictive distribution for $Q(u^{\mathrm{H}})$, whose mean is an unbiased minimum variance linear estimator and whose variance supplies credible intervals [72]. This approach is non-intrusive and attractive when $\delta(\Delta Q)$ is small but the underlying PDE solve is highly nonlinear.

The question is, what method is most suited for a given UQ problem? Control-variate MC is simplest and suffices when a single good surrogate is available. MLMC becomes attractive when one can generate a whole ladder of KL truncations or coarse meshes at negligible incremental cost. Co-kriging (or more modern deep-kernel Bayesian models) is valuable when surrogate evaluations are so cheap that thousands can be afforded, yet the physics is too nonlinear for level coupling to be effective.

These variance-reduction ideas will be combined with the VMS corrections of section 2.3 in Chapter 3 to yield an overall work complexity that scales essentially with the *coarse* solve, plus a few iteration of fine solves for bias elimination.

2.5 From Common Theory to Specialized Surrogates

The technical foundation established in Sections 2.1–2.4 is deliberately broad: it treats the high-parameter-complexity coefficient $q^{\mathrm{H}}(x,\xi)$ and its low-complexity surrogate $q^{\mathrm{L}}(x,\xi)$ inside a unified Sobolev framework, discretised by the standard finite-element machinery. Because each of the data-driven strategies developed later exploits the same weak form

$$a_\xi\big(u^{\mathrm{H}}(\xi),v\big)=\ell(v)\quad\forall v\in V,\qquad a_\xi(w,v):=\int_D q^{\mathrm{H}}\nabla w\cdot\nabla v\,\mathrm{d}x, \tag{2.10}$$

we pause to highlight the structural elements that recur throughout the remainder of the thesis.

Governing model and surrogate pair. The uniformly elliptic bound (2.3) guarantees that, for every realization of ξ, Problem (2.10) possesses a unique solution $u^{\mathrm{H}}(\cdot,\xi)\in V$. A severe truncation of the Karhunen–Loève expansion defines the low-complexity field q^{L} and an associated coarse solution $\widehat{u}(\cdot,\xi)$ via (2.5). We interpret the difference $u^{\mathrm{H}}-\widehat{u}$ as a *fine scale correction* that encapsulates all information lost by the KL truncation.

Finite-element backbone. On a common, shape-regular mesh the bilinear forms a_ξ and a_ξ^{L} give rise to stiffness matrices $\mathbf{A}^{\mathrm{H}}(\xi)$ and $\mathbf{A}^{\mathrm{L}}(\xi)$, respectively. Hence the correction satisfies $\mathbf{A}^{\mathrm{H}}(\xi)\big(u^{\mathrm{H}}-\widehat{u}\big)=\big(\mathbf{A}^{\mathrm{L}}(\xi)-\mathbf{A}^{\mathrm{H}}(\xi)\big)\widehat{u}$, revealing $\mathbf{A}^{\mathrm{H}}-\mathbf{A}^{\mathrm{L}}$ as the algebraic footprint of the discarded KL modes.

Variational multiscale interpretation. Section 2.3 recasts the foregoing algebraic relation in a parameter-oriented variational multiscale (VMS) split: the coarse component

$\widehat{u}$ solves a *restricted* problem with q^{L}, while the fine component is driven by the residual $r := f + \nabla\cdot\big(q^{\mathrm{L}}\nabla\widehat{u}\big)$. This observation both motivates and constrains the surrogate architectures that follow.

Multifidelity estimators. For practical uncertainty quantification one is rarely interested in the pathwise solution itself, but rather in statistics of a quantity of interest $Q(u^{\mathrm{H}})$. Section 2.4.1 showed that $\widehat{u}$ can act as an *effective control variate* inside a low-variance Monte-Carlo estimator, provided the discrepancy $Q(u^{\mathrm{H}}) - Q(\widehat{u})$ can be evaluated cheaply. Each of the forthcoming surrogates is designed precisely to supply such evaluations at mesh-level cost.

Before introducing the machine-learning framework, we first derive a *parameter complexity* coarse/fine decomposition that mirrors the mesh-based VMS splitting of Section 2.3. Throughout this chapter and as introduced previously, we reserve the superscripts 'H' and 'L' for the **H**igh parameter complexity and **L**ow parameter complexity coefficients defined in Section 2.1.

2.6 Low Fidelity and Correction Equations

Our reference model is the elliptic problem

$$\begin{cases} \mathcal{L}\,u = f & \text{in } D, \\ u = g & \text{on } \partial D, \end{cases} \qquad \mathcal{L} := -\nabla\cdot\big(q^{\mathrm{H}}(x,\xi)\,\nabla\cdot\big), \tag{2.11}$$

with q^{H} the **HPC** permeability field introduced in Chapter 1. In analogy with Section 2.1 we truncate the KL expansion after $N_{\mathrm{L}} \ll N_{\mathrm{H}}$ modes, obtaining the **LPC** coefficient $q^{\mathrm{L}}(x,\xi)$ and the corresponding coarse solution $u^{\mathrm{L}} := u\big(q^{\mathrm{L}}\big)$. Writing

$$q^{\mathrm{H}} \;=\; q^{\mathrm{L}} + q', \qquad u \;=\; u^{\mathrm{L}} + u',$$

where q' and u' are the high-frequency *parameter* and *solution* corrections, we insert the ansatz into (2.11). After elementary algebra, we arrive at the decoupled system

$$-\nabla\cdot\big(q^{\mathrm{L}}\nabla u^{\mathrm{L}}\big) = f \qquad \text{in } D, \qquad u^{\mathrm{L}} = g \qquad \text{on } \partial D, \tag{2.12a}$$

$$-\nabla\cdot\big(q^{\mathrm{H}}\nabla u'\big) = \nabla\cdot\big(q'\nabla u^{\mathrm{L}}\big) \qquad \text{in } D, \qquad u' = 0 \qquad \text{on } \partial D. \tag{2.12b}$$

Equation (2.12a) is inexpensive to solve on a coarse mesh, while (2.12b) retains *all* fine scale information. In what follows we learn a surrogate for u' *once*, offline, via a Conditional Variational Autoencoder (CVAE). Thereafter the surrogate supplies fine scale reconstructions in milliseconds.

Remarks 2.6.1.

(a) The operator term $\nabla\cdot\big(q^{\mathrm{H}}\nabla u'\big)$ is *independent of the boundary data*. Retraining is therefore unnecessary if g changes.

(b) Dependence on the forcing f is only through u^{L}, enabling generalisation to unseen loadings after sufficient pre-training.

(c) The training pairs (u^{L}, u') are generated by inexpensive KL sampling for q^{H} and coarse finite-element solves of (2.12a) and a resolved solution of 2.12b.

2.6.1 Hierarchical Parameter Enrichment

Let $\tilde{q}$ denote an intermediate hierarchy satisfying $q^{\mathrm{L}} \prec \tilde{q} \preceq q^{\mathrm{H}}$ and denote by $\tilde{u}$ the solution of $-\nabla\cdot\big(\tilde{q}\nabla\tilde{u}\big) = \nabla\cdot\big((q^{\mathrm{H}} - q^{\mathrm{L}})\nabla u^{\mathrm{L}}\big)$. The next theorem quantifies the error incurred by replacing q^{H} with $\tilde{q}$ in (2.12b).

Theorem 2.1 (Hierarchical approximation)**.** Let u' solve (2.12b) and $\tilde{u}$ be defined as above. Then

$$\|u' - \tilde{u}\|_{L^2(D)} \;\le\; C\,\frac{\|q^{\mathrm{H}} - \tilde{q}\|_{L^\infty(D)}}{\tilde{q}_{\min}}\,\|\nabla u'\|_{L^2(D)},$$

for a constant C depending only on D.

Proof. Plugging $q^{\mathrm{H}} \approx \tilde{q}$ and $u' \approx \tilde{u}$ into equation (2.12b), we obtain

$$-\nabla \cdot (\tilde{q} \nabla \tilde{u}) = \nabla \cdot (q' \nabla u^{\mathrm{L}}) \tag{2.13}$$

Subtract equation (2.13) from equation (2.12b) to get

$$-\nabla \cdot (q^{\mathrm{H}} \nabla u') + \nabla \cdot (\tilde{q} \nabla \tilde{u})) = 0.$$

Let $e = u' - \tilde{u}$. Then $\tilde{u} = u' - e$ and

$$\begin{aligned} 0 = -\nabla \cdot (q^{\mathrm{H}} \nabla u') + \nabla \cdot (\tilde{q} \nabla \tilde{u})) &= -\nabla \cdot (q^{\mathrm{H}} \nabla u') + \nabla \cdot (\tilde{q} \nabla (u' - e)) \\ &= -\nabla \cdot ((q^{\mathrm{H}} - \tilde{q}) \nabla u') + \nabla \cdot (\tilde{q} \nabla e). \end{aligned}$$

Thus,

$$-\nabla \cdot (\tilde{q} \nabla e) = \nabla \cdot ((q^{\mathrm{H}} - \tilde{q}) \nabla u')$$

Let $v \in V \subset H_0^1(D)$. Then

$$\begin{aligned} \int_D -\nabla \cdot (\tilde{q}(\nabla e) v dx &= \int_D -\nabla \cdot ((q^{\mathrm{H}} - \tilde{q}) \nabla u') v dx. \\ \int_D \tilde{q} \nabla v \cdot \nabla e \, dx - \int_D \tilde{q} v \nabla e \, dx &= \int_D (q^{\mathrm{H}} - \tilde{q}) \nabla v \cdot \nabla u' \, v \, dx \\ &\quad - \int_{\partial D} (q^{\mathrm{H}} - \tilde{q}) v \nabla u' \cdot \vec{n} \, dS, \end{aligned}$$

where the last equation follows from integration by parts. Since $v = 0$ on ∂D, the above equation reduces to

$$\int_D \tilde{q} \nabla v \cdot \nabla e \, dx = \int_D (q^{\mathrm{H}} - \tilde{q}) \nabla v \cdot \nabla u' dx \quad \text{for all } v \in V.$$

In particular, within the Galerkin framework, for $v = e$,

$$\int_D \tilde{q}\nabla e \cdot \nabla e \, dx = \int_D (q^{\mathrm{H}} - \tilde{q})\nabla e \cdot \nabla u' dx.$$
$$\int_D \tilde{q}\|\nabla e\|^2 dx = \int_D (q^{\mathrm{H}} - \tilde{q})\nabla e \cdot \nabla u' dx.$$

Thus, by boundedness of q^{H}, we compute as follows

$$\begin{aligned}
\tilde{q}_{min} \int_D \|\nabla e\|^2 dx &\leq \int_D \tilde{q}\|\nabla e\|^2 dx \\
&= \int_D (q^{\mathrm{H}} - \tilde{q})\nabla e \cdot \nabla u' \, dx \\
&\leq \|q^{\mathrm{H}} - \tilde{q}\|_\infty \int_D \|\nabla e\|\|\nabla \tilde{u}'\| dx \qquad (2.14) \\
&\leq \|q^{\mathrm{H}} - \tilde{q}\|_\infty \left[\int_D \|\nabla e\|^2 dx\right]^{\frac{1}{2}} \left[\int_D \|\nabla u'\|^2 dx\right]^{\frac{1}{2}}. \qquad (2.15)
\end{aligned}$$

Equations (2.14) and (2.15) both follow from the Cauchy-Schwartz inequalities for finite and infinite dimensions respectively. Now, dividing both sides of equation (2.15) by $\left[\int_D \|\nabla e\|^2 dx\right]^{\frac{1}{2}}$ and simplifying, we obtain

$$\left[\int_D \|\nabla e\|^2 dx\right]^{\frac{1}{2}} \leq \frac{\|q^{\mathrm{H}} - \tilde{q}\|_\infty}{\tilde{q}_{min}} \left[\int_D \|\nabla u'\|^2 dx\right]^{\frac{1}{2}}.$$

By Poincare inequality, we get the desired result

$$\|u' - \tilde{u}\| \leq C \frac{\|q^{\mathrm{H}} - \tilde{q}\|_\infty}{\tilde{q}_{min}} \|\nabla u'\|$$

for some constant, C. □

Corollary 2.6.1. If $\tilde{q} = q^{\mathrm{L}}$, then the following holds

$$\|u' - \tilde{u}\| \leq C \frac{\|q'\|_\infty}{q^{\mathrm{L}}_{min}} \|\nabla u'\| \qquad (2.16)$$

Proof. Since $q^{\mathrm{L}} = q^{\mathrm{H}} - q'$, setting $\tilde{q} = q^{\mathrm{L}}$ in Theorem 2.1, we obtain the desired result. □

The rest of the first part of the dissertation is dedicated to developing a method for recovering u' and the tools necessary for its implementation. In the next chapter, we propose a CVAE-Based HPC Field Reconstruction and present its development in detail.

Chapter 3

CVAE-Based HPC Field Reconstruction

3.1 Introduction

We seek to generate samples from the high parameter complexity output, given samples of the low-complexity model. Variational autoencoders are special neural networks that allow for a data-driven approach to generate high fidelity samples. These models are trained on samples of high-dimensional input vectors, which are reduced to latent variables, which in turn are used to reconstruct the high-dimensional distribution. Conditional variational autoencoders can make use of additional inputs (in our case samples of the low complexity solutions), to condition the reconstruction of the high-parameter solution.

In this chapter, we will briefly describe how conditional autoencoders work in the context of our problem, qualitative features of this approach (computational cost), and will conduct a few numerical experiments to demonstrate the effectiveness of this method.

3.2 Conditional Variational Autoencoders

In this subsection, we attempt to give a general introduction to the basic building blocks of CVAE. CVAE is a variant of VAE targeted at certain applications. VAE is a probabilistic generative version of Auto-Encoders. Auto-Encoders in turn, are stack of encoder and decoder layers of neural networks. We assume familiarity with neural networks and what make them universal function approximators, such as *backpropagation, gradient descent*, etc; see [85, 86, 15, 25].

3.2.1 Autoencoders

The concept of autoencoders has long been present in the annals of neural network research [33]. An *Autoencoder* is an unsupervised neural networks trained with the aim of reconstructing its inputs [5, 33]. Autoencoders have historically been employed for feature learning or dimensionality reduction. Autoencoders have recently gained prominence in generative modeling due to theoretical linkages between them and latent variable models. The training of autoencoders is a particular case of feedforward networks. It can be done using any of the same methods, notably mini-batch gradient descent after gradients are calculated via back-propagation [33].

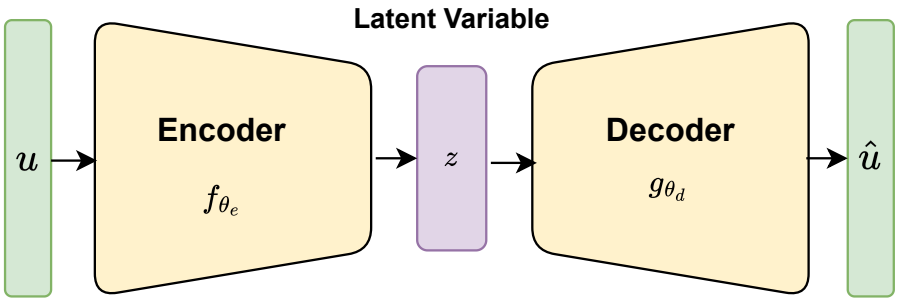


Figure 3.1: An Autoencoder Neural Networks Architecture

Figure 3.1 shows typical autoencoder architecture. It consist of two components: an encoder and a decoder. The encoder transforms a high-dimensional input into a low-dimensional representation, known as a latent or hidden representation. The decoder reconstructs the original input from the latent representation. The autoencoder is trained by minimizing the difference between the original and reconstructed inputs [33].

Without loss of generality, let the input $U = (u_1, \cdots, u_n)$ be a vector. Extensions of autoencoder to different data types exist, such as convolutional autoencoders of [59] for image data. We however, stick to the vanilla type architecture for the sake of illustrating the main idea. Let $f_{\theta_e} : \mathcal{U} \to \mathcal{Z}$ be the encoder function and $g_{\theta_d} : \mathcal{Z} \to \mathcal{U}$, the decoder where $\mathcal{U}, \mathcal{Z}$ are the input and latent spaces, respectively. The encoder and decoder deterministic

forms are

$$z = f_{\theta_e}(u) = \sigma(W_e u + b_e), \qquad g_{\theta_d}(z) = \sigma(W_d z + b_d)$$

with $\theta_e = (W_e, b_e)$, $\theta_d = (W_d, b_d)$ and σ, not necessarily same for both encoder and decoder, is an activation function. W_e, W_d are weight matrices and b_e, b_d are biases. The input units transmit their values to the output units through a hidden layer acting as a bottleneck with a restricted capacity that must optimally encode the input vectors when utilizing this sort of network to achieve dimension reduction by auto-association [8]. Finding the best weight matrices and bias vectors to minimize mean-square error, which is a go-to optimization criterion for continuous MLP training, is the task at hand [8]. Formally, the goal is to solve the optimization problem

$$\underset{(\theta_e,\theta_d)\in\Theta}{\text{minimize}}\ E = \frac{1}{N}\sum_{i=1}^{N} \|u_i - \hat{u}_i\|^2. \tag{3.1}$$

where $\hat{u} = g_{\theta_d}(z)$. Other popular loss functions for regression problems include *mean-square logarithmic loss, mean-absolute loss.* In a classification setting, popular loss functions are *cross-entropy loss, Kullback Leibler divergence loss, Brier score, etc.* The choice of loss function to use is purely problem dependent. However, the optimization setup remains the same. The study and creation of loss functions is an active area of research and designing loss functions for deep learning techniques has emerged in recent years as one of the most challenging issues [77].

As an instance of feedforward neural networks, autoencoders can be trained using the same methods as feedforward networks, generally using minibatch gradient descent with gradients update through back-propagation. Other training methods exist, such as *recirculation* but are rarely used in applications pertaining to machine learning [33].

Since its inception, many variants of the autoencoder architecture have been developed, including *sparse, denoising, contractive, variational, convolutional* autoencoders [21]. For

simplicity of presentation, we focus our energy on variational autoencoders from which conditional variational autoencoders is born. For details on other types of autoencoders, see [21, 33].

3.2.2 Variational Autoencoders (VAE)

VAE was developed to efficiently carry out inference and learning in directed probabilistic models with continuous latent variables, when the posterior distributions are intractable and the datasets are large [47]. Built on neural networks, the architecture is similar to autoencoders in addition to being a probabilistic model. It belongs to the class of *generative models*.

VAE is a deep Bayesian network that models the probabilistic relationship between a latent variable Z and a variable of interest U. A prior is typically chosen for Z, such as the multivariate unit Gaussian distribution $\mathcal{N}(0, I)$ [88]. The encoder $p_\theta(z|u)$, a conditional distribution is trained to push forward the measure on the space of the input U onto the chosen prior, space of Z. The decoder, $p_\gamma(u|z)$ in turn reconstructs U from the distribution of the prior. We assume each density is parametrized appropriately by weights and biases of neural networks and drop the θ, γ notations. In the Bayesian setting,

$$p(z|u) = \frac{p(u,z)}{p(u)}. \tag{3.2}$$

The marginal $p(u) = \int p(u|z)p(z)dz$ is intractable. It involves iterated integrals in multiple dimensions. Consequently, $p(z\big|u)$ is not tractable by directly using (3.2). Moreover, if we knew the distribution of U, our problem is fully solved. A strategy that works is to approximate $p(z\big|u)$ with $q(z\big|u)$ so as to solve the optimization problem

$$\underset{(\theta,\gamma)\in\Theta}{\text{minimize}}\ D_{KL}\left(q_\gamma(z|u)\big|\big|p_\theta(z|u)\right) \tag{3.3}$$

where $D_{KL}(\cdot||\cdot)$ is the *Kullback-Leibler Divergence*. By definition and (3.2),

$$\begin{aligned} D_{KL}\left(q(z|u)||p(z|u)\right) &= \mathbb{E}_q\left[\log\frac{q(z|u)}{p(z|u)}\right] \\ &= \mathbb{E}_q\left[\log q(z|u)\right] - \mathbb{E}_q\left[\log p(z,u)\right] + \log p(u) \\ &= -\left(\mathbb{E}_q\left[\log p(z,u)\right] - \mathbb{E}_q\left[\log q(z|u)\right]\right) + \log p(u) \end{aligned} \tag{3.4}$$

But by importance sampling, Jensen's inequality, and for any distribution q,

$$\begin{aligned} \log p(u) &= \log\int p(z,u)dz \\ &= \log\int p(z,u)\frac{q(z|u)}{q(z|u)}dz \\ &= \log\mathbb{E}_q[\frac{p(z,u)}{q(z|u)}] \\ &\geq \mathbb{E}_q\left[\log\frac{p(z,u)}{q(z|u)}\right] \\ &= \mathbb{E}_q\left[\log p(z,u)\right] - \mathbb{E}_q\left[\log q(z|u)\right] \end{aligned} \tag{3.5}$$

Clearly, if there exist q such that equality holds in (3.5), then such q minimizes the KL divergence in (3.4). Rearranging terms in (3.4) and using the relation $p(z,u) = p(u|z)p(z)$,

$$\begin{aligned} p(u) - D_{KL}\left(q(z|u)||p(z|u)\right) &= \left(\mathbb{E}_q\left[\log p(z,u)\right] - \mathbb{E}_q\left[\log q(z|u)\right]\right) \\ &= \mathbb{E}_q[\log p(u|z)] - \left(\mathbb{E}_q[\log q(z|u) - \log p(z)]\right) \\ &= \underbrace{\mathbb{E}_q[\log p(u|z)] - D_{KL}\left(q(z|u)||p(z)\right)}_{ELBO=\ell(\theta,\beta,\gamma)} \end{aligned} \tag{3.6}$$

where θ, β and γ are learnable parameters of the posterior $p(u|z)$, latent distribution $q(z|u)$, and the prior $p(z)$ respectively.

Known as the *variational lower bound* in [47] and recently as the *evidence lower bound* (*ELBO*), equation (3.6) which contains the *ELBO* forms the basis of the variational autoencoder [20]. The terms in the left hand side capture both the initial problem of computing

$p(u)$ in (3.2) and the alternate optimization problem in (3.4). Of great importance is the fact that the right hand side (the *ELBO*) is independent of $p(u)$ and $p(z|u)$. It enables the use of optimization methods such as *stochastic gradient descent* to maximize the left hand side, in search for a q that solves problem (3.4). Essentially, we combined the two problems in (3.2) and (3.4) into the single problem

$$\underset{q\in\mathcal{P}}{\text{maximize}}\; p(u) - D_{KL}\left(q(z|u)\middle|\middle|p(z|u)\right), \tag{3.7}$$

where $\mathcal{P}$ is a set of probability distributions. The new optimization problem is equivalent to the preceding problems in two ways: (1) while we cannot compute $p(u)$ explicitly to find $p(z|u)$, it makes sense to maximize it, and (2) the second term of the objective function is the dual problem of (3.4). Furthermore, by virtue of the equality in (3.6), we simply need to optimize the *ELBO* to solve our new, and consequently, old problems. For specific derivation such as when the prior is Gaussian, as well as the re-parametrization trick, see [47, 20].

Once the encoder and decoder parameters are optimized viz-a-viz minimization of the *ELBO*, VAE generates new samples similar to the training data by sampling from the prior and pushing forward the measure to the posterior of X via the decoder only.

3.2.3 Conditional Variational Autoencoders

Conditional variational autoencoders (CVAE) is similar to variational autoencoders (VAE), except that it makes assumptions about a label or categorization as a second aspect of the data [73]. The CVAE ELBO is the consequence of this, and is given as

$$\ell(\theta,\beta,\gamma|l) = \mathbb{E}_q[\log p(u|z,l)] - D_{KL}\left(q(z|u,l)\middle|\middle|p(z)\right). \tag{3.8}$$

It makes the recognition model and true posterior dependent on both X and the label, l [73, 36]. Although initially proposed for classification problems, CVAE has found applications in crowd counting via attention mechanism [14], multimodal image-to-image traslation [91], flow reconstruction from limited observations [36], among others.

Using CVAE, we learn the fine scale solutions of differential equations with rough parameters conditioned on the coarse scale realizations. Solutions with high resolution are prohibitive to compute using methods such as finite differences, finite elements or finite volumes. Coarse solutions are cheap with finite elements, but are not rich enough to capture microscale heterogeneity posed by the roughness of parameters of the governing equations. Let x_c be a coarse scale realization from a finite element forward solver. Similar to the work in [36], to approximate the probability distribution $p(u|z, u_c)$ of the fine scale solution, we condition latent representation z on the coarse scale x_c. The ELBO stays the same as in equation (3.8), but now with a continuous conditioning rather than discrete class label.

Using a CVAE we infer the high parameter complexity (HPC) solution u^{H} of the stochastic PDE from its low parameter complexity (LPC) surrogate u^{L}. Computing u^{H} which resolves the full set of N_{H} KL modes is prohibitively expensive with classical finite–difference, finite element, or finite volume schemes. By contrast, the LPC solution, obtained by retaining only the leading $N_{\mathrm{L}} \ll N_{\mathrm{H}}$ modes, is inexpensive but fails to capture the sub-grid heterogeneity induced by the truncated high order modes. Let u^{L} denote such an LPC realization produced by the forward finite element solver. Adapting the semi-supervised strategy of [36], we model the conditional distribution $p(u^{\mathrm{H}}|z, u^{\mathrm{L}})$ by letting the latent variable z depend on u^{H}. The resulting evidence lower bound is still given by (3.8), the only change being that the discrete class label is replaced by the continuous LPC conditioning variable u^{L}.

3.2.4 CVAE Architecture and Loss

Figure 3.2 distills the conditional variational autoencoder we designed to turn a cheap LPC field u^{L} into a high fidelity reconstruction of the expensive HPC solution u^{H}.

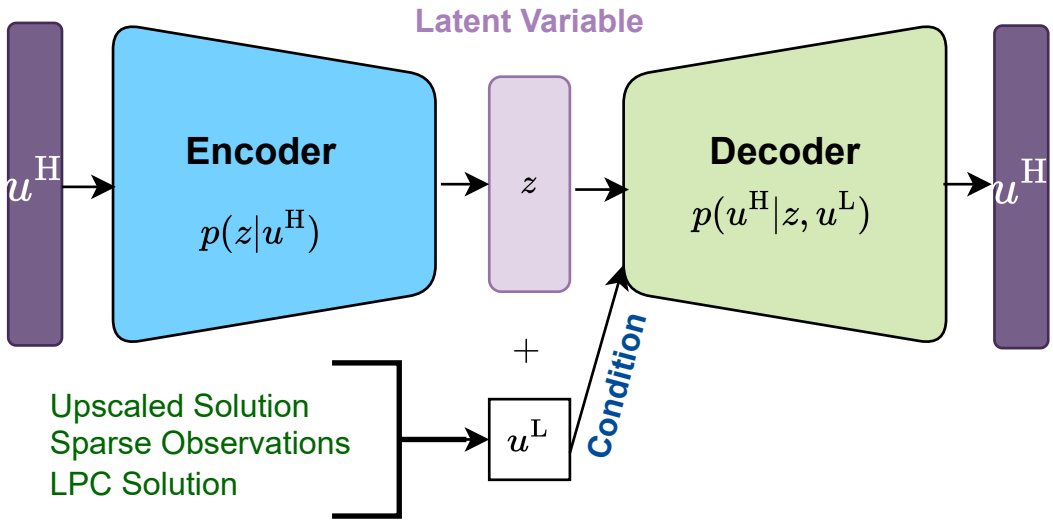


Figure 3.2: CVAE Architecture for Reconstructing HPC Solutions

Encoder $p_\theta(z \mid u^{\mathbf{H}})$. For one-dimensional benchmarks, u^{H} is merely a vector on a fine mesh, so a lightweight multilayer perceptron (MLP) with $\sim 10^4$ weights suffices to map it to the latent code $z \in \mathbb{R}^{d_z}$. In two and three dimensions the spatial correlations are richer; here we replace the MLP by a small CNN stack (conv-batchnorm-RELU blocks followed by global average pooling). Either way, the encoder outputs the parameters $\big(\mu_\theta, \log \sigma_\theta^2\big)$ of a diagonal Gaussian and samples z through the re-parameterisation trick [47].

Conditioning channel. Instead of feeding the LPC field through the encoder, we splice it in after the stochastic bottleneck (*late fusion*). We broadcast z to the resolution of u^{L} and concatenate the two tensors, so the decoder can blend global latent information with the physics-based coarse field.

Decoder $p_\phi(u^{\mathbf{H}} \mid z, u^{\mathbf{L}})$. The decoder is the mirror image of the encoder: an MLP for 1-D and a transpose-convolution stack for 2-D/3-D. Its final layer predicts the per-pixel/voxel mean μ_ϕ and log-variance $\log \sigma_\phi^2$ of a Gaussian likelihood.

Training objective. With mini-batch $\mathcal{B}$ and for $z \sim \mathcal{N}(0, I)$, the loss reads

$$\mathcal{L}_{\mathrm{CVAE}} = \frac{1}{|\mathcal{B}|} \sum_{(u^{\mathrm{H}}, u^{\mathrm{L}}) \in \mathcal{B}} \Big[D_{\mathrm{KL}}\big(p_\theta(z \mid u^{\mathrm{H}}) \,\|\, p(z)\big) - \mathbb{E}_{z \sim p_\theta}\big[\log p_\phi(u^{\mathrm{H}} \mid z, u^{\mathrm{L}})\big] \Big], \tag{3.9}$$

the usual ELBO specialized to our conditional setting. Section **??** explains how we add a weak PDE penalty to (3.9) so that reconstructions respect the governing operator.

3.2.5 Evidence Lower Bound with Physics Regularization

Training the network hinges on maximizing a *physics-augmented* evidence lower bound (ELBO). In its vanilla form the conditional variational autoencoder seeks the parameters θ, β that maximize

$$\mathcal{L}_{\text{ELBO}} \;=\; \underbrace{\mathbb{E}_{q_\beta(z\,|\,u^{\text{H}})}\big[\log p_\theta(u^{\text{H}}\,|\,z,u^{\text{L}})\big]}_{\text{reconstruction}} \;-\; \lambda_{\text{KL}}\;\underbrace{D_{\text{KL}}\big(q_\beta(z\,|\,u^{\text{H}})\;\|\;p(z)\big)}_{\text{latent prior}}, \tag{3.10}$$

where $z \sim \mathcal{N}(0, I)$ and λ_{KL} is the standard annealing coefficient [47]. In practice, however, a model that only sees data can exploit spurious correlations and drift away from the governing PDE when extrapolating. To tether the network to the underlying physics we append a residual penalty

$$\mathcal{R}_{\text{PDE}} := \;\Big\|-\nabla\cdot\big(q^{H}\big)\,\nabla u' - \nabla\cdot\big(q'\nabla u^{L}\big)\Big\|^2_{L^2(D)}, \tag{3.11}$$

evaluated on a Sobolev training grid or with automatic differentiation (e.g., `torch.autograd`) at Monte-Carlo points $x_j \sim \text{Unif}(D)$, and a boundary mismatch term

$$\mathcal{R}_{\partial D} := \;\big\|\,u'\big\|^2_{L^2(\partial D)}. \tag{3.12}$$

The complete objective thus reads

$$\mathcal{L} \;=\; -\,\mathcal{L}_{\text{ELBO}} \;+\; \lambda_{\text{PDE}}\,\mathcal{R}_{\text{PDE}+\,\lambda_{\partial D}\,\mathcal{R}_{\partial D}} \quad (\lambda_{\text{PDE}}, \lambda_{\partial D} > 0), \tag{3.13}$$

where the minus sign turns the maximization of the ELBO into a minimization task. For training, we adopt a two-stage curriculum: *(i)* warm-up with $\lambda_{\text{PDE}}{=}\lambda_{\partial D}{=}0$ for 30 epochs to

Algorithm 1 Offline generation of low-mode / high-mode solution pairs

Require: number of seeds N_{seed}; KL bandwidths (m_L, m_H) with $m_L \ll m_H$; FEM mesh $\mathcal{T}_h$ (N nodes); resamples per seed K; covariance kernel $C(|x-y|)$

Pre–compute the first m_H KL eigenpairs $\{(\lambda_k, \phi_k)\}_{k=1}^{m_H}$ on $\mathcal{T}_h$
for $i \leftarrow 1$ **to** N_{seed} **do**
    Draw $\boldsymbol{\xi}^{(i)} \sim \mathcal{N}(\mathbf{0}, I_{m_H})$
    Low-mode field $g^L(x) \leftarrow \sum_{k=1}^{m_L} \sqrt{\lambda_k}\, \phi_k(x)\, \xi_k^{(i)}$
    $q^L(x) \leftarrow \exp\big(g^L(x)\big)$
    $u^L \leftarrow \textsc{SolvePDE}\big(q^L\big)$
    for $j \leftarrow 1$ **to** K **do**
        Draw new $\widehat{\boldsymbol{\xi}}_{m_L+1:m_H}^{(i,j)} \sim \mathcal{N}(\mathbf{0}, I)$
        $g^H(x) \leftarrow g^L(x) + \sum_{k=m_L+1}^{m_H} \sqrt{\lambda_k}\, \phi_k(x)\, \widehat{\xi}_k^{(i,j)}$
        $q^H(x) \leftarrow \exp\big(g^H(x)\big)$
        $u^H \leftarrow \textsc{SolvePDE}\big(q^H\big)$
        Store $\big(u^L,\, u^H,\, q^H\big)^{(i,j)}$
    end for
end for

let the autoencoder learn a cohesive latent space; *(ii)* a max function schedule that gradually raises $\lambda_{\text{PDE}} \,/\, \lambda_{\partial D}$ to unity, following the ideas of PINNs [63] but without hand-tuned weights for each collocation point.

We retain the notation of Section 2.6. The superscripts $(\cdot)^{\text{L/H}}$ always signal low or high parameter complexity fields, while the operator in $\mathcal{R}_{\text{PDE}}$ uses the high fidelity coefficient q^{H}, the quantity the CVAE is meant to reconstruct.

3.2.6 Generation of LPC-HPC Training Pairs

A reliable CVAE hinges on an ample supply of paired snapshots $\big(u^{\text{L}}(\xi), u^{\text{H}}(\xi)\big)$ covering the admissible parameter range. We therefore construct the offline data set in four deterministic steps, Algorithm 1, each grounded in the mathematical formalism of Section 2.1.

Sampling strategy. To avoid the curse of dimensionality we draw the KL weights ξ_k independently from $\mathcal{N}(0,1)$ and discard modes once the eigenvalue ratio $\sum_{k>N_{\mathrm{H}}} \lambda_k / \sum_{k\geq 1} \lambda_k$ drops below 10^{-6}.

The resulting triplet $\mathcal{D} := \left\{(u^{\mathrm{L}}, u^{\mathrm{H}}, q^{\mathrm{L}})^{(i)}\right\}_{i=1}^{N_{\mathrm{tr}}}$ constitutes the sole offline input for CVAE train. No fine scale gradients or fluxes are required. The data acquisition cost aligns with standard Monte-Carlo workflows.

3.3 Numerical experiments

We demonstrate the effectiveness of our CVAE architecture on two representative problems. The first is a one dimensional elastic bar problem that has become a staple in uncertainty quantification studies [34, 48]. The spatially varying Young modulus is modeled as a lognormal Gaussian field; its first few Karhunen-Loève modes define the LPC (coarse) input, while the discarded high–index modes generate the HPC (fine) target that the network must reconstruct. Our second benchmark is a two–dimensional linear elliptic equation on the unit square with a heterogeneous lognormal diffusion coefficient, again split into LPC and HPC components via mode truncation. Taken together, the 1D and 2D cases test the surrogate in distinct low and high dimensional settings while keeping the underlying governing physics deliberately simple. The following subsections detail the data generation, training protocol, and quantitative results for both benchmarks.

3.3.1 One-dimensional Elastic Bar

We investigate the one-dimensional steady state diffusion equation

$$-\frac{\mathrm{d}}{\mathrm{d}x}\Big(q(x,\boldsymbol{\xi})\,u_x(x,\boldsymbol{\xi})\Big) = f(x), \qquad x\in(0,1), \qquad u(0,\boldsymbol{\xi}) = u(1,\boldsymbol{\xi}) = 0, \tag{3.14}$$

where q is a lognormal Gaussian random field. We expand $g = \log q$ in a Karhunen–Loève (KL) basis computed on a uniform mesh with 256 cells,

$$g(x, \boldsymbol{\xi}) = \sum_{k=1}^{N_H} \sqrt{\lambda_k}\, \phi_k(x)\, \xi_k, \qquad \xi_k \sim \mathcal{N}(0,1),$$

using a Matérn covariance kernel $C(x, y) = \sigma^2 \exp(-|x - y|/\ell)$ with $\sigma^2 = 10^{-3}$ and $\ell = 0.05$. Two resolution levels are distinguished. A low parameter coefficient is obtained by truncating the expansion after $m_L = 2$ modes, whereas the high parameter coefficient retains the first one hundred modes. The right–hand side is not fixed; instead we draw f from four families previously employed in the DeepGreen paper [32], namely isolated Gaussian bumps, single cosine waves, cubic monomials centered at $x = \pi$, and their quadratic-cubic superpositions. Each forcing is compiled on the fly as a FEniCS Expression.

3.3.2 Data generation

Given the huge amount of data involved, we produced training data lazily during optimization. For a randomly selected forcing f we first draw a low rank coefficient q^{L}, solve (3.14) to obtain u^{L}, and then generate one hundred conditional refinements q^{H} whose high-frequency KL coefficients are sampled conditionally on the retained low–frequency coefficients. Each high-rank coefficient is solved in turn, yielding the target pair $(u^{\mathrm{L}}, \mathrm{q}^{\mathrm{H}}, u^{\mathrm{H}})$. The procedure never stores samples on disk; it therefore consumes a constant amount of memory while potentially offering an unlimited stream of data.

3.3.3 Network architecture and training protocol

Both the encoder and decoder are fully connected neural networks with three hidden layers. The 128-dimensional latent space is sufficient to capture the variability observed in the high fidelity solutions. We optimize the usual CVAE objective augmented by a physics

term that penalizes the strong residual of (3.14). In particular the loss reads

$$\mathcal{L} = \underbrace{\|u^{\mathrm{H}} - \hat{u}^{\mathrm{H}}\|_2^2}_{\text{reconstruction}} + \beta\, D_{\mathrm{KL}}\big[q_\phi(z \mid u^{\mathrm{H}}) \,\|\, p_\theta(z)\big] + \gamma \left\| -(q^{\mathrm{H}}\hat{u}_x)_x - f \right\|_{L^2(0,1)}^2,$$

where β and γ are ramped linearly during the first 30 epochs to 0.1 and 0.01, respectively. We train for about 2000 epochs with a batch size of 32 and the Adam optimizer at a learning rate of 10^{-3}. One epoch corresponds to about 1000 - 5000 newly generated high fidelity samples, which are sufficient for the loss to stabilize.

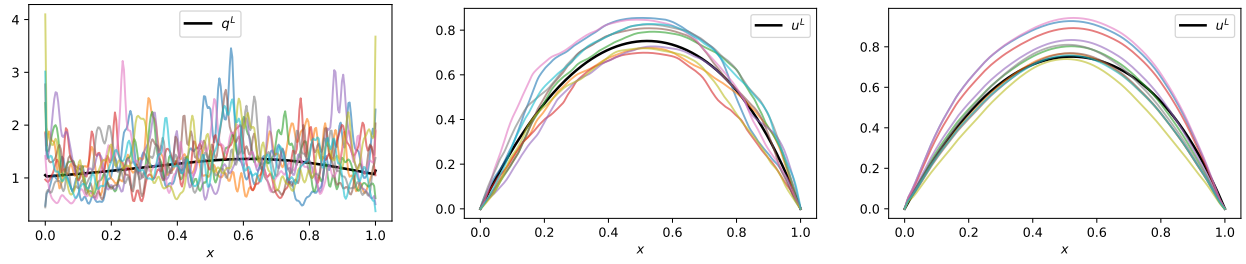
3.3.4 Test set and model evaluation

A separate Monte-Carlo archive is generated for the single forcing $f(x) = (x - \pi)^2$. It contains 100 low-rank seeds and, for each of them, 10000 conditional high-rank draws. Pointwise reconstruction accuracy is measured through the mean-square and mean absolute errors, while physical consistency is monitored by the residual norm appearing in the training loss. Finally, we quantify how well the CVAE matches the entire conditional distribution $p(u^{\mathrm{H}}|u^{\mathrm{L}})$ by computing a symmetric Kullback–Leibler divergence between histograms of the true and the generated solutions.

3.3.5 Results

All results reported below are obtained with a CVAE whose latent-space dimension is 128. For each conditioning field q^L we draw 10000 latent samples, decode them into HPC solutions $\hat{u}^H$, and compare these against the reference FEM solutions u^H. On the entire test data, the model attains a mean squared error of 1.1×10^{-4}, a mean absolute error of 8.9×10^{-3}, and an average strong residual of 3.5×10^{-3}, reflecting good physical consistency. The symmetric KL divergence between the empirical distributions of $\hat{u}^H$ and u^H, averaged over all 1000 low-rank seeds, is 3.4×10^{1}. The worst single seed reaches 7.6×10^{1}.

Qualitative behavior. Figure 3.3 visualizes one representative low-rank seed. Panel 3.3a shows the coarse random coefficient q^L (black) together with several conditional HPC realizations q^H (colored). Panel 3.3b overlays the corresponding LPC solution u^L with FEM samples of u^H, while Panel 3.3c replaces the FEM samples by CVAE predictions $\hat{u}^H$. The visual match between panels (b) and (c) confirms that the decoder reproduces the diversity of the HPC solution manifold.

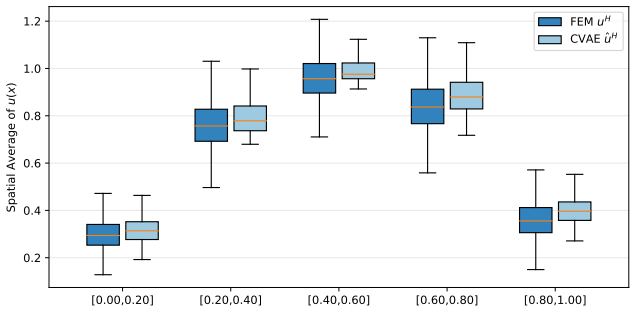


(a) q^H samples (b) FEM u^H samples (c) CVAE $\hat{u}^H$ samples

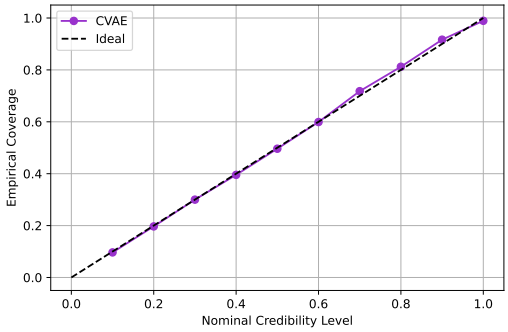
Figure 3.3: Visual comparison of CVAE and FEM solutions conditioned on one LPC solution. The right hand side is $f(x) = (x - \pi)^2$. The LPC field is generated with $m_L = 2$ KL modes, whereas the HPC field uses 100 modes.

Subinterval statistics. To probe spatial accuracy we partition the domain into five equal subintervals and compute the spatial average of u^H and $\hat{u}^H$ over each part. Figure 3.4 (a) presents paired boxplots for these quantities, built from 10 000 Monte-Carlo samples per subinterval. The near overlap of the boxes indicates that the decoder preserves both the central tendency and spread of the LPC conditioned fine scale averages.

Reliability of predicted uncertainty. Finally, we evaluate probabilistic calibration for the global quantity, namely the spatial average over the full domain. Figure 3.4 (b) shows the resulting reliability curve. Each point represents the empirical coverage of a symmetric prediction interval versus its nominal confidence level. The curve tracks the diagonal closely, with the mean absolute calibration error below 3%, demonstrating that the model's variance estimates are well calibrated.



(a) Paired box-plots of sub-interval spatial averages of u^H (dark blue) and $\hat{u}^H$ (light blue).



(b) Reliability diagram for the spatial average.

Figure 3.4: Statistical comparison of FEM and CVAE predictions.

Chapter 4

GeoAdaLer: Geometric Insights into Adaptive Stochastic Gradient Descent Algorithms

4.1 Introduction

Stochastic gradient descent (SGD) optimization methods [66, 70] play an important role in various scientific fields. When applied to machine learning algorithms, the objective is to adjust a set of parameters with the goal of optimizing an objective function. This optimization usually involves a series of iterative adjustments made to the parameters in each step as the algorithm progresses [46]. In the vanilla gradient descent approach, the magnitude of the gradient is the predominant annealing factor causing the algorithm to take larger steps when you are away from the optimum and smaller steps when closer to an optimum [68]. This method, however, becomes less effective near an optimal point, necessitating the selection of a smaller learning rate, which in turn affects the speed of convergence. The raw magnitude of the gradient does not always align with the optimal descent step size, thus necessitating a manually chosen learning rate. If the learning rate is too big, overshooting may occur and convergence rate is slow if the learning rate is too small. This challenge has led to the development of adaptive learning algorithms [61]. In this paper, we explore gradient descent algorithms that optimize the objective function with emphasis on the update rule. In this context, we focus on the update rule by breaking it into three components: the learning rate, the annealing factor, and the descent direction. This approach allows us to evaluate the effectiveness of an algorithm by examining the impact of its learning rate, annealing factor, and descent direction on the optimization process.

In this paper, we propose GeoAdaLer (short for Geometric Adaptive Learner), a new adaptive learning method for SGD optimization that is based on the geometric properties of the objective landscape. We use cosine of θ (where θ is the acute angle between the normal

to the tangent hyperplane and the horizontal hyperplane) as an annealing factor, which takes values close to zero when the optimization is traversing points close to an optimum and close to one for points far away from an optimum. Our method has surprising similarities to other adaptive learning methods.

Some of the advantages of GeoAdaLer is that it introduces a geometric approach for the annealing factor and outperforms standard SGD optimization methods due to the cosine of θ. Through both theoretical analysis and empirical observation, we identified similarities between our proposed method and other adaptive learning techniques. Our method enhances the understanding of existing algorithms and opens up more opportunities for geometric interpretability of how the algorithms traverse the objective manifold.

Additionally, we analyze the convergence of GeoAdaLer. We frame the optimization process as a fixed-point problem and split the analysis into deterministic and stochastic cases. Under the deterministic framework, we assume convexity and the existence of a finite optimal value. We utilize the Lipschitz continuity property of the gradient to establish upper bounds of convergence. We further employ the co-coercivity and quadratic upper bound properties to establish the lower bounds of convergence [82]. Under the stochastic framework, we employ the regret function, which measures the overall difference between our method and the known optimum point, ensuring that as time tends to infinity, the regret function over time tends to zero. By ensuring that the upper bound of the regret function goes to zero, we determine the overall convergence of the stochastic method to an optimum point. Both the deterministic and stochastic analyses demonstrate the robustness of GeoAdaLer and enhance our understanding of its practical applications.

4.2 Related Work

In the field of adaptive stochastic gradient descent algorithms, we continue to see improvements, often due to the need to address shortcomings of previous methods. The pioneering approach is AdaGrad, which focuses on the concept of per-parameter adaptive

learning rates. The foundation of AdaGrad is also the source of its limitation: the monotonic accumulation of squared gradients that could prematurely stifle learning rates [89, 22].

Subsequent optimizers like AdaDelta, RMSprop, and the popular Adam sought to address this issue [46, 78, 89]. AdaDelta introduced a decaying average of past squared gradients, while RMSprop utilized a similar exponential decay mechanism to limit the aggressive reduction in learning rates. Adam combined RMSprop's adaptive learning rates with the concept of momentum for smoother updates. However, Adam has been challenged by AMSGrad which modifies the algorithm in order to imbue it with "long term memory" which addresses issues with sub-optimal convergence under specific conditions and improves empirical performance [64]. As pointed out by [54, 71, 90], the assumptions on the hyperparameters of Adam were made before constructing the counter examples in [64]. Insights gained from these examples are invaluable for the ongoing exploration of stochastic gradient descent algorithms.

The literature on adaptive optimization algorithms for SGD reveals that each algorithm addresses the problems evident in its predecessors. The iterative approach, while successful in many ways, arguably emphasizes the lack of fundamental intuition regarding the dynamics of the AdaGrad family of optimizers.

GeoAdaLer is a novel adaptive learning method for SGD, employing the properties of the objective landscape. The innovative idea behind the GeoAdaLer approach is that the acute angle between the normal to the tangent plane at x and the horizontal plane conveys significant curvature-related information. This information could potentially recover the power of second order methods, which are not feasible in large-scale machine learning optimization. An understanding of this geometric approach to analyze adaptive methods for SGD can shed new light on the behavior of other algorithms in the AdaGrad family, potentially revealing the rationale for their strengths and weaknesses.

By discussing the geometric implications of adaptive step processes, we are able to potentially come up with optimizers that:

- **have more optimal annealing**: A geometric perspective that informs strategies to control learning rate decay more effectively, preventing premature convergence or extensively slow convergence.

- **provide parameter-sensitive adaptivity**: The geometry reveals how to tailor updates for individual parameters in a more principled manner.

- **increase robustness**: Understanding the geometric implications leads to optimizers that are less dependent on sensitive hyper-parameter tuning.

In essence, a geometric framework promises to move beyond the reactive development pattern, allowing us to proactively design adaptive optimizers that address the core issues in the AdaGrad family with greater intuition and foresight.

4.3 Mathematical Formulation

4.3.1 Deterministic Optimization

Consider the minimization of the convex objective functional $f : \mathbb{R}^n \to \mathbb{R}$ using the gradient descent algorithm. The vanilla gradient descent (GD) algorithm that maximizes the benefit of *gradient annealing* for smooth functions is

$$x_{t+1} = x_t + \delta x_t \tag{4.1}$$

where $\delta x_t = -\gamma g_t$ and g_t is $\nabla f(x_t)$, the gradient of the objective function at time t. This is the update step and is largely responsible for how far a step is taken in the descent direction. To see its real contribution, we decompose it further into

$$\delta x_t = -\gamma |g_t| \overline{g_t} \tag{4.2}$$

where γ is the learning rate which is responsible for manually scaling the step size, $|g_t|$ which is our annealing factor and lastly $-\overline{g_t}$ which gives us our optimal descent direction.

It is established that the annealing factor tends to be sub-optimal and can cause overshooting which we need to compensate for by applying a smaller learning rate, increasing convergence time [61]. To combat this issue, we propose an annealing factor based on the cosine of θ, where θ is the acute angle between the normal to the tangent hyperplane and the horizontal hyperplane. As shown in Figure 4.1, θ holds a vital information about the location of the current gradient step on the objective function. We harness this information using the cosine of θ and prove the following theorem.

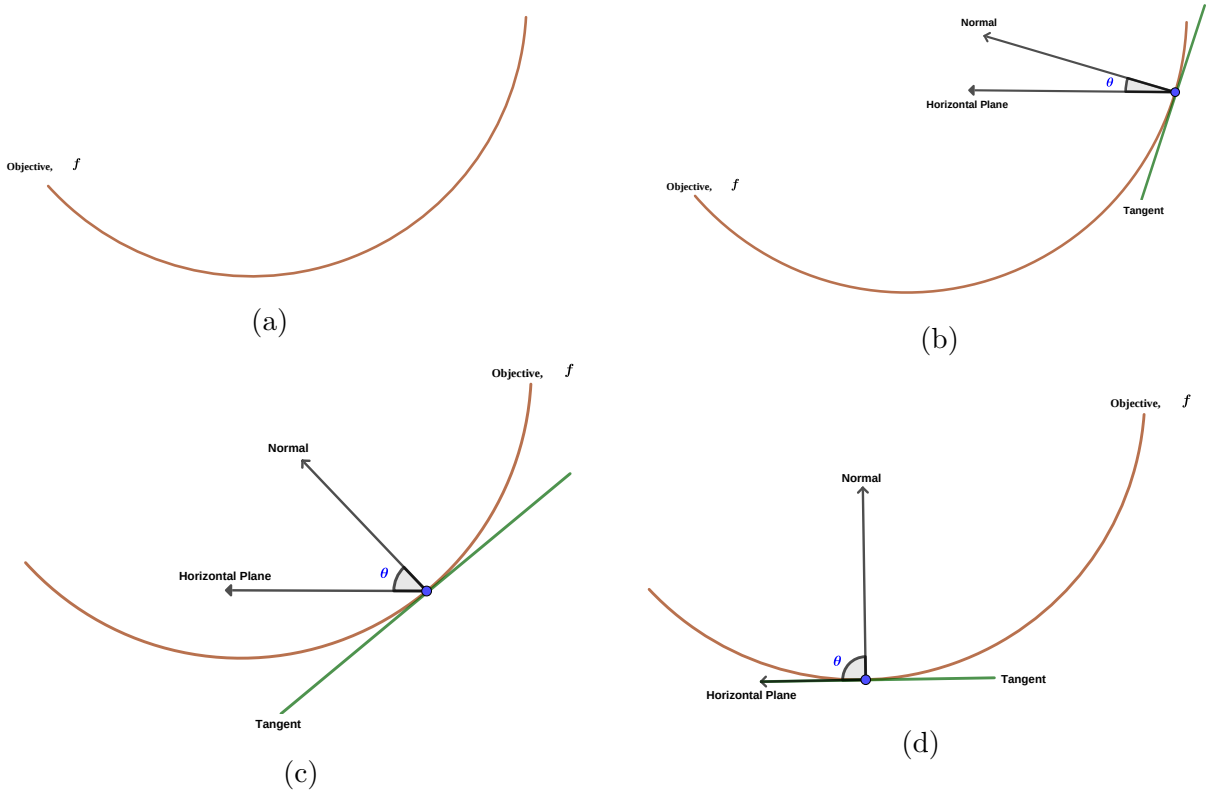


Figure 4.1: Geometric view of θ as the GeoAdaLer step traverses the objective function.

Theorem 4.1 (Geohess)**.** Let θ be the acute angle between the normal to an objective function $f : \mathbb{R}^n \to \mathbb{R}$ which is differentiable at x. Let $\|\cdot\|$ be the norm induced by the inner

product on $\mathbb{R}^n$. Then

$$\cos\theta = \frac{\|\nabla f(x)\|}{\sqrt{\|\nabla f(x)\|^2 + 1}} \tag{4.3}$$

We call Theorem 4.1 *Geohess* since this formulation mimics the curvature information traditionally found in the full hessian matrix.

Proof. Let x_i be an arbitrary point in $\mathbb{R}^n$. Then $[-\nabla f(x_i), 1]^T$ is orthogonal to $[x - x_i, y - f(x_i)]$ where (x, y) lies on the tangent hyperplane to f at x_i. i.e., $[-\nabla f(x_i), 1]^T$ is normal to $[x - x_i, y - f(x_i)]$. Let θ be the angle this normal makes with the horizontal hyperplane at the point $(x_i, f(x_i))$ in the descent direction $-\nabla f(x_i)$, i.e., a vector parallel to $[-\nabla f(x_i), 0]$. Then by vector calculus,

$$\cos\theta = \frac{[-\nabla f(x_i), 1]^T \cdot [-\nabla f(x_i), 0]^T}{\|[-\nabla f(x_i), 1]^T\| \|[-\nabla f(x_i), 0]^T\|} = \frac{\|\nabla f(x_i)\|}{\sqrt{\|\nabla f(x_i)\|^2 + 1}} \tag{4.4}$$

as required. □

The proposed GeoAdaLer update step is as follows:

$$\delta x_t = -\gamma\, \overline{g_t} \cos\theta = -\gamma \frac{\|g_t\|}{\sqrt{\|g_t\|^2 + 1}} \overline{g_t} \tag{4.5}$$

Equation (4.5) follows from the Geohess theorem (Theorem 4.1). In equation (4.5), if g_t is zero, then the update step is zero and algorithm has converged. If g_t is not zero, then equation (4.5) reduces to

$$\delta x_t = -\gamma \frac{1}{\sqrt{\|g_t\|^2 + 1}} g_t$$

4.3.2 Properties of GeoAdaLer Annealing Factor

The acute angle θ depends on g_t, and therefore, the annealing factor possesses the following properties:

1. $\dfrac{\|g_t\|}{\sqrt{\|g_t\|^2+1}} \to 1 \quad \text{as} \quad g_t \to \infty$ 2. $\dfrac{\|g_t\|}{\sqrt{\|g_t\|^2+1}} \to 0 \quad \text{as} \quad g_t \to 0$

In practice, we do not want $g_t \to \infty$ but as long as the magnitude of the gradient is large, the sufficiency in item 1 is guaranteed. Of utmost importance is item 2. In comparison to gradient annealing in SGD, we find that $\cos\theta$ possesses a logarithmic decay as a function of g_t as it tends to an optimum, resulting in a much more controllable annealing while standard SGD decays linearly.

The algorithm for GeoAdaLer is given in Algorithm 2. Note: The deterministic setting is achieved by using the objective function over the entire dataset instead of the stochastic objective function, in which case we set $\beta = 0$.

4.3.3 Stochastic Optimization

Online learning and stochastic optimization are closely linked and can be essentially used interchangeably [22, 13]. In online learning, the learner iteratively predicts a point $x_t \in X \subseteq \mathbb{R}^n$, often representing a weight vector that assigns importance values to different features. The objective of the learner is to minimize regret compared to a fixed predictor x^* within the closed convex set $x_t \in X \subseteq \mathbb{R}^n$ across a sequence of functions $\{f_1, f_2, \cdots\}$. Geometrically, gradients are vectors and have directions. Under a suitable distribution, a convex hull of these gradients approximates the true gradient for stochastic optimizations.

A common and intuitive approach to approximating the expected gradient is by employing an exponential moving average (EMA). As is the practice in literature [37, 46] we use a momentum-like term to replace the instantaneous gradient, thereby mitigating stochastic

fluctuations and revealing underlying trends in the gradient values. That is,

$$m_{t+1} = \beta m_t + (1 - \beta) g_t. \tag{4.6}$$

where $\beta \in [0, 1)$. The term β is constructed as a weighted average of the historical gradients and the current gradient. This modification enhances our ability to approximate the true underlying gradient more effectively in time. It also has the ability to update and learn as new streams of data are observed.

Algorithm 2 presents a pseudo-code of the proposed GeoAdaLer learning method.

Algorithm 2 GeoAdaLer

Require: γ : Learning rate
Require: β : Exponential decay rate for weighted average
Require: f_t : Stochastic objective function
Require: x_0 : Initial parameter vector
  $t \leftarrow 0$
  while x_t not converged **do**
    $g_t \leftarrow \nabla f_t(x_t)$ (Get gradients w.r.t stochastic objective)
    if $t = 0$ **then**
      $m_t \leftarrow g_t$ (initial weighted average)
    else
      $m_t = \beta m_{t-1} + (1 - \beta) g_t$ (Updated weighted average)
    end if
    $x_{t+1} = x_t - \gamma \cdot m_t / (\sqrt{\|m_t\|^2 + 1})$ (Update parameters)
    $t \leftarrow t + 1$
  end while
  return x_t

GeoAdaMax

Due to stochasticity and varying frequencies of occurrence of certain model inputs, such as in deep neural networks, adaptive stochastic gradient descent methods sometimes encounter issues with non-increasing squared gradients. For instance, consider the convex

objective function presented in [64], defined as follows:

$$f_t(x) = \begin{cases} Cx, & \text{if } t \mod 3 = 1 \\ -x, & \text{otherwise,} \end{cases}$$

In such scenarios, the adaptive step employed by the optimizer can still function effectively as a form of annealing, but the vanilla adaptive step leads to a suboptimality.

To address this problem, an approach was developed by [64], which involves retaining the maximum of the normalizing denominator over iterations. This approach reduces to a self scaling SGD with momentum [64]. By implementing a similar idea, we observe related results for GeoAdaLer. We call this method GeoAdaMax, indicating the use of maximum of the variance term.

GeoAdaMax dynamically adjusts the denominator using the largest historical value of the EMA. When this denominator remains unchanged, the update scale reflects its historical maximum, preserving proportionality in all future updates. This mechanism fine-tunes step sizes while maintaining alignment with past gradient magnitudes. The summary of the algorithm is presented in Algorithm 3

Geometrically, this is akin to increasing the angle between the normal and the horizontal plane by using a vector different from the normal. This leads to relatively smaller step sizes than if the original angle were used. Theorem 4.2 shows that indeed, the acute angle θ is increased.

Theorem 4.2. Let θ be the acute angle between the normal and the horizontal hyperplanes at the current iteration and let $\hat{\theta}$ be the acute angle between the horizontal hyperplane and the normal that maximizes the norm up to the current iteration. Then

$$\theta \leq \hat{\theta}.$$

Proof. By Algorithm 2 and Theorem 4.1,

$$\cos\theta = \frac{\|m_t\|}{\sqrt{\|m_t\|^2 + 1}} \geq \frac{\|m_t\|}{\sqrt{\max_t \|m_t\|^2 + 1}} = \cos\hat{\theta}$$

Applying the inverse cosine function on both sides gives the desired result since $\cos^{-1}$ is monotone decreasing on $[0, 1]$. □

Algorithm 3 GeoAdaMax

Require: γ: Learning rate
Require: β: Exponential decay rate for weighted average
Require: f_t: Stochastic objective function
Require: x_0: Initial parameter vector
  $t \leftarrow 0$
  $u_t \leftarrow 0$
  while x_t not converged **do**
    $g_t \leftarrow \nabla f_t(x_t)$ (Get gradients w.r.t stochastic objective)
    if $t = 0$ **then**
      $m_t \leftarrow g_t$ (initial weighted average)
    else
      $m_t = \beta m_{t-1} + (1 - \beta) g_t$ (Update weighted average)
    end if
    $u_t = max(\|m_t\|^2 + 1, u_{t-1})$
    $x_{t+1} = x_t - \gamma \cdot m_t / (\sqrt{u_t})$ (Update parameters)
    $t \leftarrow t + 1$
  end while
  return x_t

4.4 Relationship to the AdaGrad Family

The AdaGrad family of methods adjusts learning rates based on the accumulation of past gradients. These methods dynamically adapt the step size during optimization to handle varying gradient magnitudes across dimensions. For a given time step t, the learning rate adjustment in these methods depends on the accumulated gradient information, which we represent as G_t. Below is a summary of how G_t is defined for the main methods in the AdaGrad family:

AdaGrad: For AdaGrad, G_t is given as

$$G_t = \sum_{i=1}^{t} g_i^2$$

where g_i is the gradient at step i. AdaGrad accumulates the squared gradients over time, leading to decreasing learning rates [22].

RMSProp: The G_t step in RMSProp involves exponential moving average and is given as

$$G_t = \beta G_{t-1} + (1 - \beta) g_t^2.$$

RMSProp introduces an exponential decay factor β, preventing the learning rate from decreasing too quickly by giving more weight to recent gradients [37].

AdaGrad and RMSProp modify the learning rate as follows:

$$x_{t+1} = x_t - \frac{\gamma}{\sqrt{G_t + \epsilon}} \nabla f_t(x_t),$$

where η is the global learning rate and ϵ is a small constant for numerical stability.

Adam: In the case of Adam, an adjustment is not only made to the squared gradients but also to the gradients where a bias-corrected moving average is applied to each:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$$
$$G_t = \beta_2 G_{t-1} + (1 - \beta_2) g_t^2$$

where m_t and G_t are the moving average of the gradients and the squared gradients respective, $\hat{m}_t = m_t/(1 - \beta_1^t)$ and $\hat{G}_t = G_t/(1 - \beta_2^t)$ are the bias-corrected moving average of gradients and squared gradients respectively. Adam combines the ideas of momentum and adaptive learning rates for smoother updates [46].

The Adam's update rule therefore is as follows:

$$x_{t+1} = x_t - \frac{\gamma}{\sqrt{\hat{G}_t} + \epsilon}\hat{m}_t,$$

4.4.1 GeoAdaLer in Comparison

The core of GeoAdaLer's update is based on the cosine of the angle θ between the normal to the gradient and the horizontal hyperplane. It effectively uses the geometry of the optimization landscape to inform its adaptivity. Hence, with an EMA update

$$m_t = \beta m_{t-1} + (1-\beta)g_t,$$

GeoAdaLer updates as:

$$x_{t+1} = x_t - \frac{\gamma}{\sqrt{\|m_t\|^2 + 1}}m_t.$$

As shown above, GeoAdaLer's update is similar to that of the AdaGrad family with some function of the squared gradient playing a big role in the optimization step. The main differences include:

(a) The use of norm-based scaler

(b) The stability term in GeoAdaLer is naturally derived from the choice of the reference plane, for example with the choice of the normal to the gradient, our scaler produces a stability term of 1.

(c) There is only one moving average which is used in the estimation of the gradient and the function of the squared gradients

(d) Directly comparing to Adam, we also agree that G_t is always bigger for Adam since by Jensen's inequality,

$$[\beta m_{t-1} + (1-\beta)g_t]^2 \leq \beta m_{t-1}^2 + (1-\beta)g_t^2,$$

where the LHS G_t is for GeoAdaLer and the RHS for Adam.

We remark that all of these differences stem from the geometric intuition behind GeoAdaLer, which we believe lends itself to better understanding of the optimization paths as well as interpreting optimal values. Also, as noted in [84], the use of norm-based adaptivity ensures GeoAdaLer is robust to its hyperparameters.

4.5 Convergence Analysis

4.5.1 Deterministic Setting

We analyze the convergence of GeoAdaLer, first for the deterministic case, and then for the stochastic setting. Our setup remains the same, namely:

$$\underset{x}{\text{minimize}}\, f(x) \tag{4.7}$$

using the new adaptive gradient descent (GeoAdaLer) method where $f : \mathbb{R}^n \to \mathbb{R}$ is an objective function. We recast the optimization problem (4.7) into a fixed point iteration. Let $T : \mathbb{R}^n \to \mathbb{R}^n$ be a nonlinear operator defined as:

$$Tx = \left(I - \gamma \frac{\nabla f}{\sqrt{\|\nabla f\|^2 + 1}} \right)(x) \tag{4.8}$$

where I is the identity map.

Theorem 4.3. Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuous and ∇f Lipschitz continuous with $\gamma \leq \frac{1}{L}$ where L is the Lipschitz constant for ∇f. Assume f attains an optimal value at $x^* = \arg\min_x f(x)$. Then T defined in (4.8) is a contraction map with contraction parameter $\alpha = \sqrt{1 + \gamma^2 L_G^2 - 2\gamma \frac{L_G^2}{L}} < 1$ where L_G is the Lipschitz constant for $\dfrac{\nabla f}{\sqrt{\|\nabla f\|^2 + 1}}$ and $\|\cdot\|$

is the Euclidean norm in $\mathbb{R}^n$. That is

$$\|Tx - Ty\| \leq \alpha \|x - y\|. \tag{4.9}$$

A critical component of Theorem 4.3 involves demonstrating that the mapping T is a contraction. This allows us to invoke the Banach Fixed Point Theorem, which asserts that any contraction mapping on a complete metric space possesses a unique fixed point [16, 69, 75, 35]. By iteratively applying the contraction map, starting from an initial point, we ensure convergence to this fixed point. The recursive iterations converge to the unique fixed point with a geometric rate of convergence α. This fixed point corresponds to the minimizer we are seeking [9]. We remark that Theorem 4.3 holds if we replace continuity of f with lower semicontinuity and gradient with subgradient.

4.5.2 Stochastic Setting

We examine the convergence properties of the GeoAdaLer optimizer within the framework of online learning, as originally introduced in [92]. This framework involves a sequence of convex cost functions $\{f_1, f_2, \ldots, f_T\}$, each of which becomes known only at its respective timestep. The objective at each step t is to estimate the parameter x_t and evaluate it using the newly revealed cost function f_t. Given the unpredictable nature of the sequence, we assess the performance of our algorithm by computing the regret. Regret is defined as the cumulative sum of the differences between the online predictions $f_t(x_t)$ and the optimal fixed parameter $f_t(x^*)$ within a feasible set X across all previous time-steps. Specifically, the regret is defined as follows [92, 46]:

$$R(T) = \sum_{t=1}^{T} \left(f_t(x_t) - f_t(x^*) \right) \tag{4.10}$$

where the optimal parameter x^* is determined by $x^* = \arg\min_{x \in X} \sum_{t=1}^{T} f_t(x)$. We demonstrate that GeoAdaLer achieves a regret bound of $O(\sqrt{T})$, with a detailed proof provided in

the appendix .3. This result aligns with the best known bounds for the general convex online learning problem. We carry over all notations from the deterministic setting. Assuming the learning rate γ_t is of order $O(t^{-1/2})$ and β_t is exponentially decaying with exponential constant λ very close to 1, we obtain the following regret bounds for online learning with GeoAdaLer algorithm.

Theorem 4.4. For all $x \in \mathbb{R}^n$, and $t \leq T$, assume the gradient norm $\|\nabla f_t(x)\| \leq G$. Let $\gamma_t = \frac{\gamma}{\sqrt{t}}$, $\beta_t = \beta\lambda^t$, $\lambda \in (0,1)$, and $\beta \in [0,1)$. For any $k \in \{1,\ldots,T\}$, the separation between any point x_k generated by GeoAdaLer and the minimizer of an offline objective computed after all data is known is bounded as $\|x_k - x^*\| \leq G$. Then, for any $T \geq 1$, GeoAdaLer Algorithm achieves the regret bound:

$$R(T) \leq \frac{D^2\sqrt{G^2+1}\sqrt{T} + G(2\sqrt{T}-1)}{2(1-\beta)} + \frac{DG\beta(1-\lambda^T)}{(1-\beta)(1-\lambda)} \tag{4.11}$$

From Theorem 4.4, we observe that the GeoAdaLer algorithm achieves a sublinear regret bound of $O(\sqrt{T})$ over T iterations, which is consistent with the results typically found in the literature for similar algorithms. Our proof, akin to the approach taken in the Adam algorithm [46], relies significantly on the decay of β_t to ensure convergence.

In contrast to other adaptive stochastic gradient descent methods that adapts the current employs norm based scaling. This approach ensures that each parameter benefits from the collective dynamics of all parameters at the current time while retaining historical information in the exponential moving average of the gradients used for the adaptation. We remark that the convergence analysis for GeoAdaMax follows trivially from Theorem 4.4.

Corollary 4.5.1. Assume that for any $x \in \mathbb{R}^n$, the function f_t is convex and satisfies the gradient bounds $\|\nabla f_t(x)\|_2 \leq G$. Also, assume that the distance between any parameter x_k generated by GeoAdaLer Algorithm and the minimizer of an offline objective computed after all data is known is bounded, namely $\|x_k - x^*\|_2 \leq D$ for any $k \in \{1,\cdots,T\}$. Then, GeoAdaLer achieves the following guarantee for all $T \geq 1$:

$$\limsup_{T\to\infty} \frac{R(T)}{T} \leq 0$$

The corollary is derived by dividing the result in Theorem 4.4 by T and applying the limit superior operation to both sides of the inequality 4.11. It is important to note that when $R(T)$ yields a negative value, it indicates a favorable performance of the iterates produced by the GeoAdaLer algorithm. Specifically, such results suggest that the algorithm's execution leads to an expected loss that is lower than that of the best possible offline algorithm, which has full foresight of all cost functions and selects for a single optimal vector as proposed in [92].

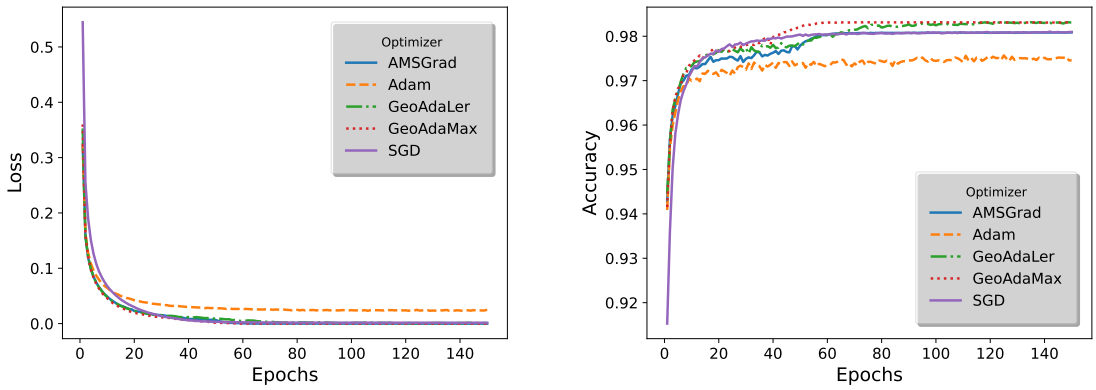
4.6 Experiments

We compare GeoAdaLer to other algorithms such as Adam, AMSGrad, and SGD with momentum. This comparison aims to assess how the geometric approach performs against these popular methods on the standard CIFAR-10 and MNIST datasets. Additionally, we examine how key hyperparameters influence the convergence of GeoAdaLer across different datasets. For all simulations, the GeoAdaLer learning rate $\gamma = 0.01$ and the exponential moving average parameter $\beta = 0.9$.

The hyperparameter settings for other optimizer algorithms are detailed as follows. These settings represent default values unless otherwise specified as recommended in the literature or considered standard for their respective packages. For SGD, the learning rate was set to 0.01, momentum to 0.9 and dampening to 0.9. For Adam & AMSGrad, the learning rate was set to 0.001, β_1 to 0.9 and β_2 to 0.99. All CPU calculations were performed on an AMD Ryzen 8-core CPU, and GPU calculations were conducted on a NVIDIA 3080ti.

4.6.1 MNIST Dataset

In the MNIST [52] experiment we trained a fully connected feed forward neural networks. It consists of three fully connected layers: the first layer takes in 784 input features (flattened 28x28 grayscale images) and outputs 128 features, the second layer reduces these to 64 features, and the final layer outputs 10 logits corresponding to class scores. Each of the first two layers is followed by a ReLU activation function. The final layer provides raw class scores. Using cross-entropy loss, GeoAdaLer and GeoAdaMax algorithms alongside the baseline optimizers were executed for 150 epochs and for 30 different weights initializations. Figure 4.2 and Table 4.1 illustrate the averaged results on the test dataset. GeoAdaLer demonstrates comparable initial performance to algorithms such as Adam and SGD, yet it achieves better long-run performance, converging to a higher validation accuracy. GeoAdaMax further improved this performance by faster convergence.



(a) Training loss (b) Validation Accuracy

Figure 4.2: MNIST

4.6.2 CIFAR-10 Dataset

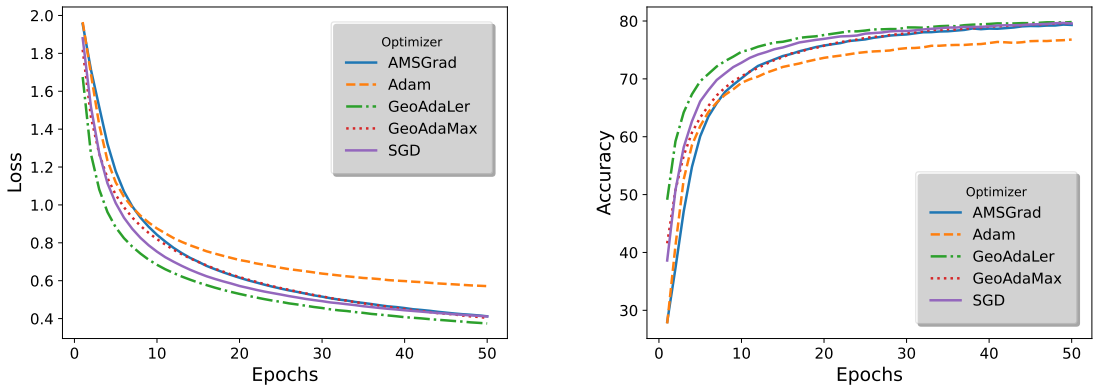
In our CIFAR-10 [50] experiments, we run GeoAdaLer, GoeAdaMax and the baseline optimizers for 50 epochs on a network consisting of six convolutional layers with 3x3 kernels and padding, progressively increasing in sizes of $32, 32, 64, 64, 128$ and 128 filters. Each

convolution is followed by a batch normalization layer and a ReLU activation. After every two convolutional layers, max-pooling with a 2×2 kernel is applied to reduce spatial dimensions, followed by dropout layers with rates $0.2, 0.3$ and 0.4. The output from the convolutional block, consisting of 128 filters with a 4x4 spatial size, is flattened and passed to two fully connected layers: the first reduces the feature size to 128 with a ReLU activation and a 0.5 dropout, and the second outputs 10 logits corresponding to class scores which are passed to a softmax function. The model is trained using cross-entropy loss and re-run 30 times for each optimizer with different initializations of the weights. The averaged results on test data are shown in Figure 4.3 and Table 4.2. GeoAdaLer shows early run performance comparable to baseline optimizers and occasionally exceeds them in validation accuracy. Its long run performance was only matched by Adam and GeoAdaMax. The consistent performance of GeoAdaLer in various runs reinforces the value of incorporating a geometric perspective into its design.

In the CIFAR-10 [50] experiment, we trained a convolutional neural network (CNN) model designed specifically for image data with RGB channels. The architecture consists of six convolutional layers, progressively increasing in feature map depth (32, 64, and 128 channels), followed by max-pooling and dropout layers to reduce overfitting and improve generalization. Each convolutional layer is followed by batch normalization to stabilize learning and expedite convergence [43]. Finally, two fully connected (linear) layers map the feature space to the 10 CIFAR-10 class scores, following common practices for classification in CNN architectures [51].

We train the model using cross-entropy loss for multi-class classification [33], and all adaptive gradient descent algorithms (including GeoAdaLer and GeoAdaMax) were executed for 100 epochs. To account for random initializations, we average the losses and accuracies over 30 different initializations. Model results are illustrated in Figure 4.3 and Table 4.2. GeoAdaLer demonstrates initial performance on par with other adaptive optimizers, such as Adam [46] and SGD [53], but achieved superior long-term accuracy, converging to higher

validation accuracy. GeoAdaMax provided further benefits, resulting in faster convergence and smoother training dynamics due to its stability near the optimal point.



(a) Training loss

(b) Validation Accuracy

Figure 4.3: CIFAR 10 Dataset

Table 4.1: MNIST Final Accuracy

Optimizer	Accuracy
GeoAdaLer	0.9831
GeoAdaMax	0.9831
Adam	0.9746
AMSGrad	0.9809
SGD	0.9810

Table 4.2: CIFAR Final Accuracy

Optimizer	Accuracy
GeoAdaLer	0.7982
GeoAdaMax	0.7962
Adam	0.7679
AMSGrad	0.7932
SGD	0.7957

Table 4.3: Fashion MNIST Final Accuracy

Optimizer	Accuracy
GeoAdaLer	0.9044
GeoAdaMax	0.9042
Adam	0.8838
AMSGrad	0.8993
SGD	0.8969

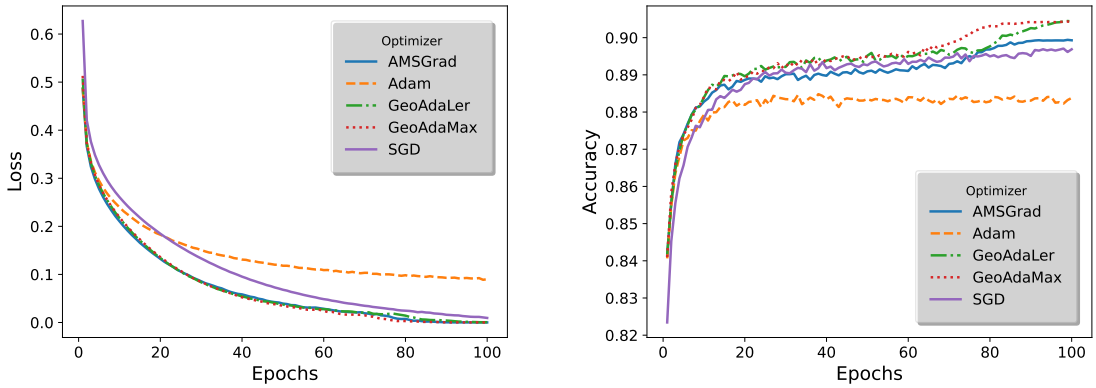
4.6.3 Fashion MNIST

In the Fashion MNIST [87] experiment we train a fully connected feed forward neural networks. It consists of three fully connected layers: the first layer takes in 784 input features (flattened 28×28 grayscale images) and outputs 512 features, the second layer reduces these to 256 features, and the final layer outputs 10 logits corresponding to class scores. Each of the first two layers is followed by a ReLU activation function. The final layer provides raw class scores. Using cross-entropy loss, GeoAdaLer and GeoAdaMax algorithms alongside the baseline optimizers were executed for 100 epochs and for 30 different weights initializations. Figure 4.4 and Table 4.3 illustrate the averaged results on the test dataset. GeoAdaLer once again shows comparable results to that demonstrated by the other benchmark algorithms with GeoAdaMax showing further improvement and faster convergence on the Fashion MNIST dataset.

4.6.4 Alternative Normal Plane Vectors

By introducing a hyper-parameter ϵ in the update rule, we can explore different vectors within the normal plane:

$$x_{t+1} = x_t - \frac{\gamma}{\sqrt{\|m_t\|^2 + \epsilon}} m_t.$$



(a) Training loss (b) Validation Accuracy

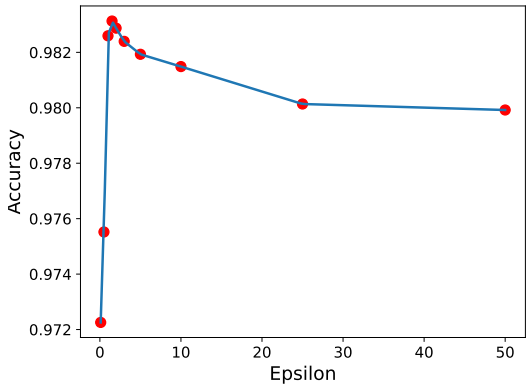
Figure 4.4: Fashion MNIST Dataset

This approach allows us to select vectors with varying angles relative to the horizontal plane. This investigation stems from observing how GeoAdaMax modifies the effective angles by maximizing the denominator of the update.

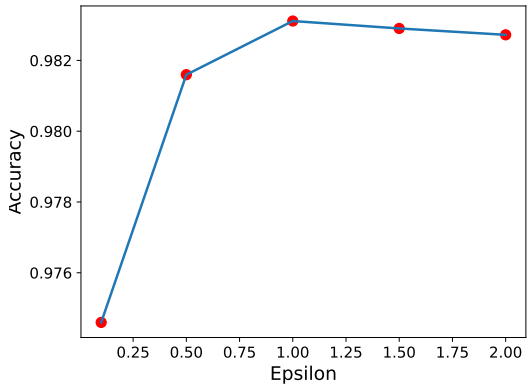
In this experiment, we compared GeoAdaLer and GeoAdaMax on the MNIST (4.5) and CIFAR-10 (4.6) datasets using 30 different weight initializations for selected values of ϵ, all other parameters were as mentioned above in their individual experiments. The mean accuracies for each ϵ value are plotted versus the value of ϵ. The results indicate that while the normal vector may not always be the most optimal choice, the optimal ϵ value tends to be close to the default associated with the normal vector. although some improvement do exist through the use of larger values of epsilon it is dependent on the data and not constant through all the experiments.

4.7 Conclusion

In this paper, we investigate the adaptive stochastic gradient descent algorithm and propose a geometric approach where the normal vector to the tangent hyperplane plays a crucial role in providing curvature-like information. We call this approach GeoAdaLer, and we show that it is derived from a fundamental understanding of optimization geometry.

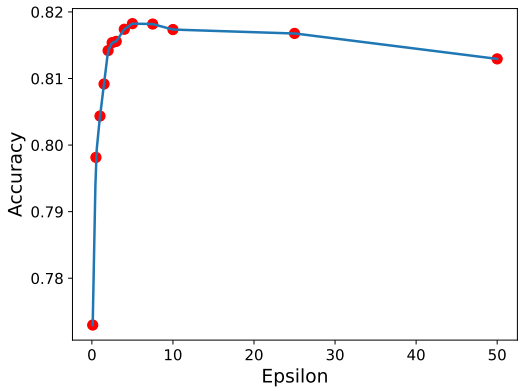


(a) GeoAdaLer Accuracy

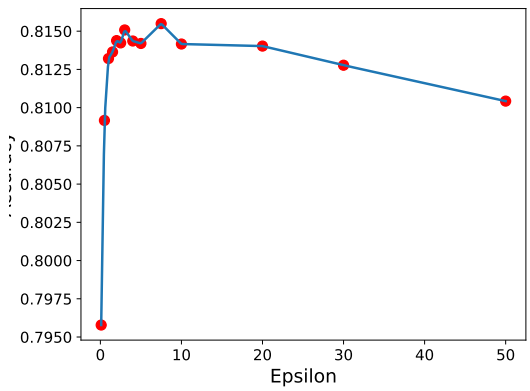


(b) GeoAdaMax Accuracy

Figure 4.5: MNIST Dataset



(a) GeoAdaLer Accuracy



(b) GeoAdaMax Accuracy

Figure 4.6: CIFAR 10 Dataset

We present theoretical proof for both deterministic and stochastic settings. Empirically, we find that GeoAdaLer is competitively comparable with other optimization techniques. Under certain conditions, it offers better performance and stability. GeoAdaLer provides a general geometric framework applicable to most, if not all, large-scale adaptive gradient-based optimization methods. We believe that this presents a significant step towards the development of interpretable machine learning algorithms through the lens of optimization.

Chapter 5

Summary and Future Work

5.1 Summary

In this work, we presented two complementary directions at the intersection of computational science, machine learning, and optimization. While each part addresses a distinct problem, both converge on the broader goal of enhancing computational tools for physical systems by leveraging structural and geometric insights.

In first part, we focused on the problem of reconstructing fine scale solutions from low fidelity models in hierarchical parameter systems. Such systems frequently arise in the modeling of multiscale physical phenomena, where computational cost limits the resolution or complexity of simulations. Our contribution here was to show that Conditional Variational Autoencoders (CVAE) can serve as effective surrogate mechanisms for learning the mapping between low complexity and high complexity parameter-induced solution fields. Our methods respects the physics and augments existing coarse simulations by learning statistically consistent corrections in function spaces. The approach was demonstrated on 1D benchmark problems and validated with respect to both classical norms and quantities of interest.

For the second part, we shifted focus to a foundational aspect of training adaptive learning algorithms, namely, understanding the geometry of stochastic optimization algorithms. Motivated by the geometric limitations of existing optimizers like AdaGrad and Adam, we introduced *GeoAdaLer*, a geometrically motivated adaptive learning algorithm that modulates learning rates by projecting gradients onto normal planes. While the second part is still in its early developmental stages and we did not pursue to unify the two parts in this work, we note that GeoAdaLer has the potential to contributes to a rigorous and interpretable framework for guiding learning in high dimensional spaces. GeoAdaLer was theoretically

analyzed in both deterministic and stochastic regimes, and empirically tested on standard benchmark datasets.

This dissertation thus represents a two-pronged contribution: one toward data efficient physical simulation and another toward robust optimization in machine learning.

5.2 Future Work

Several directions naturally emerge from this work. While CVAEs proved effective in reconstructing high fidelity solutions, alternative physics-informed or hybrid approaches remain to be explored. These include:

- Learning Green's functions or stiffness matrix representations directly from low fidelity fields.
- Using operator learning (e.g., DeepONets or Fourier Neural Operators) as alternative surrogates.
- Extending CVAEs to time-dependent or nonlinear PDEs where hierarchical complexity appears both spatially and temporally.

Another compelling future direction is to explore how geometric learning principles from GeoAdaLer can inform or enhance model training in multiscale settings. For example

- Can normal plane projections help regularize or stabilize the training of encoders/decoders in CVAEs?
- Can one design optimization schemes that exploit the physical manifold structure of PDE solution spaces?

Though empirical performance was promising, a deeper theoretical understanding of CVAE-based field reconstructions in PDE settings remains to be developed. Potential research could include

- Error estimates in energy norms.

- Guarantees on posterior consistency for conditioned fields.

Bibliography

[1] Robert J Adler. *The geometry of random fields*. SIAM, 2010.

[2] F. Alexander, M. Anitescu, J. Bell, D. Brown, M. Ferris, M. Luskin, S. Mehrotra, B. Moser, A. Pinar, A. Tartakovsky, K. Willcox, S. Wright, and V. Zavala. *A Multi-faceted Mathematical Approach for Complex Systems*. March 2012.

[3] R. G. Baca, R. C. Arnett, and D. W. Langford. Modelling fluid flow in fractured-porous rock masses by finite-element techniques. *International Journal for Numerical Methods in Fluids*, 4(4):337–348, April 1984.

[4] Krzysztof Banaś, Przemysław Płaszewski, and Paweł Macioł. Numerical integration on gpus for higher order finite elements. *Computers & Mathematics with Applications*, 67(6):1319–1344, 2014.

[5] Dor Bank, Noam Koenigstein, and Raja Giryes. Autoencoders. *arXiv preprint arXiv:2003.05991*, 2020.

[6] A. Bensoussan, J.-L. Lions, and G. Papanicolaou. *Asymptotic Analysis for Periodic Structures*. North-Holland, 1978.

[7] Sebastian Bock, Josef Goppold, and Martin Weiß. An improvement of the convergence proof of the adam-optimizer. *arXiv preprint arXiv:1804.10587*, 2018.

[8] H. Bourlard and Y. Kamp. Auto-association by multilayer perceptrons and singular value decomposition. *Biological Cybernetics*, 59(4–5):291–294, September 1988.

[9] S Boyd and EK Ryu. A primer on monotone operator methods (survey). *Appl. Comput. Math*, 15(1):3–43, 2016.

[10] Susanne C. Brenner and L. Ridgway Scott. *The Mathematical Theory of Finite Element Methods*, volume 15 of *Texts in Applied Mathematics*. Springer, New York, 3rd edition, 2008.

[11] H Brezis. Operateurs maximaux monotones et semi-groupes de contractions dans les espaces de hilbert. *Math. Studies*, 1973.

[12] Cris Cecka, Adrian J Lew, and Eric Darve. Assembly of finite element methods on graphics processors. *International journal for numerical methods in engineering*, 85(5):640–669, 2011.

[13] Nicolo Cesa-Bianchi, Alex Conconi, and Claudio Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

[14] Binghui Chen, Zhaoyi Yan, Ke Li, Pengyu Li, Biao Wang, Wangmeng Zuo, and Lei Zhang. Variational attention: Propagating domain-specific knowledge for multi-domain learning in crowd counting. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, page 16045–16055. IEEE, October 2021.

[15] Bing Cheng and D. M. Titterington. Neural networks: A review from a statistical perspective. *Statistical Science*, 9(1), February 1994.

[16] Charles Chidume. *Geometric Properties of Banach Spaces and Nonlinear Iterations*. Springer, 2009.

[17] Philippe G. Ciarlet. *The Finite Element Method for Elliptic Problems*. Society for Industrial and Applied Mathematics, January 2002.

[18] K. A. Cliffe, M. B. Giles, R. Scheichl, and A. L. Teckentrup. Multilevel monte carlo methods and applications to elliptic pdes with random coefficients. *Computing and Visualization in Science*, 14(1):3–15, January 2011.

[19] Nick C Dexter, Clayton G Webster, and Guannan Zhang. Explicit cost bounds of stochastic galerkin approximations for parameterized pdes with random coefficients. *Computers & Mathematics with Applications*, 71(11):2231–2256, 2016.

[20] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[21] Ganggang Dong, Guisheng Liao, Hongwei Liu, and Gangyao Kuang. A review of the autoencoder and its variants: A comparative perspective from target recognition in synthetic-aperture radar images. *IEEE Geoscience and Remote Sensing Magazine*, 6(3):44–68, September 2018.

[22] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[23] Y. Efendiev and T. Y. Hou. *Multiscale Finite Element Methods: Theory and Applications*. Springer, 2009.

[24] Yalchin Efendiev, Juan Galvis, Raytcho Lazarov, Svetozar Margenov, and Jun Ren. Robust two-level domain decomposition preconditioners for high-contrast anisotropic flows in multiscale media. *Computational Methods in Applied Mathematics*, 12(4):415–436, 2012.

[25] M. Egmont-Petersen, D. de Ridder, and H. Handels. Image processing with neural networks—a review. *Pattern Recognition*, 35(10):2279–2301, October 2002.

[26] Alexandre Ern and Jean-Luc Guermond. *Theory and Practice of Finite Elements*, volume 159 of *Applied Mathematical Sciences*. Springer, New York, 2004.

[27] Lawrence Evans. *Partial Differential Equations*. American Mathematical Society, March 2010.

[28] Leopoldo P. Franca, Alexandre L. Madureira, and Frederic Valentin. Towards multiscale functions: enriching finite element spaces with local but not bubble-like functions. *Computer Methods in Applied Mechanics and Engineering*, 194(27–29):3006–3021, July 2005.

[29] Shubin Fu and Eric T Chung. Constraint energy minimizing generalized multiscale finite element method for high-contrast linear elasticity problem. *arXiv preprint arXiv:1809.03726*, 2018.

[30] M. B. Giles. Multilevel monte carlo methods. *Acta Numerica*, 24:259–328, 2015.

[31] Michael B. Giles. Multilevel monte carlo path simulation. *Operations Research*, 56(3):607–617, June 2008.

[32] Craig R Gin, Daniel E Shea, Steven L Brunton, and J Nathan Kutz. DeepGreen: deep learning of Green's functions for nonlinear boundary value problems. *Scientific reports*, 11(1):21614, 2021. Publisher: Nature Publishing Group UK London.

[33] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[34] Recep M. Gorguluarslan and Seung-Kyum Choi. An improved stochastic upscaling method for multiscale engineering systems. In *Volume 2B: 40th Design Automation Conference*, IDETC-CIE2014. American Society of Mechanical Engineers, August 2014.

[35] Andrzej Granas and James Dugundji. *Fixed point theory*, volume 14. Springer, 2003.

[36] Kristian Gundersen, Anna Oleynik, Nello Blaser, and Guttorm Alendal. Semi-conditional variational auto-encoder for flow reconstruction and uncertainty quantification from limited observations. *Physics of Fluids*, 33(1), January 2021.

[37] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

[38] U. Hornung, editor. *Homogenization and Porous Media*. Springer, 1997.

[39] Ulrich Hornung, editor. *Homogenization and Porous Media*. Springer New York, 1997.

[40] Thomas Y Hou and Xiao-Hui Wu. A multiscale finite element method for elliptic problems in composite materials and porous media. *Journal of computational physics*, 134(1):169–189, 1997. Publisher: Elsevier.

[41] Thomas JR Hughes. Multiscale phenomena: Green's functions, the Dirichlet-to-Neumann formulation, subgrid scale models, bubbles and the origins of stabilized methods. *Computer methods in applied mechanics and engineering*, 127(1-4):387–401, 1995. Publisher: Elsevier.

[42] Thomas JR Hughes, Gonzalo R Feijóo, Luca Mazzei, and Jean-Baptiste Quincy. The variational multiscale method—a paradigm for computational mechanics. *Computer methods in applied mechanics and engineering*, 166(1-2):3–24, 1998. Publisher: Elsevier.

[43] Sergey Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[44] Norman P Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th annual international symposium on computer architecture*, pages 1–12, 2017.

[45] M. Kennedy. Predicting the output from a complex computer code when fast approximations are available. *Biometrika*, 87(1):1–13, March 2000.

[46] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[47] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[48] P.S. Koutsourelakis. Stochastic upscaling in solid mechanics: An excercise in machine learning. *Journal of Computational Physics*, 226(1):301–325, September 2007.

[49] Nikola B. Kovachki, Zongyi Li, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew M. Stuart, and Animashree Anandkumar. Neural operator: Learning maps between function spaces. *arXiv preprint arXiv:2108.08481*, 2021.

[50] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[51] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[52] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[53] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[54] Haochuan Li, Alexander Rakhlin, and Ali Jadbabaie. Convergence of adam under relaxed assumptions. *Advances in Neural Information Processing Systems*, 36, 2024.

[55] Peter C Lichtner and Satish Karra. Modeling multiscale-multiphase-multicomponent reactive flows in porous media: Application to CO2 sequestration and enhanced geothermal energy using PFLOTRAN. *Computational Models for CO2 Geo-sequestration*

& Compressed Air Energy Storage, pages 81–136, 2014. Publisher: CRC Press, http://www. crcnetbase. com/doi/pdfplus/10.1201/b16790-6.

[56] Gabriel J Lord, Catherine E Powell, and Tony Shardlow. *An introduction to computational stochastic PDEs*, volume 50. Cambridge University Press, 2014.

[57] Lu Lu, Pengzhan Jin, Guofei Pang, Zhiping Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3:218–229, 2021.

[58] Axel Målqvist and Daniel Peterseim. Localization of elliptic multiscale problems. *Mathematics of Computation*, 83(290):2583–2603, 2014.

[59] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. *Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction*, page 52–59. Springer Berlin Heidelberg, 2011.

[60] George J Minty. Monotone (nonlinear) operators in hilbert space. *Project Euclid*, 1962.

[61] Kamil Nar and Shankar Sastry. Step size matters in deep learning. *Advances in Neural Information Processing Systems*, 31, 2018.

[62] Benjamin Peherstorfer, Karen Willcox, and Max Gunzburger. Optimal model management for multifidelity monte carlo estimation. *SIAM Journal on Scientific Computing*, 38(5):A3163–A3194, January 2016.

[63] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378:686–707, 2019.

[64] Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. *arXiv preprint arXiv:1904.09237*, 2019.

[65] Michael Renardy and Robert C. Rogers. *An Introduction to Partial Differential Equations*, volume 13 of *Texts in Applied Mathematics*. Springer, New York, 2 edition, 2004.

[66] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

[67] R. Tyrrell Rockafellar. *Convex Analysis*. Princeton University Press, 1970.

[68] Sebastian Ruder. An overview of gradient descent optimization algorithms, June 2017. arXiv:1609.04747 [cs].

[69] W Rudin. Principles of mathematical analysis third edition mcgraw-hill, 1976.

[70] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[71] Naichen Shi and Dawei Li. Rmsprop converges with proper hyperparameter. In *International conference on learning representation*, 2021.

[72] Ralph C. Smith. *Uncertainty Quantification: Theory, Implementation, and Applications*. Society for Industrial and Applied Mathematics, January 2013.

[73] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

[74] P. D. Spanos and Roger Ghanem. Stochastic Finite Element Expansion for Random Media. *Journal of Engineering Mechanics*, 115(5):1035–1053, May 1989. Publisher: American Society of Civil Engineers (ASCE).

[75] Wilson A Sutherland. *Introduction to metric and topological spaces*. Oxford University Press, 2009.

[76] A. L. Teckentrup, R. Scheichl, M. B. Giles, and E. Ullmann. Further analysis of multilevel monte carlo methods for elliptic pdes with random coefficients. *Numerische Mathematik*, 125(3):569–600, March 2013.

[77] Yingjie Tian, Duo Su, Stanislao Lauria, and Xiaohui Liu. Recent advances on loss functions in deep learning for computer vision. *Neurocomputing*, 497:129–158, August 2022.

[78] T. Tieleman. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude, 2012.

[79] Andrea Toselli and Olof Widlund. *Domain Decomposition Methods—Algorithms and Theory*, volume 34 of *Springer Series in Computational Mathematics*. Springer, Berlin /Heidelberg, 2005.

[80] Rupak Tripathy and Ilias Bilionis. Deep uq: Learning deep generative models of uncertainty in high dimensions. In *Proceedings of the 35^{th} International Conference on Machine Learning*, 2018.

[81] Elisabeth Ullmann. A kronecker product preconditioner for stochastic galerkin finite element discretizations. *SIAM Journal on Scientific Computing*, 32(2):923–946, 2010.

[82] Lieven Vandenberghe. Optimization methods for large-scale systems lecture notes, 2022.

[83] Erik Vanmarcke. *Random fields: analysis and synthesis*. World Scientific, 2010.

[84] Rachel Ward, Xiaoxia Wu, and Leon Bottou. Adagrad stepsizes: Sharp convergence over nonconvex landscapes. *The Journal of Machine Learning Research*, 21(1):9047–9076, 2020.

[85] Paul Werbos. Beyond regression:" new tools for prediction and analysis in the behavioral sciences. *Ph. D. dissertation, Harvard University*, 1974.

[86] P.J. Werbos. Backpropagation through time: what it does and how to do it. *Proceedings of the IEEE*, 78(10):1550–1560, 1990.

[87] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[88] Haowen Xu, Yang Feng, Jie Chen, Zhaogang Wang, Honglin Qiao, Wenxiao Chen, Nengwen Zhao, Zeyan Li, Jiahao Bu, Zhihan Li, Ying Liu, Youjian Zhao, and Dan Pei. Unsupervised anomaly detection via variational auto-encoder for seasonal kpis in web applications. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web - WWW '18*, WWW '18, page 187–196. ACM Press, 2018.

[89] Matthew D Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

[90] Yushun Zhang, Congliang Chen, Naichen Shi, Ruoyu Sun, and Zhi-Quan Luo. Adam can converge without any modification on update rules. *Advances in neural information processing systems*, 35:28386–28399, 2022.

[91] Jun-Yan Zhu, Richard Zhang, Deepak Pathak, Trevor Darrell, Alexei A Efros, Oliver Wang, and Eli Shechtman. Toward multimodal image-to-image translation. *Advances in neural information processing systems*, 30, 2017.

[92] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 928–936, 2003.

.1 Deterministic Convergence Proof

Definition 1. *A differentiable function f on $\mathbb{R}^n$ is said to be co-coercive if*

$$\frac{1}{L}\|\nabla f(x) - \nabla f(y)\|^2 \leq \langle \nabla f(x) - \nabla f(y), x - y\rangle, \text{ for all } x, y \in \mathbb{R}^n. \tag{1}$$

Lemma 1. *Let ∇f be Lipschitz continuous with constant $L > 0$. Then $\frac{\nabla f}{\sqrt{\|\nabla f\|^2+1}}$ is Lipschitz continuous.*

Proof. Follows since $\frac{\nabla f}{\sqrt{\|\nabla f\|^2+1}}$ is a continuous function of ∇f. □

Lemma 2. *Suppose ∇f is Lipschitz continuous with parameter L, domain of f is $\mathbb{R}^n$ and f has a minimum at x^*. Then*

$$\frac{1}{2L}\|\nabla f(z)\|^2 \leq f(z) - f(x^*)$$

Proof. The proof relies on the following quadratic upper bound property

$$f(y) \leq f(z) + \langle \nabla f(z), y - z\rangle + \frac{L}{2}\|y - z\|^2 \quad \text{for all } y, z \in \text{ dom}(f) \tag{2}$$

where dom(f). the domain of f is a convex set. i.e., taking infimum on both sides of 2 gives

$$\begin{aligned} f(x^*) = \inf_y f(y) &\leq \inf_y (f(z) + \langle \nabla f(z), y - z\rangle + \frac{L}{2}\|y - z\|^2 \\ &= \inf_{\|v\|=1} \inf_t \left(f(z) + t\langle \nabla f(z), v\rangle + \frac{Lt^2}{2} \right) \\ &= \inf_{\|v\|=1} \left(f(z) - \frac{1}{2L}\langle \nabla f(z), v\rangle \right) \\ &= f(z) - \frac{1}{2L}\|\nabla f(z)\|^2 \end{aligned} \tag{3}$$

Rearranging the terms in 3 gives the desired result. □

Lemma 3 (Co-coercivity)**.** *Assume f is convex, proper and lower semicontinuous. Let ∇f be Lipschitz and let L_G be the Lipschitz constant for $\frac{\nabla f}{\sqrt{\|\nabla f\|^2+1}}$. Then the following co-coercivity property holds:*

$$\frac{1}{L_G}\left\|\frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2+1}}-\frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2+1}}\right\|^2 \leq \left\langle \frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2+1}}-\frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2+1}}, x-y\right\rangle \tag{4}$$

Proof. For any x and y, let

$$f_x(z) \coloneqq F(z) - \frac{\nabla f(x)^T z}{\sqrt{\|\nabla f(x)\|^2+1}}, \tag{5}$$

$$f_y(z) \coloneqq F(z) - \frac{\nabla f(y)^T z}{\sqrt{\|\nabla f(y)\|^2+1}} \tag{6}$$

for some constant a where $F(z) \coloneqq \int_a^z \frac{\nabla f(u)}{\sqrt{\|\nabla f(u)\|^2+1}} du$ is a scalar potential function for the vector field integrand. Notice that F is well defined since the integrand is a conservative vector field. Also, F(z) is convex since the integrand is a maximal monotone operator resulting from a convex, proper and lower semi-continuous function, f [60, 11, 67].

Thus, f_x and f_y are well defined and convex since the difference of a convex function and a linear function is convex. f_x is minimized at $z = x$, thus evaluating f_x at y and x and subtracting the results gives

$$\begin{aligned} F(y) - F(x) - \left\langle \frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2+1}}, y-x\right\rangle & \\ = f_x(y) - f_x(x) & \\ \geq \frac{1}{2L_G}\|\nabla f_x(y)\|^2 & \quad (7) \\ = \frac{1}{2L_G}\left\|\frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2+1}}-\frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2+1}}\right\|^2 & \quad (8) \end{aligned}$$

Equation 7 follows from Lemma 2 and equation 8 from taking the gradient of f_x at x.

Similarly, $z = y$ minimizes f_y and

$$\begin{aligned} F(x) - F(y) - \left\langle \frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}}, x - y \right\rangle \\ &= f_y(x) - f_y(y) \\ &\geq \frac{1}{2L_G} \|\nabla f_y(x)\|^2 \\ &= \frac{1}{2L_G} \left\| \frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}} - \frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2 + 1}} \right\|^2 \end{aligned} \tag{9}$$

Adding the inequalities 8 and 9, we obtain

$$\frac{1}{L_G} \left\| \frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2 + 1}} - \frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}} \right\|^2 \leq \left\langle \frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2 + 1}} - \frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}}, x - y \right\rangle.$$

□

Theorem .1. Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuous and ∇f Lipschitz continuous with $\gamma \leq \frac{1}{L}$ where L is the Lipschitz constant for ∇f. Assume f attains an optimal value at $x^* = \arg\min_x f(x)$. Then T defined in (4.8) is a contraction map with contraction parameter $\alpha = \sqrt{1 + \gamma^2 L_G^2 - 2\gamma \frac{L_G^2}{L}}$ where L_G is the Lipschitz constant for $\dfrac{\nabla f}{\sqrt{\|\nabla f\|^2 + 1}}$ and $\|\cdot\|$ is the Euclidean norm in $\mathbb{R}^n$.

Proof. It suffices to show that for some $\alpha \in [0, 1)$,

$$\|Tx - Ty\| \leq \alpha \|x - y\|. \tag{10}$$

To this end, we compute as follows

$$\begin{aligned}
\|Tx - Ty\|^2 &= \left\| x - \frac{\gamma \nabla f(x)}{\sqrt{\|\nabla f(x)\|^2 + 1}} - y + \frac{\gamma \nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}} \right\|^2 \\
&= \left\| (x - y) - \gamma \left(\frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2 + 1}} - \frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}} \right) \right\|^2 \\
&= \|x - y\|^2 - 2\gamma \left\langle x - y, \frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2 + 1}} - \frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}} \right\rangle \\
&+ \gamma^2 \left\| \frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2 + 1}} - \frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}} \right\|^2 \\
&\leq \left\| x - y \right\|^2 + \left(\gamma^2 - \frac{2\gamma}{L_G} \right) \left\| \frac{\nabla f(x)}{\sqrt{\|\nabla f(x)\|^2 + 1}} - \frac{\nabla f(y)}{\sqrt{\|\nabla f(y)\|^2 + 1}} \right\|^2 \qquad (11) \\
&\leq \|x - y\|^2 + L_G^2 \left(\gamma^2 - \frac{2\gamma}{L_G} \right) \|x - y\|^2 \qquad (12) \\
&\leq \left(1 + \gamma^2 L_G^2 - \frac{2\,\gamma L_G^2}{L} \right) \|x - y\|^2 \qquad (13)
\end{aligned}$$

Inequality (11) follows from co-coercivity (see Lemma 3) while 12 follows from the continuity of $\dfrac{\nabla f}{\sqrt{\|\nabla f\|^2 + 1}}$ and the Lipschitz continuity of ∇f. □

.2 Stochastic Convergence Proof

.2.1 Important Lemmas and Definitions

In this section, we provide proof of convergence of our algorithm. To this end, we start with some definitions and lemmas necessary for the main theorem.

Definition 2. *A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if for all $x, y \in \mathbb{R}$,*

$$f(y) \geq f(x) + \nabla f(x)^T (y - x). \qquad (14)$$

For the rest of the paper, we make the following assumptions on the stochastic objective function $f_t : \mathbb{R}^n \to \mathbb{R}$ where the iteration counter $t \geq 1$.

Assumptions 1. *1. f_t is convex for all t.*

2. *For all t, f_t is differentiable.*

3. *For all t, there exists $G \geq 0$ such that $\|\nabla f_t(x)\| \leq D$ for all $x \in X \subseteq \mathbb{R}^n$ where X is the feasible set.*

4. *$A \coloneqq \{x_1, x_2, \cdots\}$, the iterates generated by GeoAdaLer algorithm.*

5. *$x^* \coloneqq \arg\min_{x \in X} \sum_{t=1}^{T} f_t(x)$ exists and $\|x_k - x^*\| \leq D$ for all $x_k \in A$.*

6. *$\beta_t \coloneqq \beta\lambda^{t-1}$ where $\lambda \in (0,1), \beta \in [0,1)$.*

Also, for notational convenience, we take $g_t = \nabla f_t(x_t)$.

Lemma 4. *Under Assumptions 1, no. 3, the exponential moving average*

$$m_t = \beta_t m_{t-1} + (1 - \beta_t) g_t \tag{15}$$

is bounded for all t.

Proof. Iteratively expanding out m_t, we obtain

$$m_t = (1 - \beta_t) \sum_{i=1}^{t} \beta_t^{t-i+1} g_i.$$

Taking the Euclidean norm on both sides and using the boundedness of g_t gives

$$\begin{aligned}
\|m_t\| &= \|(1 - \beta_t) \sum_{i=1}^{t} \beta_t^{t-i+1} g_i\| \\
&\leq (1 - \beta_t) \sum_{i=1}^{t} \beta_t^{t-i+1} \|g_i\| \\
&\leq (1 - \beta_t) G \sum_{i=1}^{t} \beta_t^{t-i+1} \\
&= G(1 - \beta_t^t) \leq G
\end{aligned}$$

□

Lemma 5. *Under Assumptions 1, no. 6, the following inequalities hold*

1. $\frac{\beta_t}{1-\beta_t} \leq \frac{\beta}{1-\beta}$
2. $\frac{1}{1-\beta_t} \leq \frac{1}{1-\beta}$
3. $\sum_{t=1}^{T} \frac{\beta_t}{1-\beta_t} \leq \frac{\beta(1-\lambda^T)}{(1-\beta)(1-\lambda)}$

for all t.

Proof. 1). For all $t, \lambda < 1$ gives $\lambda^{t-1} < 1$ so that $\frac{1}{\lambda^{1-t}} < 1$. Thus

$$\frac{\beta_t}{1-\beta_t} = \frac{\beta\lambda^{t-1}}{1-\beta\lambda^{t-1}} = \frac{\beta}{\lambda^{1-t}-\beta} \leq \frac{\beta}{1-\beta}.$$

2). $\beta_t = \beta\lambda^{t-1} \leq \beta$ for all t since $\lambda \in (0,1), \beta \in [0,1)$. So, $1-\beta \leq 1-\beta\lambda^{t-1}$ leads to $\frac{1}{1-\beta_t} \leq \frac{1}{1-\beta}$ as required.

3). By Lemma 5 no. 2, we have

$$\sum_{t=1}^{T} \frac{\beta_t}{1-\beta_t} \leq \sum_{t=1}^{T} \frac{\beta_t}{1-\beta} = \frac{\beta}{1-\beta} \sum_{t=1}^{T} \lambda^{t-1} = \frac{\beta(1-\lambda^T)}{(1-\beta)(1-\lambda)}$$

□

Lemma 6. *For all $t \geq 1$ and for all $T \geq t$, the following inequality holds*

$$\sum_{t=1}^{T} \frac{1}{\sqrt{t}} \leq 1 + \int_1^T \frac{1}{\sqrt{t}} dt \tag{16}$$

Proof trivially follows from the integral test for convergence of series and is also given in [7].

Lemma 7. *Under Assumptions 1, no. 1 and 2, the following inequality holds for all t*

$$R(T) \leq \sum_{t=1}^{T} g_t \cdot (x_t - x^*). \tag{17}$$

Proof. By converxity of f_t for each t,

$$f_t(x^*) - f_t(x_t) \geq \nabla f_t(x_t) \cdot (x^* - x_t)$$

It then follows that

$$f_t(x_t) - f_t(x^*) \leq \nabla f_t(x_t) \cdot (x_t - x^*) = g_t \cdot (x_t - x^*)$$

Hence, summing both sides from $t = 1$ to T gives the required result. □

.3 Proof of Theorem 4.4

From the update rule in Algorithm 2

$$x_{t+1} = x_t - \gamma_t \frac{m_t}{\sqrt{\|m_t\|^2 + 1}}.$$

Subtracting x^* from both sides and taking norm squared results

$$\begin{aligned} \|x_{t+1} - x^*\|^2 &= |x_t - \gamma_t \frac{m_t}{\sqrt{\|m_t\|^2 + 1}}|^2 \\ &= \|x_t - x^*\|^2 - \frac{2\gamma_t}{\sqrt{\|m_t\|^2 + 1}} m_t \cdot (x_t - x^*) + \gamma_t^2 \frac{\|m_t\|^2}{\|m_t\|^2 + 1}. \end{aligned}$$

Substituting $m_t = \beta_t m_{t-1} + (1-\beta_t) g_t$, we obtain

$$\begin{aligned}\|x_{t+1} - x^*\|^2 = \|x_t - x^*\|^2 &- \frac{2\gamma_t}{\sqrt{\|m_t\|^2+1}} m_{t-1} \cdot (x_t - x^*) \\ &- \frac{2\gamma_t(1-\beta_t)}{\sqrt{\|g_t\|^2+1}} g_t \cdot (x_t - x^*) + \gamma_t^2 \frac{\|m_t\|^2}{\|m_t\|^2+1}.\end{aligned}$$

Rearrange to have $g_t \cdot (x_t - x^*)$ on the left hand side:

$$\begin{aligned} g_t \cdot (x_t - x^*) &= \frac{\sqrt{\|m_t\|^2+1}}{2\gamma_t(1-\beta_t)} \left[\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2\right] - \frac{\beta_t}{1-\beta_t} m_{t-1} \cdot (x_t - x^*) \\ &\qquad + \frac{\gamma_t}{2(1-\beta_t)} \frac{\|m_t\|^2}{\sqrt{\|m_t\|^2+1}} \\ &\leq \frac{\sqrt{\|m_t\|^2+1}}{2\gamma_t(1-\beta_t)} \left[\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2\right] \\ &\qquad + \frac{\beta_t}{1-\beta_t} \|m_{t-1}\| \|(x^* - x_t)\| + \frac{\gamma_t \|m_t\|^2}{2(1-\beta_t)}. \end{aligned} \tag{18}$$

where inequality 18 follows from the Cauchy-Schwartz inequality applied to the second term $-m_{t-1} \cdot (x_t - x^*) = m_{t-1} \cdot (x^* - x_t)$ and the fact that $\frac{\|m_t\|^2}{\sqrt{\|m_t\|^2+1}} \leq 1$ for all t. Summing both sides from $t = 1$ to T and using Lemma 4 and Assumptions 1 no. 3 and 5, we further obtain

$$\begin{aligned} \sum_{t=1}^{T} g_t \cdot (x_t - x^*) &\leq \sum_{t=1}^{T} \frac{\sqrt{G^2+1}}{2\gamma_t(1-\beta_t)} \left[\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2\right] \\ &\qquad + DG \sum_{t=1}^{T} \frac{\beta_t}{1-\beta_t} + G^2 \sum_{t=1}^{T} \frac{\gamma_t}{2(1-\beta_t)} \\ &\leq \sum_{t=1}^{T} \frac{\sqrt{G^2+1}}{2\gamma_t(1-\beta_t)} \left[\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2\right] \\ &\qquad + \frac{DG\beta(1-\lambda^T)}{(1-\beta)(1-\lambda)} + \frac{G^2}{2(1-\beta)} \sum_{t=1}^{T} \gamma_t. \end{aligned} \tag{19}$$

where the last inequality follows from Assumptions 1 no. 3 and Lemma 5 no. 2 and 3. By expanding the first term and and rewriting the what is left in compact form, we obtain

$$\begin{aligned}\sum_{t=1}^{T} \frac{1}{\gamma_t(1-\beta_t)} \left[\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2\right] &= \frac{1}{\gamma_1(1-\beta_1)}\|x_1 - x^*\|^2 - \frac{1}{\gamma_T(1-\beta_T)}\|x_{T+1} - x^*\|^2 \\ &\quad + \sum_{t=2}^{T} \left(\frac{1}{\gamma_t(1-\beta_t)} - \frac{1}{\gamma_{t-1}(1-\beta_{t-1})}\right) \|x_t - x^*\|^2 \\ &\leq \frac{\|x_1 - x^*\|^2}{\gamma_1(1-\beta_1)} \\ &\quad + \sum_{t=1}^{T} \left(\frac{1}{\gamma_t(1-\beta_t)} - \frac{1}{\gamma_{t-1}(1-\beta_{t-1})}\right) \|x_t - x^*\|^2.\end{aligned}$$

where the last inequality follows from dropping the second term. By the Assumptions 1 no. 5, it further simplifies to

$$\begin{aligned}\sum_{t=1}^{T} \frac{1}{\gamma_t(1-\beta_t)} \left[\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2\right] &\leq \frac{D^2}{\gamma_1(1-\beta_1)} + D^2 \sum_{t=2}^{T} \left(\frac{1}{\gamma_t(1-\beta_t)} - \frac{1}{\gamma_{t-1}(1-\beta_{t-1})}\right) \\ &= \frac{D^2}{\gamma_1(1-\beta_1)} - \frac{D^2}{\gamma_1(1-\beta_1)} + \frac{D^2}{\gamma_T(1-\beta_T)} \\ &= \frac{D^2}{\gamma_T(1-\beta_T)}.\end{aligned}$$

From Lemma 5 we have.

$$\sum_{t=1}^{T} \frac{1}{\gamma_t(1-\beta_t)} \left[\|x_t - x^*\|^2 - \|x_{t+1} - x^*\|^2\right] \leq \frac{D^2}{\gamma_T(1-\beta)}.$$

Therefore equation 19 become:

$$\sum_{t=1}^{T} g_t \cdot (x_t - x^*) \leq \frac{D^2\sqrt{G^2+1}}{\gamma_T(1-\beta_T)} + \frac{DG\beta(1-\lambda^T)}{(1-\beta)(1-\lambda)} + \frac{G^2}{2(1-\beta)} \sum_{t=1}^{T} \gamma_t. \tag{20}$$

Assuming $\gamma_t = \frac{1}{\sqrt{t}}$ we further obtain:

$$\sum_{t=1}^{T} g_t \cdot (x_t - x^*) \leq \frac{\sqrt{T} D^2 \sqrt{G^2 + 1}}{(1 - \beta_T)} + \frac{DG\beta(1 - \lambda^T)}{(1 - \beta)(1 - \lambda)} + \frac{G^2}{2(1 - \beta)} \sum_{t=1}^{T} \frac{1}{\sqrt{t}} \tag{21}$$

$$\leq \frac{\sqrt{T} D^2 \sqrt{G^2 + 1}}{(1 - \beta_T)} + \frac{DG\beta(1 - \lambda^T)}{(1 - \beta)(1 - \lambda)} + \frac{G^2(2T - 1)}{2(1 - \beta)}. \tag{22}$$

Equation 22 follows from Lemma 6 and so our regret is bounded above by:

$$R(T) \leq \frac{\sqrt{T} D^2 \sqrt{G^2 + 1}}{(1 - \beta_T)} + \frac{DG\beta(1 - \lambda^T)}{(1 - \beta)(1 - \lambda)} + \frac{G^2(2T - 1)}{2(1 - \beta)}.$$